

Yannick Müller muelleya@student.ethz.ch



MADE EASY

Computer Science
Basisprüfung - Block 1

January 2019

Version 1.4

Letzte Bearbeitung: 30. May 2019

Diese Zusammenfassung besteht aus den Vorlesungs- und Übungsstundennotizen. In Linearer Algebra und Diskreter Mathematik wurde versucht den Aufbau möglichst nah an das Skript anzulehnen. Die Kapitel und Satznummern sollten grösstenteils übereinstimmen. Jedoch sollten diese überprüft werden, bevor sie in die Zusammenfassung für die Prüfung genommen werden.

Für die Richtigkeit dieses Skriptes kann ich nicht garantieren. Viel Spass beim Lernen.

Table 1: Prüfungsplan

Datum	Zeit	Nummer	Fach	Hilfsmittel	Raum
Mo 21.01	09:00-12:00	252-0025-00 S	Diskrete Mathematik	6 A4 Seiten	HIL G41, G15, G61
Do 24.01	09:00-16:00	252-0027-00 S	Ein. Programmierung		ONA E25, E7
Mo 28.01	08:30-17:30	252-0026-00 S	D & A		ONA E25, E7
Fr 01.02	14:00-17:00	401-0131-00 S	Lineare Algebra	12 A4 Seiten	HIL G41, G15, G61

Table 2: Wochenplan

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
5:00							
6:00		Swimming					
7:00							
8:00							
9:00							
10:00				Yoga		Running	Running
11:00							
12:00							
13:00							
14:00	Yoga						
15:00							
16:00							
17:00							
18:00							
19:00			Triathlon				
20:00							
21:00							
22:00							

Ziele

1. Lernplan einhalten
 2. In Gruppen lernen
 3. Prüfung bestehen

“The best way to predict your future is to create it.”

—Abraham Lincoln

“Live as if you were to die tomorrow. Learn as if you were to live forever.”

—Gandhi

Table 3: Vorbereitungstests

Table 4: Study Time Plan

Saturday	22. December	Finish Exercises
Sunday	23. December	LaTeX Correction
Monday	24. December	LaTeX Correction
Tuesday	25. December	Create One Big LaTeX Summary
Wednesday	26. December	DiscMath Study Summary
Thursday	27. December	LinAlg Study Summary
Friday	28. December	DnA Study Summary
Saturday	29. December	EProg Study Summary
Sunday	30. December	DiscMath Test HS 16 with Summary
Monday	31. December	LinAlg Test HS 16 with Summary
Tuesday	01. January	DnA TestHS 16 with Summary
Wednesday	02. January	EProg Test HS 16 with Summary
Thursday	03. January	DiscMath Test FS 17
Friday	04. January	LinAlg Test FS 17
Saturday	05. January	DnA Test FS 17
Sunday	06. January	EProg Test FS 17
Monday	07. January	Write DiscMath Test Summary
Tuesday	08. January	DiscMath Test HS 17
Wednesday	09. January	Write LinAlg Test Summary
Thursday	10. January	LinAlg Test HS 17
Friday	11. January	DnA Test HS 17
Saturday	12. January	EProg Test HS 17
Sunday	13. January	Whatever Needs to be Done
Monday	14. January	DiscMath Test FS 18
Tuesday	15. January	LinAlg Test FS 18
Wednesday	16. January	DnA Test FS 18
Thursday	17. January	EProg Test FS 18
Friday	18. January	DiscMath Final Preparation
Saturday	19. January	DiscMath Final Preparation
Sunday	20. January	DiscMath Final Preparation
Monday	21. January	TEST DiscMath
Tuesday	22. January	EProg Final Preparation
Wednesday	23. January	EProg Final Preparation
Thursday	24. January	TEST EProg
Friday	25. January	DnA Final Preparation
Saturday	26. January	DnA Final Preparation
Sunday	27. January	DnA Final Preparation
Monday	28. January	TEST DnA
Tuesday	29. January	LinAlg Final Preparation
Wednesday	30. January	LinAlg Final Preparation
Thursday	31. January	LinAlg Final Preparation
Friday	01. February	TEST LinAlg
Saturday	02. February	Skiing

Diskrete Mathematik - Lecture

Prof. Ueli Maurer

Contents

1 Einführung	4
1.1 Organisation	4
1.2 Teasers	4
2 Begründen, Beweise und Logik	5
2.1 What is a Proof	5
2.2 Propositions and Logical Formulas	5
2.2.1 Logische Folgerungen	6
2.2.2 Definition $A \rightarrow B$	6
2.3 Predicates and Quantifiers	7
2.3.1 Was ist eine Primzahl	7
2.4 Some Proof Patterns	8
3 Mengenlehre	10
3.1 Mengen und Operationen	10
3.1.1 Russell's Paradox	10
3.1.2 Zermelo-Fraenkel	11
3.1.3 Leere Menge	11
3.1.4 Kardinalität	11
3.1.5 Definition der natürlichen Zahlen	12
3.2 Relation	12
3.3 Äquivalente Relationen	15
3.4 Partial Order Relations (Ordnungsrelationen)	16
3.4.1 Hasse Diagramm	17
3.5 Funktionen	17

3.5.1	Wann ist $A \sim B$	17
3.6	Endliche Folgen	20
4	Zahlentheorie	21
4.1	Einführung	21
4.2	Teiler	22
4.3	Primfaktorenzerlegung	24
4.4	Fakten über Primzahlen	24
4.5	Congruences and Modular Arithmetic	24
4.5.1	Modular Congruences	24
4.5.2	Äquivalenzklassen	25
4.5.3	Multiplikative Inverse	25
4.5.4	Chinesischer Restsatz	26
4.6	RSA Verschlüsselung	26
5	Algebra	27
5.1	Einführung	27
5.2	Groups	29
5.3	Struktur von Gruppen	31
5.3.1	Direct Products of Groups	31
5.3.2	Group Homomorphisms	31
5.3.3	Subgroups	32
5.3.4	The Order of Group Elements and of a Group	32
5.3.5	Cyclic Groups	33
5.3.6	Repetition	34
5.3.7	The Order of Subgroups	34
5.3.8	Euler's Function	35
5.4	RSA	35
5.4.1	Potenzen	36

5.5	Ring	36
5.5.1	Definition	36
5.5.2	Teilbarkeit	38
5.5.3	Units, Zerodivisors and Integral Domains	38
5.5.4	Polynomial Rings	38
5.5.5	Fields	39
5.6	Polynomials over a Field	39
5.7	Polynomials as Functions	40
5.7.1	Roots	40
5.7.2	Interpolation	41
5.8	Konstruktion von komplexen Zahlen	43
5.9	Kodierungstheorie	43
6	Logik	44
6.1	Beweissystem	44
6.2	n ist eine Primzahl	45
6.3	Elementary General Concepts in Logic	45
6.3.1	Definition Syntax	46
6.4	Literal	47
6.5	Prädikatenlogik	48
6.5.1	Syntax	48
6.5.2	Schritte für den Aufbau einer Syntax und Semantics	49
6.5.3	Erweiterung um Gleichheitszeichen	51
6.5.4	Substitution	53
6.5.5	Theorems and Theories (6.3.5)	53
6.6	Normal Forms	54
6.7	Beyond Predicate Logic (Nicht Prüfungsstoff)	55
6.8	Kalküle (6.4)	55

6.9 Resolutionskalkül (6.5.6)	55
6.9.1 Resolutionsregel	58

The following was presented in the lecture on 19. September 2018.

1 Einführung

1.1 Organisation

<https://crypto.ethz.ch/teaching/lectures/DM18/>

Prüfung ist 3 Stunden und es dürfen 6 handgeschriebene Seiten verfasst werden.

1.2 Teasers

Exercise 1. 1.1 Schachbrett $k \cdot k$ Für jedes Feld, das markiert wird, kann man das Restfeld mit L-Formen abdecken. existiert eine Belegung von L-Formen.

$\Rightarrow \forall$ Feld, das markiert wird, \exists eine Belegung von L-Formen.

$P(3) = 0$, da nur 8 Felder übrig bleiben.

$P(4) = 1$, da man die L-Formen in jede Ecke legen kann.

$P(5) = 0$, da wenn man ein Punkt bei 3,2 setzt, auf einer Seite ein 2 hohes Feld frei wird.

$P(6) = 0$

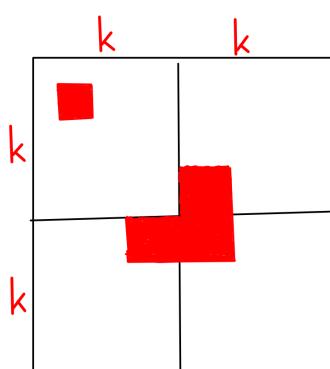
$P(7) = \text{Exercise 1 1.4}$

$$k^2 \not\equiv_3 1 \Rightarrow P(k) = 0 \quad (1)$$

$$k^2 \equiv_3 1 \Rightarrow k \equiv_3 1 \text{ or } k \equiv_3 -1 \equiv_3 2 \quad (2)$$

$$\forall n P(3n) = 0 \quad k \equiv_3 0 \quad (3)$$

Rule 1. $P(k) = 1 \Rightarrow P(2k) = 1$



Daraus kann dann diese Gleichung gefolgt werden:

$$\Rightarrow P(2^n) = 1 \quad (4)$$

2 Begründen, Beweise und Logik

Behauptung: $1 < 0$

$1 < 0 \Rightarrow 2 < 1 \Rightarrow 2 > 1$ Deshalb stimmt es.

$S \Rightarrow T$ und T beweisen. Aber man nicht daraus schliessen, das deshalb S wahr ist.

Korrekt: $S \Rightarrow T$ und S beweisen.

The following was presented in the lecture on 24. September 2018.

2.1 What is a Proof

$$S = 1 + 2 + 4 + 8 + 16 + \dots$$

Behauptung: $S = -1$

Beweis:

$$2S = 2 + 4 + 8 + 16 \dots$$

$$\Rightarrow 2S - S = S = -1 \square$$

Problem: S ist gar nicht definiert.

Example 2.1

Behauptung: $2^{143} - 1$ ist keine Primzahl

Beweis: $2^{ab} - 1 = (2^a - 1)(2^{(b-1)}a + 2^{(b-2)}a + \dots + 2^a + 1)$

Punkte fehlen im Skript bei 2.1.1

$$\Rightarrow 2^{11} - 1 \text{ teilt } 2^{143} - 1 \quad \square$$

Behauptung: $2^{8191} - 1$ ist eine Primzahl

2.2 Propositions and Logical Formulas

Definition 1. A proposition is a mathematical statement that is either true or false.

Table 1: Definition des And-Operators und Or-Operator

A	B	$A \wedge B$	$A \vee B$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Table 2: Definition Umkehroperation

A	$\neg A$
0	1
1	0

Claudia geht zu drei verschiedenen Ärzten

A: Claudia hat Masern

B: Claudia hat Windpocken

C: Claudia hat Lungenentzündung

Aussage Arzt 1: $F = A \vee (B \wedge C)$

Aussage Arzt 2: $G = \neg B \vee (\neg A \wedge \neg C)$

Aussage Arzt 3: $H = \neg A \vee (A \wedge B \wedge C)$

Table 3:

A	B	C	F	G	H	$F \wedge G \wedge H$	$F \vee G \vee H$
0	0	0	0	1	1	0	1
0	0	1	0	1	1	0	1
0	1	0	0	1	1	0	1
0	1	1	1	0	1	0	1
1	0	0	1	1	0	0	1
1	0	1	1	1	0	0	1
1	1	0	1	0	0	0	1
1	1	1	1	0	1	0	1

2.2.1 Logische Folgerungen

Tautologie (eng.: tautology): Ist immer wahr / gültig.

$F \wedge G \equiv A \wedge \neg B$

$F \wedge G \vDash A$

The following was presented in the lecture on 26. September 2018.

2.2.2 Definition $A \rightarrow B$

$$A \rightarrow B \equiv \neg A \vee B \quad (5)$$

Formel:

$$(A \vee \neg C) \wedge (\neg B \rightarrow C) \quad (6)$$

Table 4:

A	B	$A \rightarrow B$	$\neg A \vee B$	$A \wedge (A \rightarrow B)$	$(A \wedge (A \rightarrow B)) \rightarrow B$
0	0	1	1	0	1
0	1	1	1	0	1
1	0	0	0	0	1
1	1	1	1	1	1

$F \vDash G$ (Umgangssprachlich: G muss gleich oft oder öfters wahr sein als F.)

$A \rightarrow B$

$A \stackrel{\text{def}}{=} \text{"X liest das Skript und löst alle Übungen."}$

$B \stackrel{\text{def}}{=} \text{"X besteht die Prüfung."}$

Lemma 2.2

F is tautology if and only if $\neg F$ is unsatisfiable.

Lemma 2.3

$F \rightarrow G$ is a tautology if and only if $F \vDash G$

Unterschied von Formel und Aussage:

Mit einer Aussage (Statement) sagt man etwas aus. Eine Formel (Formula) kann einen beliebigen Wert annehmen.

2.3 Predicates and Quantifiers

Universum U

Prädikatenlogik $P(x,y)$

$\text{less}(x,y) = 1$ g.d.w (genau dann wenn) $x < y$

$\text{prime}(x) = 1$ g.d.w x eine Primzahl ist.

$\text{less}(3, 2) \rightarrow \text{prime}(7)$

$\exists x P(x)$

$\forall x P(x)$

$\forall x \text{less}(x, x + 1)$ Das würde stimmen.

$\forall x \text{less}(x^2, x)$ Das würde nicht stimmen.

$\exists x (\text{less}(x, 2) \wedge \text{prime}(x))$

$\forall x \exists y \text{less}(x, y)$

2.3.1 Was ist eine Primzahl

$\text{prime}(x) \stackrel{\text{def}}{=} x > 1 \wedge (\forall y \forall z ((y \cdot z = x) \vDash (y = 1) \vee (z = 1)))$

The following was presented in the lecture on 01. October 2018.

Example 1. $U = \text{Menschen}$ (inkl. gestorbene)

$$m(x) = 1 \stackrel{\text{def}}{=} x \text{ ist männlich}$$

$$w(x) = 1 \stackrel{\text{def}}{=} x \text{ ist weiblich}$$

$$\forall x(m(x) \leftrightarrow \neg w(x)) \vDash \forall x(\neg(m(x) \wedge w(x)))$$

$$elt(x, y) = 1 \stackrel{\text{def}}{=} x \text{ ist Elternteil von } y$$

$$mut(x, y) = 1 \stackrel{\text{def}}{=} elt(x, y) \wedge w(x)$$

$$kind(x, y) = 1 \stackrel{\text{def}}{=} x \text{ ist Kind von } y$$

$$kind(x, y) \stackrel{\text{def}}{=} elt(y, x)$$

$$\forall x \neg elt(x, x)$$

$$\forall x \forall y \neg(elt(x, y) \wedge elt(y, x))$$

$$gelt(x, y) \stackrel{\text{def}}{=} \exists z(elt(x, z) \wedge elt(z, y))$$

$$\forall x \exists mut(y, x)$$

$$\forall x \forall y \forall z \neg(mut(y, x) \wedge mut(z, x) \wedge y \neq z) \equiv \forall x \neg(\exists y \exists z \dots)$$

$$\exists x \exists y(elt(x, y) \wedge gelt(x, y)) \text{ wahr}$$

$$ges(x, y) \stackrel{\text{def}}{=} \exists z(elt(z, x) \wedge elt(z, y) \wedge (x \neq y))$$

$$tan(x, y) \stackrel{\text{def}}{=}$$

Reihenfolge ist sehr wichtig, betrachte folgende Beispiele:

$$\forall x \exists y elt(x, y)$$

$$\exists y \forall x elt(x, y)$$

$$\forall y \exists x elt(x, y)$$

$$\exists x \forall y elt(x, y)$$

$$vorf(x, y) \quad x \text{ ist Vorfahre von } y$$

$$\forall x \forall y elt(x, y) \rightarrow vorf(x, y)$$

$$\forall x \forall y \forall z((vorf(x, y) \wedge vorf(y, z)) \rightarrow vorf(x, z))$$

Example 2. $U = \text{Leute im Saal}$

$$P(x) \stackrel{\text{def}}{=} x \text{ versteht Quantoren}$$

$$Q(x) \stackrel{\text{def}}{=} x \text{ kann Übung 3 lösen}$$

$$\forall(P(x) \rightarrow Q(x)) \vDash (\forall x P(x)) \rightarrow (\forall x Q(x)) \quad (7)$$

2.4 Some Proof Patterns

Lemma 2.5

$$\neg B \rightarrow \neg A \vDash A \rightarrow B \quad (8)$$

Lemma 2.6

$$(A \rightarrow B) \wedge (B \rightarrow C) \vDash A \rightarrow C \quad (9)$$

The following was presented in the lecture on 03. October 2018.

Lemma 2.7

$$A \wedge (A \rightarrow B) \vDash B \quad (10)$$

S beweisen:

1. Aussage R formulieren
- 1!. R beweisen
2. $R \Rightarrow S$ beweisen

Example 3. Aussage $S = 2^{3000} \equiv_{3001} 1$

Beweis: Für jede Primzahl p gilt $2^{p-1} \equiv_p 1$

$R = 3001$ ist prim

$R \Rightarrow S$ ist ein Spezialfall vom Theorem

Lemma 2.8

$$(A_1 \vee A_2) \wedge (A_1 \rightarrow B) \wedge (A_2 \rightarrow B) \vDash B \quad (11)$$

R_1, \dots, R_k

$R_1 \Rightarrow S$

Lemma 2.9

$$(\neg A \rightarrow B) \wedge \neg B \vDash A \quad (12)$$

S beweisen:

1. Aussage T formulieren
2. S falsch \Rightarrow T beweisen
3. T falsch beweisen

Example 4. Aussage $\sqrt{2}$ ist irrational

Beweis: S falsch $\Rightarrow \sqrt{2}$ ist rational

$$\Rightarrow \sqrt{2} = \frac{m}{n} \text{ mit } ggT(m, n) = 1$$

$$\Rightarrow m^2 = 2n^2$$

$\Rightarrow m^2$ ist gerade

$\Rightarrow m$ ist gerade

$$\Rightarrow 4|m^2$$

$$\Rightarrow 4|2n^2$$

$$\Rightarrow 2|n^2$$

$\Rightarrow n$ ist gerade

$$\Rightarrow ggT(m, n) \geq 2 \text{ und } ggT(m, n) = 1$$

Deshalb ist S wahr. \square

Example 5. 2.4.9

$S_x = x > 500$ und x ist Primzahl — $x \in \mathbb{N}$ und $x+2$ ist Primzahl.

Behauptung: Aussage: S_x ist für mindestens ein x wahr. $\exists x P(x)$

Beweis $x = 521$ Beweis prüfen:

1. $x > 500$

x ist Primzahl

$x+2$ ist Primzahl

Example 6. Es gibt ∞ viele Primzahlen

m sei die grösste Primzahl. Betrachte die Zahl $m! + 1$

Example 7. 2.4.10

Das Schuhfacherprinzip besagt, dass ein Paar l_l und d_l existiert.

Table 5:

1	1	2	3	4	5	6	7	8	9	10
a_l	7	1	8	3	5	10	2	9	4	6
l_l	3	4	2	3	2	1	3	1	2	1
d_l	3	1	2	2	2	3	1	2	1	1

Example 8. 2.4.11

$$\neg \forall x P(x) \equiv \exists x \neg P(x) \quad (13)$$

Example 9. 2.4.12 Induktionsprinzip

$P(0) \wedge \forall n(P(n) \rightarrow P(n+1)) \vDash \forall n P(n)$

Bsp: 4,5,8,9,10,12,13,14,15

$P(n) \stackrel{\text{def}}{=} n + 15$ kann in 4 und 5 zerlegt werden.

$P(0)$ is true

$P(n) \rightarrow P(n+1)$

Annahme: $P(x)$ ist wahr

Fallunterscheidung: 1. Fall: $n+15$ ist zerlegbar mit einer 4.

2. Fall $n+15$ ist zerlegbar rein in 5er.

The following was presented in the lecture on 08. October 2018.

3 Mengenlehre

3.1 Mengen und Operationen

3.1.1 Russell's Paradox

$$R = \{A | A \notin A\}$$

$$R \in R$$

$$R \in R \Rightarrow R \notin R$$

$$R \notin R \Rightarrow R \in R$$

3.1.2 Zermelo-Fraenkel

Definition 2. $A = B \Leftrightarrow \forall x(x \in A \Leftrightarrow x \in B)$

Lemma 3.1

$$\{a\} = \{b\} \Leftrightarrow a = b$$

Proof 3.1

Indirekter Beweis einer Implikation $a \neq b \Rightarrow \{a\} \neq \{b\}$ \square

Example 10. $(a, b) = (c, d) : \Leftrightarrow a = c \wedge b = d$

Definition 3. $(a, b) \stackrel{\text{def}}{=} \{\{a\}, \{a, b\}\}$ $A \subseteq B \wedge B \subseteq C \Rightarrow A \subseteq C$
 $\forall (x \in A \rightarrow x \in B) \wedge \forall (x \in B \rightarrow x \in C)$
 $\Rightarrow x((x \in A \rightarrow x \in B) \wedge (x \in B \rightarrow x \in C))$
 $\Rightarrow x \in A \rightarrow x \in C$
 $\Rightarrow \forall x(x \in A \rightarrow x \in C)$

3.1.3 Leere Menge

Definition 4. $\forall x(x \notin \emptyset)$

Lemma 3.2

$$\forall A(\emptyset \subseteq A)$$

Lemma 3.3

Es gibt genau eine leere Menge.

$$[A = B \Leftrightarrow A \subseteq B \wedge B \subseteq A] \quad (14)$$

3.1.4 Kardinalität

$$|\{\emptyset\}| = 1$$

$$|\emptyset| = 0$$

3.1.5 Definition der natürlichen Zahlen

$$\begin{aligned} 0 &\stackrel{\text{def}}{=} \emptyset \\ 1 &\stackrel{\text{def}}{=} \{\emptyset\} \\ 2 &\stackrel{\text{def}}{=} \{\emptyset, \{\emptyset\}\} \\ 3 &\stackrel{\text{def}}{=} s(2) \end{aligned}$$

$$s(n) = n \cup \{n\} \quad (15)$$

$$\begin{aligned} P(A) &\stackrel{\text{def}}{=} \{S \mid S \subseteq A\} \\ P(\{\emptyset, \{\emptyset\}\}) &= \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}, \{\emptyset, \{\emptyset\}\}\} \end{aligned}$$

Example 11. Gilt $A \subseteq P(A)$ für alle A ?

Annahme es ist falsch, Beweis mit Gegenbeispiel:

$$\begin{aligned} P(\{1\}) &= \{\emptyset, \{1\}\} \\ \{1\} &\notin \{\emptyset, \{1\}\} \quad \square \end{aligned}$$

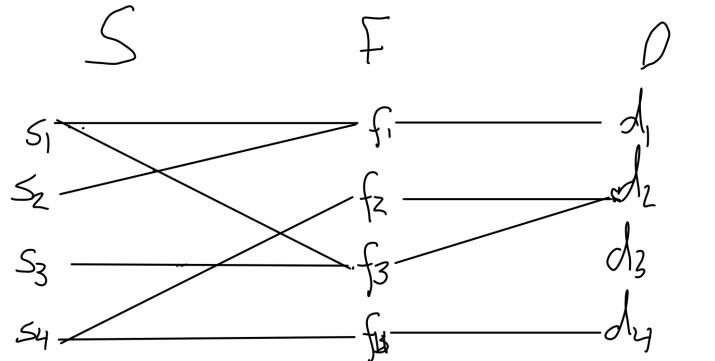
The following was presented in the lecture on 10. October 2018.

3.2 Relation

Example 12. S = Menge der Studierenden an der ETH

F = Menge der Fächer

D = Menge der Dozenten



$$\begin{aligned} bel &\subseteq S \times F \\ gen &\subseteq F \times S \\ gen &= \overbrace{bel}^{\text{gen}} (bel^{-1}) \\ (bel \circ \text{gen}) &\subseteq S \times D \end{aligned}$$

Als Matrixform

	f_1	f_2	f_3	f_4
s_1	1	0	1	0
s_2	1	0	0	0
s_3	0	0	1	0
s_4	0	1	0	1

$$\begin{aligned} p &\subseteq A \times B \\ (a, b) &\in \rho \\ (a, c) &\notin \rho \\ a\rho b &\quad a \not\sim b \end{aligned}$$

Example 13. Kleiner als Zeichen

$$\begin{aligned} < &\subseteq \mathbb{Z} \times \mathbb{Z} \\ (4, 5) &\in < \\ (7, 3) &\notin < \end{aligned}$$

Example 14. $H = \text{Menge der Menschen}$

$$\begin{aligned} x \text{ im } y &\Leftrightarrow x \text{ ist Mutter von } y \\ x \text{ iv } y &\Leftrightarrow x \text{ ist Vater von } y \\ ie = im \cup iv &(\text{ie} = \text{ist Elternteil}) \rightarrow \Leftrightarrow x \text{ im } y \vee x \text{ iv } y \\ ie &= im \cup iv \\ ik = \widehat{ie} &- xiey \Leftrightarrow y ik x \\ ige = ie \circ ie &= ie^2 \\ ig = (ik \circ im) \cap (ik \circ iv) \setminus id & \end{aligned}$$

Definition 5. $\widehat{\rho} = \{(b, a) | a \rho b\}$
 $b \widehat{\rho} a \Leftrightarrow ab$

Definition 6. $a \rho \circ \sigma c : \Leftrightarrow \exists b \in B (a \rho b \wedge b \sigma c)$

$$\begin{aligned} \rho &\subseteq A \times B \\ \sigma &\subseteq B \times C \\ \rho \circ \sigma &\subseteq A \times C \end{aligned}$$

Example 15. $\leq \circ \leq = \leq \quad A \times A$

$$\begin{aligned} \widehat{\leq} &= \geq \\ < \circ < \neq < &\quad A = \mathbb{Z} \\ < \circ < = < &\quad A = \mathbb{R}, \mathbb{Q} \\ \leq \cap \geq &\subseteq id \end{aligned}$$

$$(\rho \circ \sigma) \circ \phi$$

Lemma 3.5

$$(\rho \sigma) \phi = \rho(\sigma \phi) \tag{16}$$

Proof 3.5

$$\rho(\sigma \phi) \subseteq (\rho \sigma) \phi \tag{17}$$

$$(a, d) \in \rho(\sigma \phi) \Rightarrow \exists b \in B (a, b) \in \rho \wedge \underbrace{(b, d) \in \sigma \phi}_{\exists c \in C ((b, c) \in \sigma \wedge (c, d) \in \phi)} \tag{18}$$

$$\Rightarrow \exists b \exists c \tag{19}$$

$$\Rightarrow (a, d) \in \rho(\sigma\phi) \quad (20)$$

□

Lemma 3.6

$$\widehat{\rho\sigma} = \widehat{\sigma}\widehat{\rho} \quad (21)$$

Proof 3.6

$$c\widehat{\rho\sigma}a \Leftrightarrow a\rho\sigma c \quad \text{— Definition der Inversen} \widehat{\wedge} \text{ und von } \circ \quad (22)$$

$$\Leftrightarrow \exists b \subseteq B(a\rho b \wedge b\sigma c) \quad (23)$$

$$\Leftrightarrow \exists b \subseteq B(b\widehat{\rho}a \wedge c\widehat{\sigma}b) \quad \text{— Definition von } \wedge \quad (24)$$

$$\Leftrightarrow \exists b \subseteq B(c\widehat{\sigma}b \wedge b\widehat{\rho}a) \quad \text{— Definition von } \circ \quad (25)$$

$$\Leftrightarrow c\widehat{\sigma}\widehat{\rho}a \quad (26)$$

□

Note 1. $\rho^3 = \rho\rho\rho = \rho \circ \rho \circ \rho$ Everything is the same. Notation is not that important.

Table 6:

Eigenschaften von ρ	Formel	Menge	Graphen	Matrix
reflexiv	$\forall a(a\rho a)$	$id \subseteq \rho$		
irreflexiv	$\forall a(a \not\rho a)$	$\rho \cap id = \emptyset$		
symmetrisch	$a\rho b \Leftrightarrow b\rho a$	$\rho = \hat{\rho}$		
antisymmetrisch	$\forall a\forall b(a\rho b \wedge b\rho a) \rightarrow a = b$	$\rho \cap \hat{\rho} \subseteq id$		
transitiv	$\forall a\forall b\forall c(a\rho b \wedge b\rho c) \rightarrow a\rho c$	$\rho^2 \subseteq \rho$		

$$\rho^* = \rho \cup \rho^2 \cup \rho^3 \dots \quad (27)$$

(Beispiel von vorher: $ie^* = ivorfaire$)

The following was presented in the lecture on 15. October 2018.

3.3 Äquivalente Relationen

Example 16. 3.39 $A = \text{Menge konvexen Polygonen}$

Kann man mit mehreren Schnitten der Polygone (Verknüpfung) auf ein anderes Polygon kommen.

Wenn man mit einer Form die andere Form überdecken kann, muss es nicht heißen, dass man mit der anderen Form diese Form überdecken kann.

Example 17. 3.3 Bsp $A = \mathbb{Z}$

$$a \equiv_m b : \Leftrightarrow (a - b) = km \text{ für } k \in \mathbb{Z} \quad (28)$$

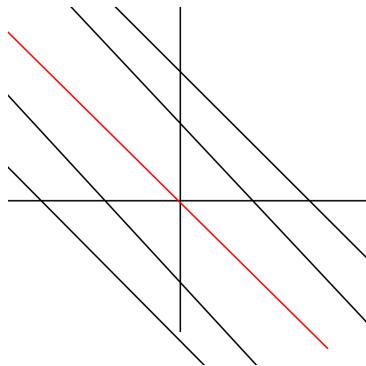
$$\text{reflektiv } a \equiv_m a \text{ mit } (a - a) = 0m \quad (29)$$

$$\text{symmetrisch } a \equiv_m b \Rightarrow b \equiv_m a \text{ mit } (a - b) = km \Rightarrow (b - a) = (-k)m \quad (30)$$

$$\text{transitiv } a \equiv_m b \wedge b \equiv_m c \Rightarrow a \equiv_m c \text{ mit } (a - b) = km \text{ und } (b - c) = k'm \Rightarrow (a - c) = (k + k')m \quad (31)$$

Example 18. $A = \mathbb{R} \times \mathbb{R}$

$$(x, y) \rho (x', y') : \Leftrightarrow x + y = x' + y' \quad (32)$$



Note 2. ρ Steht für irgendeine Relation und θ steht für eine Äquivalenzrelation

$$[a]_\theta \stackrel{\text{def}}{=} \{b \in A \mid b\theta a\} \quad (33)$$

$$A_\theta \stackrel{\text{def}}{=} \{[a]_\theta \mid a \in A\} \quad (34)$$

Example 19. $A = \mathbb{Z} \times (\mathbb{Z} \setminus \{0\})$

$$(a, b) \sim (c, d) :\Leftrightarrow ad = bc \quad (35)$$

reflektiv: $(a, b) \sim (a, b)$ weil $ab = ba$

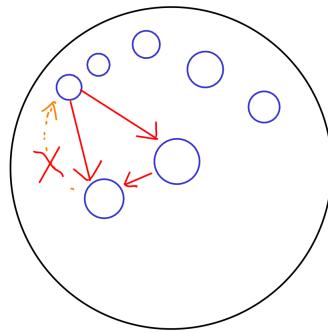
symmetrisch: weil $ad = bc \Rightarrow bc = ad$

transitiv: $ad = bc$ und $cf = de \Rightarrow adc f = bcde \Rightarrow bdf = bce \Rightarrow$ mit Fallunterschiedung bei c folgt $af = be \Leftrightarrow (a, b) \wedge (e, f)$

Note 3. Definition Rationaler Zahlen:

$$(\mathbb{Z} \times \mathbb{Z} \setminus \{0\})_\sim \stackrel{\text{def}}{=} \mathbb{Q} \quad (36)$$

3.4 Partial Order Relations (Ordnungsrelationen)



$$(A; \preccurlyeq) a \preccurlyeq b :\Leftrightarrow a \preccurlyeq b \wedge a \neq b \quad (37)$$

(B, \sqsubseteq)

Example 20. Ehemalige Prüfungsaufgabe **KOMMT AN DER PRÜFUNG**

Zeige, dass (für jede Ordnungsrelation \preccurlyeq) \prec transitiv ist.

Beweis: $a \prec b \wedge b \prec c \Rightarrow a \prec c \Leftarrow a \underset{\substack{\preccurlyeq \\ 1,3 \text{ Transitive}}}{\preccurlyeq} c \wedge a \neq c$ Gegeben:

$$1. a \preccurlyeq b$$

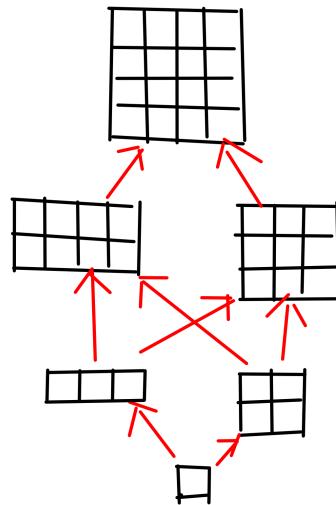
$$2. a \neq b$$

$$3. b \preccurlyeq c$$

4. $b \neq c$

Beweis, dass $a \neq c$ mittels Widerspruch

Annahme: $a = c \Rightarrow^1 c \leq b \Rightarrow^3 b = c$ Widerspruch wegen 4 \square



$$(A; \leq)(B; \sqsubseteq)$$

$$(A \times B; \sqsubseteq)(a_q, b_1) \subseteq (a_2, b_2) :\Leftrightarrow$$

$$:\Leftrightarrow a_1 \leq a_2 \wedge b_1 \sqsubseteq b_2$$

$$(A \times B, \leq)(a_1, b_1) \leq_l ex(a_2, b_2) :\Leftrightarrow a_1 \leq a_2 \vee (a_1 = a_2 \wedge b_1 \sqsubseteq b_2)$$

The following was presented in the lecture on 17. October 2018.

3.4.1 Hasse Diagramm

Im Skript lesen

3.5 Funktionen

Im Skript lesen

3.5.1 Wann ist $A \sim B$

$$A \sim B :\Leftrightarrow \text{es gibt } B_{ij} \quad A \rightarrow B \tag{38}$$

$$A \leq B :\Leftrightarrow \text{es gibt } B_{ij} \quad A \rightarrow C \text{ mit } C \subseteq B \tag{39}$$

$$A \leq \mathbb{N} \tag{40}$$

Example 21. Es gibt gleich viele Primzahlen wie natürliche Zahlen

$$\text{Primzahlen} \sim \mathbb{N} \quad (41)$$

Example 22. Es gibt gleich viele ganze Zahlen wie natürliche Zahlen

$$\mathbb{Z} \sim \mathbb{N} \quad (42)$$

Lemma 3.13

$$(i) A \leq B \wedge B \leq C \Rightarrow A \leq C \quad (43)$$

$$(ii) A \subseteq B \Rightarrow A \leq B \quad (44)$$

Theorem 3.14

$$A \leq B \wedge B \leq A \Rightarrow A \sim B \quad (45)$$

Example 23.

$$\underbrace{(0, 1)}_{(0,1) \sim \mathbb{R}} \subseteq \mathbb{R} \quad (46)$$

Theorem 3.15

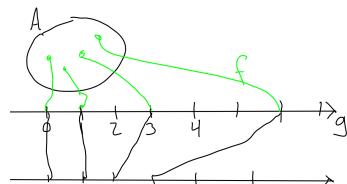
$$A \leq \mathbb{N} \Rightarrow A \text{ endlich oder } A \sim \mathbb{N} \quad (47)$$

Proof 3.15

(\Leftarrow) ✓ (Trivial)

(\Rightarrow) Sei A unendlich und $A \leq \mathbb{N}$

$$g \circ f \text{ ist } B_{ij} A \rightarrow \mathbb{N} \quad (48)$$



□

Theorem 3.16

$$\{0, 1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \dots\} \leq \mathbb{N} \quad (49)$$

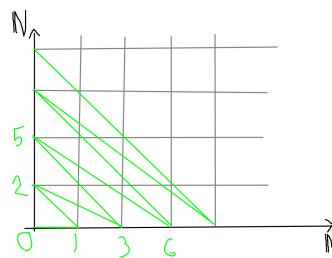
Proof 3.16

Injektion $\{0, 1\}^* \rightarrow \mathbb{N}$
 $101001 \mapsto 41$

□

Theorem 3.17

$$\mathbb{N} \times \mathbb{N} \sim \mathbb{N} \quad (50)$$



Corollary 3.18

$$A \leq \mathbb{N} \wedge B \leq \mathbb{N} \Rightarrow A \times B \leq \mathbb{N} \quad (51)$$

Proof 3.18

$$A \xrightarrow{G} \mathbb{N} \quad (52)$$

$$B \xrightarrow{G} \mathbb{N} \quad (53)$$

Aus beiden folgt $\xrightarrow{g} \mathbb{N}$

□

Theorem 3.20

A, A_0, A_1, A_2 seien abzählbar

(i) für jedes $a \in \mathbb{N}$ ist A^n abzählbar.

$$(ii) \quad A_0 \cup A_1 \cup A_2 \cup \dots \quad (54)$$

$$\bigcup_{i=0}^{\infty} A_i \leq \mathbb{N} \quad (55)$$

$$(iii) \quad A^* \sim \mathbb{N} \quad (56)$$

Proof 3.20

(i) Mittels Induktion
 A^1 ist abzählbar

$$A^n \leq \mathbb{N} \Rightarrow A^{n+1} \leq \mathbb{N} \quad (57)$$

$$A^n \times A \quad (58)$$

(ii) trivial

$$(iii) \quad A^* = \{\epsilon, A, A^2, A^3, \dots\} \quad (59)$$

□

The following was presented in the lecture on 22. October 2018.

3.6 Endliche Folgen

$$(\{0,1\}^*)^* \rightarrow \{0,1\}^* \quad \text{Injektion angeben} \quad (60)$$

Example 24.

$$(2, 37, 5, 88) \rightarrow (10, 001, 111110, 101) \rightarrow (\text{Codierung: } 0 \rightarrow 00, 1 \rightarrow 01, " \rightarrow 11) \rightarrow 010001011100000111 \quad (61)$$

$$\{0,1\}^* \rightarrow \{0,1\} \quad (62)$$

$$\mathbb{N} \rightarrow \{0,1\} \quad (63)$$

Theorem 3.21

$\{0,1\}^\infty$ ist überabzählbar.

Proof 3.21

Beweis: Widerspruchsbeweis.

Annahme: $\{0,1\}^\infty \leq \mathbb{N}$, das heisst es gibt eine Bijektion (es ist abzählbar)

<u>i</u>	<u>i-te Folge der Aufzählung</u>
0	0 1 1 0 1 0 0
1	1 0 1 1 0 1 0
2	1 1 0 0 0 1 1
3	1 0 1 0 1 0 1 0
4	0 1 0 1 0 1 0 1 0
5	0 0 0 0 0 0 0 0 0
6	1 1 1 1 1 1 0 1 1
7	0 0 0 1 1 1 0 0 0
n	1 1 0 1 1 1 0 1 0 1 ↓

Wir definieren
eine Folge wo
wir in jeder
Zeile ein Bit
flippen.

⇒ Widerspruch folge kann nicht abzählbar sein. \square

Corollary 3.22

Es gibt nicht berechenbare Funktionen.

Proof 3.22

$$\{0,1\}^* \prec \{0,1\}^\infty \quad (64)$$

\square

Theorem 3.23: Nicht Prüfungsrelevant

Theorem: Jede natürliche Zahl ist in dreizehn Worten beschreibbar. $W = \text{Wortmenge}$ (endlich)
 $W \cup W^2 \cup \dots \cup W^{13}$

Proof 3.23: Nicht Prüfungsrelevant

Widerspruchsbeweis Annahme: Es gibt eine Zahl die nicht durch dreizehn Worte beschreibbar ist. $B = (\text{Die kleinste Zahl, die nicht durch dreizehn Worte beschrieben werden kann.})$
Lösung: Die Definition verwendet die ihre Defintion. \square

4 Zahlentheorie

4.1 Einführung

Siehe Skript

4.2 Teiler

Definition 7.

$$a|b : \Leftrightarrow \exists c \ b = ac \quad (65)$$

Example 25. $a|b \wedge b|c \Rightarrow a|c$
 $a|b \wedge a|c \Rightarrow a|(b+c)$

Theorem 4.1: Euklid

$$\begin{aligned} a, d &\neq 0 \\ a &= dq + r \text{ mit } 0 \leq r \leq |d| \end{aligned}$$

$$d|a \wedge d|b \wedge \forall c((c|a \wedge c|b) \rightarrow c|d) \quad (66)$$

The following was presented in the lecture on 24. October 2018.

$$a|b : \Leftrightarrow \exists c \ b = ca \quad (67)$$

Lemma 4.2

$$\gcd(m, n - qm) = \gcd(m, n) \quad (68)$$

Proof 4.2

$$d|m \wedge d|n \Leftrightarrow d|m \wedge d|n - qm \quad (69)$$

$$\Rightarrow \begin{cases} d|m \Rightarrow m = kd \\ d|n \Rightarrow n = ld \end{cases} \Rightarrow n - qm = ld - qkd = (l - qk)d \Rightarrow d|n - qm \quad (70)$$

\Leftarrow is left as an exercise

□

Definition 8. $(a, b) \stackrel{\text{def}}{=} \{ua + vb \mid u, v \in \mathbb{Z}\}$
 $(c) \stackrel{\text{def}}{=} \{uc \mid u \in \mathbb{Z}\}$

Example 26. $(10, 16) = \{10, 16, 20, 6, 0, 26, -10, -16, 4, 2, \dots\} = (2)$
 $(7, 3) = (1) = \mathbb{Z}$

Lemma 4.3

In \mathbb{Z} : $(a, b) = (d)$ für ein d

Proof 4.3

$$\Leftrightarrow (a, b) \subseteq (d) \wedge (d) \subseteq (a, b) \quad (71)$$

Definition von d im Skript

(1) Beweis von $(d) \subseteq (a, b)$

$$d \in (a, b) \Leftrightarrow d = ua + vb \quad (72)$$

$$\Rightarrow kd = kua + kvb \in (a, b) \quad (73)$$

(2) Beweis $(a, b) \subseteq (d)$

Sei $c \subseteq (a, b)$ beliebig

$c = qd + r$ mit $0 \leq r < d$

$$c, d \in (a, b) = \underbrace{c - qd}_{r} \in (a, b) \quad (74)$$

$$\Rightarrow r = 0 \Rightarrow c = qd \in (d) \quad (75)$$

□

Algorithmus 4.1: Finde grösster gemeinsamer Teiler

```

1  $(s_1, s_2) := (a, b)$ 
2 while  $s_2 > 0$  do
3    $q := s_1 \text{ div } s_2$ 
4    $(s_1, s_2) := (s_2, s_1 - qs_2)$ 
5 end
6  $d := s_1$ 

```

Behauptung: $d = gcd(a, b)$

Proof 4.4

Terminierung weil s_2 immer strikt kleiner wird.

Invariante: $gcd(s_1, s_2) = gcd(a, b)$

Schluss: $s_2 = 0$, $gcd(s_1, 0) = s_1$

□

Note 4. $(a, b) = (d)$

$gcd(a, b) = ua + vb$ für $u, v \in \mathbb{Z}$

Algorithmus 4.2: Erweiterung des gcd Algorithmus

```

1  $(s_1, s_2) := (a, b)$ 
2  $(u_1, u_2) := (1, 0)$ 
3  $(v_1, v_2) := (0, 1)$ 
4 while  $s_2 > 0$  do
5    $q := s_1 \text{ div } s_2$ 
6    $(s_1, s_2) := (s_2, s_1 - qs_2)$ 
7    $(u_1, u_2) := (u_2, u_1 - qu_2)$ 

```

```

8      $(v_1, v_2) := (v_2, v_1 - qv_2)$ 
9 end
10  $d := s_1$ 
11  $u := u_1$ 
12  $v := v_1$ 

```

4.3 Primfaktorenzerlegung

$$\sqrt{n} = \frac{a}{b} \Rightarrow a^2 = nb^2 \quad (76)$$

4.4 Fakten über Primzahlen

4.4 ist nicht Prüfungsrelevant

The following was presented in the lecture on 29. October 2018.

4.5 Congruences and Modular Arithmetic

4.5.1 Modular Congruences

Example 27.

$$x^3 - x - 1 = y^2 \quad (77)$$

$$x \equiv_3 0, 1, 2 \Rightarrow x^3 - x - 1 \equiv_3 \text{Gibt immer } 2 \quad (78)$$

$$y \equiv_3 0, 1, 2 \Rightarrow y^2 \equiv_3 0 \text{ oder } 1 \quad (79)$$

Daraus folgt das die beiden Seiten nie gleich sein können.

Lemma 4.15

1. $a \equiv_m b \wedge c \equiv_m d \Rightarrow a + c \equiv_m b + d$
2. $a \equiv_m b \wedge c \equiv_m d \Rightarrow a \cdot c \equiv_m b \cdot d$

Proof 4.15: (1)

$$\left. \begin{array}{l} m|(a-b) \\ m|(c-d) \end{array} \right\} \Rightarrow m|(a-b) + (c-d) \Rightarrow m|[(a+c) - (b+d)] \Rightarrow a + c \equiv_m b + d \quad (80)$$

□

4.5.2 Äquivalenzklassen

$[0], [1], \dots [m - 1]$ Z ist definiert als folgendes Set.

$$Z_m = \{0, 1, \dots, m - 1\} \quad (81)$$

Lemma 4.17

1. $a \equiv_m R_m(a)$
2. $a \equiv_m b \Rightarrow R_m(a) = R_m(b)$

Proof 4.17

1. $a = qm + r_m(a) \Rightarrow m|(a - R_m(a)) \Rightarrow a \equiv_m R_m(a)$
2. Homework

□

Lemma 4.18

1. $R_m(a + b) = R_m(R_m(a) + R_m(b))$
2. $R_m(a \cdot b) = R_m(R_m(a) \cdot R_m(b))$

Example 28. $247 \cdot 3158 = 780028$

$$247 \equiv_9 2 + 4 + 7 \equiv_9 4$$

$$3158 \equiv_9 3 + 1 + 5 + 8 \equiv_9 8$$

$$247 \cdot 3158 \equiv_9 4 \cdot 8 \equiv_9 5 \not\equiv_9 7 + 8 + 2 + 8 \equiv_9 7$$

Man merkt nun, dass es nicht stimmen kann.

4.5.3 Multiplikative Inverse

$$ax \equiv_m b \quad (82)$$

Wir betrachten den Fall $b = 1$

Lemma 4.19

$ax \equiv_m 1$ hat eine Lösung $\Leftrightarrow \gcd(a, m) = 1$ und $x \in \mathbb{Z}_m$ ist eindeutig.

Proof 4.19

$$(\Rightarrow) ax \equiv_m 1 \Rightarrow \exists k ax = km + 1 \quad (83)$$

Nach Definition gilt:

$$\left. \begin{array}{l} \gcd(a, m) | a \\ \gcd(a, m) | m \end{array} \right\} \Rightarrow \gcd(a, m) | \underbrace{ax - km}_1 \quad (84)$$

Andere Richtung:

$$(\Leftarrow) \gcd(a, m) = 1 \Rightarrow \exists u \exists v \underbrace{ua}_{\equiv_m 1} + \underbrace{vm}_{\equiv_m 0} = 1 \quad (85)$$

$$vm \Rightarrow x = R_m(u) \quad (86)$$

□

4.5.4 Chinesischer Restsatz

Theorem 4.20

$$x \equiv_{m_1} a_1 \quad x \in \mathbb{Z}_M \quad (87)$$

$$\dots \quad (88)$$

$$x \equiv_{m_r} a_r \quad (89)$$

$$M = \prod_{i=1}^r m_i \quad (90)$$

$$\gcd(m_i, m_j) = 1 \quad (91)$$

Proof 4.20

$$M_i = \frac{M}{m_i} = m_1 \cdots m_{i-1} \cdot m_{i+1} \cdots m_r \quad (92)$$

$$M_i N_i \equiv_{m_i} 1 \text{ weil } \gcd(M_i, m_i) = 1 \quad (93)$$

$$M_i N_i \underbrace{\equiv_{m_k} 0}_{k \neq i} \quad (94)$$

$$x = R_M \left(\sum_{i=1}^r a_i M_i N_i \right) \equiv_{m_k} a_k \text{ für } 1 \leq k \leq r \quad (95)$$

□

4.6 RSA Verschlüsselung

$$R_P(g^{x_A}) = y_A \leftrightarrow y_B = R_P(g^{x_B}) \quad (96)$$

$$\underbrace{R_P(y_B^{x_A})}_{\substack{}} \quad \underbrace{R_P(y_A^{x_B})}_{\substack{}} \quad (97)$$

$$(g^{x_B})^{x_A} = g^{x_B x_A} \underbrace{=}_{k_{AB}} g^{x_A x_B} \quad (98)$$

The following was presented in the lecture on 31. Oktober 2018.

5 Algebra

5.1 Einführung

$$\langle S : \Omega \rangle \quad (\omega_1, \dots, \omega_r) \quad \star : S \times S \rightarrow S \quad (99)$$

$$R = \mathbb{Z} : \langle \mathbb{Z}; +, -, 0, \cdot, 1 \rangle \quad (100)$$

$$S = \{a, b, c\} \quad (101)$$

Table 7:

\star	a	b	c
a	b	c	$\not\in a$
b	c	a	b
c	a	b	c

Definition 9. $e \in S$ is LNE falls $e \star a = a$ für alle a
 $e \in S$ is RNE falls $a \star e = a$ für alle a
Falls es LNE und RNE ist, nennt man es einfach Neutralelement.

Example 29. $S = \{0, 1\}^*$

$$a \parallel \epsilon = a$$

$$\epsilon \parallel a = a$$

$$a \parallel b \neq b \parallel a$$

Lemma 5.1

Falls e LNE und e' RNE ist, dann ist $e = e'$.

Proof 5.1

$$e \star e' = e' \text{ (weil } e \text{ LNE ist)}$$

$$e \star e' = e \text{ (weil } e \text{ RNE ist)}$$

$$\Rightarrow e' = e$$

□

$$\star \text{ assoziativ} : (a \star b) \star c = a \star (b \star c)$$

$\langle M; \star, e \rangle$ Monoid

$$\langle \underbrace{G}_{geradeZahl}; \cdot \rangle$$

Definition 10. Für a ist b Linksinverse falls $b \star a = e$
und c ist Rechtsinverse falls $a \star c = e$

Lemma 5.2

Falls a in Monoid ein Linksinverses b und ein Rechtsinverses c hat, dann gilt $b=c$

Proof 5.2

$$b = b * e = b * (a * c) = (b * a) * c = e * c = c \quad (102)$$

□

Gruppe: $\langle G; *, \widehat{\cdot}, e \rangle$

G1: $*$ assotiativ

G2: Neutralelement e , $a * e = e * a = a$

G3: Jedes $a \in G$ hat ein Inverses \widehat{a} :

$$\widehat{a} * a = a * \widehat{a} = e$$

Lemma 5.3

- $(\widehat{\cdot}) = a$
- $\widehat{a * b} = \widehat{b} * \widehat{a}$
- $a * b = a * c \Rightarrow b = c$
- $b * a = c * a \Rightarrow b = c$
- $a * x = b$ hat eindeutige Lösung x

Proof 5.3

$$\begin{aligned} 5) \widehat{a} * (a * x) &= \widehat{a} * b \\ \Rightarrow (\widehat{a} * a) * x &= \widehat{a} * b \\ \Rightarrow e * x &= \widehat{a} * b \\ \Rightarrow x &= \widehat{a} * b \end{aligned}$$

□

Lemma 5.3: Abgeschwächte Axiome

Man definiert nur ein Rechtsneutralelement und Rechtsinverses.

$$a * e \Rightarrow e * a$$

$$a * \widehat{a} \Rightarrow \widehat{a} * a$$

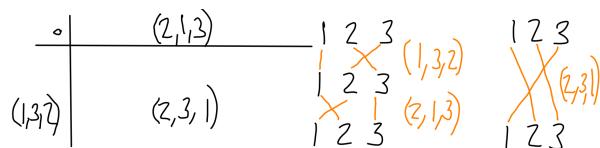
Proof 5.3

$$\begin{aligned}
 \widehat{a} * a &= (\widehat{a} * a) * e \quad (e \text{ ist RNE}) \\
 &= (\widehat{a} * a) * (\widehat{a} * \widehat{a}) \quad (\text{Definition des RI von } \widehat{a}) \\
 &= \widehat{a} * (a * (\widehat{a} * \widehat{a})) \quad (\text{Assoziativ}) \\
 &= \widehat{a} * ((a * \widehat{a}) * \widehat{a}) \quad (\text{Assoziativ}) \\
 &= \widehat{a} * (e * \widehat{a}) \quad (\text{Eigenschaft von } \widehat{a}) \\
 &= (\widehat{a} * e) * \widehat{a} \quad (\text{Assoziativ}) \\
 &= \widehat{a} * \widehat{a} \quad (\text{einsetzen RNE}) \\
 &= e
 \end{aligned}$$

□

Example 30. Bijektionen auf Menge A

Example 31. $A = \{1, 2, 3\}$, S_3



The following was presented in the lecture on 05. November 2018.

5.2 Groups

$< G_i \star, \hat{\cdot}, e >$

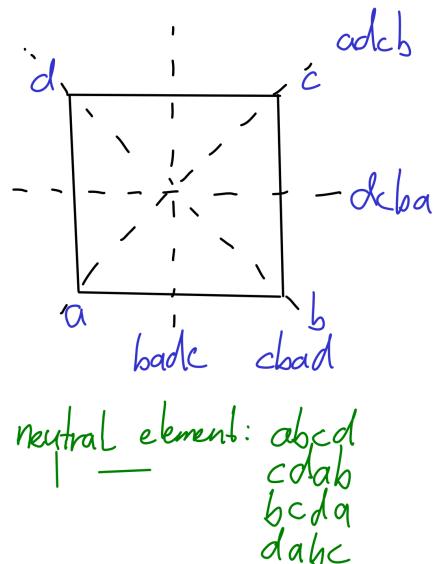
\star associative

e is neutral element: $a \star e = e \star a = a$

$\hat{\cdot}$ is the inverse:

$$a \star \hat{a} = \hat{a} \star a = e \quad (103)$$

Example 32. S_n



Example 33. $\langle \mathbb{Z}_m; \oplus, \ominus, 0 \rangle$

The above is the same as $\langle \mathbb{Z}_m; \oplus \rangle$ or \mathbb{Z}_m

Example 34. $\langle \mathbb{Z}_m^*; \odot \rangle$

The above is not a group yet: But the following is: $\mathbb{Z}_m^* = \{a \in \mathbb{Z}_m | \gcd(a, m) = d\}$

Definition 11. Direct Product

$$\langle G \times \dots \times G_n; 0 \rangle \quad (104)$$

0 is component-wise

Example 35. $S_3 \times \langle \mathbb{Z}_2; \oplus \rangle$

$$((321), 1) \circ ((123), 1) = ((321), 0) \quad (105)$$

Example 36.

$$\langle \mathbb{Z}_2; \oplus \rangle \times \langle \mathbb{Z}_2; \oplus \rangle \quad (106)$$

Table 8:

0	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	(0,0)			
(0,1)	(0,1)	(0,0)		
(1,0)	(1,0)	(1,1)	(0,0)	
(1,1)	(1,1)			(0,0)

5.3 Struktur von Gruppen

5.3.1 Direct Products of Groups

5.3.2 Group Homomorphisms

$$< \mathbb{R}^{>0}; \cdot > \quad (107)$$

$$\log(a, b) = \log(a) + \log(b) \quad (108)$$

$$< \mathbb{R}; + > \quad (109)$$

Definition 12. Group Homomorphism

A function $\psi : G \rightarrow H$

such that

$$\psi(a * b) = \psi(a) *' \psi(b) \quad (110)$$

if ψ + a bijection, then it is a group Isomorphism

Example 37. • $\det(AB) = \det(A)\det(B)$

- $R_m : \mathbb{Z} \rightarrow \mathbb{Z}_m$

Lemma 5.5

If $f : G \rightarrow H$ is a Group Homomorphism then

1. $f(e) = e'$
2. $f(\hat{a}) = \widetilde{f(a)}$

Proof 5.5

$$\textcircled{1} f(e) \stackrel{\text{G2}}{=} f(e * e) \stackrel{\text{GH}}{=} f(e) * f(e) \quad (111)$$

$$\Rightarrow \widetilde{f(e)} *' f(e) = \widetilde{f(e)} *' (f(e) *' f(e)) \stackrel{\text{G1}}{=} (\widetilde{f(e)} *' f(e)) *' f(e) \stackrel{\text{G3}}{=} e' *' f(e) \stackrel{\text{G2}}{=} f(e) \quad (112)$$

$$\textcircled{2} f(\hat{a}) *' f(a) \stackrel{\text{GH}}{=} f(\hat{a} * a) \stackrel{\text{G3}}{=} f(e) \stackrel{\text{(1)}}{=} e' \quad (113)$$

□

Example 38. Groups with 4 elements:

$$< \mathbb{Z}_4; \oplus >, < \mathbb{Z}_2; \oplus >, < \mathbb{Z}_2; \oplus > \quad (114)$$

$$5 : \mathbb{Z}_5 \quad (115)$$

$$6 : \mathbb{Z}_6, S_3 \quad (116)$$

$$< \mathbb{Z}_3; \oplus > \star < \mathbb{Z}_2; \oplus > \quad (117)$$

$$\psi(x) = (R_3(x), R_2(x)) \quad (118)$$

5.3.3 Subgroups

Definition 13. H is a subgroup of $\langle G; \star, \hat{\cdot}, e \rangle$ if $\langle H; \star, \hat{\cdot}, e \rangle$ is group.

Example 39. • $\{e\}$, G - trivial subgroups

- SG $\langle \mathbb{Z}_{10}; \oplus \rangle$: $\{0\}, \mathbb{Z}_{10}, \{0, 2, 4, 6, 8\}, \{0, 5\}$
- $\langle \mathbb{Z}_{11}^*; \odot \rangle$: $\{1\}, \mathbb{Z}_{11}^*, \{1, 10\}, \{1, 3, 5, 8, 9\}$

5.3.4 The Order of Group Elements and of a Group

Notation: $a \cdot b = ab, a^{-1}$

Powers: a^n

$$a^0 = e$$

$$a^n = a \cdot a^{n-1}$$

$$a^{-m} = (a^{-1})^m$$

Lemma 5.6

$$a^{n+m} = a^n \cdot a^m$$

$$a^{nm} = (a^n)^m$$

Definition 14. Order of a ($\text{Ord}(a)$) - the least $m \geq 1$ such that $a^m = e$ (or $\text{ord}(a) = a$)

Example 40. • $\text{ord}(e) = 1$

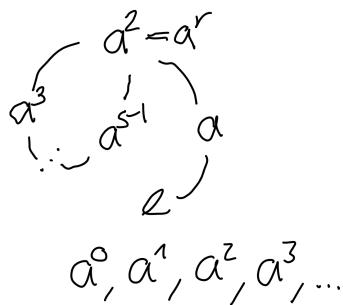
- $\langle \mathbb{Z}_{20}; \oplus \rangle$: $\text{ord}(0) = 1, \text{ord}(1) = 20, \text{ord}(6) = 10$

- $\langle \mathbb{Z}; + \rangle$ $\text{ord}(1) = \infty$

Lemma 5.6

G is finite \Rightarrow every a has a finite order.

Proof 5.6



$$a^s = a^r \text{ for some } s > r \quad (119)$$

$$A^{s-r} = a^s \cdot a^{-r} = a^r \cdot a^{-r} = a^{r-r} = a^0 = e \quad (120)$$

$$ord(a) \leq s - r \quad (121)$$

□

Definition 15. Order of G is $|G|$

5.3.5 Cyclic Groups

Definition 16. $\langle a \rangle = \{a^n | n \in \mathbb{Z}\}$ - the group generated by a

Definition 17. G is cyclic if $G = \langle g \rangle$ for some g, which is called generator.

Example 41. • $\langle \mathbb{Z}_{20}; \oplus \rangle: \{0, 5, 10, 15\} = \langle 5 \rangle$

$$\langle 1 \rangle = \mathbb{Z}_{20}$$

- $\langle \mathbb{Z}; + \rangle: \langle 0 \rangle = \{0\}$

$$\langle 1 \rangle = \mathbb{Z}$$

- $\langle \mathbb{Z}_{11}^*; \odot \rangle: \langle 10 \rangle = \{1, 10\}$

Note 5. G ist finite

- $a^m = a^{R_{Ord}(a)(m)}$

Proof 5.7

$$n = ord(a)$$

$$a^m = a^{nQ+R_n(m)} = (a^n)^Q \cdot a^{R_n(m)} = e^Q \cdot a^{R_n(m)} = a^{R_n(m)} \quad (122)$$

□

$$\langle a \rangle = \{e, a, a^2, \dots, a^{ord(a)-1}\} \quad (123)$$

$\langle \mathbb{Z}_n; \oplus \rangle$ is cyclic and the generator is 1

Theorem 5.7

G is cyclic, $|G| = n \Rightarrow G$ is isomorphism to \mathbb{Z}_n

Proof 5.7

Gi $\phi: \mathbb{Z}_n \rightarrow G$

Let $G = \langle g \rangle$, $\phi(i) = g^i$

1. bijection, because $G = \langle g \rangle$

2. homomorphism

$$\phi(i \oplus j) = g^{i \oplus j} = g^{R_n(i+j)} = g^{i+j} = g^i \cdot g^j = g^{R_n(i)} \cdot g^{R_n(j)} = \phi(i) \cdot \phi(j) \quad (124)$$

□

The following was presented in the lecture on 07. November 2018.

5.3.6 Repetition

Example 42. $\langle \mathbb{Z}_{20}^*; \odot \rangle \leftarrow \{1, 3, 7, 9, 11, 13, 17, 19\}$

Example 43. $\langle \mathbb{Z}_{20}^*; \oplus \rangle \langle 8 \rangle = \{0, 8, 16, 4, 12\}$
 $ord(8) = 5$

Example 44. $\langle \mathbb{Z}_{20}^*; \odot \rangle$

$\langle 7 \rangle = \{1, 7, 9, 3\}$

$ord(7) = 4$

Example 45. $\langle \mathbb{Z}_{13}^*; \odot \rangle \leftarrow \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$
 $\langle 2 \rangle = \{1, 2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7\}$

5.3.7 The Order of Subgroups

G, H ist UG von G

Theorem 5.8: LaGrange

—H— teilt —G—

Corollary 5.9

$ord(a)$ teilt —G—

$$|\langle a \rangle| \quad (125)$$

Corollary 5.10

$$a^{|G|} = e \quad (126)$$

Proof 5.10

$$|G| = k \cdot ord(a) \quad (127)$$

$$a^{|G|} = a^{ord(a) \cdot k} = \underbrace{(a^{ord(a)})}_e^k = e^k = e \quad (128)$$

□

5.3.8 Euler's Function

Theorem 5.13

$$\langle \mathbb{Z}_m^*; \odot_m, ^{-1}, 1 \rangle \text{ ist eine Gruppe} \quad (129)$$

Proof 5.13

Abgeschlossenheit:

$$gcd(a, m) = 1 \text{ und } gcd(b, m) = 1 \Rightarrow gcd(a \odot b, m) = gcd(ab, m) = 1 \quad (130)$$

□

$$|\mathbb{Z}_m^*| = \phi(m)$$

$$|\mathbb{Z}_{p^e}^*| = \phi(p^e) = p^e \frac{p-1}{p} = p^{e-1}(p-1) \quad (131)$$

$$\phi(m) = \prod_{i=1}^r \underbrace{p_i^{e_i-1}(p_i-1)}_{\phi(p_i^{e_i})} \quad (132)$$

Corollary 5.14: Fermat, Euler

$$a^{\phi(n)} \equiv_n 1 \quad (133)$$

$$a^{p-1} \equiv_p 1 \quad (134)$$

5.4 RSA

$$G : x \mapsto x^{e'} \quad (135)$$

$$y \mapsto y^d \quad ed \equiv_{|G|} 1 \quad (136)$$

$$(y^d)^e = y^{ed} = y^{k|G|+1} = \underbrace{(y^{|G|})^k}_e \cdot y = y \quad (137)$$

$$gcd(e, |G|) = 1 \quad (138)$$

$$m = p \cdot q \quad (139)$$

$$|\mathbb{Z}_m^*| = (p-1)(q-1) = \phi(m) \quad (140)$$

Lemma 5.7: Nicht relevant

Aus $\phi(m)$ und m kann man effizient p und q berechnen.

Proof 5.14

$$\phi(m) = pq - p - q + 1 \quad (141)$$

$$\Rightarrow q = m - \phi(m) - p + 1 \quad (142)$$

$$pq = \underbrace{m - p(m - \phi(m) - p + 1)}_{\text{quadratische Gleichung für } p} \quad (143)$$

□

Note 6. Im Skript steht n anstatt m

The following was presented in the lecture on 12. November 2018.

5.4.1 Potenzen

$$a^{23} : a \rightarrow a^2 \rightarrow a^4 \rightarrow a^8 \rightarrow a^{16} \rightarrow a^{20} \rightarrow a^{22} \rightarrow a^{23} \quad (144)$$

a^{23} berechnet in nur 7 Schritten.

$$a^{23} : a \rightarrow a^2 \rightarrow a^3 \rightarrow a^5 \rightarrow a^{10} \rightarrow a^{20} \rightarrow a^{23} \quad (145)$$

a^{23} berechnet in nur 6 Schritten.

5.5 Ring

5.5.1 Definition

$$\langle R; +, -, 0, \cdot, 1 \rangle \quad (146)$$

$\langle R; +, -, 0 \rangle$ ist eine kommutative Gruppe (Siehe folgende Lemmas) (147)

$\langle R; \cdot 1 \rangle$ ist ein Monoid (148)

Lemma 5.17

1. $a \cdot 0 = 0 = 0 \cdot a$
2. $(-a) \cdot b = -(a \cdot b)$
3. $(-a)(-b) = ab$
4. nicht relevant

Example 46. $R = \mathbb{Z}^4$

$$(a, b, c, d) + (a', b', c', d') = (a + a', b + b', c + c', d + d') \quad (149)$$

$$(a, b, c, d) \cdot (a', b', c', d') = (aa' + bc', ab' + bd', ca' + dc', ab' + dd') = (1, 0, 0, 1) = 1 \quad (150)$$

Matrixmultiplikation zwei mal zwei Matrizen.

Example 47. $R = \mathbb{Z}^*$ endlich lange Folgen

$$(a_0, a_1, a_2) + (b_0, b_1, b_2, b_3) = (a_0 + b_0, a_1 + b_1, a_2 + b_2, b_3) \quad (151)$$

$$(f_o, f_1, f_2, f_3) = (a_0, a_1, a_2) \cdot (b_0, b_1, b_2, b_3) = (a_0 b_0, a_0 b_1 + a_1 b_0, a_0 b_2 + a_1 b_1 + a_2 b_0, \dots) \quad (152)$$

$$f_i = \sum_{u,v:u+v=i} a_u b_v \quad (153)$$

Proof 5.17: (1)

$$0 = -(a_0) + a_0 \quad (154)$$

$$= -(a_0) + a(0 + 0) \quad (155)$$

$$= -(a_0) + (a_0 + a_0) \quad (156)$$

$$= (-(a_0) + a_0) + a_0 \quad (157)$$

$$= 0 + a_0 \quad (158)$$

$$= a_0 \quad (159)$$

□

Lemma 5.18

+ ist kommutativ

Proof 5.18

$$(1 + 1)(a + b) = (1 + 1)a + (1 + 1)b \quad (160)$$

$$= 1a + 1a + 1b + 1b \quad (161)$$

$$= (a + a) + (b + b) \quad (162)$$

$$= a + ((a + b) + b) \quad (163)$$

$$(1 + 1)(a + b) = 1(a + b) + 1(a + b) \quad (164)$$

$$= (1a + 1b) + (1a + 1b) \quad (165)$$

$$= (a + b) + (a + b) \quad (166)$$

$$= a + ((b + a) + b) \quad (167)$$

$$a + ((a + b) + b) = a + ((b + a) + b) \quad (168)$$

$$((a + b) + b) = ((b + a) + b) \quad (169)$$

$$((a + b)) = ((b + a)) \quad (170)$$

$$a + b = b + a \quad (171)$$

□

5.5.2 Teilbarkeit

$$a|b : \Leftrightarrow \exists c \quad b = ac \quad (172)$$

$$\text{Lemma } a|b \text{ und } b|c \Rightarrow a|c \quad (173)$$

$$a|b \Rightarrow a|bc \text{ für } c \neq 0 \quad (174)$$

$$a|b \text{ und } a|c \Rightarrow a|(b + c) \quad (175)$$

$$a \cdot b = 0 \Rightarrow a = 0 \text{ oder } b = 0 \quad (176)$$

Example 48. $\langle \mathbb{Z}_m; \oplus_m, \ominus, 0, \odot_m, 1 \rangle$

$$Z_{20} : 8 \odot 15 = 0$$

5.5.3 Units, Zerodivisors and Integral Domains

Definition 18. Einheit $u \in R$

falls $u \cdot v = 1$ für ein v

Definition 19. $R^* =$ Menge der Einheit in R

Lemma 5.19

$$\langle R^*; \cdot, (\cdot)^{-1}, 1 \rangle \quad \text{ist eine Gruppe} \quad (177)$$

Proof 5.19

$$(u \cdot v)^{-1} = v^{-1} \cdot u^{-1} \quad (178)$$

□

5.5.4 Polynomial Rings

$$R[\times]$$

Example 49. \mathbb{Z}_5 :

$$a(x) = 2x^2 + 3x + 1 \quad (1, 3, 2) \quad (179)$$

$$b(x) = 3x + 4 \quad (4, 3) \quad (180)$$

$$a(x) + b(x) = 2x^2 + x \quad (0, 1, 2) \quad (181)$$

$$a(x) \cdot b(x) = x^3 + 2x^2 + 4 \quad (4, 0, 2, 1) \quad (182)$$

$$D[\times]^* = D^*$$

5.5.5 Fields

$$\langle F; \dots \rangle \quad F^* = F \setminus \{0\} \quad (183)$$

$F[\times] \mathbb{Z}_p$ (p=prim) ist ein Körper
 $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$

Example 50. $F = \{0, 1, A, B\}$

The following was presented in the lecture on 14. November 2018.

5.6 Polynomials over a Field

Note 7. $\mathbb{Z}_p : GF(p)$

Example 51. $GF(5)[x]$

$$(2x + 3)(3x + 1) = x^2 + x + 3 \quad (184)$$

Example 52. $GF(2)[x]$

$$(x^2 + x + 1)(x + 1) = x^3 + 1 \quad (185)$$

Example 53. $F[x]$

$$\mathbb{R}[x] \quad x^2 - 3x + 2 = (x - 1)(x - 2) \quad (186)$$

π ist ein Teiler der obigen Gleichung:

$$\pi \left(\frac{1}{\pi}x^2 - \frac{3}{\pi}x + \frac{2}{\pi} \right) \quad (187)$$

$$gcd(x^2 - 3x + 2, x^2 - 4x + 3) = x - 1 \quad (188)$$

Example 54. $\mathbb{R}[x]$: Irreducible Polynom $ax + b$ Zum Beispiel: $x^2 + 1$

Example 55. $GF(2)[x] \quad x^3 + x + 1$

Example 56. $\mathbb{R}[x]$:

$$(x^4 + 5x^2 + 6) : (x^2 - x + 2) = x^2 + x + 4 \quad Rest(2x - 2) \quad (189)$$

Example 57. $GF(2)[x]$

$$(x^4 + x + 1) : (x^2 + x + 1) = x^2 + x \quad Rest(1) \quad (190)$$

5.7 Polynomials as Functions

5.7.1 Roots

Example 58. $GF(5)$

$$a(x) = 2x^3 + 3x + 1 \quad (191)$$

Table 9:

α	$a(\alpha)$	$a(\alpha) + 1$
0	1	2
1	1	2
2	3	4
3	4	0
4	1	2

Lemma 5.28

α ist Wurzel von $a(x) \Leftrightarrow (x - \alpha)$ teilt $a(x)$

Proof 5.27

$$(\Rightarrow) \quad a(x) = (x - \alpha)q(x) + r(x) \quad (192)$$

$$r = a(x) - (x - \alpha)q(x) \quad (193)$$

$$x = \alpha: \quad r = a(\alpha) - 0 \cdot q(x) = 0 \quad (194)$$

(\Leftarrow) im Skript □

Theorem 5.30

$$0 \neq a(x) \in F[x], \deg(a(x)) = d \quad (195)$$

$\Rightarrow a(x)$ hat $\leq d$ Nullstellen

Proof 5.30

Widerspruchsbeweis:

Annahme: Es hat $e > d$ Nullstellen $\alpha_1, \dots, \alpha_e$

$$\Rightarrow \prod_{i=1}^e (x - \alpha_i) \text{ teilt } a(x) \quad (196)$$

□

5.7.2 Interpolation**Example 59.**

$$a(x) = a_2 x^2 + a_1 x + a_0 \quad (197)$$

$$a(1) = 2 \quad a(3) = 1 \quad a(4) = 4 \quad (198)$$

lösbar mittels Gauss-Verfahren.

The following was presented in the lecture on 19. November 2018.

$$F[x] \quad GF(5) \quad (199)$$

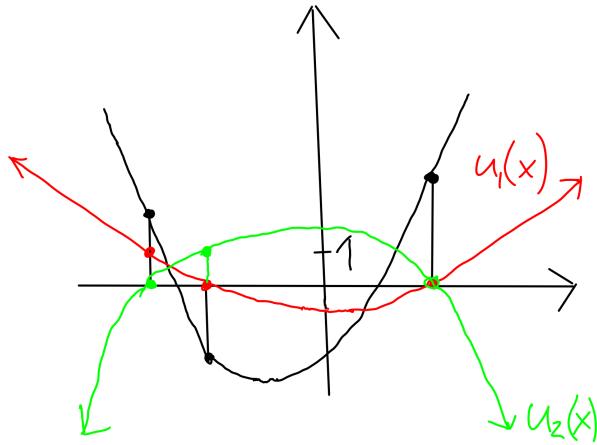
$$a(1) = 2 \quad (200)$$

$$a(3) = 1 \quad (201)$$

$$a(4) = 4 \quad (202)$$

$$(203)$$

$$a(x) = a_2 x^2 + a_1 x + a_0 \quad (204)$$



$$u_1(x) = \frac{(x - \alpha_1)(x - \alpha_2) \cdot \dots \cdot (x - \alpha_{i-1})(x - \alpha_{i+1}) \cdot \dots \cdot (x - \alpha_{d+1})}{(\alpha_i - \alpha_1)(\alpha_i - \alpha_2) \cdot \dots \cdot (\alpha_i - \alpha_{d+1})} \quad (205)$$

$$a(x) = \sum_{i=1}^{d+1} \beta_i u_i(x) \quad (206)$$

Mit einem konstruktiven Existenzbeweis kann man nun zeigen

$$a(\alpha_j) = \beta_j \quad a'(x) \quad a(x) - a'(x) = 0 \quad (207)$$

$$\mathbb{Z} \quad \mathbb{Z}_m \quad \mathbb{Z}_m^* = \{a | \gcd(a, m) = 1\} \quad (208)$$

$$F[x] \quad F[x]_{m(x)} \quad (209)$$

$$a(x) \equiv_{m(x)} b(x) : \Leftrightarrow m(x) | (a(x) - b(x)) \quad (210)$$

$$\mathbb{R}[x] 5x^3 - 2x + 1 \equiv_{3x^2+2} 8x^3 + 1 \equiv_{3x^2+2} -\frac{16}{3}x + 1 \quad (211)$$

Lemma 5.33

$$\left. \begin{array}{l} \deg(m(x)) = d \\ |F| = q \end{array} \right\} \Rightarrow |F[x]_{m(x)}| = q^d \quad (212)$$

Lemma 5.34

$$\langle F[x]_{m(x)}; \oplus, \ominus, 0, \odot, 1 \rangle \text{ ist ein Ring} \quad (213)$$

Lemma 5.35

$$F[x]_{m(x)}^* = \{a(x) | \gcd(a(x), m(x)) = 1\} \quad (214)$$

$$\gcd(a(x), m(x)) = u(x)a(x) + v(x)m(x) \quad (215)$$

$$\mathbb{R}[x]_{x^2+1} : (2x - 1) \cdot (x + 4) = 7x - 6 \quad (216)$$

$$GF(2)[x]_{x^3+x+1} : (x^2 + x + 1)(x^2 + 1) = x^2 + x \quad (217)$$

Example 60.

$$\boxed{\mathbb{R}[x]_{x^2+1} = \mathbb{C}} \quad (218)$$

Nun berechnen wir das inverse Element von $2x + 3$ in diesem Körper

$$(2x + 3)(ax + b) = 1 + t(x^2 + 1) \quad (219)$$

$$gcd(2x + 3, x^2 + 1) = 1 = \left(-\frac{2}{13}x + \frac{3}{13}\right)(2x + 3) + \frac{4}{13}(x^2 + 1) \quad (220)$$

5.8 Konstruktion von komplexen Zahlen

Example 61.

$$(ax + b) \odot (cx + d) = (ad + bc) + (bd - ac) \quad (221)$$

Example 62.

$$|(GF(2)[x]_{x^3+x+1})^*| = 7 \quad (222)$$

$$x^0 = 1 \quad (223)$$

$$x^1 = x \quad (224)$$

$$x^2 = x^2 \quad (225)$$

$$x^3 = x^1 \quad (226)$$

$$x^4 = x^2 + x \quad (227)$$

$$x^5 = x^2 + x + 1 \quad (228)$$

$$x^6 = x^2 + 1 \quad (229)$$

$$x^7 = 1 \quad (230)$$

The following was presented in the lecture on 21. November 2018.

$$F[x]_{m(x)} \rightarrow F' \quad (231)$$

5.9 Kodierungstheorie

$$A = \{0, 1\}, \quad \{0, 1\}^8 \quad |A| = q \quad (232)$$

$$\text{Enkodierfunktion} : A^k \rightarrow A^n \quad (233)$$

$$(a_0, \dots, a_{k-1}) \mapsto (c_0, \dots, c_{n-1}) \quad (234)$$

$$\text{Dekodierfunktion} : A^n \rightarrow A^k \quad (235)$$

$$(r_0, \dots, r_{n-1}) \mapsto (a_0, \dots, a_{k-1}) \quad (236)$$

Die Hemmingdistanz ist die Anzahl der Fehler die aufgetreten sind.

$$d((r_0, \dots, r_{n-1}), E((a_0, \dots, a_{k-1}))) \leq t \quad (237)$$

$$\Rightarrow D((r_0, \dots, r_{n-1})) = (a_0, \dots, a_{k-1}) \quad (238)$$

$$Im(E) = C \subset A^n \quad (239)$$

Beim Dekodieren sucht man die Distanz zum nächsten Wort mit Distanz kleiner als t. Deshalb müssen aber alle Wörter einen Abstand von mindestens $2t+1$ haben, sodass das Wort eindeutig ist.

$$d_{min}(C) \geq 2t + 1 \Leftrightarrow C \text{ ist } t \text{ fehler-korrigiert.} \quad (240)$$

Example 63. Code für $A = \{0, 1\}$ mit $k = 3, n = 7, d_{min} = 4$

$$A = F \quad |F| \geq n \quad (241)$$

$$E((a_0, \dots, a_{k-1})) = (a(\alpha_0), a(\alpha_1), \dots, a(\alpha_{n-1})) \quad (242)$$

$$n - (k - 1) = n - k + 1 = d_{min} \quad (243)$$

Example 64. CD

$A = GF(2^8)$ Von $k=24$ auf $n=28$ kodiert und dann nocheinmal von $k=28$ auf $n=32$.

6 Logik

6.1 Beweissystem

$$\sum^* \quad \sum = \{0, 1\} \quad (244)$$

Beispiel: eine Zahl als Bitmuster interpretiert, welche keiner Primzahl entspricht:

S ist die Aussage

τ besagt, ob die Aussage wahr oder falsch ist.

p ist der Beweis

ϕ ist die Funktion, um den Beweis effizient zu verifizieren.

$$\pi = (\underbrace{S}_{\{0,1\}^*}, \underbrace{P}_{\{0,1\}^*}, \tau, \phi) \quad (245)$$

Note 8. Falsche Aussagen haben kein Beweis. Für jede wahre Aussage gibt es ein Beweis.

The following was presented in the lecture on 26. November 2018.

Table 10:

S	$\tau(S)$	p	$\phi(s, p)$
ϵ	1	ϵ	
0	1	ϵ	
1	1	ϵ	
00	1	ϵ	
01	1	ϵ	
10	0		
11	0		
000	1	ϵ	
001	1	ϵ	
010	0		
011	0		
101	0		
110	1	10 oder 11	
1000011	1	101 oder 111	
1000101	0		

6.2 n ist eine Primzahl

$$|\mathbb{Z}_n^*| = n - 1 \quad (246)$$

$$n - 1 = p_1 p_2 \dots p_k \quad (247)$$

$$g^{n-1} \equiv_n 1 \quad (248)$$

$$g^{\frac{n-1}{p_i}} \not\equiv_n 1 \quad (249)$$

6.3 Elementary General Concepts in Logic

Example 65. 1. $(A \vee Z) \rightarrow C$

2. $B \rightarrow A$

3. $E \rightarrow Z$

4. $(A \wedge C) \rightarrow (D \vee E)$

5. $(D \wedge A) \rightarrow Z$

6. $A \wedge B$

Die Frage ist nun, ob aus diesen Bedingungen Z folgt. Die Antwort ist Ja.

$$R_1 \ F \wedge G \vdash_{R_1} F$$

$$R_2 \ F \vdash_{R_2} F \vee G$$

$$R_3 \ \{F, F \rightarrow G\} \vdash_{R_3} G$$

$$R_4 \ \{F, G\} \vdash_{R_4} F \wedge G$$

$$R_5 \quad \{(F, G) \rightarrow H, G\} \vdash_{R_5} F \rightarrow H$$

$$R_6 \quad \{F \rightarrow H, G \rightarrow H\} \vdash_{R_6} (F \vee G) \rightarrow H$$

Beweisführung

1. Aus (6) folgt $\vdash_{R_1} A$ anhand von (7) mit $F = A, G = B$
2. Aus (7) folgt $\vdash_{R_2} A \wedge Z$ anhand von (8) mit $F = A, G = Z$
3. Aus (1) und (8) folgt $\vdash_{R_3} C$ anhand von (9) mit $F = A \vee Z, G = C$
4. Aus (7) und (9) folgt $\vdash_{R_4} A \wedge C$ anhand von (10) mit $F = A, G = C$
5. Aus (10) und (4) folgt $\vdash_{R_3} D \vee E$ anhand von (11) mit
6. Aus (5) und (7) folgt $\vdash_{R_5} D \rightarrow Z$ anhand von (12)
7. Aus (3) und (12) folgt $\vdash_{R_6} (D \vee E) \rightarrow Z$ anhand von (13)
8. Aus (11) und (13) folgt $\vdash_{R_3} Z$

$$S = \{(M, G)\} \tag{250}$$

$$\tau((M, G)) = 1 : \Leftrightarrow M \vDash G \tag{251}$$

$$P = (R_1, 6, 7, F = A, G = B), (R_2, \dots) \tag{252}$$

Nun kann man das Verifizieren, indem man es einsetzt.

6.3.1 Definition Syntax

Einführung eines Alphabets Λ dann sind $\Lambda^* \geq$ korrekte Formeln.

$$\text{AL:} \Lambda = \{A_i, \wedge, \vee, \neg, (,)\} \tag{253}$$

$$\text{PL:} \Lambda = \{A_i, \wedge, \vee, \neg, (,), \exists, \forall\} \tag{254}$$

Die Pfeile \rightarrow und \leftrightarrow sind erlaubt, da sie mit den obigen Symbolen dargestellt werden können.

The following was presented in the lecture on 28. November 2018.

Note 9. Eine Gleichung gilt, wenn für alle Werte die linke und rechte Seite des Gleichzeichens den gleichen Wahrheitswert haben.)

$$\sigma(F, \mathcal{A}) = 0/1 \tag{255}$$

$$\sigma(F \wedge G, \mathcal{A}) = 1 : \Leftrightarrow \sigma(F, \mathcal{A}) = 1 \text{ und } \sigma(G, \mathcal{A}) = 1 \tag{256}$$

Fall F unter A wahr ist, schreibt man:

$$A \vDash F \tag{257}$$

$$M = \{F_1, \dots, F_k\} \tag{258}$$

$$\mathcal{A} \vDash M \quad (259)$$

$$A \vDash F_1 \wedge F_2 \wedge \dots \wedge F_k \quad (260)$$

Die Interpretation weisst jedem Symbol einen Wert zu..

Eine Tautologie ist es, wenn es für jede einzelne Interpretation wahr ist.

$$F \vDash G : \Leftrightarrow F \text{ ist eine Teilmenge von } G \quad (261)$$

$$F \equiv G \Leftrightarrow F \vDash G \text{ und } G \vDash F \quad (262)$$

Note 10. Eine Tautologie bedeutet nicht gleich eins, sondern dass die Funktion immer wahr ist.

$$F \text{ ist unerfüllt} : \Leftrightarrow \vDash (F \rightarrow \perp) \Leftrightarrow F \equiv \perp \quad (263)$$

$$M \vdash_K G \Leftrightarrow M \vDash G \quad (264)$$

Example 66. $F = (A \vee \neg B) \wedge C$

$$G = A \rightarrow C$$

$H = B \wedge \neg C$ $\{F, G, H\}$ ist unerfüllbar.

Table 11:

A	B	C	F	G	H
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	0

6.4 Literal

$$A, \neg A, B, \neg B \quad (265)$$

$$(L_{11} \vee, \dots, \vee L_{1m_1}) \wedge (L_{21} \vee, \dots, \vee L_{2m_2}) \wedge \dots \text{ CNF} \quad (266)$$

$$(L_{11} \wedge, \dots, \wedge L_{1m_1}) \vee (L_{21} \wedge, \dots, \wedge L_{2m_2}) \vee \dots \text{ DNF} \quad (267)$$

$$\Rightarrow (\neg A \wedge B \wedge \neg C) \vee (A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \quad (268)$$

The following was presented in the lecture on 03. December 2018.

			Table 12: $(A \wedge \neg B) \vee (B \wedge \neg C)$
A	B	C	
0	0	0	0
0	0	1	0
0	1	0	1
Example 67.			0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

6.5 Prädikatenlogik

6.5.1 Syntax

$$x_i \in x, y, z, u, v, w \quad (269)$$

$$f_i^{(k)(0)} \in f, g, h, a, b, c \quad (270)$$

$$P_i^{(k)} \quad (271)$$

Terme: $x_i \in f_i^{(k)}(t_1, \dots, t_k)$ Man kann nicht einfach f schreiben $\quad (272)$

Es kommt draufan wie viele Argumente f hat. $\quad (273)$

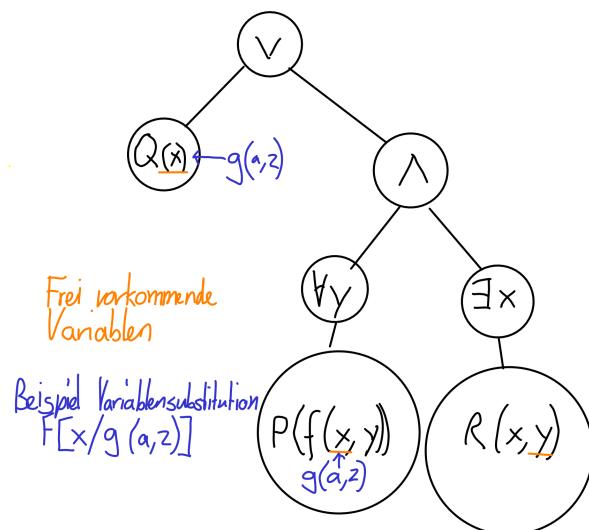
Formel: $P_i^{(k)}(t_1, \dots, t_k) \quad (274)$

Falls eine Formel F existiert dann ist auch folgendes eine Formel: $\forall x_i F, \exists x_i F \quad (275)$

$\quad (276)$

Example 68. 6.19 im Skript

$$F = Q(x) \vee (\forall y P(f(x, y)) \wedge \exists x R(x, y)) \quad (277)$$



6.5.2 Schritte für den Aufbau einer Syntax und Semantics

- Nach Definition 6.34 muss man immer zuerst ein Universum fixieren, zum Beispiel:

$$U^A \text{ oder } U = \{0, 1, 2\} \text{ oder } U = \mathbb{N} \quad (278)$$

- Allen Funktionssymbolen muss eine Funktion zugewiesen werden.

$$f_i^{(k)} \mapsto f_i^{(k)^A} : U^k \rightarrow U \quad A = (U, \phi, \psi, \xi) \quad (279)$$

- Den Prädikatsymbolen muss ein bestimmter Wert zugewiesen werden.

$$P_i^{(k)} \mapsto P_i^{(k)^A} : U^k \rightarrow \{0, 1\} \quad (280)$$

$$\psi(P_i^{(k)}) \quad (281)$$

Example 69. 6.21 im Skript

Eine mögliche Interpretierung:

$$U^A = \mathbb{N} \quad (282)$$

$$a^A = 3 \quad (283)$$

$$f^A = \text{Addition daher gilt } f(x, y) = x + y \quad (284)$$

$$P^A(x) = 1 : \Leftrightarrow x \text{ gerade} \quad (285)$$

Die obige Interpretierung wäre richtig

Oder eine andere Interpretierung:

$$U^{A'} = \mathbb{R} \quad (286)$$

$$a^{A'} = 2 \quad (287)$$

$$f^{A'} = \text{Multiplikation daher gilt } f(x, y) = x \cdot y \quad (288)$$

$$P^{A'}(x) = 1 : \Leftrightarrow x \geq 0 \quad (289)$$

Die obige Interpretierung wäre falsch.

Auswertung Interpretierung

$$\forall x(P(x) \vee P(f(x, a))) \quad (290)$$

Definition 20. 6.36 im Skript

$$\mathcal{A}(F) \quad (291)$$

$$1. \mathcal{A}(t)$$

Mit $\mathcal{A}_{[x \rightarrow u]}$ kann man das x mit u überschreiben.

$$\mathcal{A}(\forall x G) = 1 : \Leftrightarrow \mathcal{A}_{[x \rightarrow u]}(G) \text{ für alle } U \quad (292)$$

Example 70.

$$\forall xP(x) \wedge \exists xQ(x) \wedge \forall x(Q(x) \leftrightarrow \neg P(x)) \quad (293)$$

Example 71.

$$\forall xP(x,x) \wedge \exists x\exists y\exists z(\neg P(x,y) \wedge \neg P(x,z) \wedge \neg P(y,z)) \quad (294)$$

Example 72.

$$F_1 : \underbrace{\forall xP(x,f(x))}_{f(x) \text{ gibt anderen Wert als } x} \wedge \underbrace{\forall x\neg P(x,x)}_{\text{Irreflektiv}} \wedge \underbrace{\forall x\forall y\forall z((P(x,y) \wedge P(y,z)) \rightarrow P(x,z))}_{\text{Transitiv}} \quad (295)$$

$u, f(u), f(f(u)), \dots$: Aus dem obigen Beispiel folgt, dass das Universum Unendlich gross sein muss.

Example 73.

$$F_2 : \forall xP(x,x) \text{ Reflektiv} \quad (296)$$

$$F_3 : \forall x\forall y(P(x,y) \rightarrow P(y,x)) \text{ Symmetrisch} \quad (297)$$

Aus F_1, F_2, F_3 folgt eine Äquivalenzrelation. Also $\{F_1, F_2, F_3\} \vDash G$

The following was presented in the lecture on 05. December 2018.

Example 74. Modell \mathcal{A}

$$U^A = \mathbb{N}, f^A : x \mapsto x + 1$$

$$P^A(x,y) = 1 : \Leftrightarrow x < y$$

Lemma 6.10

Jedes Modell für M hat ein unendliches U.

Proof 6.10

Sei A ein beliebiges Modell für M. Sei $u \in U$ ein beliebiges fixes Element.

Def: u_i für $i \in \mathbb{N}$; $u_0 = u$, $u_{i+1} = f(u_i)$ für $i \geq 1$

$$P(u_i, u_{i+1}) = 1 \quad (F_1) \quad (298)$$

$$P(u_i, u_i) = 0 \quad (F_2) \quad (299)$$

$$P(u_i, u_i) = 1 \text{ für } j \neq i \text{ wegen Transitivität} \quad (F_3) \quad (300)$$

Widerspruchsbeweis:

Sei U^A endlich dann gibt es r und s mit $r \neq s$ sodass $u_r = u_s$

Es gilt $P(u_r, u_s) = 0$

Es gilt $P(u_r, u_s) = 1$ Widerspruch!

□

Lemma 6.11

$\forall x F \vDash F[x/t]$ für alle Formeln F und Terme t

Proof 6.11

Sei t ein beliebiger Term und f eine beliebige Formel.

Falls $\mathcal{A}(\forall x F) = 1$, dann gilt

$A_{[x \rightarrow u]}(F) = 1$ für alle $u \in U^{\mathcal{A}}$

also auch für $u = A(t) \Rightarrow \mathcal{A}(F[x/t]) = 1$

□

6.5.3 Erweiterung um Gleichheitszeichen

$$\mathcal{A}(t_1 = t_2) = 1 \Leftrightarrow \mathcal{A}(t_1) = \mathcal{A}(t_2) \quad (301)$$

Note 11. Das sind zwei unterschiedliche Gleichheitszeichen. Auf der linken Seite ist das Logik-gleichheitszeichen. Auf der rechten Seite das Gleichheitszeichen wie wir es im Alltag verstehen.

Example 75.

$$GA\{\text{G1: } \forall x(f(x, a) = x) \quad (302)$$

$$\text{G2: } \forall x \forall y \forall z(f(x, f(y, z)) = f(f(x, y), z)) \quad (303)$$

$$\text{G3: } \forall x(f(x, g(x)) = a)\} \quad (304)$$

$$GA \vDash G \quad (305)$$

Lemma 6.12

$$GA \vDash \forall x(f(a, x) = x) \quad (306)$$

Lemma 6.13

$$GA \vDash \forall x(f(g(x), x) = a) \quad (307)$$

Proof 6.12

$$\text{G1: } f(\underbrace{f(g(x), x)}_t, a) = f(g(x), x) \text{ Universal Instantiation für } t = f(g(x), x) \quad (308)$$

$$\text{G2: } f(\underbrace{f(g(x), x)}_t, a) = f(g(x), f(x, a)) \text{ Universal Instantiation für } t \text{ für } x \ g(x) \quad (309)$$

□

Proof 6.13: konventiell

$$\hat{x}x = (\hat{x}x)e = \hat{x}(xe) = \hat{x}(x(\hat{x}\hat{x})) = \hat{x}(e(\hat{x})) = \hat{x}\hat{x} = e \quad (310)$$

□

Lemma 6.14

$$\{(t_1 = t_2), (t_2 = t_3)\} \vDash (t_1 = t_3) \quad (311)$$

Lemma 6.15

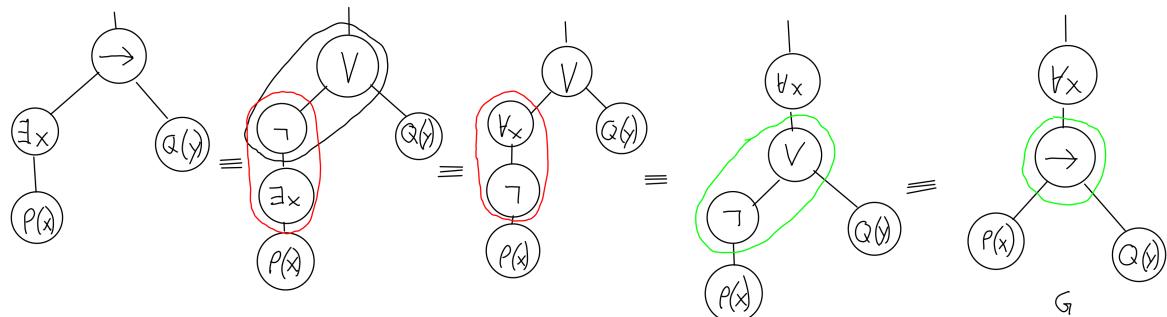
$$(t_1 = t_2) \vdash (t_2 = t_1) \quad (312)$$

Example 76.

$$F = (\exists x P(x)) \rightarrow Q(y) \quad (313)$$

$$F = \forall x(P(x)) \rightarrow Q(y) \quad (314)$$

Behauptung $F \vDash G$



The following was presented in the lecture on 10. December 2018.

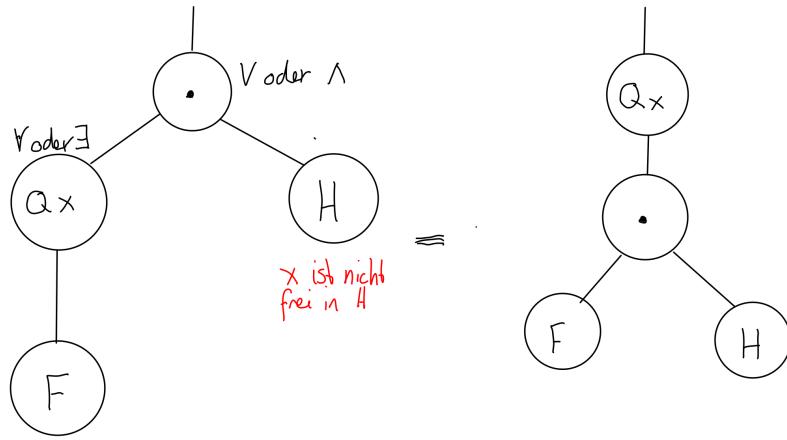
Lemma 6.8

$$\neg \forall x F \equiv \exists x \neg F \quad (315)$$

$$\forall x F \wedge \forall x G \equiv \forall x(F \wedge G) \quad (316)$$

$$\forall x F \vee \forall x G \vDash \forall x(F \vee G) \quad (317)$$

Es reicht aus \vDash oder \Rightarrow zu zeigen, sodass beide eine logische Folgerung des Anderen sind.



6.5.4 Substitution

$$F[x/t] \quad (318)$$

Lemma 6.9

$$\forall x G \equiv \forall y G[x/y] \quad (319)$$

$$\exists x G \equiv \exists y G[x/y] \quad (320)$$

Proof 6.10

Sei A eine beliebige Interpretation und sei $u \in U^A$ beliebig.
Dann gilt:

$$A_{[x \rightarrow u]}(G) = A_{y \rightarrow u}(G[x/y]) \quad (321)$$

□

Example 77.

$$(\forall x P(x)) \wedge Q(x) \quad (322)$$

Diese Form kann auch bereinigt werden und sieht dann so aus:

$$(\forall y P(y)) \wedge Q(x) \quad (323)$$

Example 78. 6.24 im Skript

See Script

6.5.5 Theorems and Theories (6.3.5)

$$T \vDash F \quad (324)$$

$$\{\} \vDash F \text{ wird einfach geschrieben als } \vDash F \quad (325)$$

$$T \subseteq T' \Rightarrow \quad (326)$$

6.6 Normal Forms

Theorem 6.13

$$\neg \exists x \forall y (P(y, x) \leftrightarrow \neg P(y, y)) \quad (327)$$

$$\forall x \exists y \neg (P(y, x) \leftrightarrow \neg P(y, y)) \quad (328)$$

$\forall x \exists y (P(y, x) \leftrightarrow P(y, y))$ ist eine Tautologie (329)

Proof 6.13

Sei A eine beliebige passende Interpretation. $U^A p^A$

Sei $u \in U^A$ beliebig.

$$A_{[x \rightarrow y]}(\exists y (P(y, x) \leftrightarrow P(y, y))) = 1 \text{ für alle } u \quad (330)$$

weil

$$A_{[x \rightarrow u][y \rightarrow v]}(P(y, x) \leftrightarrow P(y, y)) = 1 \text{ für alle } u \quad (331)$$

weil $A(P(u, u) \leftrightarrow P(u, u)) = 1$ für alle u

□

Corollary 6.16: Interpretation

$U = \text{alle Mengen}$

$P(x, y) = 1 \Leftrightarrow x \in y$

Corollary 6.15: Interpretation 2

$U = \mathbb{N}$

$\{0, 1\}^\infty$ ist überabzählbar

Proof 6.15

$$U = \mathbb{N}, \quad P(y, x) = 1 \Leftrightarrow y\text{-te Bit der } x\text{-ten Folge ist } 1 \quad (332)$$

$$y \mapsto \neg P(y, y) \quad (333)$$

□

Corollary 6.17

$U = \mathbb{N}$

$$P(y, x) \Leftrightarrow \text{Outputbit von Program (mit Index) } x \text{ für Input } y \quad (334)$$

The following was presented in the lecture on 12. December 2018.

Example 79.

$$F = \forall x \exists y (P(x, f(x)) \vee \neg P(x, y)) \quad (335)$$

Gilt genau dann wenn $y = f(x)$

$$\forall x (P(x, f(x)) \vee \exists y \neg P(x, y)) \quad (336)$$

Sei $\mathcal{A}(U^A, P^A, f^A)$ beliebig.
 $u \in U^A$ sei beliebig.

$$A_{[x \rightarrow u]}(P(x, f(x))) = 0 \Leftrightarrow P(u, f(u)) = 0 \text{ für jedes } u \quad (337)$$

$$\Leftrightarrow A_{[x \rightarrow u][y \rightarrow f(u)]}(\neg P(x, y)) = 1 \text{ für jedes } u \quad (338)$$

$$\Rightarrow A_{[x \rightarrow u]}(\exists y \neg P(x, y)) = 1 \quad (339)$$

$$\text{Deshalb } A_{[x \rightarrow u]}(P(x, f(x)) \vee \exists y \neg P(x, y)) = 1 \text{ für jedes } u \quad (340)$$

$$\Rightarrow A(\forall x (P(x, f(x)) \vee \exists y \neg P(x, y))) = 1 \quad (341)$$

Deshalb ist F eine Tautologie.

6.7 Beyond Predicate Logic (Nicht Prüfungsstoff)

$$\forall x \exists y P(x, y) \equiv \exists f \forall x P(x, \overline{f(x)}) \quad (342)$$

6.8 Kalküle (6.4)

$$\underbrace{\{F_1, F_2, F_3, F_4, \dots, F_k\}}_{M \vdash_K G} \quad F_{k+1} \quad , \dots, G \quad (343)$$

mit der Regel R_i kann F_{k+1} hergeleitet werden.

$$\{F, G\} \vdash_{R_i} F \wedge G \quad (344)$$

$$\neg \neg F \vdash F \quad (345)$$

Ein Kalkül ist eine Menge von Regeln. $\{R_1, \dots, R_m\}$

$$M \vdash_K G \Rightarrow M \vDash G \quad \text{Korrektheit} \quad (346)$$

$$M \vdash_K G \Leftarrow M \vDash G \quad \text{Vollständigkeit} \quad (347)$$

Example 80.

$$\{F \rightarrow G, G \rightarrow F\} \vdash_R \underbrace{F \wedge G}_{\neq} \quad (348)$$

Die obige Regel ist falsch.

6.9 Resolutionskalkül (6.5.6)

Lemma 6.3

$$\underbrace{\{F_1, \dots, F_k\}}_M \vDash G \Leftrightarrow \{F_1, \dots, F_k, \neg G\} \text{ ist unerfüllbar} \quad (349)$$

Proof 6.3

Linke Seite bedeutet: Jede Interpretation passend für F_1, \dots, G , die Modell für M ist, ist auch Modell für G . (Äquivalent: ist nicht Modell für nicht G)

Rechte Seite bedeutet: Es gibt keine Interpretation, die Modell $\{F_1, \dots, F_k, \neg G\}$ ist. \square

$$F = (L_{11} \vee L_{12} \vee L_{1m}) \wedge (\quad) \quad (350)$$

$$\kappa(F) = \{\{L_{11}, L_{12}, \dots, L_{1m}\}, \{L_{21}, L_{22}, \dots, L_{2m}\}, \dots\} \quad (351)$$

$$\kappa(M) = \bigcup_{i=1}^k \kappa(F_i) \quad (352)$$

Example 81.

$$\{\{A, \neg B, C\}, \{\neg A, C, D\}, \{\neg A, \neg D\}\} \quad (353)$$

Nun kann man zum Beispiel A und $\neg A$ identifizieren und dann folgt folgendes

$$\{\neg B, C, D\} \quad (354)$$

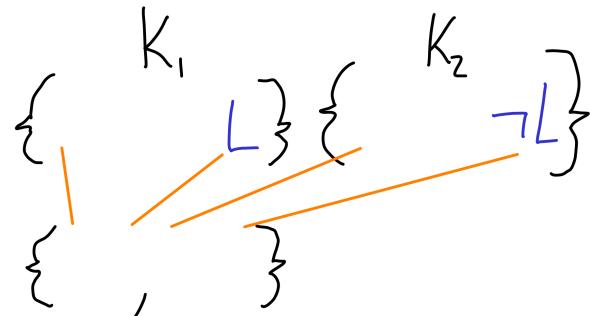
Oder

$$\{\neg B, C, \neg D\} \quad (355)$$

Aus den obigen Gleichungen kann man nun D eliminieren und es folgt.

$$\{\neg B, C\} \quad (356)$$

Das Ziel von diesem Kalkül ist es eine leere Menge also $\{\}$ zu finden. Dann folgt: \perp



The following was presented in the lecture on 17. December 2018.

$$\text{Kalkül } M \vdash_K G \Rightarrow M \vDash G \text{ Korrektheit} \quad (357)$$

$$M \vdash_K G \Leftarrow M \vDash G \text{ Vollständigkeit (gilt im Resolutionskalkül nicht)} \quad (358)$$

Example 82.

$$1. A \wedge Z \rightarrow Z \quad (359)$$

$$2. B \rightarrow A \quad (360)$$

$$3. E \rightarrow Z \quad (361)$$

$$4. (A \wedge C) \rightarrow (D \vee E) \quad (362)$$

$$5. (D \wedge A) \rightarrow Z \quad (363)$$

$$6. A \wedge B \quad (364)$$

Folgt aus diesen sechs Formeln Z ? \Rightarrow Zuerst in CNF bringen und Klausel aufschreiben.

$$1. \neg(A \vee Z) \vee C \stackrel{\text{DG}}{\equiv} (\neg A \wedge \neg Z) \vee C \quad \text{als Klausel } \{\neg A, C\}, \{\neg Z, C\} \quad (365)$$

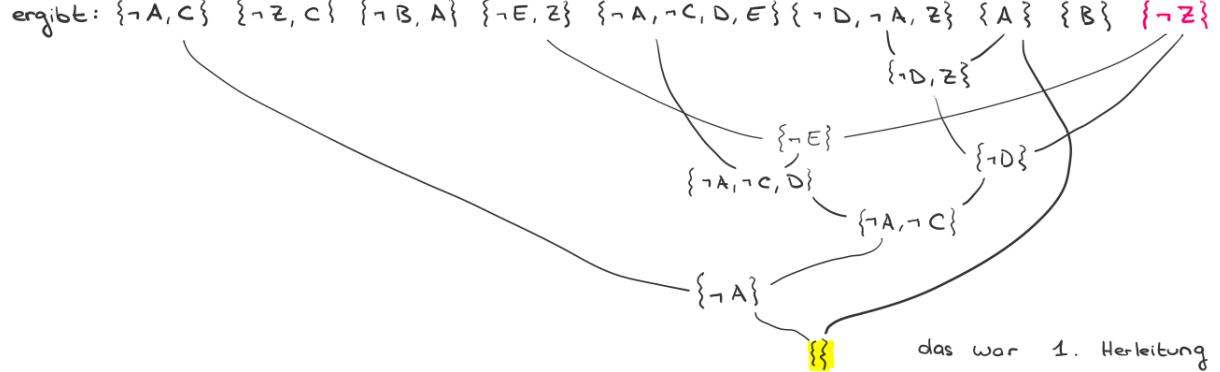
$$2. \neg B \vee A \quad \text{als Klausel } \{\neg B, A\} \quad (366)$$

$$3. \neg E \vee Z \quad \{\neg E, Z\} \quad (367)$$

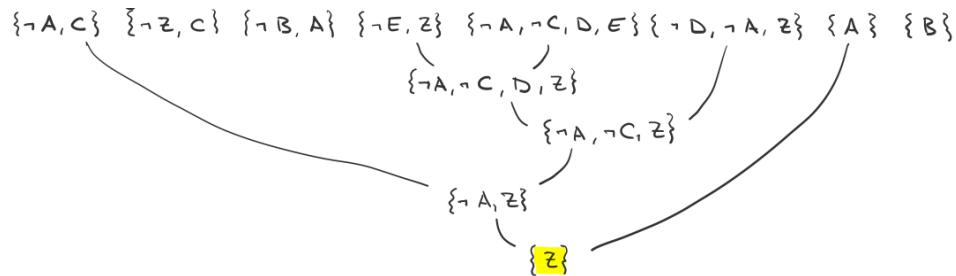
$$4. \neg(A \wedge C) \vee D \vee E \equiv \neg A \vee \neg C \vee D \vee \quad \{\neg A, \neg C, D, E\} \quad (368)$$

$$5. \neg D \vee \neg A \vee Z \quad \{\neg D, \neg A, Z\} \quad (369)$$

$$6. A \wedge B \quad \{A\}, \{B\} \quad (370)$$



$$\Rightarrow M \cup \{\neg Z\} \text{ ist unerfüllbar} \quad (371)$$

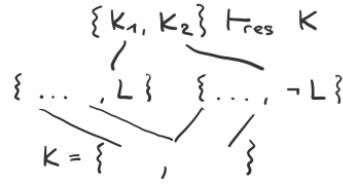


So kann man direkt herleiten, dass Z eine logische Folgerung ist.

Note 12. Beide Herleitungen sind korrekt. Die 2. Herleitung ist meistens schwieriger.

6.9.1 Resolutionsregel

$Res = \{res\}$ besteht aus einer Regel (372)



K ist die Vereinigung dieser beiden.

Lemma 6.6

$$\mathcal{K} \vdash_{Res} K \Rightarrow \mathcal{K} \models K \text{ Korrektheitsbehauptung} \quad (373)$$

Proof 6.6

Wir beweisen \vdash_{Res} also $\{K_1, K_2\} \models K$
Fallunterscheidung:

1. Wenn L wahr ist, ist mindestens ein Symbol aus K_2 wahr.
2. Und wenn L falsch ist, ist mindestens etwas aus K_1 wahr.

□

Example 83.

$$\{\{A\}, \{B\}\} \not\vdash_{Res} \{A, B\} \quad (374)$$

$$A \wedge B \vDash A \vee B \quad (375)$$

$$\neg(A \vee B) \equiv \neg A \wedge \neg B \quad (376)$$



Note 13. Resolutionskalkül Vollständig bezüglich: Wenn etwas unerfüllbar ist, muss leere Klausel herleitbar sein.

Theorem 6.7

$$M \text{ ist unerfüllbar} \Leftrightarrow \mathcal{K}(M) \vdash_{Res} \emptyset$$

Proof 6.7

$$(\Leftarrow) \mathcal{K}(M) \vdash_{Res} \emptyset \Rightarrow \underbrace{\mathcal{K}(M)}_{\text{unerfüllbar}} \vDash \underbrace{\emptyset}_{\text{unerfüllbar}} \quad (377)$$

\Rightarrow Induktionsbeweis über Anzahl atomare Formeln.

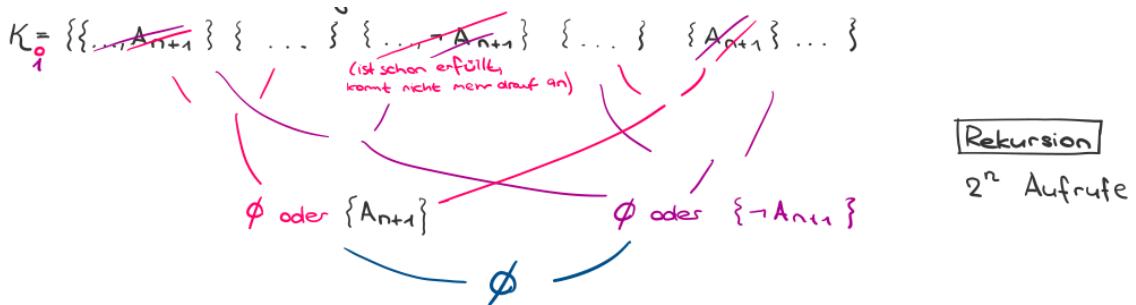
Verankerung ($n = 1$)

$$\emptyset, \{A_1\}, \underbrace{\{\neg A_1\}}_{\{\}}, \{A_1, \neg A_1\}$$

Induktionsschritt:

Annahme: Jede Klauselmenge \mathcal{K} mit atomaren Formeln A_1, \dots, A_n ist genau dann unerfüllbar wenn $\mathcal{K} \vdash_{Res} \emptyset$

Sei \mathcal{K} eine Klauselmenge mit atomaren Formeln A_1, \dots, A_{n+1}



Durch Streichen erhalten wir neue Klausel \mathcal{K}_0

\mathcal{K}_0 ist genau dann erfüllbar, wenn \mathcal{K} erfüllbar ist.

□

Diskrete Mathematik - Exercise

Prof. Ueli Maurer
TA: Jiamin Zhu H17 CAB

Contents

1	Exercise	3
2	Exercise (1/2, 0/2, 0/3)	5
2.1	Difference between Formula and Statement	6
3	Exercise (3.5/4, 2/4) Paper	9
3.1	Statement and Formula	9
4	Exercise (2/3, 0/6) Paper	10
4.1	Relation	10
5	Exercise (2.5/4, 1/4)	11
5.1	Hasse diagram	12
5.2	Hasse diagram definitions	12
5.3	Countable / Uncountable	12
6	Exercise (0.5/3, 0/5)	13
6.1	Modulo	14
6.2	Correction 6.6	15
7	Exercise (3/3, 3/5)	16
7.1	Algebra	16
7.2	$\langle s, \star \rangle$ Algebraic structure	16
7.3	$\langle m \star \rangle$ Monoid	16
7.4	Group (G, \star)	17
7.5	Commutative Group	17
7.6	Homomorphism	17

7.7	Isomorphism	17
7.8	Cyclic group	17
7.9	Prove Extended gcd Algorithm	18
7.10	Chinese Remainder Theorem	18
8	Exercise (3/5, 0/3)	18
8.1	Diffie Hellman	19
8.2	Rings	20
9	Exercise (2/2, 0/3)	21
10	Exercise (2/4, 2/4)	23
10.1	Distance	24
10.2	Logic	25
10.3	Proof System	25
10.4	Syntax	25
10.5	Semantics	25
10.6	Propositional Logic	25
11	Exercise (3.5/4, 1.5/4)	26
11.1	Aussagenlogik	27
11.2	Syntax	27
11.3	Semantik	27
11.4	CNF	27
11.5	DNF	28
11.6	$F \vDash G$	28
11.7	Prädikatenlogik	28
11.8	Auswertung	29
12	Exercise (3/3, 5/5)	31

12.1 Ersetzen	32
12.2 Rectified Formulas	33
12.3 Pränex Form	33
12.4 Uncomputable Functions	34

1 Exercise

1.1 Hilbert's Hotel

a)

1 moves to 2, 2 to 3, 3 to 4, etc.

Roger Federer to 1

b)

1 to 2, 2 to 4, 3 to 6, etc.

Guest 1 to 1, Guest 2 to 3, Guest 3 to 5

Correction: Bus People do not know where to go!

$H, n \rightarrow 2^n$

$B, n \rightarrow 2n - 1$

c)

$H, n \rightarrow 2^n$

$B_1, n \rightarrow 3^n$

$B_2, n \rightarrow 5^n$

...

$B_{100}, n \rightarrow Prim(100)^n$

There are many more solutions.

1.2 Alice

- It is not Sunday because then both would say the truth.
- Case 1 Monday: Tweedledum would say it is Tuesday, Wednesday, ... and Sunday.
- Case 2 Tuesday: Tweedledum would say it is Monday, Wednesday, ... and Sunday.
- Case 3 Wednesday: Tweedledum would say it is Monday, Tuesday, ... and Sunday.
- Case 4 Thursday: Tweedledum would say it is Thursday, but not every other day
- Case 5 Friday: Tweedledum would say it is Friday, but not every other day
- Case 6 Saturday: Tweedledum would say it is Saturday, but not every other day

Correction: The above is wrong.

Table 1:

M,T,W	Twe	Twu
Twe	0	1
Twu	0	1

Table 2:

T,F,S	Twe	Twu
Twe	1	0
Twu	1	0

Table 3:

Sun	Twe	Twu
Twe	1	0
Twu	0	1

It is Sunday and everybody tells the truth.

1.3

$x \leq 250$
 $x \equiv_8 1 \Rightarrow 1, 9, 17, 25, \dots$
 $x \equiv_7 2 \Rightarrow 2, 9, 16, 23, \dots$
 $x \equiv_5 3 \Rightarrow 3, 8, 13, 18, \dots$

Python Programm:

Daraus folgt Sie haben 233 Kokusnüsse gesammelt.

1.4

a

- 1 Middle
 - 3 Up
 - 6 Corner
- So you need to check at least 10

b

Make drawings. The middle three cases are difficult.

The following exercise was presented on 24. September 2018.

2 Exercise (1/2, 0/2, 0/3)

2.1 Should be clear

2.2 neither A nor B $\Rightarrow \neg A \wedge \neg B$

either A or B $\Rightarrow (A \vee B) \wedge \neg(A \wedge B) \equiv (A \wedge \neg B) \vee (\neg A \wedge B)$

$$A \rightarrow B \stackrel{\text{def}}{=} B \vee \neg A$$

Table 4:

A	B	$A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Logical Formula

$F \rightarrow G$ Statement and not the same as above:

$F \vDash G$

Table 5:

A	B	F	G
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

$F \vDash G$ is **Correct**.

$G \vDash F$ is incorrect.

The same as: F is a consequence of G is incorrect

$F \equiv G$ means that everything in the truth table is always the same. Statement.

\perp is always false.

\top is always true

Formula1 \rightarrow Formula2

Statement1 \Rightarrow Statement2

Table 6:

A	\perp	$A \vee \perp$
0	0	0
1	0	1

2.1 Difference between Formula and Statement

A formula (or well-formed formula) is a string in the language of the mathematical logic you are using. Formulas are built from primitive formulas (like 1 or x) or are built from combinations of smaller formulas (like $1+x$ or $x=1$).

For instance, in first-order logic, the string $((x \wedge T) \vee y)$ is a formula, specifically an expression of boolean logic with two variables.

A statement is a formula which has a well-defined (although possibly unknown) truth-value. $\forall x, y \in \mathbb{N} : (Sx < Sy) \Rightarrow (x < y)$ is a statement, true under the standard definitions of natural numbers, successor, and less-than. The example formula above is not a statement because its truth-value depends on the values of x, y .

There is a subtle distinction between a statement being “true” and a statement being “provable”. Kurt Gödel showed that there can be statements which can neither be proved nor disproved in any sufficiently powerful mathematical system, but some such statements can be true. The details of this are beyond the scope of this answer, but is something to be aware of.

(Source: <https://www.quora.com/What-is-the-difference-between-a-statement-and-a-formula-in-logic>)

Script Example 2.6

$$(A \wedge \neg B) \vee (B \wedge \neg C) \equiv (A \vee B) \wedge \neg(B \wedge C) \quad (1)$$

Table 7:

A	B	C	F	G	$F \vee G$	H	I	$H \wedge I$
0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	1	0
0	1	0	0	1	1	1	1	1
0	1	1	0	0	0	1	0	0
1	0	0	1	0	1	1	1	1
1	0	1	1	0	1	1	1	1
1	1	0	0	1	1	1	1	1
1	1	1	0	0	0	1	0	0

Script Example 2.10

$$(A \rightarrow B) \wedge (B \rightarrow C) \vDash A \rightarrow C \quad (2)$$

Table 8:

A	B	C	$A \rightarrow B$	$B \rightarrow C$	$A \rightarrow C$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	1

The following was corrected in the exercise on 1. October 2018.

2.1 A Proof

You cannot just assume $n^2 \leq n$. Because does not have to be true for the largest natural number. You would first have to prove that. And that is not possible.

The above is wrong. But you cannot assume that n is the largest number. You would have to prove first that there is a largest number.

2.2 Propositional Logic

- (a) $\neg A =$ The monkey does not sit on the palm tree
 $\neg B =$ The monkey does not have a banana
 So for i) it would be: $F_1 :=$ The monkey
 So for ii) it would be: $F_2 :=$ The monkey does not
- (b) i) $F_3 := \neg(A \vee B)$
 ii) $F_4 := (A \vee B) \wedge \neg(A \wedge B)$
- (c) The monkey sits on the palm tree or has a banana. $\neg F_3 = (A \vee B)$
 The monkey either doesn't sit on the palm tree and doesn't have a banana or he has both.
 $\neg F_4 = (A \leftrightarrow B)$

Correction: Totally wrong!

2.3 Function Tables and Equivalence

Table 9:
(a)

A	B	C	$B \rightarrow C$	$A \rightarrow C$	$A \vee B$	$(\neg(A \rightarrow C) \wedge \neg(A \vee B))$	$(B \rightarrow C) \rightarrow (\neg(A \rightarrow C) \wedge \neg(A \vee B))$
0	0	0	1	1	0	0	0
0	0	1	1	1	0	0	0
0	1	0	0	1	1	0	1
0	1	1	1	1	1	0	0
1	0	0	1	0	1	0	0
1	0	1	1	1	1	0	0
1	1	0	0	0	1	0	1
1	1	1	1	1	1	0	0

(b) $B \wedge \neg C$

(c) $(B \rightarrow C) \rightarrow (\neg(A \rightarrow C) \wedge \neg(A \vee B))$

$(\neg B \vee C) \rightarrow (\neg(\neg A \vee C) \wedge (\neg A \wedge \neg B))$ | the alternative notation of \rightarrow and de Morgan's rule

$(\neg B \vee C) \rightarrow ((A \wedge \neg C) \wedge (\neg A \wedge \neg B))$ | double negation

$\neg(\neg B \vee C) \vee ((A \wedge \neg C) \wedge (\neg A \wedge \neg B))$ | the alternative notation of \rightarrow

$(B \wedge \neg C) \vee ((A \wedge \neg C) \wedge (\neg A \wedge \neg B))$ | double negation

$(B \wedge \neg C) \vee ((A \wedge \neg A) \wedge (\neg C \wedge \neg B))$ | Associativity

$(B \wedge \neg C) \vee (\perp \wedge (\neg C \wedge \neg B))$ | $F \wedge \neg F \equiv \perp$

$(B \wedge \neg C) \vee \perp$ | $F \wedge \perp \equiv F$

$B \wedge \neg C$ | $F \vee \perp \equiv F$

2.4 Logical Consequence

Note 1. It would have been easier with truth table.

(a) $A \wedge (A \rightarrow B) \equiv A \wedge (\neg A \vee B) \equiv (A \wedge \neg A) \vee (A \wedge B) \equiv \perp \vee (A \wedge B) \equiv A \wedge B \models B \square$

(b) $A \rightarrow B \models \neg A \rightarrow \neg B$

$\neg A \vee B \models \neg \neg A \vee \neg B$

$\neg A \vee B \models A \vee \neg B$

Disproved!

(c) $(A \rightarrow B) \wedge (B \rightarrow C) \models (A \rightarrow C)$

$(A \rightarrow C) \models (\neg A \vee C)$ | Lifting

$(\neg A \vee C) \models (\neg A \vee C) \square$

2.5 Satisfiability and Tautologies

(a) $(A \vee B) \wedge \neg A \equiv (A \wedge \neg A) \vee (B \wedge \neg A) \equiv B \wedge \neg A$

The formula is satisfiable, because it is true for one truth assignment. But it is not a tautology, because it is not true for all of them.

(b) $((A \rightarrow B) \wedge (B \rightarrow C)) \wedge \neg(A \rightarrow C) \vDash (A \rightarrow C) \wedge \neg(A \rightarrow C) \equiv \perp$
 It is unsatisfiable. (It is important to use the \vDash sign here and not \equiv)

2.6 Knights and Knaves

Where would your friends from the other village say, the village is? Then take the other path.

Better way:

Question: If I were to ask "is it the right way" will you say yes? $A \equiv$ The guy is knight

$\neg A \equiv$ The guy is knave

$B \equiv$ is the right way

$\neg B \equiv$ is the not right way

$C \equiv$ the answer to "is the right way" is yes

$\neg C \equiv$ the answer to "is the right way" is no

$D \equiv$ the answer to the whole sentence is yes

$\neg D \equiv$ the answer to the whole sentence is no

Prove $\vDash D \leftrightarrow B$

$$\vDash A \wedge B \rightarrow C$$

$$\vDash A \wedge \neg B \rightarrow \neg C$$

$$\vDash \neg A \wedge B \rightarrow \neg C$$

$$\vDash \neg A \wedge \neg B \rightarrow C$$

$$\vDash A \wedge C \rightarrow D$$

$$\vDash A \wedge \neg C \rightarrow \neg D$$

$$\vDash \neg A \wedge C \rightarrow \neg D$$

$$\vDash \neg A \wedge \neg C \rightarrow D$$

3 Exercise (3.5/4, 2/4) Paper

3.1

$$F := (A \vee \neg C) \wedge (A \vee B \vee C) \wedge (\neg A \vee \neg B) \equiv (A \vee \neg C) \wedge (\neg A \vee \neg B) \equiv \neg C \wedge \neg B$$

The following was presented in the exercise on 01. October 2018.

3.1 Statement and Formula

If a formula has only one interpretation then it is also a statement.

The following was presented in the exercise on 08. October 2018.

4 Exercise (2/3, 0/6) Paper

4.1

Aus $S \neg T$

$\neg S T$

4.3

Wie kann man durch die Punkte schneiden?

4.5 und 4.6 sind typische Prüfungsfragen

\in und \subseteq nicht verwechseln und strikt unterscheiden.

4.6a) $A \subseteq B \Leftrightarrow P(A) \subseteq P(B)$

let $S \leftarrow \dots$

4.6b)

$\Rightarrow (x, y) \in (A \times B) \cap (A \times C)$

The following was presented in the lecture on 15. October 2018. relation is a set

Relation from A to B is a subset of $A \times B$

Relation from A to A is a subset of $A \times A$

Example 1. $A = \{1, 2\}$ $B = \{2, 3\}$

$A \times B = \{(1, 2), (1, 3), (2, 2), (2, 3)\}$

$$\rho_1 = \rho_2 \rightarrow \text{if } x \in \rho_1 \Rightarrow x \in \rho_2 \text{ and if } x \in \rho_2 \Rightarrow x \in \rho_1 \quad (3)$$

$$\rho^{-1} = \{(a, b) | (b, a) \in \rho\} \quad (4)$$

4.1 Relation

1. Definitions 2. Operations

1. \cup, \cap, \setminus

2. composition

3. inverse

4. p^*

3. Properties

1. reflexive $\forall a (a, a) \in \rho$

2. irreflexive $\forall a(a, a) \notin \rho$
3. symmetric $\forall a \forall b[(a, b) \in \rho \rightarrow (a, b) \in \rho]$
4. antisymmetric $\forall a \forall b[(a, b) \in \rho \wedge (b, a) \in \rho \rightarrow a = b]$
5. transitive $\forall a \forall b \forall c[(a, b) \in \rho \wedge (b, c) \in \rho \rightarrow (a, c) \in \rho]$
6. equivalence *reflexive \wedge symmetric \wedge transitive*

equivalence class

if ρ is equivalence relation on A

$$[a]_\rho = \{b \in A | (a, b) \in \rho\} \quad (5)$$

The following was presented in the exercise on 22. October 2018.

5 Exercise (2.5/4, 1/4)

Proof 5.1

We have to prove that $a(\rho \cup \sigma)b = a(\rho \circ \sigma)b$ for arbitrary a and b in A.

Part 1 - $\rho \cup \sigma \Rightarrow \rho \circ \sigma$

1. Form $a(\rho \cup \sigma)b$ we can follow that either $a\rho b$ or $a\sigma b$ has to be true.
2. Definition 3.20 says that equivalence relations are reflexive, symmetric and transitive, therefore: $a\rho a$, $a\sigma a$, $b\rho b$ and $b\sigma b$.
3. So it is true that: $a\rho b\sigma b$ or $a\rho a\sigma b$ (From step 2 and 3)
4. Now we use Definition 3.13 which says $x\rho\sigma z \Leftrightarrow \exists y \in Y(x\rho y \wedge y\sigma z)$
Therefore $a\rho b\sigma b = a\rho\sigma b$ and $a\rho a\sigma b = a\rho\sigma b$
5. The above is the same, therefore $a(\rho \cup \sigma)b \Rightarrow a(\rho\sigma)b \Rightarrow a(\rho \circ \sigma)b$ must be true.

Part 2 - $\rho \circ \sigma \Rightarrow \rho \cup \sigma$

1. We have $a(\rho \circ \sigma)b$
2. It exists a y so that $a(\rho y \sigma)b$ because of Definition 3.13 $a\rho\sigma b \Leftrightarrow \exists y \in Y(a\rho y \wedge y\sigma b)$
3. Now we have the following two cases $y\rho b$ or $a\sigma y$
4. Which we can connect those two terms: $a\sigma y \rho b$

5. The Definition 3.20 says that equivalence relations are symmetric therefore we can switch σ and ρ
Therefore: $a\rho y \sigma b$
6. And because of transitivity from $a\rho y \sigma b$ follows $a(\rho \cup \sigma)b$

Conclusion Therefore $\rho \cup \sigma = \rho \circ \sigma$ must be true.

Correction: Above is not correct, see online solution. □

5.1 Hasse diagram

Hasse diagram \Leftrightarrow poset (partially ordered set) (6)

1. Port represents reflexive
2. Path represents relation
3. edge represents cover
4. Direction on edge (or higher is bigger)

5.2 Hasse diagram definitions

minimal / maximal a is minimal in

$$S \subseteq A \stackrel{\text{def}}{=} a \in S \wedge \forall b \in S \neg(b < a) \quad (\text{nothing is smaller than } a)$$

least / greatest a is least in $S \subseteq A \stackrel{\text{def}}{=} a \in S \wedge \forall b \in S(a \leq b)$

lower bound / upper bound a is a lower bound of $S \subseteq A \stackrel{\text{def}}{=} \forall b \in S(a \leq b)$

greatest lower bound Among all lower bound the greatest

5.3 Countable / Uncountable

Countable $A \exists f : A \rightarrow N$ so that f is injective

Prove: find f so that $f : A \rightarrow N$ and f is injective

Uncountable If it is not injective and bijective.

The following was presented in the exercise on 29. October 2018.

6 Exercise (0.5/3, 0/5)

6.3 The strictly less relation

The relation $\lessdot \cap \neq$ is equivalent to the following:

$$a \lessdot b \wedge a \neq b \quad (7)$$

Now we have to show that it is transitive so we define $a \neq a$ and $c \neq b$. To show that is is transitive we do the following:

$$(a \lessdot b \wedge a \neq b) \wedge (b \lessdot c \wedge b \neq c) \quad (8)$$

The lemma 2.1 says we can use associativity:

$$(a \lessdot b \wedge b \lessdot c) \wedge (a \neq b \wedge b \neq c) \quad (9)$$

Lemma 3.13 says the left side is transitive. Because we defined $c \neq a$ and $c \neq b$ the right hand side can be simplified as well.

$$(a \lessdot c) \wedge (a \neq c) \quad (10)$$

Which is the definition for

$$a < c \quad (11)$$

So we have shown that $<$ is transitive.

6.6 Nonincreasing Functions

We have to prove that the nonincreasing function $f \in \mathbb{N}^{\mathbb{N}}$ are countable.

Base case

Let's define the function $f(k)$ at the point $k \in \mathbb{N}$

For $k = 0$ the function is $f(0)$ and the image is defined for a countable amount of possibilities, because the image is a set \mathbb{N} . Because \mathbb{N} is countable the image of function $f(k)$ for $k = 0$ is countable as well.

Hypothesis

If the image of $f(0)$ is countable the image of function is countable for all k .

Induction Step

If the image of $f(k)$ is countable then the image of $f(k+1)$ is countable as well.

Because we have a nonincreasing function we know that $f(k+1) \lessdot f(k)$

Therefore if the image of $f(k)$ is countable then the image of $f(k+1)$ has to be as well.

Now we know the image values of $f(k)$ are countable at every k .

Second Induction

Now we prove that the product of all those images is countable as well and therefore $f(x)$ is countable.

Base case

Let's define the following for $k = 0$:

$$p(0) = f(0) \quad (12)$$

As shown above $f(0)$ is countable, therefore $p(0)$ is countable as well.

Hypothesis

If $p(k)$ is countable the product is countable for all k .

Induction Step

$$p(k + 1) = f(k + 1) \times p(k) \quad (13)$$

$f(k + 1)$ is countable as shown above and $p(k)$ is countable because of the base case. Corollary 3.18 says: The product of two countable sets is countable. Therefore $p(k + 1)$ is countable as well.

Conclusion

We can conclude that the nonincreasing function $f \in \mathbb{N}^{\mathbb{N}}$ is countable. □

6.1 Modulo

gcd and lcm are always bigger than zero.

$$(gcd(a, b)) = (a, b) \quad (14)$$

$$d = gcd(a, b), \quad (d) = (a, b) \quad (15)$$

Proof 6.1

Prove if $a|bc$, $gcd(a, c) = 1$ then $a|b$

$$gcd(a, c) = 1 \quad (1) = (a, c) \quad (16)$$

$$\exists u, v \in \mathbb{Z} \quad 1 = u \cdot a + v \cdot c \quad (17)$$

$$b = u \cdot ab + v \cdot cb \quad (18)$$

$$a|u \cdot a \cdot b, \quad a|v \cdot c \cdot b \Rightarrow a|b \quad (19)$$

□

Algorithmus 6.1: Extended gcd Algorithm

input: a, b

output: $u, v, \gcd(a, b), \quad \text{sogcd}(a, b) = au + bv$

Table 10:

s_1	u_1	v_1	s_2	u_2	v_2	q	s'_1	u'_1	v'_1	s'_2	u'_2	v'_2
18	1	0	15	0	1	1	15	0	1	3	1	-1
15	0	1	3	1	-1	5	3	1	-1	0	-5	6

The following was presented in the exercise on 5. November 2018.

6.2 Correction 6.6

$$A = \{f : \mathbb{N} \rightarrow \mathbb{N} \mid f \text{ is non-increasing}\} \quad (20)$$

$$1. \quad g : A \rightarrow N^*$$

$$2. \text{ define } g$$

$$3. \text{ } g \text{ is injective}$$

$$M_f = \{m \mid \forall m' > m, f(m') = f(m)\} \quad (21)$$

$$M_f \neq \emptyset \forall f \in A \quad (22)$$

$$m_f^* = \min M_f \quad (23)$$

$$g(f) = (f(0), f(1), \dots, f(m_f^*)) \quad (24)$$

if $g(f_1) = g(f_2)$

$m_{f_1}^* = m_{f_2}^*$, denote it as m if $n \leq m$ $f_1(n) = f_2(n)$

if $n > m$ $f_1(n) = f_1(m) = f_2(m) = f_2(n)$

$$\forall n \leftarrow \mathbb{N} f_1(n) = f_2(n) \quad (25)$$

7 Exercise (3/3, 3/5)

7.3 a)

$$m \equiv_4 n \Rightarrow m = 4q + n \text{ for some } q \quad (26)$$

Because we only care about the last digit (\equiv_{10}), we can do the following simplification.

$$123^m \equiv_{10} 33^n \Leftrightarrow 3^m \equiv_{10} 3^n \quad (27)$$

Insert (26) into the equation.

$$3^{4q+n} \equiv_{10} 3^n \quad (28)$$

$$3^n \cdot 3^{4q} \equiv_{10} 3^n \quad (29)$$

$$81^q \equiv_{10} 1 \Leftrightarrow 1^q \equiv_{10} 1 \quad (30)$$

For any q the above equation is true. Thus is $123^m \equiv_{10} 33^n$, if $m \equiv_4 n$.

7.3 c)

Proof. (\Rightarrow) If x satisfies $ax \equiv_m b$, then $ax = km + b$ for some k . Note that $\gcd(a, m)$ divides both a and m , hence also $ax - km$, which is b . Thus $\gcd(a, m) = b$.

(\Leftarrow) Assume now that $\gcd(a, m) = b$. According to Lemma 4.4 there exist integers u and v such that $ua + vm = \gcd(a, m) = b$. Since $vm \equiv_m 0$ we have $ua \equiv_m b$. Hence $x = u$ is a solution in \mathbb{Z} and thus $x = R_m(u)$ is a solution in \mathbb{Z}_m . \square

7.1 Algebra

7.2 $\langle s, \star \rangle$ Algebraic structure

$$G0: \text{closure } \forall a, b \in s \quad a \star b \in S \quad (Z/\frac{1}{2}) \notin Z(N, " - ") \quad (31)$$

7.3 $\langle m \star \rangle$ Monoid

G0

$$G1 : \exists e \in S \quad e \star a = a \forall a \in S \quad a \star e = a \forall a \in S \quad (32)$$

$$G2 : \forall a, b, c \in S (a \star b) \star c = a \star (b \star c) \quad (33)$$

7.4 Group (G, \star)

G1 $\forall a, b \in S : a \star b \in S$

G2 $\exists e \in S \forall a \in S : e \star a = a \quad a \star e = a$

G3 $\forall a, b, c \in S : a \star (b \star c) = (a \star b) \star c$

G4 $\forall a \in S \exists b \in S \quad a \star b = e \quad (e \text{ in } G)$

7.5 Commutative Group

$$\forall a \in S \forall b \in S : a \star b = b \star a \quad (34)$$

Quiz 8.2 Groups

$$\langle S, \star \rangle \text{ is a group} \quad (35)$$

Prove $a = (a^{-1})^{-1}$ for all $a \in S$

$$a = a \star e = a \star (a^{-1} \star (a^{-1})^{-1}) = (a \star a^{-1}) \star (a^{-1})^{-1} = e \star (a^{-1})^{-1} = (a^{-1})^{-1} \quad (36)$$

7.6 Homomorphism

$$\langle G, t \rangle \quad \langle H, \star \rangle \quad (37)$$

$$h : G \rightarrow H \text{ so that } H(a) \star H(b) = H(a + b) \forall a, b \in G \quad (38)$$

7.7 Isomorphism

7.8 Cyclic group

if $a^n = e$

then $\langle e, a, a^2, a^3, \dots, a^{n-1} \rangle$ is a group.

generated by a. $\langle a \rangle$ group generated by a.

$b \in \langle a \rangle$

$\min\{n | b^n = e\} = \text{ord}(b)$

$b^{\text{ord}(b)} = e$

Correction Exercise 7

if $ua + vb = 1$ prove $\gcd(a, b) = 1$

$d = \text{gec}(a, b)$ and $(a, b) = (d)$

Since $ua + vb = 1 \quad 1 \in (a, b) \text{ so } 1 \in (d)$

so $d|1$ then $d = 1$

$u = d, v = -d$

$$\begin{aligned} a &= 3, b = 2 \\ ua + vb &= d \\ \gcd(a, b) \end{aligned}$$

7.9 Prove Extended gcd Algorithm

If the following two equations are true:

$$s_1 = a \cdot u_1 + b \cdot v_1 \quad (39)$$

$$s_2 = a \cdot u_2 + b \cdot v_2 \quad (40)$$

Then the following has to be true:

$$s'_1 = au_2 + bv_2 = au'_1 + bv'_1 \quad (41)$$

$$s'_2 = s_1 + q \cdot s_2 \quad s'_2 = au'_2 + bv'_2 \quad (42)$$

$$u'_2 = u_1 - qu_2 \quad (43)$$

$$v'_2 = v_1 - qv_2 \quad (44)$$

So our assumption is always true

In the end $s_1 = \gcd(a, b)$, so $\gcd(a, b) = au_1 + bv_1$

7.10 Chinese Remainder Theorem

if $\gcd(n, m) = 1$

then $a \equiv_{mn} b \Leftrightarrow a \equiv_m b$ and $a \equiv_n b$

(\Rightarrow) if $a \equiv_{mn} b$ $b - a = k \cdot kn \quad |k \in \mathbb{Z}$

then $m|b - a$ and $n|b - a$ (\Leftarrow) if $a \equiv_m b$ $b - a = k_1m \quad |k_1 \in \mathbb{Z}$

$a \equiv_n b$ $b - a = k_2n \quad |k_2 \in \mathbb{Z}$

$$k_1m = k_2n \quad (45)$$

$$\gcd(m, n) = 1, \quad n|k, \quad \text{so } k_1pn \quad p \in \mathbb{Z} \quad (46)$$

$$b - a = k_1m = pnm \quad \text{so } nm|b - a \quad (47)$$

Check the master solution for the whole solution.

The following was presented in the exercise on 12. November 2018.

8 Exercise (3/5, 0/3)

8.4 b)

It is to prove that if $S_{a_1}^H \cap S_{a_2}^H \neq \emptyset \Rightarrow S_{a_1}^H = S_{a_2}^H$

Proof.

$$S_{a_1}^H \cap S_{a_2}^H \underset{\substack{\Rightarrow \\ \text{Definition 3.6}}}{\implies} \exists x (x \in S_{a_1}^H \wedge S_{a_2}^H) \quad | \text{ Definition } S_a^H \quad (48)$$

$$\Rightarrow \exists x (x = h_1 * a_1 \wedge x = h_2 * a_2) \quad (49)$$

$$\Rightarrow \exists x (x = h_1 * a_1 = h_2 * a_2) \quad (50)$$

$$\Rightarrow h_1 * a_1 = h_2 * a_2 \quad | * \widehat{a}_1 \quad (51)$$

$$\Rightarrow h_1 * a_1 * \widehat{a}_1 = h_2 * a_2 * \widehat{a}_1 \quad | G3 \quad (52)$$

$$\Rightarrow h_1 * e = h_2 * a_2 * \widehat{a}_1 \quad | G2 \quad (53)$$

$$\Rightarrow h_1 = h_2 * a_2 * \widehat{a}_1 \quad | * \widehat{h}_2 \quad (54)$$

$$\Rightarrow \widehat{h}_2 * h_1 = \widehat{h}_2 * h_2 * a_2 * \widehat{a}_1 \quad | G3 \quad (55)$$

$$\Rightarrow \widehat{h}_2 * h_1 = e * a_2 * \widehat{a}_1 \quad | G2 \quad (56)$$

$$\Rightarrow \widehat{h}_2 * h_1 = a_2 * \widehat{a}_1 \quad (57)$$

Because $\widehat{h}_2 * h_1$ is an element of H we can conclude that $a_2 * \widehat{a}_1$ has to be an element of H aswell.

The following is an element of $S_{a_1}^H$

$$(\widehat{h}_2 * h_1) * a_1 \in S_{a_1}^H \quad (58)$$

Insert equation (10) into (11)

$$\Rightarrow (a_2 * \widehat{a}_1) * a_1 \in S_{a_1}^H \quad | G1 \quad (59)$$

$$\Rightarrow a_2 * (\widehat{a}_1 * a_1) \in S_{a_1}^H \quad | G3 \quad (60)$$

$$\Rightarrow a_2 * e \in S_{a_1}^H \quad | G2 \quad (61)$$

$$\Rightarrow a_2 \in S_{a_1}^H \quad (62)$$

And because a_2 is an element of both $S_{a_1}^H$ and $S_{a_2}^H$ we can conclude

$$\boxed{S_{a_1}^H = S_{a_2}^H} \quad (63)$$

□

8.4 c)

$$\text{Subgroups of } \langle \mathbb{Z}_4, \oplus \rangle : \{\{0\}, \{0, 1, 2, 3\}, \{0, 2\}\} \quad (64)$$

$$\text{Subgroups of } \langle \mathbb{Z}_5, \oplus \rangle : \{\{0\}, \{0, 1, 2, 3, 4\}\} \quad (65)$$

$$\begin{aligned} \text{Subgroups of } \langle \mathbb{Z}_4, \oplus \rangle \times \langle \mathbb{Z}_5, \oplus \rangle : & \{(\{0\}, \{0\}), (\{0\}, \{0, 1, 2, 3, 4\}), \\ & (\{0, 1, 2, 3\}, \{0\}), (\{0, 1, 2, 3\}, \{0, 1, 2, 3, 4\}), (\{0, 2\}, \{0\}), (\{0, 2\}, \{0, 1, 2, 3, 4\})\} \end{aligned} \quad (66)$$

8.1 Diffie Hellman

1. find an a so that $\gcd(a, g) = 1$

2. Compute

$$R_n(a \cdot R_n(g \cdot x) R_n(g \cdot y)) = R^n(g \cdot x \cdot y) = R_n(a \cdot g \times g \cdot y) = R_n(g \times y) \quad (67)$$

Correction: 8.3a)

$$g \star h = e \quad (68)$$

$$\widehat{g} = \widehat{g} \star e = \widehat{g} \star (g \star h) = (\widehat{g} \star g) \star h = e \star h = h \quad (69)$$

Theorem 8.1

Any finite order group G, every element a has a finite order.

Any finite order group G, $\text{ord}(a) = |G|$

Cyclic group of order m is isomorphism to \mathbb{Z}_m

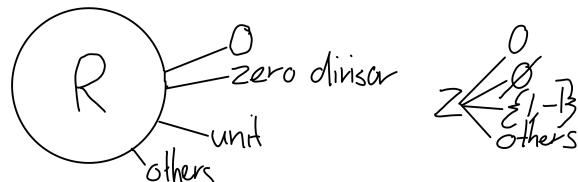
A group with prime order is cyclic

A group with prime order p is isomorphism to \mathbb{Z}_p

set $\mathbb{Z}_m^* = \{k \in \mathbb{Z}_m \mid \gcd(k, m) = 1\}$

8.2 Rings

1. $\langle R; + \rangle$ is a group
2. $\langle R; \star \rangle$ is a monoid
3. $(a + b) \star c$



1. finite Ring has only 0 zerodivisors and units.
2. units of R, denoted as R^* is a group.
3. Rings with no zerodivisor are integral.

$$a \neq 0 \wedge a \times b = a \times c \rightarrow a = c \quad (70)$$

4. Rings with only units and 0 are fields.

The following was presented in the exercise on 19. November 2018.

9 Exercise (2/2, 0/3)

9.3a)

$$\begin{aligned}
 (-a)b &= (-a)b + \{(ab) + [-(ab)]\} && | 0 = -c + c \text{ for all } c \in \mathbb{R} && (71) \\
 &= [(-a)b + (ab)] + [-(ab)] && | \text{Associativity of } + && (72) \\
 &= \{[(-a) + a]b\} + [-(ab)] && | \text{Distributiv law} && (73) \\
 &= 0b + [-(ab)] && | \text{Definition neutral element} && (74) \\
 &= -(ab) && | \text{Lemma 5.17 (i)} && (75)
 \end{aligned}$$

9.3b)

$$\begin{aligned}
 c &:= (-b) && | \text{Auxiliary variable} && (76) \\
 (-a)(-b) &= (-a)c && | \text{Definition c} && (77) \\
 &= -(ac) && | \text{Lemma 5.17 (ii)} && (78) \\
 &= -(ca) && | \text{Commutativity} && (79) \\
 &= (-c)a && | \text{Lemma 5.17 (ii)} && (80) \\
 &= [(-b)]a && | \text{Definition c} && (81) \\
 &= [b]a && | \text{Defintion inverse} && (82) \\
 &= ab && | \text{Commutativity} && (83)
 \end{aligned}$$

Table 11:

	0	Units	Zerodiosor	Others
zero ring	✓	X	X	X
non tirval ring	✓	✓	OK	OK
integral domain	✓	✓	X	OK
Field	✓	✓	X	X
$\mathbb{R}^*(group)$	X	✓	X	X

Correction: 9.1.2

$$\mathbb{Z}_{mn} \sim \mathbb{Z}_m^* \times \mathbb{Z}_n^* \quad (84)$$

Define f $\mathbb{Z}_{mn} \sim \mathbb{Z}_m^* \times \mathbb{Z}_n^*$ as $f(x) = (R_m(x), R_n(x))$

1. f is well defined $\forall x \in \mathbb{Z}_{mn}^*$
2. f is injective
3. f is surjective

4. f is homomorphism

$$\textcircled{1} \text{ if } \gcd(x, mn) = 1 \quad (85)$$

$$\text{Assume } d = \gcd(x, m) \quad (86)$$

$$d|x, d|m \Rightarrow d|mn \quad (87)$$

$$d|\gcd(x, mn) \quad (88)$$

$$d|1 \text{ so } d = 1 \quad (89)$$

$$\gcd(R_n(x), m) = 1, \gcd(R_n(x), n) = 1 \quad (90)$$

$$\textcircled{2} \text{ if } R_m(x) = R_m(y), R_n(x) = R_n(y) \quad x, y \in \mathbb{Z}_{mn}^* \quad (91)$$

$$x \equiv y(m + dn) \quad (92)$$

$$x \equiv \quad (93)$$

$$\textcircled{3} |\mathbb{Z}_{mn}^*| = |\mathbb{Z}_m^* \times \mathbb{Z}_n^*| \quad (94)$$

$$\text{and } f \text{ is injective, so } f \text{ is surjective} \quad (95)$$

$$\textcircled{4} \forall x, y \in \mathbb{Z}_{mn}^* \quad (96)$$

$$f(x * y) = f(x) * f(y) \quad (97)$$

$$\Leftrightarrow (R_m(R_{mn}(xy)), R_n(R_{mn}(xy))) = (R_m(R_m(x) * R_m(y)), R_n(R_n(x) * R_n(y))) \quad (98)$$

$$\Leftrightarrow R_m(R_{mn}(xy)) = R_m(R_m(x)R_m(y)) \quad (99)$$

Correction: 9.2

$$c_1 = m^3(\bmod n_1) \quad (100)$$

$$c_2 = m^3(\bmod n_2) \quad (101)$$

$$c_3 = m^3(\bmod n_3) \quad (102)$$

$$(103)$$

find m if $d_{12} \neq 1$ $d_{12}|n_1$ $d_{12}|n_2$

First compute the following

$$d_{12} = \gcd(n_1, n_2) \quad (104)$$

$$d_{23} = \gcd(n_2, n_3) \quad (105)$$

$$d_{13} = \gcd(n_1, n_3) \quad (106)$$

$$(107)$$

So we know $p_1 q_1 = n_1$

compute d so that

$$ed \equiv 1(\bmod(p_i - 1)(q_i - 1)) \quad (108)$$

compute

$$c_i^d(\bmod n_i) = m \quad (109)$$

if $d_{12} = d_{23} = d_{31} = 1$

$$x = c_1(\bmod n_1) \quad (110)$$

$$x = c_2(\bmod n_2) \quad (111)$$

$$x = c_3(\bmod n_3) \quad (112)$$

Compute m^3

Discussion Lecture

$$F[x] = \{a_k x^k + a_{k-1} x^{k-1} + \dots + a_1 x + a_0 \mid a_i \in R\} \quad (113)$$

$\langle R[x], +, \star \rangle$ is a Ring if R is a ring. (114)

Table 12:

Z	F[x]
$a b$ if $\exists c$ so that $ac = b$ $d = \gcd(a, b)$ $d > 0$ p is prime if $p \neq 0$ $a p \Leftrightarrow a = 1, -1, p, -p$	$a(x) b(x)$ if $\exists c(x)$ so that $a(x)c(x) = bc(x)$ $d(x) = \gcd(a(x), b(x))$ $d(x)$ is monic $p(x)$ is irreducible if $p(x) \neq 0$ $a(x) p(x) \Leftrightarrow a(x)$ is constant $\Leftrightarrow a(x)$ is constant multiple $f p(x)$
$b = a \cdot q + r$ $0 \geq r < a $	$b(x) = a(x)q(x) + r(x)$ $\deg(r(x)) < \deg(a(x))$ $\deg(a(x)) \stackrel{\text{def}}{=} \max\{k \mid a_k \neq 0\}$ $\deg(0) = -\infty$ $\deg(a(x)b(x)) = \deg(a(x)) + \deg(b(x))$
$b \equiv r \pmod{a}$ $b \equiv_a r$ $\mathbb{Z}_m = \{x \in \mathbb{Z} \mid 0 \leq x < m\}$	$b(x) \equiv_{a(x)} r(x)$ $a(x) (b(x) - r(x))$ $F[x]_{\min} = \{a(x) \in F[x] \mid \deg(a(x)) < m(x)\}$ $ F[x]_{\min} = F ^k$ if F is finite and $\deg(m(x)) = k$
$ax \equiv_m 1$ has unique solution in \mathbb{Z}_m iff $\gcd(a, m) = 1$ \mathbb{Z}_m is a ring \mathbb{Z}_m^* is a group \mathbb{Z}_p is a field iff p is prime. We called that $GF(p)$ $\mathbb{Z}^* = \mathbb{Z}_p \setminus \{0\}$ iff p is prime	$a(x)X(x) \equiv_{m(x)} 1$ has unique solution in $F[x]_{\min}$ iff $\gcd(a(x), m(x)) = 1$ $F[x]_{m(x)}$ is a ring, $F[x]_{m(x)}^*$ is a group $F[x]_{m(x)}$ is a field iff $m(x)$ is irreducible.

Example 2.

$$GF(2) = \{0, 1\} \quad (115)$$

$$GF(2)[x] = \{a_k x^k + \dots + a_1 x + a_0 \mid a_i \in GF(2)\} \quad (116)$$

$$GF(2)[x]_{x^2+x+1} = \{0, 1, x, x+1\} \quad (117)$$

The following was presented in the exercise on 26. November 2018.

10 Exercise (2/4, 2/4)

10.1 a)

Proof. To show: a is a unit $\Rightarrow (a) = R$

By definition of a unit $a \cdot v = 1$ for some $v \in R$. Therefore $1 \in (a)$ and $1 \in R$.

$$\Rightarrow \forall r \in R : 1 \cdot r = r \in (a) \wedge 1 \cdot r = r \in R \quad (118)$$

Hence if a is a unit, then $R \subseteq (a)$

The definition of (a) applies $a \cdot r = b$ for some $b \in (a)$

The definition of a ring is that it is closed under its operations, therefore $b \in R$

$$\Rightarrow (a) \subseteq R$$

Hence if a is a unit, then $(a) = R$

To show: $(a) = R \Rightarrow a$ is a unit

By definition of a ring: $1 \in R$. Because of the assumption the neutral element must be part of the ideal (a) as well. Therefore one can create the neutral element with $(a) := \{a \cdot r | r \in R\}$ for some r . Since $a \cdot r = 1$ and R is a commutative group: $a \cdot r = r \cdot a = 1$. This is the definition of a unit. Hence $(a) = R \Rightarrow a$ is a unit. \square

10.1 b)

Proof by Contradiction. To show: D is an integral domain $\Rightarrow D[x]$ is an integral domain.

Let's assume that $D[x]$ is not an integral domain.

Therefore the negation of the definition of an integral domain must be true.

$$\Rightarrow \overline{\forall a(x) \forall b(x) (a(x) \cdot b(x) = 0 \rightarrow a(x) = 0 \vee b(x) = 0)} \quad (119)$$

$$\Rightarrow \exists a(x) \exists b(x) (a(x) \cdot b(x) = 0 \wedge \neg(a(x) = 0) \wedge \neg(b(x) = 0)) \quad |\text{Definitions of Negations} \quad (120)$$

One can get D if $\deg(a(x)) = 0$ and $\deg(b(x)) = 0$

$$\Rightarrow \exists a \exists b (a \cdot b = 0 \wedge \neg(a = 0) \wedge \neg(b = 0)) \quad (121)$$

Which is a contradiction to D is an integral domain.

$$\Rightarrow \forall a(x) \forall b(x) (a(x) \cdot b(x) = 0 \rightarrow a(x) = 0 \vee b(x) = 0) \quad (122)$$

Hence D is an integral domain $\Rightarrow D[x]$ is an integral domain. \square

10.1 Distance

Table 13:

Z	$F[x]$
Z_m	$F[x]_{m(x)}$
Z_p	$GF(p)$

Hamming Distance Distance of two strings

Minimal Distance Distance of code

Correction: 10.2 b)

1. $\forall a \in F \setminus \{0\} x - a | x^{q-1} - 1$
2. so $\prod_{a \in F \setminus \{0\}} (x - a) | x^{q-1} - 1$
3. $\exists b(x) \in F(x)$ so that $x^{q-1} - 1 = b(x) \prod_{a \in F \setminus \{0\}} (xa)$
4. Prove $b(x) = 1$

10.2 Logic

10.3 Proof System

S all Statements

P all Proofs

$$\tau : S \rightarrow \{0, 1\}$$

$$\phi : S \times P \rightarrow \{0, 1\}$$

$\langle S, P, \tau, \phi \rangle$ must be sound and complete.

10.4 Syntax

1. alphabet Λ
2. Formula F
3. set of Formulas M

10.5 Semantics

1. Free Symbols
2. Universe
3. interpretation (Assign symbols to values)
4. suitable interpretation for F for M (Give values to free symbols)
5. model for F for M (Give values to formulas F, $A(F) = 1$)

10.6 Propositional Logic

Syntax $A_0, A_1, A_2 \forall i \in \mathbb{N}, \wedge, \vee, \neg$

A_0, A_1, A_2 are atomic formulas if F and G are formulas then $F \vee G, F \wedge G, \neg F$ are formulas as well.

Table 14:

$$\begin{array}{c} m \vDash F & G \vDash F \\ \forall A \vDash M, A \vDash F & \forall A \vDash G, A \vDash F \end{array}$$

The following was presented in the exercise on 03. December 2018.

11 Exercise (3.5/4, 1.5/4)

11.3 a)

$$F = (\neg A \vee B) \wedge (B \rightarrow (\neg C \wedge \neg A)) \wedge (A \vee C) \quad (123)$$

$$= (\neg A \vee B) \wedge (\neg B \vee \neg C) \wedge (\neg B \vee \neg A) \wedge (A \vee C) \quad (124)$$

Table 15:

A	B	C	$\neg A \vee B$	$\neg B \vee \neg C$	$\neg B \vee \neg A$	$A \vee C$	F
0	0	0	1	1	1	0	0
0	0	1	1	1	1	1	1
0	1	0	1	1	1	0	0
0	1	1	1	0	1	1	0
1	0	0	0	1	1	1	0
1	0	1	0	1	1	1	0
1	1	0	1	1	0	1	0
1	1	1	1	0	0	1	0

The model for F must contain the following set $\{A = 0, B = 0, C = 1\}$.

$$G = \neg(A \rightarrow B) \vee (C \rightarrow A) = (A \wedge \neg B) \vee (\neg C \vee A) \quad (125)$$

Table 16:

A	B	C	$A \wedge \neg B$	$\neg C \vee A$	G
0	0	0	0	1	1
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	1	0	1	1

Each model for F must contain the following sets $\{\{A = 0, B = 0, C = 0\}, \{A = 0, B = 1, C = 0\}, \{A = 1, B = 0, C = 0\}, A = 1, B = 0, C = 1\}, \{A = 1, B = 1, C = 0\}, \{A = 1, B = 1, C = 1\}$.

Hence F and G are whether equivalent nor a logical consequence of each other.

11.3 b)

$(\Rightarrow) \{F_1, \dots, F_k\} \vDash G \Rightarrow ((F_1 \wedge \dots \wedge F_k) \rightarrow G) \text{ is a tautology}$

Because of Definition 6.12, if a term is a model of F then it is a model of G as well. Therefore G must always be true. Thus it is a model for $\{F_1, \dots, F_k\}$ then it must be a model for $(F_1 \wedge \dots \wedge F_k)$ as well.

$(\Leftarrow) ((F_1 \wedge \dots \wedge F_k) \rightarrow G) \text{ is a tautology} \Rightarrow \{F_1, \dots, F_k\} \vDash G$

If G is true If G is true then it doesn't matter what the truth values of all the F_i ($1 \leq i \leq k$) are. Therefore the expression on the right side must always be true as well.

If G is false Because it is a tautology at least one of the F_i ($1 \leq i \leq k$) has to be wrong. ($\neg F_1 \vee \dots \vee \neg F_k \vee G$). Therefore $\{F_1, \dots, F_k\} \vDash G$ because one of the elements F_i is false and G is false.

11.1 Aussagenlogik

11.2 Syntax

1. Form F_i , $i \in \mathbb{N}$: Atomare Formel
2. Falls F , G Formeln sind, dann auch $(F \wedge G)$, $(F \vee G)$, $\neg F$

11.3 Semantik

$$\alpha : M \rightarrow \{0, 1\} \text{ Belegung} \quad (126)$$

$$\mathcal{A}(A_i) := \underbrace{\alpha(A_i)}_{\text{atomare Formel}} \quad (127)$$

Example 3.

$$A \rightarrow B \equiv \neg A \vee B \quad (128)$$

$$F, G \text{ sind Formeln, dann ist auch } F \rightarrow G \text{ eine Formel} \quad (129)$$

$$\mathcal{A}(F \rightarrow G) = 1 \Leftrightarrow \mathcal{A}(F) = 0 \text{ oder } \mathcal{A}(G) = 1 \Leftrightarrow \mathcal{A}(\neg F \vee G) = 1 \quad (130)$$

11.4 CNF

$$(L_{11} \vee L_{12} \vee \dots) \wedge (L_{21} \vee L_{22} \vee \dots) \wedge \dots \quad (131)$$

Table 17:

A	B	$\neg B \rightarrow A$	F	$A \vee B$	$\neg A \vee B$
0	0	0	0	0	1
0	1	1	1	1	1
1	0	1	0	1	0
1	1	1	1	1	1

Example 4.

$$F := B \wedge (\neg B \rightarrow A) \quad (132)$$

$$\Rightarrow (A \vee B) \wedge (\neg A \vee B) \quad (133)$$

11.5 DNF

$$(L_{11} \wedge L_{12} \wedge \dots) \vee (L_{21} \wedge L_{22} \wedge \dots) \vee \dots \quad (134)$$

Example 5. Gleiche Wahrheitstabelle wie bei CNF

$$\Rightarrow (\neg A \wedge B) \vee (A \wedge B) \quad (135)$$

11.6 $F \vDash G$

Für jede Interpretation die suitable für F ist und suitable für G ist, dann gilt falls es ein Modell für F ist dann ist es auch ein Modell für G.

11.7 Prädikatenlogik

Muss wieder am Anfang Syntax und Semantics definieren.

Variabel Symbol $x_i, i \in \mathbb{N}$

Funktionssymbol $f_i^k, i, k \in \mathbb{N}$

$\Rightarrow k = 0$: Konstante

Term 1. Variable ist ein Term. 2. Für Terme t_1, \dots, t_m ist $f_i^{(k)}(t_1, \dots, t_k)$ ein Term

Prädikantensymbol $P_i^{(k)}$

Formel 1. $P_i^{(k)}(t_1, \dots, t_k)$ Atomare Formel. 2. Falls F, G Formeln, dann auch $(F \wedge G), (F \vee G), \neg F, \forall_i F, \exists x_i F$ Formeln

Semantics • $\mathcal{A} = \{U, \phi, \psi, \xi\}$

- U : Universum $U \neq \emptyset$
- ϕ Wertet Funktionen aus:
 $\psi(f) : U^k \rightarrow U$
- ψ wertet Prädikate aus
 $\psi : \text{wertet Prädikate aus}$
 $\psi(p) : U^k \rightarrow \{0, 1\}$
- ξ Wertet Variablen aus
 $\xi(u) \in U$

Structure

11.8 Auswertung

Auswertung **Term** 1. Falls t eine Variable $A(t) = \xi(k)$ 2. Falls t Form $f^k(t_1, \dots, t_k)$

$$\mathcal{A}(t) = \phi(f)(A(t_1), \dots, A(t_k))$$

Prädikate 1. $A(P(t_1, \dots, t_k)) = \psi(p)(A(t_1), \dots, A(t_k))$

$$\mathcal{A}(\forall x G) = \begin{cases} 1, & A_{[x \rightarrow u]}(G) = 1 \text{ für alle } u \in U \\ 0, & \text{sonst} \end{cases} \quad (136)$$

$$\mathcal{A}(\exists x G) = \begin{cases} 1, & A_{[x \rightarrow u]}(G) = 1 \text{ für alle } 1 \text{ } u \in U \\ 0, & \text{sonst} \end{cases} \quad (137)$$

Example 6.

$$\forall x \exists y P(x) \wedge \forall x (\exists x Q(x) \wedge R(x)) \wedge \underbrace{S(x)}_{x \text{ ist frei}} \quad (138)$$

Alle anderen x sind gebunden.

Proof 11.1: $\neg(\forall x F) \equiv \exists x \neg F$

Sei A Struktur suitable für beide und $A(\neg(\forall x F)) = 1$

$$\Rightarrow A(\forall x F) = 0 \quad (139)$$

$$\Rightarrow A_{[x \rightarrow u]}(F) = 0 \text{ für mindestens } 1 \text{ } u \in U \quad (140)$$

$$\Rightarrow A_{[x \rightarrow u]}(\neg F) = 0 \text{ für mindestens } 1 \text{ } u \in U \quad (141)$$

$$\Rightarrow A(\exists x \neg F) = 1 \quad (142)$$

Sei $A(\neg(\forall x F)) = 0$

$$\Rightarrow A(\forall x F) = 1 \quad (143)$$

$$\Rightarrow A_{[x \rightarrow u]}(F) = 1 \text{ für alle } u \in U \quad (144)$$

$$\Rightarrow A_{[x \rightarrow u]}(\neg F) = 0 \quad (145)$$

$$\Rightarrow A(\exists x \neg F) = 0 \quad (146)$$

□

Hausaufgabe 10,1 a)

$$(a) := \{a \cdot r \mid r \in R\} \quad (147)$$

$$(a) = R \Leftrightarrow a \text{ unit} \quad (148)$$

$$(\Rightarrow) 1 \in (a) \quad (149)$$

$$\Rightarrow \exists r \in R (a \cdot r = 1) \quad (150)$$

$$\Rightarrow a \text{ a is a unit} \quad (151)$$

$$(\Leftarrow) (a) \subseteq R \checkmark \quad (152)$$

$$R \subseteq (a) : \quad (153)$$

$$\exists a^{-1} \in R (a \cdot a^{-1} = 1) \quad (154)$$

$$s \in R \quad s = 1 \cdot s = (a \cdot a^{-1}) \cdot s \quad (155)$$

$$= a \underbrace{(a^{-1} \cdot s)}_R \quad \Rightarrow S \in (a) \quad (156)$$

Hausaufgabe 10,1 b)

D, Zeige, dass $D[x]$ auch in B

$$a(x), b(x) \quad (157)$$

$$\deg(a(x)) = d \quad (158)$$

$$\deg(b(x)) = d \quad (159)$$

$$\deg(a(x)b(x)) = d + d' \quad (160)$$

$$a(x) = a_d x^d + a_{d-1} x^{d-1} + \dots \quad (161)$$

$$b(x) = b_{d'} x^{d'} + \dots \quad (162)$$

Hausaufgabe 11,1

$$a(y) = xy^3 + xy^2 + (x+1)y + x \in GF(2)[x]_{x^2+x+1}[y] \quad (163)$$

$$a(0) = x \quad (164)$$

$$a(1) = x + x + x + 1 + x = 1 \quad (165)$$

$$a(x) = x^4 + x^3 + x^2 + x + x = x^2(x^2 + x + 1) = 0 \Rightarrow (y - x) = (y + x) \quad (166)$$

Note 2. Bei einem Körper muss man immer Modulo rechnen können, sonst kann man zum Beispiel kein Inverses bestimmen. Mit der Polynomdivision bekommt man dann $xy^2 + y + 1$. Nun muss man noch auf eine doppelte Nullstelle überprüfen.

11.2 a)

$$c \in \{0, 1\}^n \quad B_r(c) := \{c' \in \{0, 1\}^n \mid d(c, c') \leq r\} \quad (167)$$

$\sum_{i=0}^r \binom{n}{i}$ viele unterschiedliche Stellen.

11.2 b)

$$B_t(c) \cap B_t(c') = \emptyset \quad (168)$$

$$\sum_c \in C |B_t(c)| = m \cdot b_t \Rightarrow m \leq \frac{2^n}{b_t} \quad (169)$$

The following was presented in the exercise on 10. December 2018.

12 Exercise (3/3, 5/5)

12.4 a)

Table 18:

A	B	C	$A \rightarrow C$	$\neg(A \rightarrow C)$	$A \rightarrow B$	$(\neg(A \rightarrow C)) \leftrightarrow (A \rightarrow B)$
0	0	0	1	0	1	0
0	0	1	1	0	1	0
0	1	0	1	0	1	0
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	1	1	0	1	1
1	1	0	0	1	1	1
1	1	1	1	0	1	0

DNF

$$(A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \quad (170)$$

CNF

$$(A \vee B \vee C) \wedge (A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge (\neg A \vee B \vee C) \wedge (\neg A \vee \neg B \vee \neg C) \quad (171)$$

12.6 b) i)

$$\mathcal{A} = \{U, \phi, \psi, \xi\}$$

$$U^{\mathcal{A}} = \{0, 1\}$$

$$f^{\mathcal{A}} = \{\}$$

$$P^{\mathcal{A}} = \{\}$$

$$x^{\mathcal{A}} = \{\}$$

12.6 b) ii)

$$\mathcal{A} = \{U, \phi, \psi, \xi\}$$

$$U^{\mathcal{A}} = \{0, 1\}$$

$$f^{\mathcal{A}} = \{\}$$

$$P^{\mathcal{A}}(x, y) = 0$$

$$x^{\mathcal{A}} = \{\}$$

12.6 b) iii)

$$\mathcal{A} = \{U, \phi, \psi, \xi\}$$

$$U^{\mathcal{A}} = \mathbb{Z}_7$$

$$f^{\mathcal{A}} = \{\}$$

$$P^{\mathcal{A}}(x, y) = (x + 1 = y)$$

$$x^{\mathcal{A}} = \{\}$$

12.1 Ersetzen

$$A(\forall x F) = \begin{cases} 1, & A_{[x \rightarrow u]}(F) = 1 \text{ für alle } u \in U \\ 0, & ; \text{sonst} \end{cases} \quad (172)$$

$$A(\exists x F) = \begin{cases} 1, & A_{[x \rightarrow u]}(F) = 1 \text{ für } 1 u \in U \\ 0, & ; \text{sonst} \end{cases} \quad (173)$$

Definition 1. 6.33 im Skript

$F[x/t]$: Ersetze alle freien Vorkommen von x mit Term t

$$U^A = \mathbb{N} \quad F[x/f(y)] \quad (174)$$

$$A_{[x \rightarrow u]} \quad u \in U \quad (175)$$

$$\xi(x) = u \quad (176)$$

$$A(x) = u \quad (177)$$

$$A = (U, \phi, \psi, \xi) \quad (178)$$

Lemma 12.1: 6.10 im Skript

$$\forall xG \equiv \forall yG[x/y]G \text{ enthält kein } y \quad (179)$$

$$\exists xG \equiv \exists yG[x/y] \quad (180)$$

12.2 Rectified Formulas

Keine Variable tritt gebunden und frei auf.

Dieselbe Variable kommt nicht 2 Mal an anderen Quantoren gebunden vor.

Lemma 12.2: 6.11 im Skript

$$\forall xF \vDash F[x/t] \quad (181)$$

Proof 12.1

$$\mathcal{U}, \phi, \psi, \xi \quad (182)$$

$$A(\forall xF) = 1 \quad (183)$$

$$\Rightarrow A_{[x \rightarrow u]}(F) = 1 \text{ für alle } u \in U \quad (184)$$

$$A(t) = u', \quad u' \in U \quad (185)$$

$$A_{[x \rightarrow u']}(F) = 1 \Rightarrow A(F[x/t]) = 1 \quad (186)$$

□

12.3 Pränex Form

$$Q_1 x_1 Q_2 x_2 \dots Q_n x_n F \quad (187)$$

Q ist entweder \forall oder \exists

1. Formel rectified machen
2. Transformieren mit L 6.8

$$\neg(\forall xP(x, y, z) \vee \exists yQ(x, y)) \underset{\substack{\equiv \\ rename}}{\sim} \neg(\forall aP(a, y, z) \vee \exists bQ(x, b)) \quad (188)$$

$$\equiv \neg(\forall a(P(a, y, z) \vee \exists bQ(x, b))) \quad (189)$$

Zuerst Kommutativität anwenden

$$\equiv \neg(\forall a(\exists bQ(x, b) \vee P(a, y, z))) \quad (190)$$

$$\underset{10}{\equiv} \neg(\forall a(\exists b(Q(x, b) \vee P(a, y, z)))) \quad (191)$$

$$\underbrace{\equiv}_{1} \exists a \neg \exists b (Q(x, b) \vee P(a, y, z)) \equiv \exists a \exists b \neg (Q(x, b) \vee P(a, y, z)) \quad (192)$$

Theorem 12.1: 6.13 im Skript

$$\neg \exists x \forall y (P(y, x) \leftrightarrow \neg P(y, y)) \quad (193)$$

$$\equiv \forall x \forall y \neg (P(y, x) \leftrightarrow \neg P(y, y)) \quad (194)$$

$$\equiv \forall x \forall y (P(y, x) \leftrightarrow P(y, y)) \quad (195)$$

$$\text{Setze } y = 0 \quad P(x, x) \leftrightarrow P(x, x) \quad (196)$$

12.4 Uncomputable Functions

$$\mathbb{N} \rightarrow \{0, 1\} \quad (197)$$

$$P^A(y.x) = 1 \Leftrightarrow \text{x-tes Programm gibt für Input y, den Wert 1 aus.} \quad (198)$$

$$\neg \exists x \forall y (P(y, x) \leftrightarrow \neg P(y, y)) \quad (199)$$

Table 19:

	y_1	y_2	$y_3 \dots$
x_1	α_{11}	α_{12}	α_{13}
x_2	α_{21}	α_{22}	α_{23}
x_3	α_{31}	α_{32}	α_{33}
\hat{x}	$\overline{\alpha_{11}}$	$\overline{\alpha_{12}}$	$\overline{\alpha_{13}}$

Nachbesprechung

11.3 b)

\Rightarrow Wähle Interpretation A

Angenommen:

$$A(\neg(F_1 \wedge \dots \wedge F_k) \vee G) = 0 \quad (200)$$

$$\Rightarrow A(G) = 0 \& A(F_1 \wedge \dots \wedge F_k) = 1 \quad (201)$$

$$\Rightarrow A(G) = 0 \& A(F_1) = A(F_2) = \dots = 1 \quad (202)$$

$\Leftarrow (F_1 \wedge \dots \wedge F_k) \rightarrow G \equiv T$ Wähle A

Zeige dass es gleich Null ist

Vorbesprechung 12

$$F := A, G := A \wedge \neg A \quad (203)$$

F erfüllt

$F \rightarrow G$ ist auch erfüllt

$$A(A) = 0$$

12.4 b)

$$(A \wedge \neg B) \vee (\neg A \wedge (C \wedge D)) \quad (204)$$

$$\equiv ((A \wedge \neg B) \vee \neg A) \wedge ((A \wedge \neg B) \vee (C \wedge D)) \quad (205)$$

$$\equiv (\neg A \vee (A \wedge \neg B)) \wedge ((A \wedge \neg B) \vee (C \wedge D)) \quad (206)$$

$$\equiv ((\neg A \vee A) \wedge (\neg A \vee \neg B)) \wedge ((A \wedge \neg B) \vee C) \wedge ((A \wedge \neg B) \vee D) \quad (207)$$

F valid $\Leftrightarrow \forall x_1 \forall x_2 \dots \forall x_n F$ ist auch valid

$$A = (U, \psi, \phi, \xi)$$

$$A_{[u_1, u_2, \dots, u_n]}$$

$G := \forall x_1 \forall x_2 \dots \forall x_n F$ is valid

\Leftrightarrow Für jede Struktur A suitable für G gilt: $A(G) = 1 \Leftrightarrow A_{[u_1, \dots, u_n]}(F) = 1$ für alle $u_i \in U$

$\Leftrightarrow A(F) = 1$ für jede suitable Struktur

The following was presented in the exercise on 17. December 2018.

Example 7.

$$\{\{A_1, A_2, A_3\}, \{A_1, \neg A_2, \underbrace{\neg A'_3}_{=0}\}\} \quad (208)$$

$$K = \{\{A_1, \neg A_2\}\} \quad (209)$$

Angenommen: K ist unerfüllbar.

$$\Rightarrow K_0 \equiv \perp, \quad K_1 \equiv \perp \quad (210)$$

$$\Rightarrow K_0 \vdash_R \emptyset, \quad K_1 \vdash_R \emptyset \quad (211)$$

K wie auf K_0 :

1. Fall 1 $K \vdash_{Res} \emptyset$

2. Fall 2 $K \vdash_{Res} \{A_{n+1}\}$

K wie auf K_1 :

1. Fall 1: $K \vdash_{Res} \emptyset$

2. Fall 2: $K \vdash_{Res} \{\neg A_{n+1}\}$

Auf beide Fall 2 kann nun noch einmal \vdash_{Res} angewendet werden und man erhält \emptyset

Example 8.

$$F = (A \vee C) \wedge (\neg D \vee \neg A) \wedge B \wedge (\neg B \vee \neg C) \wedge \neg C \wedge (C \vee D) \quad (212)$$

$$\{\{A, C\}, \{\neg D, \neg A\}, \{B\}, \{\neg B, C\}, \{\neg C\}, \{C, D\}\} \quad (213)$$

$$\{C, \neg D\} \quad \{C\} \quad \{D\} \quad (214)$$

$$\{\neg D\} \quad \{D\} \Rightarrow \emptyset \quad (215)$$

Quiz

a)

$$\{\{A\}, \{\neg A, B\}, \{\neg B\}\} \quad (216)$$

$$\{B\} \quad \{\neg B\} \Rightarrow \emptyset \quad (217)$$

b)

$$(A \vee C) \rightarrow (B \vee D) \quad (218)$$

$$(A \vee C) \wedge \neg B \wedge \neg D \quad (219)$$

$$\{A, C\}, \{\neg B\}, \{\neg D\} \quad (220)$$

$$A \rightarrow B \equiv \neg A \vee D \quad (221)$$

$$C \rightarrow D \equiv \neg C \vee D \quad (222)$$

$$\{A, C\}, \{\neg B\}, \{\neg D\}, \{\neg A, B\} \quad (223)$$

$$\{\neg A\} \quad \{\neg C, D\} \quad (224)$$

$$\{C\} \quad \{\neg C\} \Rightarrow \emptyset \quad (225)$$

Sets, Relations and Functions

03.1 $A=B \Leftrightarrow \forall x(x \in A \Leftrightarrow x \in B)$

03.2 Number of elements \Leftrightarrow Cardinality $\Rightarrow |A|$

03.1 For any a and b $\{a\} = \{b\} \Rightarrow a = b$

D 3.3 $A \subseteq B \Leftrightarrow \forall x(x \in A \rightarrow x \in B)$

$A = B \Leftrightarrow (A \subseteq B) \wedge (B \subseteq A)$ Follows from D.3.1

0.3.4 Empty set $\emptyset \Leftrightarrow \nexists x(x \in \emptyset)$

L 3.2 $\forall A (\emptyset \subseteq A)$

Operation	Notation	Definition
Power set	$P(A) := \{\text{S} \mid S \subseteq A\}$	03.5
union	$A \cup B := \{x \mid x \in A \vee x \in B\}$	03.6
intersection	$A \cap B := \{x \mid x \in A \wedge x \in B\}$	03.6
difference	$A \setminus B := \{x \mid x \in A \wedge x \notin B\}$	03.8
cartesian product	$A \times B$	03.9
Complement	$\overline{A} := \{x \in U \mid x \notin A\}$	03.7
Cartesian product for finite sets	$ A \times B = A \cdot B $	
Russell paradox	$R = \{x \mid x \notin x\}$	

03.10 A (binary) relation p from a set A to a set B is a subset of $A \times B$

if $A = B$ then p is called a relation on A ($(ab) \in p \Leftrightarrow a \sim b$)

D 3.20 Equivalence Relation
refl, sym and trans

Property	Definition
reflexivity	$\forall a (a \sim a)$
irreflexivity	$\forall a (\neg(a \sim a))$
symmetry	$\forall a \forall b ((a \sim b) \rightarrow (b \sim a))$
antisymmetry	$\forall a \forall b ((a \sim b) \wedge (b \sim a) \rightarrow a = b)$
transitivity	$\forall a \forall b \forall c ((a \sim b) \wedge (b \sim c) \rightarrow a \sim c)$

D 3.12 Inverse relation $\forall a \in A \forall b \in B (a \sim b \Leftrightarrow b \sim a)$

D 3.13 $\forall p \forall q \exists b \in B (a \sim b \wedge b \sim c) = (p \circ q)$

L 3.5 composition of relations is associative $p \circ (q \circ r) = (p \circ q) \circ r$

L 3.7 p is transitive iff $p \subseteq p^2$

03.19 transitive closure p^* = $\bigcup_{n=1}^{\infty} p^n$

03.27 b covers a if $a \sim b$ and $\forall c \in C$ with $a \sim c$ and $c \sim b$	03.29 a is minimal if no $b \sim a$ / same for maximal
2. a is least lower bound if $a \leq b$ for all b / same for greatest	3. $a \in S$ is greatest lower bound if a is greatest of all lower bound / least upper bound if it is totally ordered (any two elements being comparable) and if every non-empty subset of S has a least element
03.30 Posed is well-ordered if it is totally ordered (any two elements being comparable)	
03.31 meet-greatest lower bound / join least upper bound	
03.32 A poset which every pair of elements has a meet and join is a lattice	
03.33 A function $f: A \rightarrow B$ from a domain A to a codomain B is a relation $A \times B$:	
1. $\forall a \in A \exists b \in B a \sim b$ (f is totally defined)	
2. $\forall a \in A \forall b_1, b_2 \in B (a \sim b_1 \wedge a \sim b_2 \rightarrow b_1 = b_2)$ (f is well-defined)	
03.34 For $f: S \rightarrow B$ and subset S_0 of S the range of f is $f(S) = \{f(a) \mid a \in S\}$	
03.40 Injektive, surjektive, bijektive both collisions / collisions, Every value in codomain taken, bijective both	
03.42 the compositions of functions $g \circ f := (g \circ f)(a) = g(f(a))$	
(i) f dominates A: $A \subseteq B$ if $A \sim C$ for $C \subseteq B$	
(ii) A countable iff $A \sim \mathbb{N}$ otherwise uncountable	
T 3.15 A is countable iff its finite or $A \sim \mathbb{N}$ C 3.19, Q is cont.	
03.47 equinumerous: A and B	
03.16 set $\{C_1, 0, 1, 00, 01, \dots\}$ of finite binary sequences is countable	
C 3.18 Cartesian product of two count. sets is count. $A \times B \subseteq \mathbb{N} \Rightarrow A \times B \subseteq \mathbb{N}$	
T 3.20 \mathbb{N}^n of n-tuples over A is count. (i) union of count. sets is count. (ii) sequences of elements from A is count.	
T 3.21 Set $\mathbb{N}^{<\mathbb{N}}$ is uncountable (Cantor diagonalization argument)	
C 3.22 There are uncountable functions $\mathbb{N} \rightarrow \mathbb{C}(\beta)$	
Hands-on	
Assumption $P(A) = P(B)$ Conclusion (to show): $A = B$	
Assumption $x \in A \Rightarrow x \in B \in P(A) \Rightarrow \{x\} \in P(B) \Rightarrow x \in B$	
A $\subseteq B$: Let $x \in A$ be any element of $A \Rightarrow x \in B \in P(A) \Rightarrow \{x\} \in P(B) \Rightarrow x \in B$	
B $\subseteq A$: $\forall x \in B \Rightarrow \exists x \in A (x \in B) \Rightarrow \{x\} \in P(A) \Rightarrow \{x\} \in P(B) \Rightarrow x \in B$	
(T 3.2) The set $\mathbb{N}^{<\mathbb{N}}$ is uncountable. See Number Theory, Cantor diagonalization argument	
03.23 Zermelo-Fraenkel Axioms	
03.24 Partially Ordered Set	
ref, sym and trans	
03.25 reflexivity	
03.26 symmetry	
03.27 antisymmetry	
03.28 transitivity	
03.29 Inverse relation $\forall a \in A \forall b \in B (a \sim b \Leftrightarrow b \sim a)$	
03.30 Partially ordered set (poset) ref, antisymm and trans	
03.31 composition of relations is associative $p \circ (q \circ r) = (p \circ q) \circ r$	
03.32 p is transitive iff $p \subseteq p^2$	
03.33 p^* = $\bigcup_{n=1}^{\infty} p^n$	

Number Theory

04.1 If there exists an integer c so that $b = ac$
For all integers a and $d \neq 0$, exist unique integers q and r satisfying

$$\text{division} \quad d = \gcd(a, b) := d | a \wedge d | b \wedge \forall c ((c | a \wedge c | b) \Rightarrow c | d)$$

If $\gcd(a, b) = 1$, then we call a and b relatively prime

$$L4.2 \quad \gcd(pn - qm) = \gcd(pn)$$

$$L4.3 \quad \gcd(a_1 b_1) = (a_1, b_1)$$

$$(x_1, y_1) \leftarrow (1, 0); (x_2, y_2) \leftarrow (0, 1);$$

$$(x_3, y_3) \leftarrow (x_1, y_1) - q(x_2, y_2); (x_4, y_4) \leftarrow (x_2, y_2)$$

$$D4.3 \quad \text{GCD } \text{Algo} \quad d \leftarrow s_1; u \leftarrow u_1; v \leftarrow v_1;$$

$$D4.4 \quad (a, b) := \begin{cases} u & a \equiv 0 \pmod{b} \\ \text{GCD}(u, b) & \text{otherwise} \end{cases}$$

D4.3 For $a, b \in \mathbb{Z}$ (not both 0) there exists $d \in \mathbb{Z}$ such that $(a|b) = (d)$

L4.4 Let $a, b \in \mathbb{Z}$ (not both 0) then if $(a|b) = (d)$ then d is gcd of a and b .

T4.9 \sqrt{n} is irrational unless n is a square ($n = c^2$ for some $c \in \mathbb{Z}$)

D4.6 $l = \text{lcm}(a, b) := \text{lcm}(1, b/m) \wedge l \mid m \mid l|m$

$$\text{lcm}(a, b) = \prod_{p \in \text{primes}} p^{\max\{e_i, e_j\}}$$

$$\Rightarrow \gcd(a, b) \cdot \text{lcm}(a, b) = ab$$

$$D4.8 \quad a \equiv m \pmod{b} \iff m|(a - b)$$

$$\Rightarrow a - b \equiv a \equiv m \pmod{b} \quad a \neq b \Rightarrow a \neq m$$

$$L4.5 \quad \text{If } a \equiv m \pmod{b} \text{ and } c \equiv n \pmod{b} \quad a + c \equiv m + n \pmod{b}$$

$$L4.7 \quad (i) \quad a \equiv n \pmod{b} \quad (ii) \quad a \equiv m \pmod{b} \quad \text{Modular Alg.}$$

$$L4.8 \quad (i) \quad \text{lcm}(R_m(a), R_n(b)) = R_m(bc) = R_m(b) \cdot R_n(b)$$

$$(ii) \quad \text{lcm}(ab) = R_m(a) \cdot R_n(b)$$

$$L4.9 \quad ax \equiv n \pmod{b} \quad \text{has a unique solution } x \in \mathbb{Z}_{\geq 0} \text{ iff } \gcd(a, b) = 1$$

$$L4.9 \quad \text{If } L4.9 \text{ then } x \text{ is multiplicative inverse of a product } m.$$

One can work $x \equiv m^{-1} \pmod{a}$ or $x \equiv m \pmod{a}$

To calculate the multipl. inv. one can use the gcd algo and use the equation $ax + my = 1$

DIVISION L4.1 $X \equiv m_1 \pmod{m_1} \wedge \dots \wedge X \equiv m_r \pmod{m_r}$ for every list m_1, \dots, m_r with $0 \leq a_i < m_i$

for X has a unique solution x satisfying $0 \leq x < M$

To construct a explicit solution one can use:

$$\begin{aligned} M_i N_i &\equiv m_i \quad 1 \\ X_i &\text{ random between } 0 \text{ and } m_i - 1 \\ X &= R_M \left(\sum_{i=1}^r a_i M_i N_i \right) \end{aligned}$$

$$\begin{aligned} Y_A &:= R_p(g \cdot x_A) \\ Y_B &:= R_p(g \cdot x_B) \\ K_{AB} &:= R_p(y_A \cdot y_B) \end{aligned}$$

$$K_{AB} \equiv p X_A X_B \equiv p \cdot (g \cdot x_B) \cdot X_B \equiv p \cdot g \cdot X_B \equiv p \cdot K_{AA}$$

$$L7 \quad \text{LCM Alg.} \quad d \leftarrow s_1; u \leftarrow u_1; v \leftarrow v_1;$$

$$D4.4 \quad (a, b) := \begin{cases} \text{lcm}(u, b) & u \equiv 0 \pmod{b} \\ \text{lcm}(a, b) & \text{otherwise} \end{cases}$$

D4.3 For $a, b \in \mathbb{Z}$ (not both 0) there exists $d \in \mathbb{Z}$ (not both 0) such that $(a|b) = (d)$

L4.4 Let $a, b \in \mathbb{Z}$ (not both 0) then if $(a|b) = (d)$ then d is lcm of a and b .

T4.9 \sqrt{n} is irrational unless n is a square ($n = c^2$ for some $c \in \mathbb{Z}$)

D4.6 $l = \text{lcm}(a, b) := \text{lcm}(1, b/m) \wedge l \mid m \mid l|m$

$$\text{lcm}(a, b) = \prod_{p \in \text{primes}} p^{\max\{e_i, e_j\}}$$

$$\Rightarrow \gcd(a, b) \cdot \text{lcm}(a, b) = ab$$

$$D4.8 \quad a \equiv m \pmod{b} \iff m|(a - b)$$

$$\Rightarrow a - b \equiv a \equiv m \pmod{b} \quad a \neq b \Rightarrow a \neq m$$

$$L4.5 \quad \text{If } a \equiv m \pmod{b} \text{ and } c \equiv n \pmod{b} \quad a + c \equiv m + n \pmod{b} \quad a \neq m \text{ or } c \neq n$$

$$L4.7 \quad (i) \quad a \equiv n \pmod{b} \quad (ii) \quad a \equiv m \pmod{b} \quad \text{Modular Alg.}$$

$$L4.8 \quad (i) \quad \text{lcm}(R_m(a), R_n(b)) = R_m(bc) = R_m(b) \cdot R_n(b)$$

$$(ii) \quad \text{lcm}(ab) = R_m(a) \cdot R_n(b)$$

$$L4.9 \quad ax \equiv n \pmod{b} \quad \text{has a unique solution } x \in \mathbb{Z}_{\geq 0} \text{ iff } \gcd(a, b) = 1$$

$$L4.9 \quad \text{If } L4.9 \text{ then } x \text{ is multiplicative inverse of a product } m.$$

One can work $x \equiv m^{-1} \pmod{a}$ or $x \equiv m \pmod{a}$

Multiplic. Inv. one can use the gcd algo and use the equation $ax + my = 1$

T4.20 Let m_1, m_2, \dots, m_r be pairwise relatively prime integers and let $M = \prod_{i=1}^r m_i$. For every list a_1, \dots, a_r with $0 \leq a_i < m_i$ for $1 \leq i \leq r$, the system of congruence equations

$$X \equiv a_1 \pmod{m_1}, X \equiv a_2 \pmod{m_2}, \dots, X \equiv a_r \pmod{m_r}$$

These Chinese remainder theorem Diff. tell them

Let m_1, m_2, \dots, m_r be pairwise relatively prime integers and let $M = \prod_{i=1}^r m_i$. For every list a_1, \dots, a_r with $0 \leq a_i < m_i$ for $1 \leq i \leq r$, the system of congruence equations

$$X \equiv a_1 \pmod{m_1}, X \equiv a_2 \pmod{m_2}, \dots, X \equiv a_r \pmod{m_r}$$

These Chinese remainder theorem Diff. tell them

$$\begin{aligned} \text{Hands-On} \\ \text{L4.1} \quad p &= \gcd(x-2y) \Rightarrow p \mid (x-2y) \wedge p \mid (p+2y) \Rightarrow p \mid x \\ &= p \mid (x+y) \Rightarrow p \mid x \\ \text{L4.2} \quad \text{L4.3} \quad \text{L4.4} \quad \text{L4.5} \quad \text{L4.6} \quad \text{L4.7} \quad \text{L4.8} \quad \text{L4.9} \end{aligned}$$

$$\begin{aligned} \text{L4.10} \quad \text{L4.11} \quad \text{L4.12} \quad \text{L4.13} \quad \text{L4.14} \quad \text{L4.15} \quad \text{L4.16} \quad \text{L4.17} \quad \text{L4.18} \quad \text{L4.19} \end{aligned}$$

Algebra

An algebra is a pair $\langle S; \Omega \rangle$ where S is a set and $\Omega = \{\omega_1, \dots, \omega_n\}$ is a list of operations on S .

Associativity. $\Omega \cdot 4$ $a \omega (b \omega c) = (a \omega b) \omega c$

Identity / neutral element. $\Omega \cdot 5, 3$ $\exists e \in G$ $\forall a \in G$ $a \omega e = e \omega a = a$

Inverse $\Omega \cdot 5, 6$ $\exists a \in G$ $a \omega a = e$ $\forall a \in G$ $a \omega a^{-1} = e$ $a^{-1} \omega a = e$

Commutativity $\Omega \cdot 6$ $a \omega b = b \omega a = a$ if a group has a commutative operation

Associativity Identity $\Omega \cdot 7$ Commutativity

Monoid	Needed	Not needed	Not needed	Not needed
Groups	Needed	Needed	Needed	Needed
Abelian Groups	Needed	Needed	Needed	Needed

L5.3 For a group $\langle G; *, \emptyset \rangle$ we have
 (i) $a * b = b * a$ (ii) left cancellation law $a * b = a * c \Rightarrow b = c$
 (iii) right cancellation law $b * a = c * a \Rightarrow b = c$

L5.4 For a group $\langle G; *, \emptyset \rangle$ we have
 (i) $a * x = b$ and $x * a = b$ have unique solutions

L5.5 A set of elements H is a subgroup of group $\langle G; *, \emptyset \rangle$ if
 (i) $\langle H, *, \emptyset \rangle$ is a group
 (ii) $H \subseteq G$
 (iii) for all $a, b \in H$, $a * b \in H$ (closed regarding identity)

These always exist two trivial subgroups: $\{e\}$ and G itself.
L5.6 A function ψ from a group $\langle G; *, \emptyset \rangle$ to a group $\langle H, \cdot, \emptyset' \rangle$ is a group homomorphism if, for all a and b : $\psi(a * b) = \psi(a) \cdot \psi(b)$

If ψ is a bijection from G to H then it is called an isomorphism and we say that G and H are isomorphic and write $G \cong H$.

L5.7 A group homomorphism ψ from $\langle G; *, \emptyset \rangle$ to $\langle H, \cdot, \emptyset' \rangle$ satisfies
 (i) $\psi(e) = e'$
 (ii) $\psi(a') = \psi(a)$ for all a

05.14 $\langle a \rangle = \{a^n | n \in \mathbb{Z}\}$ and for finite groups: $\langle a \rangle = \{e, a, a^2, \dots, a^{n-1}\}$

05.15 A group $\langle G, \cdot, \emptyset \rangle$ generated by an element $a \in G$ is called cyclic and a is called a generator of G .
 \Rightarrow if a is a generator then a is G -1

05.16 A cyclic group of order n is isomorphic to $\langle \mathbb{Z}_n; + \rangle$ (why?)
 $\text{ord}(a) = 16$ \Leftrightarrow a is a generator of G . All cyclic groups with same order are isomorphic.
 $\text{ord}(a) = 16$ \Leftrightarrow For $\langle \mathbb{Z}_n; + \rangle$ every element a with $\text{ord}(a) = 16$ is a generator of \mathbb{Z}_n .
 \Rightarrow a is a generator if and only if $\text{ord}(a) = n$.

L5.10 Let G be a finite group and H be a subgroup of G . Then the order of H divides the order of G : $|H|$ divides $|G|$

C5.10 Let G be a finite group. Then $a^{16} = e$ for every $a \in G$

D5.11 $\mathbb{Z}_m^* = \{a \in \mathbb{Z}_m | \text{gcd}(a, m) = 1\}$ (See L4.19)

D5.12 $\mathbb{Z}_m^* \cong \mathbb{Z}_{\frac{m}{\text{lcm}(p_1, \dots, p_r)}}$ prime factorization $m = \prod_{i=1}^r p_i^{e_i}$ $\Rightarrow \mathbb{Z}_m^* = \prod_{i=1}^r (\mathbb{Z}_{p_i})^{e_i - 1}$

T5.13 $\langle \mathbb{Z}_m; \cdot, \emptyset \rangle$ is a group

C5.14 For all $m \geq 2$ and all a with $\text{gcd}(a, m) = 1$, $a^{(m)} = e$

which is for primes $a^{p-1} = e$

T5.15 The group \mathbb{Z}_m is cyclic iff $m = 2, 4, p$ or $2p^e$ (where p is prime)

T5.16 The unique e -th root of eG is $x \in G$ satisfying $x^e = eG$ modulo G : $eG \in \langle x \rangle$

Primes P, Q
 $P = \{P_1, \dots, P_m\}$
 $Q = \{Q_1, \dots, Q_n\}$
 $Q \subseteq P$ because P is a commutative group
 $Q = \{q_1, \dots, q_k\}$

T5.17 A ring $\langle R, +, \cdot, 0, 1 \rangle$ is an algebra for which (i) $\langle R, +, 0 \rangle$ is a commutative group (ii) $\langle R, \cdot, 1 \rangle$ is a commutative group (iii) $a(b+c) = (ab)+(ac)$ for all $a, b, c \in R$ (left distributivity)

A ring is called commutative if multiplication is commutative ($ab = ba$)

T5.18 (i) $0a = a0 = 0$ (ii) $(ab)c = a(bc)$ (iii) $(ab)^{-1} = b^{-1}a^{-1}$ (any commutative ring with all and $b \neq 0$ has an inverse $a(b)$)

(iv) If $a, b \in R$ and $a \neq 0$, a is invertible \Leftrightarrow $\text{V}(a) = 1$ (i.e. a is invertible in its integral domain)
 05.24 The integral domain is a commutative ring without zero divisors

T5.19 A polynomial is defined $a(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n \in R[x]$

T5.20 For any ring R , $R[\mathbb{F}]$ is a ring

Fields
 05.23 \mathbb{Z}_p is a field iff p is prime, $\mathbb{F}_p = \mathbb{Z}_p$ is an integral domain then \mathbb{F}_p is a field

T5.24 A field is an integral domain then so is $\mathbb{F}[x]$

L5.22 If D is an integral domain then $\text{V}(D)$ is the root of all

L5.33 Let $a(x) \in R[x]$ then $a(x)$ is a divisor of $f(x)$ with degree at least 1 if $f(x)$ is called irreducible

D5.28 A polynomial $a(x) \in F[x]$ with $\text{deg}(a) \geq 1$ is called a root of a if $a(x) = 0$ for some $x \in F$

T5.25 Every polynomial is divisible only by constant polynomials and by constant multiples of a .

T5.27 A cyclic group of order n is isomorphic to $\langle \mathbb{Z}_n; + \rangle$ (why?)
 $\text{ord}(a) = 16 \Leftrightarrow$ a is a generator of \mathbb{Z}_n . All cyclic groups with same order are isomorphic.
 $\text{ord}(a) = 16 \Leftrightarrow$ For $\langle \mathbb{Z}_n; + \rangle$ every element a with $\text{ord}(a) = 16$ is a generator of \mathbb{Z}_n .

Polynomials of degree 1 always irreducible | $Pd=2,3$ irred. if they have not

MINDS-ON

$\phi(Rg \cdot Rn) = \phi(Rg) \cdot \phi(Rn) = R\phi(g) \cdot R\phi(n) = R(\phi(g) \cdot \phi(n))$. So R is given by the definition of an isomorphism.

isomorphisms do exist? One has to find some of those groups of order $n > 2$. How many different isomorphisms do exist?

the first time I ever saw him, he was no taller than a man's shoulder, and had a very pale face, with a thin, bony body.

The air craft society of the English Association was by no means the only one applying to each of the governors of the West Indies. If we want to repeat this procedure it will demand a great deal of time and obviously create the same kind of associations.

Second generation of the first type and second generation of the first type are completely homogeneous between two cyclic groups of order q .

and all roots of $\alpha(x) = x^2 + 3x + 1$ are $\sqrt{5} \pm i$, so $\alpha(3) = 3$, $\alpha(\bar{3}) = -3$, $\alpha(i) = -3i$, $\alpha(-i) = 3i$.

check if $X^4 + x^2 + \epsilon$ is irreducible over $\mathbb{Z}[\epsilon]$. One root is $\sqrt{-\epsilon}$. Check roots. One root needs to be of least 2. Check $\pm \frac{1}{2}(1 \pm i\sqrt{3})$. Roots are simple roots.

The area of the trapezoid is $\frac{1}{2} \times (x+2) \times x = x^2 + x$. The area of the triangle is $\frac{1}{2} \times x \times x = x^2$. The area of the rectangle is $x \times x = x^2$. The total area of the figure is $x^2 + x + x^2 = 2x^2 + x$.

Logic

D6.25 Literal Atomic formula or negation of atomic formula
 $\text{L} = \text{L}_1 \vee \dots \vee \text{L}_{m_1}$ (1)

Implication $A \Rightarrow B = \neg A \vee B$ Double implication $A \leftrightarrow B = (A \Rightarrow B) \vee (\neg A \Rightarrow \neg B)$

Logical consequence $F \models G \Leftrightarrow F \rightarrow G$ is a tautology L2.3 Wahrheit/Kehrwahr

Every interpretation suitable for F and G , which is a model for G .

Equivalence $F \equiv G \Leftrightarrow F \models G$ and $G \models F$ D6.12

For all yields the same truth values for F and G .

Tautology L2.2 F is tautology $\Leftrightarrow \neg F$ unsatisfiable

True/False of Quantifiers: $\exists y \forall x P(x,y) \models \forall x \exists y P(x,y)$ is true

Model: Suitable interpretation A for which $A(F) = 1 \mid A \models F$

Satisfiability: satisfiable if there is a model for F

Modus Ponens L2.7 $A \wedge A \rightarrow B \models B$

Indirect Proof L2.5 $\neg B \rightarrow \neg A \models \neg B$

Composition of implications L2.6 $(A \rightarrow B) \wedge (B \rightarrow C) \models A \rightarrow C$

Case distinction L2.8 $(A_1, V_{\dots}, A_n) \wedge (A_1 \rightarrow B) \wedge \dots \wedge (A_n \rightarrow B) \models B$

Proof by contradiction L2.9 $(\neg A \rightarrow B) \wedge \neg B \models A$

If a set of ~~non~~ objects is partitioned into K sets

sets at least one set contains at least $\lceil \frac{n}{K} \rceil$ objects

Existence Proof L2.10

$P(x) \equiv \exists x \, P(x)$

Proof by counterexample $\neg P(n) \models \neg P(n+1) \models \neg P(n)$

Proof by induction T2.11 $P(0) \wedge \forall n (P(n) \rightarrow P(n+1)) \models P(n)$

Proof by contradiction T2.12

$\neg P(n) \models \neg P(n+1)$

Proof by quadruple principle

Proof by existence proof

Proof by soundness T2.13

$\neg P(n) \models \neg P(n+1)$

Sound: No false statement has a proof

Sound $\Leftrightarrow \forall s \in S [(\exists p \in P, \varphi(s,p) = 1) \rightarrow T(s) = 1]$

Complete: Every tree of statements has a proof

It is complete $\Leftrightarrow \forall s \in S [T(s) = 1 \rightarrow (\exists p \in P, \varphi(s,p) = 1)]$

Derivation rule: $M \vdash G$ if G can be derived from set M by rule R . Notation

Calculus: Finite set of derivation rules $K = \{R_1, \dots, R_m\}$

$M \vdash F, M \vdash G \vdash F \wedge G$

$M \vdash F, M \vdash G \vdash F \vee G$

$M \vdash F, M \vdash G \vdash \neg F$

$M \vdash F, M \vdash G \vdash F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F$

$M \vdash F, M \vdash G \vdash \neg F \rightarrow \neg G$

$M \vdash F, M \vdash G \vdash \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

Logical consequence $F \models G \Leftrightarrow F \rightarrow G$ is a tautology

Every interpretation suitable for F and G , which is a model for G .

Interpretation is also called truth assignment.

Interpretation is also called structure.

Assigns to each ~~function~~ symbol a function $U_K \rightarrow U_L$

Assigns to each ~~variable~~ symbol a ~~function~~ symbol $U_K \rightarrow U_L$

Assigns to each variable symbol a value in U_L

Suitable if it defines all free symbols:

All function symbols all predicate symbols all free variables:

Bound if a variable x occurs in the form $\forall x G$ or $\exists x G$ otherwise it's free

D6.33 $F[X/T]$ denotes the formula after substituting every free occurrence of x by T

vs L6.1D For a formula in which y doesn't occur $\neg \forall x G \equiv \forall y G[T/y]$ ($\forall x G \equiv \forall y G$)

D6.37 Rectified Form: No variable occurs bound and free.

All bound variables that appear directly after a Quantifier symbol are distributed

L6.11 Universal instantiation $\forall x F \models F[X/t]$

D6.38 Special Form $Q_1, X_1, Q_2, X_2, \dots, Q_n, X_n, G$ (Q_i are Quantifiers)

D6.21 A set of literals is a Clause

D6.29 The set of clauses associated to a formula in CNF

Calculation $K = \{L_1 \vee \dots \vee L_m\} \wedge \neg A_1 \wedge \dots \wedge \neg A_n$ denoted as $K(F)$

in the set $K(F) = \{\{L_1\}, \{L_2\}, \dots, \{L_m\}, \{A_1\}, \{A_2\}, \dots, \{A_n\}\}$

D6.30 Resolution: Clause K is resolvent of K_1 and K_2 if there's a literal

$L \in K_1, \neg L \in K_2$ and $K = (K_1 \setminus \{L\}) \cup (K_2 \setminus \{\neg L\})$

if two clauses K_1 and K_2 have a resolvent K , then K is derived by the

rule $\text{res}: \neg K_1, K_2 \models \neg K$

Resolution Calculus $\text{res} = \text{res3}$

L6.6 The resolution calculus is sound

T6.7 A set of formulas is unsatisfiable iff $K(M) \vdash \text{res} \emptyset$

D6.2 Basic Equivalences

$F \models F \wedge F$ (idempotence)

$F \models F \vee F$ (complementarity)

$F \models \neg \neg F$ (double negation)

$F \models F \rightarrow F$ (distribution)

$F \models F \wedge (F \rightarrow G) \vdash G$ (distributivity)

$F \models F \vee (F \rightarrow G) \vdash G$ (double negation)

$F \models F \rightarrow (F \wedge G) \vdash F$ (absorption)

$F \models F \rightarrow (F \vee G) \vdash F$ (distributivity)

$F \models F \wedge (F \rightarrow G) \vdash F$ (double negation)

$F \models F \vee (F \rightarrow G) \vdash F$ (distributivity)

$F \models F \rightarrow (F \wedge G) \vdash F$ (absorption)

$F \models F \rightarrow (F \vee G) \vdash F$ (distributivity)

$F \models F \wedge (F \rightarrow G) \vdash F$ (double negation)

$F \models F \vee (F \rightarrow G) \vdash F$ (distributivity)

$F \models F \rightarrow (F \wedge G) \vdash F$ (absorption)

$F \models F \rightarrow (F \vee G) \vdash F$ (distributivity)

Sound: No false statement has a proof

Sound $\Leftrightarrow \forall s \in S [(\exists p \in P, \varphi(s,p) = 1) \rightarrow T(s) = 1]$

Complete: Every tree of statements has a proof

It is complete $\Leftrightarrow \forall s \in S [T(s) = 1 \rightarrow (\exists p \in P, \varphi(s,p) = 1)]$

Derivation rule: $M \vdash G$ if G can be derived from set M by rule R . Notation

Calculus: Finite set of derivation rules $K = \{R_1, \dots, R_m\}$

$M \vdash F, M \vdash G \vdash F \wedge G$

$M \vdash F, M \vdash G \vdash F \vee G$

$M \vdash F, M \vdash G \vdash \neg F$

$M \vdash F, M \vdash G \vdash F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow \neg \neg G$

$M \vdash F, M \vdash G \vdash \neg \neg F \rightarrow G$

- $F_1 = \forall x. \exists y. P(x,y) \wedge \neg P(f(x),y) \wedge Q(y, z) \wedge \neg Q(x, z)$. Find a structure which is 1) satisfiable and model 2) unsatisfiable. Because P is incomplete.
- $(\exists x. R + (\forall x. f(x) = x)) \wedge (\forall x. f(x) = -2x) \wedge P(x,y) = \neg \exists y. Q(x,y) = 1, 2 \wedge (\exists x. K(x) = K(y) \wedge P(x,y) = x) = 3, 4$ because for Q it is always 20.
- The resolution calculus is not complete: Sol: To do so, it suffices to provide two formulas F and G such that $F \vdash G$ but $F \not\models G$. For example, if it uses only $\{C_i\}$ as variables, then obviously $\{\forall x. P(x)\} \vdash_{\text{resolution}} \{\exists x. P(x)\}$, which can be easily verified by looking at the truth-tables. But there is no way to derive $K(A \wedge B)$ from $K(A)$ because we cannot 'apply' K .
The only way to pick two clauses on the left side is to pick $K_1 = K(A)$ and $K_2 = K(B)$. But those do not provide a resolvent as described in Definition 4.22.)
- Given Form: $F = \forall x. \exists y. P(x,y) \wedge \neg P(f(x),y) \wedge Q(x) \wedge \neg Q(f(x),z) \wedge \exists z. \forall y. P(y,z) \wedge \neg P(f(y),z) \wedge Q(z) \wedge \neg Q(f(z),x) \wedge \forall x. \exists y. P(x,y) \vdash F$
 $\exists y. P(x,y) \vdash \exists z. \forall y. P(y,z) \wedge \neg P(f(y),z) \wedge Q(z) \wedge \neg Q(f(z),x)$ Sol: Assume that the formula $\exists y. \forall z. \neg P(f(z),y)$ is true. By the definition of \exists , there exists at least one y such that $\neg P(f(z),y)$ is true. Let y^* be such a y . By the definition of \forall , we have that $\neg P(x,y^*)$ is true for all x . Therefore, we have that for all x there exists a y^* named $y(x)$ for which $\neg P(x,y)$ is true.
This means exactly that $\neg \exists y. P(x,y)$ is false. Consider the following counterexample: the universe is the set \mathbb{Z} of all integers and P is the predicate \leq , $\exists y. x \leq y$ is true, but the statement $\exists y. x > y$ is false.
- Extend Syntax and Semantics by NOT : Sol: To the definition of syntax (Def 2.5) we add the following statement: If F and G are formulas, then also $(\text{NOT } G)$ is a formula.
To the definition of semantics (Def 2.9) we add just the following, and not just!
 $A(\text{NOT } G) = 1 \iff A(G) = 0$ or $A(\text{NOT } G) = 0 \iff A(G) = 1$. Given truth table and some logical consequence rules ($\{A \wedge B\} \vdash C$) implies that $\{A\} \vdash B$. But this makes towards a contradiction that once this together with $\{B\} \vdash C$ means that $\{A \wedge B\} \vdash C$, but this is a contradiction to $\{A\}$. Overall, we showed that for any G such that $\text{NOT } G$ implies that G , we have $\text{NOT } \text{NOT } G = G$. This by definition of NOT means that $\text{NOT } \text{NOT } G$ is equivalent to G . So reflexivity is here to show that the relation \prec is transitive. Sol: Let $a \prec b$ and $b \prec c$. By the definition of \prec , this means that $A(a) \neq 0 \wedge A(b) \neq 0$ and $A(c) \neq 0$. From (1) and (2) and the transitivity of \prec it follows that $a \prec c$. It remains to show that $a \prec c$ implies that $A(c) = 1$. But this follows from a contradiction that once this together with $\{c\} \vdash c$ and the responsibility of \vdash implies that $\text{NOT } c$ (which is a contradiction to (2)).
- $\text{NOT } P(x) \vdash P(y)$
Solv: There exist a student such that if this student solves Exercise 13 then all students solved it. (all students solved Exercise 13)
So 1: $\text{NOT } P(x) \vee P(y)$ and $P(y) \vee P(x)$
So 2: $\text{NOT } P(x) \wedge P(y)$ and $P(y) \wedge P(x)$
- $\text{NOT } P(x) \vdash P(y)$
Solv: Show that F is a tautology. Sol: $F \equiv \neg \exists x. (P(x) \vee \neg P(x)) \rightarrow P(y)$ (def of \rightarrow)
 $\equiv \neg \exists x. (\exists x. P(x)) \wedge (\neg \exists x. P(x)) \equiv \neg \exists x. (P(x) \vee \neg P(x)) \wedge (\neg P(x) \vee P(x)) \equiv \neg \exists x. \neg (P(x) \vee P(x)) \equiv \neg \exists x. \neg T \equiv \neg F$
 $\text{NOT } P(x) \wedge P(y)$ and $P(y) \wedge P(x)$
 $\text{NOT } P(x) \vdash P(y)$
 $\text{NOT } P(x) \wedge P(y) \vdash P(y)$ (def of \wedge)
 $\text{NOT } P(x) \wedge (\text{NOT } P(x) \vee P(x)) \vdash P(y)$ (def of \vee)
 $\text{NOT } P(x) \vdash P(y)$
- $P(x) \rightarrow P(y)$
Solv: $\text{NOT } P(x) \vdash P(y)$
So 1: $P(x) \rightarrow P(y)$ and $\text{NOT } P(x) \wedge P(y)$
So 2: $P(x) \rightarrow P(y) \wedge \text{NOT } P(x) \vdash P(y)$
- $\text{NOT } P(x) \wedge P(y) \vdash P(x)$
Solv: Show that F is a tautology. Sol: $F \equiv \neg \exists x. (P(x) \vee \neg P(x)) \rightarrow P(y)$ (def of \rightarrow)
 $\equiv \neg \exists x. (\exists x. P(x)) \wedge (\neg \exists x. P(x)) \equiv \neg \exists x. (P(x) \vee \neg P(x)) \wedge (\neg P(x) \vee P(x)) \equiv \neg \exists x. \neg (P(x) \vee P(x)) \equiv \neg \exists x. \neg T \equiv \neg F$
 $\text{NOT } P(x) \wedge P(y) \vdash P(x)$
 $\text{NOT } P(x) \wedge (\text{NOT } P(x) \vee P(x)) \vdash P(x)$
- $P(x) \wedge \text{NOT } P(x)$
Solv: $\text{NOT } P(x) \vdash P(x)$
So 1: $P(x) \wedge \text{NOT } P(x)$
So 2: $P(x) \wedge (\text{NOT } P(x) \vee P(x)) \vdash P(x)$

- Prof T is complete. Sol: We have to show that for an arbitrary $S \subseteq S$ with $T(S) \subseteq S$ there exists a $P \in P$ with $P(S) = 1$. Thus let $S \subseteq S$ be arbitrary such that $T(S) = 1$. By definition of T , S encodes a suitable formula F of propositional logic that uses only $\{C_i\}$ as variables. Hence there exists a suitable formula F and G such that $F \vdash G$ but $F \not\models G$. For example, if it uses only $\{C_i\}$ as variables, then obviously $\{\forall x. P(x)\} \vdash_{\text{resolution}} \{\exists x. P(x)\}$.
Then obviously $\{\forall x. P(x)\} \vdash_{\text{resolution}} \{\exists x. P(x)\}$.
• Prof T is sound. Sol: Let $S \subseteq S$ and P encodes a syntactically correct formula F , and that P encodes a formula of F , we know that S encodes a formula F , and that P encodes a formula of F , which is 1) a model for F , and 2) it satisfies S . Because F is complete, it suffices to provide two formulas F and G such that $F \vdash G$ but $F \not\models G$. For example, if it uses only $\{C_i\}$ as variables, then obviously $\{\forall x. P(x)\} \vdash_{\text{resolution}} \{\exists x. P(x)\}$.
- Prof T is not complete. Sol: To do so, it suffices to provide two formulas F and G such that $F \vdash G$ but $F \not\models G$. For example, if it uses only $\{C_i\}$ as variables, then obviously $\{\forall x. P(x)\} \vdash_{\text{resolution}} \{\exists x. P(x)\}$.
- $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
Proof: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$ is a tautology
So 1: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$ is unsatisfiable
So 2: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
 $\text{NOT } P \wedge (\text{NOT } P \vee Q) \vdash P \vee Q$
 $\text{NOT } P \vdash P \vee Q$
 $\text{NOT } P \vdash \text{NOT } P \vee Q$
 $\text{NOT } P \vdash Q$
 $\text{NOT } P \vdash P \vee Q$
- $P \rightarrow Q \vdash P \wedge Q$
Proof: $P \rightarrow Q \vdash P \wedge Q$ is a tautology
So 1: $P \rightarrow Q \vdash P \wedge Q$ is unsatisfiable
So 2: $P \rightarrow Q \vdash P \wedge Q$
 $P \rightarrow Q \vdash P \wedge Q$
 $P \rightarrow Q \vdash (P \rightarrow Q) \wedge Q$
 $P \rightarrow Q \vdash \text{NOT } P \vee Q \wedge Q$
 $P \rightarrow Q \vdash \text{NOT } P \vee Q$
 $P \rightarrow Q \vdash Q$
 $P \rightarrow Q \vdash P \wedge Q$
- $P \vee Q \vdash P \wedge Q$
Proof: $P \vee Q \vdash P \wedge Q$ is a tautology
So 1: $P \vee Q \vdash P \wedge Q$ is unsatisfiable
So 2: $P \vee Q \vdash P \wedge Q$
 $P \vee Q \vdash P \wedge Q$
 $P \vee Q \vdash (P \vee Q) \wedge Q$
 $P \vee Q \vdash \text{NOT } P \vee Q \wedge Q$
 $P \vee Q \vdash \text{NOT } P \vee Q$
 $P \vee Q \vdash Q$
 $P \vee Q \vdash P \wedge Q$
- $\text{NOT } P \vdash P$
Proof: $\text{NOT } P \vdash P$ is a tautology
So 1: $\text{NOT } P \vdash P$ is unsatisfiable
So 2: $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash (\text{NOT } P) \vee P$
 $\text{NOT } P \vdash \text{NOT } P$
 $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash P$
- $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
Proof: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$ is a tautology
So 1: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$ is unsatisfiable
So 2: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash (P \wedge \text{NOT } Q) \vee (Q \wedge \text{NOT } P)$
 $\text{NOT } P \wedge \text{NOT } Q \vdash P \wedge \text{NOT } Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash Q \wedge \text{NOT } P$
 $\text{NOT } P \wedge \text{NOT } Q \vdash \text{NOT } P$
 $\text{NOT } P \wedge \text{NOT } Q \vdash Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
- $P \rightarrow Q \vdash P \wedge Q$
Proof: $P \rightarrow Q \vdash P \wedge Q$ is a tautology
So 1: $P \rightarrow Q \vdash P \wedge Q$ is unsatisfiable
So 2: $P \rightarrow Q \vdash P \wedge Q$
 $P \rightarrow Q \vdash P \wedge Q$
 $P \rightarrow Q \vdash (P \rightarrow Q) \wedge Q$
 $P \rightarrow Q \vdash \text{NOT } P \vee Q \wedge Q$
 $P \rightarrow Q \vdash \text{NOT } P \vee Q$
 $P \rightarrow Q \vdash Q$
 $P \rightarrow Q \vdash P \wedge Q$
- $P \vee Q \vdash P \wedge Q$
Proof: $P \vee Q \vdash P \wedge Q$ is a tautology
So 1: $P \vee Q \vdash P \wedge Q$ is unsatisfiable
So 2: $P \vee Q \vdash P \wedge Q$
 $P \vee Q \vdash P \wedge Q$
 $P \vee Q \vdash (P \vee Q) \wedge Q$
 $P \vee Q \vdash \text{NOT } P \vee Q \wedge Q$
 $P \vee Q \vdash \text{NOT } P \vee Q$
 $P \vee Q \vdash Q$
 $P \vee Q \vdash P \wedge Q$
- $\text{NOT } P \vdash P$
Proof: $\text{NOT } P \vdash P$ is a tautology
So 1: $\text{NOT } P \vdash P$ is unsatisfiable
So 2: $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash (\text{NOT } P) \vee P$
 $\text{NOT } P \vdash \text{NOT } P$
 $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash P$
- $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
Proof: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$ is a tautology
So 1: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$ is unsatisfiable
So 2: $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash (P \wedge \text{NOT } Q) \vee (Q \wedge \text{NOT } P)$
 $\text{NOT } P \wedge \text{NOT } Q \vdash P \wedge \text{NOT } Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash Q \wedge \text{NOT } P$
 $\text{NOT } P \wedge \text{NOT } Q \vdash \text{NOT } P$
 $\text{NOT } P \wedge \text{NOT } Q \vdash Q$
 $\text{NOT } P \wedge \text{NOT } Q \vdash P \vee Q$
- $P \rightarrow Q \vdash P \wedge Q$
Proof: $P \rightarrow Q \vdash P \wedge Q$ is a tautology
So 1: $P \rightarrow Q \vdash P \wedge Q$ is unsatisfiable
So 2: $P \rightarrow Q \vdash P \wedge Q$
 $P \rightarrow Q \vdash P \wedge Q$
 $P \rightarrow Q \vdash (P \rightarrow Q) \wedge Q$
 $P \rightarrow Q \vdash \text{NOT } P \vee Q \wedge Q$
 $P \rightarrow Q \vdash \text{NOT } P \vee Q$
 $P \rightarrow Q \vdash Q$
 $P \rightarrow Q \vdash P \wedge Q$
- $P \vee Q \vdash P \wedge Q$
Proof: $P \vee Q \vdash P \wedge Q$ is a tautology
So 1: $P \vee Q \vdash P \wedge Q$ is unsatisfiable
So 2: $P \vee Q \vdash P \wedge Q$
 $P \vee Q \vdash P \wedge Q$
 $P \vee Q \vdash (P \vee Q) \wedge Q$
 $P \vee Q \vdash \text{NOT } P \vee Q \wedge Q$
 $P \vee Q \vdash \text{NOT } P \vee Q$
 $P \vee Q \vdash Q$
 $P \vee Q \vdash P \wedge Q$
- $\text{NOT } P \vdash P$
Proof: $\text{NOT } P \vdash P$ is a tautology
So 1: $\text{NOT } P \vdash P$ is unsatisfiable
So 2: $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash (\text{NOT } P) \vee P$
 $\text{NOT } P \vdash \text{NOT } P$
 $\text{NOT } P \vdash P$
 $\text{NOT } P \vdash P$

Lineare Algebra - Lecture

Prof. Olga Sorkine-Hornung and Prof. Özlem Imamoglu

Contents

0.1	Prüfung	4
0.2	Aufbau	4
0.3	Übungsabgabe	4
1	Komplexe Zahlen (0.3)	4
1.1	Definition der komplexen Zahlen	4
1.2	Wie rechnet man mit komplexen Zahlen?	5
1.3	Konjugiert Komplexe Zahlen	6
1.4	Die Darstellung in Polarform	6
1.5	Wurzel ziehen bei komplexen Zahlen	7
2	Matrizen und Vektoren im \mathbb{R}^n oder \mathbb{C}^n	7
2.1	Das Rechnen mit Matrizen	9
2.2	Spezielle Fälle des Matrixprodukts	11
2.3	Die Transponierte einer Matrix: Symmetrische und Hermitesche Matrizen	14
3	Lineare Gleichungssysteme	19
3.1	Im Falle von n-linearen Gleichungen mit n Unbekannten gilt:	19
3.2	$m < n$ (Weniger Gleichungen als Unbekannte)	19
3.3	$m > n$ (Mehr Gleichungen als Unbekannte)	19
3.4	Algorithmus um Lineare Gleichungen zu lösen	19
3.5	Gauss Algorithmus	20
3.6	Die Lösungsmenge eines LGS	21
3.7	Die Inverse einer Matrix	23
3.8	Orthogonale und Unitäre Matrizen	24

3.9	LR Zerlegung	25
4	Vektorräume	27
4.1	Vektorraum Axiome ($V, +, \cdot$)	28
4.2	Unterräume, Erzeugendensysteme (subspace, spanning sets)	30
4.3	Lineare Abhängigkeit, Basen und Dimensionen	32
4.4	Basiswechsel und Koordinatentransformation	37
5	Lineare Abbildungen	39
5.1	Definition und Beispiele	42
5.2	Kern, Bild, Rang	44
5.3	Matrizen als lineare Abbildungen	47
5.4	Affine Räume und die allgemeine Lösung eines inhomogenes Gleichungssystems $Ax = b$	51
5.5	Die Abbildungsmatrix bei Koordinatentransformationen	51
5.5.1	Basiswechsel	51
6	Vektorräume mit Skalarprodukt	52
6.1	Orthonormalbasen (6.3)	55
6.2	Zusammenfassung	58
6.3	Warum orthonormale Basen?	59
6.4	Orthogonale Komplemente	59
6.5	Basiswechsel und Koordinatentransformation von Orthonormalbasen	60
6.5.1	Darstellung von linearen Abbildungen in Orthonormalbasen	62
6.5.2	Selbstabbildung	62
6.6	Orthogonale und unitäre Abbildungen	62
6.7	Normen von linearen Abbildungen und Matrizen (Operatornorm)	64
6.8	Konditionszahl einer regulären Matrix $A \in \mathbb{E}^{n \times n}$	67
6.8.1	Relativer Fehler:	67

7 Die Methode der kleinsten Quadrate	68
7.1 Approximation von Daten mit einer analytischen Funktion. (auch Regression genannt)	70
8 Determinante	71
8.1 Permutationen	71
8.2 Definition der Determinante	72
8.3 Spezielle Eigenschaften für Dreiecksdreiecksmatrizen und Linksdreiecksmatrizen .	73
8.4 Verifizierung durch die Permutationsformel	73
8.5 Eigenschaften der Determinante	73
8.6 Geometrische Bedeutung von \det	80
9 Eigenwerte und Eigenvektoren	81
9.1 Zusammenfassung	81
9.1.1 LR-Zerlegung	81
9.1.2 QR-Zerlegung	81
9.1.3 Spektralzerlegung	81
9.1.4 Singulärwertzerlegung	81
9.2 Spektralzerlegung	81
9.3 Dimension von Eigenwerten	84
9.4 Wie findet man die Eigenwerte für A	84
9.5 Wie finden wir Eigenvektoren?	85
9.6 Spektralzerlegung / Eigenwertzerlegung (spectral decomposition)	89
9.7 Eigenwertzerlegung symmetrischer und hermitischer Matrizen (the spectral theorem)	91
10 Die Spektralnorm und $\kappa_2(A)$	93
10.1 Geometrische Intuition für Eigenwertzerlegung	96
11 Singulärwertzerlegung (Nicht Prüfungsrelevant)	97

0.1 Prüfung

Letzte 4 Prüfungen sind relevant!
Prüfung besteht nur aus abgeänderten Hausaufgaben.

- Theorie Aufgaben
- Multiple Choice Aufgaben

The following was presented in the lecture on 19. September 2018.

0.2 Aufbau

<https://igl.ethz.ch/teaching/linear-algebra/la2018/>
<https://www.video.ethz.ch/lectures/d-math/2018/autumn/401-0131-00L.html>
Username: ima-18w
Password: 26Px7Rs

0.3 Übungsabgabe

Bis Freitag 15:00 der nächsten Woche
Auch per E-Mail möglich

1 Komplexe Zahlen (0.3)

1.1 Definition der komplexen Zahlen

Wie ist man auf komplexe Zahlen gekommen?

$$x^2 + 1 = 0 \quad (1)$$

Definition 1. Wir definieren eine neue Zahl, i , welche die Gleichung $x^2 = -1$ lösst. d.h $i^2 = -1$ gilt. Wir nennen diese Zahl die imaginäre Einheit.
Das Wort komplex steht für zusammengesetzt.

Definition 2. Eine Zahl z der Form $z = x + yi$ nennen wir eine komplexe Zahl. x und y sind dabei reelle Zahlen. i steht für die imaginäre Einheit. Jede komplexe Zahl $z=x+iy$ ist durch die reellen Zahlen x und y eindeutig festgelegt.

$$x + iy = a + ib \Leftrightarrow x = a, y = b. \quad (2)$$

Definition 3. x heisst der Realteil von $z = x + iy$

Man schreibt $x = \operatorname{Re}(z)$

y heisst der Imaginärteil von $z = x + iy$

Man schreibt $y = \operatorname{Im}(z)$

$$\mathbb{C} := z = x + iy \mid x, y \in \mathbb{R} \quad (3)$$

Die Menge der reellen Zahlen kann man als Teilmenge der komplexen Zahlen auffassen.

$$\mathbb{R} := z \in \mathbb{C} \mid \operatorname{Im}(z) = 0 \quad (4)$$

Bsp. $z = 1 + 2i \in \mathbb{C}$ $\operatorname{Re}(z) = 1$ $\operatorname{Im}(z) = 2$

$$0 = 0 + 0i \in \mathbb{R} \subset \mathbb{C} \quad (5)$$

- Zahlen der Form iy heissen rein imaginär (purely imaginary)
- $z = x + iy \in \mathbb{C} \longleftrightarrow (x, y) \in \mathbb{R} \times \mathbb{R}$.
Bsp: $2 + 5i \longrightarrow (2, 5)i \longrightarrow (0, 1)$
 $(e, \pi) \longrightarrow e + \pi i \longrightarrow (2, 0)$
- Man kann komplexe Zahlen als Punkte in der Zahlenebene darstellen.
- Die horizontale Achse nennen wir die reelle Achse
Die vertikale Achse heisst imaginäre Achse (wird mit $i\mathbb{R}$ bezeichnet.)

Einschub: Was bedeutet $\mathbb{R} \times \mathbb{R}$?

$$\mathbb{R} \times \mathbb{R} := (a, b) \mid a \in \mathbb{R}, b \in \mathbb{R} \quad (6)$$

$$A \times B := (a, b) \mid a \in A, b \in B \quad (7)$$

1.2 Wie rechnet man mit komplexen Zahlen?

Definition 4. Addition zweier komplexen Zahlen: $z = x + iy, w = u + iv$:
 $z + w = w + z := (x + u) + i(y + v)$.

Definition 5. Multiplikation mit reelen Zahlen: Sei $\alpha \in \mathbb{R}$

$z = x + iy \in \mathbb{C}$

$\alpha \cdot z := \alpha x + i\alpha y$

Definition 6. Multiplikation zweier komplexer Zahlen

$z = x + iy, w = u + iv$

$$wz = (u + iv)(x + iy) = ux + iuy + ivx - vy$$

$$\boxed{wz := (ux - vy) + i(uy + vx)}$$

Definition 7. Sei $z = x + iy$ eine komplexe Zahl.

Die zu z konjugiert komplexe Zahl ist die Zahl $x - iy$

Man schreibt \bar{z} (Aussprache: z konjugiert oder z bar)

$$z\bar{z} = (x + iy)(x - iy) = x^2 + y^2 \geq 0 \quad (8)$$

Definition 8. Die nicht negative reelle Zahl $\sqrt{z\bar{z}} =: |z|$ heisst der Betrag (modulus) oder Absolutbetrag der komplexen Zahl z.

Definition 9. Division bei komplexen Zahlen:

sei $z = x + iy, w = u + iv \neq 0$

$$\frac{z}{w} := \frac{z}{w} \frac{\bar{w}}{\bar{w}} = \frac{z\bar{w}}{|w|^2} \quad (9)$$

$$|z| = \sqrt{z\bar{z}} = \sqrt{x^2 + y^2}$$

The following was presented in the lecture on 19. September 2018.

1.3 Konjugiert Komplexe Zahlen

$$\overline{zw} = \bar{z} \cdot \bar{w} \quad (10)$$

$$\overline{\left(\frac{z}{w}\right)} = \frac{\bar{z}}{\bar{w}} \quad (11)$$

$$\bar{\bar{z}} = z \quad (12)$$

$$|\bar{z}| = |z| \quad (13)$$

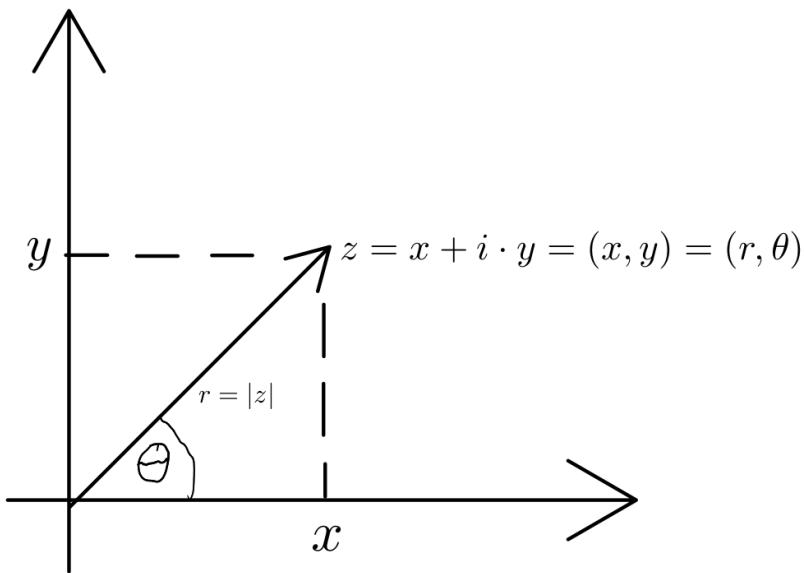
$$|zw| = |z| \cdot |w| \quad (14)$$

$$\left| \frac{z}{w} \right| = \frac{|z|}{|w|} \quad (15)$$

Dreiecksungleichung

$$|z + w| \leq |z| + |w| \quad (16)$$

1.4 Die Darstellung in Polarform



- $x = \cos \Theta$
- $y = \sin \Theta$

$$z = x + iy = r \cos \Theta + ir \sin \Theta = r(\cos \Theta + i \sin \Theta) \quad (17)$$

Eulerische Formel

$$e^{i\Theta} := \cos \Theta + i \sin \Theta \quad (18)$$

$$e^{i \cdot 0} = 1 \quad (19)$$

$$e^{i \frac{\pi}{2}} = i \quad (20)$$

$$e^{i2\pi} = 1 \quad (21)$$

$z = 0$ stellt ein Sonderfall dar

$r = |z| = 0$, der Winkel $\Theta = \arg(0)$ ist frei wählbar.

Multiplikation mit i gibt eine Drehung von 90° im Uhrzeigersinn.

1.5 Wurzel ziehen bei komplexen Zahlen

In \mathbb{C} kann man die Gleichung $z^2 = c$, $c \in \mathbb{C}$ für jede komplexe Zahl $c = se^{i\Phi}$ lösen. Ansatz:

$$z = re^{i\Theta}$$

$$c = se^{i\Phi}$$

$$z^q = z^2 = r^2 e^{2i\Theta} = c = se^{i\Phi}$$

Nun kann man den Term vor dem e und der Term in Exponent von e vergleichen.

$$\Rightarrow r = \sqrt{s} \quad (22)$$

r ist immer positiv, da man den Betrag nimmt

$$q\Theta = \Phi + 2\pi k$$

$$\Rightarrow \Theta = \frac{\Phi}{q} + \frac{2\pi k}{q} \quad | \quad k \text{ von } 0 \text{ bis } q-1 \quad (23)$$

Allgemein gilt: Für jede Zahl $c = se^{i\Phi} \in \mathbb{C}, q \in \mathbb{N}$, haben wir die Lösung der Gleichung

$z^q = c = se^{i\Phi}$

Es gibt immer q Lösungen. Also $z^8 = 1$ hat 8 Lösungen.

The following was presented in the lecture on 26. September 2018.

2 Matrizen und Vektoren im \mathbb{R}^n oder \mathbb{C}^n

Lineare Gleichung

$$ax + by + cz = d \quad (24)$$

Ein lineares Gleichungssystem (linear system of equations) von m Gleichungen mit n Unbekannten hat die Form:

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \quad (25)$$

$$a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \quad (26)$$

$$\dots \quad (27)$$

$$a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \quad (28)$$

Gegeben sind die Zahlen:

a_{ij} $i = 1, \dots, m$ $j = 1, \dots, n$ die Koeffizienten des Systems

b_i $i = 1, \dots, m$ die rechte Seite

Die Größen x_1, x_2, \dots, x_n sind die Unbekannten der Gleichung.

Matrix:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & & & \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_m \end{pmatrix} \quad (29)$$

$$\Rightarrow Ax = b \quad (30)$$

A ist die Koeffizienten Matrix

$$A = (a_{ij}) \quad 1 \leq i \leq m, \quad 1 \leq j \leq n \quad (31)$$

Definition 10. Eine $m \times n$ Matrix ist eine Anordnung (ein rechteckiges Schema) von mn Skalaren $a_{ij} \in \mathbb{R}$ (oder \mathbb{C}) in m Zeilen und n Spalten der Form:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & & & \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \quad (32)$$

kurz: $A = (a_{ij})_{i,j=1}^{m,n}$

$(a_{ij}) \quad 1 \leq i \leq m \quad 1 \leq j \leq n$

Die Zahlen $a_{11}, a_{12}, \dots, a_{mn}$ sind die Elemente der Koeffizienten der Matrix A.

Merksatz: Zeilenindex Zuerst, Spaltenindex Später

Notation: $a_{ij} = (A)_{ij}$

Definition 11. Die Elemente a_{jj} ($j = 1, \dots, \min\{m, n\}$) heißen Diagonalelemente

Eine $n \times n$ Matrix heißt quadratische Matrix

Eine $m \times 1$ Matrix heißt Spaltenvektor oder Kolonnenvektor oder m-Vektor

Eine $1 \times n$ Matrix heißt Zeilenvektor oder n-tuple

Das k-te Elemente des Kolonnenvektors S und des Zeilenvektors z nennen wir k-te Komponente des Vektors und bezeichnen wir mit s_k oder z_k

Definition 12. Zwei Matrizen A und B sind gleich, wenn sie die gleiche Anzahl von Zeilen und Spalten haben und die entsprechenden Elemente gleich sind.

Definition 13. Spezielle Matrizen:

Eine $m \times n$ Matrix heisst Nullmatrix, falls jedes Element gleich Null ist. Jede Nullmatrix wird mit 0 bezeichnet.

Einen Nullvektor wird mit 0 bezeichnet

Definition 14. Eine quadratische Matrix D heisst Diagonalmatrix (oder nur Diagonal) falls $d_{ij} = 0$ für $i \neq j$

Für die Diagonalmatrix mit gegebenen Diagonalen $d_{11}, d_{22}, \dots, d_{nn}$ schreiben wir $D = \text{diag}(d_{11}, d_{22}, \dots, d_{nn})$

Definition 15. Die $n \times n$ Diagonalmatrix $I_n = \text{diag}(1, \dots, 1)$ heisst Einheitsmatrix oder Identitätsmatrix (identity matrix)

Definition 16. Eine quadratische Matrix R heisst Dreiecksmatrix oder Rechtsdreiecksmatrix falls $r_{ij} = 0$ für $i > j$

Definition 17. Eine $n \times n$ Matrix L heisst untere Dreiecksmatrix oder Linksdreiecksmatrix falls $l_{ij} = 0$ für $i < j$

Definition 18. $\mathbb{R}^{m \times n} := \{M = (m_{ij}) \mid (m_{ij}) \in \mathbb{R}\}$

$\mathbb{C}^{m \times n} =$ Die Menge der Komplexen $m \times n$ Matrizen

$$\mathbb{R}^n := \left\{ \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} \mid x_i \in \mathbb{R}, 1 \leq i \leq n \right\}$$

2.1 Das Rechnen mit Matrizen

Definition 19. Die Multiplikation einer Matrix mit einer Zahl (einem Skalar):

Eine $m \times n$ Matrix A wird mit einer Zahl $\alpha \in \mathbb{R}$ (oder \mathbb{C}) multipliziert, indem man jedes Element von A mit α multipliziert.

Die resultierende $m \times n$ Matrix wird mit αA bezeichnet.

$$(\alpha A)_{ij} := \alpha(A)_{ij}$$

Definition 20. Addition zweier Matrizen:

Es seien A und B zwei $m \times n$ Matrizen A und B werden addiert, indem man entsprechende Elemente addiert.

$$(A + B)_{ij} := (A)_{ij} + (B)_{ij}$$

Definition 21. Das Produkt zweier Matrizen:

Sei A eine $m \times n$ Matrix.

Sei B eine $n \times p$ Matrix.

Produkt AB eine $m \times p$ Matrix. Das Produkt AB ist wie folgt definiert.

$$(AB)_{ij} := \sum_{k=1}^n (A)_{ik} (B)_{kj} = (A)_{i1}(B)_{1j} + (A)_{i2}(B)_{2j} + \dots + (A)_{in}(B)_{nj} \quad (33)$$

$$\text{Sei } C_{m \times p} := A_{m \times n} B_{n \times p} \quad A = (a_{ij}) \quad B = (b_{ij}) \quad C = (c_{ij}) \quad (34)$$

$$c_{ij} := \sum_{k=1}^n a_{ik} b_{kj} \quad (35)$$

Das Produkt AB kann nur gebildet werden, wenn die Anzahl Spalten von A mit der Anzahl Zeilen von B übereinstimmt.

Definition 22. $A_{m \times n}$ Matrix, AI_n und I_mA sind definiert.
 $\Rightarrow AI_n = A$, und $I_mA = A$

Theorem 2.1

- $(\alpha\beta)A = \alpha(\beta A)$
- Assotativgesetz: $(\alpha A)\beta = \alpha(A\beta) = A(\alpha\beta)$
- Distributivgesetz: $(\alpha + \beta)A = \alpha A + \beta A$
- Distributivgesetz: $\alpha(A + B) = \alpha A + \alpha B$
- Kommutativgesetz: $A + B = B + A$
- Kommutativgesetz: $(A + B) + C = A + (B + C)$
- Assoziativgesetz: $A(BC) = (AB)C$
- Distributivgesetz: $(A + B)C = AC + BC$
- Distributivgesetz: $A(B + C) = AB + AC$

Selbst für quadratische Matrizen gleicher Ordnung, sodass AB und BA definiert sind, im Allgemeinen gilt **nicht** dass $AB=BA$ ist.

Definition 23. Gilt für zwei Matrizen A, B sodass $AB = BA$, so sagt man dass diese Matrizen kommutieren.

1. Für jede $n \times n$ Matrix A , $I_n A = A I_n = A$
2. Für jede $m \times n$ Matrix A gilt $I_m A = A$ und $A I_n = A$
3. Für jede $m \times n$ Matrix A , $0_{n \times p}, 0_{s \times m}$ gilt $A_{m \times n} 0_{n \times p} = 0_{m \times p}$ und $0_{s \times m} A_{m \times n} = 0_{s \times n}$

The following was presented in the lecture on 28. September 2018.

Theorem 2.2

1. Es gibt eine $m \times n$ Nullmatrix 0 definiert durch $(0)_{ij} := 0 \quad \forall i, j$ so dass für jede $m \times n$ Matrix A gilt $A + 0 = 0 + A = A$
2. Zu jeder $m \times n$ Matrix A , gibt es eine $m \times n$ Matrix $-A$, definiert durch $(-A)_{ij} := -(A)_{ij}$ so dass $A + -A = 0 = -A + A$
3. Zu zwei $m \times n$ Matrizen A und B gibt es eine $m \times n$ Matrix \underline{X} definiert durch $(\underline{X})_{ij} := (B)_{ij} - (A)_{ij} \quad \forall i, j$ so dass $A + \underline{X} = B$

Note 1. 1. Sei $G = \mathbb{R}^{m \times n}$ oder $\mathbb{C}^{m \times n}$ haben wir folgende Eigenschaften für die Menge bezüglich $+$

1. $g^i \quad \exists$ ein Element $0 \in G$ so dass $A + 0 = 0 + A = A \quad \forall A \in G$
2. $g^{ii} \quad \forall A \in G, \exists$ ein Element $-A \in G$ sodass $A + -A = 0 = -A + A$
3. $g^{iii} \quad (A + B) + C = A + (B + C) \quad A, B, C \in G$

$g^i, g^{ii}, g^{iii} \Rightarrow (G, +)$ ist eine Gruppe.

Da $A + B = B + A$ ist, ist G eine Abelsch Gruppe (Abelian group) (kommutative Gruppe)

2. Beschränken wir uns auf die Menge der quadratischen Matrizen der Ordnung n in $\mathbb{R}^{n \times n}, \mathbb{C}^{n \times n}$, dann haben wir:

- R1 $\mathbb{R}^{n \times n}$ ist eine Abelsch Gruppe
- R2 $(AB)C = A(BC) \quad \forall A, B, C \in \mathbb{R}^{n \times n}$
- R3 $A(B + C) = AB + A$
 $(A + B)C = AC + BC \quad \forall A, B, C \in \mathbb{R}^{n \times n}$

R1, R2, R3 bedeuten dass $(\mathbb{R}^{n \times n}, +, \cdot)$

Die Menge der $n \times n$ Matrizen bezüglich Matrix Addition und Multiplikation einen Ring bildet. Weil $I_n A + A I_n = A \quad \forall A \in \mathbb{R}^{n \times n}$, sagt man dass die Menge $\mathbb{R}^{n \times n}$ einen Ring mit Eins bildet. (ring with identity)

Für $\alpha, \beta \in \mathbb{C}$ gilt $\alpha\beta = 0 \Rightarrow \alpha = 0$ oder $\beta = 0$ sein muss.

Beim Matrix Produkt ist das **nicht** der Fall.

Definition 24. Zwei $n \times n$ Matrizen A und B mit $AB = 0$ heissen Nullteiler (zero divisor).

Note 2. $(\mathbb{R}^{n \times n}, +, \cdot)$ ist ein nicht kommutativer Ring mit Eins und Nullteilern.

Beispiel für ein Ring: $R = \{0, 1, 2, 3, 4, 5\}$

$$a \oplus b := a + b \text{ mod } 6$$

$$2 \oplus 5 = 7 \text{ mod } 6 = 1$$

$$\forall a \in R$$

$$a \oplus 0 = 0 \oplus a = a$$

$$a \otimes 1 = 1 \otimes a = a$$

(R, \oplus, \otimes) ist ein kommutativer Ring mit 1 mit Nullteilern.

2.2 Spezielle Fälle des Matrixprodukts

1. Matrix-Vektor Produkt, einer $m \times n$ Matrix A mit einem n -Vektor $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$

$$A_{m \times n} x_{n \times 1} \Rightarrow b_{m \times 1} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix} \quad (36)$$

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \quad b_i = \sum_{k=1}^n a_{ik} x_k \quad 1 \leq i \leq m \quad (37)$$

2. Produkt eines Kolonenvektors x_{x+1} mit einer 1×1 Matrix α die man als Skalar auflösen kann

$$\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} (\alpha)_{1 \times 1} = \begin{pmatrix} x_1 \alpha \\ \cdots \\ x_n \alpha \end{pmatrix} = \begin{pmatrix} \alpha x_1 \\ \cdots \\ \alpha x_n \end{pmatrix} = \alpha \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \quad (38)$$

Erste Multiplikation ist das Produkt zweier Matrizen, die letzte Multiplikation ist das Skalarprodukt.

SEHR WICHTIG

Definition 25. Eine Linearkombination (linear combination) der Vektoren $\bar{a}_1, \bar{a}_2, \dots, \bar{a}_n$ ist ein Ausdruck der Form $\alpha_1 \bar{a}_1, \alpha_2 \bar{a}_2, \dots, \alpha_n \bar{a}_n$ worin die Skalare $\alpha_1, \alpha_2, \dots, \alpha_n$ Koeffizienten der Linearkombination (LK) sind.

Schreibweise: Um die Spaltenstruktur einer Matrix zu betonen schreiben wir:

$$A_{m \times n} = (\bar{a}_1 \ \bar{a}_2 \ \dots \ \bar{a}_n) = \begin{pmatrix} | & & & | \\ \bar{a}_1 & \cdots & \bar{a}_n \\ | & \cdots & | \end{pmatrix} \quad (39)$$

$$\bar{a}_1 = \begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{pmatrix} \quad \bar{a}_n = \begin{pmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{pmatrix} \quad (40)$$

Im folgenden Satz sehen wir, dass das “Matrix mal Vektor” Produkt geschrieben werden kann als Linearkombination der Spaltenvektoren der Matrix mit den Komponenten des Vektors als Koeffizienten.

Theorem 2.3

Sind $\bar{a}_1, \bar{a}_2, \dots, \bar{a}_n$ die Kolonenvektoren der $m \times n$ Matrix A, und x ein n-Vektor, so gilt:

$$Ax = \bar{a}_1 x_1, \bar{a}_2 x_2, \dots, \bar{a}_n x_n = x_1 \bar{a}_1, x_2 \bar{a}_2, \dots, x_n \bar{a}_n \quad (41)$$

Ist insbesondere e_j der j-te Kolonenvektor der Einheitsmatrix $I_n = (\bar{e}_1, \bar{e}_2, \dots, \bar{e}_n)$

$$e_j = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad (1 \text{ ist das } j\text{-te Element}), \text{ so gilt}$$

$$A\bar{e}_j = \bar{a}_j \quad (42)$$

Proof 2.3

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} a_{11}x_1 \cdots a_{1n}x_n \\ \vdots \\ a_{m1}x_1 \cdots a_{mn}x_n \end{pmatrix} = \begin{pmatrix} a_{11}x_1 \\ \vdots \\ a_{m1}x_1 \end{pmatrix} + \begin{pmatrix} a_{12}x_2 \\ \vdots \\ a_{m2}x_2 \end{pmatrix} + \dots + \begin{pmatrix} a_{1n}x_n \\ \vdots \\ a_{mn}x_n \end{pmatrix} \quad (43)$$

$$Ax = \begin{pmatrix} a_{11} \\ \vdots \\ a_{m1} \end{pmatrix} x_1 + \begin{pmatrix} a_{12} \\ \vdots \\ a_{m2} \end{pmatrix} x_2 + \dots + \begin{pmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{pmatrix} x_n = \bar{a}_1 x_1 + \bar{a}_2 x_2 + \dots + \bar{a}_n x_n \quad (44)$$

$$Ae_j = A \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \bar{a}_1 \cdot 0 + \bar{a}_2 \cdot 0 + \dots + \bar{a}_j \cdot 1 + \dots + \bar{a}_n \cdot 0 \quad (45)$$

$$\boxed{A\bar{e}_j = \bar{a}_j} \quad (46)$$

□

Im folgenden Satz zeigen wir, dass man das Matrixprodukt als die Zusammensetzung von Matrix mal Spaltenvektorprodukt interpretieren kann.

Theorem 2.4

Ist A eine $m \times n$ Matrix und $B = \begin{pmatrix} | & \cdots & | \\ \bar{b}_1 & \cdots & \bar{b}_p \\ | & \cdots & | \end{pmatrix}$ eine $n \times p$ Matrix so gilt:

$$AB = \begin{pmatrix} | & \cdots & | \\ A\bar{b}_1 & \cdots & A\bar{b}_p \\ | & \cdots & | \end{pmatrix} \quad (47)$$

Proof 2.4

Nach Satz 2.3 gilt $\bar{b}_i = B\bar{e}_i$. Ebenso die i -te Spalte der Produktmatrix AB ist $AB\bar{e}_i$. Mit dem Assoziativgesetz der Matrixmultiplikation folgt:

$$(AB)\bar{e}_i = A(B\bar{e}_i) = A\bar{b}_i \quad (48)$$

□

Corollary 2.5

Das Gleichungssystem $Ax = b$ hat genau dann eine Lösung, wenn b eine LK der Spalten von A ist.

$Ax = b$ hat eine Lösung $\Leftrightarrow b$ ist ein LK von $\bar{a}_1, \dots, \bar{a}_n$

Proof 2.5

$$A\bar{x} = \bar{b} \text{ hat eine L\"osung } \bar{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

$$\Rightarrow A \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = b \Rightarrow \bar{a}_1 x_1 + \bar{a}_2 x_2 + \dots + \bar{a}_n x_n = b$$

$\Rightarrow b$ ist eine LK der Kolonenvektoren $\bar{a}_1, \dots, \bar{a}_n$ von A
 b ist eine LK der $\bar{a}_1, \dots, \bar{a}_n$ von $A \Rightarrow \exists \alpha_1, \dots, \alpha_n \in \mathbb{R}$
 sodass $\alpha_1 \bar{a}_1 + \alpha_2 \bar{a}_2 + \dots + \alpha_n \bar{a}_n = b$

$$\Rightarrow \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_n \\ | & \cdots & | \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = b \Rightarrow Ax = b \text{ hat die L\"osung } x = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix}$$
□

The following was presented in the lecture on 03. October 2018.

Clarification

Matrixprodukt = Skalarprodukt:

$$\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \cdot \alpha = \alpha \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \quad (49)$$

injective: one-to-one

sujective: every value is taken.

For f to be "invertible" it has to be injective and sujective.

2.3 Die Transponierte einer Matrix: Symmetrische und Hermitesche Matrizen

Definition 26. Sei A eine $m \times n$ Matrix. Dann heisst die Matrix $n \times m$ Matrix A^T mit $(A^T)_{ij} := (A)_{ji}$, die Transponierte von A oder die zu A transponierte Matrix (transpose).

1. Eine Matrix heisst symmetrisch (symmetric) falls $A^T = A$ d.h. $(A)_{ij} = (A)_{ji} \forall i, j$
2. Ist A eine komplexe Matrix, so ist \bar{A} mit $(\bar{A})_{ij} = (\bar{A})_{ji}$ die zu A konjugierte komplexe Matrix.
3. $A^H := (\bar{A})^T = (\bar{A}^T)$ ist die zu A konjugiert-transponiert Matrix (conjugate transpose) oder Hermitesch-transponierte (Hermitian transpose) Matrix
4. Eine Matrix heisst Hermitesch (Hermitian) falls $A^H = A$ d.h. $\bar{A}^T = A$ d.h. $(A)_{ij} = \overline{(A)_{ji}} \forall i, j$
 Eine reelle hermitesche Matrix ist nat\"urlich symmetrisch.

Definition 27. Eine quadratische Matrix A ist schiefsymmetrisch (skew-symmetric) falls $A^T = -A$

Jede schiefsymmetrische Matrix hat lauter Nullen in der Diagonale.

Theorem 2.6

1. Für jede Matrix A gilt

$$(A^T)^T = A, \quad (A^H)^H = A \quad (50)$$

2. Für jede Matrix A und jedem Skalar $\alpha \in \mathbb{C}$ gilt:

$$(\alpha A)^T = \alpha A^T, \quad (\alpha A)^H = \bar{\alpha} A^H \quad (51)$$

3. Für $m \times n$ Matrizen A und B gilt:

$$(A + B)^T = A^T + B^T \quad (A + B)^H = A^H + B^H \quad (52)$$

4. Für jede $m \times n$ Matrix A und $n \times p$ Matrix B:

$$(AB)^T = B^T A^T \quad (AB)^H = B^H A^H \quad (53)$$

Note 3. Im Allgemeinen ist das Produkt zweier symmetrischen Matrizen nicht symmetrisch.

Theorem 2.7

Für symmetrische Matrizen A, B gilt $AB \text{ sym} \Leftrightarrow AB = BA$.

Für beliebige Matrizen gilt $A^T A$ und AA^T sind symmetrisch.

Wir können in den Sätzen 2.3 und 2.4 gemachten Aussagen "transponieren". Schreibweise:
Wollen wir die Zeilenstruktur einer $m \times n$ Matrix A betonen, so schreiben wir:

$$A = \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \\ \underline{a}_m & \cdots \end{pmatrix} \text{ oder } A = \begin{pmatrix} \underline{a}_1 \\ \vdots \\ \underline{a}_m \end{pmatrix} \quad (54)$$

A_k = k-ter Zeilenvektor von A.

Corollary 2.8

Werden die Zeilenvektoren der $m \times n$ Matrix A und der $n \times p$ Matrix B gemäss Spaltenvektor und ist:

$\underline{y} = (y_1 \ y_2 \ \dots \ y_n)_{1 \times n}$ ein Zeilenvektor, so gilt:

$$\underline{y}_{1 \times n} B_{n \times p} = y_1 \underline{b}_1 + y_2 \underline{b}_2 + \dots + y_n \underline{b}_n$$

$e_i^T B = \underline{b}_i$ ((i-te Zeile von I_n) $\times B$ = i-te Zeile von B)

$$\text{und } AB = \begin{pmatrix} \underline{a}_1 B \\ \vdots \\ \underline{a}_m B \end{pmatrix}$$

2.4 Das Skalarprodukt und die Norm von Vektoren in $\mathbb{R}^n, \mathbb{C}^n$. $x = \begin{pmatrix} x_1 \\ \vdots \\ x_2 \end{pmatrix}, y = \begin{pmatrix} y_1 \\ \vdots \\ y_2 \end{pmatrix} \in \mathbb{R}^2$

Skalarprodukt zweier Vektoren ist definiert durch $\langle x, y \rangle := x_1 y_1 + x_2 y_2$.

$$\cos \phi = \frac{\langle x, y \rangle}{\|x\| \|y\|} \quad (55)$$

$$\|x\| := \sqrt{\langle x, y \rangle} \quad (56)$$

Definition 28. Das (Euklische) Skalarprodukt oder innere Produkt (scalar, innerproduct) zweier Vektoren $x, y \in \mathbb{E}^n$ ($\mathbb{E} = \mathbb{R}$ oder \mathbb{C}) ist die Zahl

$$\langle x, y \rangle := x^H y = \sum_{k=1}^n \bar{x}_k y_k \text{ mit } x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

was sich im Falle \mathbb{R}^n reduziert auf

$$\langle x, y \rangle := \sum_{k=1}^n x_k y_k = x^T y$$

$$\langle \cdot, \cdot \rangle : \mathbb{E}^n \times \mathbb{E}^n \rightarrow \mathbb{C}$$

$$x, y \rightarrow \langle x, y \rangle = x^H y = \bar{x}^T y = \sum \bar{x}_k y_k$$

The following was presented in the lecture on 05. October 2018.

$$(x, y) \mapsto \langle x, y \rangle := x^H y = \sum_{k=1}^n \bar{x}_k y_k \quad (57)$$

Theorem 2.9

Für das Skalarprodukt im \mathbb{R}^n gilt:

s1 Es ist linear im zweiten Faktor:

$$\langle x, y + z \rangle = \langle x, y \rangle + \langle x, z \rangle \quad (58)$$

$$\langle x, \alpha y \rangle = \alpha \langle x, y \rangle \quad (59)$$

s2 Es ist symmetrisch:

$$\langle x, y \rangle = \langle y, x \rangle \quad (60)$$

s3 Es ist positiv definit:

$$\langle x, x \rangle \geq 0 \quad \forall x \in \mathbb{E}^n, \langle x, x \rangle = 0 \Leftrightarrow x = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \quad (61)$$

Für das Skalarprodukt in \mathbb{C}^n gelten:

s1, s3,

s2 wird ersetzt durch:

s2': $\langle x, y \rangle = \overline{\langle y, x \rangle}$ Es ist Hermetisch

Corollary 2.9

Das Skalarprodukt im \mathbb{R}^n ist linear im ersten Faktor, d.h. es ist bilinear. Es gilt s4:

$$\langle w + x, y \rangle = \langle w, y \rangle + \langle x, y \rangle \quad (62)$$

$$\langle \alpha x, y \rangle = \alpha \langle x, y \rangle \quad (63)$$

In \mathbb{C}^n ist es sesquilinear d.h. s4':

$$\langle w + x, y \rangle = \langle w, y \rangle + \langle x, y \rangle \quad \forall x, y \in \mathbb{C}^n \quad (64)$$

$$\langle \alpha x, y \rangle = \bar{\alpha} \langle x, y \rangle \quad \alpha \in \mathbb{C} \quad (65)$$

Definition 29. Die Länge oder 2-Norm oder Euklidische Norm eines Vektors $x \in \mathbb{E}^n$ ist die nicht negative reelle Zahl, $\|x\|$ definiert durch $\|x\| := \sqrt{\langle x, x \rangle} = \sqrt{x^H x} = \sqrt{\sum_{k=1}^n \bar{x}_k x_k} = \sqrt{\sum_{k=1}^n |x_k|^2}$

Aus der Definition der Norm und den Eigenschaften des Skalarprodukts folgt:

Theorem 2.11: Cauchy-Schwarz Ungleichung

$$|\langle x, y \rangle| \leq \|x\| \|y\| \quad (66)$$

Das Gleichheitszeichen gilt $\Leftrightarrow y$ ist ein Vielfaches von x .

Doppelte Betragsstriche bedeuten, dass es eine Norm von Vektoren ist. Man schreibt es wie folgt:

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

Proof 2.11: Beweis Cauchy-Schwarz Ungleichung

Schritt 1 - Der Betrag eines Vektor ist immer positiv:

$$0 \leq \|tx + y\| \quad | \text{ Definition Norm} \quad (67)$$

$$0 \leq \sqrt{\langle tx + y, tx + y \rangle} \quad | \text{ quadrieren} \quad (68)$$

$$0 \leq \langle tx + y, tx + y \rangle \quad | \text{ Definition Skalarprodukt} \quad (69)$$

$$0 \leq |tx + y| \cdot |tx + y| \cos 0 \quad | \text{ ausmultiplizieren} \quad (70)$$

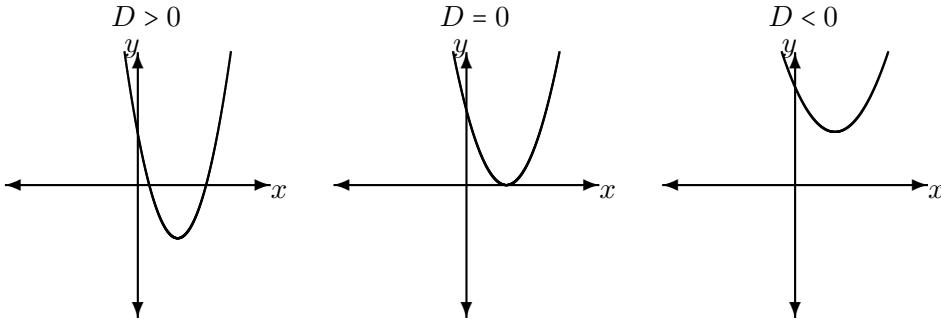
$$0 \leq t^2 \langle x, x \rangle + 2t \langle x, y \rangle + \langle y, y \rangle \quad | \text{ Definition Norm} \quad (71)$$

$$0 \leq t^2 \|x\|^2 + 2t \langle x, y \rangle + \|y\|^2 \quad (72)$$

Schritt 2 - Eigenschaften der Diskriminanten herausfinden:

$$D = (2\langle x, y \rangle)^2 - 4\|x\|^2\|y\|^2 \quad (73)$$

Wie wir gesehen haben, muss die Parabel immer Positiv sein. Welche Rückschlüsse kann man daraus auf die Diskriminante ziehen?



Wenn die Diskriminante Positiv wäre, sieht man das die Parabel auch ins Negative geht. Dies kann aber nicht der Fall sein, wie wir bei der Gleichung von Oben gesehen haben. Deshalb muss die Diskriminante Negativ oder Null sein.

$$\Rightarrow (2xy)^2 - 4\|x\|^2\|y\|^2 \leq 0 \quad (74)$$

Schritt 3 - Nun muss man nur noch diese Gleichung auflösen:

$$(2xy)^2 \leq 4\|x\|^2\|y\|^2 \quad (75)$$

$$(xy)^2 \leq \|x\|^2\|y\|^2 \quad | \text{ Wurzel ziehen} \quad (76)$$

Man muss den Betrag beachten, denn es gilt nur wenn der Term positiv ist.

$$|x \cdot y| \leq \|x\|\|y\| \quad | \text{ Definition vom Skalarprodukt} \quad (77)$$

$$|\langle x, y \rangle| \leq \|x\|\|y\| \quad (78)$$

□

Die Eigenschaften der Norm sind:

Theorem 2.12

Für die 2. Norm in \mathbb{E}^n gilt:

N1) Sie ist positiv definit:

$$\|x\| \geq 0 \forall x \in \mathbb{R}^n \quad (79)$$

$$\|x\| = 0 \Leftrightarrow x = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \quad (80)$$

N2) Sie ist dem Betrag homogen:

$$\|\alpha x\| = |\alpha|\|x\| \quad \alpha \in \mathbb{E}, x \in \mathbb{E}^n \quad (81)$$

N3) Die Dreiecksungleichung:

$$\|x + y\| \leq \|x\| + \|y\| \quad (82)$$

Definition 30. Sei $x, y \in \mathbb{E}^n$ der Winkel zwischen $x, y \in \mathbb{R}^n$ ist definiert durch

$$\cos \phi := \frac{\operatorname{Re} \langle x, y \rangle}{\|x\|\|y\|} \quad (83)$$

Im reellen Fall entfällt der Realteil

$$\cos \phi := \frac{\langle x, y \rangle}{\|x\| \|y\|} \quad (84)$$

Zwei n-Vektoren x, y sind zueinander orthogonal oder stehen senkrecht aufeinander, falls $\langle x, y \rangle = 0$. Wir schreiben $x \perp y$

Note 4. Clarification: $\mathbb{E} = \mathbb{R}$ or \mathbb{C}

3 Lineare Gleichungssysteme

3.1 Im Falle von n-linearen Gleichungen mit n Unbekannten gilt:

In der Regel gibt es genau eine Lösung

Es gibt eine schar von Lösungen wenn eine Gleichung als eine Summe von mehrfachen der anderen dargestellt werden kann ("Lineare Kombination")

3.2 $m < n$ (Weniger Gleichungen als Unbekannte)

Wenn $m < n$, so gibt es in der Regel unendlich viele Lösungen. Es kann aber auch keine geben, jedoch nie endlich viele.

3.3 $m > n$ (Mehr Gleichungen als Unbekannte)

Wenn $m > n$, so gibt es in der Regel keine Lösung. Es kann aber unedlich oder endlich Viele geben, wenn gewisse Abhängigkeiten zwischen den Gleichungen bestehen.

3.4 Algorithmus um Lineare Gleichungen zu lösen

Lineares Gleichungssystem:

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \quad (85)$$

$$\vdots \quad (86)$$

$$a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \quad (87)$$

In Matrixschreibweise:

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix} \quad (88)$$

$$Ax = b \quad (89)$$

Erweiterte Koeffizienten Matrix:

$$(A|b) = \begin{pmatrix} a_{11} & \cdots & a_{1n} & | & b_1 \\ \vdots & \ddots & \vdots & | & \vdots \\ a_{m1} & \cdots & a_{mn} & | & b_m \end{pmatrix} \quad (90)$$

Definition 31. Die Menge aller Lösungen eines LGS heisst Lösungsmenge

$$Solution(A : b) := \{x \in \mathbb{R}^n | Ax = b\} \quad (91)$$

Definition 32. Zwei Gleichungen sind äquivalent, falls sie die selbe Lösungsmenge haben.

Note 5. Das Ziel ist die Entwicklung eines effizienten Verfahrens zur Bestimmung der Lösungsmenge LGS

Die Idee: LGS wird so umgeformt, sodass ein neues neues LGS mit derselben Lösungsmenge entsteht wie das Anfangssystem, aber einfacher zu lösen ist.

The following was presented in the lecture on 10. October 2018.

3.5 Gauss Algorithmus

Operationen, die ein Lineares Gleichungssystem in ein äquivalentes überführen sind:

1. Das Vertauschen von Gleichungen
2. Die Addition / Subtraktion eines Vielfachen einer Gleichung zu / von einer anderen Gleichung.
3. Multiplikation einer Gleichung mit einer Zahl $\neq 0$

Idee von Gauss Elimination: Bringe das LGS durch die Schritte 1 und 2 in eine einfach zu lösende Form.

Note 6. Pivot ist die Zahl in der Matrix mit der wir die unteren Zeilen subtrahieren.

Algorithmus 3.1: Gauss'scher Algorithmus (1.1)

1. Bestimme die Spalte, die am weitesten auf der linken Seite steht und nicht verschwindende Elemente enthält.
2. Ist die oberste Zahl in der im Schritt 1 gefundenen Spalte Null, so vertauschen wir die erste Zeile mit einer anderen, so dass das oberste Element in der ersten Spalte nicht Null ist. Dieses Element nennen wir Pivot.
3. Subtrahiere ein passendes Vielfaches der obersten Zeile zu den übrigen Zeilen, um Unterhalb des Pivots Nullen zu erzeugen.
4. Wende die Schritte 1 bis 3 auf die Untermatrix an, die durch das Streichen der ersten Zeile und Spalte entsteht.

Nun erhält man ein neues System, welches man mit Rückwärtseinsetzen lösen kann.

Definition 33. Das nach jedem Schritt weiter verkleinerte System heisst Restgleichungssystem. Die Zeile beziehungsweise Kolone, in der des Pivotelement liegt heisst Pivotzeile b.z.w. Pivotkolone.

$$\begin{pmatrix} 1 & 0 & 0 \\ factor_1 & 1 & 0 \\ factor_2 & factor_3 & 1 \end{pmatrix} := L \quad R := \begin{pmatrix} element_1 & element_2 & element_3 \\ 0 & element_4 & element_5 \\ 0 & 0 & element_6 \end{pmatrix} \quad (92)$$

$$L \cdot R = A \quad A = \text{Anfangsgleichung} \quad (93)$$

Falls eine Umtauschung von Zeilen vorgenommen wurde, muss noch eine Permutationsmatrix darauf angewendet werden.

Definition 34. Permutationsmatrizen sind quadratische Matrizen, die in jeder Kolone und in jeder Zeile genau eine Eins und sonst lauter Nullen haben. Eine $n \times n$ Matrix ist eine Permutationsmatrix falls sie aus I_n durch Vertauschung von Zeilen hervorgegangen ist.

Note 7. Wenn eine Spalte kein Pivot enthält ist die Unbekannte dieser Spalte frei wählbar.

Note 8. Wenn ein System konsistent ist, bedeutet das, dass es Lösungen hat.

Note 9. Wenn das LGS keine frei wählbaren Parameter hat, hat das System eine eindeutige Lösung. Wenn es frei wählbare Parameter hat, hat es mehrere Lösungen.

The following was presented in the lecture on 12. October 2018.

Definition 35. In der Zeilenstufenform des Systems nennt man die Anzahl r der Pivotelemente, Rang der Matrix A und des Gleichungssystems.

Die im Falle $m > r$ für die Existenz einer Lösung notwendigen Bedienungen

$$c_{r+1} = c_{r+2} = \dots = c_m = 0$$

sind die Verträglichkeitsbedingungen des Systems

$$r \leq \min\{m, n\}$$

$n - r$ = Anzahl der freien Parameter

3.6 Die Lösungsmenge eines LGS

Note 10. Der Rang des Systems = Anzahl der Pivotelement = die Anzahl der Zeilen in der Zeilenstufenform, die nicht lauter Nullen enthalten = Anzahl der Zeilen mit Pivotelementen = Anzahl der Spalten mit Pivotelementen

$$r = \text{Rank } A \quad r \leq m = \text{Anzahl der Zeilen}$$

n = Anzahl der Spalten

$$0 \leq r \leq \min\{m, n\}$$

- Falls $r = m \Rightarrow$ Keine Nullzeilen im Endschema \Rightarrow kein Verträglichkeitsbedingungen VB \Rightarrow GS hat eine Lösung
- Falls $r < m = \text{Anzahl der Gleichungen} \Rightarrow$ Im Hauptteil des Endschemas treten Nullzeilen auf. \Rightarrow Es gibt VB (Verträglichkeitsbedingungen)

- Wenn mindestens eine dieser rechten Seiten $\neq 0$ ist so enthält das System einen Widerspruch
 \Rightarrow Keine Lösung

Theorem 3.1: (1.1)

Das System $Ax = b$ hat genau dann mindestens eine Lösung wenn entweder $r = m$ ist (d.h. kein VB)
oder $r < m$ und $c_i = 0 \forall i = r+1, \dots, m$
daher VB eindeutig erfüllt.

Gibt es Lösungen, so gilt:

- 1) Falls $r = n$, die Lösung ist eindeutig
- 2) Falls $r < n$, Es gibt eine $(n-r)$ parametrische Lösungsschar.
(Da die $n-r$ Variablen sind frei wählbar)

Note 11. Die Lösung eines LGS, falls existiert, ist eindeutig

- a) \Leftrightarrow Falls es keinen freien Parameter gibt $\Leftrightarrow n - r = 0 \Leftrightarrow n = r$
- b) Falls es eine Lösung gibt $\Leftrightarrow \{r = m\}$ oder $r < m$ und $c_{r+1} = \dots = c_m = 0$
- a) + b) $\Rightarrow Ax = b$ hat genau eine Lösung $\Leftrightarrow r = m = n$ oder $r = n < m$ und $c_i = 0 \quad i = r+1, \dots, m$

Corollary 3.2: (1.2)

Der Rang r hängt nur von der Koeffizientenmatrix A ab, aber nicht von den gewählten Pivotelementen oder von der rechten Seite b .

Definition 36. Ein LGS heißt homogen (eng.: homogeneous) falls die rechte Seite aus Nullen

besteht: $b = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$

Andernfalls heißt es inhomogen.

$Ax = 0$ homogene LGS $A = m \times n$

Die Lösung $x_1 = 0, x_2 = 0, \dots, x_n = 0$ eines homogenen GS $AX = 0$, heißt triviale Lösung.

Bei einem homogenen GS sind die rechten Seiten c_i mit $i = 1, \dots, m$ im Endschema alle Null.

Dass heißt, dass die VB bei einem homogenen GS immer erfüllt ist.

Note 12. Falls bei einem homogenen GS die Anzahl der Variablen $= n > m =$ Anzahl der Gleichungen gilt, so besitzt es nicht triviale Lösungen. (Mehr Unbekannte als Gleichungen)

Note 13. Sei A eine quadratische Matrix

$Ax = 0$ hat nur die triviale Lösung \Rightarrow Es gibt kein Freiparameter $\Leftrightarrow n - r = 0 \Leftrightarrow r = n$

$\Leftrightarrow n = m = r$ (A ist quadratisch)

$\Rightarrow Ax = b$ hat für jedes b mindestens eine Lösung.

Corollary 3.6: (1.6)

Eine quadratische LGS (mit $m = n$) ist genau dann für eine beliebige rechte Seite lösbar, wenn das zugehörige homogene System nur die triviale Lösung besitzt.

3.7 Die Inverse einer Matrix

$\alpha, \beta \in \mathbb{C}$, und $\alpha\beta = 0 \Rightarrow \alpha = 0$ oder $\beta = 0$ Bei Matrizen ist das nicht der Fall.

Gilt es für Matrizen das es immer ein inverses Element gibt?

Als 1 nehmen wir I_n

Nein es gilt nicht, beispielsweise $A = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$

Definition 37. Eine $n \times n$ Matrix A ist invertierbar, falls es eine $n \times n$ Matrix $\underline{\underline{X}}$ gibt, sodass $A\underline{\underline{X}} = I_n = \underline{\underline{X}}A$. Die Matrix $\underline{\underline{X}}$ heisst Inverse von A und wird mit A^{-1} bezeichnet.

Note 14. I_n ist immer invertierbar mit $I_n^{-1} = I_n$

Theorem 3.16: (2.16)

Ist A invertierbar, so ist die Inverse eindeutig bestimmt.

Proof 3.16

Sei $\underline{\underline{X}}, \underline{\underline{Y}}$ zwei Inversen von A
 $zz : \underline{\underline{X}} = \underline{\underline{Y}}$

$$\underline{\underline{X}} = \underline{\underline{X}}I_n = \underline{\underline{X}}(AU) = (XA)\underline{\underline{Y}} = I_n\underline{\underline{Y}} = \underline{\underline{Y}} \quad (94)$$

□

Theorem 3.17: (2.17)

Die folgenden Aussagen über eine $n \times n$ Matrix A sind äquivalent.

1. A ist invertierbar
2. $\exists n \times n$ Matrix $\underline{\underline{X}}$ mit $A\underline{\underline{X}} = \underline{\underline{X}}I_n = \underline{\underline{X}}(AU) = (XA)\underline{\underline{Y}} = I_n\underline{\underline{Y}} = \underline{\underline{Y}} = I_n$
3. \exists genau eine $n \times n$ Matrix $\underline{\underline{X}}$ mit $A\underline{\underline{X}} = I_n$
4. A ist regulär daher Rang $A = n$

A ist invertierbar $\Rightarrow A$ ist regulär daher Rang $A = n \Rightarrow$ Für jedes b gibt es genau eine Lösung für $Ax = b$. $\Rightarrow Ax = e_j \quad j = 1, \dots, n \quad I_n = \begin{pmatrix} | & \cdots & | \\ \bar{e}_1 & \cdots & \bar{e}_n \\ | & \cdots & | \end{pmatrix}$

Sagen wir $Ax_j = e_j$

$$A \begin{pmatrix} | & \cdots & | \\ \bar{x}_1 & \cdots & \bar{x}_n \\ | & \cdots & | \end{pmatrix} = \begin{pmatrix} | & \cdots & | \\ \bar{e}_1 & \cdots & \bar{e}_n \\ | & \cdots & | \end{pmatrix} = I_n \quad (95)$$

Sei A eine $n \times n$ Matrix. Führe das Gauss Verfahren für die erweiterte Matrix $(A \mid I_n)$ aus.

$$(A \mid I_n) = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & & & \\ 0 & 0 & \dots & 1 \end{pmatrix} \quad (96)$$

- Ist Rang $A = n$ (somit A ist regulär) so können wir mit Hilfe einer 3. Zeilenoperation alle Pivots auf den Wert 1 normieren
- Nach dem alle Pivots auf den Wert 1 normiert sind, können wir mit der Zeilenoperation 2 über den Pivots Nullen erzeugen.
- Danach steht im Schema Linkerhand die Einheitsmatrix I_n und Rechterhand A^{-1}

$$(A \mid I_n) \underset{\text{GaussVerfahren}}{\stackrel{=}{\sim}} (I_n \mid A^{-1}) \quad (97)$$

The following was presented in the lecture on 17. October 2018.

Theorem 3.18: (2.18)

Sind A, B reguläre Matrizen so gilt:

1. A^{-1} ist regulär und $(A^{-1})^{-1} = A$
2. AB ist regulär und $(AB)^{-1} = B^{-1}A^{-1}$
3. A^T und A^H sind regulär und $(A^T)^{-1} = (A^{-1})^T$ und $(A^H)^{-1} = (A^{-1})^H$

Theorem 3.19: (2.19)

Ist A regulär, so hat das Gleichungssystem $Ax = b$ für jede rechte Seite b, eine eindeutige Lösung und zwar ist $x = A^{-1}b$

3.8 Orthogonale und Unitäre Matrizen

Definition 38. Eine $n \times n$ Matrix A heisst unitär (unitary) falls $A^H A = I_n$. Eine reelle unitäre Matrix heisst auch orthogonal. Für sie gilt $A^T A = I_n$.

Theorem 3.20: (2.20)

Sind A und B unitär (bzw. orthogonal) $n \times n$ Matrizen, so gilt:

1. A ist regulär und $A^{-1} = A^H$ (bzw. $A^{-1} = A^T$)
2. $AA^H = I_n$ (bzw. $AA^T = I_n$)
3. A^{-1} ist unitär (orthogonal)

4. AB ist unitär (bzw. orthogonal)

Sei A eine orthogonale (oder unitäre) $n \times n$ Matrix

Sei $L_A : \mathbb{R}^n \mapsto \mathbb{R}^n$ die durch A definierte Abbildung

$$x \mapsto Ax$$

$$L_A(X) := Ax$$

Dann gilt folgender Satz

Theorem 3.21: (2.21)

Die durch eine orthogonale (oder unitäre) $n \times n$ Matrix A definierte Abbildung ist längentreu (length-preserving) oder isometrisch (isometry) und winkeltreu (angle-preserving) d.h.

$$\forall x, y \in \mathbb{R}^n (\mathbb{C}^n) \quad \|Ax\| = \|x\| \text{ (längentreu)} \quad (98)$$

$$\langle Ax, Ay \rangle = \langle x, y \rangle \quad (99)$$

Proof 3.21

Nach Voraussetzung $A^H A = I$

$$\langle Ax, Ay \rangle = (Ax)^H (Ay) = x^H \underbrace{A^H A}_{I_n} y = x^H I_n y = x^H y = \langle x, y \rangle \quad (100)$$

Wählt man $y := x$ folgt

$$\langle Ax, Ax \rangle = \langle x, x \rangle \Rightarrow \|Ax\| = \|x\| \quad (101)$$

□

Example 1. Permutationsmatrizen sind orthogonal.

- Eine $n \times n$ Matrix ist eine Permutationsmatrix falls sie ws I_n durch Vertauschung von Zeilen hervorgegangen ist.
- Sie sind quadratische Matrizen, die in jeder Zeile und in jeder Kolone genau eine 1 sonst lauter Nullen haben.

3.9 LR Zerlegung

$$Ax = b \quad (102)$$

1. Wir führen den Gauss Algorithmus durch, mit dem Unterschied, dass wir die rechte Seite nicht mitbehandeln.

2. Die Nullen unterhalb der Pivoten enthalten keine Information. Auf dem Computer ist es nicht sinnvoll diese zu speichern.

Aber die Quotienten l_{kj} (Beim Gaussverfahren wird das l_{kj} -fache der $j-te$ Zeile von der k -ten Zeile subtrahiert.) spielen eine wichtige Rolle. Anstelle der Nullen speichern wir die Quotienten l_{kj}

Sei A eine $n \times n$ Matrix. Die Durchführung der Gausselemination sei möglich ohne Zeilen zu vertauschen. Dann liefert das Gaussverfahren, angewandt auf die Matrix A , eine $m \times m$ invertierbare Linksdreiecksmatrix L mit Einsen in der Diagonale und eine $m \times n$ Matrix R in Zeilenstufenform, sodass $A = LR$ gilt.

$$L = (l_{kj}) = \begin{cases} \text{die Faktoren} & \forall k > j \\ 1 & \forall k = j \\ 0 & \forall k < j \end{cases} \quad (103)$$

- Note 15.**
1. Falls $m = n$, so ist R eine Rechtsdreiecksmatrix.
 2. $A = LR$ ist invertierbar $\Leftrightarrow R$ ist invertierbar.
 3. Falls beim Gaussverfahren, Zeilenumtauschungen nötig sind, wählt man eine Permutationsmatrix P , sodass für PA keine Zeilenumtauschung nötig sind. Dann liefert das Gaussverfahren die LR Zerlegung für $PA = PA = LR$

Note 16. Zeilenumtauschungen sind nötig, da Nullen oder relativ kleine Zahlen nicht als Pivot gewählt werden dürfen. Bei der LR Zerlegung, sollen die Pivots so gewählt werden, dass Rundungsfehler soweit wie möglich vermieden werden.

Zur Buchführung der Permutationsmatrix werden die Zeilenumtauschungen auch in der Permutationsmatrix vorgenommen.

Lösung für $Ax = b$

Falls Zeilenumtauschungen nötig sind. \Rightarrow LR Zerlegung von A ist $PA = LR$ für eine Permutationsmatrix P .

Wir können $Ax = b$ wie folgt lösen:

1. $Ax = b \Rightarrow PAx = Pb = \tilde{b}$
2. $PAx = \tilde{b} \Rightarrow L \underbrace{Rx}_{c} = \tilde{b}$
 - Löse $Lc = \tilde{b}$ nach c mit Vorwärtseinsetzen auf
 - Löse $Rx = c$ nach x mit Rückwärtseinsetzen auf

Theorem 3.1

$$PA = LR, \quad Lc = Pb, \quad Rx = c \quad (104)$$

Anleitung zum LR System im quadratischen System im Skript lesen.

Theorem 3.3

$$PA = LR, \quad Lc = Pb, \quad Rx = c \quad (105)$$

Anleitung zum LR System für ein beliebiges System im Skript lesen.

The following was presented in the lecture on 19. October 2018.

4 Vektorräume

1. Vektoraddition
2. Skalarprodukt: Multiplikation eines Vektors mit einer Zahl

Dabei gelten die folgenden Rechensregeln:

1. $x + y = y + x$
2. $(x + y) + z = x + (y + z)$
3. Für den Nullvektor 0 gilt, $x + 0 = x$
4. Für den zu x entgegengesetzten Vektor $-x$ gilt $x + (-x) = 0$
5. $\alpha(x + y) = \alpha x + \alpha y$
6. $(\alpha\beta)x = \alpha(\beta x)$
7. $(\alpha + \beta)x = \alpha x + \beta x$
8. $1 \cdot x = x$

Note 17. Es gibt andere Objekte, die sich addieren und mit Zahlen multiplizieren lassen, sodass die obigen 8 Rechensregeln gelten. Eine Funktion ist bestimmt durch folgende Angaben:

1. Der Definitionsbereich
2. Das Bild oder Wertebereich
3. Die Abbildungsvorschrift

Example 2. Definition von Funktionen:

$$V = F(\mathbb{R}, \mathbb{R}) := \{f : \mathbb{R} \rightarrow \mathbb{R} \mid f \text{ ist eine Funktion}\} \quad (106)$$

Gegeben:

$$f, g \in V, \quad \alpha \in \mathbb{R}$$

Definiere eine neue Abbildung:

$$(f + g)(s) := f(s) + g(s) \quad (107)$$

Definiere eine Multiplikation von $\alpha \in \mathbb{R}$, mit $f \in V$:

$$(\alpha f)(s) := \alpha f(s) \quad (108)$$

$$\mathcal{O}(s) = 0 \quad (109)$$

$$(f + \mathcal{O})(s) = f(s) + \mathcal{O}(s) = f(s) \Rightarrow f + \mathcal{O} = f \quad (110)$$

$$f + (-f) = \mathcal{O} \quad (111)$$

$$(-f)(s) := -f(s) \quad (112)$$

Definition 39. Für zwei Mengen A, B ist das kartesische Produkt $A \times B$, die Menge aller geordneten Paare

$$A \times B := \{(a, b) \mid a \in A, b \in B\} \quad (113)$$

Definition 40. Ein Vektorraum (Vector space) V über \mathbb{R} (oder \mathbb{C}) ist eine nicht leere Menge auf der eine Addition (innere Operation)

$$+ : V \times V \rightarrow V \quad (114)$$

$$(x, y) \mapsto x + y \quad (115)$$

und eine Skalare Multiplikation (äussere Opreation)

$$\cdot : \mathbb{R} \times V \rightarrow V \quad (116)$$

$$(\alpha, x) \mapsto \alpha x \quad (117)$$

definiert sind, sodass folgende Axiome gelten:

4.1 Vektorraum Axiome $(V, +, \cdot)$

Der Vektorraum ist definiert in \mathbb{E} (also \mathbb{R} oder \mathbb{C})

- | | |
|---|---|
| <p>(V1) $x + y = y + x, \quad \forall x, y \in V$</p> <p>(V2) $(x + y) + z = x + (y + z)$</p> <p>(V3) $\exists 0 \in V$ so dass $\forall x \in V : x + 0 = x$</p> <p>(V4) $\forall x \in V, \exists !$ (einzig bestimmt) $-x \in V$ so dass $x + (-x) = 0$</p> <p>(V5) $\alpha(x + y) = \alpha x + \alpha y \quad \forall \alpha \in \mathbb{E}, \forall x, y \in V$</p> <p>(V6) $(\alpha + \beta)x = \alpha x + \beta x \quad \forall \alpha, \beta \in \mathbb{E}, \forall x \in V$</p> <p>(V7) $(\alpha\beta)x = \alpha(\beta x), \quad \forall \alpha, \beta \in \mathbb{E}, \forall x \in V$</p> <p>(V8) $1, x = x \quad \forall x \in V$</p> | $\left. \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \right\} (V, +)$ ist eine Abelsche Gruppe |
|---|---|

- Die Elemente von V heißen Vektoren
- Der Vektor $0 \in V$ heißt Nullvektor
- Die Elemente von \mathbb{E} bezeichnet man als Skalare

Note 18. 1. Vektoren sind Elemente in einem Vektorraum

Wenn wir die Menge aller Funktionen betrachten, dann sind die Funktionen Vektoren.

$$f : \mathbb{R} \rightarrow \mathbb{R} \quad (118)$$

2. Was bedeuten die Operationen? VR ist immer abgeschlossen bezüglich Addition und Skalarmultiplikation.

$$+ : V \times V \rightarrow V \quad (119)$$

$$x, y \mapsto x + y \quad (120)$$

$$\cdot : \mathbb{R} \times V \rightarrow V \quad (121)$$

$$(\alpha, x) \mapsto \alpha \cdot x \quad (122)$$

3. Was bedeuten diese Eigenschaften V1-V8?

Das heisst, dass man mit Elementen des VRs (also Vektoren) besonders “schön” rechnen kann.

Theorem 4.1

Sei V ein Vektorraum über \mathbb{E} (\mathbb{R} oder \mathbb{C})

Sei $x, y \in V, \alpha \in \mathbb{E}$, Dann gilt:

1. Falls $y + x = x$, dann folgt $y = 0 \notin V$
2. Für $0 \in \mathbb{E}, x \in V$ $0x = 0 \in V$
3. Für $\alpha \in \mathbb{E}, 0 \in V$ ist $\alpha 0 = 0 \in V$
4. Für $x \in V, \alpha \in \mathbb{E}$ ist $(-\alpha)x = \alpha(-x) = -(\alpha x)$

Proof 4.1

1. Sei $x, y \in V, y + x = x$
Sei $-x \in V$, so dass $x + -x = 0$
 $y + x = x \Leftrightarrow (y + x) + (-x) = x + (-x) = 0$
Ass
 $\Leftrightarrow y + (x + -x) = 0$
 $\Leftrightarrow y + 0 = 0$
(V3)
 $\Leftrightarrow y = 0$
2. Für $0 \in \mathbb{E}, x \in V$, $0 \cdot x = 0 \in V$
 $0 + 0 = 0, x \in V$
 $(0 + 0) \cdot x = x \cdot x$
(V6)
 $\Leftrightarrow 0 \cdot x + 0 \cdot x = 0 \cdot x \Leftrightarrow 0 \cdot x = 0$
3. Homework
4. Homework

□

Theorem 4.2

Zu $x, y \in V$, existiert $z \in V$ mit $x + z = y$, wobei z eindeutig bestimmt ist und es gilt $z = y + (-x)$

Proof 4.2

Definiere $z := y + (-x)$

$$x + z = x + (y + (-x)) \stackrel{\text{Ass}}{=} (x + y) + (-x) \quad (123)$$

$$\stackrel{\text{Kom}}{=} (y + x) + (-x) \quad (124)$$

$$\stackrel{\text{Ass}}{=} (y) + (x - x) \quad (125)$$

$$= y + \mathcal{O} \quad (126)$$

$$= y \quad (127)$$

□

Definition 41. In einem VR V ist die Subtraktion zweier Vektoren $x, y \in V$ definiert durch

$$y - x := y + (-x) \quad (128)$$

4.2 Unterräume, Erzeugendensysteme (subspace, spanning sets)

Definition 42. Eine nichtleere Teilmenge U eines Vektorraumes V heisst Unterraum falls sie bezüglich Addition und Skalarer Multiplikation abgeschlossen ist d.h. $\forall x, y \in U, \alpha \in \mathbb{E}, x + y \in U$ und $\alpha x \in U$

Note 19. In einem Unterraum (UR), $U \subseteq V$ sind die Addtion und die skalare Multiplikation gleich defniert wie in V.

daher $(U, +, \cdot)$

Zudem gelten die Axiome V1,V2 und V5-V8 weil sie in V gelten.

$$\mathcal{O}_0 x \in U \Rightarrow \mathcal{O} \in U \quad (V3) \quad (129)$$

$$-1 \cdot x \in U \Rightarrow -x \in U \quad (\text{falls } x \in U) \quad (130)$$

$$\Rightarrow x + (-x) = \mathcal{O} \quad (V4) \quad (131)$$

$$\Rightarrow S \quad (132)$$

Theorem 4.3

Ein Unterraum ist selbst ein Vektorraum.

The following was presented in the lecture on 24. October 2018.

Example 3. $V = \mathbb{C}^2$

$U = \mathbb{R}^2$

Ist U ein Unterraum von V?

U ist kein Unterraum von \mathbb{C}^2 , da falls $\alpha \in \mathbb{C} \setminus \mathbb{R}$, da $\alpha U = \begin{pmatrix} \alpha a \\ \alpha b \end{pmatrix} \notin \mathbb{R}^2$

Note 20. Es gibt immer die trivialen Unterräume des Vektorraums V:
V selbst und $\{0\}$. Die Menge die nur aus dem Nullvektor besteht.

Theorem 4.4

Ist $A \in \mathbb{R}^{m \times n}$ eine reelle $m \times n$ Matrix,
 $\mathcal{L}_0 := \{x \in \mathbb{R}^n | Ax = 0\}$, so ist \mathcal{L}_0 eine Unterraum von \mathbb{R}^n
(Die Lösungsmenge des Homogenen LGS $Ax = 0$)

Proof 4.4

Falls $x, y \in \mathcal{L}_0$, daher $Ax = 0$ und $Ay = 0$ ist $x + y \in \mathcal{L}_0$?

Ist $A(x + y) \stackrel{?}{=} 0$

$$A(x) + A(y) = 0 + 0 = 0 \Rightarrow x + y \in \mathcal{L}_0$$

$$A(\alpha x) = \alpha Ax = 0 \Rightarrow \alpha x \in \mathcal{L}_0$$

□

Note 21. $Ax = b$

Die Lösungsmenge $\mathcal{L}_b := \{x \in \mathbb{R}^n | Ax = b\}$ ist kein Unterraum. Falls $x, y \in \mathcal{L}_b$

$$Ax = b \quad Ay = b \quad \Rightarrow A(x + y) = Ax + Ay = 2b$$

$b \neq 2b$ falls $b \neq 0$

Note 22. Im Allgemeinen sei U_1, U_2, \dots, U_m Unterräume von V/\mathbb{R} .

Definiere $U_1 + U_2 + \dots + U_m := \{\alpha_1 U_1 + \alpha_2 U_2 + \dots + \alpha_m U_m | \alpha \in \mathbb{R}\}$

Dann ist $U_1 + U_2 + \dots + U_m$ eine Unterraum

Beweis als Übung

Definition 43. Es seien V ein Vektorraum über \mathbb{E} und $a_1, a_2, \dots, a_l \in V$ ausgewählte Vektoren

Ein Vektor der Form:

$$x := \gamma_1 a_1 + \gamma_2 a_2 + \dots + \gamma_l a_l = \sum_{i=1}^l \gamma_i a_i \quad (133)$$

mit $\gamma_1, \gamma_2, \dots, \gamma_l \in \mathbb{E}$ ist eine Linearkombination von a_1, a_2, \dots, a_l

Die Gesamtheit aller Linearkombinationen von a_1, a_2, a_l ist ein Unterraum

Da, falls x, y Linearkombinationen von a_1, a_2, \dots, a_l sind , d.h.

$$x = \gamma_1 a_1 + \gamma_l a_l \text{ mit } \gamma_i \in \mathbb{R} \quad (134)$$

$$y = \alpha_1 a_1 + \alpha_l a_l \text{ mit } \alpha_i \in \mathbb{R} \quad (135)$$

$$x + y = (\gamma_1 + \alpha_1) a_1 + \dots + (\gamma_l + \alpha_l) a_l \Rightarrow x + y \text{ ist eine Linearkombination von } a_1, \dots, a_l \quad (136)$$

$$\alpha x = (\alpha \gamma_1) a_1 + \dots + (\alpha \gamma_l) a_l \Rightarrow \alpha x \text{ ist eine Linearkombination von } a_1, \dots, a_l \quad (137)$$

Dieser Vektorraum ist der von a_1, a_2, \dots, a_l aufgespannte oder erzeugte Unterraum. (subspace spanned by a_1, a_2, \dots, a_l) (Lineare Hülle) (linear hull)

Es wird mit $\text{span}\{a_1, \dots, a_l\}$ bezeichnet.

$$\text{span}\{a_1, a_2, \dots, a_l\} := \left\{ \sum_{i=1}^l \gamma_i a_i | \gamma_i \in \mathbb{R} \right\} = U \quad (138)$$

Die Vektoren a_1, a_2, \dots, a_l sind Erzeugendensystem (Spanning set)

Falls $S \subset V$ eine unendliche Menge von Vektoren ist, dann ist $\text{span } S$ definiert als

$$\text{span } S := \left\{ \sum_{k=1}^m \gamma_k a_k | a_1, \dots, a_m \in S, m \in \mathbb{N}, \gamma_1, \dots, \gamma_m \in \mathbb{R} \right\} \quad (139)$$

= Gesamtheit aller Linearkombinationen endlich vieler Vektoren

Die Menge S ist das Erzeugendensystem.

Example 4. Gegeben seien $a_1, a_2, \dots, a_k \in \mathbb{R}^n$

$$\text{Sei } A = \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_1 \\ | & \cdots & | \end{pmatrix} k \in \mathbb{R}^{n \times k} \quad (140)$$

a_1, a_2, \dots, a_k sind ein Erzeugendensystem von \mathbb{R}^n

\Rightarrow Für jeden Vektor $b \in \mathbb{R}^n, \exists \alpha_1, \alpha_2, \dots, \alpha_k \in \mathbb{R}$

sodass $b = a_1, a_2, \dots, a_k$

\Rightarrow Für jeden Vektor b , besitzt das LGS $Ax = b$ eine Lösung , $x = \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix}$

$\Leftrightarrow \text{Rang } A = n$

(Kor 1.3 $Ax = b$ hat für jedes b eine Lösung $\Leftrightarrow \text{Rang } A = n$)

Lemma 4.24

Seien $a_1, a_2, \dots, a_k \in \mathbb{R}^n$ und

$$A = \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_k \\ | & \cdots & | \end{pmatrix} \in \mathbb{R}^{n \times k} \quad (141)$$

Dann sind die folgenden Aussagen äquivalent:

1. a_1, a_2, \dots, a_k sind ein Erzeugendensystem in \mathbb{R}^n
2. Jeder Vektor $b \in \mathbb{R}^n$ ist eine Linearkombination von a_1, a_2, \dots, a_k
3. Für jeden Vektor $b \in \mathbb{R}^n$ besitzt $\sum_{i=1}^k a_i x_i = b$ eine Lösung.
4. Für jeden Vektor $b \in \mathbb{R}^n$ besitzt das LGS $Ax = b$ eine Lösung.
5. Rang $A = n$

Note 23. Ein Erzeugendensystem muss nicht immer eindeutig sein
($\{1, t, t^2\}$ und $\{1, t, t^2 + t\}$ sind das gleiche Erzeugendensystem)

4.3 Lineare Abhängigkeit, Basen und Dimensionen

Definition 44. Die Vektoren $a_1, a_2, \dots, a_l \in V$ sind linear abhängig (linear dependent) falls es Skalare $\gamma_1, \dots, \gamma_l$ gibt, die nicht alle Null sind und für die gilt:

$$\gamma_1 a_1 + \gamma_2 a_2 + \dots + \gamma_l a_l = 0 \quad (142)$$

Andernfalls ist a_1, a_2, \dots, a_l linear unabhängig. (linear independent)

Note 24. Wenn a_1, a_2, \dots, a_l linear unabhängig sind, so geht die Darstellung von Nullvektoren als Linearkombination von Vektoren a_1, a_2, \dots, a_l nur auf eine Weise, nämlich wenn alle Koeffizienten $a_1, a_2, \dots, a_l = 0$ sind. $\Leftrightarrow a_1, a_2, \dots, a_l$ sind linear unabhängig

Lemma 4.5

Die $l \geq 2$ Vektoren a_1, a_2, \dots, a_l sind linear abhängig \Leftrightarrow Einer dieser Vektoren lässt sich als Linearkombination der Anderen schreiben.

a_1, a_2, \dots, a_l sind linear abhängig $\Leftrightarrow \exists \alpha_1, \alpha_2, \dots, \alpha_l$ sodass :

$$a_k = \sum_{i=1, i \neq k}^l \alpha_i a_i \quad (143)$$

Proof 4.5

Sei a_1, a_2, \dots, a_l linear abhängig.

$$\Rightarrow \exists \gamma_1, \gamma_2, \dots, \gamma_l \in \mathbb{R}, \gamma_1 a_1 + \gamma_2 a_2 + \dots + \gamma_l a_l = 0 \quad (144)$$

und mindestens ein $\gamma_k \neq 0$ für ein $k \quad 1 \leq k \leq l$

$$\frac{\gamma_k \alpha_k}{\gamma_k} = \frac{-\gamma_1 a_1 - \gamma_2 a_2 - \dots - \gamma_{k-1} a_{k-1} - \gamma_{k+1} a_{k+1} - \dots - \gamma_l a_l}{\gamma_k} \quad (145)$$

$$\Rightarrow a_k = -\frac{\gamma_1}{\gamma_k} a_1 - \dots - \frac{\gamma_l}{\gamma_k} a_l \quad (146)$$

$\Rightarrow a_k$ ist eine Linearkombination der $a_1, a_2, \dots, a_{k-1}, a_{k+1}, \dots, a_l$.

Sei a_k ist eine Linearkombination von $a_1, a_2, \dots, a_{k-1}, a_{k+1}, \dots, a_l$.

$\Rightarrow \exists$ Zahlen $\alpha_1, \alpha_2, \dots, \alpha_l \in \mathbb{R} (l \neq k)$ sodass

$$a_k = \alpha_1 a_1 + \alpha_2 a_2 + \dots + \alpha_{k-1} a_{k-1} + \alpha_{k+1} a_{k+1} + \dots + \alpha_l a_l \quad (147)$$

$$\Rightarrow 1 \cdot a_k - \alpha_1 a_1 - \alpha_2 a_2 - \dots - \alpha_l a_l = 0 \quad (148)$$

\Rightarrow Eine Darstellung des Nullvektors 0, als eine Linearkombination von a_1, \dots, a_l mit mindestens einem Koeffizienten ungleich Null $\Rightarrow a_1, \dots, a_l$ sind linear abhängig. \square

The following was presented in the lecture on 26. October 2018.

Example 5. Seien $a_1, a_2, \dots, a_k \in \mathbb{R}^n$

$$A = \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_k \\ | & \cdots & | \end{pmatrix} \in \mathbb{R}^{n \times k} \quad (149)$$

a_1, a_2, \dots, a_k sind linear unabhängig

$\Rightarrow x_1 a_1, x_2 a_2, \dots, x_k a_k = 0$ hat nur die triviale Lösung

$\Rightarrow A \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$ hat nur die triviale Lösung.

\Rightarrow keine frei wählbaren Parameter \Rightarrow Rang $A = k$

Definition 45. Ein linear unabhängiges Ereugendensystem \mathbb{B} , eines Vektorraumes V , heisst Basis (eng.: basis) von V . Daher eine Basis $\mathbb{B} = \{b_1, \dots, b_n\}$ ist erzeugend und linear unabhängig. Daher $\text{span } \mathbb{B} = V$ ist erzeugend.

$$\alpha_1 b_1 + \alpha_2 b_2 + \dots + \alpha_n b_n = 0 \Rightarrow \alpha_1 = \alpha_2 = \dots = \alpha_n = 0 \text{ linear unabhängig} \quad (150)$$

Note 25. $\{e_1, e_2, \dots, e_n\}$ nennt man Standard Basis für \mathbb{R}^n

Example 6.

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \rightarrow \text{rank } A = 3 \quad (151)$$

(mittels Gaussverfahren)

Voraussetzung: Im folgenden, beschränken wir uns auf Vektorräume, die ein endliches Erzeugendensystem besitzen. Ein solcher Vektorraum nennen wir endlich Dimensional.

Lemma 4.6

Gibt es zu einem $V (\neq \emptyset)$ Vektorraum ein endliches Erzeugendensystem, so besitzt er eine Basis, die eine Teilmenge des Erzeugendensystems ist. Daher jedes Erzeugendensystem in einem endlich dimensionalen Vektorraum lässt sich zu einer Basis reduzieren.

Proof 4.6

V ist endlich Dimensional \Rightarrow Es gibt eine Menge von Vektoren $\{v_1, \dots, v_n\}$ so dass

$$\text{span}\{v_1, \dots, v_n\} = V \quad (152)$$

$$E = \{v_1, \dots, v_n\} \quad (153)$$

Ist $\{v_1, \dots, v_n\}$ linear unabhängig?

Ja $\{v_1, \dots, v_n\}$ ist eine Basis.

Nein Nach Lemma 4.5 $\exists v_k$ sodass $v_k = \alpha_1 v_1 + \dots + \alpha_{k-1} v_{k-1} + \alpha_{k+1} v_{k+1} + \dots + \alpha_n v_n$ für $\alpha_i \in \mathbb{R}$

$$E' \leftarrow \{\alpha_1 v_1 + \dots + \alpha_{k-1} v_{k-1} + \alpha_{k+1} v_{k+1} + \dots + \alpha_n v_n\} \quad (154)$$

Dann bilden die Restlichen ein neues Erzeugendensystem und man kann sich wieder die Frage stellen: "Ist $\{v_1, \dots, v_n\}$ linear unabhängig" bis wir ein Ja erhalten.

□

Note 26. Frage: Kann ein Vektorraum mit endlich Erzeugendensystem 2 Basen haben, eine mit n Vektoren und eine andere mit m Vektoren?

Antwort: Nein

Theorem 4.7

Besitzt ein Vektorraum ein endliches Erzeugendensystem, so besitzt jede Basis von V die selbe Anzahl von Vektoren.

Definition 46. Die Zahl der Basis Vektoren (in jeder Basis) eines Vektorraumes mit einem endlichen Erzeugendensystem heisst die Dimension von V und wird mit $\dim V$ bezeichnet. Falls $\dim V = n$ ist, sagen wir auch V ist n -dimensional.

Note 27. Man setzt für $V = \{0\}$, $\dim V := 0$

Lemma 4.8

Ist $\{b_1, \dots, b_m\}$ ein Erzeugendensystem von V , so ist jede Menge $\{a_1, \dots, a_l\} \subset V$ von $l > m$ Vektoren linear abhängig.

Note 28. Daher sei $\{b_1, \dots, b_m\}$ ein Erzeugendensystem und $\{a_1, \dots, a_l\}$ gegebenen l -Vektoren. $l > m \Rightarrow \{a_1, \dots, a_l\}$ ist linear abhängig.
oder $\{a_1, \dots, a_l\}$ ist linear unabhängig $\Rightarrow l \leq m$.

Proof 4.7: für Theorem 4.7

Sei $\{a_1, \dots, a_l\}$ und $\{b_1, \dots, b_m\}$ zwei Basen von V .

$\{b_1, \dots, b_m\}$ ist erzeugend und $\{a_1, \dots, a_l\}$ ist linear unabhängig $\xrightarrow{\text{Lemma 4.8}} l \leq m$
Umgekehrt:

$\{a_1, a_2, \dots, a_l\}$ sind erzeugend und $\{b_1, b_2, \dots, b_m\}$ sind linear unabhängig $\Rightarrow m \leq l$
 $\Rightarrow l = m$ □

Theorem 4.9

Jede Menge linearer unabhängiger Vektoren aus einem endlich dimensionalen Vektorraum V lässt sich Ergänzen zu einer Basis von V .

Note 29. Daher man kann ein linear unabhängig Menge von Vektoren zu einer Basis vergrössern.
Proof is Homework.

Corollary 4.10

Sei V ein n -dimensionaler Vektorraum. Dann ist jede Menge von n linearen unabhängigen Vektoren eine Basis.

Proof 4.10

Wäre eine solche Menge (eine Menge von l -linear unabhängigen Vektoren) keine Basis, lässt sie sich (nach Satz 4.9) zu einer Basis ergänzen. Diese würde aber mehr als $n = \dim V$ Vektoren enthalten. Das ist ein Widerspruch zu Satz 4.7. (Jede Basis besteht aus derselben Anzahl von Vektoren) □

Es sei V ein Vektorraum mit $\dim V = n$

Dann gilt:

1. Mehr als n Vektoren in V sind linear abhängig

2. Weniger als n Vektoren in V sind nicht erzeugend
3. n Vektoren in V sind linear unabhängig
 - \Leftrightarrow sie sind erzeugend
 - \Leftrightarrow sie bilden eine Basis

Koordinaten in endlich dimensionalen Vektorraum

Sei V ein Vektorraum mit $\dim V = n$

Sei $\{b_1, \dots, b_n\}$ eine Basis von V .

Sei $x \in V$, ein Vektor, weil Basis ein Erzeugendensystem ist, gibt es Zahlen $\alpha_1, \dots, \alpha_n \in \mathbb{E}$

$$\text{sodass } x = \alpha_1 b_1 + \dots + \alpha_n b_n = \sum_{i=1}^n \alpha_i b_i$$

Die Zahlen $\alpha_1, \dots, \alpha_n$ sind eindeutig bestimmt.

Wäre nämlich $x = \zeta_1 b_1 + \zeta_2 b_2 + \dots + \zeta_n b_n$

eine zweite Darstellung von x als Linearkombination von $\{b_1, \dots, b_m\}$

$$\Rightarrow 0 = x - x = \alpha_1 b_1 + \dots + \alpha_n b_n - (\zeta_1 b_1 + \zeta_2 b_2 + \dots + \zeta_n b_n) = (\alpha_1 - \zeta_1) b_1 + \dots + (\alpha_n - \zeta_n) b_n \quad (155)$$

Da b_1, \dots, b_n linear unabhängig sind:

$$\Rightarrow \alpha_1 - \zeta_1 = 0 = \alpha_n - \zeta_n \Rightarrow \alpha_i = \zeta_i \quad (156)$$

Definition 47. Die Koeffizienten ζ_1, \dots, ζ_n , ($x = \sum_{i=1}^n \zeta_i b_i$) heißen Koordinaten von x bezüglich der Basis $\{b_1, \dots, b_n\}$

$\begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix}$ ist der Koordinatenvektor

Die Abbildung $\mathcal{K} : V \rightarrow \mathbb{R}^n$

$$x \mapsto \begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix} =: \mathcal{K}(x) = [x]_{\mathcal{B}} \quad (157)$$

$\mathcal{B} = \{b_1, b_2, \dots, b_n\}$ heißt Koordinatenabbildung. Sie ist bijektive.

Theorem 4.12

$\{b_1, b_2, \dots, b_n\}$ ist genau dann eine Basis von V wenn sich jeder Vektor $x \in V$ eindeutig als Linearkombination von $\{b_1, b_2, \dots, b_n\}$ darstellen lässt.

The following was presented in the lecture on 31. October 2018.

Mehrere Unterräume

V ein VR.

Sei $M = \{b_1, \dots, b_l\}$ eine Menge linear unabhängiger Vektoren, $l < n$

M ist nicht erzeugend.

$U := \text{span}\{b_1, \dots, b_l\}$ U ist ein (echter) Unterraum.

$U \subset V$ (Nicht gleich)

$\{b_1, b_2, \dots, b_l\}$ ist eine Basis für U . Sei $N := \{b_{l+1}, \dots, b_n\}$ die Menge zusätzlicher Basis Vektoren, sodass $\{b_1, \dots, b_l, b_{l+1}, \dots, b_n\}$ eine Basis für V ist.

Jeder Vektor $x \in V$ hat eine eindeutige Koordinatendarstellung:

$$x = \underbrace{\sum_{k=1}^l \zeta_k b_k}_{\epsilon U = \text{span}\{b_1, b_2, \dots, b_l\}} + \underbrace{\sum_{k=l+1}^n \zeta_k b_k}_{\epsilon U'} \quad (158)$$

Sei $U' := \text{span}\{b_{l+1}, \dots, b_n\}$

Definition 48. Zwei Unterräume U und U' eines Vektorraumes V mit der Eigenschaft, dass jede $x \in V$ eine eindeutige Darstellung $x = u + u'$ mit $u \in U$ $u' \in U'$ hat, heissen komplementär (complimentary). Wir nennen den Vektorraum V die direkte Summe von U und U' und schreiben $V = U \oplus U'$

Note 30. $U \cap U' = \{0\}$

Sei $v \neq 0, v \in U \subset V$ und $v \in U' \subset V$

$$v = \sum_{i=1}^l \alpha_i b_i \quad \text{da} \quad v \in U = \text{span}\{b_1, b_2, \dots, b_l\} \quad (159)$$

$$\text{und } v = \sum_{i=l+1}^n \beta_i b_i \quad \text{da} \quad v \in U' = \text{span}\{b_{l+1}, \dots, b_n\} \quad (160)$$

$$\text{Dann } v = \sum_{i=1}^n \alpha_i b_i \quad \text{mit} \quad \alpha_i = 0 \quad i > l \quad (161)$$

$$\Rightarrow v = \alpha_1 b_1 + \dots + \alpha_l b_l + 0b_{l+1} + \dots + 0b_n \quad (162)$$

$$\Rightarrow v = \beta_{l+1} b_{l+1} + \dots + \beta_{l+n} b_n + 0b_1 + \dots + 0b_l \quad (163)$$

Zwei verschiedene Darstellungen von v als Linearkombination der Basis Vektoren b_1, \dots, b_n ↗

Note 31. Clicker Antworten an diesem Tag sind falsch. 1. Richtig 2. Falsch 3. Richtig 4. Falsch 5. Richtig 6. Richtig 7. Falsch

Note 32. Allgemein kann man einen Vektorraum als direkte Summe mehrerer Unterräume darstellen.

Note 33. V n-dimensionaler Vektorraum mit Basis= $\{b_1, b_2, \dots, b_n\}$

$$V = \text{span}\{b_1\} \oplus \text{span}\{b_2\} \oplus \dots \oplus \text{span}\{b_n\} \quad (164)$$

4.4 Basiswechsel und Koordinatentransformation

Es sei $\{b_1, b_2, \dots, b_n\}$ eine vorgegebene (alte) geordnete Basis des Vektorraumes V .

Wir wählen nun eine neue Basis $B' = \{b'_1, b'_2, \dots, b'_n\}$.

Jeder Vektor $x \in V$ kann als Linearkombination der Vektoren $\{b_1, b_2, \dots, b_n\}$ und $\{b'_1, b'_2, \dots, b'_n\}$ dargestellt werden.

$$x = \sum_{i=1}^n \zeta_i b_i = \sum_{i=1}^n \zeta'_i b'_i \quad (165)$$

$$[x]_B = \underbrace{\begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix}}_{\text{Koordinatenvektor bez\"uglich } B} \quad [x]_{B'} = \underbrace{\begin{pmatrix} \zeta'_1 \\ \vdots \\ \zeta'_n \end{pmatrix}}_{\text{Koordinatenvektor bez\"uglich } B'} \quad (166)$$

Frage: Was ist der Zusammenhang zwischen $\begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix}$ und $\begin{pmatrix} \zeta'_1 \\ \vdots \\ \zeta'_n \end{pmatrix}$?

Wir werden die neuen Basisvektoren b'_1, b'_2, \dots in der alten Basis darstellen.

$$b'_k = \tau_{1k} b_1 + \tau_{2k} b_2 + \dots + \tau_{nk} b_n \quad k = 1, \dots, n \quad (167)$$

Definition 49. Die $n \times n$ Matrix

$$T = (\tau_{ik}) = \begin{pmatrix} \tau_{11} & \cdots & \tau_{1n} \\ \vdots & \ddots & \vdots \\ \tau_{n1} & \cdots & \tau_{nn} \end{pmatrix} = \begin{pmatrix} \hline & \cdots & \hline \\ \overline{[b'_1]_B} & \cdots & \overline{[b'_n]_B} \\ \hline & \cdots & \hline \end{pmatrix} \quad (168)$$

heisst Tranformationsmatrix des Basiswechsels.

Theorem 4.13

Es sei $\zeta = \begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix} = [x]_B$ der Koordinatenvektor eines beliebigen Vektors $x \in V$ bez\"uglich der alten Basis B und sei $\zeta' = \begin{pmatrix} \zeta'_1 \\ \vdots \\ \zeta'_n \end{pmatrix} = [x]_{B'}$ der Koordinatenvektor des Vektors x bez\"uglich einer neuen Basis $B' = \{b'_1, b'_2, \dots, b'_n\}$ die durch $b'_k = \sum_{i=1}^n \tau_{ik} b_i \quad k = 1, \dots, n$ definiert ist. Dann gilt f\"ur diese Koordinatentransformation ($T = (\tau_{ik})$)

$$\zeta_i = \sum_{k=1}^n \tau_{ik} \zeta'_k \quad (169)$$

daher $\boxed{\zeta = T \zeta'}$ wobei die Transformationsmatrix T regul\"ar ist, sodass $\boxed{\zeta' = T^{-1} \zeta}$

Note 34. In der k -ten Kolone von T stehen gerade die Koordinaten des k -ten neuen Basisvektors bez\"uglich der alten Basis

$$T = \begin{pmatrix} \hline & \cdots & \hline \\ \overline{[b'_1]_B} & \cdots & \overline{[b'_n]_B} \\ \hline & \cdots & \hline \end{pmatrix} \quad (170)$$

Note 35. In $\boxed{b'_k = \sum_{i=1}^n \tau_{ik} b_i}$ und die neue Basis in der alten Basis dargestellt.

In $\boxed{\zeta_k = \sum_{k=1}^n \tau_{ik} \zeta'_k}$ werden die alten Koordinaten von x als Funtionen der neuen Koordinaten

beschrieben.

$$\begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix} = \begin{pmatrix} \tau_{11} & \cdots & \tau_{1n} \\ \vdots & \ddots & \vdots \\ \tau_{n1} & \cdots & \tau_{nn} \end{pmatrix} \begin{pmatrix} \zeta'_1 \\ \vdots \\ \zeta'_n \end{pmatrix} \equiv [x]_B = T[x]_{B'} \quad (171)$$

$$\begin{array}{ccc}
V, B & \xrightarrow{\hspace{2cm}} & V, B' \\
x & \xrightarrow{\hspace{2cm}} & x \\
\downarrow & & \downarrow \\
\varsigma = [x]_B \in E^n & \xrightarrow{T^{-1}} & \varsigma' = [x]_{B'} \in E^n \\
\varsigma = T \cdot \varsigma' & \xleftarrow[T]{\hspace{2cm}} & T^{-1} \cdot \varsigma = \varsigma'
\end{array}$$

Note 36. Theorem 4.13 sagt folgendes aus: $\Rightarrow \zeta = T\zeta' \Rightarrow \zeta' = T^{-1}\zeta$

The following was presented in the lecture on 02. November .2018.

5 Lineare Abbildungen

Seien V, W zwei Vektorräume.

$$F : (\cdot, +, V) \rightarrow (W, \oplus, \odot) \quad (172)$$

Example 7.

$$V = \mathbb{R}^2 \quad W = \mathbb{R}^{2 \times 2} \quad (173)$$

$$v_1 = (a, b) \mapsto \begin{pmatrix} a, 0 \\ 0, b \end{pmatrix} =: w_1 \quad (174)$$

$$v_2 = (c, d) \mapsto \begin{pmatrix} c, 0 \\ 0, d \end{pmatrix} =: w_2 \quad (175)$$

$$v_1 + v_2 = (a + c, b + d) \mapsto \begin{pmatrix} a + c, 0 \\ 0, b + d \end{pmatrix} \quad (176)$$

$$w_1 + w_2 = \begin{pmatrix} a, 0 \\ 0, b \end{pmatrix} \quad (177)$$

$$\alpha v_1 = (\alpha a, \alpha b) \mapsto \begin{pmatrix} \alpha a, 0 \\ 0, \alpha b \end{pmatrix} \quad (178)$$

$$(a, b) \mapsto \begin{pmatrix} a, 0 \\ 0, b \end{pmatrix} \xrightarrow{\alpha \cdot_w} \begin{pmatrix} \alpha a, 0 \\ 0, \alpha b \end{pmatrix} \quad (179)$$

$$F(v_1) +_w F(v_2) = F(v_1 +_v v_2) \checkmark \quad (180)$$

$$F(\alpha \cdot_v v_1) = \alpha \cdot_w F(v_1) \checkmark \quad (181)$$

Example 8.

$$F : \mathbb{R}^2 \rightarrow \mathbb{R}^3 \quad (182)$$

$$v_1 = (a, b) \mapsto (a^2, b^2, a+b) =: w_1 = F(v_1) \quad (183)$$

$$(a+c, b+d) \xrightarrow{F} \underbrace{(a+c)^2, (b+d)^2, (a+c+b+d)}_{F(v_1+v_2)} \quad (184)$$

$$v_2 = (c, d) \mapsto (c^2, d^2, c+d) =: w_2 = F(v_2) \quad (185)$$

$$F(v_1) + F(v_2) = (a^2 + c^2, b^2 + d^2, a+b+c+d) \quad (186)$$

$$F(v_1) + F(v_2) \stackrel{?}{=} F(v_1 + v_2) \quad (187)$$

Die obere Gleichung gilt nicht, denn F ist nicht linear.

Note 37. Sei F linear und V ein endlich dimensionaler Vektorraum mit Basis $B = \{b_1, b_2, \dots, b_n\}$ und W mit Basis $C = \{c_1, c_2, \dots, c_m\}$

$$F : V \rightarrow W \quad (188)$$

$$b_1 \rightarrow F(b_1) \quad (189)$$

$$b_n \rightarrow F(b_n) \quad (190)$$

Dann gilt, $\forall x \in V$

$$V \exists x = \alpha_1 b_1 + \dots + \alpha_n b_n \quad (191)$$

Des Weiteren gilt

$$F(x) = F(\alpha_1 b_1 + \dots + \alpha_n b_n) = F(\alpha_1 b_1) +_w \dots +_w F(\alpha_n b_n) = \alpha_1 F(b_1) + \dots + \alpha_n F(b_n) \quad (192)$$

Definition 50. Sei $F : \underline{X} \rightarrow Y$ eine Abbildung

Die Menge $F(\underline{X}) := \{F(x) \in Y \mid x \in \underline{X}\} \subset Y$

der Bildpunkte von F heisst Wertebereich (range) oder Bild (Image) von F .

Sie wird $\text{Im } F$ bezeichnet.

- Falls $F(\underline{X}) = Y$, so heisst F surjektive (onto)
- Falls $F(x) = F(x')$ folgt, dass $x = x'$ gilt, so heisst F injektive (one-to-one)
- Die Abbildung heisst bijektive, falls sie injektive und surjektive ist.

Ist F bijektive, so ist die inverse Abbildung F^{-1} definiert. Aber im Allgemeinen, wenn wir F^{-1} schreiben, ist F^{-1} nicht als Abbildung definiert.

Der Ausdruck $F^{-1}(Z), Z \subset Y$, für eine Teilmenge, bezeichnet die Menge der Urbilder von Elementen $z \in Z \subseteq Y$

$$F^{-1}(Z) := \{x \in \underline{X} \mid F(x) \in Z\} \subseteq X \quad (193)$$

Definition 51. Eine Abbildung $F : \underline{x} \rightarrow Y$ zwischen zwei Vektorräumen \underline{x} und Y über \mathbb{E} heißt lineare Abbildung (linear map, linear transformation) wenn $\forall x, x' \in \underline{x}, \forall \gamma \in \mathbb{E}$ gilt

$$F(x + \underline{x} x') = F(x) +_y F(x') \quad (194)$$

$$F(\gamma \cdot_{\underline{x}} x) = \gamma \cdot_y F(x) \quad (195)$$

oder anders gesagt: $\forall \alpha, \beta \in \mathbb{E}, x, x' \in \underline{X}$

$$F(\alpha x + \beta x') = \alpha F(x) + \beta F(x') \quad (196)$$

- Ist $Y = \mathbb{E}$ = der Skalar Körper, so bezeichnet man $F : \underline{x} \rightarrow \mathbb{E}$ als lineares Funktional (linear functional)
- Sind Definition und Bildraum Funktionsräume; zum Beispiel:
 $\underline{x} = \mathcal{F}(\mathbb{R}, \mathbb{R}) = \{f : \mathbb{R} \rightarrow \mathbb{R}\}$
 $y = \mathcal{F}(\mathbb{R}, \mathbb{R})$
spricht man von einem linearen Operator.

Example 9. Sei $A \in R^{m \times n}$ $m \times n$ Matrix

Definere eine Abbildung

$$\mathbb{R}^n \rightarrow \mathbb{R}^m \quad (197)$$

$$x \rightarrow L_A(x) = Ax \in \mathbb{R}^m \quad (198)$$

$$A = \begin{pmatrix} 2 & 3 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} \quad (199)$$

$$L_A : \mathbb{R} \rightarrow \mathbb{R}^3 \quad (200)$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto A \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 & 3 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2x_1 + 3x_2 \\ x_1 \\ x_1 \end{pmatrix} \in \mathbb{R}^3 \quad (201)$$

$$L_A(x + x') = A(x + x') = Ax + Ax' = L_A(x) + L_A(x') \quad (202)$$

$$L_A(\alpha x) = A(\alpha x) = \alpha Ax = \alpha L_A(x) \quad (203)$$

Example 10.

$$V = \mathbb{P}_2 \quad W = \mathbb{P}_1 \quad (204)$$

$$D : \mathbb{P}_2 \rightarrow \mathbb{P}_1 \quad (205)$$

$$p = a + bt + ct^2 \mapsto D(p) := b + 2ct \quad (206)$$

$\Rightarrow D$ ist linear.

Theorem 5.0

Jede lineare Abbildung $F : X \rightarrow Y$ lässt sich durch eine $m \times n$ Matrix darstellen.

Dazu wählt man die Basen $\mathcal{B} = \{b_1, b_2, \dots, b_n\}$ in X und $\mathcal{C} = \{c_1, c_2, \dots, c_m\}$ in Y

Seien $[x]_{\mathcal{B}} = (\zeta_1, \dots, \zeta_n)^T$ und $[y]_{\mathcal{C}} = [F(x)]_{\mathcal{C}} = (\eta_1, \dots, \eta_m)^T$ die Koordinatenvektoren von x und $y = F(x)$ bezüglich \mathcal{B} oder reziproke \mathcal{C} . Dann gilt:

$$[y]_{\mathcal{C}} = A[x]_{\mathcal{B}} \quad (207)$$

Hierbei gilt:

$$A = ([F(b_1)]_{\mathcal{C}} \ [F(b_2)]_{\mathcal{C}} \ \dots \ [F(b_n)]_{\mathcal{C}}) \quad (208)$$

Daher die Spalten von A sind die Koordinatenvektoren von $F(b_i)$ bezüglich \mathcal{C}

$\{b_1, b_2, \dots, b_n\} = \mathcal{B}$ eine Basis für $\underline{\underline{X}}$, $\dim x = n$
 $\{c_1, c_2, \dots, c_m\} = \mathcal{C}$ eine Basis für $\underline{\underline{Y}}$, $\dim y = m$

$$\begin{array}{ccc}
 X, B & \xrightarrow{F} & Y, C \\
 x & \xrightarrow{F} & F \cdot x = y \\
 \downarrow \kappa_{X,B} & & \downarrow \kappa_{Y,C} = \kappa_Y \\
 [x]_B \in E^n & \xrightarrow{A} & [Fx]_C = [y]_C \in E^m \\
 & & [y]_C = A[x]_B
 \end{array}$$

$$A := Mat(F, \mathcal{B}, \mathcal{C}) \quad (209)$$

The following was presented in the lecture on 07. November 2018.

5.1 Definition und Beispiele

Example 11.

$$E_2 : \mathcal{P}_3 \rightarrow \mathbb{R} \text{ Evaluations Abbildung} \quad (210)$$

$$p(t) \mapsto p(2) \quad (211)$$

$$\underline{\underline{x}} = \mathcal{P}_3 y = \mathbb{R} = \mathbb{R}' \quad (212)$$

$$\mathcal{B} = \{1, t, t^2, t^3\} \quad C = \{1\} \quad (213)$$

$$Mat(E_2, B, C) = ([E_2(1)]_C [E_2(t)]_C [E_2(t^2)]_C [E_2(t^3)]_C) \quad (214)$$

$$= (1 \ 2 \ 4 \ 8) \quad (215)$$

Example 12. Die Koordinaten Abbildung

Sei $\underline{\underline{x}}$ ein Vektorraum und $\mathcal{B} = \{b_1, b_2, \dots, b_n\}$ eine Basis.

Die Abbildung:

$$K_{\underline{\underline{x}}, B} : \underline{\underline{X}} \rightarrow \mathbb{E}^n \quad (216)$$

$$x \mapsto [x]_B = \zeta = \begin{pmatrix} \zeta_1 \\ \vdots \\ \zeta_n \end{pmatrix} \quad (217)$$

$$x = \zeta_1 b_1 + \dots + \zeta_n b_n, \mathcal{E} = \{e_1, e_2, \dots, e_n\} \quad (218)$$

$$K_{\underline{\underline{x}}, B} \text{ ist eine lineare Abbilung} \quad (219)$$

$$Mat(K_{\bar{X}, B}, B, \mathcal{E}) = b_1 = 1b_1 + 0b_2 + \dots + 0b_n \quad (220)$$

$$b_1 \in B \quad K(b_1) = [b_1]_B = \begin{pmatrix} 1 \\ 0 \\ \dots \\ 0 \end{pmatrix} \quad (221)$$

$$Mat(K, B, \mathcal{E} = ([K(b_1)]_{\mathcal{E}} [K(b_2)]_{\mathcal{E}} \dots [K(b_n)]_{\mathcal{E}}) \quad (222)$$

$$\left(\begin{bmatrix} 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}_{\mathcal{E}} \begin{bmatrix} 0 \\ 1 \\ \dots \\ 0 \end{bmatrix}_{\mathcal{E}} \dots \begin{bmatrix} 0 \\ 0 \\ \dots \\ 1 \end{bmatrix}_{\mathcal{E}} \right) = I_n \quad (223)$$

Example 13.

$$id_x : \bar{X}, \mathcal{B} \rightarrow \bar{X}, \mathcal{B}' \quad (224)$$

$$x \mapsto x \quad (225)$$

id_x ist linear.

$$Mat(id_x, \mathcal{B}, \mathcal{B}') = \begin{pmatrix} | & | & \dots & | \\ [id_x(b_1)]_{\mathcal{B}'} & [id_x(b_2)]_{\mathcal{B}'} & \dots & [id_x(b_n)]_{\mathcal{B}'} \\ | & | & \dots & | \end{pmatrix} = ([b_1]_{\mathcal{B}'} [b_2]_{\mathcal{B}'} \dots [b_n]_{\mathcal{B}'}) = T^{-1} \quad (226)$$

$$id_x : \bar{X}, \mathcal{B}' \rightarrow \bar{X}, \mathcal{B} \quad (227)$$

$$Mat(id_x, \mathcal{B}', \mathcal{B}) = \begin{pmatrix} | & | & \dots & | \\ [id_x(b'_1)]_{\mathcal{B}} & [id_x(b'_2)]_{\mathcal{B}} & \dots & [id_x(b'_n)]_{\mathcal{B}} \\ | & | & \dots & | \end{pmatrix} = ([b'_1]_{\mathcal{B}} [b'_2]_{\mathcal{B}} \dots [b'_n]_{\mathcal{B}}) = T \quad (228)$$

Definition 52. Ein bijektive linear Abbildung von \bar{X} auf Y heisst Isomorphismus.
Falls $\bar{X} = Y$ ist, so heisst sie Automorphismus.

Theorem 5.1

Ist $F : \bar{X} \rightarrow Y$ ein Isomorphismus, so ist auch, $F^{-1} : Y \rightarrow X$ die Inverse Abbildung.

Corollary 5.1

Wird der Isomorphismus F bezüglich festen Basen durch die Matrix A dargestellt, so ist A regulär und die Umkehrabbildung F^{-1} wird durch A^{-1} dargestellt.

Falls die Bildmenge einer Abbildung F im Definitionsbereich einer zweiten Abbildung G liegt, kann man diese zusammensetzen zur Komposition $G \circ F$. Dabei gilt:

Lemma 5.3

Sind, x, y, z , Vektorräume über \mathbb{E} und $F : x \rightarrow y$ $G : y \rightarrow z$ lineare Abbildungen, so ist auch $G \circ F : F \rightarrow Z$ linear.

Sind A und B die Abbildungsmatrizen zu F und G bezüglich festen Basen in X, Y, Z, so hat $G \circ F$ die Abbildungsmatrix BA.

5.2 Kern, Bild, Rang

Sei $F : X \rightarrow Y$ eine lineare Abbildung

$$\dim X = n \quad \dim Y = m \quad (229)$$

Das Bild von F ist $\text{Im}F : \{F(x) | x \in \underline{X}\} \subseteq Y$

Definition 53. Der Kern (Kernel) von F , $\ker F$, ist das Urbild von $0_y \in Y$.

$$\ker F := \{x \in \underline{X} | F(x) = 0_y\} \subseteq X \quad (230)$$

Example 14.

$$\mathcal{D} = \mathcal{P}_2 \rightarrow \mathcal{P}_2 \quad (231)$$

$$a + bt + ct^2 \mapsto b + 2ct \quad (232)$$

$$\ker \mathcal{D} := \{p(t) | \mathcal{D}(p(t)) = 0\} = \{a + bt + ct^2 | b + 2ct = 0\} \quad (233)$$

$$= \{a + bt + ct^2 | b = 0 \text{ und } c = 0\} = \{a | a \in \mathbb{R}\} \quad (234)$$

$$= \mathbb{R} = \text{alle konstanten Polynomen} \quad (235)$$

$$\text{Im} \mathcal{D} = \{D(p(t)) | p(t) \in \mathcal{P}_2\} = \{b + 2ct | b, c \in \mathbb{R}\} = \mathcal{P}_1 = \{A + Bt | A, B \in \mathbb{R}\} \quad (236)$$

$$\ker F = \{x \in \underline{X} | F(x) = 0_y\} = \left\{ (a, b) \in \mathbb{R}^2 | \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \right\} = \{(0, 0)\} \quad (237)$$

$$\ker F = \{0_{\underline{x}}\} \quad (238)$$

$$\text{Bild} F = \left\{ F \begin{pmatrix} a \\ b \end{pmatrix} | a, b \in \mathbb{R} \right\} = \left\{ \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} | a, b \in \mathbb{R} \right\} = \text{span} \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right\} \quad (239)$$

$$\text{Bild} F = \left\{ a \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + b \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} | a, b \in \mathbb{R} \right\} \quad (240)$$

Lemma 5.4

$\ker F$ ist ein Unterraum von \underline{X} und $\text{Im}F$ ist ein Unterraum von Y .

Proof 5.4

Es seien $x_1, x_2 \in \ker F$

Zu zeigen:

$$1. x_1 + x_2 \in \ker F$$

$$2. \alpha x_1 \in \ker F$$

$$x_1 \in \ker F \Rightarrow F(x_1) = 0 \quad (241)$$

$$x_2 \in \ker F \Rightarrow F(x_2) = 0 \quad (242)$$

$$F(x_1 + x_2) \underset{\substack{= \\ F \text{ ist linear}}}{=} F(x_1) + F(x_2) \underset{x_1, x_2 \in \ker F}{=} 0_y + 0_y = 0 \quad (243)$$

$$F(x_1 + x_2) = 0 \Rightarrow x_1 + x_2 \in \ker F \quad (244)$$

$$F(\alpha x_1) \underset{\substack{= \\ F \text{ ist linear}}}{=} \alpha F(x_1) \underset{x_1 \in \ker F}{=} \alpha 0 = 0 \Rightarrow \alpha x_1 \in \ker F \quad (245)$$

□

⇒ $\ker F$ ist ein Unterraum von \underline{X} .

Übung: DmF ist ein Unterraum von Y .

Lemma 5.5

Sei $F : X \rightarrow Y$ linear.

Ist U ein Unterraum von \underline{X} , so ist dessen Bild:

$$F(U) := \{F(u) | u \in U\} \subset Y \quad (246)$$

ein Unterraum von Y .

Ist W ein Unterraum von ImF , so ist dessen Urbild:

$$F^{-1}W := \{x \in \underline{X} | Fx \in W \subset ImF\} \quad (247)$$

ein Unterraum von \underline{X} .

$$\ker F = F^{-1}(\{0_y\}) \quad ImF = F(\underline{X}).$$

Example 15. Sei A eine $m \times n$ Matrix.

Sei A , durch die Matrix A definierte Abbildung.

$$A : \mathbb{E}^n \rightarrow \mathbb{E}^m \quad (248)$$

$$x \mapsto Ax \quad (249)$$

Dann ist $\ker A = \{x \in \mathbb{E}^n | Ax = 0\} = \mathcal{L}_0$ = Lösungsmenge des homogenen linearen Gleichungssystems $Ax = 0$

\mathcal{L}_0 heisst Nullraum (Nullspace) wird mit $\mathcal{N}(A)$ bezeichnet.

$$ImA := \{x | x \in \mathbb{E}^n\} \subset \mathbb{E}^m \quad A = \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_n \\ | & \cdots & | \end{pmatrix} \quad (250)$$

$$= \{\bar{a}_1 x_1 + \dots + \bar{a}_n x_n | x_i \in \mathbb{E}\} \quad (251)$$

$$= span\{\bar{a}_1, \dots, \bar{a}_n\} = \text{von den Kolonnen von } A \text{ aufgespannte Unterraum} \quad (252)$$

Dieser Unterraum heisst auch Kolonenraum von A (column space of A) und ist mit $\mathcal{R}(A)$ bezeichnet.

Das Lineare Gleichungssystem $Ax = b$ ist lösbar

$$\Leftrightarrow \exists x_i \in \mathbb{E} \text{ so dass } \bar{a}_1 x_1 + \dots + \bar{a}_n x_n = b \quad (253)$$

$$\Leftrightarrow b \in span\{\bar{a}_1, \dots, \bar{a}_n\} = \mathcal{R}(A) = Im(A) \quad (254)$$

$Ax = b$ ist lösbar ⇔ b liegt im Kolonenraum von A .

Frage: Wann ist eine lineare Abbildung injektive?

Theorem 5.6

$F : x \rightarrow y$ ist eine lineare Abbildung. Es ist genau dann injektive, wenn $\ker F = \{0\}$

Proof 5.6

Sei $F : x \rightarrow y$ eine lineare Abbildung.

$F(0_x) = 0_y$ für jede lineare Abbildung.

Da $F(0_x) = F(0_x + 0_x) = F(0_x) + F(0_x)$.

$\Rightarrow F(0_x) = 0_y$.

Falls F injektive ist, so gilt $F(0_x) = 0_y$, $0_y \in Y$

hat ein einziges Urbild, nämlich 0_x , $\ker F = \{0_x\}$

F injective $\Rightarrow \ker F = \{0_x\}$

Umgekehrt, sei $\ker F = \{0\}$ und $x, x' \in X$ und $x \neq x'$

Dann folgt:

$$F(x - x') \neq 0 \Rightarrow F(x) - F(x') \neq 0 \Rightarrow F(x) \neq F(x') \quad (255)$$

Aus $x \neq x'$ folgt dass $F(x) \neq F(x')$ $\Rightarrow F$ ist injective. \square

Es gilt folgender grundlegender Zusammenhang zwischen $\ker F \subset X$ und $\text{Im } F \subset Y$ für eine lineare Abbildung $F : X \rightarrow Y$.

Theorem 5.7: Dimensionsformel

Sei $F : \underline{X} \rightarrow Y$ eine lineare Abbildung.

Dann gilt:

$$\dim(\underline{X}) = \dim(\ker F) + \dim(\text{Im } F) \quad (256)$$

The following was presented in the lecture on 09. November 2018.

Definition 54. Der Rang einer Linearen Abbildung F ist gleich der Dimension des Bildraums von F .

$$\text{Rang } F : \dim \text{Im } F \quad (257)$$

$$\dim X = \dim \ker F + \dim \text{Im } F = \dim \ker F + \text{Rang } F \quad (258)$$

Corollary 5.8

Es gelten die folgenden drei Äquivalenten

$$(1) F : X \rightarrow Y \text{ injektive} \Leftrightarrow \text{Rang } F = \dim X \quad (259)$$

$$\textcircled{2} F : X \rightarrow Y \text{ bijektive } \Leftrightarrow \text{Rang } F = \dim X = \dim Y \quad (260)$$

$$\textcircled{3} F : X \rightarrow Y \text{ bijektive } \Leftrightarrow \text{Rang } F = \dim X \Leftrightarrow \ker F = \{0\} \quad (261)$$

Proof 5.8

$$\begin{aligned} \textcircled{1} \quad F \text{ injektive} &\Leftrightarrow \ker F = \{0\} \\ &\Leftrightarrow \dim \ker F = 0 \\ &\Leftrightarrow \dim X = \dim \ker F + \dim \text{Im } F = \dim \text{Im } F = \text{Rang } F \\ \textcircled{2} \quad F \text{ ist surjektive} &\Leftrightarrow \text{Im } F = Y \Leftrightarrow \dim \text{Im } F = \dim Y \\ &\Leftrightarrow \text{rang } F = \dim Y \\ F \text{ ist injektive} &\Leftrightarrow \text{rang } F = \dim \overline{X} \\ F \text{ ist bijektive} &\Leftrightarrow \text{rang } F = \dim X = \dim Y \end{aligned}$$

□

Definition 55. Zwei Vektorräume \underline{X} und Y heißen isomorph wenn es einen Isomorphismus $F : X \rightarrow Y$ gibt.

Theorem 5.9

Zwei Vektorräume endlicher Dimension sind isomorph $\Leftrightarrow \dim X = \dim Y$

Corollary 5.10

Sind $F : X \rightarrow Y, G : Y \rightarrow Z$ lineare Abbildungen (wobei $\dim X < \infty, \dim Y < \infty$) so gilt:

1. $\text{Rang } G \circ F \leq \min\{\text{Rang } F, \text{Rang } G\}$
2. G injektiv $\Rightarrow \text{Rang } G \circ F = \text{Rang } F$
3. F surjektiv $\Rightarrow \text{Rang } G \circ F = \text{Rang } G$

5.3 Matrizen als lineare Abbildungen

Theorem 5.11

A ist $m \times n$ Matrix, $A : \mathbb{E}^n \rightarrow \mathbb{E}^m \quad x \mapsto Ax$

$$\ker A = \mathcal{N}(A) = \mathbb{L}_0 = \{x | Ax = 0\} \quad (262)$$

$$\text{Im } A = \text{span}\{\bar{a}_1, \bar{a}_2, \dots, \bar{a}_n\} = R(A) = \text{Kolonenraum von } A \quad (263)$$

$$Ax = b \text{ ist lösbar} \Leftrightarrow b \in \text{span}\{\bar{a}_1, \bar{a}_2, \dots, \bar{a}_n\} \Leftrightarrow b \in \text{Im } A = R(A) \quad (264)$$

$$\ker A = \mathcal{N}(A) \subset \mathbb{E}^n \quad (265)$$

$$\text{Im } A = R(A) \subset \mathbb{E}^m \quad (266)$$

Wir wollen die Dimension der Unterräume $\mathcal{N}(A)$ und $R(A)$ bestimmen.

$\ker A$ = Lösungsmenge des Homogenen LGS $Ax = 0$

Diese Lösungsmenge hat $n - r$ frei wählbare Parameter. Dabei r =Anzahl der Pivots die beim Gaussverfahren auftreten.

Ohne Einschränkung der Allgemeinheit nehmen wir an, dass die freien Parameter am Schluss auftreten.

Nämlich, $x_{r+1}, x_{r+2}, \dots, x_n$ sind frei wählbar.

$$A \xrightarrow[\text{GaussVerfahren}]{} R = \begin{pmatrix} r_{11} & x & x & x & x & x \\ 0 & r_{22} & x & x & x & x \\ 0 & 0 & r_{33} & x & x & x \\ 0 & 0 & 0 & r_{44} & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (267)$$

Man kann auf genau $n - r$ Arten einen freien Parameter gleich 1 und alle anderen gleich Null setzen.

Wir bilden $n - r$ Lösungsvektoren von $Ax = 0$ durch Rückwärtseinsetzen im Gauss Endschema. Erster Lösungsvektor u_1 : Setze $x_{r+1} = 1, x_{r+2} = x_{r+3} = \dots = x_n = 0$

$$u_1 = \begin{pmatrix} ? \\ \dots \\ ? \\ 1 \\ 0 \\ \dots \\ 0 \end{pmatrix} \quad \begin{array}{l} \text{Mit Rückwärtseinsetzen bestimmen.} \\ r+1 \end{array} \quad (268)$$

Zweiter Lösungsvektor u_2 : Setze $x_{r+1} = 0, x_{r+2} = 1, x_{r+3} = x_{r+4} = \dots = x_n = 0$

$$u_2 = \begin{pmatrix} ? \\ \dots \\ ? \\ 0 \\ 1 \\ 0 \\ \dots \\ 0 \end{pmatrix} \quad \begin{array}{l} \text{Mit Rückwärtseinsetzen bestimmen.} \\ r+1 \\ r+2 \end{array} \quad (269)$$

(n-r) Lösungsschar

$$u_{n-r} = \begin{pmatrix} ? \\ \dots \\ ? \\ 0 \\ 0 \\ \dots \\ 1 \end{pmatrix} \quad \begin{array}{l} \text{Mit Rückwärtseinsetzen bestimmen.} \\ r+1 \\ n \end{array} \quad (270)$$

Definition 56. Sei $A = \begin{pmatrix} \underline{a}_1 & \cdots & \underline{a}_m \\ \vdots & & \\ \underline{a}_n & \cdots & \end{pmatrix}$ mit $\underline{a}_1, \dots, \underline{a}_m \in \mathbb{E}$ die Zahlen von A.

Bezeichnen wir durch die Zeilenvektoren $\underline{a}_1, \dots, \underline{a}_m$ aufgespannte Unterraum als Zeilenraum (row space) von A

$$\text{Zeilenraum}(A) = \text{span}\{\underline{a}_1, \dots, \underline{a}_m\} \quad (271)$$

Theorem 5.13

Der Rang einer $m \times n$ Matrix A entspricht:

1. der Anzahl Pivotelemente bei der Reduktion von A auf Zeilenstufenform
2. dem Rang der linearen Abbildung $A : \mathbb{E}^n \rightarrow \mathbb{E}^m \quad x \mapsto Ax$ definiert als $\dim \text{Im } A$
3. der Dimension des Kolonnenraumes definiert als Anzahl linear unabhängiger Kolonnen von A (Kolonnenrang)
4. der Dimension des Zeilenraumes (Zeilenrang) definiert als Anzahl linear unabhängiger Zeilen von A.

Corollary 5.14

$$\text{Rang } A^T = \text{Rang } A^H = \text{Rang } A \quad (272)$$

Proof 5.13: (4)

Es sei R die Zeilenstufenform von A.

Jede Zeile der Matrix R ist eine Linearkombination der Zeilen von A.

Da jeder Eliminationsschritt beim Gaußverfahren rückgängig gemacht werden kann, ist auch jede Zeile von A eine Linearkombination der Zeilen von R.

Die Zeilen von A und die Zeilen von R spannen also den gleichen Raum auf.

Zeilenraum A = Zeilenraum von R

Aus der Zeilenstufenform R ist ersichtlich, dass die Zeilen von R einen r-dimensionalen Unterraum aufspannen.

$$R = \begin{pmatrix} r_{11} & x & x & x & x & x \\ 0 & r_{22} & x & x & x & x \\ 0 & 0 & r_{33} & x & x & x \\ 0 & 0 & 0 & r_{44} & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (273)$$

$$r_1 = \begin{pmatrix} r_{11} \\ x \\ x \\ x \\ x \\ x \end{pmatrix} \quad r_2 = \begin{pmatrix} 0 \\ r_{22} \\ x \\ x \\ x \\ x \end{pmatrix} \quad r_3 = \begin{pmatrix} 0 \\ 0 \\ r_{33} \\ x \\ x \\ x \end{pmatrix} \quad (274)$$

$$r = \dim \text{span}\{r_1, \dots, r_n\} = \dim \text{span}\{a_1, \dots, a_n\} = \text{Rang } A \quad (275)$$

$$\text{Rang } A = \dim(\text{Kolonnenraum}(A)) = \dim(\text{Zeilenraum}(A)) \quad (276)$$

□

Note 38. Beim folgenden muss man aufpassen:

$\dim \text{Im } A = \dim \text{Im } R$

aber $\text{Im } A \neq \text{Im } R$

Kolonenraum A \neq Kolonenraum R

Zeilenraum A = Zeilenraum R

Theorem 5.15

Für den Kolonenraum einer $m \times n$ Matrix A gilt:

$$ImA = BildA = R(A) = \text{span}\{\bar{a}_1, \dots, \bar{a}_n\} = \text{span}\{a_{n_1}, \dots, a_{n_r}\} \quad (277)$$

worin a_{n_1}, \dots, a_{n_r} die Pivotkolonnen von A sind.

Theorem 5.18

Für eine quadratische Matrix $A \in \mathbb{E}^{n \times n}$ sind folgende Aussagen äquivalent:

1. A ist invertierbar
2. A ist regulär
3. Rang $A = n$
4. Die n Kolonenvektoren von A sind linear unabhängig.
5. Die n Zeilenvektoren von A sind linear unabhängig.
6. $ImA = Q(A) = \mathbb{E}^n$
7. $\ker A = N(A) = \{0\}$
8. Die lineare Abbildung $A : \mathbb{E} \rightarrow \mathbb{E}^n$ ist ein Automorphismus
9. A ist die Transformationsmatrix einer Koordinatentransformation in \mathbb{E}^n

The following was presented in the lecture on 14. November 2018.

Theorem 5.16

Es seien $A \in \mathbb{E}^{m \times n}$, $B \in \mathbb{E}^{p \times m}$ Dann gilt:

1. $\text{Rang}BA \leq \min\{\text{Rang}B, \text{Rang}A\}$
2. $\text{Rang}B = m (\leq p) \Rightarrow \text{Rang}BA = \text{Rang}A$
3. $\text{Rang}A = m (\leq n) \Rightarrow \text{Rang}BA = \text{Rang}B$

Corollary 5.17

Es seien $A \in \mathbb{E}^{m \times m}$, $B \in \mathbb{E}^{m \times m}$ Dann gilt:

1. $\text{Rang}BA \leq \min\{\text{Rang}B, \text{Rang}A\}$
2. $\text{Rang}B = m \Rightarrow \text{Rang}BA = \text{Rang}A$

$$3. \text{ } \text{Rang}A = m \Rightarrow \text{Rang}BA = \text{Rang}B$$

5.4 Affine Räume und die allgemeine Lösung eines inhomogenen Gleichungssystems $Ax = b$

Definition 57. Sei U ein echter Unterraum eines Vektorraumes V und zusätzlich $u_0 \in V$. Die Menge $u_0 + U := \{u_0 + u \mid u \in U\}$ heisst ein affiner Teilraum (affine subspace).

Ist $F : X \rightarrow Y$ eine lineare Abbildung und $y_0 \in Y$ so ist $H : X \rightarrow y_0 + Y$

$x \mapsto y_0 + F(x)$ eine affine Abbildung.

Note 39. Ein affiner Teilraum ist im Allgemeinen kein Unterraum. Falls $u_0 \in U$ ist, dann ist $u_0 + U$ kein Unterraum.

Ebenso ist eine affine Abbildung nicht linear. Falls $y_0 \neq 0$ ist, dann $H(0) = y_0 \neq 0$.

$\Rightarrow H$ ist keine lineare Abbildung.

Wir betrachten hier den Zusammenhang mit dem linearen Gleichungssystems. Sei $Ax = b$ und $\mathcal{L}_b = \{x \mid Ax = b\}$ (die Lösungsmenge). \mathcal{L}_b ist kein Unterraum, aber \mathcal{L}_b ist ein affiner Raum.

Theorem 5.19

Es sei x_p irgendeine Lösung des inhomogenen Systems $Ax = b$. Und es bezeichne \mathcal{L}_0 den Lösungsraum des homogenen Systems $Ax = 0$. Dann ist die Lösungsmenge L_b von $Ax = b$ gleich dem affinen Teilraum $\mathcal{L}_b = x_p + \mathcal{L}_0$

x_p heisst Partikulär Lösung. Daher $\forall x \in \mathcal{L}_b$ gilt: $x = x_p + x_h$ wobei x_p eine spezielle Lösung von $Ax = b$ ist. x_h eine Lösung von $Ax = 0$

Wie findet man x_p

Eine spezielle Lösung x_p von $Ax = b$ finden wir mit Rückwärtseinsetzen in die Zeilenstufenform in der alle freien Variablen auf Null gesetzt sind.

Proof 5.19

(\Rightarrow) Aus $Ax_0 = b$ und $Ax = 0 \Rightarrow A(x_0 + x) = b \Rightarrow x_0 + \mathcal{L}_0 \subseteq \mathcal{L}_b$

(\Leftarrow) Umgekehrt ist x_1 (noch) irgendeine Lösung von $Ax = b$ ($x_1 \in \mathcal{L}_b$)

So gilt:

$$A(x_0 - x_1) = Ax_0 - Ax_1 = b - b = 0 \Rightarrow x_0 - x_1 \in \mathcal{L}_0 \Rightarrow x_1 \in x_0 + \mathcal{L}_0 \quad (278)$$

$$\Rightarrow \mathcal{L}_b \subseteq x_0 + \mathcal{L}_0 \Rightarrow x_0 + \mathcal{L}_0 = \mathcal{L}_b \quad (279)$$

□

5.5 Die Abbildungsmatrix bei Koordinatentransformationen

5.5.1 Basiswechsel

Falls $Y = X$, $B = C$, $B' = C'$, dann heisst der Übergang $A \rightarrow B = T^{-1}AT$ Ähnlichkeitstransformation (Similarity transformation).

Example 16. $D : \mathbb{P}_2 \rightarrow \mathbb{P}_2$

$$a + bt + ct^2 \mapsto b + 2ct$$

$$B = \{1, t, t^2\} \quad C = \{1, t, t^2\}$$

$$B' = \{1, t^2, t\} \quad C = \{1, t+1, t^2\}$$

Sei S der Basiswechsel von C nach C':

$$S = ([1]_c [1+t]_c [t^2]_c) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (280)$$

Sei T der Basiswechsel von B nach B':

$$T = ([1]_B [t^2]_B [t]_B) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad (281)$$

$$Mat(D, B, C) = ([D(1)]_c [D(t)]_c [D(t^2)]_c) = ([0]_c [1]_c [2t]_c) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix} = A \quad (282)$$

$$Mat(D, B', C') = ([D(1)]_{c'} [D(t^2)]_{c'} [D(t)]_{c'}) = ([0]_{c'} [2t]_{c'} [1]_{c'}) = \begin{pmatrix} 0 & -2 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix} = B \quad (283)$$

$$B = S^{-1}AT \quad (284)$$

$$\begin{pmatrix} 0 & -2 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad (285)$$

6 Vektorräume mit Skalarprodukt

Note 40. Erinnerung: $x, y \in \mathbb{R}^n$ oder $\mathbb{C}^n(\mathbb{E}^n)$

Das euklidische Skalarprodukt (innere Produkt) ist die Zahl $\langle x, y \rangle$ definiert durch:

$$\langle x, y \rangle := x^H y = \sum_{k=1}^n \bar{x}_k y_k \quad (286)$$

$$\text{Für reelle Vektoren } x, y \in \mathbb{R}^n, \langle x, y \rangle := x^T y = \sum_{k=1}^n x_k y_k$$

Die Euklidische Norm von $x \in \mathbb{E}^n$ (auch 2-Norm) ist die nicht negative reelle Zahl $\|x\|$ definiert durch:

$$\|x\| := \sqrt{\langle x, x \rangle} \quad (287)$$

Verallgemeinerung:

Definition 58. Eine Norm in einem Vektorraum V über \mathbb{E} ist eine Funktion $\|\cdot\| : V \rightarrow \mathbb{R}$ und $x \mapsto \|x\|$ mit den Eigenschaften N1, N2, N3.

N1 positiv definit $\|x\| \geq 0 \quad \forall x \in V$
 $\|x\| = 0 \Rightarrow x = 0$

N2 homogen: $\|\alpha x\| = |\alpha| \|x\| \quad \forall x \in V, \alpha \in \mathbb{E}$

N3 Dreiecksungleichung: $\|x + y\| \leq \|x\| + \|y\| \quad \forall x, y \in V$

Ein Vektorraum mit einer Norm heisst normierter Vektorraum (normed vector space) oder normierter Raum (normed linear space)

Example 17.

$$V = \mathbb{P}_n^{[0,1]} = \{p : [0, 1] \rightarrow \mathbb{R}, p(x) = a_0 + a_1 x + \dots + a_n x^n\} \quad (288)$$

$$\|p\|_\infty := \max_{x \in [0,1]} |p(x)| \quad (289)$$

N1, N2 als Übung lösen

$$N3 : \|p + q\|_\infty \stackrel{?}{\leq} \|p\|_\infty + \|q\|_\infty \quad (290)$$

$$\|p + q\|_\infty \stackrel{\text{def}}{=} \max_{x \in [0,1]} |p(x) + q(x)| \quad (291)$$

$$\leq \max_{x \in [0,1]} (|p(x)| + |q(x)|) \quad (292)$$

$$\leq \max_{x \in [0,1]} (|p(x_1)|) + \max_{x \in [0,1]} (|q(x_2)|) \quad (293)$$

$$\stackrel{\text{def}}{=} \|p\|_\infty + \|q\|_\infty \Rightarrow N3\checkmark \quad (294)$$

Die Norm $\|\cdot\|_\infty$ nennt man auch L_∞ - Norm oder auch Maximumnorm.

Definition 59. Ein Skalarprodukt (inner product) in einem Vektorraum über \mathbb{E} ist eine Funktion: $\langle \cdot, \cdot \rangle : V \times V \rightarrow \mathbb{E}$
 $x, y \mapsto \langle x, y \rangle$ mit den Eigenschaften

S1 Linear im zweiten Faktor

$$\langle x, y + z \rangle = \langle x, y \rangle + \langle x, z \rangle \quad \forall x, y, z \in V \quad (295)$$

$$\langle x, \alpha y \rangle = \alpha \langle x, y \rangle \quad \forall x, y \in V, \alpha \in \mathbb{E} \quad (296)$$

S2 Falls $\mathbb{E} = \mathbb{R}$ ist es symmetrisch: $\langle x, y \rangle = \langle y, x \rangle \quad \forall x, y \in V$

Falls $\mathbb{E} = \mathbb{C}$ ist es hermitisch: $\langle x, y \rangle = \overline{\langle y, x \rangle} \quad \forall x, y \in V$

S3 Positiv definit: $\langle x, x \rangle \geq 0 \quad \forall x \in V$

$$\langle x, x \rangle = 0 \Rightarrow x = 0$$

V mit $\langle \cdot, \cdot \rangle$ heisst ein Vektorraum mit Skalarprodukt (inner product space). Wenn $\dim V < \infty$ nennt man V auch:

1. falls $\mathbb{E} = \mathbb{R}$: Euklidischer Vektorraum oder orthogonaler Vektorraum.

2. falls $\mathbb{E} = \mathbb{C}$: unitärer Vektorraum (unitary vector space)

Note 41. 1. Wir können die Addition und Multiplikation mit Skalar verbinden:

2. Faktor:

$$\langle x, \alpha y, \beta z \rangle \stackrel{S1}{=} \langle x, \alpha y \rangle + \langle x, \beta z \rangle \stackrel{S1}{=} \alpha \langle x, y \rangle + \beta \langle x, z \rangle \quad (297)$$

1. Faktor

$$\langle \alpha w + \beta x, y \rangle \stackrel{S2}{=} \overline{\langle y, \alpha w + \beta x \rangle} = \overline{\alpha \langle y, w \rangle + \beta \langle y, x \rangle} = \overline{\alpha} \overline{\langle y, w \rangle} + \overline{\beta} \overline{\langle y, x \rangle} \stackrel{S2}{=} \overline{\alpha} \langle w, y \rangle + \overline{\beta} \langle x, y \rangle \quad (298)$$

2. Wenn $\mathbb{E} = \mathbb{R}$, dann ist $\overline{\alpha} = \alpha$ und $\overline{\beta} = \beta$, sodass das Skalarprodukt auch im 1. Argument linear ist.

Definition 60. In einem Vektorraum V mit Skalarprodukt können wir die Norm oder Länge eines Vektors $x \in V$ definieren: $\|x\| := \sqrt{\langle x, x \rangle_v}$

Man kann zeigen, dass N1-N3 erfüllt sind (siehe Satz 2.12)

Example 18. \mathbb{R}^n mit dem Skalarprodukt $\langle x, y \rangle = x^T y$ ist ein Euklidischer Vektorraum.
 \mathbb{C}^n mit dem Skalarprodukt $\langle x, y \rangle = x^H y$ ist ein unitärer Vektorraum.

Example 19. “Gewichtetes” Skalarprodukt in \mathbb{E}^n :

Sei W eine diagonale Matrix, $w_{jj} > 0 \quad \forall j = 1, \dots, n$

$$\langle x, y \rangle := x^H W y$$

$$\|x\| = \sqrt{\langle x, x \rangle} = \sqrt{x^H W x} = \sqrt{\sum_{j=1}^n w_{jj} |x_j|^2} \quad (299)$$

The following was presented in the lecture on 16. November 2018.

Example 20. Der Vektorraum der stetigen Funktion $f : [a, b] \rightarrow \mathbb{R} : C[a, b]$

Definiere $\langle f, g \rangle := \int_a^b f(x)g(x) dx$

$$L_2 - \text{Norm} \quad \|f\| = \sqrt{\langle f, f \rangle} = \sqrt{\int_a^b f(x)f(x) dx} = \sqrt{\int_a^b |f(x)|^2 dx} \quad (300)$$

$$F : [0, 1] \rightarrow \mathbb{R} \quad f(x) = 3 + x^2 \quad (301)$$

$$g : [0, 1] \rightarrow \mathbb{R} \quad g(x) = x^2 \quad (302)$$

$$\langle f, g \rangle = \int_0^1 (3 + x^2)x^2 dx = \int_0^1 3x^2 + x^4 dx = x^3 + \frac{x^5}{5} \Big|_0^1 = \frac{6}{5} \quad (303)$$

Theorem 6.0

Sei V ein Vektorraum über \mathbb{E} , $\dim V < \infty$. Dann kann man immer ein Skalarprodukt in V definieren.

Proof 6.0

Sei $\{b_1, b_2, \dots, b_n\} \subset V$ eine Basis. Jeder Vektor $x \in V$ hat eine eindeutige Darstellung als

$$x = \sum_{j=1}^n \alpha_j b_j, \quad \alpha_j \in \mathbb{E} \quad (304)$$

Definieren wir das Skalarprodukt $\langle x, y \rangle = \sum_{j=1}^n \overline{\alpha_j} \gamma_j$ wobei $y = \sum_{j=1}^n \gamma_j b_j$.

S_1, S_2, S_3 sind erfüllt. □

Theorem 6.1: Cauchy-Schwartz Ungleichung

$$\forall x, y \in V \quad |\langle x, y \rangle| \leq \|x\| \cdot \|y\| \quad \|x\| = \sqrt{\langle x, x \rangle} \quad (305)$$

Das Gleichheitszeichen gilt genau dann, wenn x und y linear abhängig sind.

Definition 61. Der Winkel $\phi = \angle(x, y)$ ($0 \leq \phi \leq \pi$) zwischen $x, y \in V$ ist gegeben durch:

$$\angle(x, y) = \phi := \arccos \frac{\langle x, y \rangle}{\|x\| \cdot \|y\|} \text{ falls } \mathbb{E} = \mathbb{R} \quad (306)$$

$$\angle(x, y) = \phi := \arccos \frac{\operatorname{Re}(\langle x, y \rangle)}{\|x\| \cdot \|y\|} \text{ falls } \mathbb{E} = \mathbb{C} \quad (307)$$

Definition 62. Zwei Vektoren $x, y \in V$ sind zueinander orthogonal oder senkrecht (perpendicular) falls $\langle x, y \rangle = 0 \quad x \perp y$

Zwei Teilmengen $M, N \subseteq V$ sind zueinander orthogonal falls $\langle x, y \rangle = 0 \quad \forall x \in M, y \in N \quad M \perp N$

Note 42. Uns werden Unterräume M, N interessieren.

Der Nullraum $\{0\}$ ist orthogonal zu allen Unterräumen.

Der Nullvektor $0 \perp x \quad \forall x \in V$.

Theorem 6.2: Pythagoras

V ist Vektorraum mit Skalarprodukt.

$$x, y \in V : \quad x \perp y \Rightarrow \|x \pm y\|^2 = \|x\|^2 + \|y\|^2 \quad (308)$$

6.1 Orthonormalbasen (6.3)

Theorem 6.3

Eine Menge M von paarweise orthogonalen Vektoren ist linear unabhängig falls $0 \notin M$.

Proof 6.3

Es seien $\{x_1, x_2, \dots, x_m\} \in M$ und $\{\gamma_1, \gamma_2, \dots, \gamma_m\} \in \mathbb{E}$ sodass

$$\sum_{k=1}^m \gamma_k x_k = 0 \quad |\text{multiplizieren von links mit } x_j \quad j = 1, \dots, m \quad (309)$$

$$\forall j=1, \dots, m \quad 0 = \langle x_j, \sum_{k=1}^m \gamma_k x_k \rangle = \sum_{k=1}^m \underbrace{\gamma_k \langle x_j, x_k \rangle}_{=0 \text{ falls } j \neq k} = \gamma_j \langle x_j, x_j \rangle \quad (310)$$

$$\Rightarrow 0 = \gamma \underbrace{\langle x_j, x_j \rangle}_{>0 \text{ (S3), weil } x_j \neq 0} \Rightarrow \gamma_j = 0 \quad \forall j=1, \dots, m \quad (311)$$

□

Wenn $\dim V = n$, dann bilden n paarweise orthogonale, von 0 verschiedene Vektoren eine Basis von V .

Definition 63. Eine Basis $\{b_1, b_2, \dots, b_n\}$ heisst orthogonal falls $\langle b_k, b_l \rangle = 0 \quad \forall_{k,l} \quad k \neq l$
 Eine Basis heisst orthonormal wenn zusätzlich alle Basisvektoren die Länge 1 haben. $\langle b_k, b_k \rangle = 1 \quad \langle b_k, b_l \rangle = \delta_{kl}$

Note 43. Wir können aus einer orthogonalen Basis $\{b_1, b_2, \dots, b_n\}$ eine orthonormale Basis gewinnen indem wir jeden Basisvektor normieren: $\tilde{b}_j := \frac{b_j}{\|b_j\|}$

Dann gilt: $\langle \tilde{b}_k, \tilde{b}_l \rangle = \delta_{kl}$

Theorem 6.4

Ist V ein Vektorraum mit Skalarprodukt, $\dim V = n$ und $\{b_1, b_2, \dots, b_n\}$ eine Orthonormalbasis, so gilt für alle $x \in V$:

$$x = \sum_{k=1}^n \langle b_k, x \rangle b_k \quad (312)$$

Daher die Koordinaten von x bezüglich einer Orthonormalbasis sind einfach

$$\xi_k = \langle b_k, x \rangle \quad (313)$$

Proof 6.4

$\{b_1, b_2, \dots, b_n\}$ ist eine Basis $\Rightarrow \forall x \in V$ gibt es eine Darstellung $x = \sum_{k=1}^n \xi_k b_k$, mit eindeutig bestimmten Koordinaten ξ_k . Multiplizieren wir von links mit b_j :

$$\langle b_j, x \rangle = \langle b_j, \sum_{k=1}^n \xi_k b_k \rangle \stackrel{S1}{=} \sum_{k=1}^n \xi_k \underbrace{\langle b_j, b_k \rangle}_{=\delta_{jk}} = \xi_j \quad (314)$$

$$\Rightarrow \forall_{j=1, \dots, n} : \xi_j = \langle b_j, x \rangle \quad (315)$$

□

Theorem 6.5: Parsevalsche Formel

Sei $\{b_1, b_2, \dots, b_n\}$ eine Orthonormalbasis von V und $x, y \in V$.

$$\xi_k := \langle b_k, x \rangle_v \quad \eta_k := \langle b_k, y \rangle_v \quad k = 1, \dots, n \text{ (Koordinaten)} \quad (316)$$

Dann gilt:

$$\langle x, y \rangle = \sum_{k=1}^n \overline{\xi_k} \eta_k = \xi^H \eta = \langle \xi, \eta \rangle_{\mathbb{E}^n} \quad (317)$$

wobei $\xi = (\xi_1, \dots, \xi_n)^T$, $\eta = (\eta_1, \dots, \eta_n)^T \in \mathbb{E}^n$ die Koordinatenvektoren von x beziehungsweise y sind.

$$\text{Folge: } \|x\| = \|\xi\|, \quad \sphericalangle(x, y) = \sphericalangle(\xi, \eta) \quad x \perp y \Leftrightarrow \xi \perp \eta \quad (318)$$

Proof 6.5

$$\langle x, y \rangle_v = \left\langle \sum_{k=1}^n \xi_k b_k, \sum_{l=1}^n \eta_l b_l \right\rangle_v \quad (319)$$

$$= \sum_{k=1}^n \overline{\xi_k} \langle b_k, \sum_{l=1}^n \eta_l b_l \rangle_v \quad (320)$$

$$= \sum_{k=1}^n \sum_{l=1}^n \overline{\xi_k} \eta_l \underbrace{\langle b_k, b_l \rangle_v}_{\delta_{kl}} \quad (321)$$

$$= \sum_{k=1}^n \overline{\xi_k} \eta_k = \xi^H \eta = \langle \xi, \eta \rangle_{\mathbb{E}^n} \quad (322)$$

□

Gibt es immer eine Orthonormalbasis? Ja! Sei V ein Vektorraum mit $\dim V = m$, dann wissen wir dass es in V immer eine Basis mit m Vektoren gibt. Der folgende Algorithmus erzeugt aus jeder Basis eine orthonormale Basis.

Algorithmus 6.1: Gram-Schmidt-Orthogonalisierungsverfahren

Seien $\{a_1, a_2, \dots, a_m\} \subseteq V$ lineare unabhängig.

Berechne m neue Vektoren

$$\{b_1, \dots, b_m\} : \quad b_1 := \frac{a_1}{\|a_1\|} \quad (323)$$

für $k=2,3,\dots,m$:

$$\tilde{b}_k := a_k - \sum_{j=1}^{k-1} \langle b_j, a_k \rangle b_j \quad (324)$$

$$b_k := \frac{\tilde{b}_k}{\|\tilde{b}_k\|} \quad (325)$$

Theorem 6.6

Die Vektoren $\{b_1, b_2, \dots, b_m\}$ sind normiert und paarweise orthogonal.

Nach k Schritten gilt: $\text{span}\{a_1, a_2, \dots, a_k\} = \text{span}\{b_1, b_2, \dots, b_k\}$.

Ist $\{a_1, a_2, \dots, a_m\}$ eine Basis von V , so ist $\{b_1, b_2, \dots, b_m\}$ eine Orthonormalbasis von V .

Proof 6.6

$\{b_1, b_2, \dots, b_m\}$ sind normiert ✓

Zu zeigen: $\langle b_l, b_k \rangle = 0 \quad \forall k \neq l$

Per Induktion (über m):

Verankerung: $k=1$ - leere Aussage ✓

Schritt: Nehmen wir an, dass $\{b_1, b_2, \dots, b_{k-1}\}$ paarweise orthogonal sind.

$$0 \stackrel{?}{=} \langle b_l, b_k \rangle = \langle b_l, \frac{\tilde{b}_k}{\|\tilde{b}_k\|} \rangle = \frac{1}{\|\tilde{b}_k\|} \left\langle b_l, a_k - \sum_{j=1}^{k-1} \langle b_j, a_k \rangle b_j \right\rangle \quad (326)$$

$$= \frac{1}{\|\tilde{b}_k\|} \left(\langle b_l, a_k \rangle - \sum_{j=1}^{k-1} \langle b_j, a_k \rangle \underbrace{\langle b_l, b_j \rangle}_{\delta_{lj}} \right) = \frac{1}{\|\tilde{b}_k\|} (\langle b_l, a_k \rangle - \langle b_l, a_k \rangle) = 0 \quad (327)$$

$\Rightarrow \{b_1, b_2, \dots, b_k\}$ sind paarweise orthogonal.

Durch Induktion ist klar, dass $b_k \in \text{span}\{a_1, \dots, a_k\}$

$$\Rightarrow \text{span}\{b_1, \dots, b_k\} \subseteq \{a_1, \dots, a_k\} \forall k \quad (328)$$

$\{b_1, b_2, \dots, b_k\}$ sind paarweise orthonormal $\Rightarrow \{b_1, b_2, \dots, b_k\}$ sind linear unabhängig, also $\dim \text{span}\{b_1, b_2, \dots, b_k\} = k \Rightarrow \text{span}\{b_1, b_2, \dots, b_k\} = \text{span}\{a_1, a_2, \dots, a_k\}$ \square

Corollary 6.7

In jedem Vektorraum mit Skalarprodukt V , $\dim V < \infty$, gibt es eine orthonormierte Basis.

Example 21.

$$\mathbb{R}^2 \quad a_1 = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \quad a_2 = \begin{pmatrix} 0 \\ 2 \end{pmatrix} \quad (329)$$

$$b_1 = \frac{a_1}{\|a_1\|} = \frac{1}{\sqrt{2^2 + 2^2}} \begin{pmatrix} 2 \\ 2 \end{pmatrix} = \frac{1}{\sqrt{8}} \begin{pmatrix} 2 \\ 2 \end{pmatrix} = \frac{1}{2\sqrt{2}} \begin{pmatrix} 2 \\ 2 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad (330)$$

$$\tilde{b}_2 = a_2 - \langle b_1, a_2 \rangle b_1 = \begin{pmatrix} 0 \\ 2 \end{pmatrix} - \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}^T \begin{pmatrix} 0 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (331)$$

$$b_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad (332)$$

The following was presented in the lecture on 21. November 2018.

6.2 Zusammenfassung

Normierte Vektorräume $\|\cdot\|$ (N1) (N2) (N3)

Vektorraum mit Skalarprodukt $\langle \cdot, \cdot \rangle$ (S1) (S2) (S3)

Die induzierte Norm:

$$\|x\| = \sqrt{\langle x, x \rangle} \quad (333)$$

$x \perp y \Leftrightarrow \langle x, y \rangle = 0$ Orthogonale Vektoren

Orthogonale Vektoren $\neq 0$ sind linear unabhängig

Orthogonale Basen Es gibt Basen für Vektorraum, die paarweise orthogonal sind.

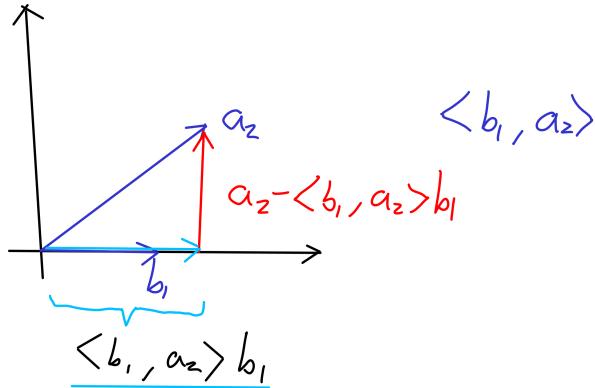
Alle Basenvektoren normiert \rightarrow orthonormale Basis

Gram-Schmidt Algorithmus Aus einer Menge von linear unabhängigen Vektoren können wir immer eine orthogonale Menge mit dem gleichen "span" bekommen!

6.3 Warum orthonormale Basen?

- Die Koordinaten bezüglich orthonormalen Basen sind einfach zu berechnen.
 $\{b_1, b_2, \dots, b_n\}$ orthonormale Basis, $x \in V$

$$\Rightarrow x = \sum_{k=1}^n \langle b_k, x \rangle b_k \quad (334)$$



6.4 Orthogonale Komplemente

Erinnerung

V ist ein Vektorraum, $U_1, U_2 \subset V$ Unterräume.

U_1, U_2 heissen komplementär wenn $\forall x \in V$ es eine eindeute Darstellung gibt als:

$$x = x_1 + x_2, \quad x_1 \in U_1, \quad x_2 \in U_2 \quad (335)$$

$$V = U_1 \oplus U_2 \text{ (direkte Summe)} \quad (336)$$

Sei $U \subset V$ ein Unterraum von V . Wir können eine orthonormale Basis $\{b_1, b_2, \dots, b_l\}$ für U finden. Ergänzen wir diese Basis zu einer Basis von V : $\{\underbrace{b_1, \dots, b_l}_{\text{orthonormale Basis von } U}, \underbrace{a_{l+1}, \dots, a_n}_{\text{orthonormale Basis von } U^\perp}\}$ führen wir Gram-Schmidt aus.

Basis von V .

$\underbrace{\{b_1, \dots, b_l\}}_U, \underbrace{\{a_{l+1}, \dots, a_n\}}_{U^\perp}$ Das Gesamte ist eine Basis für V . $U' = \{a_{l+1}, \dots, a_n\}$

$$V = U \oplus U' \quad (337)$$

$$U \perp U' \quad \forall u \in U, u' \in U', \quad u \perp u' \quad (338)$$

U' wird auch als U^\perp bezeichnet.

U^\perp ist das orthogonale Komplement von U in V .

Definition 64.

$$U^\perp = \{y \in V | y \perp U\} \quad (339)$$

$$(U^\perp)^\perp = U \quad (340)$$

$$V = U \oplus U^\perp \Rightarrow \dim V = \dim U + \dim U^\perp \quad (341)$$

(Das gilt für alle direkten Summen vom Unterraum.)

Example 22. $V = \mathbb{E}$, sei $A \in \mathbb{E}^{m \times n}$ und $x \in \mathbb{E}^n$

$$Ax = 0 \Leftrightarrow x \perp \text{allen Zeilen von } A \Leftrightarrow x \perp \text{allen Spalten von } A^H \quad (342)$$

$$\begin{pmatrix} \underline{a_1} \\ \vdots \\ \underline{a_m} \end{pmatrix} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \Leftrightarrow \langle a_1, x \rangle = 0 \Leftrightarrow a_1 \perp x \quad \text{Gilt f\"ur alle Zeilen m} \quad (343)$$

$$\mathbb{E} = \mathbb{R}: \quad \mathcal{N}(A) \Leftrightarrow Ax = 0 \Leftrightarrow x \perp \mathcal{R}(A^T) \Leftrightarrow x \in (\mathcal{R}(A^T))^\perp \quad (344)$$

$$\mathbb{E} = \mathbb{C}: \quad \mathcal{N}(A) \Leftrightarrow Ax = 0 \Leftrightarrow x \perp \mathcal{R}(A^H) \Leftrightarrow x \in (\mathcal{R}(A^H))^\perp \quad (345)$$

Theorem 6.9

F\"ur eine komplexe $m \times n$ -Matrix mit Rang r gilt

$$\mathcal{N}(A) = (\mathcal{R}(A^H))^\perp \subset \mathbb{E}^n \quad \mathcal{N}(A^H) = (\mathcal{R}(A))^\perp \subset \mathbb{E}^m \quad (346)$$

Insbesondere ist

$$\mathcal{N}(A) \oplus (\mathcal{R}(A^H))^\perp = \mathbb{E}^n \quad \mathcal{N}(A^H) \oplus (\mathcal{R}(A))^\perp = \mathbb{E}^m \quad (347)$$

und diese abge\"anderte Dimensionsformel gilt:

$$\dim \mathcal{R}(A) = r, \quad \dim \mathcal{N}(A) = n - r \quad (348)$$

$$\dim \mathcal{R}(A^H) = r, \quad \dim \mathcal{N}(A^H) = m - r \quad (349)$$

Bei reellen Matrizen kann \"uberall A^H durch A^T ersetzt werden.

6.5 Basiswechsel und Koordinatentransformation von Orthonormalbasen

Der Basiswechsel zwischen Orthonormalbasen ist leicht zu berechnen. Anstatt Invertierung haben wir nur Transponierung.

Sei V ein Vektorraum mit Skalarprodukt

$\{b_1, b_2, \dots, b_n\}$ Orthonormalbasis "die alte Basis" \mathcal{B}

$\{b'_1, b'_2, \dots, b'_n\}$ Orthonormalbasis "die neue Basis" \mathcal{B}'

Was ist nun die Basistransformationsmatrix T ?

Schreiben wir b'_k in Koordinaten der "alten" Basis \mathcal{B} :

$$b'_k = \sum_{j=1}^n \tau_{jk} b_j \text{ Basistransformationsmatrix } T = (\tau_{jk}) \quad (350)$$

Basistransformation von Koordinaten:

$$T\xi' = \xi \quad (351)$$

$$T^{-1}\xi = \xi' \quad (352)$$

$$T^{-1} = T^H \text{ in diesem Fall!} \quad (353)$$

$$T^{-1} = T^H \quad (354)$$

$$\delta_{kl} = \langle b'_k, b'_l \rangle = \left\langle \sum_{j=1}^n \tau_{jk} b_j, \sum_{m=1}^n \tau_{ml} b_m \right\rangle = \sum_{m=1}^n \tau_{ml} \left\langle \sum_{j=1}^n \tau_{jk} b_j, b_m \right\rangle = \sum_{m=1}^n \sum_{j=1}^n \bar{\tau}_{jk} \tau_{ml} \underbrace{\langle b_j, b_m \rangle}_{\delta_{jm}} \quad (355)$$

$$= \sum_{j=1}^n \bar{\tau}_{jk} \tau_{jl} \text{ ist ein Element von } T^H T \quad (356)$$

$$T^H T = \begin{pmatrix} \bar{\tau}_{11} & \cdots & \bar{\tau}_{1n} \\ \vdots & \ddots & \vdots \\ \bar{\tau}_{n1} & \cdots & \bar{\tau}_{nn} \end{pmatrix} \begin{pmatrix} \tau_{11} & \cdots & \tau_{1n} \\ \vdots & \ddots & \vdots \\ \tau_{n1} & \cdots & \tau_{nn} \end{pmatrix} \quad (357)$$

Element kl von $T^H T$

The following was presented in the lecture on 23. November 2018.

$$(T^H T)_{k,l} = \delta_{kl} \Rightarrow T^H T = I_{n \times n} \quad (358)$$

Theorem 6.10

Die Transformationsmatrix einer Basistransformation zwischen orthonormalen Basen ist unitär ($\mathbb{E} = \mathbb{C}$) beziehungsweise orthogonal ($\mathbb{E} = \mathbb{R}$).

$$T^{-1} = T^H \quad (359)$$

$$T^H T = T T^H = I \quad (360)$$

Theorem 6.11

$$\xi = T \xi' \quad \xi' = T^T \xi \quad (361)$$

Corollary 6.12

Seien $\mathcal{B}, \mathcal{B}'$ orthonormale Basen für V und $x, y \in V$. Das Skalarprodukt der Koordinaten bleibt erhalten.

$$\xi = [x]_{\mathcal{B}}, \quad \xi' = [x]_{\mathcal{B}'}, \quad \eta = [y]_{\mathcal{B}}, \quad \eta' = [y]_{\mathcal{B}'}, \quad (362)$$

$$\text{Dann gilt: } \xi^H \eta = \langle \xi, \eta \rangle = \langle \xi', \eta' \rangle = \xi'^H \eta' \quad (363)$$

$$\|\xi\| = \|\xi'\|, \quad \|\eta\| = \|\eta'\|, \quad \langle \xi, \eta \rangle = \langle \xi', \eta' \rangle, \quad \xi \perp \eta \Leftrightarrow \xi' \perp \eta' \quad (364)$$

Proof 6.12

Entweder algebraisch:

$$\xi' = T^H \xi \quad \eta' = T^H \eta \quad T^H T = I \quad (365)$$

Oder direkt durch die Parsevalsche Formel:

$$\langle x, y \rangle_v = \langle \xi, \eta \rangle_{\mathbb{E}^n} \quad \langle x, y \rangle_v = \langle \xi', \eta' \rangle_{\mathbb{E}^n} \quad (366)$$

□

6.5.1 Darstellung von linearen Abbildungen in Orthonormalbasen

$F : X \rightarrow Y$ lineare Abbildung X, Y sind Vektorräume mit Skalarprodukt (367)

(beide über \mathbb{R} oder beider über \mathbb{C})

X hat Orthonormalbasen $\mathcal{B}, \mathcal{B}'$

Y hat Orthonormalbasen $\mathcal{D}, \mathcal{D}'$

$$\underbrace{[F]_{\mathcal{B}, \mathcal{D}}}_{A} \xleftrightarrow{?} \underbrace{[F]_{\mathcal{B}', \mathcal{D}'}}_{B} \quad (368)$$

6.5.2 Selbstabbildung

$$X = Y, \mathcal{B} = \mathcal{D}, \mathcal{B}' = \mathcal{D}' \quad (369)$$

Dann: $T = S$ und

$$B = T^H A T \quad (370)$$

$$A = T B T^H \quad (371)$$

Corollary 6.13

Wenn F eine Selbstabbildung ist, dann:

1. A Hermitisch \Leftrightarrow B Hermitisch
2. $A^H A = I \Leftrightarrow B^H B = I$

6.6 Orthogonale und unitäre Abbildungen

Definition 65. X, Y Vektorraum mit Skalarprodukt. Eine lineare Abbildung $F : X \rightarrow Y$ heisst unitär (\mathbb{C})/ orthogonal (\mathbb{R}) falls

$$\forall x, w \in X : \langle F(x), F(w) \rangle_Y = \langle x, w \rangle_x \quad (372)$$

Theorem 6.13

Sei $F : X \rightarrow Y$ eine unitäre / orthogonale Abbildung.

1. F ist längentreu / isometrisch: $\forall x \in X : \|F(x)\|_Y = \|x\|_X \quad \|v\|_x = \sqrt{\langle v, v \rangle_X}$
2. $v \perp w \Leftrightarrow F(v) \perp F(w)$
3. $\ker F = \{0\}$ das heisst F ist injektiv

Zusätzlich, wenn $\dim X = \dim Y < \infty$:

4. F ist ein Isomorphismus
5. $\{b_1, b_2, \dots, b_n\}$ orthonormale Basis von $X \Rightarrow \{F(b_1), \dots, F(b_n)\}$ ist eine Orthonormalbasis von Y .
6. F^{-1} ist unitär / orthogonal
7. Die Abbildungsmatrix A bezüglich einer Orthonormalbasis von X und von Y ist unitär / orthogonal: $A^H A = A A^H = I$

Proof 6.13: (3)

Zu zeigen: $\ker F = \{0\}$

$$F(v) = 0 \Leftrightarrow \langle F(v), F(v) \rangle_y = 0 \stackrel{F \text{ unitär}}{\Leftrightarrow} \langle v, v \rangle_x = 0 \stackrel{(S3)}{\Leftrightarrow} v = 0 \quad (373)$$

$$F(v) = 0 \Leftrightarrow v = 0 \quad (374)$$

$$\Rightarrow \ker F = \{0\} \checkmark \quad (375)$$

$\Rightarrow F$ ist injektiv: $F(v) = F(w) \Rightarrow v = w$

□

Proof 6.13: (4)

$\dim X = \dim Y < \infty \quad F$ injektiv \checkmark

$$\dim Y = \dim X \stackrel{\substack{= \\ \text{Dimensionsformel}}}{=} \underbrace{\dim \ker F}_{0} + \dim \text{Im } F \quad (376)$$

$$\Rightarrow \dim \text{Im } F = \dim Y \quad (377)$$

$$\Rightarrow Y = \text{Im } F \quad (378)$$

F ist ein Isomorphismus \checkmark

□

Note 44. Dimension ist nicht gleich Kardinalität.

Proof 6.13: (7)

$$A^H A = I$$

$$\begin{pmatrix} \underline{a}_1 & & \\ \vdots & & \\ \underline{a}_n & & \end{pmatrix} \begin{pmatrix} | & \cdots & | \\ \overline{a}_1 & \cdots & \overline{a}_n \\ | & \cdots & | \end{pmatrix} = I \quad (379)$$

$$a_k^H a_l = \delta_{kl} \Leftrightarrow \langle a_k, a_l \rangle = \delta_{kl} \quad (380)$$

$\{a_1, a_2, \dots, a_n\}$ sind eine orthonormale Menge.

Das Gleiche gilt für die Zeilenvektoren von A .

□

Wir wollen zeigen, dass A orthonormale Kolonnen hat.

Wählen wir: $\xi_1 = e_k, \quad \xi_2 = e_l \quad e_k = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 1 \\ \dots \\ 0 \\ 0 \end{pmatrix}$ an der k-ten Stelle

$$Ae_k = a_k \quad (381)$$

$$Ae_l = a_l \quad (382)$$

$$\langle e_k, e_l \rangle = \langle A_{l_k}, A_{e_k} \rangle \quad (383)$$

$$\Rightarrow \delta_{kl} = \langle a_k, a_l \rangle \quad (384)$$

\Rightarrow Die Kolonnen von A sind orthonormal.

\Rightarrow A ist eine unitäre / orthonormale Matrix.

The following was presented in the lecture on 28. November 2018.

Example 23. $F : \mathbb{R}^3 \rightarrow \mathbb{R}^4$

$$F\left(\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right) = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 0 \end{pmatrix} \quad (385)$$

F ist orthogonal. F ist aber nicht invertierbar weil $\dim X \neq \dim Y$

$$\langle F(v), F(w) \rangle = \langle v, w \rangle \quad (386)$$

Note 45. Es ist nicht möglich das die Ursprungsmatrix grösser ist als die Bildmatrix bei einer orthogonalen Abbildung.

6.7 Normen von linearen Abbildungen und Matrizen (Operatornorm)

Motivation: Wann ist die Lösung von einem LGS zuverlässig?

$$Ax = b, \quad Ax' = b', \quad A''x'' = b \quad (387)$$

wenn $\|b - b'\|$ klein ist und $\|A - A''\|$ klein ist.

Das Ziel ist nun, dass $\|x - x'\|$ und $\|x - x''\|$ möglichst klein ist.

Das ist der Fall, wenn A gut konditioniert ist.

Definition 66. Die Konditionszahl einer Matrix $A \in \mathbb{E}^{n \times n}$

$$\kappa(A) = \|A\| \cdot \|A^{-1}\| \text{ für reguläre A} \quad (388)$$

$$\kappa(A) := \infty \text{ für singuläre A} \quad (389)$$

Je grösser $\kappa(A)$, desto schlechter konditioniert die Matrix A und das LGS $Ax = b$.

Definition 67. Es seien X, Y normierte Vektorräume mit $\|\cdot\|_x$ und $\|\cdot\|_y$

Eine lineare Abbildung $F : X \rightarrow Y$ heisst beschränkt (bounded), wenn es eine Zahl gibt,

$$\gamma_F \geq 0 \text{ so dass } \forall x \in X, \|F(x)\|_y \leq \gamma_F \|x\|_x \quad (390)$$

Die Gesamtheit von allen solchen Abbildungen zwischen X und Y wird mit $\mathcal{L}(X, Y)$ bezeichnet.

Note 46. Die Normen $\|\cdot\|_x$ $\|\cdot\|_y$ können von einem Skalarprodukt induziert sein, müssen es aber nicht. $\mathcal{L}(X, Y)$ ist ein Vektorraum. Die Operationen + und Multiplikation mit Skalar:

$$\mathcal{L}(X, Y) \ni F + G \quad x \in X \mapsto (F + G)(x) := F(x) + G(x) \quad (391)$$

$F + G$ ist beschränkt.

$$\mathcal{L}(X, Y) \ni \alpha F \quad x \in X \mapsto (\alpha F)(x) := \alpha F(x) \quad (392)$$

αF ist beschränkt.

Example 24. $\mathcal{L}(\mathbb{E}^n, \mathbb{E}^m)$ sind die Matrizen in $\mathbb{E}^{m \times n}$

Definition 68. Supremum, kleinste obere Schranke

Sei $\emptyset \neq M \subseteq \mathbb{R}$ eine Teilmenge von \mathbb{R} . Das Supremum von M , $\sup M$ ist eine Zahl $c \in \mathbb{R}$, so dass $\forall x \in M$ gilt:

1. $x \leq c$
2. $\forall c' < c, \exists y \in M$ so dass $y > c'$

Example 25. $M_1 = \{x \in \mathbb{R} \mid 0 < x < 17\}$

$\sup M_1 = 17$ und $\sup M_1 \notin M_1$

$M_2 = \{x \in \mathbb{R} \mid 0 < x < 17\}$

$\sup M_2 = 17$ und $17 \in M_2$ und $\sup M_2 = \max M_2$

$M_3 = \{x \in \mathbb{R} \mid 42 < x\}$

Supremum von M_3 ist nicht definiert.

Note 47. Die grösste untere Schranke, Infimum, $\inf M$ ist analog definiert.

Definition 69. Die induzierte Operatornorm $\mathcal{L}(X, Y)$ ist definiert als:

$$\|\cdot\| : \mathcal{L}(X, Y) \rightarrow \mathbb{R} \quad (393)$$

$$F \mapsto \|F\| := \sup \left\{ \frac{\|F(x)\|_Y}{\|x\|_X} \mid x \in X, x \neq 0 \right\} \quad (394)$$

$$\text{andere Notation } \|F\| = \sup_{x \neq 0} \frac{\|F(x)\|_Y}{\|x\|_X} \quad (395)$$

Ist $X = \mathbb{E}^n, Y = \mathbb{E}^m$, definieren wir eine Matrixnorm induziert durch die Vektornormen in $\mathbb{E}^n, \mathbb{E}^m$:

$$\|\cdot\| : \mathbb{E}^{m \times n} \rightarrow \mathbb{R}, \quad A \mapsto \|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} \quad (396)$$

Example 26. Sei die Norm in $\mathbb{E}^n(\mathbb{E}^m)$ die 2-Norm, $\|x\|_2 = \sqrt{x^H x}$, so heisst die reduzierte Operatornorm die Spektralnorm oder 2-Norm:

$$\|A\|_2 = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} \quad (397)$$

Lemma 6.17

Die Operatornormen erfüllen:

$$F \in \mathcal{L}(X, Y) : \|F\| = \sup_{\|x\|_X=1} \|F(x)\|_Y \quad (398)$$

$$A \in \mathbb{E}^{m \times n} : \|A\| = \sup_{\|x\|=1} \|Ax\| \quad (399)$$

Proof 6.17

Für $x \neq 0$, setzen wir $\alpha := \|x\|_x$ und $\tilde{x} := \frac{x}{\alpha}$ ein:

$$\|F(x)\|_Y = \|F(\alpha \cdot \frac{x}{\alpha})\|_Y \underset{F \text{ ist homogen}}{=} \|\alpha F(\frac{x}{\alpha})\|_Y = \|\alpha F(\tilde{x})\|_Y \quad (400)$$

Da die Norm immer homogen ist folgt nun:

$$\alpha \|F(\tilde{x})\|_Y = \|x\|_X \cdot \|F(\tilde{x})\|_Y \text{ und } \|\tilde{x}\| = 1 \quad (401)$$

$$\Rightarrow \|F\| = \sup_{x \neq 0} \frac{\|F(x)\|_Y}{\|x\|_X} = \sup_{x \neq 0} \|F(\tilde{x})\|_Y = \sup_{\|\tilde{x}\|_X=1, \tilde{x}=\frac{x}{\|x\|_X}} \|F(\tilde{x})\|_Y \quad (402)$$

□

Example 27.

$$D = \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ \dots & & \\ 0 & 0 & d_n \end{pmatrix} \quad \|D\|_2 = \max_{k=1, \dots, n} |d_k| \quad (403)$$

Proof 6.18

$$\|D\|_2^2 = \left(\sup_{\|x\|_2=1} \|Dx\|_2 \right)^2 = \sup_{\|x\|_2=1} \|Dx\|_2^2 = \sup_{\|x\|=1} \sum_{k=1}^n (d_k x_k)^2 = \max_{\|x\|=1} \sum_{k=1}^n |d_k|^2 |x_k|^2 \quad (404)$$

weil $\|x\|_2 = 1 : \forall k, \|x_k\| \leq 1$

Sei $B := \max_{k=1, \dots, n} |d_k|$;

$$\forall k : |d_k|^2 \leq B^2 \quad (405)$$

$$|x_k|^2 |d_k|^2 \leq |x_k|^2 B^2 \quad (406)$$

$$\sum_{k=1}^n |x_k|^2 |d_k|^2 \leq B^2 \sum_{k=1}^n |x_k|^2 = B^2 \underbrace{\|x\|_2^2}_1 = B^2 \quad (407)$$

(408)

$$\Rightarrow \|D\|_2^2 \leq B^2 = (\max |d_k|)^2 \Rightarrow \|D\|_2 \leq B \quad (409)$$

Zeigen wir, dass es wirklich " $=$ " ist. Die Schranke wird erreicht.

k^* ist der Index, sodass $|d_k^*| = \max_k |d_k|$ $c_k = (0, \dots, 0, 1, 0, \dots, 0)^T$

$$\|D_{e_k^*}\|_2 = \|(0, \dots, 0, d_k^*, 0, \dots, 0)^T\|_2 = |d_k^*| = \max_{k=1, \dots, n} |d_k| = B \quad (410)$$

$$\Rightarrow \|D\|_2 = B = \max_{k=1, \dots, n} |d_k| \quad (411)$$

□

The following was presented in the lecture on 30. November 2018.

Note 48. Eine Matrix-Norm kann eine induzierte Norm sein, muss es aber nicht.

Example 28. Die Frobenius-Norm $\|A\|_F$, $A \in \mathbb{E}^{n \times m}$

$$\|A\|_F := \sqrt{\sum_{k=1}^n \sum_{l=1}^m (a_{kl})^2} \quad (412)$$

$\|A\|_F$ ist keine induzierte Matrixnorm.

6.8 Konditionszahl einer regulären Matrix $A \in \mathbb{E}^{n \times n}$

$$\kappa(A) = \|A\| \cdot \|A^{-1}\| \quad (413)$$

Note 49. $\|\cdot\|$ ist die gewählte Matrixnorm am häufigsten wird $\|\cdot\|_2$ verwendet. Intuition zu dieser Definition von $\kappa(A)$

$$Ax = b \Rightarrow x = A^{-1}b \quad \text{Fehler in } b : b' = b + \epsilon \quad (414)$$

Was ist nun der maximale relative Fehler in der Lösung x ?

$$Ax' = b' \Rightarrow x' = A^{-1}b' = A^{-1}(B + \epsilon) = A^{-1}b + A^{-1}\epsilon = x + A^{-1}\epsilon \quad (415)$$

6.8.1 Relativer Fehler:

$$\frac{\frac{\|A^{-1}\epsilon\|}{\|x\|}}{\frac{\|\epsilon\|}{\|b\|}} \text{ ist das Gleiche wie } \frac{\text{relativer Fehler in } x}{\text{relativer Fehler in } b} \quad (416)$$

$$\frac{\frac{\|A^{-1}\epsilon\|}{\|x\|}}{\frac{\|\epsilon\|}{\|b\|}} = \frac{\|A^{-1}\epsilon\|}{\|x\|} \cdot \frac{\|b\|}{\|\epsilon\|} = \frac{\|A^{-1}\epsilon\|}{\|\epsilon\|} \cdot \frac{\|b\|}{\|x\|} = \frac{\|A^{-1}\epsilon\|}{\|\epsilon\|} \cdot \frac{\|b\|}{\|A^{-1}b\|} \quad (417)$$

$$\sup_{\epsilon, b \neq 0} \underbrace{\frac{\|A^{-1}\epsilon\|}{\|\epsilon\|}}_{\text{hängt nur von } \epsilon \text{ ab}} \cdot \underbrace{\frac{\|b\|}{\|A^{-1}b\|}}_{\text{hängt nur von } b \text{ ab}} = \sup_{\epsilon \neq 0} \underbrace{\frac{\|A^{-1}\epsilon\|}{\|\epsilon\|}}_{A^{-1}} \cdot \sup_{b \neq 0} \underbrace{\frac{\|b\|}{\|A^{-1}b\|}}_{\forall b \exists x, \text{ so dass } Ax=b, x=A^{-1}b} \quad (418)$$

$$\|A^{-1}\| \cdot \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \|A^{-1}\| \cdot \|A\| =: \kappa(A) \quad (419)$$

7 Die Methode der kleinsten Quadrate

Least-squares solution of a linear system.

Sei $Ax = b$ ein LGS mit mehr Gleichungen als Variablen.

$$m \begin{array}{c} | \\ A \\ | \\ n \end{array} \quad \begin{array}{c} | \\ X \\ | \end{array} = \begin{array}{c} | \\ b \\ | \end{array} \quad m \times 1$$

$A \in \mathbb{E}^{m \times n}$
LGS nicht lösbar.

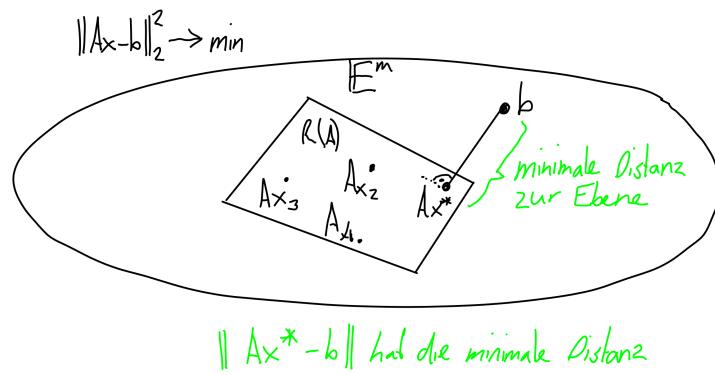
Gesucht ist nun ein Kompromiss. $Ax = b \Rightarrow Ax \approx b$

Daher:

$$x^* = \operatorname{argmin}_{x \in \mathbb{E}^n} \|Ax - b\|_2^2 \quad (420)$$

Eine Lösung im Sinne der kleinsten Quadrate.

$$Ax - b \quad Ax \in \mathcal{R}(A) = \underbrace{\operatorname{span} \{a_1, a_2, \dots, a_n\}}_{\text{Kolonen von } A} \quad a_k \in \mathbb{E}^m \quad m > 0 \quad (421)$$



Wir suchen nach x^* , so dass $(Ax^* - b) \perp \mathcal{R}(A)$

$$(Ax^* - b) \perp \mathcal{R}(A) = \operatorname{span} \{a_1, a_2, \dots, a_n\} \quad (422)$$

$$(Ax^* - b) \in \mathcal{R}(A)^\perp \underset{\substack{\text{Satz 6.9}}}{=} \mathbb{N}(A^H) \quad (423)$$

$$\Rightarrow A^H(Ax^* - b) = 0 \quad (424)$$

$$A^H Ax^* - A^H b = 0 \quad (425)$$

$$A^H Ax^* = A^H b \quad (426)$$

$$(Ax^* - b) \perp \text{Allen Spalten von } A \quad a_1^H(Ax^* - b) = 0 \quad (427)$$

$$(Ax^* - b) \perp a_2 \quad a_2^H(Ax^* - b) = 0 \quad (428)$$

$$\dots \dots \quad \dots \dots \quad (429)$$

$$(Ax^* - b) \perp a_n \quad a_n^H(Ax^* - b) = 0 \quad (430)$$

$$A^H \cdot (Ax^* - b) = 0 \quad (431)$$

Daraus folgt nun die Normalgleichung (the normal equation)

$$\boxed{\Rightarrow A^H Ax^* = A^H b} \quad (432)$$

$$A^H A \in \mathbb{E}^{n \times n} \quad A^H Ax^* = A^H b \quad (433)$$

Wann ist $A^H A$ regulär? Wenn Rang $R = n$

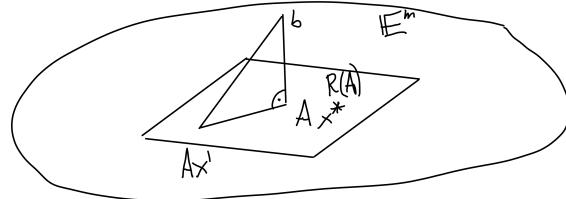
Note 50. Als Übung beweisen. Prüfungsfrage!

Note 51. Wenn Rang $A < n$, hat die Normalgleichung unendlich viele Lösungen. Man muss eine Selektieren, häufig: die Minimalnormlösung

$$\|Ax^* - b\|_2^2 \rightarrow \min \text{ und } \|x^*\|_2^2 \rightarrow \min$$

Das gibt uns die Singulärwertzerlegung (SVD)

Proof 7.1: Senkrechter Weg gibt kürzeste Distanz



$$\text{Zu zeigen } \forall x', \|Ax^* - b\|_2 \leq \|Ax' - b\|_2 \quad (434)$$

$$(Ax^* - b) \perp \underbrace{(Ax^* - Ax')}_{\in \mathcal{R}(A)} \quad (435)$$

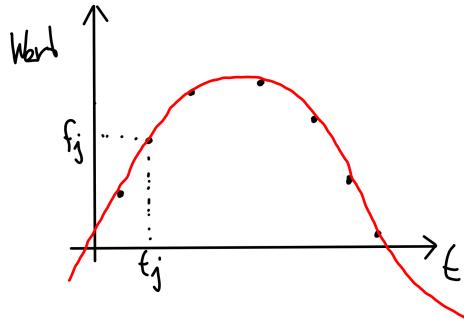
Mit Pythagoras (Cauchy Schwarz Ungleichung) kann man es nun wie folgt umformen.

$$\|Ax' - b\|^2 = \|Ax^* - b\|^2 + \|Ax' - Ax^*\|^2 \quad (436)$$

$$\Rightarrow \|Ax' - b\|^2 \geq \|Ax^* - b\|^2 \quad (437)$$

⇒ Der senkrechte Weg ist am kürzesten. \square

7.1 Approximation von Daten mit einer analytischen Funktion. (auch Regression genannt)



Folgendes konnte man anhand von Messungen herausfinden.

$$(t_1, f_1), (t_2, f_2) \dots (t_N, f_N) \quad (438)$$

$$f(t) = a_0 + a_1 t + \dots + a_n t^n \quad (439)$$

Finde $a_0, a_1, \dots, a_n \in \mathbb{R}$ so dass $\sum_{j=1}^N (f(t_j) - f_j)^2 \rightarrow \min$

Man hat nun folgende Wunschliste, jedoch ist $N \gg n$:

$$a_0 + a_1 t_1 + a_2 t_1^2 + \dots + a_n t_1^n = f_1 \quad (440)$$

$$a_0 + a_1 t_2 + a_2 t_2^2 + \dots + a_n t_2^n = f_2 \quad (441)$$

$$\dots \quad (442)$$

$$a_0 + a_1 t_N + a_2 t_N^2 + \dots + a_n t_N^n = f_N \quad (443)$$

$$(444)$$

Finde $a_0, a_1, \dots, a_n \in \mathbb{R}$ so dass $\sum_{j=1}^N (f(t_j) - f_j)^2 \rightarrow \min$ mit $N > n$

$$\underbrace{\begin{pmatrix} 1 & t_1 & t_1^2 & \dots & t_1^n \\ 1 & t_2 & t_2^2 & \dots & t_2^n \\ \dots & t_N & t_N^2 & \dots & t_N^n \end{pmatrix}}_{\text{Vandermonde-Matrix } U} \underbrace{\begin{pmatrix} a_0 \\ a_1 \\ \dots \\ a_n \end{pmatrix}}_a = \underbrace{\begin{pmatrix} f_1 \\ f_2 \\ \dots \\ f_N \end{pmatrix}}_f \quad (445)$$

$$\|Ua^* - f\|_2^2 \rightarrow \min \quad (446)$$

$$U^T U a^* = U^T f \leftarrow \text{Normalgleichungen} \quad (447)$$

$$a^* = (U^T U)^{-1} U^T f \quad (448)$$

The following was presented in the lecture on 05. December 2018.

$$A^H Ax = A^H b \quad \text{Normalgleichung} \quad (449)$$

$$Rang A = n \Rightarrow A^H A \text{ ist regulär} \quad (450)$$

$$X^* = (A^H A)^{-1} A^H b \quad (451)$$

Definition 70. Moore-Pensorse-Pseudoinverse von A

$$A^+ := (A^H A)^{-1} A^H \in \mathbb{E}^{n \times m} \quad (452)$$

Es gilt:

$$A^+ A = I \quad (453)$$

Note 52. Wenn Rang $A < n$, kann man trotzdem eine Pseudoinverse definieren: mittels Singulärwertzerlegung. (eng.: SVD.)

8 Determinante

Eine Zahl: $\det : \mathbb{E}^{m \times n} \rightarrow \mathbb{E}$, $A \mapsto \det A$, $|A|$

Haupteigenschaft: $\det A \neq 0 \Leftrightarrow A \text{ ist regulär}$

8.1 Permutationen

$$p : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, n\} \quad (454)$$

$$(1, 2, \dots, n) \mapsto (p(1), p(2), \dots, p(n)) \quad (455)$$

Die Funktion p ist eine Permutation falls p eineindeutig ist (inj. + surj.). Es gibt $n!$ verschiedene Permutationen von $(1, 2, \dots, n)$. Sie bilden ein Gruppe bezüglich \cdot (Zusammensetzung). Die Gruppe wird mit S_n bezeichnet. (die symmetrische Gruppe)

Theorem 8.2

Für $n > 1$ kann jede Permutation p als Produkt / Zusammensetzung von Transpositionen t_k benachbarter Elemente dargestellt werden:

$$p = t_v \circ t_{v-1} \circ \dots \circ t_2 \circ t_1 \quad (456)$$

Die Darstellung ist im Allgemeinen nicht eindeutig, aber die Anzahl der Transpositionen ist entweder immer gerade oder immer ungerade.

Example 29.

$$p : \{1, 2, 3\} \rightarrow \{1, 2, 3\} \quad (457)$$

$$(1, 2, 3) \xrightarrow{P} (2, 3, 1) \quad (458)$$

$$(1, 2, 3) \xrightarrow{t_1} (2, 1, 3) \xrightarrow{t_2} (2, 3, 1) \Rightarrow p = t_2 \circ t_1, v = 2 \quad (459)$$

Definition 71. Das Signum (sign) einer Permutation p ist:

$$\text{sign}(p) := \begin{cases} +1, & v \text{ ist gerade} \\ -1, & v \text{ ist ungerade} \end{cases} \quad (460)$$

8.2 Definition der Determinante

Definition 72. Die Determinante einer $n \times n$ Matrix $A = (a_{k,l})$, bezeichnet als $\det A$, $\det(A)$, $|A|$, ist:

$$\det A := \sum_{p \in S_n} \text{sign}(p) \cdot a_{1,p(1)} \cdot a_{2,p(2)} \cdot \dots \cdot a_{n,p(n)} \quad (461)$$

$$(1, 2, \dots, n) \xrightarrow{\text{P}} (p(1), p(2), \dots, p(n)) \quad (462)$$

Example 30. (n=1) $\det(a_{11}) = a_{11} \quad A \in \mathbb{R}^{1,1}$

$$(\mathbf{n=2}) \quad \det \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$$

(n=3)

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = \begin{matrix} +a_{11}a_{22}a_{33} & +a_{12}a_{23}a_{31} & +a_{13}a_{21}a_{32} \\ -a_{11}a_{23}a_{32} & -a_{12}a_{21}a_{33} & -a_{13}a_{22}a_{31} \end{matrix} \quad (463)$$

$$(1, 2, 3), v = 0 \oplus (2, 3, 1), v = 2 \oplus, (3, 1, 2), v = 2 \oplus \quad (464)$$

$$(1, 3, 2), v = 1 \ominus (2, 1, 3), v = 1 \ominus, (3, 2, 1), v = 3 \ominus \quad (465)$$

$$= a_{11}(a_{22}a_{33} - a_{23}a_{32}) - a_{12}(a_{21}a_{33} - a_{23}a_{31}) + a_{13}(a_{21}a_{32} - a_{22}a_{31}) \quad (466)$$

$$= a_{11} \begin{pmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{pmatrix} - a_{12} \begin{pmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{pmatrix} + a_{13} \begin{pmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix} \quad (467)$$

Theorem 8.12

$A \in \mathbb{E}^{n \times n}$ so gelten die Formeln für jedes feste $\kappa \in \{1, \dots, n\}$ und $l \in \{1, \dots, n\}$: Entwicklung nach der k -ten Zeile:

$$\det A = \sum_{j=1}^n (-1)^{k+j} a_{k,j} \cdot \det A_{[k,j]} \quad (468)$$

Entwicklung nach der l -ten Spalte

$$\det A = \sum_{j=1}^n (-1)^{j+l} a_{j,l} \cdot \det A_{[j,l]} \quad (469)$$

wobei: $A_{[k,j]}(A_{[j,l]})$ ist die $(n-1) \times (n-1)$ - Matrix definiert durch Streichen der Zeile k (bzw. j) und Spalte j (bzw. l) von A

Definition 73. Die Zahl $K_{j,l} := (-1)^{j+l} \cdot \det A_{[j,l]}$ heisst der Kofaktor $K_{j,l}$ von Element $a_{j,l}$

Example 31.

$$\begin{pmatrix} 5 & 3 & 2 & 0 \\ 1 & 4 & 6 & 8 \\ 0 & 4 & 3 & 7 \\ 1 & 5 & 8 & 6 \end{pmatrix} = 5 \begin{pmatrix} 4 & 6 & 8 \\ 4 & 3 & 7 \\ 5 & 8 & 6 \end{pmatrix} - 1 \begin{pmatrix} 3 & 2 & 0 \\ 4 & 3 & 7 \\ 5 & 8 & 6 \end{pmatrix} - 1 \begin{pmatrix} 3 & 2 & 0 \\ 4 & 6 & 8 \\ 4 & 3 & 7 \end{pmatrix} = \dots = 280 \quad (470)$$

The following was presented in the lecture on 07. December 2018.

8.3 Spezielle Eigenschaften für Dreiecksdreiecksmatrizen und Linksdreiecksmatrizen

Example 32.

$$A = \begin{pmatrix} r_{11} & \dots & \dots & r_{1n} \\ 0 & r_{kk} & \dots & r_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & r_{nn} \end{pmatrix} \quad (471)$$

$$\begin{pmatrix} r_{11} & \dots & \dots & r_{1n} \\ 0 & r_{kk} & \dots & r_{1n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & r_{nn} \end{pmatrix} = r_{11} \begin{pmatrix} r_{22} & \dots & \dots & r_{2n} \\ 0 & r_{kk} & \dots & r_{3n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & r_{nn} \end{pmatrix} = r_{11} \cdot r_{22} \begin{pmatrix} r_{33} & \dots & \dots & r_{3n} \\ 0 & r_{kk} & \dots & r_{4n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & r_{nn} \end{pmatrix} = \boxed{\prod_{j=1}^n r_{jj}} \quad (472)$$

Für die Linksdreiecksmatrix

$$\boxed{\prod_{j=1}^n l_{jj}} \quad (473)$$

8.4 Verifizierung durch die Permutationsformel

$$\det A = \sum_{p \in S_n} \text{sign}(p) \cdot a_{1,p(1)} \cdot a_{2,p(2)} \cdot \dots \cdot a_{n,p(n)} \quad (474)$$

überlebt nur die Permutation $p = id$, $(1, 2, \dots, n) \mapsto (1, 2, \dots, n)$ weil für jede andere Permutation $\exists j \in \{1, \dots, n\}$ so dass $p(j) < j$ und dann $a_{j,p(j)} = 0 \Rightarrow$ der entsprechende Summand = 0

$$\Rightarrow \det A = \underbrace{\text{sign}(id)}_1 \cdot a_{11} \cdot a_{22} \cdot \dots \cdot a_{nn} \quad (475)$$

8.5 Eigenschaften der Determinante

Theorem 8.3

Für \det gilt:

1. det ist linear in jeder Zeile von A, das heisst $\forall \alpha, \beta \in \mathbb{E}, \quad \forall l \in \{1, \dots, n\}$

$$\det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ \alpha u_l + \beta v_l & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} = \alpha \cdot \det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ u_l & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} + \beta \cdot \det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ v_l & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} \quad (476)$$

$$u_l, v_l, a_1, \dots, a_n \in \mathbb{E}^n$$

$$\overbrace{\det(\alpha A + \beta B)} = \alpha \cdot \det A + \beta \cdot \det B \quad (477)$$

Die obige Gleichung gilt nicht.

2. Werden zwei Zeilen in Matrix A vertauscht, so wechselt det A das Vorzeichen:

$$\det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ \underline{a}_k & \cdots \\ \vdots & \cdots \\ \underline{a}_l & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} = - \det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ \underline{a}_l & \cdots \\ \vdots & \cdots \\ \underline{a}_k & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} \quad (478)$$

3. $\det I = 1$

Proof 8.3: (3)

I ist eine Rechtsdreiecksmatrix \Rightarrow Produkt der Diagonale, $\det I = 1 \cdot \dots \cdot 1 = 1$

□

Proof 8.3: (1)

Folgt direkt aus der Permutationsformel:

$$\det A = \sum_{p \in S_n} \text{sign}(p) \cdot a_{1,p(1)} \cdot \dots \cdot (\alpha u_{l,p(l)} + \beta v_{l,p(l)}) \cdot \dots \cdot a_{n,p(n)} \quad (479)$$

$$= \left(\alpha \sum_{p \in S_n} \text{sign}(p) \cdot a_{1,p(1)} \cdot \dots \cdot u_{l,p(l)} \cdot \dots \cdot a_{n,p(n)} \right) \quad (480)$$

$$+ \left(\beta \sum_{p \in S_n} \text{sign}(p) \cdot a_{1,p(1)} \cdot \dots \cdot v_{l,p(l)} \cdot \dots \cdot a_{n,p(n)} \right) \quad (481)$$

Per Definition gilt nun:

$$\Rightarrow \det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ u_l & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} + \det \begin{pmatrix} \underline{a}_1 & \cdots \\ \vdots & \cdots \\ v_l & \cdots \\ \vdots & \cdots \\ \underline{a}_n & \cdots \end{pmatrix} \quad (482)$$

□

Proof 8.3: (2)

Hausaufgabe

Hinweis: Wenn Zeilen k, l vertauscht werden, können wir das mit einer Transposition kompensieren, dabei ändert sich $\text{sign}(p)$, aber das restliche Produkt $a_1, p(1) \cdot \dots \cdot a_n, p(n)$ ändert sich nicht. \square

Theorem 8.4

Das Funktional \det hat folgende weitere Eigenschaften:

- iv Hat A eine Zeile aus lauter Nullen, so ist $\det A = 0$, wegen (i)
- v $\det(\gamma A) = \gamma^n \det A$, wegen(i)
- vi Hat A zwei Gleiche Zeilen, so ist $\det A = 0$
- vii Addiert man zu einer Zeile von A ein Vielfaches einer anderen Zeile von A , so ändert sich der Wert von $\det A$ nicht.
- viii Ist A eine Diagonalmatrix, so ist $\det A$ gleich dem Produkt der Diagonalmatrix.
- ix Ist A eine Dreiecksmatrix, so ist $\det A$ gleich dem Produkt der Diagonalmatrix.

Proof 8.4: (vi)

$$\det \begin{pmatrix} \underline{a}_1 & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_k & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_l & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_n & \underline{\quad\quad\quad} \end{pmatrix} \stackrel{(ii)}{=} -\det \begin{pmatrix} \underline{a}_1 & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_l & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_k & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_n & \underline{\quad\quad\quad} \end{pmatrix} \quad (483)$$

$$\Rightarrow \det A = -\det A \Rightarrow \det A = 0 \quad (484)$$

\square

Proof 8.4: (vii)

$$A = \begin{pmatrix} \underline{a}_1 & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_k & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_l & \underline{\quad\quad\quad} \\ \vdots & \vdots \\ \underline{a}_n & \underline{\quad\quad\quad} \end{pmatrix} \quad (485)$$

Addieren wir γa_k zu Zeile l

$$\det \begin{pmatrix} \underline{a_1} & & \\ \vdots & & \\ \underline{a_k} & & \\ \vdots & & \\ \underline{a_l + \gamma a_k} & & \\ \vdots & & \\ \underline{a_n} & & \end{pmatrix} \stackrel{(i)}{=} \underbrace{\begin{pmatrix} \underline{a_1} & & \\ \vdots & & \\ \underline{a_k} & & \\ \vdots & & \\ \underline{a_l} & & \\ \vdots & & \\ \underline{a_n} & & \end{pmatrix}}_A + \gamma \cdot \det \begin{pmatrix} \underline{a_1} & & \\ \vdots & & \\ \underline{a_k} & & \\ \vdots & & \\ \underline{a_k} & & \\ \vdots & & \\ \underline{a_n} & & \end{pmatrix} \quad (486)$$

2 identische Zeilen $\Rightarrow \det = 0$ (iv)

□

Proof 8.4: (viii)

$$\det \begin{pmatrix} a_{11} & \dots & 0 \\ \dots & \dots & \\ 0 & \dots & a_{nn} \end{pmatrix} \stackrel{(i) \text{ n Mal anwenden}}{=} a_{11} \cdot \dots \cdot a_{nn} |I| \stackrel{(iii)}{=} a_{11} \cdot \dots \cdot a_{nn} \quad (487)$$

□

Theorem 8.5

Wenden wir den Gauss Algorithmus auf $A \in \mathbb{E}^{n \times n}$ an, dann gilt:

$$\boxed{\det A = (-1)^\theta \prod_{k=1}^n r_{kk}} \quad (488)$$

wobei θ die Anzahl Zeilenumtauschungen ist und r_{kk} sind die Diagonalelemente der resultierenden Zeilenstufenform.

Zudem gilt:

$$\det A \neq 0 \Leftrightarrow \text{Rang } A = n \Leftrightarrow A \text{ ist regulär} \quad (489)$$

Proof 8.5

$$A \xrightarrow[v \text{ Zeilenumtauschungen} + \text{Addieren } \gamma a_k]{} \begin{pmatrix} r_{11} & \dots & \dots & r_{1n} \\ 0 & r_{kk} & \dots & r_{2n} \\ \dots & \dots & \dots & \\ 0 & 0 & 0 & r_{nn} \end{pmatrix} = r_{11} \cdot \dots \cdot r_{nn} \quad (490)$$

$$A \text{ regulär} \Leftrightarrow r_{kk} \neq 0 \quad \forall k = 1, \dots, n \Leftrightarrow \text{Rang } A = n \Leftrightarrow \det A \neq 0 \quad (491)$$

□

Note 53. Gauss Algorithmus kosten $\mathcal{O}(n^3)$ \Rightarrow eine billigere Art, die Determinante zu berechnen.

Theorem 8.7

$A, B \in \mathbb{E}^{n \times n}$

$$\boxed{\det(AB) = \det(A) \cdot \det(B)} \quad (492)$$

Theorem 8.6

Die Determinante ist die einzige Funktion $\mathbb{E}^{n \times n} \rightarrow \mathbb{E}$, die die Eigenschaften (i),(ii),(iii) besitzt.

Proof 8.6

$$\forall k = 1, \dots, n, \quad a_k \in \mathbb{E}^n, \quad a_k = a_{k1}e_1 + a_{k2}e_2 + \dots + a_{kn}e_n \quad (493)$$

e_l hat an der l-ten Stelle eine 1 sonst überall 0. Sei $f : \mathbb{E}^{n \times n} \rightarrow \mathbb{E}$ eine Funktion, die die Eigenschaften (i),(ii),(iii) hat.

$$f(A) = f\left(\begin{array}{c} \underline{a_1} \\ \vdots \\ \underline{a_n} \end{array}\right) = f\left(\begin{array}{c} a_{11}e_1 + a_{12}e_2 + \dots + a_{1n}e_n \\ a_{21}e_1 + a_{22}e_2 + \dots + a_{2n}e_n \\ \vdots \\ a_{n1}e_1 + a_{n2}e_2 + \dots + a_{nn}e_n \end{array}\right) \quad (494)$$

$$= \sum_{k_1=1}^n \sum_{k_2=1}^n \dots \sum_{k_n=1}^n a_{1,k_1} \cdot a_{2,k_2} \cdot \dots \cdot a_{n,k_n} = f\left(\begin{array}{c} \underline{e_{k_1}} \\ \vdots \\ \underline{e_{k_n}} \end{array}\right) \quad (495)$$

$$f\left(\begin{array}{c} \underline{e_{k_1}} \\ \vdots \\ \underline{e_{k_n}} \end{array}\right) \neq 0 \quad (496)$$

Die obige Gleichung stimmt genau dann wenn keine 2 Zeilen gleich sind. \Leftrightarrow nur wenn (k_1, \dots, k_n) eine Permutation von $(1, \dots, n)$ ist.

Aus der obigen langen Gleichung folgt nun die Permutationsformel:

$$\Rightarrow \sum_{(k_1, \dots, k_n) \in S_n} a_{1,k_1} \dots a_{n,k_n} \cdot \text{sign}((k_1, \dots, k_n)) \quad (497)$$

$\left(\begin{array}{c} \underline{e_{k_1}} \\ \vdots \\ \underline{e_{k_n}} \end{array}\right)$ ist eine Zeilenumtauschung von I. □

Proof 8.7

1. $\det B = 0 \Rightarrow B$ ist singulär. $\Rightarrow AB$ ist singulär $\Rightarrow \det(AB) = 0$ und dann

$$\underbrace{\det(AB)}_0 = \underbrace{\det(A)}_0 \cdot \underbrace{\det(B)}_0 \checkmark \quad (498)$$

2. $\det B \neq 0$ Definieren wir:

$$F(A) := \frac{\det(AB)}{\det(B)} \quad f : \mathbb{E}^{n \times n} \rightarrow \mathbb{E} \quad (499)$$

Zeigen wir, dass f die Eigenschaften (i),(ii),(iii) besitzt.
 $\Rightarrow f(A) = \det(A)$, laut (8.6)

$$\Rightarrow \det(A) = \frac{\det(AB)}{\det(B)} \Rightarrow \det(A)\det(B) = \det(AB) \quad (500)$$

$$(iii) f(I) = \frac{\det(I \cdot B)}{\det(B)} = \frac{\det(B)}{\det(B)} = 1 \checkmark \quad (501)$$

□

Note 54. (i) und (ii) als Übung. Hinweis zur Übung:

$$AB = \begin{pmatrix} B \cdot a_1 \\ \vdots \\ B \cdot a_n \end{pmatrix} \quad (502)$$

a_k ist die k-te Zeile von A

The following was presented in the lecture on 12. December 2018.

Eigenschaften der Determinante (Fortsetzung)

$$\det(AB) = \det(A) \cdot \det(B) \quad (503)$$

Corollary 8.8

$$A \text{ ist regulär} \Rightarrow \det(A^{-1}) = \frac{1}{\det(A)} \quad (504)$$

Proof 8.8

$$A^{-1} \cdot A = I \quad (505)$$

$$\det(A^{-1} \cdot A) = \det(I) = 1 \quad (506)$$

$$\det(A^{-1} \cdot A) = \det(A^{-1}) \cdot \det(A) \quad (507)$$

□

Theorem 8.9

Es gilt $\det(A^T) = \det(A)$ und $\det(A^H) = \overline{\det(A)}$

Proof 8.9

$$\det(A^T) = \sum_{p \in S_n} \text{sign}(p) \cdot a_{p(1),1} \cdot a_{p(2),2} \cdots a_{p(n),n} \quad (508)$$

Für jede $p \in S_n$ $\exists \tilde{p} \in S_n$ so dass $p \circ \tilde{p} = id$, $\tilde{p} = p^{-1}$

□

$$\det(A^T) = \sum_{\tilde{p} \in S_n} \text{sign}(\tilde{p}) \cdot a_{1,\tilde{p}(1)} \cdot a_{2,\tilde{p}(2)} \cdots a_{n,\tilde{p}(n)} = \det(A) \quad (509)$$

$$\det(A^H) = \sum_{p \in S_n} \text{sign}(p) \overline{a_{p(1),1}} \cdot \overline{a_{p(2),2}} \cdots \overline{a_{p(n),n}} = \overline{\det(A)} \quad (510)$$

Note 55. Auch wenn $A \in \mathbb{C}^{n \times n}$, $\det(A^T) = \det(A)$

$$A^H \neq A^T \text{ über } \mathbb{C} \quad (511)$$

Example 33. Sei $U \in \mathbb{C}^{n \times n}$ eine unitäre Matrix.

$$I = U^H U \Rightarrow \det(I) = \det(U^H) \cdot \det(U) \quad (512)$$

$$1 = \underbrace{\det(U)}_{\text{komplexe Zahl}} \cdot \underbrace{\det(U)}_{\text{komplexe Zahl}} \quad (513)$$

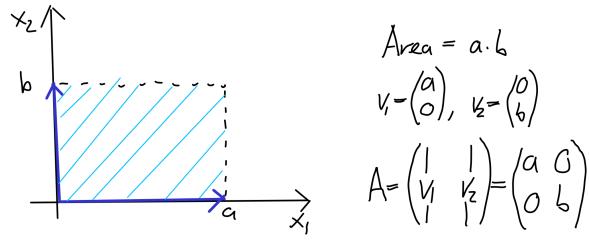
$$1 = |\det(U)|^2 \Rightarrow |\det(U)| = 1 \quad (514)$$

Corollary 8.10

Die Eigenschaften (i) und (ii) aus Satz 8.3 sowie die Eigenschaften (iv), (vi), (vii) aus Satz 8.4 gelten auch wenn man Zeile durch Kolone ersetzt.

1. \det ist eine lineare Funktion jeder einzelnen Zeile (Spalte) der Matrix daher für alle $\gamma, \gamma' \in \mathbb{E}$ und alle $l \in \{1, \dots, n\}$ ist.
2. Werden in A zwei Zeilen (Spalten) vertauscht, so wechselt $\det(A)$ das Vorzeichen.
3. Hat A eine Zeile (Spalte) aus lauter Nullen, so ist $\det(A) = 0$
4. Hat A zwei gleiche Zeilen (Spalten), so ist $\det(A) = 0$
5. Addiert man zu einer Zeile (Spalte) von A ein Vielfaches einer anderen Zeile (Spalte) von A , so ändert sich der Wert von $\det(A)$ nicht.

8.6 Geometrische Bedeutung von \det

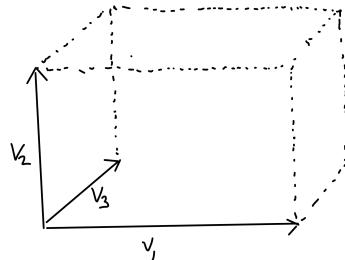


$$\det(A) = a \cdot b = \text{Oberfläche vom Rechteck} \quad (515)$$

$$\det \begin{pmatrix} | & | \\ \bar{v}_1 & \bar{v}_2 \end{pmatrix} = \det \begin{pmatrix} | & | \\ \bar{v}_1 & \bar{v}_2 + \alpha v_1 \end{pmatrix} \quad (516)$$

Deformation vom Rechteck $\begin{pmatrix} | & | \\ \bar{v}_1 & \bar{v}_2 \end{pmatrix}$ ist eine Scherung (shear). Das ist eine Abbildung, die das Volumen beibehält.

3D



$$\text{Volumen} = \|v_1\| \cdot \|v_2\| \cdot \|v_3\| = \left\| \det \begin{pmatrix} | & | & | \\ \bar{v}_1 & \bar{v}_2 & \bar{v}_3 \end{pmatrix} \right\| \quad (517)$$

In 3D und in einem n-dimensionalen Raum.

Das Vorzeichen der Determinante: Rechte-Hand-Regel (Wie in Physik)

Note 56. Wenn lineare Abhängigkeit herrscht, ist die Fläche 0, da die Vektoren kein Volldimensionaler Raum aufspannen.

9 Eigenwerte und Eigenvektoren

9.1 Zusammenfassung

9.1.1 LR-Zerlegung

Eine Variation falls A symmetrisch ist die LLT-Zerlegung. Da ist die R Matrix die L Matrix transponiert.

9.1.2 QR-Zerlegung

Eine unitäre / orthogonale Matrix Q wird mit einer Rechtsdreiecksmatrix R multipliziert.

9.1.3 Spektralzerlegung

Eine reguläre Matrix V wird mit einer Diagonalmatrix und der inversen von V multipliziert.
 $A = V \cdot \lambda \cdot V^{-1}$

Die Spektralzerlegung exisitiert nicht für jede Matrix A.

9.1.4 Singulärwertzerlegung

Gilt für jedes A. Eine unitäre / orthogonale Matrix V wird mit einer Diagonalmatrix Σ und einer unitären / orthogonalen Matrix U^T multipliziert. $A = V \cdot \Sigma \cdot U^T$

9.2 Spektralzerlegung

$F : V \rightarrow V$, und V ist ein Vektorraum mit $\dim V < \infty$

Wir interessieren uns für Vektoren $x \in V$, die F nur skaliert.

$$x \rightarrow [F] \rightarrow \lambda x \quad (518)$$

Definition 74. Die Zahl $\lambda \in \mathbb{E}$ heisst Eigenwert (eigenvalue) (E.W.) der linearen Abbildung $F : V \rightarrow V$, falls es einen Vektor $x \in V$ mit $x \neq 0$ gibt, sodass $F(x) = \lambda x$. Der Vektor x heisst dann Eigenvektor (eigenvector) (E.V.) von F, der zum Eigenwert λ gehört.

Definition 75. Die Menge aller Eigenvektoren, die zum Eigenwert λ gehören und der Nullvektor, bilden einen Unterraum.

$$E_\lambda := \{v \in V | F(v) = \lambda v\} \quad (519)$$

Beweis für Unterraum: Wir müssen zeigen: Abgeschlossen bezüglich + und Multiplikation mit Skalar.

$$\forall \alpha, \beta \in \mathbb{E}, \quad v, w \in E_\lambda : \quad (\alpha v + \beta w) \in E_\lambda \quad (520)$$

Zu zeigen:

$$F(\alpha v + \beta w) = \lambda(\alpha v + \beta w) \quad (521)$$

$$F(\alpha v + \beta w) = \alpha F(v) + \beta F(w) = \alpha \cdot \lambda v + \beta \cdot \lambda w = \lambda(\alpha v + \beta w) \checkmark \quad (522)$$

Definition 76. Die Menge aller Eigenwerte von F heisst Spektrum von F .

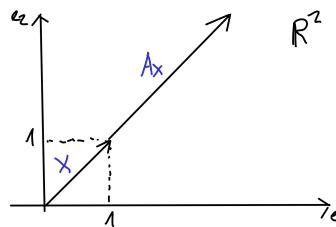
Die Definition gilt auch für $V = \mathbb{E}^n$ und Matrizen von Selbstabbildungen: $A \in \mathbb{E}^{n \times n}$

$\xi \in \mathbb{E}^n$ ist ein Eigenvektor zum Eigenwert $\lambda \in \mathbb{E}$ von Matrix $A \in \mathbb{E}^{n \times n}$ falls $\xi \neq 0$ und $A\xi = \lambda\xi$

Example 34. $V = \mathbb{R}^2$, $A : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, $A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

Der Vektor $x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ ist ein Eigenvektor von A mit Eigenwert $\lambda = 3$

$$Ax = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \end{pmatrix} = 3 \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 3x \checkmark \quad (523)$$



Auch alle Vektoren der Form $\alpha \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ sind Eigenvektoren von $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$ zu Eigenwerten $n = 3$

$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$ A ist singulär. A hat einen Eigenwert $\lambda = 0$

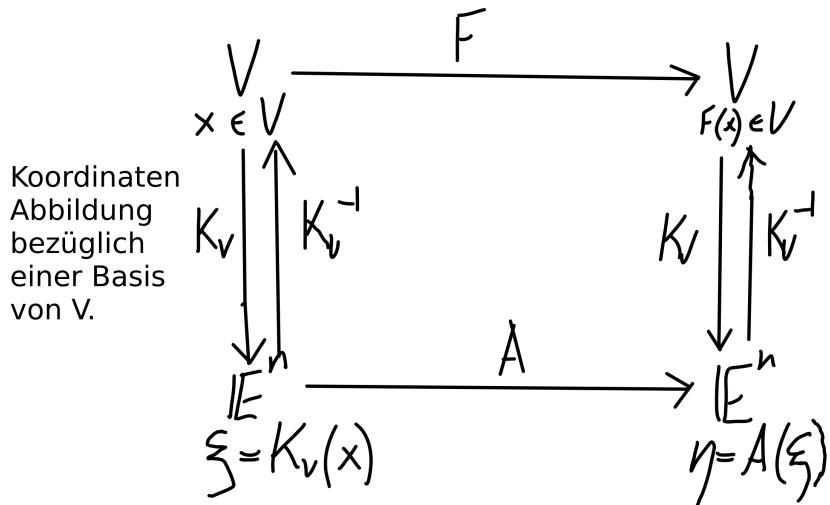
$$x = \begin{pmatrix} 1 \\ -1 \end{pmatrix}: \quad Ax = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0 \cdot \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad (524)$$

Eine Matrix in $\mathbb{E}^{n \times n}$ ist singulär \Leftrightarrow Die Matrix hat Eigenwert $\lambda = 0$

Theorem 9.1

Eine lineare Abbildung $F : V \rightarrow V$ und ihre Matrixdarstellung bezüglich einer Basis in V haben die gleichen Eigenwerte.

Die Eigenvektoren sind über die Koordinatenabbildung verbunden.



$$A = K_V \circ F \circ K_V^{-1} \quad (525)$$

Sei $x \in V$ ein Eigenvektor von F mit Eigenwert λ . Der Koordinatenvektor $\xi = K_V(x)$ ist ein Eigenvektor von Matrix A mit Eigenwert λ !

$$A_x i = K_V \circ F \circ (K_V^{-1}(x)) = K_V \circ F(x) = K_V \circ \lambda x = \lambda K_V(x) = \lambda \xi \quad (526)$$

x ist ein Eigenvektor von F , daher $x \neq 0$, K_V ist regulär, daher $K_V(x) \neq 0 \Rightarrow \xi \neq 0$
 $\Rightarrow \xi \neq 0$, $A\xi = \lambda\xi \Rightarrow \xi$ ist Eigenvektor von A zu λ . Es gilt auch umgekehrt. Sei $u \in E^n$ ein Eigenvektor von A mit Eigenwert $\alpha \in E$ daher $Au = \alpha u$, $\alpha \neq 0$.

Sei $x \in V$ sodass $K_V(x) = u$

Dann gilt:

$$Au = \alpha u \quad (527)$$

$$K_V \circ F \circ (K_V^{-1}u) = \alpha u \quad (528)$$

$$K_V \circ F(x) = \alpha K_V(x) \quad (529)$$

$$F(x) = \alpha x \quad (530)$$

$\Rightarrow x \neq 0$, x ist Eigenvektor von F zum Eigenwert α

Theorem 9.2

$$Ax = \lambda x, \quad x \neq 0 \quad (531)$$

$$(A - \lambda I)x = 0, \quad x \neq 0 \quad (532)$$

$\Rightarrow (A - \lambda I)$ ist eine singuläre Matrix.

$$\det(A - \lambda I) = 0 \quad (533)$$

The following was presented in the lecture on 14. December 2018.

9.3 Dimension von Eigenwerten

$A \in \mathbb{E}^{n \times n}$, λ ist der Eigenwert.

$$E_\lambda = \{x \in \mathbb{E}^n \mid Ax = \lambda x\} \quad \dim E_\lambda = ? \quad (534)$$

$\dim E_\lambda \geq 1$, weil es einen Eigenvektor $\neq 0$ gibt.

$$\exists x \neq 0, \quad Ax = \lambda x \Leftrightarrow (A - \lambda I)x = 0, \quad \exists x \neq 0 \quad (535)$$

$$\Leftrightarrow (A - \lambda I) \text{ ist singulär} \Leftrightarrow \dim(\ker(A - \lambda I)) \geq 1 \quad (536)$$

$$x \in E_\lambda \Leftrightarrow x \in \ker(A - \lambda I) \quad (537)$$

$$\Rightarrow \boxed{\ker(A - \lambda I)} \quad (538)$$

Definition 77. Die geometrische Vielfachheit (geometric multiplicity) eines Eigenwertes λ ist gleich:

$$\dim E_\lambda = \dim \ker(A - \lambda I) \quad (539)$$

Example 35. Eine Matrix $A \in \mathbb{E}^{n \times n}$, so dass

$$\det A = 0 \Leftrightarrow A \text{ ist singulär} \Leftrightarrow \exists x \neq 0, Ax = 0x \Leftrightarrow A \text{ hat einen Eigenwert } \lambda = 0 \quad (540)$$

$$E_0 = \ker(A - 0 \cdot I) = \ker A = \mathbb{N}(A) \quad (541)$$

\Rightarrow Geometrische Vielfachheit von $\lambda = 0$ ist $\dim \ker A = n - \text{Rang } A$

Example 36. Die Identitätsmatrix $I \in \mathbb{E}^{n \times n} : \forall x, Ix = 1 \cdot x$

$\Rightarrow I$ hat Eigenwert $\lambda = 1$ und:

$E_1 = \mathbb{E}^n$, weil jeder Vektor in \mathbb{E}^n ist Eigenvektor. \Rightarrow Die geometrische Vielfachheit von $\lambda = 1$ ist n .

9.4 Wie findet man die Eigenwerte für A

λ ist ein Eigenwert von $A \in \mathbb{E}^{n \times n} \Leftrightarrow (A - \lambda I)$ singulär $\Leftrightarrow \det(A - \lambda I) = 0$

\Rightarrow Suchen wir, für welche Werte von A , $\det(A - \lambda I) = 0$?

Example 37.

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \quad \det(A - \lambda I) = \det \begin{pmatrix} 2 - \lambda & 1 \\ 1 & 2 - \lambda \end{pmatrix} = (2 - \lambda)^2 - 1 = (\lambda - 3)(\lambda - 1) \quad (542)$$

$\Rightarrow \lambda_1 = 3$, und $\lambda_2 = 1$ sind Eigenwerte von A

Example 38.

$$A = \begin{pmatrix} 2 & -2 & 2 \\ 0 & 1 & 1 \\ 3 & 0 & 2 \end{pmatrix} \quad (543)$$

$$\det(A - \lambda I) = \det \begin{pmatrix} 2 - \lambda & -2 & 2 \\ 0 & 1 - \lambda & 1 \\ 3 & 0 & 2 - \lambda \end{pmatrix} \quad (544)$$

$\Rightarrow \lambda_1 = 2$, $\lambda_2 = -1$ und $\lambda_3 = 4$ sind Eigenwerte von A

9.5 Wie finden wir Eigenvektoren?

Lösen von LGS $Ax = \lambda x$ oder Bestimmung von Kern von $(A - \lambda I)$

Für Beispiel D, $E_{\lambda_1} = ?$ oder $E_3 = ?$ $\ker(A - 3I)$

$$A - 3I = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} \Rightarrow \text{lösen } \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (545)$$

$$E_3 = \ker \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} \quad \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \alpha \in \mathbb{R} \quad (546)$$

$$\Rightarrow E_3 = \ker \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} = \text{span} \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}, \quad \dim E_3 = 1 \quad (547)$$

$$(\lambda_2) \quad E_1 = \ker(A - 1 \cdot I) = \ker \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \left\{ \alpha \begin{pmatrix} 1 \\ -1 \end{pmatrix} \mid \alpha \in \mathbb{R} \right\} \Rightarrow \dim E_1 = 1 \quad (548)$$

Definition 78. Das Polynom $\det(A - \lambda I) =: \chi_A(\lambda)$ heisst das charakteristische Polynom der Matrix $A \in \mathbb{E}^{n \times n}$ und die Gleichung $\chi_A(\lambda) = 0$ heisst die charakteristische Gleichung.

Theorem 9.5

$\lambda \in \mathbb{E}$ ist ein Eigenwert von $A \in \mathbb{E}^{n \times n} \Leftrightarrow \lambda$ ist eine Nullstelle von $\chi_A \Leftrightarrow \lambda$ ist eine Lösung der charakteristischen Gleichung

Note 57. Fundamentaler Satz der Algebra: Ein Polynom von Grad n über \mathbb{C} hat genau n (nicht unbedingt unterschiedliche) Nullstellen.

$$p(x) = cx^n + a_{n-1}x^{n-1} + \dots + a_1x + a_0 = c(x - x_1)(x - x_2)\dots(x - x_n) \quad (549)$$

$\mathbb{C} \ni x_j$ sind Nullstellen. \Rightarrow Über \mathbb{C} gibt es immer Eigenwerte. Aber nicht unbedingt über \mathbb{R} $A \in \mathbb{E}^{n \times n}$

$$\chi_A(\lambda) = \det(A - \lambda I) = \det \begin{pmatrix} a_{11} - \lambda & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} - \lambda & a_{23} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} - \lambda \end{pmatrix} \quad (550)$$

$$= \underbrace{(a_{11} - \lambda)(a_{22} - \lambda)\dots(a_{nn} - \lambda)}_{\text{Permutation p=id}} + \underbrace{\dots}_{\text{alle anderen Permutationen die Exponente von } \lambda \leq n-2} \quad (551)$$

$$(-1)^\lambda \lambda^n + (-1)^{\lambda-1} \lambda^{n-1} (a_{11} + a_{22} + \dots + a_{33}) + \dots \text{mit Grad } n \leq n-2 \quad (552)$$

Definition 79. Die Summe der Diagonalelemente einer Matrix $A \in \mathbb{E}^{n \times n}$ heisst Spur (trace).

$$\text{Spur } A = \text{tr } A = a_{11} + a_{22} + a_{33} + \dots + a_{nn} \quad (553)$$

Lemma 9.4

$$\chi_A(\lambda) = (-1)^n \lambda^n + (-1)^{n-1} \text{Spur } A \lambda^{n-1} + \dots + \det A \quad (554)$$

Proof 9.4

$$\chi_A(\lambda) = (-1)^n \lambda^n + b_{n-1} \lambda^{n-1} + \dots + b_1 \lambda + b_0 \quad (555)$$

$$\lambda = 0$$

$$\chi_A(0) = b_0 = \det(A - 0 \cdot I) = \det A \quad (556)$$

□

Definition 80. Die algebraische Vielfachheit (algebraic multiplicity) eines Eigenwertes ist die Vielfachheit vom Eigenwert als Nullstelle des charakteristischen Polynoms $\chi_A(\lambda)$

$$\chi_A(\lambda) = (\lambda_1 - \lambda)(\lambda_2 - \lambda)\dots(\lambda_n - \lambda) \quad \lambda_1, \dots, \lambda_n \text{ sind Eigenwerte} \quad (557)$$

wie viel Mal λ_j als Nullstelle erscheint.

Example 39.

$$A = \begin{pmatrix} 3 & 1 \\ 0 & 3 \end{pmatrix}, \quad \chi_A(\lambda) = \det(A - \lambda I) = (3 - \lambda)(3 - \lambda) \quad (558)$$

$\Rightarrow \lambda = 3$ ist Eigenwert und die algebraische Vielfachheit = 2

$$\text{Die geometrische Vielfachheit} = \dim E_3 = \dim \text{Ker} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$\text{Ker} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = \text{span} \left\{ \underbrace{\begin{pmatrix} 1 \\ 0 \end{pmatrix}}_{\dim=1} \right\} \Rightarrow \text{geometrische Vielfachheit } \lambda = 3 \text{ ist 1} \quad (559)$$

Example 40.

$$B = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix} \quad \chi_B(\lambda) = \det(B - \lambda I) = (3 - \lambda)(3 - \lambda) \quad (560)$$

$\Rightarrow \lambda = 3$ ist Eigenwert von B und die algebraische Vielfachheit = 2

$$E_3^B = ?$$

$$\text{ker}(B - 3I) = \text{ker} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} = \mathbb{R}^2 \Rightarrow E_3^B = \mathbb{R}^2 \quad (561)$$

$\dim E_3^B = 2$ und die geometrische Vielfachheit = 2 von $\lambda = 3$

Theorem 9.13

Für jeden Eigenwert λ gilt:

$$\boxed{\text{geometrische Vielfachheit} \leq \text{algebraische Vielfachheit}} \quad (562)$$

Proof 9.13

$A \in \mathbb{E}^{n \times n}$, sei λ_0 ein Eigenwert von A, $\dim E_{\lambda_0} = k$

Geometrische Vielfachheit von λ_0 ist k

Sei $\underbrace{\{v_1, v_2, \dots, v_k\}}_{\text{linear unabhängig}}$ eine Basis für E_{λ_0}

Ergänzen wir diese Menge zu einer Basis von \mathbb{E}^n :

$$\underbrace{\{v_1, v_2, \dots, v_k\}}_{\text{Basis f. } E\lambda_0} \underbrace{\{v_{k+1}, \dots, v_n\}}_{\text{Ergänzung}} \quad (563)$$

$$\begin{pmatrix} | & | & \cdots & | & | & \cdots & | \\ \bar{v}_1 & \bar{v}_2 & \cdots & \bar{v}_k & \bar{v}_{k+1} & \cdots & \bar{v}_n \\ | & | & \cdots & | & | & \cdots & | \end{pmatrix} \in \mathbb{E}^{n \times n} \text{ diese Matrix ist regulär} \quad (564)$$

$$V^{-1}AV = V^{-1} \begin{pmatrix} | & | & \cdots & | & | & \cdots & | \\ \overline{Av}_1 & \overline{Av}_2 & \cdots & \overline{Av}_k & \overline{Av}_{k+1} & \cdots & \overline{Av}_n \\ | & | & \cdots & | & | & \cdots & | \end{pmatrix} \quad (565)$$

$$= V^{-1} \begin{pmatrix} | & | & \cdots & | & | & \cdots & | \\ \lambda_0 \bar{v}_1 & \lambda_0 \bar{v}_2 & \cdots & \lambda_0 \bar{v}_k & \overline{Av}_{k+1} & \cdots & \overline{Av}_n \\ | & | & \cdots & | & | & \cdots & | \end{pmatrix} \quad (566)$$

weil $V^{-1} \cdot V = I$, gilt $V^{-1}v_j = e_j = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ mit 1 an der j-ten Stelle

$$= \begin{pmatrix} | & | & \cdots & | & | & \cdots & | \\ \lambda_0 e_1 & \lambda_0 e_2 & \cdots & \lambda_0 e_k & V^{-1} \overline{Av}_{k+1} & \cdots & V^{-1} \overline{Av}_n \\ | & | & \cdots & | & | & \cdots & | \end{pmatrix} = \begin{pmatrix} \boxed{\lambda_0 I_{k \times k}} & \boxed{M} \\ 0 & \boxed{T} \end{pmatrix} =: C \quad (567)$$

A hat Eigenwert λ_0 mit geometrischer Vielfachheit = k

$$A = VCV^{-1} \quad C = \begin{pmatrix} \boxed{\lambda_0 I_{k \times k}} & \boxed{M} \\ 0 & \boxed{T} \end{pmatrix} \quad (568)$$

$$\chi_C(\lambda) = \det \begin{pmatrix} \boxed{\lambda_0 I - \lambda I} & \boxed{M} \\ 0 & \boxed{T - \lambda I} \end{pmatrix} = \underbrace{\det((\lambda_0 - \lambda)I)}_{=(\lambda_0 - \lambda)^k} \cdot \underbrace{\det(T - \lambda I)}_{\lambda \dots} \quad (569)$$

\Rightarrow Die algebraische Vielfachheit von λ_0 in C ist $\geq k$

$$C = V^{-1}AV \quad (570)$$

C und A sind ähnlich \equiv Darstellungen von derselben linearen Abbildung in verschiedenen Basen.

A und C haben daselbe charakteristische Polynom, laut nächstem Satz

\Rightarrow Die allgemeine Vielfachheit von λ_0 in Matrix A ist $\geq k$ = geometrische Vielfachheit von λ_0

□

The following was presented in the lecture on 19. December 2018.

Theorem 9.7

Ähnliche Matrizen (A und C) haben dasselbe charakteristische Polynom, die gleiche Spur, die gleiche Determinante und die gleichen Eigenwerte.

Proof 9.7

$$C = T^{-1}AT \quad (571)$$

$$\chi_C(\lambda) = \det(C - \lambda I) = \det(T^{-1}AT - \lambda T^{-1}T) = \det(T^{-1}(A - \lambda I)T) \quad (572)$$

$$= \det(T^{-1})\det(A - \lambda I)\det(T) = \chi_A(\lambda) \quad (573)$$

\Rightarrow Die Koeffizienten von $\chi_C(\lambda)$ und $\chi_A(\lambda)$ sind gleich \Rightarrow zu $\lambda^{n-1} : (-1)^{n-1} \text{Spur } C = (-1)^{n-1} \text{Spur } A$

Nullstellen sind gleich \Rightarrow Eigenwerte sind gleich. \square

Theorem 9.11

Eigenvektoren zu verschiedenen Eigenwerten sind linear unabhängig.

Proof 9.11

Induktion auf die Anzahl verschiedener Eigenwerte. Seien $\lambda_1, \dots, \lambda_m$ verschiedene Eigenwerte.

$m = 1$ Sei v_1 Eigenvektoren zu λ_1 $\lambda_1 \neq 0 \Rightarrow \{v_1\}$ linear unabhängig.

Nehmen wir an die Aussage stimmt für $m = k$ Eigenwerte und beweisen wir für $m = k + 1$ $\lambda_1, \dots, \lambda_k, \lambda_{k+1}$ sind verschiedene Eigenwerte. Seien v_1, \dots, v_k, v_{k+1} Eigenvektoren.

$$\alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_k v_k + \alpha_{k+1} v_{k+1} = 0 \quad (574)$$

Zu zeigen: $\alpha_1 = \dots = \alpha_{k+1} = 0$

$$\alpha_1 \lambda_1 v_1 + \dots + \alpha_k \lambda_k v_k + \alpha_{k+1} \lambda_{k+1} v_{k+1} = 0 \quad (575)$$

$$\alpha_{k+1} v_{k+1} = -\alpha_1 v_1 - \alpha_2 v_2 - \dots - \alpha_k v_k \quad (576)$$

In die obige Gleichung einsetzen.

$$\alpha_1 \lambda_1 v_1 + \dots + \alpha_k \lambda_k v_k + \lambda_{k+1}(-\alpha_1 v_1 - \alpha_2 v_2 - \dots - \alpha_k v_k) = 0 \quad (577)$$

$$\alpha_1 \cdot (\lambda_1 - \lambda_{k+1}) \cdot v_1 + \alpha_2 \cdot (\lambda_2 - \lambda_{k+1}) \cdot v_2 + \dots + \alpha_k \cdot (\lambda_k - \lambda_{k+1}) \cdot v_k = 0 \quad (578)$$

$\{v_1, \dots, v_k\}$ sind linear unabhängig $\Rightarrow \forall j = 1, \dots, k \ \alpha_j \underbrace{(\lambda_j - \lambda_{k+1})}_{\neq 0} = 0 \Rightarrow \alpha_j = 0$

$$0v_1 + \dots + 0v_k + \alpha_{k+1} v_{k+1} = 0 \quad (579)$$

$$\Rightarrow \alpha_{k+1} = 0 \Rightarrow \alpha_1 = \alpha_2 = \dots = \alpha_k = \alpha_{k+1} = 0 \quad (580)$$

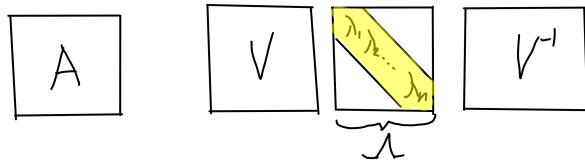
\square

Corollary 9.1

Eine lineare Abbildung kann maximal $n = \dim V$ verschiedene Eigenwerte haben.

9.6 Spektralzerlegung / Eigenwertzerlegung (spectral decomposition)

Wenn \mathbb{E}^n eine Basis aus Eigenvektoren von A besitzt: $\{v_1, \dots, v_n\}$ und $\lambda_1, \dots, \lambda_n$ die entsprechenden Eigenwerte sind (nicht zwingend verschieden) dann ist A ähnlich zu einer Diagonalmatrix Λ :



$$AV = V\Lambda \Rightarrow A = V\Lambda V^{-1} \quad (581)$$

Wir sagen, dass A diagonalisierbar ist, oder "diag durch V "

Note 58. v_1, \dots, v_n lineare unabhängige Eigenvektoren von $A \in \mathbb{E}^{n \times n}$

$$Av_1 = \lambda_1 v_1 \quad (582)$$

$$Av_2 = \lambda_2 v_2 \quad (583)$$

$$\dots \quad (584)$$

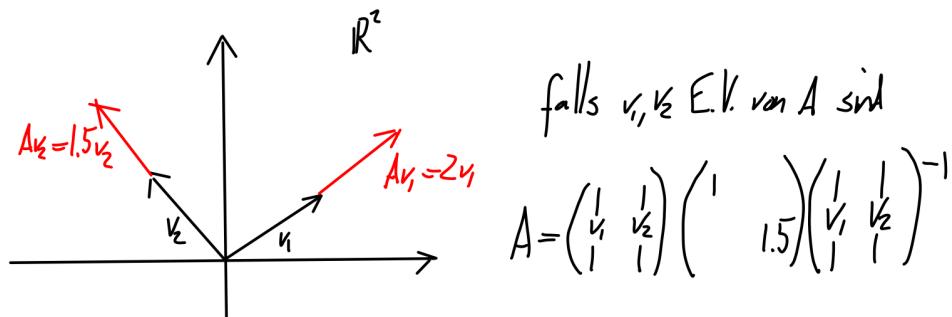
$$Av_n = \lambda_n v_n \quad (585)$$

$\lambda_1, \dots, \lambda_n$ sind Eigenwerte und nicht unbedingt verschieden.

$$A \underbrace{\begin{pmatrix} | & \cdots & | \\ \overline{v}_1 & \cdots & \overline{v}_n \\ | & \cdots & | \end{pmatrix}}_V = \underbrace{\begin{pmatrix} | & \cdots & | \\ \overline{v}_1 & \cdots & \overline{v}_n \\ | & \cdots & | \end{pmatrix}}_V \underbrace{\begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}}_{\Lambda} \quad (586)$$

$$A = V\Lambda V^{-1} \quad (587)$$

$\Rightarrow A$ ist eine Skalierung der Achsen v_1, v_2, \dots, v_n



Example 41.

$$\begin{pmatrix} 2 & -2 & 2 \\ 0 & 1 & 1 \\ 3 & 0 & 2 \end{pmatrix} \in \mathbb{E}^{3 \times 3} \quad (588)$$

$$\chi_A(\lambda) = \det(A - \lambda I) = \dots = (2 - \lambda)(1 + \lambda)(-4 + \lambda) \Rightarrow \lambda_1 = 2, \lambda_2 = -1, \lambda_3 = 4 \quad (589)$$

$\Rightarrow A$ hat 3 verschiedene Eigenwerte $\Rightarrow A$ hat 3 lineare unabhängige Eigenvektoren.

$\Rightarrow \mathbb{E}^3$ hat eine Basis aus Eigenvektoren von $A \Rightarrow A$ ist diagonalisierbar.

$$\exists V \in \mathbb{E}^{3 \times 3}, \quad A = V \begin{pmatrix} 2 & & \\ & -1 & \\ & & 4 \end{pmatrix} V^{-1} \quad (590)$$

Example 42.

$$A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad (591)$$

Drehung um 90 deg in der Ebene \mathbb{R}^2

$$\chi_A = \det \begin{pmatrix} -\lambda & -1 \\ 1 & -\lambda \end{pmatrix} = \lambda^2 + 1 = (\lambda - 1)(\lambda + 1) \quad (592)$$

A hat keine reellen Eigenwerte $\Rightarrow A$ ist nicht diagonalisierbar in \mathbb{R}

A hat 2 verschiedene Eigenwerte über $\mathbb{C} : i, -i$

$\Rightarrow A$ ist diagonalisierbar über \mathbb{C}

$$\text{Löse } \begin{pmatrix} -i & -1 \\ 1 & -i \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \begin{pmatrix} i & -1 \\ 1 & i \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (593)$$

$$\text{z.B. } v_1 = \begin{pmatrix} i \\ 1 \end{pmatrix}, \quad v_2 = \begin{pmatrix} -i \\ 1 \end{pmatrix}$$

Example 43.

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \chi_A(\lambda) = \det \begin{pmatrix} 1 - \lambda & 1 \\ 0 & 1 - \lambda \end{pmatrix} = (1 - \lambda)^2 \quad (594)$$

$\Rightarrow A$ hat einen Eigenwert $\lambda = 1 \dim E_{\lambda=1} ?$

Example 44.

$$A - 1I = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (595)$$

$\Rightarrow x_2 = 0, x_1$ ist frei wählbar.

$\Rightarrow \dim E_{\lambda=1} = 1 < 2$

\Rightarrow Es gibt keine Eigenbasis für $\mathbb{E}^2 \Rightarrow A$ ist nicht diagonalisierbar. (Weder über \mathbb{C} noch über \mathbb{R})

Theorem 9.14

$A \in \mathbb{C}^{n \times n}$ ist diagonalisierbar (über \mathbb{C})

\Leftrightarrow die geometrische Vielfachheit jedes Eigenwertes = seiner algebraischen Vielfachheit.

Note 59. geometrische Vielfachheit \leq algebraische Vielfachheit

Proof 9.14

Eine Matrix A diagonalisierbar $\Leftrightarrow \exists$ Eigenbasis von \mathbb{C}^n
A hat Eignewerte $\lambda_1, \dots, \lambda_k$ (verschieden)

$$E_{\lambda_1} = \text{span} \left\{ v_1^1, \dots, \underbrace{v_{d(\lambda_1)}^1}_{\text{geometrische Vielfachheit von } \lambda_1} \right\}, \dots, E_{\lambda_k} = \text{span} \left\{ v_1^k, v_2^k, \dots, v_{d(\lambda_k)}^k \right\} \quad (596)$$

$$\dim E_{\lambda_1} d(\lambda_1), \dots, \dim E_{\lambda_k} = d(\lambda_k) \quad (597)$$

Eine lineare unabhängige Menge:

$$\left\{ \underbrace{v_1^1, v_2^1, \dots, v_{d(\lambda_1)}^1}_{\text{algebraische Vielfachheit } \lambda_1}, \underbrace{v_1^2, \dots, v_{d(\lambda_2)}^2}_{\text{algebraische Vielfacheit } \lambda_2}, \dots, \underbrace{v_1^k, \dots, v_{d(\lambda_k)}^k}_{\text{algebraische Vielfachheit } \lambda_k} \right\} \quad (598)$$

$$\text{Eine Eigenbasis } \Leftrightarrow d(\lambda_1) + d(\lambda_2) + \dots + d(\lambda_k) = n \quad (599)$$

$$\chi_A(\lambda) = (-1)^n (\lambda - \lambda_1)^{a_1} (\lambda - \lambda_2)^{a_2} \dots (\lambda - \lambda_k)^{a_k} \quad (600)$$

$$a_j = \text{algebraische Vielfachheit von } \lambda_j \quad (601)$$

$$a_1 + a_2 + \dots + a_k = n \quad (602)$$

$$d(\lambda_j) \leq a_j \quad \forall j = 1, \dots, k \quad (603)$$

$$\exists \text{ Eigenbasis } \Leftrightarrow d(\lambda_1) + \dots + d(\lambda_k) = n \Leftrightarrow d(\lambda_j) = a_j \quad \forall j = 1, \dots, k \quad (604)$$

□

9.7 Eigenwertzerlegung symmetrischer und hermitischer Matrizen (the spectral theorem)

Theorem 9.15

$A \in \mathbb{C}^{n \times n}$ hermitesch $A^H = A$

1. Alle Eigenwerte $\lambda_1, \dots, \lambda_n$ sind reel
2. Die Eigenvektoren zu verschiedenen Eigenwerten sind paarweise orthogonal in \mathbb{C}^n
3. Es gibt eine orthonormale Basis des \mathbb{C}^n mit Eigenvektoren u_1, \dots, u_n
4. Für die unitären Matrix $U = \begin{pmatrix} | & \cdots & | \\ \overline{u}_1 & \cdots & \overline{u}_n \\ | & \cdots & | \end{pmatrix}$ gilt:

$$U^H A Y = \Lambda = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \dots & \lambda_n \end{pmatrix} \quad (605)$$

Corollary 9.16

1. Alle Eigenwerte $\lambda_1, \dots, \lambda_n \in \mathbb{R}$
2. Die reellen Eigenvektoren zu verschiedenen Eigenwerten sind paarweise orthogonal in \mathbb{R}^n
3. Es gibt eine orthonormale Basis des \mathbb{R}^n mit Eigenvektoren u_1, \dots, u_n
4. Für die orthogonale Matrix $U = \begin{pmatrix} | & \cdots & | \\ \overline{u}_1 & \cdots & \overline{u}_n \\ | & \cdots & | \end{pmatrix}$ gilt:

$$U^T A U = \Lambda = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \cdots & & & \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix} \quad (606)$$

Proof 9.15

Von (1) und (2):
Seien $u, v \in \mathbb{E}^n$

$$\langle u, Av \rangle = u^H (Av) \underset{A=A^H}{=} (u^H A^H)v = (Au)^H v = \langle Au, v \rangle \quad (607)$$

Nehmen wir an $u \neq 0, v \neq 0, Au = \lambda u, Av = \mu v$ (u, v sind Eigenvektoren)

$$\langle u, Av \rangle = \langle Au, v \rangle \quad (608)$$

$$\langle u, \mu v \rangle = \langle \lambda u, v \rangle \quad (609)$$

$$\mu \langle u, v \rangle = \bar{\lambda} \langle u, v \rangle \quad (610)$$

$$\Rightarrow (\mu - \bar{\lambda}) \langle u, v \rangle = 0 \quad (611)$$

1.

$$\lambda = \mu \Rightarrow (\mu - \bar{\mu}) \underbrace{\langle v, v \rangle}_{\neq 0} = 0 \Rightarrow \mu - \bar{\mu} = 0 \Rightarrow \mu = \bar{\mu} \Rightarrow \mu \in \mathbb{R} \quad (612)$$

\Rightarrow Alle Eigenwerte sind reell. (1)

2.

$$\lambda \neq \mu \Rightarrow (\mu - \lambda) \underbrace{\langle u, v \rangle}_{\neq 0} = 0 \Rightarrow \langle u, v \rangle = 0 \Rightarrow u \perp v \quad (2) \quad (613)$$

□

The following was presented in the lecture on 21. December 2018.

$$A \in \mathbb{E}^{n \times n} \quad A^H = H \text{ (Hermitisch)} \quad (614)$$

$$\Rightarrow A = U \Lambda U^H \quad (615)$$

\Rightarrow Alle Hermiteschen / reelsymmetrischen Matrizen sind Skalierungen von orthonormalen Axen.

10 Die Spektralnorm und $\kappa_2(A)$

$A \in \mathbb{E}^{n \times n}$, A regulär \Rightarrow Konditionszahl $\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2$

Erinnerung: $\|A\|_2 = \sup_{\|x\|_2=1} \|Ax\|_2$

$\|A\|_2, \|A^{-1}\|_2$ und $\kappa_2(A)$ ist was? Mithilfe Spektralzerlegung!

Wir kennen $\|A\|_2$ für diagonale Matrizen A:

$$\left\| \begin{pmatrix} d_1 & & & \\ & d_2 & & \\ & & .. & \\ & & & d_n \end{pmatrix} \right\|_2 = \max_j |d_j| \quad (616)$$

Eine beliebige $A \in \mathbb{E}^{n \times n}$ ist nicht immer diagonalisierbar aber $A^H A$ ist Hermitisch und daher diagonalisierbar.

Wir werden $\|A\|_2$ aus der Spektralzerlegung von $A^H A$ kriegen.

$$(A^H A)^H = A^H (A^H)^H = A^H A; \quad (AA^H)^H = (A^H)^H A^H = AA^H \quad (617)$$

Die obige Gleichung gilt auch wenn A rechteckig ist.

Definition 81. $A \in \mathbb{E}^{n \times n}$ und $A^H = A$ Die Matrix A heisst positiv definit falls:

$\forall x \in \mathbb{E}^n \ x \neq 0$ gilt $x^H A x > 0$

A heisst positiv semidefinit, falls $\forall x \in \mathbb{E}^n \ x^H A \geq 0$

Theorem 10.1

$A \in \mathbb{E}^{n \times n} \ A^H = A$ und A positiv definit \Rightarrow alle Eigenwerte von A > 0

$A \in \mathbb{E}^{n \times n} \ A^H = A$ und A positiv semidefinit \Rightarrow alle Eigenwerte von A ≥ 0

Proof 10.1

A ist hermitisch $\exists U$ unitär, so dass

$$A = U \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & .. & \\ & & & \lambda_n \end{pmatrix} U^H, \quad \lambda_j \in \mathbb{R} \quad (618)$$

Sei $x \in \mathbb{E}^n$

$$x^H A x = x^H U \Lambda U^H x = (U^H x)^H \Lambda (U^H x) \quad (619)$$

A positiv semidefinit $\Rightarrow (U^H x)^H \Lambda (U^H x) \geq 0$ Sei $y := U^H x$

$$\Rightarrow y^H \Lambda y \geq 0 \quad \forall y \in \mathbb{E}^n \exists x \text{ so dass } y = U^H x \quad (620)$$

$$\Rightarrow [y \in \mathbb{E}^n \text{ gilt } y^H \Lambda y \geq 0] \quad (621)$$

Daher, insbesondere für alle $y = e_j, j = 1, \dots, n$: $E_J^H \Lambda e_j \geq 0$

$$0 \leq e_j \Lambda e_j = (0, \dots, 0, 1, 0, \dots, 0) \text{ an der } j\text{-ten Stelle} \begin{pmatrix} \lambda_1 & & & & & 0 \\ & \ddots & & & & \vdots \\ & & \lambda_j & & & 1 \\ & & & \ddots & & 0 \\ & & & & \lambda_n & .. \\ & & & & & 0 \end{pmatrix} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ .. \\ 0 \end{pmatrix} = \lambda_j \quad (622)$$

$$\Rightarrow \forall j = 1, \dots, n \quad \lambda_j \geq 0$$

Äquivalent für positiv definit ($\lambda_j > 0$) □

Corollary 10.2

$$f(x, y) := x^H A y \quad (623)$$

ist ein Skalarprodukt in \mathbb{E}^n wenn A Hermitesch und positiv definit ist.

Theorem 10.7

Die Spektralnorm von $A \in \mathbb{E}^{m \times n}$, $\|A\|_2 = \max\{\sqrt{\lambda}, \lambda \text{ Eigenwert von } A^H A\}$

Proof 10.7

$$\|A\|_2^2 = \left(\sup_{\|x\|_2=1} \underbrace{\|Ax\|_2}_{\geq 0} \right)^2 = \sup_{\|x\|_2=1} \|Ax\|_2^2 \quad (624)$$

$$\|Ax\|_2^2 = (Ax)^H (Ax) = x^H A^H A x \quad (625)$$

Die Matrix $A^H A$ ist positiv semidefinit

$A^H A$ ist Hermitisch

$$\forall x \in \mathbb{E}^n \quad x^H (A^H A) x = (Ax)^H (Ax) = \|Ax\|_2^2 \geq 0 \quad (626)$$

$$\Rightarrow A^H A = U \underbrace{\begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & .. & \\ & & & \lambda_n \end{pmatrix}}_{\Lambda} U^H \quad (627)$$

wobei $U \in \mathbb{E}^{n \times n}$ unitär ist und $\lambda_j \geq 0 \quad \forall j = 1, \dots, n$
 U, U^H sind unitär, daher:

$$\forall y \in \mathbb{E}^n \quad \exists x \in \mathbb{E}^n \text{ sodass } y = U^H x \text{ und } \underbrace{\|y\|_2 = \|x\|_2}_{U^H \text{ ist längentreu}} \quad (628)$$

Man kann die gleiche Methode verwenden, wie im Beweis von vorhin.

$$\|A\|_2^2 = \sup_{\|x\|_2=1} x^H A^H A x = \sup_{\|x\|_2=1} (x^H U) \Lambda (U^H x) = \sup_{\|x\|_2=1} \underbrace{(U^H x)}_y \Lambda \underbrace{(U^H x)}_y \quad (629)$$

$$\sup_{\|y\|_2=1} y^H \Lambda y = \sup_{\|y\|_2=1} (y^H \Lambda^{\frac{1}{2}}) (\Lambda^{\frac{1}{2}} y) = \sup_{\|y\|_2=1} \|\Lambda^{\frac{1}{2}} y\|_2^2 = \|\Lambda^{\frac{1}{2}}\|_2^2 \quad (630)$$

$$\Lambda^{\frac{1}{2}} = \begin{pmatrix} \sqrt{\lambda_1} & & & \\ & \sqrt{\lambda_2} & & \\ & & .. & \\ & & & \sqrt{\lambda_n} \end{pmatrix} \quad \sqrt{\lambda_j} \in \mathbb{R} \text{ weil } \lambda_j \geq 0 \quad (631)$$

$$\Rightarrow \|A\|_2^2 = \|\Lambda^{\frac{1}{2}}\|_2^2 \Rightarrow \|A\|_2 = \|\Lambda^{\frac{1}{2}}\|_2 \quad \stackrel{=}{{}_{\text{2-Norm einer Diagonalmatrix}}} \quad \max_{j=1,\dots,n} \sqrt{\lambda_j} \quad (632)$$

$$= \max\{\sqrt{\lambda}, \lambda \text{ ist Eigenwert von } A^H A\} \quad (633)$$

□

Corollary 10.3

1. $A \in \mathbb{E}^{n \times n}$ Hermitisch $\Rightarrow \|A\|_2 = \max\{|\lambda|, \lambda \text{ Eigenwert von } A\}$
2. $A \in \mathbb{E}^{n \times n}$ unitär $\Rightarrow \|A\|_2 = 1$

Für $\kappa_2(A)$ fehlt noch $\|A^{-1}\|_2$

Theorem 10.8

Seien $B, C \in \mathbb{E}^{n \times n}$. Die Matrizen BC und CB haben die gleichen Eigenwerte. Daher λ ist Eigenwert von BC $\Leftrightarrow \lambda$ ist Eigenwert von CB

Proof 10.8

Übung

□

$$\|A^{-1}\|_2 = ? \quad \|A^{-1}\|_2 = \max\{\sqrt{\mu}, \mu \text{ Eigenwert von } (A^{-1})^H (A^{-1})\} \quad (634)$$

$$(A^{-1})^H (A^{-1}) = (A^H)^{-1} A^{-1} = (A \cdot A^H)^{-1} \quad (635)$$

$$AA^H \text{ Hermitisch} \Rightarrow AA^H = W \Lambda W^H \quad (636)$$

$$(AA^H)^{-1} = (W \Lambda W^H)^{-1} = (W^H)^{-1} \Lambda^{-1} W^{-1} = W \begin{pmatrix} \frac{1}{\lambda_1} & & & \\ & \frac{1}{\lambda_2} & & \\ & & .. & \\ & & & \frac{1}{\lambda_n} \end{pmatrix} W^H \quad (637)$$

$$\|A^{-1}\|_2 = \max_{j=1,\dots,n} \left\{ \sqrt{\frac{1}{\lambda_j}}, \lambda_j \text{ Eigenwert von } A^H A \right\} = \max_{j=1,\dots,n} \left\{ \frac{1}{\sqrt{\lambda_j}}, \lambda_j \text{ Eigenwert von } A^H A \right\} \quad (638)$$

$$= \frac{1}{\min\{\sqrt{\lambda}, \lambda \text{ Eigenwert von } A^H A\}} \quad (639)$$

$$\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \frac{\max\{\sqrt{\lambda}, \lambda \text{ Eigenwert von } A^H A\}}{\min\{\sqrt{\lambda}, \lambda \text{ Eigenwert von } A^H A\}} \quad (640)$$

A Hermitisch oder reell-symmetrisch:

$$\kappa_2(A) = \frac{\max\{\sqrt{\lambda}, \lambda \text{ Eigenwert von } A\}}{\min\{\sqrt{\lambda}, \lambda \text{ Eigenwert von } A\}} = \frac{|\lambda|_{\max}}{|\lambda|_{\min}} \quad (641)$$

$\forall A, \kappa_2(A) \geq 1$

Example 45.

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 3.999 \end{pmatrix} \quad \kappa_2(A) \approx 25'000 \quad (642)$$

Example 46.

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \quad \kappa_2(A) \approx 18 \quad (643)$$

Example 47.

$$A = \begin{pmatrix} \epsilon & & & \\ & \epsilon & & \\ & & .. & \\ & & & \epsilon \end{pmatrix} \quad \kappa_2(A) = 1 \text{ (A ist bestens konditioniert)} \quad (644)$$

Example 48.

$$A = \begin{pmatrix} N & 0 \\ 0 & \frac{1}{N} \end{pmatrix} \quad \kappa_2(A) = N^2 \quad (645)$$

Aber Achtung $\det A = N \cdot \frac{1}{N} = 1$

A ist schlecht konditioniert für grosse N, aber $\det A$ ist konstant

$\Rightarrow \det$ ist keine gute Vorhersage der Konditionierung einer Matrix.

10.1 Geometrische Intuition für Eigenwertzerlegung

$A : \mathbb{E}^n \rightarrow \mathbb{E}^n$ A diagonalisierbar \Rightarrow A ist eine Skalierung der Achsen $\underbrace{\{v_1, v_2, \dots, v_n\}}_{\text{Eigenbasis}}$

$$A \begin{pmatrix} | & \cdots & | \\ \bar{v}_1 & \cdots & \bar{v}_n \\ | & \cdots & | \end{pmatrix} = \begin{pmatrix} | & \cdots & | \\ \lambda_1 \bar{v}_1 & \cdots & \lambda_n \bar{v}_n \\ | & \cdots & | \end{pmatrix} \quad AV = V \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & .. & \\ & & & \lambda_n \end{pmatrix} \quad (646)$$

$A : \mathbb{E}^n \rightarrow \mathbb{E}^n$ A Hermitesch \Rightarrow A ist eine Skalierung der orthonormalen Achsen $\underbrace{\{u_1, u_2, \dots, u_n\}}_{\text{Eigenbasis}}$

$$A \begin{pmatrix} | & \cdots & | \\ \overline{u}_1 & \cdots & \overline{u}_n \\ | & \cdots & | \end{pmatrix} = \begin{pmatrix} | & \cdots & | \\ \lambda_1 \overline{u}_1 & \cdots & \lambda_n \overline{u}_n \\ | & \cdots & | \end{pmatrix}, \quad AU = U \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & .. & \\ & & & \lambda_n \end{pmatrix} \quad (647)$$

11 Singulärwertzerlegung (Nicht Prüfungsrelevant)

$$\forall A \in \mathbb{E}^{n \times n}, \underbrace{\exists V \in \mathbb{E}^{n \times n}, U \in \mathbb{E}^{m \times m}}_{U, V \text{ unitär}}, \underbrace{\sum}_{\Sigma \text{ diagonal}} \in \mathbb{E}^{m \times n} \quad (648)$$

$$A = U \sum V^H \quad (649)$$

$A \in \mathbb{E}^{m \times n}$ ($m \geq n$):

$$A \begin{pmatrix} | & \cdots & | \\ \overline{v}_1 & \cdots & \overline{v}_n \\ | & \cdots & | \end{pmatrix} = \begin{pmatrix} | & \cdots & | \\ \tau_1 \overline{u}_1 & \cdots & \tau_n \overline{u}_n \\ | & \cdots & | \end{pmatrix} = U \sum \quad (650)$$

Singulärwerte: $\tau_1 \geq \tau_2 \geq \dots \geq \tau_n \geq 0$

$\forall A \in \mathbb{E}^{m \times n}$, $A^H A$ ist Hermitesch und positiv semidefinit.

$$\Rightarrow A^H A = V \Lambda V^H = \begin{pmatrix} | & \cdots & | \\ \overline{v}_1 & \cdots & \overline{v}_n \\ | & \cdots & | \end{pmatrix} \begin{pmatrix} \sigma_1^2 & & & \\ & \ddots & & \\ & & \sigma_r^2 & \\ & & & 0 \end{pmatrix} (V^H) \quad r = \text{Rang } A = \text{Rang}(A^H A) \quad (651)$$

$$A^H A \cdot \begin{pmatrix} | & \cdots & | & | & \cdots & | \\ \overline{v}_1 & \cdots & \overline{v}_r & \overline{v}_{r+1} & \cdots & \overline{v}_n \\ | & \cdots & | & | & \cdots & | \end{pmatrix} = \begin{pmatrix} | & \cdots & | & | & \cdots & | \\ \overline{v}_1 & \cdots & \overline{v}_r & \overline{v}_{r+1} & \cdots & \overline{v}_n \\ | & \cdots & | & | & \cdots & | \end{pmatrix} \begin{pmatrix} \sigma_1^2 & & & & & \\ & \ddots & & & & \\ & & \sigma_r^2 & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \quad (652)$$

Das Gleiche wie:

$$A^H A \cdot V_r \cdot V^{\perp} = V_r \cdot V^{\perp} \cdot \begin{pmatrix} \sigma_1^2 & & & & & \\ & \ddots & & & & \\ & & \sigma_r^2 & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \quad (653)$$

$$A^H A \cdot V_r = V_r \begin{pmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_r^2 \end{pmatrix} \quad (654)$$

$V_r^H = I$, weil die Spalten von V_r orthonormal sind.

$$V_r^H A^H A V_r = \begin{pmatrix} \sigma_1^2 & & \\ & \dots & \\ & & \sigma_r^2 \end{pmatrix} = \begin{pmatrix} \sigma_1 & & \\ & \dots & \\ & & \sigma_r \end{pmatrix} \cdot \begin{pmatrix} \sigma_1 & & \\ & \dots & \\ & & \sigma_r \end{pmatrix} \quad (655)$$

$$\sum_r := \begin{pmatrix} \sigma_1 & & \\ & \dots & \\ & & \sigma_r \end{pmatrix} \quad (656)$$

Multiplizieren wir (655) mit \sum_r^{-1} von links und von rechts:

$$V_r^H A^H A V_r = \sum_r \sum_r \quad (657)$$

$$\sum_r^{-1} V_r^H A^H A V_r \sum_r^{-1} = I_{r \times r} \quad (658)$$

$$\underbrace{\left(A V_r \sum_r^{-1} \right)^H}_{:= U_r^H} \underbrace{\left(A V_r \sum_r^{-1} \right)}_{:= U_r} = I_{r \times r} \quad (659)$$

$$U_r = A V_r \cdot \begin{pmatrix} \frac{1}{\sigma_1} & & \\ & \dots & \\ & & \frac{1}{\sigma_r} \end{pmatrix} \quad (660)$$

$$U_r^H U_r = I \Rightarrow U_r \text{ hat orthonormale Spaltenvektoren.} \quad (661)$$

$$A V_r \sum_r^{-1} = U_r \quad (662)$$

Nun hat man die reduzierte Form von SVD:

$$A V_r = U_r \sum_r = \underbrace{U_r}_{\text{orthonormale Spalten}} \begin{pmatrix} \sigma_1 & & \\ & \dots & \\ & & \sigma_r \end{pmatrix} \quad (663)$$

Ergänzen wir die Spaltenvektoren von $U_r \in \mathbb{E}^{m \times r}$ zu einer orthonormalen Basis von \mathbb{E}^m daher ergänzen wir U_r zu einer unitären Matrix $U \in \mathbb{E}^{m \times m}$

$$U := (U_r \quad | \quad U^\perp) \quad U U^H = U^H U = I_{m \times m} \quad (664)$$

Daraus folgt die full-form SVD

$$\Rightarrow A (V_r \quad | \quad V^\perp) = (U_r \quad | \quad U^\perp) \begin{pmatrix} \sum_r & | & 0 \\ 0 & | & 0 \end{pmatrix} \quad (665)$$

U, V sind unitär, Σ diagonal und \geq und w.l.o.g. $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$

$$A = U \sum V^H, \quad A = \sum_{k=1}^r u_k \sigma_k v_k^H \quad (666)$$

Note 60. Die Matrix U diagonalisiert $A^H = U \begin{pmatrix} \sum_r^2 & | & 0 \\ 0 & | & 0 \end{pmatrix} U^H$

Corollary 11.1

$$\|A\|_2 = \sigma_1, \quad \kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \frac{\sigma_1}{\sigma_n} = \frac{\text{größter Singulärwert}}{\text{kleinster Singulärwert}} \quad (667)$$

Theorem 11.1: Zentraler Satz über SVD

Sei $A = U \Sigma V^H \in \mathbb{E}^{m \times n}$

$$A_p := \sum_{k=1}^p u_k \tau_k v_k^H, \quad \text{wobei } p \leq r = \text{Rang } A \quad \text{Rang } A_p = p \quad (668)$$

Dann gilt:

A_p ist die beste Approximation von A mit einer Matrix mit Rang p , im Sinne der 2-Norm:

$$\forall B \in \mathbb{E}^{m \times n}, \quad \text{Rang } B = p \text{ gilt: } \|A - B\|_2 \geq \|A - A_p\|_2 \quad (669)$$

Und es gilt:

$$\|A - A_p\|_2 = \sigma_{p+1} \quad (670)$$

Lineare Algebra - Exercise

Prof. Olga Sorkine-Hornung and Prof. Özlem Imamoglu
TA: Adrian Spurr
Im E-Mail Header: [LinAlg HS18]

Contents

1 Komplexe Zahlen	4
1.1 Definitionen	4
1.2 Komplexe Zahlen als Vektoren	4
1.3 Arithmetik der komplexen Zahlen	4
1.4 Weitere Definintionen von z	4
1.5 Weitere Identitäten	5
1.6 Polarform darstellen	5
1.7 Multiplikation zweier komplexen Zahlen	5
1.8 Gleichungen lösen	5
2 Matrizen	7
2.1 Menge der Matrizen	7
2.2 Spezielle Matrizen	8
2.3 Matrixoperationen	8
2.4 Rechenregeln	8
2.5 Linearkombination	8
2.6 Skalarprodukt	10
2.7 Eigenschaften	10
2.8 Euklidische Norm $x \in \mathbb{E}^n$	10
2.9 Eigenschaften $\forall x, y \in \mathbb{R}^n, \forall \alpha \in \mathbb{R}$	10
2.10 Winkel zwischen zwei Vektoren	10
3 Lineare Gleichungssysteme	11
3.1 Wie findet man x ?	11

3.2 Serie 3 - Cauchy-Schwarz Ungleichung (10/10)	11
3.3 Gauss Algorithmus	13
3.4 Gauss Elimination & LR-Zerlegung	14
3.4.1 Permutationsmatrix	14
3.4.2 Rang	14
3.4.3 Verträglichkeitsbedingungen für $m > r$	15
3.4.4 Homogenes LGS, also $Ax = 0$	15
3.5 Inverse einer Matrix	15
3.5.1 Inverse einer Matrix berechnen	15
3.6 Serie 4 (10/10)	16
3.7 Unitäre / Orthogonale Matrizen	17
3.8 LR-Zerlegung	17
3.8.1 Warum braucht es eine LR-Zerlegung	17
3.8.2 LR-Zerlegung mit Permutation	18
4 Vektorräume	18
4.1 Vektorraum Axiome ($V, +, \cdot$)	18
4.2 Unterraum	19
4.3 Serie 5 (10/10)	19
4.4 Kommutation Skalarprodukt	22
4.5 Linearkombination	22
4.6 Lineare Abhängigkeit (VRV)	23
4.7 Basis	23
4.8 Dimension eines Vekotrraumes	24
4.9 Koordinaten im Vektorraum	24
5 Lineare Abbildung	24
5.1 Bild	25
5.2 Kern	25

5.3	Dimensionsformel	25
5.4	Basiswechsel	25
5.5	Serie 7 (9/10)	26
5.6	Abbildungen Begriffe und Eigenschaften	27
5.7	Lineare Abbildungen	28
5.8	Kern	28
5.9	Rang	28
5.10	Matrix als lineare Abbildung	28
6	Vektorräume mit Skalarprodukt	29
6.1	Norm	29
6.2	Skalarprodukt	29
6.3	Spur	29
6.4	Cauchy-Schwarz Ungleichung	30
6.5	Orthogonalität	30
6.6	Orthonormalbasen	30
6.7	Parsevalsche Formel	31
6.8	Gram-Schmidt-Orthogonalisierungsverfahren	31
6.9	Basiswechsel und Koordinaten Transformation von orthagonalen Basen	32
6.10	Darstellung von linearen Abbildungen	32
6.11	Orthogonale und unitäre Abbildungen	32
6.12	Induzierte Matrizen	33
6.13	Induzierte Operationen	33
6.14	Konditionszahl einer regulären Matrix $A \in \mathbb{E}^{m \times n}$	34
7	Methode der kleinsten Quadrate	34
8	Determinante	36
8.1	Permutationen	36

9	Definition der Determinante	36
10	Eigenwerte und Eigenvektoren	39
10.1	Charakteristische Gleichung	39

The following was presented in the exercise on 25. September 2018.

1 Komplexe Zahlen

1.1 Definitionen

Definition 1. Die Zahl i nennt man die imaginäre Einheit. Sie ist definiert als $i^2 := -1$

Definition 2. Eine Zahl z der Form $z = x + iy$ nennt man eine komplexe Zahl, wobei $x, y \in \mathbb{R}$ reelle Zahlen sind.

Definition 3. Die Menge der komplexen Zahlen nennt man \mathbb{C} . z ist durch x, y eindeutig festgelegt $x + iy = a + ib \Leftrightarrow x = a \wedge b = y$

Definition 4. x heisst Realteil von $z = x + iy$ $x = Re(z)$
 y heisst Imaginärteil von z $y = Im(z)$

1.2 Komplexe Zahlen als Vektoren

$$z = x + iy \Leftrightarrow (x, y) \in \mathbb{R}^2 \quad (1)$$

1.3 Arithmetik der komplexen Zahlen

$$z = x + iy, w = u + iv, x, y, u, v, \alpha \in \mathbb{R} \quad (2)$$

Addition: $z + w = x + iy + u + iv = (x + u) + i(y + v)$

Multiplikation mit Skalaren: $\alpha z = \alpha(x + iy) = ax + i\alpha y$

Multiplikation: $zw = (x + iy)(u + iv) = xu + ixu + iyv + i^2 yv = (xu - yv) + i(xu + yv)$

1.4 Weitere Definitionen von z

Definition 5. $z = x + iy$, dann ist $\bar{z} := x - iy$, die konjugiert komplexe Zahl von z

Definition 6. $|z| = \sqrt{z\bar{z}}$, ist eine nicht-negative reelle Zahl und heisst Betrag / Absolut Betrag der komplexen Zahl z

Definition 7. Division: $c = a + ib, w \neq 0$

$$\frac{z}{w} = c \Leftrightarrow z = cw \Leftrightarrow x + iy = (a + ib)(u + iv) \Leftrightarrow x + iy = (au - bv) + i(av + bu)$$

$$x = au - bv \Rightarrow a = \frac{xu + yv}{u^2 - v^2}$$

$$y = av + bu \Rightarrow b = \frac{yu - xv}{u^2 + v^2}$$

$$c = \frac{(xu + yv) + i(yu - xv)}{u^2 + v^2} = \frac{(x + iy)(u - iv)}{|w|^2} = \boxed{\frac{z \cdot \bar{w}}{|w|^2}} = \frac{z}{w} \quad (3)$$

1.5 Weitere Identitäten

$$\overline{zw} = \bar{z} \cdot \bar{w}$$

$$\overline{\bar{z}} = z$$

$$\left(\frac{z}{w}\right) = \frac{\bar{z}}{\bar{w}}$$

$$|z| = |\bar{z}|$$

$$|z + w| \leq |x| + |y|$$

$$|zw| = |z| \cdot |w|$$

$$\left|\frac{z}{w}\right| = \frac{|z|}{|w|}$$

1.6 Polarform darstellen

$$r = |z| \quad \phi = \arccos \frac{x}{r} = \arcsin \frac{y}{r}$$

$$(r, \phi) = (r, \phi + 2\pi)$$

$$(r, \phi) = (-r, \phi + \pi)$$

$$\phi \in [0, 2\pi)$$

$$x = r \cdot \cos \phi$$

$$y = r \cdot \sin \phi$$

$$z = x + iy = r(\cos \phi + i \sin \phi) \quad (4)$$

Euler Formel:

$$e^{i\phi} = \cos \phi + i \sin \phi \quad (5)$$

$$\boxed{z = r \cdot e^{i\phi}} \quad (6)$$

1.7 Multiplikation zweier komplexen Zahlen

$$z = re^{i\phi}$$

$$w = se^{i\theta}$$

$$zw = r \cdot se^{i(\phi+\theta)}$$

1.8 Gleichungen lösen

$$z^q = c \quad c, z \in \mathbb{C}, q \in \mathbb{N}$$

q unbekannt

$$\begin{aligned}
z &= re^{i\phi} \\
c &= se^{i\theta} \\
z^q = c &\Leftrightarrow (re^{i\phi})^q = se^{i\theta} \\
r^q \cdot e^{i\phi q} &= se^{i\theta} \\
r^q = s &\Leftrightarrow r = \sqrt[q]{s} \\
\phi q &= \theta + 2\pi k, \quad k \in \mathbb{Z}
\end{aligned}$$

$$\boxed{\phi = \frac{\theta}{q} + \frac{2\pi k}{q}} \quad (7)$$

ϕ_0 bis ϕ_{q-1} von $k = 0$ bis $q - 1$

Vorbesprechung

2a)

$$A = \{z \in \mathbb{C} \mid 0 < \operatorname{Re}(i\bar{z}) < 1\} \quad (8)$$

$$\operatorname{Re}(i\bar{z}) = \operatorname{Re}(y + ix) = y \Rightarrow 0 < \operatorname{Re}(i\bar{z}) < 1 \Leftrightarrow 0 < y < 1 \quad (9)$$

3a

$$z^3 + 1 = 0, \quad z \in \mathbb{C} \quad (10)$$

$$z^3 = -1 \quad (11)$$

$$r^3 e^{i\phi 3} = 1 e^{i\pi} \quad (12)$$

$$\Rightarrow r = 1 \quad (13)$$

$$3\phi = \frac{\pi}{3} + \frac{2\pi k}{3} \quad k = 0, 1, 2 \quad (14)$$

Questions for TA: What is the purpose of definition 18.

Serie 1

1a)

$$\begin{aligned}
&4 + 6i \\
&2 + 6i - i + 3 = 5 + 5i \\
&(2 + 8i - i + 4)i = 2i - 8 + 1 + 4i = -7 + 6i \\
&\frac{(1+4i)(2+i)}{(2-i)(2+i)} = \frac{2+i+9i-4}{4+1} = -\frac{2}{5} + 2i \\
&\frac{(2-i)(1-3i)}{(1+3i)(1-3i)} = \frac{2-6i-i-3}{1+9} = -\frac{1}{10} - \frac{7}{10}i \\
&2 + i
\end{aligned}$$

1b)

$$2e^{i0} = 2$$
$$1e^{i\frac{3}{2}\pi}$$
$$\sqrt{9+27} + e^{i \arctan \frac{-3}{-3\sqrt{3}}} = 6 + e^{i\frac{11\pi}{6}}$$

1c)

$$(2in225^\circ) = -\sqrt{2} - \sqrt{2}i$$
$$(4in120^\circ) = -4\tan 30^\circ + 4\cos 30^\circ = -\frac{4\sqrt{3}\pi}{3} + 2\sqrt{3}\pi i$$

5

```
1 function fib=fibonacci(n)
2 %% Berechnet die Fibonacci-Zahlen.
3 % Input: Natuerliche Zahl n.
4 % Output: Fibonacci-Zahlen f_0, ..., f_n.
5
6 % Initialisiere Null-Vektor, der die Fibonacci-Zahlen
7 % enthalten soll.
8 fib=zeros(1,n+1);
9
10 % Anfangswerte
11
12 %%%%%% hier der Loesungsvorschlag %%%%%%
13 fib(1,2) = 1;
14 for i = 3:n+1
15     fib(1,i) = fib(1,i-1) + fib(1,i-2);
16 endfor
17 %%%%%% hier der Loesungsvorschlag %%%%%%
18
19 return
```

The following was presented in the exercise on 04. October 2018.

2 Matrizen

$A \in R^{m \times n}$ ist eine $m \times n$ Matrix

Sie besitzt m n Skalare $a_{ij} \in \mathbb{R}/\mathbb{C}$ mit m Zeilen und n Spalten.
 $a_{ij} := i - te$ Zeile, $j - te$ Spalte

2.1 Menge der Matrizen

$$\mathbb{R}^{m \times n} = \{M = (m_j)_{1 \leq i \leq m \text{ und } 1 \leq j \leq n} | m_{ij} \in \mathbb{R}\}$$
$$\mathbb{C}^{m \times n} = \{M = (m_j)_{1 \leq i \leq m \text{ und } 1 \leq j \leq n} | m_{ij} \in \mathbb{C}\}$$

2.2 Spezielle Matrizen

1. $A \in \mathbb{R}^{n \times n}$ nennt man eine quadratische Matrix
2. $A \in \mathbb{R}^{m \times 1}$ nennt man ein Spaltenvektor
3. $A \in \mathbb{R}^{1 \times n}$ nennt man ein Zeilenvektor ($a_{1j} = a_j$)
4. $0 \in \mathbb{R}^{m \times n}$ nennt man eine Nullmatrix, falls $0_{ij} = 0, 1 \leq i \leq m, 1 \leq j \leq n$
5. $D \in \mathbb{R}^{n \times n}$ nennt man eine Diagonalmatrix, falls $d_{ij} = 0$ für $i \neq j$
Für Diagonalelemente d_{11}, d_{22}, d_{nn} $D := \text{diag}(d_{11}, \dots, d_m)$
6. Die Diagonalmatrix $I_n = \text{diag}(1, \dots, 1)$ heisst Einheits-/Identitätsmatrix
7. $R \in \mathbb{R}^{n \times n}$ nennt man obere Dreieckmatrix / Rechtsmatrix, falls $r_{ij} = 0$ für $i > j$
8. $L \in \mathbb{R}^{n \times n}$ nennt man untere Dreieckmatrix / Linksmatrix, falls $l_{ij} = 0$ für $i < j$

2.3 Matrixoperationen

- Matrix-Matrix Gleichheit: $A, B \in \mathbb{R}^{n \times n} : A = B \Leftrightarrow a_{ij} = b_{ij}, 1 \leq i \leq m$ und $1 \leq j \leq n$
- Matrix-Skalar Multiplikation: $(\alpha M)_{ij} = \alpha M_{ij}, \alpha \in \mathbb{R}, M \in \mathbb{R}^{m \times n}$
- Matrix-Matrix Addition: $A, B \in \mathbb{R}^{m \times n} \quad (A + B)_{ij} = A_{ij} + B_{ij}$
- Matrix-Matrix Multiplikation: $A \in \mathbb{R}^{m \times n}, \quad B \in \mathbb{R}^{n \times n}, \quad (AB)_{ij} := \sum_{k=1}^n (A)_{ik} (B)_{kj}$
- Transponierte Matrix $(A^T)_{ij} = A_{ji}$
Wenn gilt $A = A^T$ dann gilt A ist symmetrisch
- Hermistesch-transponierte Matrix $(A^H)_{ij} = \overline{A}_{ji}$
 $(A^H)^H = A$
Falls $A^H = A$, nennt man A hermitesch

2.4 Rechenregeln

- Matrix-Matrix Addition:
 1. Kommutativ $A + B = B + A$
 2. Assotativ $(A + B) + C = A + (B + C)$
 3. Matrixmultiplikation ist Distributiv über Addition: $A(B+C) = (AB+AC)$
- Matrix-Matrix Multiplikation:
 1. Assotativ $(AB)C = A(BC)$
 2. Nicht kommutativ $AB \neq BA$

2.5 Linearkombination

Eine Linearkombination (LK) der Vektoren $\vec{a}_1, \dots, \vec{a}_n$ ist ein Ausdruck der Form $\alpha_1 \vec{a}_1, \dots, \alpha_n \vec{a}_n$ $\alpha \in \mathbb{R}$, Koeffizienten

Theorem 2.3

Gegeben die Kolonnen $\vec{a}_1, \dots, \vec{a}_n$ von $A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n$
 $\Rightarrow Ax = \vec{a}_1 x_1, \dots, \vec{a}_n x_n = \sum_{i=1}^n \vec{a}_i x_i$

Theorem 2.4

$$A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times p}, B = \begin{pmatrix} | & \cdots & | \\ \vec{b}_1 & \cdots & \vec{b}_p \\ | & \cdots & | \end{pmatrix} \text{ so gilt:}$$
$$AB = \begin{pmatrix} | & \cdots & | \\ A\vec{b}_1 & \cdots & A\vec{b}_p \\ | & \cdots & | \end{pmatrix} \quad (15)$$

Theorem 2.5

$$Ax = b \text{ hat eine Lösung} \Leftrightarrow b \text{ eine LK der Kolonnen von } A \text{ ist} \quad (16)$$

Vorbesprechung Serie 2

Bonusaufgabe) Sei G eine Menge. G bildet mit der Operation $*$ eine abelsche Gruppe falls:

1. $\exists e \in G \forall A \in G : A * e = e * A = A$ (neutrales Element)
2. $\forall A \in G \exists A^{-1} \in G : A * A^{-1} = A^{-1} * A = e$ (inverses Element)
3. $\forall A, B, C \in G : (A * B) * C = A * (B * C)$ (assotativ)
4. $\forall A, B, C \in G : (A * B) = (B * A)$ (kommutativ)

Tipp: Schreibt die Operationen explizit aus

- 4) a) Def A^H anwenden
- b) $A^H = A$
- c) α, β, γ finden, $sD = D^H$
- 6) Nicht vergessen Nullteiler
 $AB = 0, AB \neq 0$

The following was presented in the exercise on 11. October 2018.

Anhang für letztes Mal

$$(AB)^T = B^T A^T \quad (17)$$

A, B symmetrisch und AB symmetrisch $\Leftrightarrow AB = BA$
 $\Leftrightarrow AB = (AB)^T = B^T A^T = BA$

2.6 Skalarprodukt

$$\begin{aligned} <\cdot, \cdot> : \mathbb{E}^n \times \mathbb{E}^n \rightarrow \mathbb{E} \quad |\mathbb{E} = \mathbb{R}/\mathbb{C} \\ (x, y) \mapsto <x, y> = x^H y = \sum_{k=1}^n \bar{x}_k y_k = \sum x_k y_k, \quad \text{für } x, y \in \mathbb{R} \end{aligned}$$

2.7 Eigenschaften

$$\forall x, y, z \in \mathbb{E}^n, \alpha \in \mathbb{E}$$

$$1) <x, y+z> = <x, y> + <x, z> \text{ (Linear im zweiten Faktor)}$$

$$<x, \alpha z> = \alpha <x, z>$$

$$2) \mathbb{E} = \mathbb{R} : <x, y> = <y, x> \text{ (symmetrisch)}$$

$$\mathbb{E} = \mathbb{C} : <x, y> = \overline{<y, x>} \text{ (hermitisch)}$$

$$3) <x, x> \geq 0, <x, x> = 0 \Rightarrow x = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \text{ (positiv definit)}$$

$$4) \mathbb{E} = \mathbb{R} < w+x, y> = <w, y> + <x, y> \text{ (Linear im ersten Faktor)}$$

$$<\alpha x, y> = \alpha <x, y> \text{ (Linear im ersten Faktor)}$$

$$4) \mathbb{E} = \mathbb{C} < w+x, y> = <w, y> + <x, y> \text{ (Linear im ersten Faktor)}$$

$$<\alpha x, y> = \overline{\alpha} <x, y> \text{ (Sesquilinear)}$$

2.8 Euklidische Norm $x \in \mathbb{E}^n$

$$\|x\| = \sqrt{<x, x>} = \sqrt{x^H x} = \sqrt{\sum_{k=1}^n \bar{x}_k x_k} = \sqrt{\sum_{k=1}^n \|x_k\|^n}, \quad x = (1, 2, 3) \Rightarrow \|x\| = \sqrt{1^2 + 2^2 + 4 + 2} = \sqrt{14}$$

2.9 Eigenschaften $\forall x, y \in \mathbb{R}^n, \forall \alpha \in \mathbb{R}$

Satz 2.11 (Cauchy-Schwarz Ungleichung)

$$1) |<x, y>| \leq \|x\| \|y\|$$

$$2) \|x\| \geq 0, \text{ dann muss gelten } x = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$3) \|\alpha x\| = |\alpha| \|x\| \text{ (Betrag nach homogen)} \quad 4) \|x+y\| \leq \|x\| + \|y\|$$

2.10 Winkel zwischen zwei Vektoren

$$\begin{aligned} x, y \in \mathbb{E}^n \quad \cos \theta = \frac{\operatorname{Re} <x, y>}{\|x\| \|y\|} \quad (\mathbb{E} = \mathbb{R} : \cos \theta = \frac{<x, y>}{\|x\| \|y\|}) \\ \Rightarrow x \perp y \Rightarrow p = \frac{\pi}{2} \Rightarrow <x, y> = 0 \end{aligned}$$

3 Lineare Gleichungssysteme

m Gleichungen und n Unbekannte

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix} \Leftrightarrow Ax = b \quad (18)$$

$m = n$: 1 Lösung oder Unendlich Viele Lösungen.

Unterbestimmt: $m < n$: Unendlich viele Lösungen oder keine Lösung aber nie nur eine Lösung.
Überbestimmt: $m > n$: Keine Lösung oder unendlich viele Lösungen.

Definition 8. Lösungsmenge $Solution(A : b) := \{x \in \mathbb{R}^n | Ax = b\}$

3.1 Wie findet man x?

Idee: Forme $Ax = b$ zu $Rx = L$ um, so dass $Solution(A, b) = Solution(R, L)$ aber Lösung x ist einfacher zu finden.

Vorbesprechung von Serie 3

- 1a) Gauss anwenden
- 1b) Gauss anwenden -Achtung unterbestimmt
- 1c) $Ax = 0 \Rightarrow x = 0$ ist triviale Lösung
finde a , sodass RL unendlich viele Lösungen hat.
2. 1) Versucht Norm als Skalarprodukt umzuschreiben. $0 \leq \|\cdot\| \Leftrightarrow 0 \leq \langle \cdot, \cdot \rangle$
- 2) Ungleichung finden $x \leq / \geq b^2 - 4ac$
- 3) Ungleichung Umfang sodass (5) rauskommt

3.2 Serie 3 - Cauchy-Schwarz Ungleichung (10/10)

Proof 3.1: Beweis Cauchy-Schwarz Ungleichung

Folgender Satz ist zu zeigen:

$$|\langle x, y \rangle| \leq \|x\| \|y\| \quad (19)$$

Schritt 1 - Der Betrag eines Vektor ist immer positiv:

$$0 \leq \|tx + y\| \quad | \text{ Definition Norm} \quad (20)$$

$$0 \leq \sqrt{\langle tx + y, tx + y \rangle} \quad | \text{ quadrieren} \quad (21)$$

$$0 \leq \langle tx + y, tx + y \rangle \quad | \text{ Definition Skalarprodukt} \quad (22)$$

$$0 \leq |tx + y| \cdot |tx + y| \cos 0 \quad | \text{ ausmultiplizieren} \quad (23)$$

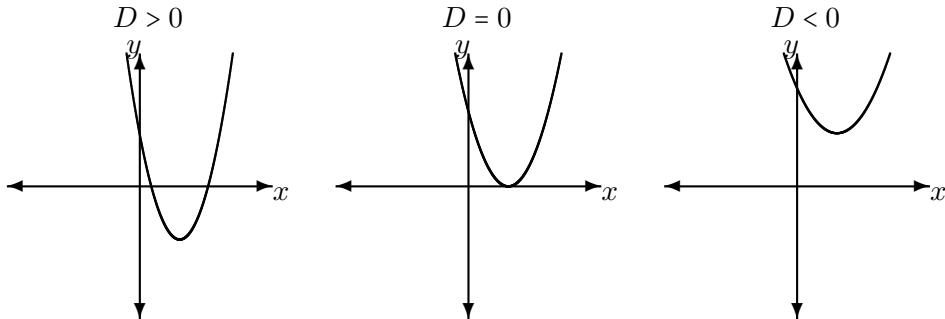
$$0 \leq t^2 \langle x, x \rangle + 2t \langle x, y \rangle + \langle y, y \rangle \quad | \text{ Definition Norm} \quad (24)$$

$$0 \leq t^2 \|x\|^2 + 2t \langle x, y \rangle + \|y\|^2 \quad (25)$$

Schritt 2 - Eigenschaften der Diskriminanten herausfinden:

$$D = (2\langle x, y \rangle)^2 - 4\|x\|^2\|y\|^2 \quad (26)$$

Wie wir gesehen haben, muss die Parabel immer Positiv sein. Welche Rückschlüsse kann man daraus auf die Diskriminante ziehen?



Wenn die Diskriminante Positiv wäre, sieht man das die Parabel auch ins Negative geht. Dies kann aber nicht der Fall sein, wie wir bei der Gleichung von Oben gesehen haben. Deshalb muss die Diskriminante Negativ oder Null sein.

$$\Rightarrow (2xy)^2 - 4\|x\|^2\|y\|^2 \leq 0 \quad (27)$$

Schritt 3 - Nun muss man nur noch diese Gleichung auflösen:

$$(2xy)^2 \leq 4\|x\|^2\|y\|^2 \quad (28)$$

$$(xy)^2 \leq \|x\|^2\|y\|^2 \quad | \text{ Wurzel ziehen} \quad (29)$$

Man muss den Betrag beachten, denn es gilt nur wenn der Term positiv ist.

$$|x \cdot y| \leq \|x\|\|y\| \quad | \text{ Definition vom Skalarprodukt} \quad (30)$$

$$\boxed{|\langle x, y \rangle| \leq \|x\|\|y\|} \quad (31)$$

□

3.3 Gauss Algorithmus

```

1 function solution = gauss(A, b)
2     pivot_x = 1;
3     pivot_y = 1;
4     n = length(A);
5     solution = zeros(1,n);
6     for i=1:n;
7         while A(:,1) == zeros(n,1);
8             pivot_x+=1;
9             if pivot_x>n;
10                 solution=false;
11                 return;
12             endif
13         endwhile
14         while A(pivot_y,pivot_x) == 0;
15             temp = A(1,:);

```

```

16     A(1,:) = A(2,:);
17     A(2,:) = A(3,:);
18     A(3,:) = A(4,:);
19     A(4,:) = temp;
20     temp = b(1);
21     b(1) = b(2);
22     b(2) = b(3);
23     b(3) = b(4);
24     b(4) = temp;
25 endwhile
26 for j=pivot_y+1:n;
27     multiply= A(j,pivot_x)/A(pivot_y,pivot_x);
28     A(j,:)= A(j,:)-multiply*A(pivot_y,:);
29     b(j)= b(j)-multiply*b(pivot_y);
30 endfor
31 pivot_x+=1;
32 pivot_y+=1;
33 endfor
34 for i=1:n;
35     k=(5-i);
36     for j=k:4;
37         b(k)-=A(k,j)*solution(j);
38     endfor
39     if A(k,k)==0;
40         solution=false;
41         return;
42     else
43         solution(k)=b(k)/A(k,k);
44     endif
45 endfor
46 return;
47 endfunction

```

The following was presented in the exercise on 18.10.18.

3.4 Gauss Elimination & LR-Zerlegung

3.4.1 Permutationsmatrix

$$P = \{0, 1\}^{n \times n} \quad \forall i, j \in \mathbb{N} \setminus \{0\} : \sum_{k=1}^n P_{ik} = \sum_{k=1}^n P_{kj} = 1 \quad (32)$$

3.4.2 Rang

Anzahl der Pivotelemente der Matrix $A \in \mathbb{R}^{m \times n}$

$$c \leq \min\{m, n\}$$

$n - r$ = Anzahl der freien Parameter

3.4.3 Verträglichkeitsbedingungen für $m > r$

$$c_{r+1} = c_{r+2} = \dots = c_m = 0 \quad (33)$$

Falls $c_k \neq 0$ für $k > r \Rightarrow$ System hat keine Lösung

Theorem 3.1: (1.1)

System hat Lösung wenn $r = m$ oder ($r < m$ und $c_{r+1} = c_{r+2} = \dots = c_m = 0$)

Gibt es Lösungen, dann gilt $n = r \Rightarrow$ Genau eine Lösung

$r < n \Rightarrow n - r$ freie Parameter

3.4.4 Homogenes LGS, also $Ax = 0$

$x=0$ ist die triviale Lösung

Weil $b=0$, Verträglichkeitsbedingungen immer erfüllt

Falls $n > m \Rightarrow$ besitzt nicht triaviale Lösung

Falls $A \in \mathbb{R}^{n \times n}$ regulär $\Rightarrow A$ hat nur triviale Lösung ($m=n=r$)

3.5 Inverse einer Matrix

$$A \in \mathbb{R}^{n \times n} \text{ invertierbar} \Leftrightarrow \exists X \in \mathbb{R}^{n \times n} : AX = I_n = XA \quad (A \neq 0 \text{ und } X = A^{-1}) \quad (34)$$

Falls A^{-1} existiert \Rightarrow eindeutig bestimmt

Theorem 3.17

Folgende Aussagen für $A \in \mathbb{R}^{n \times n}$ Äquivalent (\Leftrightarrow)

1. A invertierbar
2. $\exists A^{-1} \in \mathbb{R}^{n \times n} \quad AA^{-1} = I_n$
3. Falls $AX = AY = I_n \Rightarrow x = y$
4. A ist regulär ($n = r$)

3.5.1 Inverse einer Matrix berechnen

$$A_{m \times n} \text{ invertierbar} \Leftrightarrow \exists X \in \mathbb{R}^{n \times n} : AX = I \quad (35)$$

$$AX = \begin{pmatrix} | & \cdots & | \\ Ax_1 & \cdots & Ax_n \\ | & \cdots & | \end{pmatrix} = \begin{pmatrix} | & \cdots & | \\ \bar{e}_1 & \cdots & \bar{e}_1 \\ | & \cdots & | \end{pmatrix} = I \quad (36)$$

$\Rightarrow A\vec{x} = \vec{e}$, eindeutig lösbar (A regulär)

\vec{x} , auflösen

3.6 Serie 4 (10/10)

4a)

$$A^2 = AB + 2A|A^{-1} \quad (37)$$

$$A = B + 2I \quad (38)$$

$$\left(\begin{array}{ccc|ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 & 1 & 1 & -1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 2 & 0 & 0 & 1 & 1 & 0 & 1 \end{array} \right) \Rightarrow \left(\begin{array}{ccc|ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 & 0 & -2 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & -1 & -1 & 1 & 0 & -1 & 1 \end{array} \right) \quad (39)$$

$$\left(\begin{array}{ccc|ccc|ccc} 1 & 1 & -1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 1 \end{array} \right) \Rightarrow \left(\begin{array}{ccc|ccc|ccc} 0 & -2 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & 1 & 0 & -1 & 1 \end{array} \right) \quad (40)$$

$$\Rightarrow A^{-1} = \begin{pmatrix} 0 & -2 & 1 \\ 0 & 1 & 0 \\ -1 & -1 & 1 \end{pmatrix} \quad (41)$$

4b i)

1. **Lemma 3.7** besagt, dass eine reell symmetrische Matrix die positiv definit ist, regulär ist.
Beide Bedingungen sind erfüllt, so kann gefolgt werden, dass A regulär ist.
2. **Satz 2.17** Dieser Satz besagt: A ist invertierbar \Leftrightarrow A ist regulär.
Daraus kann gefolgt werden, weil A regulär ist, muss A auch invertierbar sein.
3. **Satz 2.17** Daraus folgt ebenfalls, dass der Rang von $A = n$ ist, weil A regulär ist.

4b i) - Korrektur

$A \in \mathbb{R}^{n \times n}, x^T Ax > 0 \forall x \in \mathbb{R}^n \setminus \{0\}$
 $x^T A > 0 \Rightarrow x^T Ax \neq 0 \Rightarrow Ax \neq 0 \Rightarrow \forall x \in \mathbb{R}^n Ax = 0$ hat nur die triviale Lösung
 $\Rightarrow A$ vollen Rang $\Leftrightarrow A$ regulär $\Leftrightarrow A$ ist invertierbar.

ii)

1. **Definition Symmetrisch** Eine Matrix A heisst symmetrisch, falls $A^T = A$
2. **Satz 2.18 iii)** besagt für reguläre quadratische Matrizen gilt: $(A^T)^{-1} = (A^{-1})^T$
3. Wenn wir nun die Gleichung von 1) in 2) einsetzen: $A^{-1} = (A^{-1})^T$
4. **Definition Symmetrisch** Daraus folgt, dass auch A^{-1} symmetrisch ist.

4b ii) - Korrektur

$$\begin{aligned} A^{-1}A &= I & |^T \\ (A^{-1}A)^T &= I \\ A^T(A^{-1})^T &= I \\ A(A^{-1})^T &= I \\ (A^{-1})^T &= A^{-1}I \\ (A^{-1})^T &= A^{-1} \end{aligned}$$

The following was presented in the exercise on 25. October 2018.

3.7 Unitäre / Orthogonale Matrizen

$A \in \mathbb{E}^{m \times n}$ ist unitär, falls $A^H A = I_n$ falls $\mathbb{E} = \mathbb{R}$

dann ist A auch orthogonal. $A^T A = I_n$

Eigenschaften: $A, B \in \mathbb{E}^{m \times n}$ unitär

1. A ist regulär und $A^{-1} = A^H$
2. $AA^H = I_n$
3. A^{-1} ist unitär
4. AB ist unitär

A orthogonal $A^T A = I$

$$\Rightarrow a_i^T a_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (42)$$

Theorem 3.21

$A \in \mathbb{E}^{m \times n}$ unitär

$$1) \|Ax\| = \|x\| \quad 2) \langle Ax, Ay \rangle = \langle x, y \rangle \quad (43)$$

3.8 LR-Zerlegung

Idee: $A = LR$, L: Linksdreiecksmatrix und R: Rechtsdreiecksmatrix
 $A \in \mathbb{E}^{n \times n}$

$$PA = LR \quad (44)$$

3.8.1 Warum braucht es eine LR-Zerlegung

$$Ax = b \stackrel{\text{A=LR}}{\iff} LRx = b \Rightarrow 1) Lc = b, \quad 2) Rx = c \quad (45)$$

Um so grösser, dass das b wird, desto effizienter ist die LR-Zerlegung gegenüber dem Gauss Algorithmus.

3.8.2 LR-Zerlegung mit Permutation

$$PA = LR \quad Ax = b \Rightarrow PAx = Pb = LRx = Pb = \tilde{b} \quad (46)$$

$$1) Lc = \tilde{b} \quad 2) Rx = c \quad (47)$$

4 Vektorräume

Vektorräume können Vektoren sein, müssen aber nicht.

Ein Vektorraum V über \mathbb{E} ist eine nichtleere Menge, auf der eine Addition (innere Operation)

$$+: V \times V \rightarrow V$$

$$(x, y) \mapsto x + y$$

und eine skalar Multiplikation (äussere Operation)

$$\cdot: \mathbb{E} \times V \rightarrow V$$

$$(\alpha x) \rightarrow \alpha x$$

definiert sind.

4.1 Vektorraum Axiome $(V, +, \cdot)$

Der Vektorraum ist definiert in \mathbb{E} (also \mathbb{R} oder \mathbb{C})

$$(V1) \quad x + y = y + x, \quad \forall x, y \in V \text{ (Kommutativ)}$$

$$(V2) \quad (x + y) + z = x + (y + z) \text{ (Assotativ)}$$

$$(V3) \quad \exists 0 \in V \text{ so dass } \forall x \in V : x + 0 = x \text{ (Neutrales Element Addition)} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} (V, +) \text{ ist eine Abelsche Gruppe}$$

$$(V4) \quad \forall x \in V, \exists ! \text{ (eindeutig bestimmtes) } -x \in V \text{ so dass } x + (-x) = 0 \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} \text{ (Inverse Element)}$$

$$(V5) \quad \alpha(x + y) = \alpha x + \alpha y \quad \forall \alpha \in \mathbb{E}, \forall x, y \in V \text{ (Distributiv über +)}$$

$$(V6) \quad (\alpha + \beta)x = \alpha x + \beta x \quad \forall \alpha, \beta \in \mathbb{E}, \forall x \in V \text{ (Distributiv über Skalare)}$$

$$(V7) \quad (\alpha\beta)x = \alpha(\beta x), \quad \forall \alpha, \beta \in \mathbb{E}, \forall x \in V \text{ (Kommutativ über Skalare)}$$

$$(V8) \quad 1, x = x \quad \forall x \in V \text{ (Neutrales Element Multiplikation)}$$

\Rightarrow Auf diese Axiome gestützt bildet man die Theorie der Vektorräume auf.

Example 1. $\mathbb{E}^n, \mathbb{E}^{m \times n}, C[a, b]$: Raum der stetigen reellen Funktionen (KEIN VEKTOR!, aber im Vektorraum)

Innere Operation:

$$f, g \in C[a, b] : h(x) = f(x) + g(x) \quad (48)$$

$$f : [a, b] \rightarrow \mathbb{R} \quad a, b \in \mathbb{R} \quad (49)$$

Äussere Operation:

$$\alpha \in \mathbb{R}, f \in C[a, b] : \mathbb{R}x C[a, b] \rightarrow C[a, b] \quad (50)$$

$$(\alpha, f) \rightarrow \alpha f \quad \alpha f(x) \quad (51)$$

4.2 Unterraum

$U \subseteq V$ (V ist Vektorraum) und $U \neq \emptyset$

$$x + y \in U, \quad \alpha x \in U \quad (\forall x, y \in U, \forall \alpha \in \mathbb{E}) \quad (52)$$

$\Rightarrow U$ ist auch ein Vektorraum

4.3 Serie 5 (10/10)

a)

Zuerst muss man die Funktion evaluieren.

$$f(x) = f(1-x) \quad (53)$$

Diese Funktion muss sich an der Gerade $x = \frac{1}{2}$ spiegeln. Deshalb kommen nur Polynome geraden Grades in Frage. Daraus folgt:

$$f(x) = \lim_{n \rightarrow \infty} \sum_{k=0}^n a_k \cdot \left(x - \frac{1}{2}\right)^{2k} \quad (54)$$

Nun können wir prüfen, ob es ein Unterraum ist:

Abgeschlossene Addition

$$f(x) + g(x) = \lim_{n \rightarrow \infty} \sum_{k=0}^n a_k \cdot \left(x - \frac{1}{2}\right)^{2k} + \lim_{n \rightarrow \infty} \sum_{k=0}^n b_k \cdot \left(x - \frac{1}{2}\right)^{2k} = \lim_{n \rightarrow \infty} \sum_{k=0}^n (a_k + b_k) \cdot \left(x - \frac{1}{2}\right)^{2k} \quad (55)$$

Somit gilt $f(x) + g(x) \in W \forall f, g \in W$

Abgeschlossene Skalarmultiplikation

$$c \cdot f(x) = \lim_{n \rightarrow \infty} \sum_{k=0}^n a_k \cdot \left(x - \frac{1}{2}\right)^{2k} = \lim_{n \rightarrow \infty} \sum_{k=0}^n (c \cdot a_k) \cdot \left(x - \frac{1}{2}\right)^{2k} \quad (56)$$

Somit gilt $c \cdot f(x) \in W \forall f \in W \forall c \in \mathbb{R}$

Nullelement und W nicht leer

$$f(x) = \lim_{n \rightarrow \infty} \sum_{k=0}^n 0 \cdot \left(x - \frac{1}{2}\right)^{2k} = 0 \quad (57)$$

Somit gilt $0 \in W$ und W ist nicht leer

Da alle drei Eigenschaften erfüllt sind, ist W ein Unterraum von V .

b)

Die Vektoren lassen sich in folgende Matrix übersetzen:

$$\begin{pmatrix} x_1 & y_1 & z_1 & w_1 \\ x_2 & y_2 & z_2 & w_2 \\ x_3 & y_3 & z_3 & w_3 \\ x_4 & y_4 & z_4 & w_4 \end{pmatrix} \quad (58)$$

Deshalb muss dann folgende Gleichung immer erfüllt sein. Sie ist auch immer erfüllt, denn in jeder Zeile können immer drei Variablen frei gewählt werden, sodass noch eine Lösung existiert.

$$\begin{pmatrix} x_1 & y_1 & z_1 & w_1 \\ x_2 & y_2 & z_2 & w_2 \\ x_3 & y_3 & z_3 & w_3 \\ x_4 & y_4 & z_4 & w_4 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (59)$$

Nun können wir prüfen, ob es ein Unterraum ist:

Abgeschlossene Addition

$$\begin{aligned} [A + A']x &= \left[\begin{pmatrix} x_1 & y_1 & z_1 & w_1 \\ x_2 & y_2 & z_2 & w_2 \\ x_3 & y_3 & z_3 & w_3 \\ x_4 & y_4 & z_4 & w_4 \end{pmatrix} + \begin{pmatrix} x'_1 & y'_1 & z'_1 & w'_1 \\ x'_2 & y'_2 & z'_2 & w'_2 \\ x'_3 & y'_3 & z'_3 & w'_3 \\ x'_4 & y'_4 & z'_4 & w'_4 \end{pmatrix} \right] \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} \\ &= \begin{pmatrix} x_1 + x'_1 & y_1 + y'_1 & z_1 + z'_1 & w_1 + w'_1 \\ x_2 + x'_2 & y_2 + y'_2 & z_2 + z'_2 & w_2 + w'_2 \\ x_3 + x'_3 & y_3 + y'_3 & z_3 + z'_3 & w_3 + w'_3 \\ x_4 + x'_4 & y_4 + y'_4 & z_4 + z'_4 & w_4 + w'_4 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} x^*_1 & y^*_1 & z^*_1 & w^*_1 \\ x^*_2 & y^*_2 & z^*_2 & w^*_2 \\ x^*_3 & y^*_3 & z^*_3 & w^*_3 \\ x^*_4 & y^*_4 & z^*_4 & w^*_4 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \end{aligned} \quad (60)$$

Somit gilt $A + A' \in S \forall A, A' \in S$

Abgeschlossene Skalarmultiplikation

$$\begin{aligned} c \cdot Ax &= c \cdot \begin{pmatrix} x_1 & y_1 & z_1 & w_1 \\ x_2 & y_2 & z_2 & w_2 \\ x_3 & y_3 & z_3 & w_3 \\ x_4 & y_4 & z_4 & w_4 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} c \cdot x_1 & c \cdot y_1 & c \cdot z_1 & c \cdot w_1 \\ c \cdot x_2 & c \cdot y_2 & c \cdot z_2 & c \cdot w_2 \\ c \cdot x_3 & c \cdot y_3 & c \cdot z_3 & c \cdot w_3 \\ c \cdot x_4 & c \cdot y_4 & c \cdot z_4 & c \cdot w_4 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} \\ &= \begin{pmatrix} x^*_1 & y^*_1 & z^*_1 & w^*_1 \\ x^*_2 & y^*_2 & z^*_2 & w^*_2 \\ x^*_3 & y^*_3 & z^*_3 & w^*_3 \\ x^*_4 & y^*_4 & z^*_4 & w^*_4 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \end{aligned} \quad (61)$$

Somit gilt $c \cdot A \in S \forall A \in S \forall c \in \mathbb{R}$

Nullelement und S nicht leer

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2 \\ 4 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (62)$$

Somit gilt es gibt ein Nullelement und S ist nicht leer.

Da alle drei Eigenschaften erfüllt sind, ist S ein Unterraum von \mathbb{R}^4 .

LR Matlab Implementation

```
1 function [L,R,P] = lr(A)
2     [a,b] = size(A);
3     P = eye(a,b);
4     L = eye(a,b);
5     R = A;
6     pivot=1;
7
8     for i=1:a
9         [A,P,L,R]=sortAbs(A,P,L,R,i,pivot);
10        while A(i,pivot)==0 && pivot<=b
11            pivot+=1;
12        endwhile
13
14        if pivot>b
15            break
16        endif
17
18        for j=i+1:a
19            L(j,pivot)=R(j,pivot)/R(i,pivot);
20            R(j,pivot:b)=-L(j,pivot)*R(i,pivot:b);
21        endfor
22        pivot+=1;
23    endfor
24
25    return
26 end
27
28 function [A,P,L,R] = sortAbs(A,P,L,R,i,p)
29     [a,b] = size(A);
30     oldA=A;
31     oldP=P;
32     oldL=L-eye(a,b);
33     oldR=R;
34     otherL=L;
35     A=zeros(a,b);
36     R=zeros(a,b);
37     L=zeros(a,b);
38     P=zeros(a,b);
39     for k=1:i-1
40         A(k,:)=oldA(1,:);
41         P(k,:)=oldP(1,:);
42         L(k,:)=otherL(1,:);
43         R(k,:)=oldR(1,:);
44         oldA(1,:) = [];
45         oldP(1,:) = [];
46         otherL(1,:) = [];
47         oldL(1,:) = [];
48         oldR(1,:) = [];
49     endfor
50     for k=i:a-1
51         [maxVal maxInd] = max(abs(oldR));
52         A(k,:)=oldA(maxInd(p),:);
53         P(k,:)=oldP(maxInd(p),:);
54         L(k,:)=zeros(1,b);
55         L(k,1:k-1)=oldL(maxInd(p),1:k-1);
```

```

56     L(k,k)=1;
57     R(k,:)=oldR(maxInd(p),:);
58     oldA(maxInd(p),:) = [];
59     oldP(maxInd(p),:) = [];
60     oldL(maxInd(p),:) = [];
61     oldR(maxInd(p),:) = [];
62   end
63   A(a,:)=oldA(1,:);
64   P(a,:)=oldP(1,:);
65   L(a,:)=zeros(1,b);
66   L(a,1:a-1)=oldL(1,1:a-1);
67   L(a,a)=1;
68   R(a,:)=oldR(1,:);
69   return
70 endfunction

```

The following was presented in the exercise on 01. November 2018.

4.4 Kommutation Skalarprodukt

Für VR V über einem Körper K, muss das Skalarprodukt kommutieren?

Gegeben F: $M \times M \rightarrow M$

$(a_1, a_2) \rightarrow F(a_1, a_2)$

F kommutativ $\Leftrightarrow F(a_1, a_2) = F(a_2, a_1)$

Skalarprodukt: O: $K \times V \rightarrow V$

$(\alpha, r) \rightarrow \alpha r$

4.5 Linearkombination

Gegeben VRV über \mathbb{E}

$$a_i \in V; \zeta \in \mathbb{E}; x = \sum_{i=1}^n \zeta_i a_i \Rightarrow x \text{ ist eine LK} \quad (63)$$

von $\{a_1, a_2, \dots, a_n\} = S$

$$U := \{x : x = \sum_{i=1}^n \zeta_i a_i, \zeta \in \mathbb{E}, a \in S\} \quad \text{ist ein Unterraum von } V \quad (64)$$

U: Von $\{a_1, a_2, \dots, a_n\}$ aufgespannter / erzeugender Unterraum $U = \text{span } \{a_1, a_2, \dots, a_n\}$
 a_1, \dots, a_n heisst Erzeugendensystem von U.

Example 2.

$$A = \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_k \\ | & \cdots & | \end{pmatrix} \quad (65)$$

a_1, \dots, a_n ist Erzeugendensystem von \mathbb{R}^n

$$\Leftrightarrow \forall b \in \mathbb{R}^n \exists x \in \mathbb{R} : b = \underbrace{\sum_{i=1}^n x_i a_i}_{Ax}$$

$\Leftrightarrow Ax = b$ besitzt eine Lösung.

$\Leftrightarrow A = n$

Note 1. Erzeugendensysteme sind nicht eindeutig

4.6 Lineare Abhängigkeit (VRV)

$a_1 \in S \subseteq V, \gamma \in \mathbb{R} \quad a_1, \dots, a_n$ sind linear unabhängig

$$\Leftrightarrow \left(\sum_{i=1}^n \gamma_i a_i = 0 \Rightarrow \gamma_1 = \gamma_2 = \dots = \gamma_n = 0 \right) \quad (66)$$

Falls nicht: a_1, \dots, a_n linear abhängig

Lemma 4.5

$\gamma > 2$ Vektoren a_1, \dots, a_γ sind linear abhängig

\Leftrightarrow einer der Vektoren lässt sich als Linearkombination des Anderen schreiben.

Example 3.

$$A = \begin{pmatrix} | & \cdots & | \\ \bar{a}_1 & \cdots & \bar{a}_k \\ | & \cdots & | \end{pmatrix} \in \mathbb{R}^{n \times k} \quad a_1, \dots, a_k \text{ linear unabhängig} \quad (67)$$

$$\Leftrightarrow \sum_{i=1}^k x_i a_i = 0 \quad \text{hat nur die triviale Lösung } (x_1 = \dots = x_k = 0) \quad (68)$$

$\Leftrightarrow Ax = 0$ hat nur triviale Lösung \Leftrightarrow keine freien Parameter

$\Leftrightarrow \text{Rang } A = k$

4.7 Basis

Ein linear unabhängiges Erzeugendensystem B von einem Vektorraum V nennt man Basis.
Genau dann:

1. Erzeugend für V ($\text{span}\{b_1, b_2, \dots, b_n\} = V$)

2. b_1, \dots, b_n ist linear unabhängig

$(b_1, \dots, b_n) \in B$

Theorem 4.7

Besitzt ein Vektorraum mehrere Basen, so muss die Anzahl der Vektoren einer Basis immer gleich sein.

4.8 Dimension eines Vektorraumes

Anzahl der Basisvektoren

$\dim V$ erzeugt: $\dim V = n$

V ist n -dimensional

Theorem 4.9

Jede Menge von linear unabhängigen Vektoren lässt sich ergänzen zu einer Basis von V

Corollary 4.10

VRV, $\dim V = n$

Jede Menge von n linear unabhängigen Vektoren ist eine Basis für V

4.9 Koordinaten im Vektorraum

VRV, $\dim V = n$, Basis $B = \{b_1, b_2, \dots, b_n\}$

$$x \in V \Rightarrow x = \sum_{i=1}^n \alpha_i b_i \quad (69)$$

Die Koordinaten α heißen Koordinaten von x bezüglich Basis B und sind eindeutig bestimmt.

Serie 6

3a) 1. Bonus: Sie ist erzeugend (gegeben). Musst nur zeigen, dass es linear unabhängig ist.

$$\lambda_1 a_1 + \lambda_2 a_2 + \lambda_3 a_3 = 0$$

$$\Rightarrow \lambda_1 = \lambda_2 = \lambda_3 = 0$$

3a) 2. x bestimmen, $x = \sum_{i=1}^3 \lambda_i a_i$ (LGS)

3a) 3 $\dim V = 4 (|B| = 4), |A| = 3$, brauchen noch einen Vektor $v \notin \text{span} A$

3b) 1. Schreibe B in Koordinaten von Sum: $q_n = \sum_{i=1}^3 \lambda_i p_i$ argumentieren mit Hilfe dessen, warum S eine Basis ist.

3b) 2. Löse $p = \sum_{i=1}^3 \lambda_i p_i$ (LGS)

The following was presented in the exercise on 08. November 2018.

5 Lineare Abbildung

$$F : X \rightarrow Y \quad (70)$$

$$F(x + y) = F(x) + F(y) \quad \forall x, y \in X \quad (71)$$

$$F(\alpha x) = \alpha F(x) \quad (72)$$

surjektiv $F(X) = Y$

injektiv $F(x) = F(x') \Rightarrow x = x'$

bijektiv Beides

5.1 Bild

$$Bild(F) = F(x) = \{F(x) | x \in X\} \subseteq Y \quad (73)$$

$$A_F x = \sum \overline{\alpha_i} x_i = span\{\overline{\alpha_1}, \dots, \overline{\alpha_2}\} \quad (74)$$

5.2 Kern

$$Kern(F) = \{x \in X | Ax = 0\} \quad (75)$$

Example 4.

$$Ax = 0 \quad (76)$$

$$\begin{pmatrix} 1 & 4 & 2 & 1 \\ 0 & 0 & 5 & 1 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (77)$$

$$Kern(F) = span \left\{ \begin{pmatrix} -4 \\ 1 \\ 0 \\ 0 \end{pmatrix} \right\} \quad (78)$$

$$\dim(Kern(F)) = 1 \quad (79)$$

$$Bild(F) = span\{ \text{lineare unabhängigen Kolonnen von } A \} \quad (80)$$

$$\dim(Bild(F)) = 3 \quad (81)$$

5.3 Dimensionsformel

$$\dim(Kern(F)) + \dim(Bild(F)) = \dim X = Rang(A_F) \quad (82)$$

5.4 Basiswechsel

Von der Basis A nach Standardbasis

$$p = A \cdot v \quad (83)$$

Von der Basis B nach Standardbasis

$$p = B \cdot w \quad (84)$$

$$\Rightarrow Av = Bw \quad (85)$$

$$\underbrace{B^{-1}A}_{T} v = w \quad (86)$$

The following was presented in the exercise on 15. November 2018.

5.5 Serie 7 (9/10)

4a)

Zu zeigen:

$$f(a+b) = f(a) + f(b) \quad f(\alpha a) = \alpha f(a) \quad (87)$$

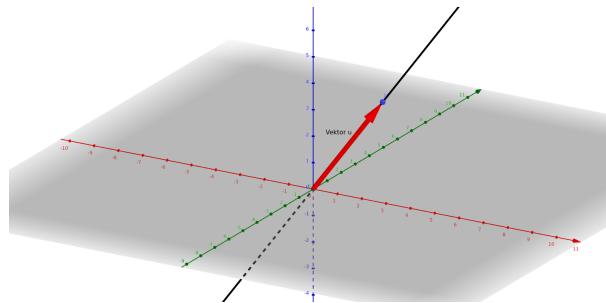
$$P(a+b) = \left(\frac{\langle u, a+b \rangle}{\langle u, u \rangle} \right) u \underset{2.9}{=} \left(\frac{\langle u, a \rangle + \langle u, b \rangle}{\langle u, u \rangle} \right) u = \left(\frac{\langle u, a \rangle}{\langle u, u \rangle} \right) u + \left(\frac{\langle u, b \rangle}{\langle u, u \rangle} \right) u = P(a) + P(b) \quad (88)$$

$$P(\alpha b) = \left(\frac{\langle u, \alpha b \rangle}{\langle u, u \rangle} \right) u \underset{2.9}{=} \left(\frac{\alpha \langle u, b \rangle}{\langle u, u \rangle} \right) u = \alpha \left(\frac{\langle u, b \rangle}{\langle u, u \rangle} \right) u = \alpha P(b) \quad (89)$$

4b)

Streckung von Vektor u (rot) um den skalaren Faktor $\left(\frac{\langle u, x \rangle}{\langle u, u \rangle} \right)$

Das bedeutet man kann einen Endpunkt überall auf der schwarzen Linie bekommen.



4c)

$$P(x) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \left(\frac{\langle u, x \rangle}{\langle u, u \rangle} \right) \cdot u \Rightarrow \left(\frac{\langle u, x \rangle}{\langle u, u \rangle} \right) = 0 \quad (90)$$

$$\Rightarrow \langle u, x \rangle = 0 \quad (91)$$

Diese obere Gleichung ist gleich Null, wenn der Vektor x senkrecht zum Vektor u steht. Deshalb muss der Vektor u der Normalvektor der folgenden Ebene sein.

$$u_1 x + u_2 y + u_3 z = 0 \quad (92)$$

Da u_3 aber nicht gleich Null sein kann und man u_3 auch nicht mit u_1 und u_2 als Linearkombination darstellen kann muss $z = 0$

$$\Rightarrow u_1 x + u_2 y = 0 \quad (93)$$

$$\ker P(x) = \text{span} \left\{ \begin{pmatrix} u_1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ u_2 \\ 0 \end{pmatrix} \right\} \quad (94)$$

Deshalb muss auch $\dim \ker P(x) = 2$ sein.

Der Bildvektor kann dem Ursprungsvektor entnommen werden.

$$\text{Im } P(x) = \text{span} \left\{ \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \right\} \quad (95)$$

Deshalb muss auch $\dim \text{Im } P(x) = 1$ sein.

4d)

$$P_1(e_1) = P_1 \left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) = \left(\frac{\left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right)}{2} \right) \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \frac{1}{2} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{pmatrix} \quad (96)$$

$$P_2(e_2) = P_2 \left(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \right) = \left(\frac{\left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \right)}{2} \right) \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \frac{1}{2} \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ 0 \\ 0 \end{pmatrix} \quad (97)$$

$$P_3(e_3) = P_3 \left(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right) = \left(\frac{\left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right)}{2} \right) \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = 0 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad (98)$$

$$\Rightarrow A = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (99)$$

Kontrolle:

$$A\mathcal{B} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \checkmark \quad (100)$$

5.6 Abbildungen Begriffe und Eigenschaften

$$F(x) = \text{Im } F \equiv \{F(x) \in Y \mid x \in X\} \subseteq Y \quad (101)$$

$$\text{Urbild } f^{-1}(y) = \{x \in X \mid F(x) = y\} \quad (102)$$

Note 2. In \mathbb{N} ist die Funktion $x \mapsto x^2$ injektiv, jedoch in \mathbb{Z} nicht.

5.7 Lineare Abbildungen

Gegeben $F : X \rightarrow Y \quad X = \text{span}\{b_{a1}, \dots, b_n\} \xrightarrow{F} Y = \text{span}\{c_{a1}, \dots, c_m\}$ (103)

$$x = \epsilon_{p=1}^n \epsilon_p b_p \quad y = \sum_{k=1}^m \eta_k c_k \quad (104)$$

$$y := F(x) = F\left(\sum_{p=1}^n \epsilon_p b_p\right) = \sum_{p=1}^n \epsilon_p F(b_p) = \sum_{l=1}^n \epsilon_p \sum_{k=1}^m a_{kp} c_k = \sum_{k=1}^m \underbrace{\left(\sum_{l=1}^n a_{kp} \epsilon_p\right)}_{\eta_k} c_k \quad (105)$$

$$\Rightarrow \eta_k = \sum_{l=1}^n a_{kl} \epsilon_l \begin{pmatrix} a_{k1} & \dots & a_{kn} \end{pmatrix} \begin{pmatrix} \epsilon_1 \\ \dots \\ \epsilon_n \end{pmatrix} \quad (106)$$

5.8 Kern

Kern $F = \text{Urbild von } 0_y \in Y$

5.9 Rang

Rang einer linearen Abbildung $\text{Rang } F = \dim \text{Im } F$

Theorem 5.1: Dimensionsformel

$$F : X \rightarrow Y \text{ lineare Abbildung} \quad (107)$$

$$\dim X = \dim \ker F + \dim \text{Im } F \quad (108)$$

Es gelten folgende Äquivalenzen:

1. $F : X \rightarrow Y$ injektiv $\Leftrightarrow \text{Rang } F = \dim X$
2. $F : X \rightarrow Y$ bijektiv $\Leftrightarrow \text{Rang } F = \dim X = \dim Y$
3. $F : X \rightarrow X$ bijektiv $\Leftrightarrow \text{Rang } F = \dim X \Leftrightarrow \ker F = \{0\}$

5.10 Matrix als lineare Abbildung

$Ax = b$ lösbar $\Leftrightarrow b \in \text{span}\{a_1, \dots, a_n\} \Leftrightarrow \text{Im } A$.

Anzahl Pivotelemente, $\ker A$: $n-r$ freiwählbare Parameter

Theorem 5.15

$$A \in \mathbb{E}^{m \times n}$$

$$\text{Im } A = R(A) = \text{span}\{a_{n1}, \dots, a_n\} = \text{span}\{a_{n2}, \dots, a_{nr}\} \quad (109)$$

n_i = i-te Pivotkolone von A

The following was presented in the exercise on 22. November 2018.

6 Vektorräume mit Skalarprodukt

6.1 Norm

Vektorraum V/\mathbb{E}

Norm ist eine Funktion $\|\cdot\| : V \rightarrow \mathbb{R}$
 $x \mapsto \|x\|$ mit folgenden Eigenschaften:

1. $\|x\| \geq 0, \forall x \in V$
 $\|x\| = 0, \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$ (positiv definit)
2. $\|\alpha x\| = |\alpha| \|x\|, \forall \alpha \in \mathbb{E}, \forall x \in V$ (homogen)
3. $\|x + y\| \leq \|x\| + \|y\|$ (Dreiecksungleichung)

Vektorraum mit Norm ist ein normierter Vektorraum

6.2 Skalarprodukt

Vektorraum $V/\mathbb{E} \quad \forall x, y, z \in V, \forall \alpha \in \mathbb{E}$
 $\langle \cdot, \cdot \rangle : V \times V \rightarrow \mathbb{E}, \quad x, y \mapsto \langle x, y \rangle$

1. Linear im zweiten Faktor
2. Symmetrisch im Reelen oder Hermitisch in den Komplexen Zahlen
3. Positiv Definit

6.3 Spur

$$\text{spur}(A) = \sum_{i=1}^n a_{ii} \quad (110)$$

$$(A^T B)_{ij} = \sum_{k=1}^n a_{ik}^t b_{kj} = \sum_{k=1}^n a_{ki} b_{kj} \quad (111)$$

$$\text{spur}(A^T B) = \sum_{i=1}^n (A^T B)_{ii} = \sum_{i=1}^n \sum_{k=1}^n a_{ki} b_{ki} = \sum_{ik} a_{ki} b_{ki} \quad (112)$$

6.4 Cauchy-Schwarz Ungleichung

$$|\langle x, y \rangle| \leq \|x\| \|y\| \quad (113)$$

Winkel:

$$\phi = \arccos \frac{\operatorname{Re} \langle x, y \rangle}{\|x\| \|y\|} \quad (114)$$

In den reellen Zahlen ist es das Gleiche wie das Skalarprodukt.

6.5 Orthogonalität

$$x, y \in V \text{ orthogonal} \Leftrightarrow \langle x, y \rangle = 0 \quad (x \perp y) \quad (115)$$

Wenn zwei Vektoren senkrecht aufeinander stehen bedeutet das nicht dass sie orthogonal sind.
(In den komplexen Zahlen)

$$x \perp y : \|x + y\|^2 = \|x\|^2 + \underbrace{2 \langle x, y \rangle}_{0} + \|y\|^2 \quad (116)$$

6.6 Orthonormalbasen

Theorem 6.3

Eine Menge M ($0 \notin M$) von paarweisen orthogonalen Vektoren ist linear unabhängig.

Das bedeutet paarweise orthogonale Vektoren spannen ihren n dimensionalen Vektorraum auf.
Eine Basis $\{b_1, b_2, \dots, b_n\}$ heisst orthogonal $\Leftrightarrow \langle b_k, b_l \rangle = 0 \forall k \neq l$
Eine Basis $\{b_1, b_2, \dots, b_n\}$ heisst orthonormal \Leftrightarrow zusätzlich gilt $\langle b_k, b_k \rangle = 1$ (daher alle Basisvektoren besitzen die Länge 1)

$$\langle b_l, b_k \rangle = \begin{cases} 1, & l = k \\ 0, & \text{else} \end{cases} \quad \text{schreibt man auch } \delta_{lk} \quad (117)$$

$\{b_1, b_2, \dots, b_n\}$ orthogonal $\Leftrightarrow \{\tilde{b}_1, \tilde{b}_2, \dots, \tilde{b}_n\}$ orthonormal $\tilde{b}_i = \frac{b_i}{\|b_i\|}$

Orthogonale Matrix $Q \Leftrightarrow Q^T Q = I$

\Rightarrow Orthogonale Matrix hat orthonormale Spalten, da sie eine orthonormale Basis bildet.

Theorem 6.4

Vektorraum V mit Skalarprodukt und $\{b_1, b_2, \dots, b_m\}$ orthonormale Basis

$$\forall x \in V : x = \sum_{k=1}^r \langle b_k, x \rangle b_k \quad (118)$$

Die Koordinaten von x bezüglich $B : \gamma_k = \langle b_k, x \rangle$

$$B = \begin{pmatrix} | & \cdots & | \\ \bar{b}_1 & \cdots & \bar{b}_n \\ | & \cdots & | \end{pmatrix} \Rightarrow \gamma = B^T x \quad (119)$$

6.7 Parsevalsche Formel

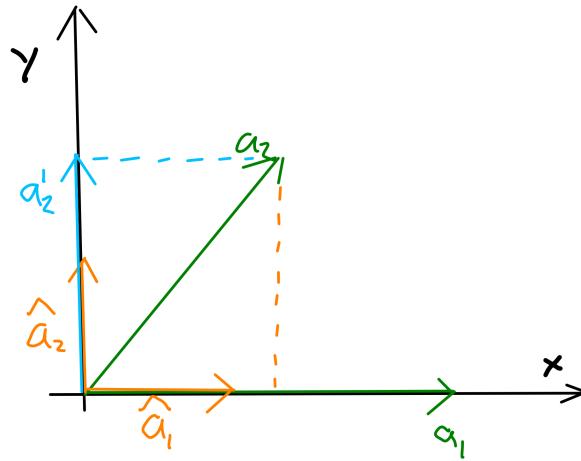
Sei $B = \{b_1, b_2, \dots, b_n\}$ eine orthonormale Basis von Vektorraum V
 $\forall x, y \in V, \gamma, \eta \delta$ die Koordinaten von x, y bezüglich B

$$\langle x, y \rangle = \langle B^T x, B^T y \rangle = \langle \gamma, \eta \rangle \quad (120)$$

$$\|x\| = \|B^T x\| = \|\gamma\| \quad (121)$$

6.8 Gram-Schmidt-Orthogonalisierungsverfahren

Gegeben $A = \left\{ \begin{pmatrix} 2 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right\}$ sind linear unabhängig aber nicht orthonormal, $\text{span } A = \mathbb{R}^2$



Algorithmus 6.1: Gram-Schmidt-Orthogonalisierungsverfahren

In $\{a_1, a_2, \dots, a_n\} \subseteq V$, linear unabhängig, output: $\{b_1, b_2, \dots, b_n\}$ die orthonormal ist.

$$b_1 = \frac{a_1}{\|a_1\|} \quad (122)$$

```

1 for k=2:n
2    $\tilde{b}_k := a_k - \sum_{j=1}^{k-1} \langle b_j, a_k \rangle b_j$ 
3    $b_k := \frac{\tilde{b}_k}{\|\tilde{b}_k\|}$ 
4 end

```

⇒ Aus jeder orthogonalen / nicht-orthogonalen Basis kann man eine orthonormale Basis konstruieren.

The following was presented in the exercise on 29. November 2018.

6.9 Basiswechsel und Koordinaten Transformation von orthhagonal Basen

$$B = \begin{pmatrix} | & \cdots & | \\ \bar{b}_1 & \cdots & \bar{b}_n \\ | & \cdots & | \end{pmatrix} \quad B' = \begin{pmatrix} | & \cdots & | \\ \bar{b}_1 & \cdots & \bar{b}_n \\ | & \cdots & | \end{pmatrix} \quad (123)$$

Die Matrizen sind orthonormal

$$BT = B' \quad T = (T_{jk}) \quad (124)$$

Basistransformationsmatrix

$$\Rightarrow T = B^{-1}B' = B^T B' \quad (125)$$

T orthogonal, da B, B' orthogonal (Satz 2.20)

6.10 Darstellung von linearen Abbildungen

$F : X \rightarrow Y$ eine lineare Abbildung, X, Y sind ein Vektorraum / \mathbb{E} mit Skalarprodukt B, B' Orthonormalbasen von X

D, D' Orthonormalbasen von X

Folgende Verhältnisse gelten:

$$A = SBT^H \quad (126)$$

$$B = S^H AT \quad (127)$$

6.11 Orthogonale und unitäre Abbildungen

X, Y ist ein Vektorraum / \mathbb{E} mit Skalarprodukt

$F : X \rightarrow Y$ ist eine lineare Abbildung und heisst orthogonal falls:

$$\forall x_1, x_2 \in X : \quad \langle F(x_1), F(x_2) \rangle_Y = \langle x_1, x_2 \rangle_X \quad (128)$$

Folgende Eigenschaften gelten

Theorem 6.13

1. $\forall x \in X : \quad \|F(x)\|_Y = \|x\|_X, \quad \|x\|_X = \sqrt{\langle x, x \rangle_X}$ (Längentreu)
2. $v \perp w \Leftrightarrow F(v) \perp F(w)$ (Winkeltreu)
3. $\ker F = \{0\}$ (F ist injektiv)

Falls $\dim X = \dim Y < \infty$ gilt zusätzlich:

4 F ist bijektiv

5 $\{b_1, b_2, \dots, b_s\}$ sind orthonormierte Basen in X $\Leftrightarrow \{F(b_1), \dots, F(b_n)\}$ sind orthonomale Basen von Y

6 Die Abbildungsmatrix von F bezüglich einer orthonomalen Basis ist orthogonal / unitär.

Example 5.

$$F : \mathbb{R}^2 \rightarrow \mathbb{R}^3, F\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) \rightarrow \begin{pmatrix} x_1 \cos \phi - x_2 \sin \phi \\ x_1 \sin \phi + x_2 \cos \phi \\ 0 \end{pmatrix} \quad (129)$$

$$A = \begin{pmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \\ 0 & 0 \end{pmatrix} \quad A' = \begin{pmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{pmatrix} \quad (130)$$

$$A'^T A' = \begin{pmatrix} \cos \phi^2 + \sin \phi^2 & 0 \\ 0 & \sin \phi^2 + \cos \phi^2 \end{pmatrix} \quad (131)$$

$$A^T A = (A'^T 0^T)(A'^T A + 0^T 0) = I_2 \quad (132)$$

$$x, y \in \mathbb{R}^2 : \langle F(x), F(y) \rangle = \langle Ax, Ay \rangle = (Ax)^T Axy = x^T A^T Ay = x^T y = \langle x, y \rangle \quad (133)$$

6.12 Induzierte Matrizen

Gegeben $A \in \mathbb{R}^{n \times n}$, Vektorraum $\|\cdot\|_V$

Die Matrizen

$$A \mapsto \|A\|_m \equiv \sup_{x \neq 0} \frac{\|Ax\|_V}{\|x\|_V} \quad (134)$$

heisst die durch die Vektornorm $\|\cdot\|_V$ induzierte Matrixraum.

$\|\cdot\|_V = \|\cdot\|_2$ heisst $\|\cdot\|_m$ Spektralnorm..

$$\alpha = \|x\|_V, \quad \tilde{x} = \frac{x}{\|x\|_V} = \frac{x}{\alpha} \quad (135)$$

$$\|Ax\|_V = \|\alpha A \frac{x}{\alpha}\|_V = \alpha \|A \frac{x}{\alpha}\|_V = \|x\|_V \|A\tilde{x}\|_V \quad (136)$$

$$\sup_{x \neq 0} \frac{\|Ax\|_V}{\|x\|_V} = \sup_{x \neq 0} \frac{\|x\|_V \|A\tilde{x}\|_V}{\|x\|_V} = \sup_{\|x\|_V=1} \|A\tilde{x}\|_V \quad (137)$$

$\|\cdot\|_m$: Die Maximallänge der Bildvektoren vom Einheitskreis.

The following was presented in the exercise on 06. December 2018.

6.13 Induzierte Operationen

$\mathcal{L}(X, Y)$ Die Menge der beschränkten Linearen Abbildungen.

$$(F : X \rightarrow Y \exists \gamma_F \geq 0 \forall x \in X : \|F(x)\|_Y \leq \gamma_F \|x\|_Y) \quad (138)$$

Induzierte Operationen: $\|\cdot\| : \mathcal{L}(X, Y) \rightarrow \mathbb{R}$

$$F \mapsto \|F\| = \sup_{x \neq 0} \frac{\|F(x)\|_Y}{\|x\|_X} = \sup_{\|x\|=1} \|F(x)\|_Y \quad (139)$$

$$X = \mathbb{E}^n, Y = \mathbb{E}^m, \mathcal{L}(X, Y) = \mathbb{E}^{m \times n} \quad (140)$$

$$\|\cdot\|_{\mathbb{E}^{m \times n}} \rightarrow \mathbb{R} \quad (141)$$

$$A \mapsto \|A\| = \sup_{x \neq 0} \frac{\|Ax\|_Y}{\|x\|_X} \quad (142)$$

6.14 Konditionszahl einer regulären Matrix $A \in \mathbb{E}^{m \times n}$

$$A = \begin{pmatrix} 1 & 2 \\ 1.001 & 2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, b_2 = \begin{pmatrix} 0.99 \\ 2.001 \end{pmatrix} = b_1 + \epsilon \quad (143)$$

$$Ax_1 = b_1 \Rightarrow x_1 \begin{pmatrix} 1000 \\ -499.5 \end{pmatrix} \quad (144)$$

$$Ax_1 = b_2 \Rightarrow x_2 \begin{pmatrix} 1011 \\ -505.005 \end{pmatrix} \quad (145)$$

$$\|x_1 - x_2\|_2 = \|A^{-1}b_1 - A^{-1}b_2\| = \|A^{-1}b_1 - A^{-1}(b_1 + \epsilon)\|_2 = \|A^{-1}\epsilon\|_2 \quad (146)$$

Wir wollen: $Ax = b$; $Ax' = b'$; $A'x'' = b$

Falls $\|A - A'\|$ und $\|b - b'\|$ ist klein

Dann soll $\|x - x'\|$ und $\|x - x''\|$ ebenfalls klein sein.

ϵ Abweichung in b; $A^{-1}\epsilon$: Abweichung in x

$$R_x = \frac{\|A^{-1}\epsilon\|}{\|x\|} \text{ ist der relative Fehler in x} \quad (147)$$

$$R_b = \frac{\|\epsilon\|}{\|b\|} \text{ Relativer Fehler in b} \quad (148)$$

Nun ist der relative Fehler

$$\frac{R_x}{R_b} \quad (149)$$

$$\frac{A^{-1}\epsilon}{\|\epsilon\|} \frac{\|b\|}{\|A^{-1}b\|} = \frac{A^{-1}\epsilon}{\|\epsilon\|} \frac{\|Ax\|}{\|x\|} \quad (150)$$

$$\sup_{\epsilon \neq 0} \frac{A^{-1}\epsilon}{\|\epsilon\|} \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \|A^{-1}\| \|A\| = k(A) \text{ Kordinationszahl} \quad (151)$$

$$A \text{ singulär } k(A) := \infty \quad (152)$$

7 Methode der kleinsten Quadrate

$$A \in \mathbb{E}^{m \times n}, m > n, Ax = b \quad (153)$$

überbestimmtes LGS \rightarrow in der Regel nicht lösbar.

$$\begin{array}{c} m \\ \times \\ | \\ A \\ | \\ n \end{array} \quad \begin{array}{c} | \\ X \\ | \end{array} = \begin{array}{c} | \\ b \\ | \end{array} \quad \begin{array}{l} A \in \mathbb{E}^{m \times n} \\ \text{LGS in der Regel} \\ \text{nicht lösbar.} \end{array}$$

\Rightarrow Suchen x , sodass $x = \arg \min \|Ax - b\|_2^2$

$$\text{Falls } \min_x \|Ax - b\|_2^2 > 0 \Rightarrow b \notin R(A) = \text{span}\{a_1, \dots, a_n\}, a_k \in \mathbb{E}^m, k = 1, \dots, n \quad (154)$$

$$Ax - b \perp R(A) \Rightarrow Ax - b \in R(A)^T \quad |R(A)^T = \mathbb{N}(A^H) \quad (155)$$

$$Ax - b \in \mathbb{N}(A^H) \quad (156)$$

$$\forall x \in \mathbb{N}(A^H) : A^H x = 0 \quad (157)$$

$$A^H(Ax - b) = 0 \quad (158)$$

$$A^H Ax - A^H b = 0 \quad (159)$$

$$A^H Ax = A^H b \quad (\text{Normalgleichung}) \quad (160)$$

\Rightarrow Falls $A^H A$ regulär

$$\Rightarrow \arg \min_{x \in \mathbb{E}^n} \|Ax - b\|_2 = (A^H A)^{-1} A^H b \quad (161)$$

Falls $\text{Rang } A < n$

$$A^H Ax = A^H b = \tilde{A}x = \tilde{b} \quad (162)$$

Die obige Gleichung hat unendlich viele Lösungen.

Example 6. Finde die Koeffizienten eines quadratischen Polynoms

$$p(t) = a_0 + a_1 t + a_2 t^2 = y \quad (163)$$

Für die folgenden Punkte: $(0, 0), (1, 1), (2, 0.5), (3, 2)$ sodass

$$\sum_{i=1}^4 (p(x_i) - y_i)^2 \quad (164)$$

$$\underbrace{\begin{pmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ 1 & t_4 & t_4^2 \end{pmatrix}}_{\text{Vandermonde-Matrix}} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \quad (165)$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0.5 \\ 2 \end{pmatrix} \quad (166)$$

$$x = (A^T A)^{-1} A^T b = \begin{pmatrix} 0.175 \\ 0.175 \\ 0.125 \end{pmatrix} \quad (167)$$

Note 3. Tips Serie 11: 1) Finde $\alpha\beta$ sodass $\sum_{i=1}^5 (\alpha g_1(x_i) + \beta g_2(x_i) - y_i)^2$ sollte minimal sein.

The following was presented in the exercise on 13. December 2018.

Note 4.

$$\text{Lineare Abbildung } F \text{ ist beschränkt} \Leftrightarrow \exists \gamma_F \geq 0 \forall x \in X \quad \|F(x)\|_Y \leq \gamma_F \|x\|_X \Leftrightarrow \frac{\|F(x)\|_Y}{\|x\|_X} \leq \gamma_F \quad (168)$$

$F : X \rightarrow Y$

$$\|F\| = \sup_{x \neq 0} \frac{\|F(x)\|_Y}{\|x\|_X} \quad (169)$$

8 Determinante

8.1 Permutationen

$$P\{1, \dots, n\} \rightarrow \{1, \dots, n\} \quad p(1234) = 2143 \quad (170)$$

Transposition ist eine Permutation, bei der nur zwei Elemente vertrauscht werden. $(1234 \rightarrow 2134)$

Die Menge aller Permutationen auf die Menge $\{1, \dots, n\}$: S_n ist $|S_n| = n!$

Theorem 8.2

Für $n > 1$: Jede Permutation $p \in S_n$ kann als Produkt von transponierten benachbarten Elementen dargestellt werden.

$$p = t_v \circ \dots \circ t_1 \quad (171)$$

Die Darstellung ist nicht eindeutig, aber v ist immer gerade oder ungerad für ein gegebenes p .

Definition 9. Das Signum von p :

$$\text{sign}(p) = \begin{cases} 1, & \text{falls } v \text{ gerade oder falls } v \text{ id ist} \\ -1, & \text{falls } v \text{ ungerade} \end{cases} \quad (172)$$

Example 7.

$$p(1234) = 2143 = p(1)p(2)p(3)p(4) \quad (173)$$

Oder man stellt es als transposition dar

$$1234 \rightarrow 1243 \rightarrow 2143 \quad (174)$$

9 Definition der Determinante

$$\det \mathbb{E}^{n \times n} \rightarrow \mathbb{E}, \quad A \mapsto \det A, \quad |A| \quad (175)$$

Haupteigenschaft:

$$\det A \neq 0 \Leftrightarrow A \text{ regulär} \quad (176)$$

$$\det A := \sum_{p \in S_n} \text{sign}(p) \cdot a_{1,p(1)} \cdot a_{2,p(2)} \cdot \dots \cdot a_{n,p(n)} = \sum_{p \in S_n} \text{sign}(p) \cdot \prod_{k=1}^n a_{k,p(k)} \quad (177)$$

$$\mathcal{O}(n \cdot n!) \Rightarrow \text{Teuer so zu berechnen} \quad (178)$$

Example 8.

$$\det \begin{pmatrix} 2 & 1 \\ 3 & 4 \end{pmatrix} = 2 \cdot 4 + (-1) \cdot 1 \cdot 3 = 5 \quad (179)$$

Permutationen:

$$\begin{pmatrix} 2 & 1 \\ 3 & 4 \end{pmatrix} \quad \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix} \quad (180)$$

$$\det \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = a_{11} \cdot a_{22} - a_{12} \cdot a_{21} \quad (181)$$

Example 9. Für 3×3 Matrizen hat man 6 Permutationen.

$$\det \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = a_{11} \cdot a_{22} \cdot a_{33} + a_{13} \cdot a_{21} \cdot a_{31} + a_{22} \cdot a_{23} \cdot a_{31} - a_{11} \cdot a_{23} \cdot a_{32} - a_{12} \cdot a_{21} \cdot a_{33} - a_{13} \cdot a_{22} \cdot a_{31} \quad (182)$$

$$(123) \Rightarrow v = 0 \quad (213) \Rightarrow v = 1 \quad (231) \Rightarrow v = 2 \quad (183)$$

$$(132) \Rightarrow v = 1 \quad (312) \Rightarrow v = 2 \quad (321) \Rightarrow v = 3 \quad (184)$$

$$\Rightarrow a_{11} \cdot (a_{22} \cdot a_{33} - a_{23} \cdot a_{32}) - a_{12} \cdot (a_{21} \cdot a_{33} - a_{23} \cdot a_{31}) + a_{13} \cdot (a_{21} \cdot a_{32} - a_{22} \cdot a_{31}) \quad (185)$$

Example 10.

$$\begin{pmatrix} r_{11} & \dots & \dots & r_{1n} \\ 0 & r_{kk} & \dots & r_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & r_{nn} \end{pmatrix} = \prod_{k=1}^n r_{kk} \quad (186)$$

Jeder andere Term fällt weg, da bei einer Permutation mit Null multipliziert wird.

Theorem 9.3

det hat folgende Eigenschaften:

1. det ist linear in jeder einzelnen Zeile

$$\det \begin{pmatrix} \underline{a_1} \\ \vdots \\ \alpha u_l + \beta v_l \\ \vdots \\ \underline{a_n} \end{pmatrix} = \alpha \cdot \det \begin{pmatrix} \underline{a_1} \\ \vdots \\ u_l \\ \vdots \\ \underline{a_n} \end{pmatrix} + \beta \cdot \det \begin{pmatrix} \underline{a_1} \\ \vdots \\ v_l \\ \vdots \\ \underline{a_n} \end{pmatrix} \quad (187)$$

$$\det(\gamma A) = \det \begin{pmatrix} \underline{\gamma a_1} \\ \underline{\gamma a_2} \\ \vdots \\ \underline{\gamma a_n} \end{pmatrix} \Rightarrow \gamma^n \cdot \det(A) \quad (188)$$

2. Vertausche zwei Zeilen von A, so wechselt das Vorzeichen von det A.
3. $\det(I) = 1$
4. Hat A eine Zeile von 0, so ist $\det A = 0$
5. $\det(\gamma A) = \gamma^n \det(A)$
6. Hat A zwei gleiche Zeilen, so ist $\det A = 0$

7. Addiert man zu einer Zeile von A ein Vielfaches einer anderen Zeile, so ändert die $\det(A)$ nicht.
8. Ist A eine Diagonal / Dreiecksmatrix, so ist $\det A = \prod_{k=1}^n a_{kk}$

Theorem 9.5

Wendet man Gauss auf A an, dann gilt:

$$\det(A) = (-1)^V \prod_{k=1}^n r_{kk} \quad (189)$$

wobei r_{ij} die Elemente der resultierender Rechtsdreiecksmatrix ist.

$$A \text{ regulär} \Leftrightarrow r_{kk} \neq 0 \forall k = 1, \dots, n \Leftrightarrow \text{Rang } A = n \Leftrightarrow \det(A) \neq 0 \quad (190)$$

Theorem 9.4

$$\det(AB) = \det(A) \cdot \det(B) \quad A, B \in \mathbb{E}^{n \times n} \quad (191)$$

Example 11. Wenn A regulär ist, was ist $\det(A^{-1})$?

$$1 = \det(I) = \det(AA^{-1}) = \det(A)\det(A^{-1}) \Rightarrow \frac{1}{\det(A)} = \det(A^{-1}) \quad (192)$$

$A \in \mathbb{E}^{n \times n}$, P ist die Permutationsmatrix $\det(AP) = \det(A) \cdot \det(P) = \det(A) \cdot (-1)^v$ wobei v ist die Anzahl der Vertauschungen.

Ebenso gilt $\det(PA) = \det(P) \cdot \det(A) = (-1)^v \cdot \det(A)$

Example 12. $A \in \mathbb{E}^{n \times n}$

$$PA = LR \Leftrightarrow A = P^T LR \quad (193)$$

$$\det(A) = \det(P^T LR) \quad (194)$$

$$= \det(P^T) \cdot \det(LR) \quad (195)$$

$$= \det(P^T) \cdot \det(L) \cdot \det(R) \quad (196)$$

$$= (-1)^v \cdot 1 \cdot \prod r_{kk} \quad (197)$$

$$= (-1)^v \cdot \prod r_{kk} \quad (198)$$

$$(199)$$

So kann man Satz 8.5 herleiten.

Example 13. $A^T : \det(A)$

$$A^T = (P^T LR)^T = R^T L^T P \quad (200)$$

$$\det(A^T) = \det(R^T L^T P) = \det(R^T) \det(L^T) \det(P) = \det(A) \quad (201)$$

Note 5. Tips Serie 12:

3. Schreib die Definition einer Determinante. Welche Permutationen sind zulässig sodass keine Permutationen in der Diagonale sind.
4. Zeilen/Kolonnen vertauschen, transformieren mittels Permutationsmatrizen (Es ist ohne Gaußelimination möglich)

The following was presented in the exercise on 20. December 2018.

10 Eigenwerte und Eigenvektoren

$A \in \mathbb{E}^{n \times n}, x \in \mathbb{E}^n$ heisst Eigenvektor zum Eigenwert $\lambda \in \mathbb{E}$ falls:

$$Ax = \lambda x \quad (202)$$

$$E_\lambda := \{x \in V | Ax = \lambda x\} \quad (203)$$

$$Ax = \lambda x \Leftrightarrow Ax - \lambda x = 0 \quad (204)$$

$$(A - I\lambda)x = 0 \quad (205)$$

$\Rightarrow \ker(A - I\lambda) = E_\lambda$ (damit $E_\lambda \neq 0$ muss $A - I\lambda$ singulär sein.)

Menge der Eigenwerten von A nennt man Spektrum von A

$$\underbrace{\dim E_\lambda}_{d_\lambda} = \dim \ker(A - I\lambda) = \text{Geometrische Vielfachheit} \quad (206)$$

Dimensionsformel besagt:

$$\underbrace{\dim V}_n = \underbrace{\dim \text{Im } \tilde{A}}_{\text{Rang } \tilde{A}} + \underbrace{\dim \ker \tilde{A}}_{d_\lambda} \quad (207)$$

10.1 Charakteristische Gleichung

$$\lambda \text{ ist ein Eigenwert von } A \Leftrightarrow \underbrace{\det(A - I\lambda)}_{\chi_A(\lambda) \text{ charakteristisches Polynom}} = 0 \quad (208)$$

λ Eigenwert von A $\Leftrightarrow \lambda$ Nullstellen von $\chi_A(\lambda)$ $\Rightarrow n \times n$ Matrix hat n Eigenwerte (nicht unbedingt verschieden)

Gegeben Eigenwert λ , wie findet man Eigenvektoren x?

$$Ax = \lambda x \Leftrightarrow (A - I\lambda)x = 0 \text{ (homogenes LGS lösen)} \quad (209)$$

Note 6. Falls A singulär ist: $Ax = \lambda x, A = 0, Ax = 0x = 0 \Rightarrow A$ hat Eigenwert $\lambda = 0$

Example 14.

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 2 & 5 \\ 0 & 0 & 3 \end{pmatrix} \quad (210)$$

Finde alle Eigenwerte und ihre Eigenvektoren.

$A - I\lambda$ muss singulär sein.

$\chi_A(\lambda) = \det(A - I\lambda) = 0$ Nullstellen finden

$$\chi_A(\lambda) = (1 - \lambda)(2 - \lambda)(3 - \lambda) \quad (211)$$

$$\lambda_1 = 1, \lambda_2 = 2, \lambda_3 = 3$$

$$Ax = \lambda x \Leftrightarrow (A - I\lambda)x = 0 \quad (212)$$

$$A = \begin{pmatrix} 1 - \lambda & 2 & 3 \\ 0 & 2 - \lambda & 5 \\ 0 & 0 & 3 - \lambda \end{pmatrix} \quad (213)$$

Für $\lambda = 1$

$$A = \left(\begin{array}{ccc|c} 0 & 2 & 3 & 0 \\ 0 & 1 & 5 & 0 \\ 0 & 0 & 2 & 0 \end{array} \right) \Rightarrow x_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad (214)$$

Kontrolle:

$$\begin{pmatrix} 1 & 2 & 3 \\ 0 & 2 & 5 \\ 0 & 0 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = 1x \quad (215)$$

Gleich für $\lambda = 2$ und $\lambda = 3$ $d_a = 1$

$$d_\lambda = n - \text{Rang}(A - \lambda I) = 3 - 2 = 1 \quad (216)$$

Example 15.

$$A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad (217)$$

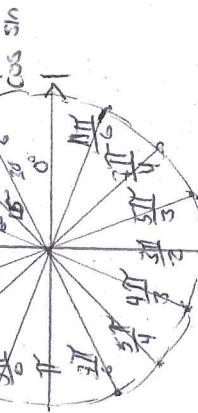
Was sind die Eigenwerte?

$$\chi_A(\lambda) = \det(A - \lambda I) = \lambda^2 + 1 \quad (218)$$

$\lambda_1 = i$ und $\lambda_2 = -i \Rightarrow$ Reelle Matrizen können komplexe Eigenwerte haben.

Komplexe Zahlen

$$\Phi = \frac{\theta + 2\pi k}{q} \quad q := \begin{cases} q & \text{real} \\ q+iq & \text{komplex} \end{cases}$$



Lineare Gleichungssysteme

Rang $r = \text{Anzahl Privatlemente}$
 Zeilen zweier Spalten spalten
 hat: (in Spalten, in Zeilen)
 $Ax = b$ hat:

- (i) genau eine Lösung $\Leftrightarrow r = n = m$ oder $r = n < m$ mit $c_m = 0$
- (ii) für jedes b mindestens eine Lösung $\Leftrightarrow r = m$
- (iii) für jedes b genau eine Lösung $\Leftrightarrow r = n < m$

Homogen: Rechte Seite besteht nur aus Nullen
 Private Lösung $x_1 = \dots = x_n = 0$ / Nicht trivial! Sonst
 Inhomogen: Sonst
 K.1.7 Quadratisches Gleichungssystem Ax = b in Gleichungen und n Unbekannten
 entweder

- (i) $r = \text{Rang } A = n \Rightarrow A$ ist regulär
- (ii) $r = \text{Rang } A < n \Rightarrow A$ singulär

$$\text{Symmetrisch: } A^T = A \quad \text{Hermetisch: } A^H = \overline{(A)}^T$$

\Rightarrow Diagonalelemente einer hermitischen Matrix sind reell!

Schießsymmetrisch: $A^H = -A$
 T 2.6.1: $(A^H)^T = A$ (i) $(\alpha A)^T = \alpha A^T$ ($\alpha \neq 0$)
 (ii) $(A+B)^T = A^T + B^T$ (iii) $(AB)^T = B^T A^T$
 (iv) $(\alpha A)^H = \overline{\alpha} A^H$ (v) $(AB)^H = B^H A^H$
 $AB = BA \Leftrightarrow A$ spiegelisch für symmetrische A, B; sonst nicht!
 $A^T A = A A^T$ für beliebiger A; analog für hermitisch

$\langle xy \rangle = x^H y = \sum_{k=1}^n x_k y_k$ Es gilt folgendes.

$$\begin{aligned} &\text{Möglichkeit 1: linear im zweiten Faktor} \\ &\langle x_1 y_1 z_1 \rangle = \langle x_1 y_1 \rangle + \langle x_1 z_1 \rangle \quad \forall x_1, y_1, z_1 \in \mathbb{C}^n \\ &\langle x_1 x_2 y_1 \rangle = \alpha \langle x_1 y_1 \rangle \end{aligned}$$

(S2) reeller Fall \rightarrow symmetrisch $\langle xy \rangle = \langle yx \rangle$
 komplex \rightarrow hermitisch $\langle xy \rangle = \overline{\langle yx \rangle}$

(S3) Positiv definit $\langle x_1 x_1 \rangle \geq 0 \quad \forall x_1 \in \mathbb{C}^n$
 $\langle x_1 x_2 \rangle = 0 \Rightarrow x_1 = 0$
 Aus (S1), (S2) folgt
 Das Skalarprodukt in \mathbb{R}^n ist bilinear / sesquilinear für C
 (S4) auch linear im ersten Faktor / konjugiert-linear
 $\langle w+x_1, y \rangle = \langle w, y \rangle + \langle x_1, y \rangle = \overline{\alpha} \langle x_1, y \rangle$

Kombination
 $\langle x_1, \alpha y + \beta z \rangle = \alpha \langle x_1, y \rangle + \overline{\beta} \langle x_1, z \rangle$
 $\langle \alpha w + Bx_1, y \rangle = \overline{\alpha} \langle w, y \rangle + \overline{B} \langle x_1, y \rangle$
 $\|x\| := \sqrt{\langle x, x \rangle} = \sqrt{x^H x} = \sqrt{\sum_{k=1}^n |x_k|^2} \geq 2$
 Bsp: $x = \begin{pmatrix} 1 & 2 \\ 1 & 3+4i \end{pmatrix} \Rightarrow \|x\| = \sqrt{1^2 + 3^2 + 4^2} = 3$

$$|\langle x_1, y \rangle| \leq \|x_1\| \cdot \|y\| \quad \forall x_1, y \in \mathbb{C}^n$$

Beweis: Für ein beliebiges $\alpha \in \mathbb{C}$ gilt:
 $0 \leq \langle \alpha x + y, \alpha x + y \rangle = \langle \alpha x, \alpha x + y \rangle + \langle y, \alpha x + y \rangle$
 $= \langle \alpha x, \alpha x \rangle + \langle \alpha x, y \rangle + \langle y, \alpha x \rangle + \langle y, y \rangle$
 $\stackrel{S1}{=} \alpha \langle \alpha x, x \rangle + \overline{\alpha} \langle x, y \rangle + \alpha \langle y, x \rangle + \langle y, y \rangle$
 Für $x = 0$ gilt die CSM offensichtlich, daher darf man $x \neq 0$ annehmen.
 Wählt $\alpha = -\frac{\langle x, y \rangle}{\|x\|^2}$. Nach Multiplikation mit $\langle x, x \rangle$ folgt:
 $0 \leq \langle K_x x, y \rangle - \frac{\langle x, y \rangle}{\|x\|^2} \cdot \langle x, y \rangle^2 - |\langle x, y \rangle|^2 + \langle x, x \rangle \cdot \langle y, y \rangle$
 $= -|\langle x, y \rangle|^2 + \langle x, x \rangle \cdot \langle y, y \rangle$
 $\Leftrightarrow |\langle x, y \rangle| \leq \sqrt{\langle x, x \rangle \cdot \langle y, y \rangle}$
 $\Leftrightarrow |\langle x, y \rangle| \leq \sqrt{\|x\| \cdot \|y\|}$
 $\Leftrightarrow |\langle x, y \rangle| \leq \sqrt{\|x\| \cdot \|y\|}$

Für die \mathbb{Z} -Norm in \mathbb{C}^n gilt

(M1) positiv definit $\|x\|=0 \Rightarrow x=0$

(M2) $\|x\| \geq 0$ $\forall x \in \mathbb{C}^n$

(M3) Beim Betrags nach homogen $\forall x \in \mathbb{C}^n \quad \alpha x \in \mathbb{C}^n$

(M4) $\|\alpha x\| = |\alpha| \|x\|$

(M5) Dreiecksungleichung

$$\|x+y\| \leq \|x\| + \|y\|$$

Beweis Dreiecksungleichung

$$\|x+y\|^2 = \langle x+y, x+y \rangle = \langle x, x \rangle + 2\langle x, y \rangle + \langle y, y \rangle = \|x\|^2 + 2\langle x, y \rangle + \|y\|^2$$

Man schätzt den zentralen Term $\langle x, y \rangle$ mittels C-S mit ab.

$$\|x+y\|^2 \leq \|x\|^2 + 2\|x\|\|y\| + \|y\|^2 = (\|x\| + \|y\|)^2$$

$$\|x+y\| \leq \|x\| + \|y\| \quad \square$$

$$\rho = \arccos \frac{\langle x, y \rangle}{\|x\|\|y\|} \quad 0 \leq \rho \leq \pi$$

Winkeln

$$\text{Winkeln sind orthogonal } \langle x, y \rangle = 0$$

$$\text{Pythagoras: } \|x+y\|^2 = \|x\|^2 + \|y\|^2 \text{ für } x, y \in \mathbb{E}^n \text{ und } x \perp y$$

$$\text{Beweis: } \|x+y\|^2 = \langle x+y, x+y \rangle = \langle x, x \rangle + 2\langle x, y \rangle + \langle y, y \rangle \stackrel{\text{Satz}}{=} \langle x, x \rangle + \langle x, y \rangle + \langle y, y \rangle = \langle x, y \rangle + \langle y, x \rangle = 2\langle x, y \rangle$$

Voraussetzung $\langle x, y \rangle = 0$

$$\|x+y\| := (\|x\|^p + \dots + \|x_n\|^p)^{\frac{1}{p}}, \text{ wobei } 1 \leq p \leq \infty$$

$$\text{oo-Norm: } \|x\|_{\text{oo}} := \max_{k=1, \dots, n} |x_k|$$

Ausseres / Fodisches Produkt eines m-Vektors und n-Vektors Y

Ist die $m \times n$ Matrix $X \cdot Y^H$ oder $XY^T \Leftrightarrow$ Matrix mit Rang 1 ist die orthogonale Projektion von X auf den

T 2.15 Die orthogonale Projektion von Y erreichte Größe durch den

die durch die Vektoren von Y erzeugte Größe durch den

Ursprung ist:

$$P_X Y = \frac{1}{\|Y\|^2} Y Y^H = \text{proj}_X Y \text{ und } c_Y \equiv \|Y\|^2$$

Es ist effizienter Y zu bestimmen

anstatt YY^H

Eine $n \times n$ Matrix heißt "unitär" falls $A^H A = I$

Eine reelle unitäre Matrix heißt orthogonal, für sie gilt $A^H A = I$

T 2.20 Sind A, B unitäre Matrizen, so gilt

(i) $A^{-1} = A^H$ (ii) $A^H A^{-1} = I$ (iii) A^{-1} ist unitär (iv) AB ist unitär

Die durch unitäre Matrizen definierten Abbildungen sind

längs- und winkelfrei, daher für alle $x, y \in \mathbb{R}^n$

$$\|Ax\| = \|x\|, \quad \langle Ax, Ay \rangle = \langle x, y \rangle$$

Orthogonal

L-R-Zerlegung

Gesucht: L (m x m) R (m x n)

Rezept: Gegeben: man Matrix A
Spaltenmaximum: Maximaler Beitrag von aktiver Zeile Tauschen.

Sei $P^{(0)} = \begin{pmatrix} 0 & 0 \\ 0 & \ddots & 0 \\ 0 & & 1 \end{pmatrix}_{(m-1)}$, $R^{(0)} = A$ (A zeigt aktuellen Schritt an)

2) Für $k=1, \dots, m-1$

$\rho^{(k)} = \rho^{(k-1)}$ und verlassene Zahlen l und k_l ($l \geq k$)

Spa-Hermessinn: Zeilen Tauschen

$R^{(k)} = R^{(k-1)}$ mit veränderten Zahlen k und l

$R^{(k)} = R^{(k-1)}$ durch max. ($m-k$) Anmerkungen von Gruss

$R^{(k)} = R^{(k-1)}$ entzieht aus R^{(k-1)} durch max. ($m-k$) Element aus Zeilen k und l vertauscht

$L^{(k)} = L^{(k-1)}$ wobei die ersten ($k-1$) Elemente aus $L^{(k-1)}$ wobei $L^{(k)}_{ij} = \frac{R^{(k)}_{ij}}{(R^{(k)}_{kk})^{1/k}}$ $i \neq k$

3) Kontrolle $PA = LR$

Rezept: Lösen eines Gleichungssystems $Ax=b$

1) Zeile $PA = LR$

2) Zeile $Ly = Pb$ nach y auf (Vorwärtsensetzen)

3) Zeile $Rx = y$ nach x auf (Rückwärtsensetzen)

Algorithmus: Für eine positiv definite reell symmetrische hermitische $n \times n$ -Matrix

Cholesky-Zerlegung

Für $1 \leq n$:

$$r_{ii} = \sqrt{a_{ii} - \sum_{j=1}^{i-1} \tilde{r}_{ij} \tilde{r}_{ij}} \quad \tilde{r}_{ik} := \frac{a_{ik} - \sum_{j=1}^{i-1} \tilde{r}_{ij} \tilde{r}_{ik}}{r_{ii}} \quad (k=1, \dots, n)$$

Sobald r_{ii} = 0 ist, bin letzten Schritt mit i=n entfallt die Z. Formel.

Lösung eines LGS

1) Algorithmus $A = R^H R$

2) Vorwärtsensetzen $Rx = c$ nach x auflösen

3) Rückwärtsensetzen $R^H y = c$ nach y auflösen

Vektorräume

Definition: nicht leere Menge mit Addition $x, y \in V \rightarrow x+y \in V$ und Multiplikation $\alpha \in \mathbb{F}, x \in V \rightarrow \alpha x \in V$ und folgende Grundregeln

$$\begin{aligned} V1) \quad & x+y = y+x \\ V2) \quad & (x+y)+z = x+(y+z) \\ V3) \quad & \exists 0 \in V \quad x+0=x \\ V4) \quad & \forall x \in V \quad \exists x' \in V \quad x+x'=0 \quad V' \quad x=x' \end{aligned}$$

$$V5) \quad \text{Sei V ein Vektorraum über Körper } K: \quad \forall x, y \in V, \quad \forall \alpha \in \mathbb{F}$$

$$\begin{aligned} (i) \quad & \alpha x=0 \quad (ii) \quad \alpha x=0 \Rightarrow x=0 \\ (iii) \quad & \alpha x=\alpha y \Rightarrow x=y \quad (iv) \quad \alpha(-x)=-(\alpha x) \end{aligned}$$

Nicht leere Menge K mit Addition $\alpha, \beta \in K \rightarrow \alpha+\beta \in K$ und Multiplikation $\alpha, \beta \in K \rightarrow \alpha \cdot \beta \in K$ und folgenden Grundregeln

$$K1) \quad \alpha+\beta=\beta+\alpha \quad K2) \quad (\alpha+\beta)+\gamma=\alpha+(\beta+\gamma)$$

$$K3) \quad \exists 0 \in K: \alpha+0=\alpha \quad K4) \quad \exists \alpha \in K: \alpha+\alpha^{-1}=0$$

$$K5) \quad \alpha \cdot \beta=\beta \cdot \alpha \quad K6) \quad \alpha \cdot (\beta+\gamma)=\alpha \cdot \beta+\alpha \cdot \gamma$$

$$K7) \quad 1 \in K, \quad 1 \neq 0: \alpha \cdot 1=\alpha \quad K8) \quad \forall \alpha \in K: \alpha \cdot \alpha^{-1}=1$$

$$K9) \quad \alpha \cdot (\beta+\gamma)=\alpha \cdot \beta+\alpha \cdot \gamma \quad K10) \quad (\alpha+\beta) \cdot \gamma=\alpha \cdot \gamma+\beta \cdot \gamma$$

$$K11) \quad \text{Sei } U \subseteq V \text{ mit } U \neq \emptyset \quad U \subseteq V \quad \text{d.h. } \forall \alpha_i, \beta_i \in U, \quad \forall \lambda \in \mathbb{F} \quad \lambda \alpha_i + \sum_{j=1}^n \beta_j = \sum_{j=1}^n \lambda \beta_j$$

Jeder Unterraum ist ein Vektorraum ($U \subseteq V$ prüfen)

$$\begin{aligned} & \text{Sei } U \text{ ein Vektorraum über } \mathbb{F}, \quad \alpha_1, \dots, \alpha_n \in U \\ & U \text{ heiess Unterraum} \Leftrightarrow U \text{ Vektorraum} \text{ und } \forall \alpha \in U, \quad \forall \lambda \in \mathbb{F} \quad \lambda \alpha \in U \\ & \text{Die Menge aller Linearkombinationen von } \alpha_1, \dots, \alpha_n \text{ heiess Lini. Komb. von } \alpha_1, \dots, \alpha_n \text{ oder linearer Komb. von } \alpha_1, \dots, \alpha_n \\ & \text{Span } \{\alpha_1, \dots, \alpha_n\} := \left\{ \sum_{k=1}^n \lambda_k \alpha_k \mid \lambda_1, \dots, \lambda_n \in \mathbb{F} \right\} \end{aligned}$$

Die Vektoren a_1, \dots, a_n heiessen Erzeugendensystem von $\text{Span}\{a_1, \dots, a_n\}$

Ein Span ist immer ein Unterraum
Ein Erzeugendensystem eines Raumes ist nicht eindeutig.
Def. von $a_1, \dots, a_n \in V$ sind linear unabhängig $\Leftrightarrow \lambda_1 a_1 + \dots + \lambda_n a_n = 0 \Rightarrow \lambda_1 = \dots = \lambda_n = 0$ sonst linear abhängig
Linear unabhängiges Erzeugendensystem eines Vektorraumes \Leftrightarrow es gibt n Elementen $a_1, \dots, a_n \in E$ einer $n \times n$ Matrix G für die G invertierbar $\Leftrightarrow G$ regulär
 \Rightarrow $\{a_1, \dots, a_n\}$ ist Basis von V $\Leftrightarrow \text{span}\{a_1, \dots, a_n\} = V$
 G ist Basis von E \Leftrightarrow G minimales Erzeugendensystem \Leftrightarrow $\text{span}\{G\} = E$

Rechenregeln bezüglich Basen, linearer Abhängigkeit
Seriern wir "an" gegeben

1) Schreibe $A = (a_1 | a_2 | \dots | a_n)$ 2) Gauss Elimination
3) $\text{rang}(A) = \dim(\text{span}\{a_1, \dots, a_n\})$ Wenn $\text{rang}(A)=n \Rightarrow a_1, \dots, a_n$ sind linear abhängig
oder richtig $\Leftrightarrow a_1, \dots, a_n$ sind linear abhängig

Thm 1: $\{a_1, \dots, a_n\} \subset V$ ist genau dann eine Basis von V , wenn sich jeder Vektor aus V eindeutig als Linearkombination a_1, \dots, a_n darstellen lässt
Def. Zwei Unterräume U und U' eines Vektorraums V sind orthogonal, wenn $\forall u \in U, u \perp v \in U' \Rightarrow u \cdot v = 0$

Def. Eine rechteckige Darstellung $x=c_1v_1+\dots+c_nv_n+d_1w_1+\dots+d_kw_k$ heißt dann die direkte Summe von $x \in V$ in $V=U \oplus U'$
Aufspaltung eines Vektorraumes in zwei sich orthogonalisierende Unterräume

Es sei $G = (g_1 | g_2 | \dots | g_m)$ die alte Basis und $G' = (g'_1 | g'_2 | \dots | g'_m)$ die neue Basis bezüglich V .
 $b_{lk} = \sum_{i=1}^m T_{ik} b_i, \quad k=1, \dots, m$ $T_{ik} = (T_{ik})$ heiess Transformationsmatrix

$B' = B T$
Sei $\tilde{g}_i = (g'_1 | \dots | g'_m)$ Koordinatenvektor aller Basis. Gesucht \tilde{g}'_i in neuer Basis

Es muss gelten $\sum_{i=1}^m \tilde{g}_i \cdot b_i = x = \sum_{i=1}^m \tilde{g}'_i \cdot b'_i$
Dann gilt $\tilde{g}'_i = \sum_{k=1}^m T_{ik} \tilde{g}_k \quad i=1, \dots, m \Rightarrow \tilde{g}'_i = T \tilde{g}_i$ oder $\tilde{g}'_i = T^{-1} \tilde{g}_i$

Spalten einer Transformationsmatrix T sind die neuen Basisvektoren aus G' bezüglich der alten Basis G dargestellt. $T_b'(b'_{lk} x)_i = [x]_i^{G'}$
 \Rightarrow Punkt aus G' auf G ausgedrückt. \rightarrow Transformation von G nach G'

Beispiel 1: Gegeben Basis B und B' : Gesucht: Transformationsmatrix T !
Transformation von B nach B'

$$\begin{array}{c|c} \text{Gegeben } B & B' \\ \hline 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{array} \rightarrow \begin{array}{c|c} I & T_{B'B} \\ \hline 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{array}$$

Lineare Abbildungen

$A = (a_{ij})_{m \times n}$ eine $m \times n$ Matrix

Kolumnenraum: Der von Kolumnen von A aufgespannte Unterraum $R(A) := \text{span}\{a_1, \dots, a_m\}$

Nullelement: Der Lösungsraum \mathcal{L} des homogenen LGS $Ax=0$, $N(A)$

als lineare Abbildung im $A = R(A)$ $\ker A = N(A)$

$Ax = b$ lösbar $\Leftrightarrow b$ liegt im Kolumnenraum von A

Lösung eindeutig $\Leftrightarrow N(A) = \{0\}$, homogenes System hat nur triviale Lösung

Der Rang einer $m \times n$ -Matrix A ist gleich:

der Anzahl Prozeßschritte bei Reduktion auf Zeilenstufenform

der Rang der linearen Abbildung $A: E^n \rightarrow E^m$ definiert als dim($\mathcal{L}(A)$)

dem Rang der Kolumnenstufenform des Kolumnenraums definiert als # lin. unabh. Kolumnenteilen

Bei Dimension des Kolumnenraums schreibt man $A = R(A) = E^m$

\Rightarrow Zeilenrang = Kolumnenrang, daher $\text{rang } A = \text{rang } A^T$

Für das quadratische Matrix $A \in E^{n \times n}$ sind folgende Aussagen äquivalent:

1) die n -Elemente $x_1, x_2 \in X$ ($x_1 + x_2 = 0$) erfüllen $Ax = 0$ 2) A ist invertierbar 3) die n -Elemente $y_1, y_2 \in Y$ ($y_1 + y_2 = 0$) erfüllen $A^T y = 0$

4) die n -Elemente $x_1, \dots, x_n \in X$ erfüllen $Ax = 0$ 5) die n -Elemente $y_1, \dots, y_n \in Y$ erfüllen $A^T y = 0$

6) $\text{rang } A = n$ 7) $\text{rang } A^T = n$ 8) A ist ein automorpheismus

9) A ist Transformationsmatrix einer Koordinatentransformation in E^n

10) A ist Transformationmatrix einer Komplexe Lösung zu $Ax = b$ berechnen \Rightarrow Zeilen

11) x berechnet; alle freien $= 0$ setzen. Gleichungssystem $Ax = b$ berechnen \Rightarrow Lösung

12) $b = \ker A = N(A)$ berechnen \Rightarrow Spezielle Lösung einer freien Variable = setzen

Erstes der Grundsatz \Rightarrow Zeilenstufenform. Dann immer eine spezielle Lösung finden und die anderen = 0 und so nachgeuntert alle Speziellen Lösungen finden.

Oberer Grundsatz nachwärts: Spaltenelimination, die Matrix A durch - und \leftrightarrow Spalten austauschen, so dass $\ker A = N(A)$ nur N_c ist, wo c ein lekter und freie wählbar Parameter

\Rightarrow $\text{rang } A = \text{rang } A^T$ \Rightarrow Koordinaten bezüglich alter Basis

\Rightarrow $\text{rang } A = \text{rang } A^T$ \Rightarrow Koordinaten bezüglich neuer Basis

\Rightarrow $\text{rang } A = \text{rang } A^T$ \Rightarrow $\text{rang } A = \text{rang } A^T$ \Rightarrow $\text{rang } A = \text{rang } A^T$

Es gilt: $\eta = A \xi$, $\xi = S^{-1} \eta$, $\eta' = S \xi'$, $\xi' = T \xi$

$B = S^{-1} AT$ \Rightarrow $A = SCT^{-1}$

Falls gilt $\xi = 0$ so auch $\eta = 0$. Dann $S = T$. Dann $B = T^{-1}AT = TBT^{-1}$

Wird der Übergang $A \mapsto B = T^{-1}AT$ wird Ähnlichkeitstransformation genannt.

Sei $F: X \rightarrow Y$ lineare Abbildung! Definition:

Menge $F(X)$ der Bildpunkte von X = Wertebereich/Bild von F ; $F(X) = \{F(x) | x \in X\}$

* injektiv, falls $F(x) = F(y) \Rightarrow x = y$ für alle $x, y \in X$

* surjektiv, falls $\forall y \in Y \exists x \in X$ mit $F(x) = y$

* bijektiv, falls F injektiv und surjektiv, dann gilt:

Gegeben: Basis $B = \{b_1, \dots, b_m\}$ von X und Basis $C = \{c_1, \dots, c_n\}$ von Y

Ein eindeutiger linearer Abbildung von X auf Y heißt Isomorphismus

bei $X = Y$ so heißt sie Automorphismus

Ist $F: X \rightarrow Y$ ein Isomorphismus so ist die lineare Abbildung $F^{-1}: Y \rightarrow X$ Invert und kontraposition

Wid. des Isomorphismus F bezüglich festen Basis durch die Matrix A dargestellt. So

ist A regulär = invertierbar und $F \rightarrow A^{-1}$ dargestellt.

Gegeben: $x \in X$, $F(x) = b$ $\Leftrightarrow x \in F^{-1}(b)$ (Lösungsmenge von $Fx = b$)

Im $F = \{F(x) | x \in X\} \subseteq Y$ (Menge von b , für die $Ax = b$ lösbar)

Dimensionsformel: $\dim(X) = \dim(\ker F) + \dim(\text{im } F)$

$\Leftrightarrow \dim(\text{im } F) = \text{rang}(F) = \dim(X) - \dim(\ker F)$

Es gilt: i) $F: X \rightarrow Y$ injektiv $\Leftrightarrow \text{rang } F = \dim X$ (d.h. isomorph)

ii) $F: X \rightarrow Y$ surjektiv $\Leftrightarrow \text{rang } F = \dim Y = \dim X$ (d.h. surjektiv)

iii) $F: X \rightarrow Y$ bijektiv $\Leftrightarrow \text{rang } F = \dim X = \dim Y < \infty$ (d.h. bijektiv)

Für $F: X \rightarrow Y$ gilt: $\text{dim } Y \geq \dim X$ $\dim Y < \infty$ lineare Abbildung gilt

i) $\text{Rang } F, G \leq \min\{\dim F, \dim G\}$

ii) G injektiv $\Rightarrow \text{rang } GF = \dim F$

iii) F surjektiv $\Rightarrow \text{rang } GF = \dim G$

Skalarprodukt

- Norm kann allgemein definiert werden und nicht unbedingt mit Skalarprodukt definiert
- \rightarrow beliebige Funktion $f: V \rightarrow \mathbb{R}$, $x \mapsto \|x\|$
- Ebenso Skalarprodukt: beliebige Funktion von 2 Variablen $\langle \cdot, \cdot \rangle: V \times V \rightarrow \mathbb{R}$
- $\langle \cdot, \cdot \rangle$ ist ein Vektorraum mit Skalarprodukt. Ist V endlich dimensional und $\mathcal{E} \subset \mathbb{R}^n$: euklidischer/orthogonaler Vektorraum $\mathcal{E} = \mathbb{C}^n$ unilateraler Vektorraum.
- Länge eines Vektors x definiert durch $\|x\| := \sqrt{\langle x, x \rangle}$
- Eine Basis heißt orthogonal falls alle basisvektoren paarweise orthogonale sind

using Part II of the Venn diagram and

Metaprotokoll, bei dem es sich um ein "C" unter Vertragshaltung handelt.

definiert durch $\|x\| := \sqrt{x_1^2 + x_2^2}$
 falls alle Basisvektoren paarweise orthogonale sind

三

falls zusätzlich $\langle b_{\alpha}, b_{\beta} \rangle = 1$

$$k_{kl} = \begin{cases} 0 & k \neq l \\ 1 & k = l \end{cases} \Rightarrow \langle b_k | b_l \rangle = \delta_{kl} \quad (1) \quad (13)$$

"dimensionales Lektoriuum und Skalaproduct) $\mathcal{E}_{bi, \text{long}}$
"abasis" in \mathbb{R}^n

dargestellt: $x = \sum_{k=1}^n (b_k x)$

$$\sum_{k=1}^n b_k (\beta_k X) = \sum_{k=1}^n b_k \beta_k X = \left(\sum_{k=1}^n \beta_k b_k \right) X$$

Die von Projektionen auf Koordinatenachsen abgesetzten Werte
 $b_k := \langle b_k, x \rangle$ und $\eta := \langle b_k, y \rangle$, $k = 1, \dots, n$ gillt:

$H^2 = \langle n, n \rangle$ ist ein \mathbb{R} -Basisvektor des \mathbb{R} -algebraischen Skalarprodukts

Best left open in VSLI given clean (initial) boundary conditions in \mathbb{R}^d .

bereitstellbare Menge linear unabhängiger Vektoren $E_{\alpha_1}, \dots, E_{\alpha_r}$

$$T = q_4 - \sum_{i=1}^{k-1} q_i \geq q_k \geq \frac{1}{6k} = \frac{1}{6k}$$

$b_k := \sum_{j=1}^n a_j k^j$, $a_j \in \mathbb{K}$, $\|b_k\| = \sqrt{\sum_{j=1}^n |a_j|^2 k^{2j}}$

573 cm⁻¹, 1525 und 1561 cm⁻¹ eine Orthogonalbasis bilden.

is eine homophile Rasse, vorwiegend Städter, von jenseits anderer Rassen verschieden. Gesetzsgemäß.

$V = U \oplus U^\perp$ mit $U = \text{Span}\{b_1, b_2\}$ und $U^\perp = \text{Span}\{b_3\}$ \Rightarrow Summe orthogonaler Komplemente

$\delta \in \mathbb{R}$ mit $\delta > 0$ gibt es die von \mathcal{C} abhängigen Konstanten $R(\mathcal{C}) = r$ und $\alpha(\mathcal{C})$ so, dass für alle $x, y \in \mathcal{C}$ gilt:

$\text{c}(\text{V}_1) = n - r \leq \text{Linker Koeffizient dim}(M_{\text{V}_1})$

Def Eine Matrix A Elementen heit orthogonal $\Leftrightarrow A^T = A^{-1} \Rightarrow A^T = A^{-1}$
 \Rightarrow alle Zeilen und Spalten von A stehen paarweise orthogonale zueinander bzgl. Skalarprodukt
 $\Rightarrow A = A^T \Leftrightarrow A$ unitar $\Leftrightarrow A^H = A^{-1}$ unitar.
 Der Begriff der orthogonalen/unitalen Matrizen ist unabhngig vom Skalarprodukt
 Der Begriff der orthogonale/unitalen Vektoren ist abhngig vom Skalarprodukt des Vektorraums
 Orthogonale/unitare Basistransformationen sind Lnge und Winkel im Koordinatenraum:
 Beispiele $\vec{g} = T\vec{s}$ bzw. $\vec{g} = T^{-1}\vec{s}$ sowie in Koordinatenvektoren:
 $\langle \vec{g}_1, \vec{g}_2 \rangle = \langle \vec{s}_1, \vec{s}_2 \rangle = \langle \vec{s}_1, \vec{n} \rangle = \langle \vec{g}_1, \vec{n} \rangle$. Eine lineare Abbildung $F: X \rightarrow Y$ heißt orthogonal/unitar falls $\langle Fx, Fy \rangle = \langle x, y \rangle$.
 Seien x_1, y_1 orthogonale/unitare Vektoren und Skalarprodukt. Eine lineare Abbildung $F: X \rightarrow Y$ heißt orthogonal/unitar falls
 hat folgende Eigenschaften
 i) $\|F(x)\| = \|x\|$ lngentreuer (isometrisch)
 ii) $x \perp y \Rightarrow Fx \perp Fy$ Winkel treu
 iii) $\ker F = \{0\}$ injektiv
 iv) $\dim X = \dim Y \geq 2$ gilt i)-iii)
 v) F ist surjektiv
 vi) F ist Abbildungsraum A bzgl ONS in X und V in Y ist
 Surj. $F: X \rightarrow Y$ und $G: Y \rightarrow Z$ unter Isomorphismen eindimensionaler Untervektorräume
 dann ist auch $G \circ F: X \rightarrow Z$ ein Isomorphismus.
 Ist V -dimensionaler unitarer Vektorraum und ONS, so ist die Koordinatenabbildung
 $K: V \rightarrow C^n$ unitär \Leftrightarrow lineare Abbildungen $A: C^n \rightarrow C^n$ unitär
 Matrix $A \in C^{n \times n}$ unitär \Leftrightarrow linear beschrnt, wenn es ein $\rho > 0$ gibt und
 Lineare Abbildung $F: X \rightarrow Y$ heißt beschrnt, wenn es ein $\rho > 0$ gibt und
 $\|F(x)\| \leq \rho \|x\| \quad (\forall x \in X)$
 \Rightarrow Die Gesamtheit solcher linearen Abbildungen heit $L(X, Y)$. Es ist ein Vektorraum
 Induzierte Operatornorm $\|A\| = \|d(A, 0)\| \rightarrow \mathbb{R}$, $F \mapsto \|F\| := \sup_{x \neq 0} \frac{\|Fx\|}{\|x\|}$
 Induzierte Normen. Ist $X = \mathbb{E}^n$ sodass F durch quadratische Matrizen
 gegeben ist $\hat{g}: \mathbb{E}^n \rightarrow \mathbb{E}^m$
 $\|A\| := \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \|A\|_2 := \sqrt{\lambda_{\max}(A^H A)}$
 Spezialfall: $\|F\| = \sup_{\|x\|=1} \|Fx\| = \|A\|_2$
 \Rightarrow Operatormodul norm erfüllt die Normbedingungen positiv definit, hngen A -Abhngig
 ausserdem $\|G \circ F\| \leq \|G\| \|F\| \quad (A, F, G \in L(X, Y))$
 $\|Fx\| \leq \|F\| \|x\| \quad (\text{Kongruenz mit Vektorraum in } X, Y)$
 Analog für Matrizen
 Konditionszahl einer reellen Matrix A bezgl. gegebener Norm
 $\kappa(A) = \|A\| \|A^{-1}\|$

Methode der Kleinsten Quadrate

Eine lineare Abbildung $\rho: \mathbb{E}^m \rightarrow \mathbb{E}^n$ heit Projektion, falls $\rho^2 = \rho$.

Gilt zustzlich $\ker \rho^\perp$ im \mathbb{P} , lsst $N(\mathbb{P}) \perp \mathcal{R}(\rho)$

So lsst sie Orthogonalprojektion. Andernfalls schreibe Projektion.

Fr einen Projektior ρ sind folgende Aussagen quivalent:

- ρ ist orthogonaler Projektior (da $\text{im}(\mathbb{I}-\rho) = \ker \rho$, $\text{ker}(\mathbb{I}-\rho) = \text{im } \rho$)
- $\mathbb{I}-\rho = \rho^\perp$
- mit Gauss-Beispiel $\rho = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$ $\rho_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ $\rho_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ $\rho_0 = \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix}$ zu minimierende Funktion $f(x) = a \sin(x) + b \cos(y) + c$ sodass $\sum (f(x_i, y_i) - z_j)^2$ minimal ist.

Rang $A = n \leq m \Leftrightarrow$ Kolonnen der extra Matrix sind linear unabhngig $\Leftrightarrow A^\perp$ rekt. von $\mathcal{R}(A)$ auf den Kolumnenraum $\mathcal{R}(A^\perp)$ im \mathbb{P} . Die Orthogonalprojektion $\rho: \mathbb{E}^m \rightarrow \mathbb{P}$ auf den Kolumnenraum $\mathcal{R}(A)$ ist einer $m \times n$ -Matrix A mit $\text{rang } A = n \leq m \leq 15$ gegeben durch:

$$\rho_A := A(A^\perp A)^{-1} A^\perp$$

$$A^\perp A x = A^\perp y$$

$$\begin{pmatrix} 2 & 1 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 4 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 10 \end{pmatrix} \Rightarrow a=2, b=2, c=\frac{3}{2}$$

$\rho_A = Q Q^\perp$ in $Q \subseteq \mathbb{E}^m$ auf den Kolumnenraum $\mathcal{R}(A) = \text{im } Q$

Die Orthogonalprojektion $\rho: \mathbb{E}^m \rightarrow \mathbb{P}$ auf den Kolumnenraum $\mathcal{R}(A)$ ist gegeben durch:
oder mehrfach

$\rho_A = Q Q^\perp$
Somit gilt fr jedes $y \in \mathbb{E}^n$: $\rho_A y = Q Q^\perp y = \sum_{j=1}^n q_j \langle q_j, y \rangle$

Eigenschaft der Orthogonalprojektion $\|y - \rho_A y\| = \min \|y - p\|$ $\forall y \in \mathbb{E}^n$ fr ein y ausschalt der Projektionsebene (\mathbb{P} im \mathbb{P}) gilt,

der am nchsten bei y liegt Punkt in \mathbb{P} .

Orthogonalprojektion liefert Lsung des linearen Approximationproblems.

Gegeben: $A \in \mathbb{C}^{m \times n}$ und $b \in \mathbb{C}^m$ $\left\langle \cdot, \cdot \right\rangle$ Skalarprodukt

Gesucht: $Q \in \mathbb{C}^{m \times n}$ und $\rho = QR$

Seien a_1, \dots, a_n Spalten von A . Wre das Gram-Schmidt-Verfahren

bezuglich der gegebenen Skalarprodukte auf Spaltenvektoren an

$\rightarrow \{a_1, \dots, a_n\}$ Orthonormalbasis zu A

$\rightarrow (q_1 | \dots | q_n) = Q \in \mathbb{C}^{m \times n}$

Gegeben: $\rho = \begin{cases} 0 & i \neq j \\ \frac{1}{\sqrt{2}}(q_1 | q_2) & i=j \end{cases}$

$\rho = \begin{pmatrix} 0 & & & \\ & 0 & & \\ & & 0 & \\ & & & 1 \end{pmatrix}$

$R = (r_{ij})$ $1 \leq i, j \leq n \in \mathbb{C}$

$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

Anwendung: Lsung eines berbestimmten LGS

Gegeben: $A \in \mathbb{C}^{m \times n}$ in \mathbb{P} $x \in \mathbb{Y}$

Gesucht: Nherungslzung x_1 , sodass der Residuenvektor $\|r\|$ minimal ist, wenn $\|x-x_1\| \geq \|x-x_2\|$

Da $\ker A = \mathbb{C}^3$

$\Rightarrow x = (A^\perp A)^{-1} A^\perp y$ oder $A^\perp A x = A^\perp y \Rightarrow A^\perp (y - Ax) = 0$

mit Gauss-Beispiel $\rho = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$ $\rho_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ $\rho_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ $\rho_0 = \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix}$ zu minimierende Funktion $f(x) = a \sin(x) + b \cos(y) + c$ sodass $\sum (f(x_i, y_i) - z_j)^2$ minimal ist.

Normalenvektor ρ^\perp im \mathbb{P}

$\Rightarrow A = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ $x = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ $y = \begin{pmatrix} 2 \\ 4 \\ 3 \end{pmatrix}$

$A^\perp A x = A^\perp y$

$$\begin{pmatrix} 2 & 1 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 4 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 10 \end{pmatrix} \Rightarrow a=2, b=2, c=\frac{3}{2}$$

Gegeben: $\lambda = (a_1, \dots, a_n) \in \mathbb{C}^{n \times n}$ $q_i := \frac{a_i}{\|a_i\|}$

$\tilde{q}_i := a_i - q_1 \langle q_1, a_i \rangle$ ($i=2, \dots, n$)

Whle $p \geq k$ mit $\|\tilde{q}_p\| \neq 0$ und verfusche Kolonnen p und k :

$q_k := \frac{\tilde{q}_k}{\|\tilde{q}_k\|}$

$\tilde{q}_i := \tilde{q}_i - q_k \langle q_k, \tilde{q}_i \rangle$ ($i=k+1, \dots, n$)

$\tilde{q}_p := \tilde{q}_p - q_k \langle q_k, \tilde{q}_p \rangle = 0$, so $\text{rang } A = k$ und man ist fertig.

$A = \begin{pmatrix} 0 & a_2 & a_3 \\ a_2 & 0 & a_1 \\ a_3 & a_1 & 0 \end{pmatrix}$ Zeigt, dass A nur dann diagonalisierbar ist, falls $a = b = c = 0$

Dann ist A diagonalisierbar, falls $a, b, c \neq 0$ und a, b, c linear unabhngig

in Linear faktoren. $(A^\perp A) = \begin{pmatrix} -a & 0 & 0 \\ 0 & -b & 0 \\ 0 & 0 & -c \end{pmatrix}$

spaltenweise bestimmt $\tilde{q}_1, \dots, \tilde{q}_3$ da die Eigenwerte zu den Eigenvektoren $\tilde{q}_1, \dots, \tilde{q}_3$ orthogonal voneinander sind und die Eigenvektoren $\tilde{q}_1, \dots, \tilde{q}_3$ linear unabhngig

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

$\tilde{q}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ $\tilde{q}_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$ $\tilde{q}_3 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$

Eigenwerte und Eigenvektoren

Abbildung $\det: \mathbb{C}^{n \times n} \rightarrow \mathbb{C}$
 $A \mapsto \det(A) = \sum_{\substack{i=1 \\ p \in S_n}} \text{sign } p \prod_{j=1}^n a_{i,j} p(j)$

Berechnung:

$$n=1: \det(a_{11}) = a_{11}$$

$$n=2: \det \begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$$

$$n=3: \det \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = aei + bfg + cdh - cei - afh - bdi$$

Allgemein: $n \times n$: Anwendung des Gauß-Algo auf A mit k Zeilenumtauschungen

$$\det(A) = (-1)^k \cdot \det(A') = (-1)^k \cdot \prod_{i=1}^n A'_{ii} \Rightarrow \text{Produkt der Diagonalelemente}$$

Eigenschaften:

- (i) \det ist eine lineare Funktion jeder einzelnen Zeile: $\forall i \in \mathbb{N}, t \in \mathbb{C}^{1 \times n}, \exists \lambda_i \in \mathbb{C}^{1 \times n}$
- (ii) die Determinante ist alternierend: Wenden zwei Zeilen vertausch, wechselt das Vorzeichen
- (iii) Determinante ist normiert: $\det(I) = 1$
- (iv) Hat A eine Nullzeile oder Nullspalte, so ist $\det(A) = 0$

$$\forall \lambda \in \mathbb{C}: \det(\lambda \cdot A) = \lambda^n \cdot \det(A)$$

v) A hat linear abhängige Zeilen $\Leftrightarrow \det A = 0$

vi) Abhäng von den Vektoren einer Zeile zu einer anderen ändert sich det nicht.

vii) Ist A eine diagonalisierbare Matrix ist $\det A$ das Produkt der Diagonalelemente

viii) $\det A \neq 0 \Leftrightarrow A$ ist regular \Leftrightarrow bigekr

$\det(AB) = \det A \cdot \det B$ aber $\det(A+B) \neq \det A + \det B$

$$\Leftrightarrow \det(A^k) = \det(A)^k$$

A regular: $\det(A^{-1}) = \frac{1}{\det A}$

$$\det(A^N) = \det A^N = \det(A) \cdot \det(A^1) \cdot \det(A^2) = \det(A)^N = \det(A)^{\det A}$$

Grill auch für diese Blockdiagonalmatrix

Eigenschaften (i)-(iii) sind charakteristisch für die Determinante

- 1) Beobachte $\det(A) = \det(A - \lambda I)$
 - 2) $R_A(\lambda) = 0 \rightarrow n$ Nullstellen $\lambda_1, \dots, \lambda_n \Rightarrow \det(A - \lambda I) = 0$
 - 3) Für jedes verschiedenen Eigenwert λ_k :
- Bestimme die Basis von $E_{\lambda_k}(A) = \ker(A - \lambda_k I)$
 Anwendbar von Gauß → Zeilenstufenform, wobei von den $n-r$ freien Parametern immer einer $\neq 0$ und die Restlichen $= 0$
- 4) Die Menge der Eigenwerte ist $\text{Span} \{ \text{Basis von } E_{\lambda_k}(A) \}_{k=1}^n \setminus \{0\}$
- A singular $\Leftrightarrow 0 \in \text{Span} \{ \text{Basis von } E_{\lambda_k}(A) \}_{k=1}^n$

Spektralzerlegung

Es existiert eine Spektralzerlegung $\Leftrightarrow \exists V \in \mathbb{C}^{n \times n}$ mit V^{-1} regulär und $A = V \Lambda V^{-1}$, wobei Λ diagonal ist.

\Leftrightarrow Eigenvektoren von A bilden eine Basis von \mathbb{C}^n (lin. unabh. Elbn.)

\Leftrightarrow Alle Eigenvektoren der gleichen geometrische und algebraische Vielfachheit.

\rightarrow Eine Diagonalisierung ist ein Basiswechsel und Transformationsmatrizen V

$V^{-1} \rightarrow$ Umkehrtransformation

Bei einer Umkehrtransformation bleibt folgendes erhalten:

Eigenwerte und deren algebraische geometrische Vielfachheit

• charakteristisches Polynom

• Rang, Detr., char. Polynom

• aber nicht die Eigenvektoren

Gegeben: Diagonalsierbare Matrix $A \in \mathbb{C}^{n \times n}$

Gesucht: $V \in \mathbb{C}^{n \times n}$ regulär, Λ diagonal sd $A = V \Lambda V^{-1}$

• Finde die Eigenwerte $\lambda_1, \dots, \lambda_n$, sowie die zugehörigen Eigenvektoren v_1, \dots, v_n . Die Eigenvektoren müssen linear unabhängig sein.

Rez 1) Test $A = V \Lambda V^{-1}$

Rez 2) Zeile i : $\sum_{j=1}^n v_{ij} \lambda_j = 0$

Rez 3) $V^{-1} = \begin{pmatrix} 1 & & & \\ 0 & \ddots & & \\ & & 1 & \\ & & & 0 \end{pmatrix}$

Rez 4) V^{-1} berechnen

Rez 5) Zeile i : $\sum_{j=1}^n v_{ij} \lambda_j = 0 \Rightarrow A \cdot \begin{pmatrix} v_{1i} & \dots & v_{ni} \end{pmatrix} = 0$

Zerlegung der diagonalisierten Matrix A in Summe von Rang-1-Matrizen:

$\Rightarrow V$ und V^{-1} in Kolumnen-/Spaltenvektoren zerlegen:

$V = (v_1 | \dots | v_n)$ $V^{-1} = \begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n1} & \dots & v_{nn} \end{pmatrix} \Rightarrow A = \sum_{k=1}^n \lambda_k v_k v_k^T$

• Dabei gilt: $v_k = \lambda_k v_k$ und $v_k^T A = \lambda_k v_k^T$ (v_k ist linear unabhängig von A)

Die Spektralzerlegung einer Matrix $A \in \mathbb{C}^{n \times n}$ ist gleich der Wurzel aus dem größten Eigenwert von $A^H A$

$\|A\|_2 = \max_{\mathbb{C}^n} \|\mathbf{x}\|_2$, wobei \mathbf{x} ist Eigenvektor von $A^H A$

Für A hermitisch gilt: $\|A\|_2 = \max_{\mathbb{C}^n} |\langle A \mathbf{x}, \mathbf{x} \rangle|$, wobei \mathbf{x} ist Eigenvektor von A

Für A nichthermitisch gilt: $\|A\|_2 = \max_{\mathbb{C}^n} \sqrt{\langle A \mathbf{x}, \mathbf{x} \rangle}$, wobei \mathbf{x} ist Eigenvektor von $A^H A$

Definition Spektralnorm: $\|A\|_2 = \sqrt{\max_{\mathbb{C}^n} \langle A \mathbf{x}, \mathbf{x} \rangle}$

Somit gilt für die Konditionszahl $K_2(A) = \|A\|_2 / \|A^{-1}\|_2$

$K_2(A) = \frac{\max_{\mathbb{C}^n} \|\mathbf{x}\|_2, \text{ w. EV von } A^H A}{\min_{\mathbb{C}^n} \|\mathbf{x}\|_2, \text{ w. EV von } A^H A}$

Fall: $K_2(A) = \frac{\min_{\mathbb{C}^n} \|\mathbf{x}\|_2, \text{ w. EV von } A^H A}{\max_{\mathbb{C}^n} \|\mathbf{x}\|_2, \text{ w. EV von } A^H A}$

ist singular, aber keine Nullzeile, so ist $K_2(A) = \infty$

ist singulär, wird $K_2(A)$ sehr gross $\rightarrow A$ ist schlecht konditioniert.

Indirekte Operatormethode $\|F(x)\|_2 = \max_{\mathbb{C}^n} \langle F(x), \mathbf{x} \rangle$

Nutzliche Normen

Sei A quadratische $n \times n$ -Matrix

- A ist invertierbar $\rightarrow \det(A) = \det(A^{-1}) = 1$
- A ist Singulär $\rightarrow \det(A) = 0$
- A ist symmetrisch $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- A ist antisymmetrisch $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = -\langle \mathbf{x}, A \mathbf{x} \rangle$
- A ist regulär $\rightarrow A^{-1} = A^{-1}$
- A ist Singulär $\rightarrow A^{-1}$ ist Singulär
- A ist Eigenwert \rightarrow alle Eigenwerte sind selbstdual $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- $A \mathbf{x} = 0$ ist nicht trivial \rightarrow $A \mathbf{x} = 0$ kann eine Lösung $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle = 0$
- $A \in \mathbb{R}^{n \times n}$ ist symmetrisch \rightarrow obwohl singular mit $S \in \mathbb{R}^{n \times n}$ und S^{-1} regulär, A ist singulär
- A ist regulär $\rightarrow A^{-1} = A^{-1}$
- A ist Singulär $\rightarrow A^{-1}$ ist Singulär
- A ist Eigenwert \rightarrow alle Eigenwerte sind selbstdual $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- $A \mathbf{x} = 0$ ist nicht trivial \rightarrow $A \mathbf{x} = 0$ kann eine Lösung $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle = 0$
- $A \in \mathbb{R}^{n \times n}$ ist antisymmetrisch \rightarrow obwohl singular mit $S \in \mathbb{R}^{n \times n}$ und S^{-1} regulär, A ist singulär
- A ist regulär $\rightarrow A^{-1} = A^{-1}$
- A ist Singulär $\rightarrow A^{-1}$ ist Singulär
- A ist Eigenwert \rightarrow alle Eigenwerte sind selbstdual $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- $A \mathbf{x} = 0$ ist nicht trivial \rightarrow $A \mathbf{x} = 0$ kann eine Lösung $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle = 0$
- $A \in \mathbb{C}^{n \times n}$ ist antisymmetrisch \rightarrow obwohl singular mit $S \in \mathbb{C}^{n \times n}$ und S^{-1} regulär, A ist singulär
- A ist regulär $\rightarrow A^{-1} = A^{-1}$
- A ist Singulär $\rightarrow A^{-1}$ ist Singulär
- A ist Eigenwert \rightarrow alle Eigenwerte sind selbstdual $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- $A \mathbf{x} = 0$ ist nicht trivial \rightarrow $A \mathbf{x} = 0$ kann eine Lösung $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle = 0$
- $A \in \mathbb{C}^{n \times n}$ ist symmetrisch \rightarrow obwohl singular mit $S \in \mathbb{C}^{n \times n}$ und S^{-1} regulär, A ist singulär
- A ist regulär $\rightarrow A^{-1} = A^{-1}$
- A ist Singulär $\rightarrow A^{-1}$ ist Singulär
- A ist Eigenwert \rightarrow alle Eigenwerte sind selbstdual $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- $A \mathbf{x} = 0$ ist nicht trivial \rightarrow $A \mathbf{x} = 0$ kann eine Lösung $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle = 0$
- $A \in \mathbb{C}^{n \times n}$ ist antisymmetrisch \rightarrow obwohl singular mit $S \in \mathbb{C}^{n \times n}$ und S^{-1} regulär, A ist singulär
- A ist regulär $\rightarrow A^{-1} = A^{-1}$
- A ist Singulär $\rightarrow A^{-1}$ ist Singulär
- A ist Eigenwert \rightarrow alle Eigenwerte sind selbstdual $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle$
- $A \mathbf{x} = 0$ ist nicht trivial \rightarrow $A \mathbf{x} = 0$ kann eine Lösung $\rightarrow \langle A \mathbf{x}, \mathbf{x} \rangle = \langle \mathbf{x}, A \mathbf{x} \rangle = 0$

Hausaufgaben

Viele Werte ist invertierbar?

Gegeben $A = \begin{pmatrix} a & b & c \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}$

$$A^{-1} = \begin{pmatrix} 1 & -b & -c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{falls } a \neq 0$$

$$\text{Gegeben } A = \begin{pmatrix} a & b & 0 & 1 & 0 & -c \\ 0 & 1 & 0 & 0 & 1 & -d \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} = I$$

$$\text{Gegeben } A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \quad \text{Zeigt das } g, h: (A-tI)(A-t^2I) = 0$$

$$\text{Gegeben } A = \begin{pmatrix} t & 0 & t \\ 0 & t & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{und } A^2 + I = \begin{pmatrix} t^2 & 0 & t^2 \\ 0 & t^2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow (A-tI)(A-t^2I) = 0$$

$$\text{Berechnen } A-I\tilde{L} = \begin{pmatrix} t & 0 & t \\ 0 & t & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad \text{Ist } A \text{ invertierbar?} \Rightarrow t \neq 0$$

$$\text{Zeigt } A \text{ invertierbar?} \Rightarrow t \neq 0$$

$$\text{Wann } A \text{ invertierbar ist, existiert keine Matrix } B \neq 0, \text{ so dass } AB = 0$$

$$\text{Falls } t=0 \text{ folgt aus (2), dass } A(A^2 + I) = 0 \text{ aber } A^2 + I \text{ ist nie gleich } 0 \Rightarrow t \neq 0$$

$$\text{Zeigt } A \text{ invertierbar ist, durch + herleit: } A(\frac{1}{t}(A^2 + I) - I) = I \text{ kann es auch als}$$

$$\text{Also ist } \frac{1}{t}(A^2 + I) - I \text{ eine Rechtsinverse von } A, \text{ man kann es auch als } (\frac{1}{t}(A^2 + I) - I)A = I \text{ schreiben } \Rightarrow A \text{ ist invertierbar mit } A^{-1} = \frac{1}{t}(A^2 + I) - I.$$

$$\text{Falls } A \text{ invertierbar ist, ist } A \text{ auch } A^T \text{ invertierbar?}$$

$$\text{Zeigt } f = T = (A^T - A)^T \Rightarrow f = A^T - A \Rightarrow A^{-1} | A^T - A | f^{-1} = 0 \Rightarrow A^T = A$$

$$f = A^T - A \Rightarrow f = (A^T - A)^T = A^T - A \Rightarrow A^T = A$$

$$\text{Somit gilt } A^T = A \Rightarrow A \text{ ist symmetrisch}$$

$$\text{Bestimme die Koeffizienten von } x_1 = 6x_1 - 5x_2 - 4x_3 \quad x_1 = 2 \\ \text{S.t.: } x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix} \Rightarrow A_1 = 3 \quad A_2 = 2 \quad A_3 = 2 \Rightarrow x = \begin{pmatrix} 3 \\ 2 \\ 2 \end{pmatrix}$$

$$\text{Oder mit } \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \Rightarrow x = \begin{pmatrix} 3 \\ 2 \\ 2 \end{pmatrix}$$

$$\text{Organize } A \text{ zu einer Basis von } V. \text{ Sei } b_1, b_2, b_3 \text{ ein Basis aller reellen Polynome } P_k(x) = x^k + \text{rest}, \text{ dann ist } b_3(x) = 3x^2 + x. \text{ Bestehe die Standardbasis } S \text{ aus den Sätzen } b_1, b_2, b_3 \text{ und } b_3(x).$$

$$T = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \sim \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{die Koeffizienten stehen unter diagonalen Elementen}$$

$$\text{Finde den Koordinatenvektor von } b_3(x) = x^2 + 2x + 3 \text{ in der Basis } S.$$

$$\text{Bestimme } T^{-1} = \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

$$\text{Bestimme } T^{-1} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 63 \\ 1 & -3 & -1 \\ 4 & 1 & 0 \end{pmatrix}$$

MC Hausgaben

- Seien ψ_1, ψ_2 endlich dimensionale "Metamorphe" über einem Körper K . Sei $F: V \rightarrow W$ eine lineare Abbildung und (v_1, \dots, v_k) eine Basis von $\text{Im } F(v_i)$ in W .
Dann kann man darüber hinaus angeben, ob F surjektiv ist. X stellt diese und andere Fragen für ψ_1, ψ_2 an F her. Es gelten dann folgende Aussagen:
Gibst eine Basis $\{w_1, \dots, w_n\}$ von $\text{Kern } F$, dann ist F surjektiv.
Sei $A \in R_{\text{min}}$ und $F: V \rightarrow W$ linear. Dann ist F surjektiv, wenn $\text{Kern } F = A$.
Durch $\text{Im } F = \{F(v) \mid v \in V\}$ erhalten wir $\text{Im } F = \{A \cdot \text{Im } K(v) \mid v \in V\}$.
Der Unterraum $\{A \cdot \text{Im } K(v) \mid v \in V\}$ ist bezüglich der Basis $B = \{B_1, \dots, B_m\}$ ein Unterraum von W .
Die Vektoren B_1, \dots, B_m sind linear unabhängig, da $\text{Im } K(v)$ linear unabhängig ist.

Prüfungsauflösung

Gegeben Matrix $A = \begin{pmatrix} 3 & 5 \\ -2 & -6 & -4 \\ 3 & 10 & 10 \end{pmatrix}$ Bestimme LR-Zerlegung $PA = LR$

$$PA = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 1 & 3 & 5 \\ 0 & 1 & 0 & 0 & 1 & -5 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\text{Gegeben } L = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 3 & 1 & 0 \end{pmatrix}, R = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 0 & 3 \\ 0 & 0 & 1 \end{pmatrix}, P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix}$$

$$PAx = (LR)x \Rightarrow Lc = Pb \Rightarrow b = \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix} \quad L = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 3 & 1 & 0 \end{pmatrix} \Rightarrow c = \begin{pmatrix} 2 \\ 3 \\ 0 \end{pmatrix}$$

$$Pcc: 0 \begin{vmatrix} 1 & 0 \\ 1 & -2 \\ 0 & 3 \end{vmatrix} \Rightarrow x = \begin{pmatrix} -\frac{1}{2} \\ 1 \\ 0 \end{pmatrix}$$

$$F(x+y) = \lambda F(x) + F(y)$$

$$F(\lambda x + y) = (\lambda x + y)M = \lambda xM + yM = \lambda F(x) + F(y)$$

Berechne die Abhängigkeitsmatrix F bezüglich der Basis

$$\lambda = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

$$b = \{(0,0), (0,1), (1,0), (0,0)\}$$

$$F(b_1) = b_1 M = \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} = 2b_1 + b_2$$

$$F(b_2) = b_2 M = \begin{pmatrix} 0 & 0 \\ 1 & 2 \end{pmatrix} = b_1 + 2b_2$$

$$F(b_3) = b_3 M = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} = b_3 + 2b_4$$

$$F(b_4) = b_4 M = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = 2b_3 + b_4$$

Damit ist die Abhängigkeitsmatrix von F bezüglich der Basis \mathcal{B}

$$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$F(\mathcal{B}) = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad |A = -\lambda I|$$

Sei V ein linearer Unterraum von K^4 . Dann ist V ein linearer Unterraum von V , wenn er abgeschlossen ist. und dass $U \neq \emptyset$

Addition und Skalar-Multiplikation abgeschlossen ist.

Mit $A, B \in U$ und $x \in K$

$$U: \quad U \subseteq A + B = -(A + B)^T = -(A + B)^T \Rightarrow A + B \in U$$

$$U: \quad A + B = -A - B = -A^T - B^T \Rightarrow A^T = -A, B^T = -B \Rightarrow A \in U, B \in U \Rightarrow A + B \in U$$

$$U: \quad \lambda A = \lambda(-A^T) = -\lambda A^T = -A^T \Rightarrow A^T = -A \Rightarrow \lambda A \in U$$

$$U: \quad \lambda A = -A^T \Rightarrow \lambda A^T = A^T \Rightarrow \lambda^2 A^T = A^T \Rightarrow \lambda^2 = 1 \Rightarrow \lambda = \pm 1 \Rightarrow \lambda A \in U$$

$$U: \quad \lambda A = -A^T \Rightarrow \lambda A^T = A^T \Rightarrow \lambda^2 A^T = A^T \Rightarrow \lambda^2 = 1 \Rightarrow \lambda = \pm 1 \Rightarrow \lambda A \in U$$

$$U: \quad \lambda A = -A^T \Rightarrow \lambda A^T = A^T \Rightarrow \lambda^2 A^T = A^T \Rightarrow \lambda^2 = 1 \Rightarrow \lambda = \pm 1 \Rightarrow \lambda A \in U$$

Seien $A, B \in \mathbb{R}^{n,n}$ zwei quadratische Matrizen. Zeigen Sie, dass wenn $A \in U$ ein Eigenwert der Matrix AB ist, auch ein Eigenwert der Matrix BA ist. Wenn $\lambda \in \mathbb{C}$ ein Eigenwert von AB ist gilt, dass es ein $x \in \mathbb{C}^n \setminus \{0\}$ gibt, das $ABx = \lambda x$ erfüllt. Wir müssen zeigen, dass es auch ein $y \in \mathbb{C}^n \setminus \{0\}$ gibt so dass $BAy = \lambda y$. Falls $y \neq 0$ und $x \neq 0 \Rightarrow BAy = \lambda y$ ist. Wir wollen $y = Bx$.

Falls $y = 0 \Rightarrow Bx = 0 \Rightarrow B(ABx) = B(\lambda x) = \lambda Bx = \lambda y$ $\Rightarrow \lambda Bx = \lambda y \Rightarrow \lambda Bx = \lambda y \Rightarrow Bx = y$ ist. $\Rightarrow \lambda Bx = \lambda y \Rightarrow \lambda Bx = \lambda Bx \Rightarrow \lambda = 1$ ist erfüllt.

Es sei $A \in \mathbb{R}^{2 \times 2}$, so dass $\text{Spur}(A) = 5, \det(A) = 6$. Was sind die Eigenwerte und die Eigenvektoren einer Matrix A sind gleich den Eigenwerten von A .

Das charakteristische Polynom von A ist:

$$\det((\lambda - \lambda_1)(\lambda - \lambda_2)) = (\lambda - \lambda_1)(\lambda - \lambda_2) - bc = \lambda^2 - (\lambda_1 + \lambda_2)\lambda + \lambda_1\lambda_2 + bc = \lambda^2 - 5\lambda + 6 = 0 \Rightarrow \lambda_1 = 3 \quad \text{und} \quad \lambda_2 = 2 \quad \text{ad} - bc = \lambda_1\lambda_2 + bc$$

Da $\det(A) = 6$ und $\det(A - \lambda_1 I) = (\lambda - \lambda_1)(\lambda - \lambda_2)$ ist A diagonalisierbar über \mathbb{R} . $\lambda = 3$ ist erfüllt.

Für welche Werte von $a, b, c \in \mathbb{R}$ ist A schief-symmetrisch ($A = -A^T$) und hat nur reelle Eigenwerte?

Die Matrix A ist schief-symmetrisch, falls $a = 0$ und $b = -c$. Eine Matrix A darf nicht alle Eigenwerte rein imaginär oder gleich 0 sein. Eine Matrix A ist diagonalisierbar über \mathbb{R} falls sie rein reelle Eigenwerte hat.

Finden eine orthogonale Basis des Unterraumes $S = \text{Span}\{v_1, v_2\}$ von \mathbb{R}^3 mit $v_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, v_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$

Dann ist T eine lineare Abbildung von \mathbb{R}^3 auf \mathbb{R}^3 . Finden wir einen Homogenen und additiven Vektor v_3 von S .

$v_3 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = q_1 v_1 + q_2 v_2$

Finden wir einen Homogenen und additiven Vektor v_3 von S .

$T(\lambda X + Y) = T(\lambda(X + Y)) = T(\lambda X + \lambda Y) = \lambda(T(X) + T(Y)) = \lambda(T(X) + T(Y))$

Finden wir eine Basis von $\ker T$. Da $T(\lambda x) = 0 \Leftrightarrow \lambda x = 0 \Leftrightarrow x = 0$ ist $\ker T = \{0\}$

Was ist die Dimension von $\ker T$? Es ist die Dimension von $\ker T + \dim \ker T = 3$

$\dim V = 4 = \dim \ker T + \dim \text{Im } T \Rightarrow \dim \text{Im } T = 3$

$\Rightarrow \text{dim } \text{Im } T = 3$ und $\text{dim } \ker T = 1$

$\Rightarrow \dim \ker T = 1$ und $\dim \text{Im } T = 3$

$\Rightarrow \dim \ker T = 1$ und $\dim \text{Im } T = 3$

MC PrüfungsAufgaben

Falsch oder falsch?
 Wenn $AB = B$, dann ist A die Einheitsmatrix X. Wenn sowohl AB als auch BA definiert sind, dann sind sowohl A als auch B quadratisch X. Wenn sowohl AB als auch BA definiert sind, dann sind beide Matrixprodukte quadratisch.

Grenzen die erste und dritte Spalte von B "abwischen", so gilt auch für die erste und dritte Spalte von AB. J. Wenn die 1. und 3. Zeile von B gleich sind, dann auch die 1. und 3. Zeile von AB. X Wenn die Erste und dritte Zeile von A gleich sind, dann auch die 1. und 3. Zeile des Matrixprodukts ABC X.

Die algebraische Vielfachheit einer Eigenvektor von A ist immer 1.

Die algebraische Vielfachheit einer Eigenvektor von A ist immer 1.

$(AB)^2 = A^2 B^2 X$

Sei V ein n-dimensionaler Vektorraum und $V \rightarrow V$ die Identitätsabbildung und BC

Basen von V. Dann ist die Determinante der Matrixdarstellung von id bezüglich der Basen A und C ungleich 0. J. Seien A und C invertierbar X Seien AB $\in \mathbb{R}^{3x2}$ $AB = I_3 \in \mathbb{R}^3$. Dann sind A und C invertierbar X Seien A $\in \mathbb{R}^{2x2}$ invertierbar dann ist auch

AB invertierbar X Seien A $\in \mathbb{R}^{2x2}$ und B $\in \mathbb{R}^{2x2}$ dann ist $\det(AB) = 0$ ✓
 $A \in \mathbb{R}^{3x3}$ und $\det(A) = 5$ invertierbar X $B = (a_{ij}) \in \mathbb{R}^{3x2}$: Richtig Falsch? ✓
 Seien AB reelle nxn Matrizen und $AB = BA$: Richtig Falsch? ✓

$\det(AB) = \det(-BA) \vee \det(A) \det(B) = -\det(A) \det(B)$ ✓
 Entweder A oder B muss singulär sein X
 Ein O-Determinante haben X A und B müssen singulär sein X

B muss eine Schleife von Lösungen X $ABx = c$ kann mehrere, keine und genau eine Lösungen haben wenn $c \in \mathbb{R}^n$ und $c \neq 0$ ✓
 Es gilt zumindest, dass $A = 0$ oder $B = 0$ ist. X

Es sei A eine reelle 3x3 Matrix mit den Eigenwerten $\lambda_1, \lambda_2, \lambda_3$. Richtig falsch? X
 Falls $\lambda_1, \lambda_2, \lambda_3$ alle verschieden sind dann ist A diagonalisierbar X
 Falls $\lambda_1, \lambda_2, \lambda_3$ verschieden, aber X
 A diagonalisierbar ist, dann haben A $\lambda_1, \lambda_2, \lambda_3$ verschiedene Werte X
 A ist diagonalisierbar wenn gilt, dass A die Eigenvektoren hat. X

Falls $\lambda_1 = 2, \lambda_2 = -2, \lambda_3 = 1$ und $B = \beta^2 - 3A^2$ dann ist B diagonalisierbar X
 Falls $AB = PO$ und O eine Diagonalmatrix ist, dann sind die Spalten von P Eigenvektoren von A. X

Korrekt oder falsch? X

Sei $T: V \rightarrow W$ eine lineare Abbildung und $y_1, \dots, y_k \in T(V), T(y_1), \dots, T(y_k) \in W$
 Dann ist die Menge $E = \{y_1, \dots, y_k\}$ auch linear unabhängig. X

Sei $T: V \rightarrow W$ eine lineare Abbildung und $y_1, \dots, y_k \in V$ und $E = \{y_1, \dots, y_k\}$ linear unabhängig. X

Sei $T: V \rightarrow W$ eine lineare Abbildung und $y_1, \dots, y_k \in W$, wobei y_1, \dots, y_k linear unabhängig. X

Seien $y_1, \dots, y_k \in \mathbb{R}^2$. Dann ist $\{T(x) | x \in \mathbb{R}^2\} = Bx + b$ ein Unterraum von \mathbb{R}^2 auch linear unabhängig. X

Sei V ein n-dimensionalter Vektorraum und $S = \{v_1, \dots, v_n\} \subset V$ und $T: V \rightarrow W$ eine lineare Abbildung und $v_1, \dots, v_n \in V$ wobei v_1, \dots, v_n linear unabhängig. X

Sei $T: V \rightarrow W$ eine lineare Abbildung und $y_1, \dots, y_k \in W$, wobei y_1, \dots, y_k linear unabhängig. X

Seien $y_1, \dots, y_k \in \mathbb{R}^2$. Die Menge S ist gegeben, dann linear unabhängig wenn $V = \text{Span}\{S\}$ ist. X

Seien T und S zwei lineare Abbildungen von $\mathbb{R}^n \rightarrow \mathbb{R}^m$ und y_1, \dots, y_k ist $\text{Span}\{T(y_1), \dots, T(y_k)\} = S(y_1), \dots, S(y_k) = S(\{y_1, \dots, y_k\})$ dann gilt auch, dass

$T(M) = T(\{y_1, \dots, y_k\}) = S(\{y_1, \dots, y_k\}) = S(M)$ ✓

Sei $A \in \mathbb{R}^{n,m}$ ein Skalarprodukt und B eine Orthogonalbasis von \mathbb{R}^n . Sei $N_A: V \rightarrow V$ ein Skalarprodukt und $N_B: W \rightarrow W$ ein Skalarprodukt. X

Sei N_A die Koordinatenabbildung bezüglich Basis B . Was ist richtig? X

Sei N_B ein Isomorphismus. Was ist richtig? X

dim $\text{Im } N_A = m$

Seien $x_1, \dots, x_m \in \mathbb{R}^n$ und $y_1, \dots, y_m \in \mathbb{R}^m$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

$\sum_{i=1}^m a_i x_i = \sum_{i=1}^m b_i y_i$ für $a_i, b_i \in \mathbb{R}$

Algorithm and Data Structures - Lecture

Prof. Markus Püschel and Prof. David Steurer

Contents

0 Einführung	5
0.1 Organisation	5
1 Graphentheorie	5
1.1 Graph als Mathematisches Objekt	5
1.2 Besondere Graphen	6
1.3 Summe der Knotengrade	6
1.4 Knotenfärbung	6
1.5 Färbbarkeit und Knotengrade	6
1.6 Vollständige Induktion	7
1.7 Beweis: Graphen ohne ungerade Kreise sind bipartit	7
1.8 Gerichteter Graph	8
1.9 Wege und Pfade	8
1.10 Erreichbarkeit und Distanz	8
1.11 Topologische Sortierung	8
2 Algorithmen	9
2.1 Gierige-Knotenfärbung(G)	9
2.2 Bipartition(G)	9
2.3 Effizienzbegriff von Algorithmen	10
2.4 StarFinden(G)	10
2.5 Allgemeines Berechnungsmodell	10
2.6 Laufzeit	10
2.7 Asymptotische Notation	11
2.8 Representationen von Graphen (im Computerspeicher)	11

2.9 Irrgärtchen erkunden	12
Algorithmus: Tiefensuche	12
2.10 Tiefensuchbaum	12
2.11 Topologische Sortierung durch Tiefensuche	14
2.12 Kürzeste Wege	14
2.13 Datenstrukturen für Graphtraversierung	14
Algorithmus: Breitensuche	14
3 Algorithmenentwurf	15
3.1 Maximum Subarray	15
3.2 Naiver Algorithmus	15
Algorithmus: Divide-and-Conquer	16
Algorithmus: Induktiv	17
3.3 Asymptotische Laufzeitanalyse	18
3.4 \mathcal{O} -Notation	18
3.5 Asymptotische obere Schranke	18
3.6 Asymptotische untere Schranke	18
3.7 Asymptotische genaues Wachstum	18
3.8 Komplexität eines Problems	18
3.9 Such Algorithmen	18
Algorithmus: Binäre Suche	18
Algorithmus: Binäre Suche Iterativ	19
Algorithmus: Interpolations Suche	20
Algorithmus: Lineare Suche	20
3.10 Sortieren	21
Algorithmus: Prüfe Sortiertheit	21
Algorithmus: Bubble Sort	21
Algorithmus: Selection Sort	21

Algorithmus: Insertion Sort	22
3.11 Billiges Maximum finden	23
Algorithmus: Heap Sort	24
Algorithmus: Merge Sort	24
3.12 Natural MergeSort	25
Algorithmus: Quick Sort	25
3.13 Geht sortieren schneller als $n \log n$	26
4 Datenstrukturen für Abstrakte Datentypen (ADTs)	26
4.1 ADT Stapel (Stack)	27
4.2 Linked List	27
4.3 ADT Schlange (Queue)	27
4.4 ADT Priority Queue	27
4.5 ADT Wörterbuch (Dictionary)	27
4.6 Linked Tree	28
Algorithmus: Linked Tree Search	28
4.7 AVL herstellen	29
4.8 upin(p)	30
5 Dynamische Programmierung	30
5.1 Längste aufsteigende Teilfolge	31
5.2 Längste gemeinsame Teilfolge	32
5.3 Minimale Editierdistanz	33
5.4 Matrixkettenmultiplikation	33
Algorithmus: Matrixkettenmultiplikation	34
5.5 Teilsummenproblem	35
Algorithmus: Teilsummen	35
5.6 Rucksackproblem	35
5.7 Approximationsschema für Rucksackproblem	36

Algorithmus: Rucksackproblem	36
5.8 Wege in Graphen zählen	37
5.9 Kürzeste Wege in Graphen	37
Algorithmus: Bellman-Ford	38
5.10 Alle kürzesten Wege	39
Algorithmus: Floyd-Warshall	39
6 Minimale Spannbäume	39
6.1 Drei Äquivalente Formulierungen zum minimalen Spannbaum Problem	39
6.2 Gierige Algorithmen	40
Algorithmus: Kruskal	40
Algorithmus: Prims	40
Algorithmus: Rückwärts-Kruskal	40
6.3 Kreisprinzip (Cycle Property)	41
6.4 Korrektheit von Rückwärts-Kruskal durch Kreisprinzip	41
6.5 Schnittprinzip	42
6.6 Korrektheit von Prim durch Schnittprinzip	42
6.7 Korrektheit von Kruskal durch Schnittprinzip	43
6.8 Union-Find	43
6.9 Amortisierte Analyse	45
7 Datenstrukturen für Graphen	45
7.1 Kürzeste Wege	45
7.2 One-To-All (Distanzgraph)	46
Algorithmus: Dijkstra	47
Algorithmus: Johnson	48
8 Interessante Divide and Conquer Algorithmen (Nicht Prüfungsrelevant)	49
8.1 Matrix Multiplikation	50

Algorithmus: Strassen	50
8.2 Auswahlproblem	51
Algorithmus: Median der Mediane	52

The following was presented in the lecture on 20. September 2018.

0 Einführung

0.1 Organisation

Vorlesungswebseite: <https://www.cadmo.ethz.ch/education/lectures/HS18/DA/index.html>
 Fragen: <moodle-app2.let.ethz.ch/auth/shibboleth/login.php>
 Slides vorher herunterladen: <talks2.dsteurer.org/graphenalgorithmen1.pdf>

1 Graphentheorie

Graphen sind informell ausgedrückt, ein mathematisches Modell eines Netzwerks, bei dem Objekte miteinander verknüpft werden.

Begriffe:

- Knoten: Die Punkte
- Kanten: Die Verbindungen zwischen den Knoten
- adjazent: Verbindung zweier Knoten
- inzident: Verbindung zwischen Kante und Knoten

1.1 Graph als Mathematisches Objekt

Definition 1. ein Graph $G = (V, E)$ besteht aus:

- Menge von Knoten $V = \{v_1, \dots, v_n\}$ (eng.: vertices)
- Menge von Kanten $E = \{e_1, \dots, e_m\}$ (eng.: edges)

Dabei ist jede Kante e_k ein ungeordnetes Paar zweier Knoten v_i, v_j (also: $e_k = \{v_i, v_j\}$)

Begriffe:

- Knoten v_i, v_j sind adjazent / benachbart in G falls $\{v_i, v_j\} \in E$
- Knoten v_i und Kante e_k sind inzident falls $v_i \in e_k$
- Nachbarschaft $N_G(v_i)$ ist die Menge der benachbarten Knoten von v_i
- Grad: $\deg_G(v_i) := |N_G(v_i)|$ = Anzahl an Nachbarn von v_i
- ein Graph $G' = (V', E')$ heisst Teilgraph von G falls $V' \subseteq V$ und $E' \subseteq E$
(Erklärung: \subseteq bedeutet Teilmenge von und kann auch gleich sein.)

1.2 Besondere Graphen

- Kreis: Knoten sind im Kreis verbunden.
- Pfad: Knoten sind in einer Reihe verbunden.
- Clique: Jeder Knoten ist mit jedem anderem Knoten verbunden.
- Stern: Nur eine Knoten ist mit allen anderen Knoten verbunden.

1.3 Summe der Knotengrade

Rule 1. Für jeden Graphen G mit Knoten $V = \{v_1, \dots, v_n\}$ und Kanten $E = \{e_1, \dots, e_m\}$ gilt
 $\deg_G(v_1) + \dots + \deg_G(v_n) = 2m$

Grad \deg_G bedeutet wie viele Kanten vom jeweiligen Knoten ausgehen.

1.4 Knotenfärbung

Eingabe: Graph $G = (V, E)$ und Anzahl k von Farben.

Ziel: Färbe die Knoten von G wenn möglich mit k Farben, so dass jede Kante $e = u, v$ in G erfüllt, dass u und v unterschiedliche Farben haben.

Begriffe:

Falls solch eine Färbung existiert, nennt man den Graph k -färbar oder k -partit

Für $k=2$: bipartit

Herausforderung: Anzahl möglicher Färbungen ist exponentiell

1.5 Färbbarkeit und Knotengrade

Rule 2. Jeder Graph mit maximalem Knotengrad Δ ist $(\Delta + 1)$ -färbar

1.6 Vollständige Induktion

Ziel: Zeige $\forall n \in \mathbb{N}. A(n)$ für eine Aussage $A(n)$, z.B.

$A(n)$: die Summe $1 + 2 + \dots + n$ ist gleich $\frac{1}{2}n \cdot (n + 1)$

Theorem 1.1

Um zu zeigen, dass eine Aussage $A(n)$ für alle natürlichen Zahlen $n \geq 1$ gilt, reicht folgendes aus:

- Induktionsanfang: Zeige, dass $A(1)$ gilt
- Induktionsschritt: Zeige, dass $A(n - 1) \rightarrow A(n)$ für alle $n \geq 2$ gilt

Proof 1.1: Induktionsbeispiel 1: Summenformel

$A(n)$: Die Summe $1 + 2 + \dots + n$ ist gleich $\frac{1}{2} \cdot n \cdot (n + 1)$

Induktionsanfang: Zeige $A(1)$: Stimmt

Induktionsschritt: Zeige $A(n - 1) \Rightarrow A(n)$ für alle $n \geq 2$

nach Induktionshypothese gilt:

$$1 + 2 + \dots + n - 1 + n = \frac{1}{2} \cdot (n - 1) \cdot n + n \\ \rightarrow \frac{1}{2} \cdot n \cdot (n + 1)$$

□

Proof 1.2: Induktionsbeispiel 2: Färbung und Knotengrad

$A(n)$: Jeder Graph mit n Knoten und maximalem Knotengrad $\leq \Delta$ ist $(\Delta + 1)$ -partit

Induktionsanfang: Zeige $A(1)$: Stimmt

Induktionsschritt: Zeige $A(n - 1) \rightarrow A(n)$ für alle $n \geq 2$

Entferne Knoten v_n von G (und all inzidente Kanten) und wende die Induktionshypothese auf den resultierenden Graph G' an

Betrachte Färbung von G' mit $\Delta + 1$ Farben und färbe v_n mit einer der $\Delta + 1$ Farben, die nicht von seinem $\leq \Delta$ Nachbarn verwendet wird.

So erhalten wir eine gültige Färbung von G mit $\Delta + 1$ Farben

□

The following was presented in the lecture on 27. September 2018.

1.7 Beweis: Graphen ohne ungerade Kreise sind bipartit

$A(n)$: jeder Graph mit $\leq n$ und ohne ungeradem Kreis ist bipartit.

Induktionsanfang: zeige $A(1)$

in der Tat ist jeder Graph mit nur einem Knoten 1-partit

Induktionsschritt zeige $A(n-1) \rightarrow A(n) \quad \forall n \geq 2$

„verschmelze“ Knoten v_n und Nachbarn zu neuem Knoten v' und wende Induktionshypothese auf resultierenden Graph G' an.

werwende für die Nachbarn von v_n diesselbe Farbe wie für v' und für v_n die übrige Farbe.

1.8 Gerichteter Graph

Beispiele:

- PageRank Algorithmus (Hyperlinks)
- Strassennetz (Einbahnstrassen)
- Twitter (Follower)

Definition 2. ein gerichteter Graph $G = (V, E)$ besteht aus einer Knotenmenge V und Kan-tenmenge E , so dass jede Kante $e \in E$ ein (geordnetes) Paar zweier Knoten $u, v \in E$ ist (also: $e = (u, v)$)

1.9 Wege und Pfade

sei $G = (V, E)$ ein gerichteter Graph

Ein Weg mit Startknoten $u_0 \in V$ und Endknoten $u_l \in V$ der Länge l ist eine Folge von Knoten $u_0, \dots, u_l \in V$, so dass $(u_0, u_1), (u_1, u_2), \dots, (u_{l-1}, u_l) \in E$

- Zyklus
- Pfad
- Kreis

1.10 Erreichbarkeit und Distanz

sei $G = (V, E)$ ein Graph (gerichtet oder ungerichtet)

1.11 Topologische Sortierung

Definition 3. eine Folge v_1, \dots, v_n von Knoten ist eine topologische Sortierung von $G = (V, E)$ falls für jede Kante $(v_i, v_j) \in E$ gilt, dass $i < j$, und $V = \{v_1, \dots, v_n\}$

Theorem 1.2

Ein gerichteter Graph hat eine topologische Sortierung genau dann wenn er keinen Zyklus

enthält ("azyklisch")

Lemma 1.1

Jeder gerichtete azyklischer Graph enthält eine Senke (d.h. Ausgangsgrad(0))

Definition 4. Eine Knotenmenge $W \subseteq V$ ist (stark) zusammenhängend in G , falls für alle Knoten $u, v \in W$ gilt, dass u von v erreichbar ist (d.h. \exists Weg von u nach v) Begriff $G = (V, E)$ ist zusammenhängend falls V in G zusammenhängt.

Theorem 1.3

Für jeden ungerichteten Graphen $G = (V, E)$ gibt es eine Partition $\{V_1, \dots, V_k\}$ von V , so dass $\{V_1, \dots, V_k\}$ jeweils zusammenhängend sind und jede Kante von G in genau einem Teil V_i verläuft.

Begriff: die Teile V_1, \dots, V_k heißen Zusammenhangskomponenten.

The following was presented in the lecture on 04. October 2018.

2 Algorithmen

2.1 Gierige-Knotenfärbung(G)

Für alle Knoten v in G (in beliebiger Reihenfolge): Färbe v mit der ersten Farbe, die noch von keinem Nachbarn von v verwendet wird.

Gierig bedeutet dass man eine Entscheidung nicht mehr Rückgängig machen kann.

Theorem 2.1: Wie viele Farben verwendet der Algorithmus höchstens

Für Graphen mit maximalen Knotengrad Δ kommt der Algorithmus mit $\Delta + 1$ Farben aus.

2.2 Bipartition(G)

1. Wähle beliebigen Knoten v und färbe ihn rot
2. Setze $S \leftarrow \{v\}$ und $t \leftarrow 0$
3. Wiederhole solange S Nachbarn in $V_G \setminus S$ hat
 1. Färbe die Nachbarn von S in $V_G \setminus S$ grün falls t gerade ist oder rot falls t ungerade ist.
 2. Setze $S \leftarrow S \cup N_G(S)$ und $t \leftarrow t + 1$

Theorem 2.2: Kanten müssen verschiedene Endknoten haben

Wann färbt der Algorithmus den Graph so, dass jede Kante einen roten und einen grünen Endknoten hat?

Der Algorithmus findet eine gültige Bipartition, wenn der Graph keinen ungeraden Kreis enthält (und zusammenhängend ist)

2.3 Effizienzbegriff von Algorithmen

Es gibt effiziente Algorithmen (z.B. gierige Knotenfärbung) und ineffiziente Algorithmen (z.B. Alle möglichen Färbungen eines Graphen durchprobieren)

Um die Effizienz zu bestimmen betrachten wir vereinfachte Berechnungs- und Kostenmodelle.

2.4 StarFinden(G)

1. setze $S \leftarrow V_G$
2. Solange S zwei Knoten $u \neq v$ enthält.
 1. Stelle Anfrage " $(u, v) \in E_G$?"
 2. Falls $(u, v) \in E_G$, entferne u von S: falls $(u, v) \notin E_G$, entferne v von S.
3. Prüfe ob verbleibender Knoten in S Star ist.

Algorithmus kommt mit $3(n - 1)$ Anfragen aus.

2.5 Allgemeines Berechnungsmodell

Besteht aus folgenden Komponenten:

1. **Speicher:** a-priori unbeschränkt viele, adressierbare Speicherzellen
2. **Prozessor:** führt elementare Operationen aus wie Rechenoperationen, Vergleichsoperationen, Lese- und Schreibzugriffe auf Speicher
3. **Bus:** verbindet Prozessor und Speicher

Anmerkung: Eine Speicherzelle kann verschiedene Daten enthalten, w.z.B. ein Bit, eine Zahl oder konstante Anzahl von Bits und Zahlen.

2.6 Laufzeit

Begriff: Laufzeit eines Algorithmus für eine Eingabe ist die Anzahl der elementaren Operationen, die der Prozessor während der elementaren Operationen, die der Prozessor während der

Berechnung ausführt. Die Laufzeit hängt von vielen Details der Eingabe ab, daher beschränken wir die Laufzeit als Funktion der "Grösse" der Eingabe

Definition 5. Ein Algorithmus A hat die Laufzeit f falls $f(n) = \text{maximale Laufzeit von } A \text{ über alle Eingaben der Grösse } n$ ist.

Nennt man auch die "Worst-case running time"

2.7 Asymptotische Notation

Wir ignorieren konstante Faktoren, da es wenig Sinn macht zwischen deren Laufzeiten zu unterscheiden.

Definition 6. Für die Funktion $f(n)$, besteht $O(f)$ aus allen Funktionen $g(n)$, sodass gilt:

$$\exists C > 0, \forall n \geq 1, g(n) \leq C \cdot f(n) \quad (1)$$

Falls $g(n) \in O(f)$, sagen wir "g(n) hat die Größenordnung höchstens $f(n)$ " oder "g(n) ist bis auf konstante Faktoren beschränkt durch $f(n)$ " und schreiben $g(n) \leq O(f(n))$

2.8 Representationen von Graphen (im Computerspeicher)

gerichteter Graph $G = (V, E)$ mit $V = \{v_1, \dots, v_n\}$

Definition 7. Die Adjazenzmatrix von G ist eine Tabelle A mit n Zeilen und n Spalten , so dass für den Eintrag A_{ij} in Zeile i und Spalte j gilt:

$$A_{ij} = \begin{cases} 0 & (v_i, v_j) \notin E \\ 1 & (v_i, v_j) \in E \end{cases} \quad (2)$$

Laufzeit $O(1)$ um zu testen ob G bestimmte Kante (v_i, v_j) enthält.

Laufzeit $O(n)$ um alle Nachfolger eines Knoten aufzuzählen.

Definition 8. Die Adjazenzliste von G ist eine Tabelle A mit n Einträgen, so dass Eintrag A_i eine Liste aller Nachfolger von v_i enthält. Laufzeit $O(\deg^+(v_i)+1)$ um zu testen ob G bestimmte Kante (v_i, v_j) enthält.

Laufzeit $O(\deg^+(v_i) + 1)$ um alle Nachfolger eines Knoten v_i aufzuzählen

2.9 Irrgärtner erkunden

- Markiere besuchte Stellen in einer Tabelle (eng.: array)
- Verwende einen Stapel (eng.: stack), um zu Verzweigungen zurückzufinden, die noch nicht vollständig erkundet sind.

Laufzeiten jeweils $O(1)$

Algorithmus 2.1: Tiefensuche(g, v) (eng.: depth-first search)

1. Beginne mit Stapel S , der nur Knoten v enthält
2. Wiederhole solange Stapel S nicht leer ist:
 - (a) sei w der oberste Knoten auf Stapel S
 - (b) falls w als besucht markiert ist, entferne w von Stapel S
 - (c) falls w nicht als besucht markiert ist, markiere w als besucht und lege alle nicht besuchten Nachfolger von w oben auf den Stapel S

The following was presented in the lecture on 11. October 2018.

2.10 Tiefensuchbaum

Die Tiefensuche kann man mithilfe eines Tiefensuchbaumes darstellen.

Algorithmus 2.2: Tiefensuche(g, v) Pseudocode

1. $S \leftarrow$ Stapel mit Knoten v
2. *while* S nicht leer *do*
 - (a) $w \leftarrow top(S)$
 - (b) *if* w schon besucht *then* $pop(w)$
 - (c) *if* w noch nicht besucht *then* markiere w als besucht und $push(S, \{u \in N_G^+(w) \mid u$ noch nicht besucht $\})$

Behauptung: Falls zu Beginn keine Knoten als besucht markiert sind, dann besucht Tiefensuche(G, v) genau die Knoten, die von v erreichbar sind.

Beweisidee: Da wir mit v beginnen und dann nur Kanten entlang gehen, ist jeder besuchte Knoten von v erreichbar.

Es bleibt zu zeigen, dass jeder erreichbare $u \in V$ besucht wird.

Betrachte Weg w_0, \dots, w_l von $w_0 = v$ nach $w_l = u$

$w_0 = v$ wird in der ersten Iteration besucht.

Falls w_i besucht wird, werden auch alle Nachfolger von w_i besucht, insbesondere w_{i+1} (*wenn* $0 \leq i \leq l$)

Per Induktion (über i) alle Knoten w_0, \dots, w_l werden besucht.

Corollary 2.1

Falls schon eine Knotenmenge $X \subseteq V$ als besucht markiert ist, dann besucht $Tiefensuche(G, v)$ genau die Knoten, die noch von v erreichbar sind mittels Wegen, die X vermeiden.

Anwendung: Zusammenhangskomponenten (G ungerichtet)

Führe Tiefensuche solange für noch unbesuchte Knoten aus bis alle Knoten besucht sind.
Jede Tiefensuche besucht dabei genau eine Z-Komponente.

Algorithmus 2.3: Tiefensuche(G)

1. Markiere alle Knoten $v \in V$ als nicht besucht
2. *for* $v \in V$ *do*
- (a) *if* v noch nicht besucht *then* $Tiefensuche(G, v)$

Laufzeit

$$Laufzeit \leq O(N_{push} + N_{pop} + N_{top} + N_{markier}) \quad (3)$$

N_k sind die Anzahl von k Operationen (Bei der push Operation zählen wir die Anzahl Knoten, die wir auf den Stapel legen).

Es gilt: $N_{pop} = 1 + N_{push}$ und $N_{top} = N_{pop} + N_{markier}$

$$N_{push} + N_{pop} + N_{top} + N_{markier} \leq O(N_{markier} + N_{push}) \quad (4)$$

Sei $W =$ besuchte Knoten während $Tiefensuche(G, v)$

dann gilt $N_{markier} = |W|$ und $N_{push} \leq \sum_{w \in W} \deg_G^+(w)$

$$\Rightarrow Laufzeit \leq O\left(|W| + \sum_{w \in W} \deg_G^+(w) \text{ für } Tiefensuche(G, v)\right)$$

Theorem 2.3

$Tiefensuche(G)$ hat die Laufzeit $\leq O(|V| + |E|)$

Jeder Knoten wird nur einmal besucht: $\sum_{w \in V} \deg_G^+(w) = |E|$ tieferes Verständnis: zu jedem Knoten

$v \in V$ können wir ein Intervall $I_v \subseteq \{1, \dots, 2n\}$ zuordnen (wobei $n = |V|$)

Intervall $I_v = \{pre_v, \dots, post_v\}$ wobei pre_v = Iteration, in der v als besucht markiert wird und $post_v$ = Iteration, in der v zum ersten Mal vom Stapel entfernt wird.

Lemma 2.1

$$\forall u, v \in V$$

I_u und I_v sind disjunkt oder ein Intervall ist im anderen komplett enthalten.

Beweisidee: falls $pre_u < pre_v < post_u$, befindet sich v über u im Stapel wenn v besucht wird, so dass $post_v < post_u$

Typ einer Kante $(u, v) \in E$: vorwärts falls $I_u \supseteq I_v$, rückwärts falls $I_u \subseteq I_v$, quer falls $I_u \cap I_v = \emptyset$

Lemma 2.2

Für jede Querkante $(u, v) \in E$ liegt I_v vor I_u

2.11 Topologische Sortierung durch Tiefensuche

Beobachtung: Für einen azyklischen Graph kann es keine Rückwärtskante bei der Tiefensuche geben.

Theorem 2.4

Wir erhalten eine topologische Sortierung, wenn wir die Knoten nach den Zahlen $post_v$ absteigend sortieren.

2.12 Kürzeste Wege

Länge eines Weges = Anzahl der Kanten

Distanz $d(u, v)$ = Länge des kürzesten Weges von u nach v (Sonderfälle $d(u, v) = \infty$ (bedeutet kein Weg existiert) und $d(u, u) = 0$)

2.13 Datenstrukturen für Graphtraversierung

Stapel (en. stack) LIFO = Last in, first out

Functions: push and pop

Schlange(en. queue) FIFO = first in, first out

Functions: enqueue and dequeue

Algorithmus 2.4: Breitensuche(G,s) (en. breadth-first search)

1. $dist[u] \leftarrow \infty$ für alle $u \in V \setminus \{s\}$ und $dist[s] \leftarrow 0$ (einfach Ausgedrückt: markiere S)
2. $Q \leftarrow$ Schlange mit Knoten s
3. *while* Q nicht leer *do*
 - (a) $u \leftarrow dequeue(Q)$
 - (b) *for* $(u, v) \in E$ *do*
 - *if* $dist[v] = \infty$ *then* $dist[v] \leftarrow dist[u] + 1$ und *enqueue* Q (vereinfacht: *if* v nicht markiert *then* markiere v und *enqueue*(Q, v))

Behauptung: Für alle $t \in \mathbb{N} \cup \{0\}$, gibt es einen Moment, so dass die folgenden Aussagen gelten:

1. Für alle Knoten $v \in V$ mit $d(s, v) \leq t$ gilt $dist[v] = d(s, v)$
2. Für alle anderen Knoten $v \in V$ gilt $dist[v] = \infty$
3. Die Schlange enthält genau die Knoten mit $d(s, v) = t$

Beweis per Induktion über t

The following was presented in the lecture on 18. October 2018.

3 Algorithmenentwurf

Problem: Komplexität beste Laufzeit über alle Algorithmen

Algorithmus: korrekt und effizient $\mathcal{O}(n^2)$

Beziehung: Algorithmus \leftrightarrow Induktion besteht.

3.1 Maximum Subarray

gegeben: Zahlen: $a_1, \dots, A_n \in \mathbb{Z}$

gesucht: Teilstück mit maximaler Summe ≥ 0

Also finde i, j , $1 \leq i \leq j \leq n$

sodass $S = \sum_{k=i}^j a_k$ maximal

3.2 Naiver Algorithmus

Algorithmus 3.1: Alles Ausprobieren

```

1 for i=1 bis n
2   for j=n bis n
3     S =  $\sum_{k=i}^j a_k$ 
4     Merke max

```

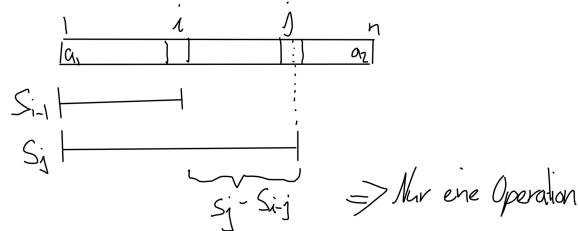
Laufzeitanalyse:

$$\sum_{i=1}^n \sum_{j=i}^n (j - i) \leq \sum_{i=1}^n \sum_{j=1}^n n = n^3 \leq \mathcal{O}(n^3) \quad (5)$$

$$\sum_{i=1}^{\frac{1}{3}n} \sum_{j=\frac{2}{3}n}^n \frac{2}{3}n = \frac{n^3}{27} \leq \mathcal{O}(n^3) \quad (6)$$

Algorithmus 3.2: Vorberechnung

Idee: Vorberechnung der Präfixsumme



```

1  $S_0 = 0$ 
2 for  $i = 1$  bis  $n$ 
3    $S_i = S_{i-1} + A_i$ 
4 for  $i = 1$  bis  $n$ 
5   for  $j = i$  bis  $n$ 
6      $S = S_j - S_{i-1}$ 
7     Merke max

```

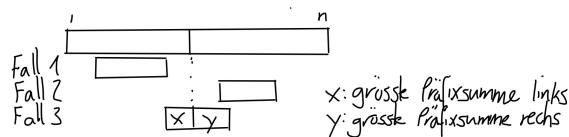
Laufzeitanalyse:

$$\sum_{i=1}^n \sum_{j=i}^n c \leq \mathcal{O}(n^2) \quad (7)$$

Algorithmus 3.3: Divide-and-Conquer

Löse kleinere Probleme → Lösung Problem

Idee halbiere Array



$n > 1$:

1. Teile Array in Hälften $\mathcal{O}(1)$
2. Löse linke Hälfte $\rightarrow M_1$ $T(n/2)$
3. Löse rechte Hälfte $\rightarrow M_2$ $T(n/2)$
4. Berechne $M_3 = x + y$ $\mathcal{O}(n)$
5. Lösung $\max(M_1, M_2, M_3)$ $\mathcal{O}(1)$

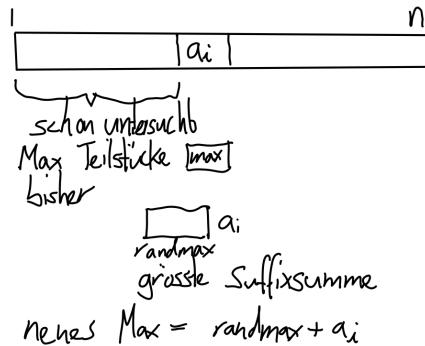
Laufzeitanalyse:

$$T(n) = \underbrace{2T\left(\frac{n}{2}\right)}_{divide} + \underbrace{an}_{conquer}, \quad T(1) = c, \quad n = 2^k \quad (8)$$

teleskopieren:

$$\log_2 n = \begin{cases} T(n) = 2(2T\left(\frac{n}{4}\right) + a\frac{n}{2}) + an = 4T\left(\frac{n}{4}\right) + 2an \\ T(n) = 4(2T\left(\frac{n}{8}\right) + a\frac{n}{4}) + 2an = 8T\left(\frac{n}{8}\right) + 3an \\ \dots \\ T(n) = n \cdot T(1) + \log_2 n(a \cdot n) \leq \mathcal{O}(n \log_2 n) \end{cases} \quad (9)$$

Algorithmus 3.4: Induktiv



```

1 max = 0
2 randmax = 0
3 for i = 1 bis n
4     randmax = randmax + ai
5     if randmax > max
6         max = randmax
7     if randmax < 0
8         randmax = 0

```

Laufzeit $\leq \mathcal{O}(n)$

Frage: Optimal?

Idee: Man muss alle a_i ansehen

Behauptung ein korrekter Algorithmus muss alle Elemente ansehen \Leftarrow Komplexität ist $\Omega(n)$

Proof 3.1

Ein indirekter Beweis:

Algorithmus sieht nicht alle a_i an \Rightarrow ist nicht korrekt

Fall 1: a_i ist in der Lösung: a_i könnte stark Negativ sein oder mathematisch $a_i = -n \cdot \max(a_j) - 1 \quad j \neq i$

Fall 2: a_i ist nicht in der Lösung: a_i könnte sehr stark Postiv sein oder mathematisch

$$a_i = n \cdot \max(a_j) + 1 \quad j \neq i$$

□

Komplexität von Max Subarray ist $\mathcal{O}(n)$ und scharf (Bedeutet, dass es nicht besser nicht geht.)
 $\Rightarrow \Theta(n)$

3.3 Asymptotische Laufzeitanalyse

Algorithmus: elementare Operationen (z.B $+, -, <, >, ==$, push, pop,...)

Die asymptotische Laufzeit ist die Anzahl in \mathcal{O} -Notation, also z.B. $T(n) = \mathcal{O}(n^2)$

3.4 \mathcal{O} -Notation

Wir betrachten positive Funktionen:

$$F_i \rho : \mathbb{N} \rightarrow \mathbb{R}^+ \quad (10)$$

3.5 Asymptotische obere Schranke

$$\mathcal{O}(g(n)) = \{f(n) | \exists c > 0 : f(n) \leq c \cdot g(n) \forall n\} \quad (11)$$

$$f(n) \in \mathcal{O}(g(n)) \Leftrightarrow f(n) \leq \mathcal{O}(g(n)) \quad (12)$$

3.6 Asymptotische untere Schranke

$$\Omega(g(n)) = \{f(n) | \exists c > 0 : f(n) \geq c \cdot g(n) \forall n\} \quad (13)$$

$$f(n) \in \Omega(g(n)) \Leftrightarrow f(n) \geq \Omega(g(n)) \quad (14)$$

3.7 Asymptotische genaues Wachstum

$$\Theta(g(n)) = \Omega(g(n)) \cap \mathcal{O}(g(n)) \quad (15)$$

$$f(n) \in \Theta(g(n)) \Leftrightarrow f(n) = \Theta(g(n)) \quad (16)$$

3.8 Komplexität eines Problems

The following was presented in the lecture on 25. October 2018.

3.9 Such Algorithmen

Input: aufsteigend sortierter Array $A : A[1] \leq A[2] \leq \dots A[n]$ und ein Element b
Output: k mit $A[k]=b$ oder "nicht gefunden" wenn b nicht in A.

Table 1:

Problem Komplexität		Beliebiger Algorithmus Komplexität
$\mathcal{O}(f(n))$	\leftarrow	$\mathcal{O}(f(n))$
nix	\leftarrow	$\Omega(f(n))$
$\mathcal{O}(f(n))$	\leftarrow	$\Theta(f(n))$
$\mathcal{O}(f(n))$	\rightarrow	nix
$\Omega(f(n))$	\rightarrow	$\Omega(f(n))$
$\Theta(f(n))$	\rightarrow	$\Omega(f(n))$

Algorithmus 3.5: Binäre Suche

```

1 BinarySearch(A,b)
2   if A empty: "nicht gefunden"
3   m = ⌊ n / 2 ⌋
4   if A[m] = b: return m
5   if b < A[m]: BinarySearch(A[1 bis m-1],b)
6   else: BinarySearch(A[m+1 bis n],b)

```

Laufzeit:

$$T(n) = T\left(\frac{n}{2}\right) + a \quad T(1) = c \quad n = 2^k \quad (17)$$

Teleskopieren:

$$\begin{aligned} &= T\left(\frac{n}{4}\right) + 2d \\ &= T\left(\frac{n}{8}\right) + 3d \end{aligned} \quad (18)$$

...

$$= T(1) + \log_2 nd$$

$$\Rightarrow c + \log_2 n \cdot d \leq \mathcal{O}(\log n) \quad (19)$$

Algorithmus 3.6: Binäre Suche Iterativ

```

1 BinarySearchIt(A,b)
2   l=1
3   r=n
4   if l>r: return "nicht gefunden"
5   while l ≤ r do
6     m = ⌊ (l+r) / 2 ⌋
7     if A[m]=b: return m
8     if b < A[m]: r = m-1
9     else: l=m+1

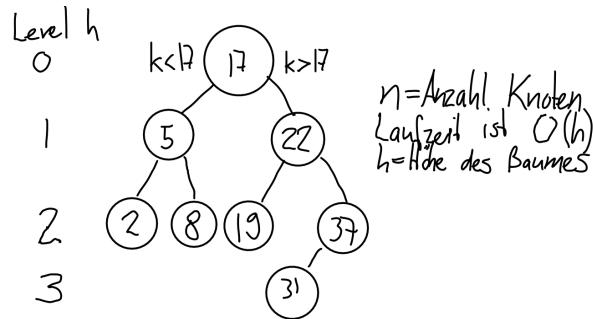
```

Laufzeit: $\mathcal{O}(\log n)$

Das ist eine vergleichsbasierte Suche

$\mathcal{O}(\log n)$ ist optimal

Beweisidee: Betrachte Suche als Entscheidungsbaum

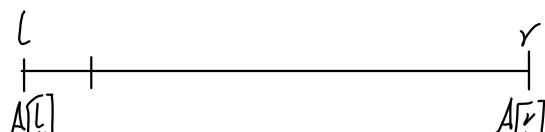


Untere Schranke: kleinstes h sodass der Baum n Knoten hat.

$$n \leq 2^0 + 2^1 + 2^2 + \dots + 2^h = 2^{(h+1)} - 1 \quad (20)$$

$$h \geq \log_2(n+1) - 1 \geq \omega(\log n) \quad (21)$$

Algorithmus 3.7: Interpolations Suche



$$m = \left\lfloor l + \frac{b - A[n]}{A[l] + A[r](r - l)} \right\rfloor \quad (22)$$

Worst case: $\mathcal{O}(n)$

Average case: $\mathcal{O}(\log \log n)$

Suche im unsortierten Array:

Algorithmus 3.8: Lineare Suche

```
1 LinearSearch(A,b):  
2     for i=1-n:  
3         if A[i]=b: return i  
4     return "nicht gefunden"
```

Laufzeit: $\mathcal{O}(n)$

Es geht nicht besser

(Anzahl Vergleiche ist $\geq \omega(n)$)

Argument 1: b muss mit allen Elementen verglichen werden.

Problem: Vergleiche innerhalb A sortiere $\mathcal{O}(\log n)$ und suche $\mathcal{O}(\log n)$

Argument 2: erlaubte Vergleiche in A

r Vergleiche in A

s Vergleiche mit b

r Vergleiche zerlegen A in Gruppen

Versuche durch Vergleiche, aber kein Vergleich zwischen Gruppen

$$r \geq n - g \text{ und } s \geq g \text{ mindestens 1 pro Gruppe} \quad (23)$$

$$r + s \geq n \geq \omega(n) \quad (24)$$

3.10 Sortieren

Problem:

Input: Array der Länge n

Output: Permutation A' von A

aufsteigend sortiert $A'[1] \leq A'[n]$

Häufig $A' = A$ ("in-place")

Algorithmus 3.9: Prüfe Sortierheit

```
1 IsSorted(A)
2   for i=1:n-1
3     if A[i] > A[i + 1]: return false
4   return true
```

Laufzeit: $\mathcal{O}(n)$

Algorithmus 3.10: Bubble Sort

```
1 for j=1:n-1
2   for i=1:n-j
3     if A[i] > A[i + 1] tausche A[i] mit A[i + 1]
```

Laufzeit:

$$\sum_{j=1}^{n-1} \sum_{i=1}^{n-j} 1 = \sum_{j=1}^{n-1} (n - j) = \mathcal{O}(n^2) \quad (25)$$

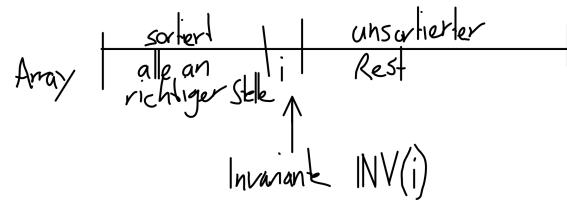
Für das Finden der unteren Schranke:

Vergleiche: $\mathcal{O}(n^2)$

Vertauschungen $\mathcal{O}(n^n)$

Algorithmus 3.11: Selection Sort

induktiv: Baue Lösung von links nach rechts.



```

1 for i=1:n-1
2     k=i
3     for j=i:n-1
4         if A[j+1] < A[k]:
5             k=j
6     Vertausche A[i] mit A[k]

```

Laufzeit: Minimum aus n-i Elementen: n-i

$$\sum_{i=1}^{n-1} (n-i) = \frac{n(n-1)}{2} \leq \mathcal{O}(n^2) \quad (26)$$

Algorithmus 3.12: Insertion Sort

induktiv: von links nach rechts



```

1 InsertionSort(A)
2     for i=1:n-1
3         j=BinarySearch(A[1:i], A[i+1])
4         A[j]=A[i+1]
5         A[j+1,...,i+1]=A[j:i]

```

Laufzeit:

$$\text{Suchen: } \sum_{i=1}^{n-1} \log(i) \leq \mathcal{O}(n \log(n)) \quad (27)$$

$$\text{Verschieben: } \sum_{i=1}^{n-1} i - 1 \leq \mathcal{O}(n^2) \quad (28)$$

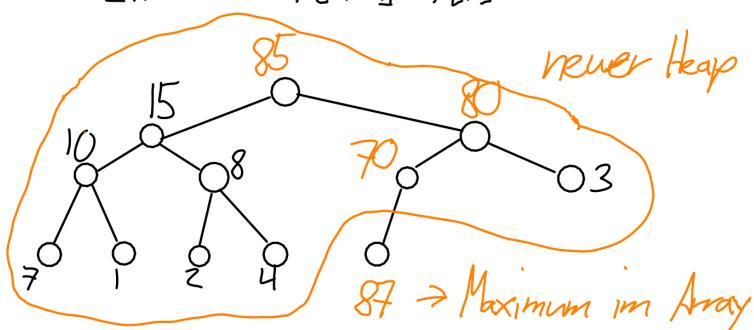
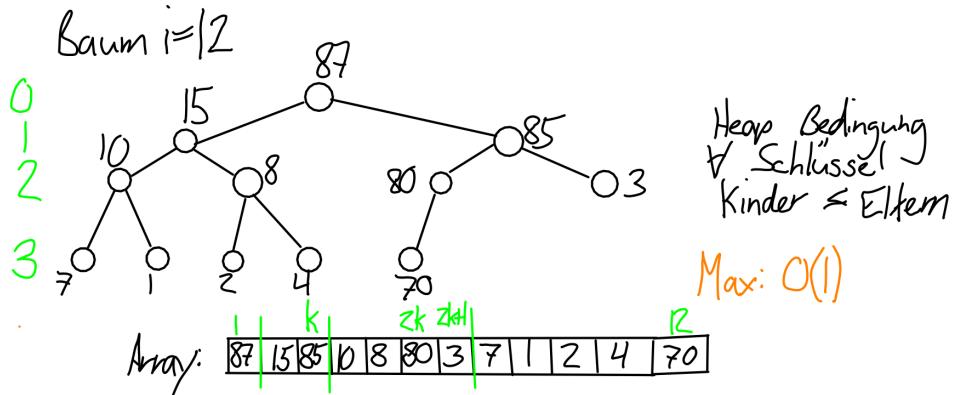
Table 2:

	Selection Sort	Insertion Sort	Bubble Sort
Vergleiche	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
Tausch / Bewegen	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$

Note 1. Bei Algorithmen muss man nicht immer den ganzen Input ansehen \Rightarrow beweist nicht immer, dass es $\Omega(n)$ ist.

3.11 Billiges Maximum finden

Lösung Max-Heap



Eigenschaften:

- Anzahl Blätter = $\lceil \frac{n}{2} \rceil$
- Höhe: $\lfloor \log_2(n) \rfloor$
- Voller Binärbaum

Restore HeapCondition(A,k,i)

$$\text{Laufzeit: Vergleichen : } \mathcal{O}\left(\sum_{i=n}^2 c \log(i)\right) \leq \mathcal{O}(n \log n)$$

Algorithmus 3.13: HeapSort

Beachte, dass man immer wieder die Heapbedingung herstellen muss.

```
1 HeapSort(A)
2   for i = ⌊n/2⌋ to 1:
3     RestoreHeapCondition(A, i, n)
4   for i=n, ..., 2
5     Vertausche(A[1], A[i])
6     RestoreHeapCondition(A, 1, i-1)
```

Laufzeit: RHC(A,i,n) $\Rightarrow \mathcal{O}(n \log n)$

Anzahl Vergleiche: $0 \cdot 2^d + 2 \cdot 2^{d-1} + 4 \cdot 2^{d-2} + \dots + 2d \cdot 2^{d-d}$

$$= \sum_{i=1}^{d-1} 2^i \cdot 2(d-i) \leq \mathcal{O}(2^d) = O(n)$$

HeapSort

- + $\mathcal{O}(n \log n)$
- + InPlace
- - Lokalität

Algorithmus 3.14: MergeSort

Idee: Divide-and-Conquer

Sortiere beide Hälften einzeln.

Man hat zwei Zeiger die durch beide Arrays durchgeht und man nimmt immer den kleineren Wert auf den dieser Zeiger zeigt.

```
1 MergeSort(A,l,r)
2   if l < r
3     m = ⌊(l+r)/2⌋
4     MergeSort(A,l,m)
5     MergeSort(A,m+1,r)
6     Merge(A,l,m,r)
7
8 Merge(A,l,m,r)
9   i=1
10  j=m+1
11  k=1
12  while i ≤ m und j ≤ r
13    if A[i] < A[j]
14      B[k]=A[i]
15      i++
16      k++
17    else
18      B[k]=A[j]
```

```

19         j++
20         k++
21     endwhile
22     Uebernimmt Rest von i bis m oder j bis r
23     A=B

```

Laufzeit: von Merge(A,l,m,r) ist $\mathcal{O}(n)$

Gesamte Laufzeit: $T(n) = 2T(\frac{n}{2}) + cn \leq \mathcal{O}(n \log n)$

Nachteil: Extraspeicher $\mathcal{O}(n)$

Example 1. Beispiel MergeSort Algorithmus:

Table 3:

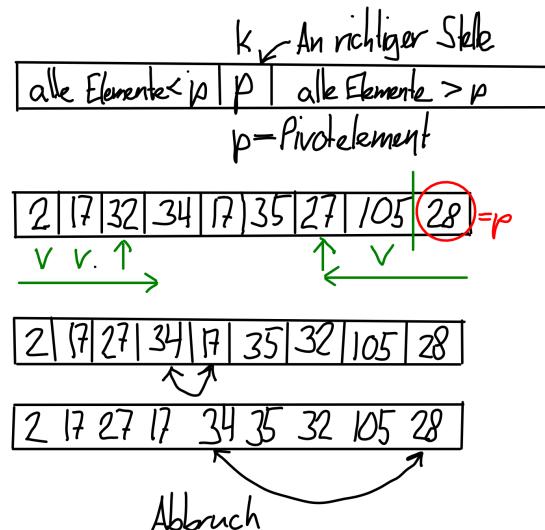
9	7	3	2	1	8	4	6
7	9	2	3	1	8	4	6
2	3	7	9	1	4	6	8
1	2	3	4	6	7	8	9

3.12 Natural MergeSort

Geht von Links nach Rechts und entdeckt ob ein Teil schon sortiert ist und lässt es dann aus.

Algorithmus 3.15: QuickSort

Idee: teile und Arbeit, danach Rekursion links und rechts



```

1 QuickSort(A,l,r)
2   Aufteilen(A,l,r)
3   QuickSort(A, l, k-1)

```

```

4     QuickSort(A,k+1,r)
5
6 Aufteilen(A,l,r)
7     i=1
8     j=r-1
9     p=A[r]
10    repeat
11        while i<r und A[i]<p
12            i++
13        while j>l und A[j]<p
14            j++
15        if i<j
16            tausche A[i], A[j]
17        until i ≥ j
18        tausche A[i], A[r]
19        return i

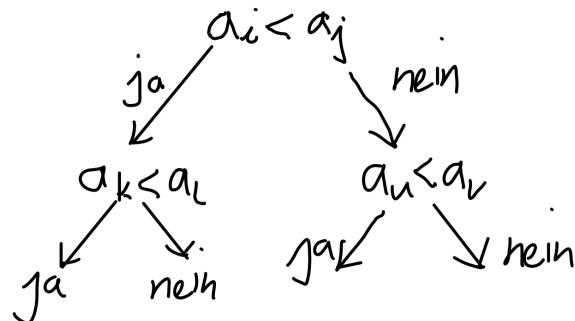
```

Laufzeit Optimal: $T(n) = 2T(\frac{n}{2}) + cn \leq \mathcal{O}(n \log n)$

Laufzeit WorstCase: $T(n) = T(n - 1) + cn \leq \mathcal{O}(n^2)$

3.13 Geht sortieren schneller als $n \log n$

Idee: Entscheidungsbaum



Blätter \leftrightarrow Mögliche Anordnungen des Array: $n!$

Laufzeit \leftrightarrow Höhe Baum

Baum der Höhe h hat \leq Blätter

$$2^h \geq n!$$

$$h \geq \log_2(n!)$$

$$h \geq \Omega(n \log n)$$

The following was presented in the lecture on 08. November 2018.

4 Datenstrukturen für Abstrakte Datentypen (ADTs)

ADT Objekte (bei uns Schlüssel $\in \mathbb{N}$)

ADT Operationen

Datenstruktur Implementierung eines ADTs

4.1 ADT Stapel (Stack)

push(x,S) legt x auf Stapel

pop(S) entfernt und liefert oberstes Element

top(S) entfernt oberstes Element

4.2 Linked List

Wir haben immer ein Objekt mit einem Schlüssel (Wert) und einem Zeiger der weiss, wo das nächste Element liegt.

Die drei oberen Operationen haben alle $\mathcal{O}(1)$

4.3 ADT Schlange (Queue)

enqueue(x,S)

dequeue(S)

4.4 ADT Priority Queue

insert(x,P) fügt x ein

extractMax(P) liefert und entfernt Maximum

Eine Priority Queue kann man mit der Datenstruktur Heap implementieren.

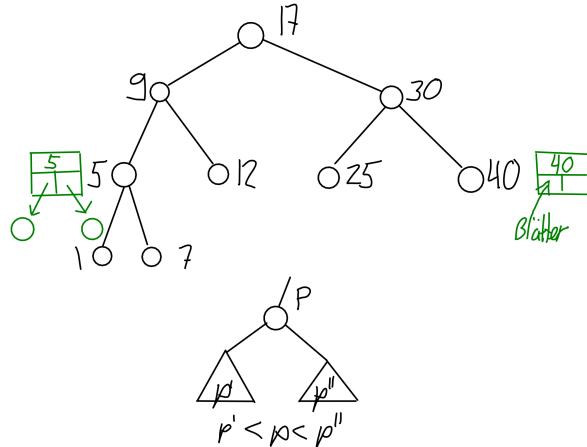
4.5 ADT Wörterbuch (Dictionary)

search(x,W) Ob x in W vorhanden ist?

insert(x, W) füge x ein (oder melde, wenn schon drin)

remove(x,W) Entferne wenn drin.

4.6 Linked Tree



Algorithmus 4.1: Linked Tree Search

```

1 Search(x,p)
2   if p=null: return " nicht gefunden"
3   if x=p.key: return "gefunden"
4   if x < p.key: Search(x, p.left)
5   else: Search(x,p.right)

```

Suche: $\mathcal{O}(h)$

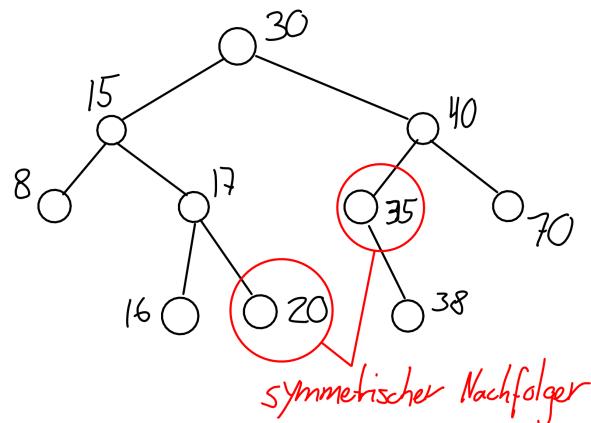
Entferne(x,p): zuerst Suche (also $\mathcal{O}(h)$)

Fall 1 x ist am Ende des Baumes \Rightarrow Einfach löschen

Fall 2 x zeigt nur noch auf ein Objekt. \Rightarrow Zeiger von x-1 auf x+1 umbiegen

Fall 3 x zeigt auf zwei Objekte \Rightarrow Tausche x mit einem symmetrischen Nachfolger von x, dann hat man entweder Fall 1 oder Fall 2 für symmetrischen Nachfolger. (Siehe folgendes Bild)

Laufzeit ist $\mathcal{O}(h)$



Problem: $h = \Theta(\log n)$ erhalten.

Idee 1: "perfekt balanziert" erhalten ist sehr schwierig, fast unmöglich in dieser Laufzeit.

Idee 2: AVL-Baum. Höhe darf um maximal 1 variieren für jeden Knoten.

Wie hoch ist ein AVL-Baum?

Für Höhe h , wieviele Knoten n gibt es mindestens.

$$n \text{ Knoten} \Leftrightarrow n + 1 \text{ Blätter} \quad (29)$$

$$MB(h) = \text{Mindestanzahl Blätter} \quad (30)$$

$$MB(h) = MB(h-1) + MB(h-2) = Fib(h+2) \quad (31)$$

$$Fib(h) = \frac{1}{\sqrt{5}} \cdot \left(\left(\frac{1+\sqrt{5}}{2} \right)^h - \left(\frac{1-\sqrt{5}}{2} \right)^h \right) \quad (32)$$

$$\Theta((1.6...)^h) \quad (33)$$

$$h \leq 1.44... \cdot \log_2 n = \Theta(\log n) \quad (34)$$

4.7 AVL herstellen

einfügen wird nun zu einfügen+ AVL herstellen

entfernen wird nun zu entfernen + AVL herstellen

Merke in jedem Knoten:

$$bal = h_r - h_l$$

if $bal = -1$: linker TB höher

if $bal = +1$: rechter TB höher

if $bal = 0$: TB sind gleich gross

Nun hat man folgende Fälle (vor dem einfügen):

1. $bal(p) = -1$ (unmöglich)

2. $bal(p) = 0$

- $bal(p) = -1$

- $bal(x) = 0$

- upin(p)

3. $bal(p) = +1$

- $bal(p) = 0$

- $bal(x) = 0$

Beim Aufruf upin(p) gilt:

1. TB p ist gewachsen

2. $bal(p) \neq 0$

3. p hat Vorgänger

4.8 upin(p)

q ist der Vorgänger von p Fallunterscheidung:

$$1. \ bal(q) = +1$$

$$bal(q) = 0$$

$$2. \ bal(q) = 0$$

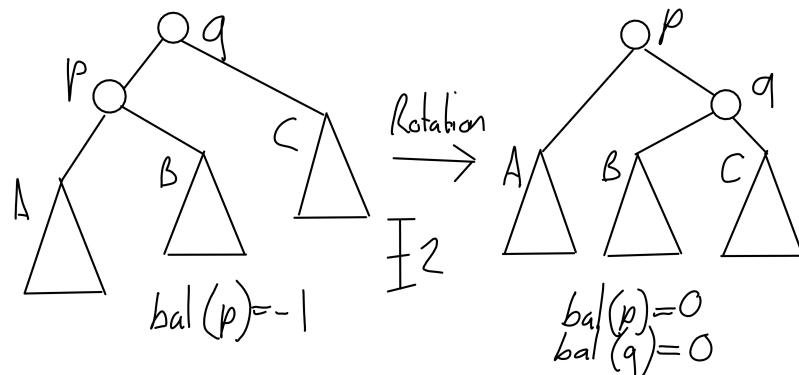
$$bal(q) = -1$$

unin(q)

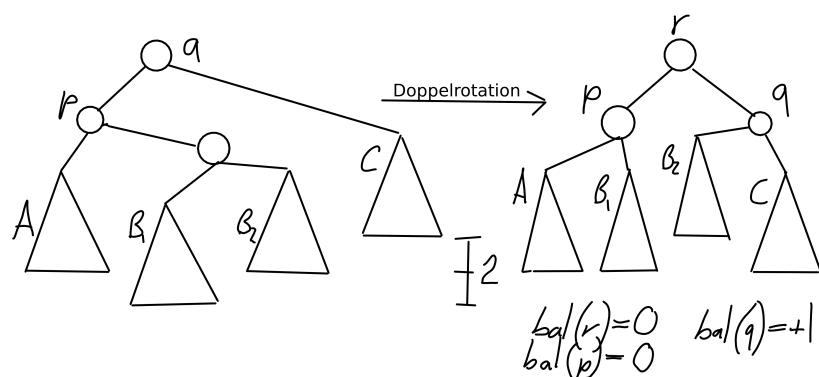
$$3. \ bal(q) = -1$$

TB q ist nicht AVL \Rightarrow Arbeit

3a) $bal(p) = -1$



3b) $bal(p) = +1$



The following was presented in the lecture on 15. November 2018.

5 Dynamische Programmierung

1. Design der Induktion

2. Bottom-up Berechnung

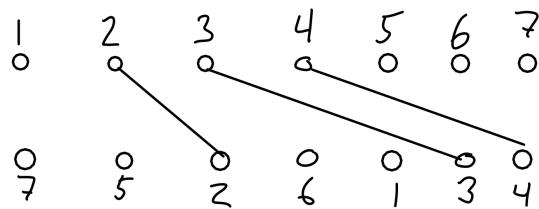
Beispiel Top-Down: Fibonacci Zahlen rekursiv gelöst rufen immer wieder die selben Berechnungen auf. Daraus folgt eine Laufzeit von $\mathcal{O}(2^{\frac{n}{2}})$

Idee: Memoization um bereits berechnete Werte zu speichern.

Daraus folgt eine verbesserte Laufzeit von $\mathcal{O}(n)$, wenn immer wieder die bereits berechneten Werte aufgerufen werden.

Nun kann man das auch Bottom-Up zu implementieren. Um auf die Laufzeit von $\mathcal{O}(n)$ und einen Speicher von $\mathcal{O}(1)$, da nur die letzten beiden Werte gespeichert werden müssen.

5.1 Längste aufsteigende Teilfolge



Ziel möglichst viele Verbindungen ohne zu Kreuzen

Äquivalent: Gegeben Folge $A[i]$ $A[n]$ von verschiedenen Zahlen. Gesucht längste aufsteigende Teilfolge

Designe Induktion: $Lat(i) = Lat$ der ersten i .

Grundidee: $Lat(ii - 1) \xrightarrow{\text{Aufsteigen}} Lat(i)$

Invariante 1

Wir haben $Lat(i-1)$

Fall 1: $A[i]$ passt \rightarrow Anhängen $\rightarrow Lat(i)$

Fall 2: $A[i]$ passt nicht

Invariante 2

Wir haben alle $Lat(1-i)$

Fall 1: $A[i]$ passt \rightarrow anhängen an alle wo passt $\rightarrow Lat(i) \checkmark$

Fall 2: $A[i]$ passt nicht \Rightarrow Man muss sich auch alle kürzeren aufsteigenden Teilerfolgen merken.

Invariante 3

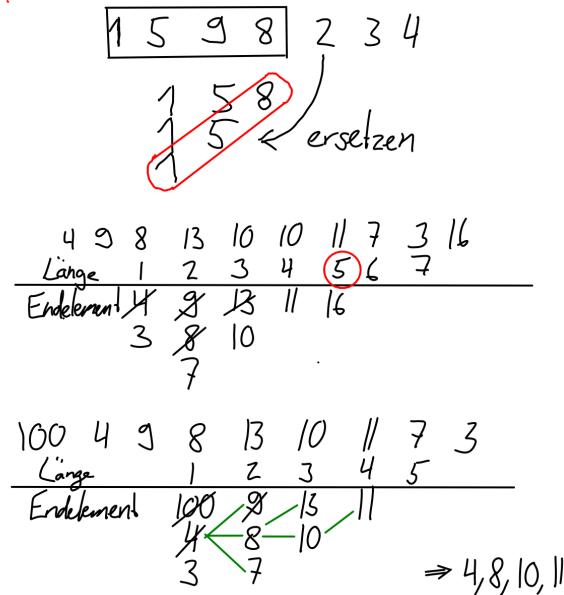
Wir haben $Lat(i-1)$ mit kleinstem Endelement.

Fall 1: $A[i]$ passt

Fall 2: $A[i]$ passt nicht

Invariante 4

Merke für jede Länge die aufsteigende Teilfolge mit kleinstem Endelement.



Fall 1: $A[i]$ passt \rightarrow neue Länge mit Endelement $A[i]$

Fall 2: $A[i]$ passt nicht

Finde Teilerfolge die mit x endet, $A[i] < x$ und Teilerfolge mit kleineren Länge hat Endelement $< A[i]$

Laufzeit: $\mathcal{O}(n \log n)$ da in jedem Schritt eine binäre Suche durchgeführt werden muss, um den Ort zu bestimmen (Suche $\mathcal{O}(\log n)$)

Speicher: $\mathcal{O}(n)$

Um die Folge zu merken, muss man sich immer den Vorgänger merken und braucht deshalb den doppelten Speicherplatz.

5.2 Längste gemeinsame Teilstrecke



4 Fälle

1. Buchstaben gekreuzt mit Leerzeichen: $LGT(n-1, m)$
2. Leerzeichen gekreuzt mit Buchstaben: $LGT(n, m-1)$
3. Buchstaben gekreuzt mit anderem Buchstaben: $LGT(n-1, m-1)$
4. Buchstaben gekreuzt mit gleichem Buchstaben: + 1

$$LGT(n, m) = \max(LGT(n-1, m), LGT(n, m-1), LGT(n-1, m-1) [+1 \text{ wenn } A[n] = B[m]]) \quad (35)$$

Laufzeit: $\mathcal{O}(nm)$

Table 4:

	-	T	I	G	E	R
-	(0)	0	0	0	0	0
Z	0	(0)	0	0	0	0
I	0	0	(1)	1	1	1
E	0	0	(1)	1	2	2
G	0	0	1	(2)	2	2
E	0	0	1	2	(3)	(3)

Speicher: $\mathcal{O}(nm)$

5.3 Minimale Editierdistanz

Gegeben: 2 Zeichenfolgen $A[1, \dots, n]$, $B[1, \dots, m]$
 Editierops:

1. Zeichen ändern
2. Zeichen löschen
3. Zeichen einfügen

Wir suchen die minimale Anzahl die notwendig ist um ein Wort in ein anderes umzuwandeln.

$$EDT(n, m) = \min(EDT(n-1, m)+1, EDT(n, m-1)+1, EDT(n-1, m-1) [+1 \text{ wenn } A[n] = B[m]]) \quad (36)$$

5.4 Matrixkettenmultiplikation

Matrixprodukt: A_1, A_2, \dots, A_n
 gesucht beste Reihenfolge (Minimale Operationen)

The following was presented in the lecture on 22. November 2018.

Table 5:

	-	T	I	G	E	R
-	0	1	2	3	4	5
Z	1	1	2	3	4	5
I	2	2	1	2	3	4
E	3	3	2	2	2	3
G	4	4	3	2	3	3
E	5	5	4	3	2	(3)

Längste gemeinsame Teilfolge

$$T(m, n) = \begin{cases} T(m-1, n-1) + 1 & \text{falls } a_m = b_n \\ \max\{T(m-1, n), T(m, n-1)\} & \text{sonst} \end{cases} \quad (37)$$

Matrixkettenmultiplikation (Fortsetzung)

Man kann die Assoziativität benutzen um Matrixmultiplikationen zu vereinfachen.
Gesucht ist die Klammerung mit den minimalen Kosten.

Note 2. Einsicht: Optimale Klammerung von A_1, \dots, A_n besteht aus einer Multiplikation von A_1, \dots, A_i mit A_{i+1}, \dots, A_n und den optimalen Klammerungen von A_1, \dots, A_i und A_{i+1}, \dots, A_n

Zerlegung in Teilproblem

$T(p, q) :=$ minimale Kosten einer Klammerung von A_p, \dots, A_q

Rekurrenz

$$T(p, q) = \min_{p < i < q} \{T(p, i) + T(i+1, q) + C_{p,i,q}\} \quad (38)$$

wobei $C_{p,i,q}$ = Kosten der Multiplikation von A_p, \dots, A_i mit A_{i+1}, \dots, A_q

Algorithmus 5.1: Matrixkettenmultiplikation

Initialisere $T(p, p) = 0$ für alle p ; dann verwende Rekkurrenz zur Berechnung aller Einträge der Form $T(p, p+1)$; danach alle Einträge der Form $T(p, p+2)$; usw.
Laufzeit um die Klammerung zu finden:
 $\mathcal{O}(n)$ Schritte pro Eintrag der Tabelle; $\mathcal{O}(n^2)$ Tabelleneinträge $\Rightarrow \mathcal{O}(n^3)$ Laufzeit.

5.5 Teilsummenproblem

Gegeben: natürliche Zahlen a_1, \dots, a_n und b

Gesucht: eine Teilmenge $S \subseteq [n]$, so dass $\sum_{i \in S} a_i = b$

Note 3. Falls eine Lösung existiert, dann muss b oder $b - a_n$ eine Teilsumme von a_1, \dots, a_{n-1} sein

Zerlegung in Teilproblem

$T(k, s) :=$ Wahrheitswert der Aussage "s ist Teilsumme von a_1, \dots, a_k "

Rekurrenz zwischen Teilproblemen

$$T(k, s) = T(k-1, s) \vee T(k-1, s-a_k) \quad (39)$$

Table 6:

T	0	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0
4	1	1	1	0	1	1	1	0	0	0
5	1	1	1	0	1	1	1	1	0	1
7	1	1	1	0	1	1	1	1	1	1

Algorithmus 5.2: Teilsummen

```

1  $T[s] \leftarrow 0$  für alle  $s \in \{1, \dots, b\}$  und  $T[0] \leftarrow 1$ 
2 for  $k=1:n$ 
3   for  $s=b:0$ 
4     if  $s \geq a_k$  und  $T[s-a_k] = 1$  do
5        $T[s] \leftarrow 1$ 

```

Laufzeit: $\mathcal{O}(b \cdot n)$

Speicherplatz: $\mathcal{O}(b)$

Eingabegrößen sind zwischen $\theta(n + \log b)$ und $\mathcal{O}(n \cdot \log b)$

5.6 Rucksackproblem

Gegeben: n Gegenstände mit Gewichten $w_1, \dots, w_n \in \mathbb{N}$ und Nutzwerten $v_1, \dots, v_n \in \mathbb{N}$, Gewichtsschranke W

Gesucht: Teilmenge S mit Gewicht $\sum_{i \in S} w_i \leq W$ und grösstmöglicher Nutzwert $\sum_{i \in S} v_i$

Note 4. Optimale Lösung besteht aus optimaler Lösung für die ersten $n - 1$ Gegenstände mit Gewichtsschranke W oder $W - w_n$

Zerlegung in Teilprobleme

$T(i, w) :=$ maximaler Nutzwert von $S \subseteq \{1, \dots, i\}$

Rekurrenz zwischen Teilproblemen

$$T(i, w) = \max\{T(i-1, w), v_i + T(i-1, w - w_i)\} \quad (40)$$

Berechnung: Betrachte $i \in \{1, \dots, n\}$ und $w \in \{0, \dots, W\}$ und berechne die Tabelle gemäss Rekurrenz.

Laufzeit $\mathcal{O}(n \cdot W)$

5.7 Approximationsschema für Rucksackproblem

Gegeben: $w_1, \dots, w_n \in \mathbb{N}, v_1, \dots, v_n \in \mathbb{N}, W \in \mathbb{N}$

Maximiere: $\sum_{i \in S} v_i$ unter Bedingung $\sum_{i \in S} w_i \leq W$

Note 5. Idee: Wähle geeignetes $K \in \mathbb{N}$ und finde optimale Lösung für gerundete Nutzwerte $v'_i = \lfloor v_i/K \rfloor$ (mittels dynamischer Programmierung) Sei $V = v_1 + \dots + v_n$; wir nehmen an $\forall i, w_i \leq W$

Laufzeit: $\mathcal{O}(n \cdot V/K)$

Bahauptung: $\sum_{i \in S'} v_i \geq \sum_{i \in S} v_i - n \cdot K$

Für $K = \epsilon \cdot \frac{V}{n^2}$ gilt nun

$$\sum_{i \in S'} v_i \geq \sum_{i \in S} v_i - \epsilon \frac{V}{n} \geq (1 - \epsilon) \sum_{i \in S} v_i \quad (41)$$

da gilt $\sum_{i \in S} v_i \geq \frac{V}{n}$ (Da der wertigste Gegenstand eine zulässige Lösung ist)

The following was presented in the lecture on 29. November 2018.

Algorithmus 5.3: Approximationsalgoritmus für das Rucksackproblem

Gegeben $\epsilon > 0$ und $\{w_i\}, \{v_i\}$, können wir in Zeit $\mathcal{O}(\frac{n^3}{\epsilon})$ eine Lösung indem man $\geq (1 - \epsilon)$ -Optimale Lösung rechnet.

Idee: Wähle geeignetes $K \in \mathbb{N}$ und finde optimale Lösung für gerundete Nutzwerte $v' = \lfloor \frac{v_i}{K} \rfloor$ (mittels dynamischer Programmierung)

Neue Laufzeit: $\mathcal{O}(\frac{nV}{K})$

Behauptung:

$$\sum_{i \in S'} v_i \geq \sum_{i \in S} v_i - n \cdot K \quad (42)$$

Es gilt:

$$\sum_{i \in S'} v'_i \geq \sum_{i \in S} v'_i \quad (43)$$

Proof 5.1

$$\sum_{i \in S'} v_i \geq k \cdot \sum_{i \in S'} v'_i \geq k \cdot \sum_{i \in S} v'_i \geq \sum_{i \in S} v_i - n \cdot K \quad (44)$$

□

5.8 Wege in Graphen zählen

Gegeben: Ein gerichteter Graph G , Knoten $u, v \in V(G)$, Länge $L \in \mathbb{N}$

Gesucht: Anzahl der Wege in G von u nach v der Länge L

Struktur der Lösungen: Jeder Weg von u nach v der Länge L besteht aus einem Weg der Länge $L-1$ zu einem Knoten w und einer Kante von w nach v

Zerlegung in Teilprobleme: $T^{(l)}(i, k) = \text{Anzahl der Wege von } i \text{ nach } k \text{ der Länge } l$

Rekurrenz zwischen Teilproblemen:

$$T^{(l)}(i, k) = \sum_{j:(j,k) \in E(G)} T^{(l-1)}(i, j) = \sum_j a_{jk} \cdot T^{(l-1)}(i, j) \quad (45)$$

wobei $A = (a_{ij})$ die Adjazenzmatrix ist.

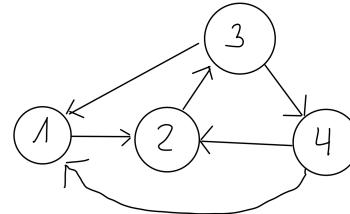


Table 7:

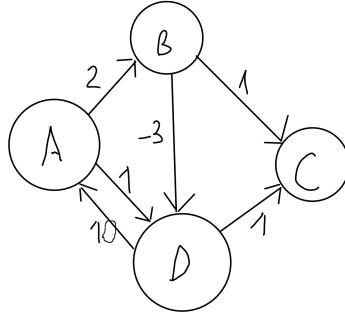
	$T^{(0)}$				$T^{(1)}$				$T^{(2)}$				$T^{(3)}$			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	1	0	0	0	0	1	0	0	0	0	1	0	1	0	0	1
2	0	1	0	0	0	0	1	0	1	0	0	1	1	2	0	0
3	0	0	1	0	1	0	0	1	1	2	0	0	0	1	2	0
4	0	0	0	1	1	1	0	0	0	1	1	0	1	0	1	1

Fazit: Matrix Multiplikation ist einfache Form der dynamischen Programmierung.

5.9 Kürzeste Wege in Graphen

Gegeben: gerichteter Graph $G = (V, E)$, Knoten $s, t \in V$, Gewichte $w : E \rightarrow \mathbb{R}$

Gesucht: Weg von s nach t in G mit kleinstem Gewicht.



Struktur: Optimaler Weg von s nach t besteht aus optimalem Weg von s zu einem Knoten i und der Kante von i nach t .

Teilprobleme: $T(j) = \text{minimales Gewicht eines Weges von } s \text{ zu } j$

Rekurrenz: $T(j) = \min_{i:(i,j) \in E} T(i) + w(i,j)$ Jedoch funktioniert es nicht, denn die Rekurrenz bezieht sich nicht auf kleinere Teilprobleme (Es gibt keine natürliche Reihenfolge) Verbesserte Struktur: Optimaler Weg von s nach t besteht aus optimalem Weg von s zu einem Knoten i und der Kante von i nach t dabei hat er optimale Weg von s zu i eine Kante weniger.

Teilprobleme: $T(j, l) = \text{minimales Gewicht eines Weges von } s \text{ zu } j \text{ mit } \leq l \text{ Kanten}$

Rekurrenz: $T(j, l) = \min\{T(j, l-1), \min_{(i,j) \in E} T(i, l-1) + w(i, j)\}$

Algorithmus 5.4: Bellman-Ford

Berechnung: Initialisiere Einträge der Form $T(j, 0)$; berechne Einträge der Form $T(j, 1)$ via Rekurrenz; danach $T(j, 2)$

Table 8:

	0	1	2	3	4
A	0	0	0	0	0
B	∞	2	2	2	2
C	∞	∞	2	0	0
D	∞	1	-1	-1	-1

Theorem 5.1

$T(j, n) < T(j, n-1)$ genau dann wenn es Wege von s nach j mit beliebig kleinem (negativen) Gewicht gibt.

Anzahl der Einträge: n^2

Berechnung eines Eintrages $\mathcal{O}(\deg^-(j))$ (Eingangsgrad)

Laufzeit naive Abschätzung: $\mathcal{O}(n^3)$

Bessere Abschätzung der Laufzeit: $\mathcal{O}\left(\sum_{l=1}^n \sum_{j \in V} N_{j,l}\right)$ wobei $N_{j,l} = \deg^-(j)$

$$\sum_{l=1}^n \sum_{j \in V} N_{j,l} = \sum_{l=1}^n \sum_{j \in V} \deg^-(j) = \sum_{l=1}^n m = n \cdot m \quad (46)$$

Laufzeit Bellman-Ford Algorithmus: $\mathcal{O}(m \cdot n)$

5.10 Alle kürzesten Wege

Gegeben: Gerichteter Graph $G = (V, E)$, Gewichte $w : E \rightarrow \mathbb{R}$

Gesucht: Tabelle $D(i, j) =$ minimales Gewicht eines $i - j$ - Weges

Baseline: Bellman-Ford berechnet für ein $i \in V$ alle Einträge $D(i, 1), \dots, D(i, n)$ in Zeit $\mathcal{O}(n \cdot m)$.

Mit n Aufrufen erhält man nun die gesamte Laufzeit von $\mathcal{O}(n^2 m)$

Die Verbesserungsidee ist nun alle Einträge gleichzeitig zu berechnen.

Struktur: Optimaler Weg von i zu j besteht aus optimalen Wegen von i zu k und von k zu j mit halb so vielen Kanten.

Teilproblem: $T(i, j, r) =$ minimales Gewicht eines $i - j$ Weges mit $\leq 2^r$ Kanten.

Rekurrenz:

$$T(i, j, r) = \min_{k \in V} T(i, k, r-1) + T(k, j, r-1) \quad (47)$$

Berechnung: Initialisiere Einträge mit $r = 0$. Danach kann die Rekurrenz für Einträge mit $r = 1, r = 2$ usw. berechnen

Gesamtlaufzeit: $\mathcal{O}(n^3 \cdot \log n)$

Algorithmus 5.5: Floyd-Warshall

Gegeben: Gerichteter Graph $G = (V, E)$, Gewichte $w : E \rightarrow \mathbb{R}$

Gesucht: Tabelle $D(i, j) =$ minimales Gewicht eines $i - j$ - Weges.

Teilprobleme: $T^{(k)}(i, j) =$ minimales Gewicht eines $i - j$ - Pfads mit allen Zwischenknoten in $\{1, \dots, k\}$

Rekurrenz:

$$T^{(k)}(i, j) = \min\{T^{(k-1)}(i, j), T^{(k-1)}(i, k) + T^{(k-1)}(k, j)\} \quad (48)$$

Laufzeit: $\mathcal{O}(n^3)$ Einträge und $\mathcal{O}(1)$ Zeit pro Eintrag

Gesamtlaufzeit: $\mathcal{O}(n^3)$

The following was presented in the lecture on 06. December 2018.

6 Minimale Spannbäume

Gegeben: ungerichteter Graph $G = (V, E)$ Gewichte $w : E \rightarrow \mathbb{R}$

Definition 9. Ein Teilgraph T von G ist ein Spannbaum, falls T ein Baum ist und alle Knoten von G enthält. Gesucht: Spannbaum mit minimalem Gewicht.

6.1 Drei Äquivalente Formulierungen zum minimalen Spannbaum Problem

1. Wähle $n - 1$ Kanten mit minimalen Gewicht so aus, dass kein Kreis entsteht.
2. Wähle $n - 1$ Kanten mit minimalen Gewicht so aus, dass alle Knoten zusammenhängend werden.
3. Entferne $m - n + 1$ Kanten mit maximalen Gewicht so, dass alle Knoten zusammenhängend bleiben.

6.2 Gierige Algorithmen

Berechne Lösung als Sequenz von Entscheidungen. Treffe jede Entscheidung best-möglich im Hinblick auf vorher getroffene Entscheidungen, (Igoniere die Zukunft). Der gierige Algorithmus ändert niemals schon getroffene Entscheidungen.

Algorithmus 6.1: Kruskal

Problemformulierung: Wähle $n - 1$ Kanten mit minimalen Gewicht so aus, dass kein Kreis entsteht.

Gieriger Algorithmus: Beginne mit leerer Kantenmenge; füge immer die günstigste Kante hinzu, die keinen Kreis verursacht.

```
1  $T \leftarrow \emptyset$ 
2 for alle Kanten  $e \in E$  sortiert nach aufsteigendem Gewicht
3   if  $T + e$  keinen Kreis enthaehlt:
4     Setze  $T \leftarrow T + e$ 
```

Beobachtung: $T + e$ enthält keinen Kreis \Leftrightarrow die Endpunkte von e sind in verschiedenen Z-Komponenten von T

Idee: Datenstruktur für Zusammenhangskomponenten (ZHK)

make(V) Erstelle Datenstruktur für leeren Graphen auf V

find(u) Gibt eindeutigen Namen der ZHK von u zurück

union(u,v) Füge Kante $\{u, v\}$ hinzu/ vereinige ZHKs von u und v

```
1 make(V)
2 for alle Kanten  $e \in E$  sortiert nach aufsteigendem Gewicht  $e=\{u, v\}$ 
3   if find(u) ≠ find(v):
4     union(u, v)
```

Algorithmus 6.2: Prims

Problemstellung: Wähle $n - 1$ Kanten mit minimalen Gewicht so aus, dass alle Knoten zusammenhängend werden.

Gieriger Algorithmus: Beginne mit beliebigen Startknoten s ; füge immer die günstigste Kante hinzu, die einen neuen Knoten zur Zusammenhangskomponente von s hinzufügt.

Algorithmus 6.3: Rückwärts-Kruskal

Problemstellung: Entferne $m - n + 1$ Kanten mit maximalen Gewicht so, dass alle Knoten zusammenhängend bleiben.

Gieriger Algorithmus: Beginne mit voller Kantenmenge; Entferne immer die teuerste Kante, so dass alle Knoten zusammenhängend bleiben.

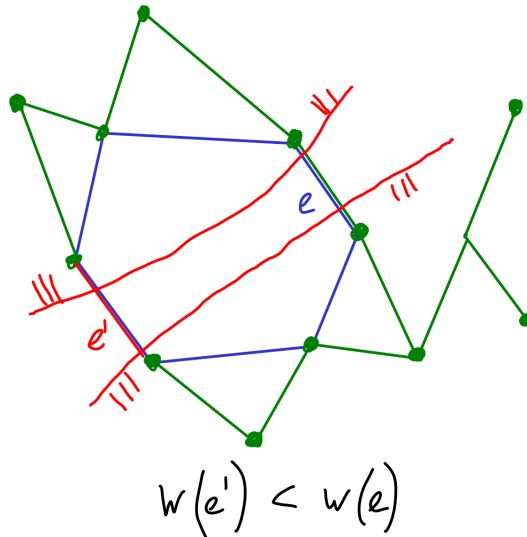
Alle drei gierigen Algorithmen sind korrekt und finden also immer einen minimalen Spannbaum.
Zwei allgemeine Prinzipien erlauben uns die Korrektheit dieser Algorithmen zu zeigen.
Annahme: Alle Kantengewichte sind verschieden

6.3 Kreisprinzip (Cycle Property)

Betrachte Kreis C im Graphen.

Sei e die Kante mit maximalem Gewicht in C ; dann enthält kein minimaler Spannbaum die Kante e .

Strategie: Betrachte Baum T mit e und konstruiere Baum T' mit kleinerem Gewicht.



Proof 6.1

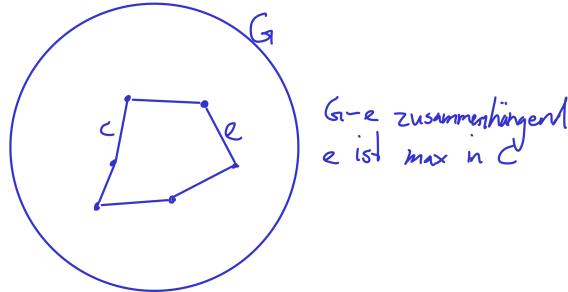
Wähle $T' = T - e + e'$ wobei $e' \in C - e$ beide Zusammenhangskomponenten in $T - e$ miteinander verbindet. \square

6.4 Korrektheit von Rückwärts-Kruskal durch Kreisprinzip

Theorem 6.1

Jede Kante, die der Rückwärts-Kruskal Algorithmus entfernt, erfüllt das Kreisprinzip.

Proof 6.2



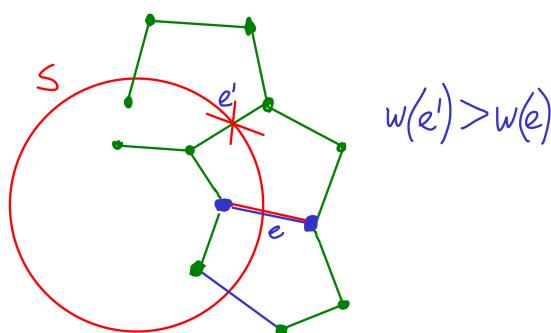
□

Annahme: Alle Kantengewichte sind verschieden.

6.5 Schnittprinzip

Betrachte Knotenmenge $S \subseteq V$ mit $0 < |S| < |V|$;
unter allen Kanten mit genau einem Endpunkt in S , sei e diejenige mit kleinstem Gewicht; dann enthält jeder minimale Spannbaum die Kante e
Strategie: Betrachte Baum T ohne e und konstruiere Baum T' mit kleinerem Gewicht.

Proof 6.3



Wähle $T' = T + e - e'$ wobei $e' \in T$ eine Kante auf dem Kreis in $T + e$ mit genau einem Endpunkt in S ist. □

6.6 Korrektheit von Prim durch Schnittprinzip

Theorem 6.2

Jede Kante, die Prims Algorithmus auswählt, erfüllt das Schnittprinzip.

Proof 6.4

Angenommen Prim wählt Kanten e_1, \dots, e_{n-1} nacheinander.
Erfüllt nun e_i das Schnittprinzip?

Betrachte Zusammenhangskomponente S des Startknoten im Graphen, der nur aus e_1, \dots, e_{i-1} besteht dann hat e_i das kleinste Gewicht unter allen Kanten mit genau einem Knoten in S

□

6.7 Korrektheit von Kruskal durch Schnittprinzip

Theorem 6.3

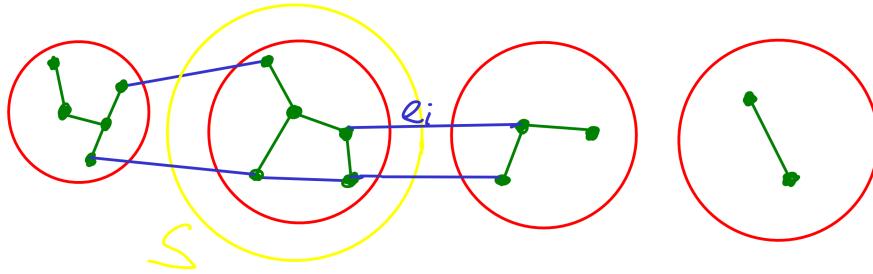
Jede Kante, die Kruskals Algorithmus auswählt, erfüllt das Schnittprinzip.

Proof 6.5

Angenommen Kruskal wählt Kanten $e_1, \dots, e_{n-1} \in E$ nacheinander.

Erfüllt nun e_i das Schnittprinzip?

Betrachte Zusammenhangskomponenten S_1, \dots, S_k im Graphen, der nur aus e_1, \dots, e_{i-1} besteht.



□

Laufzeit

Alle drei Algorithmen lassen sich mit polynomieller Laufzeit implementieren ($\mathcal{O}(n \cdot m)$)
Prim und Kruskal lassen sich sehr effizient implementieren mit Laufzeit $\mathcal{O}(m \cdot \log n)$

6.8 Union-Find

Union-Find: First Try

Speicher

`rep` $Array[u] =$ eindeutiger Repräsentant der ZHK von u .

Operationen

make(V) setze $rep[u] \leftarrow u$ für alle $u \in V$ $\mathcal{O}(n)$

find(u) return $rep[u]$ $\mathcal{O}(1)$

union(u,v) für alle $i \in V$, falls $rep[i] = rep[u]$, setze $rep[i] \leftarrow rep[v]$ $\Theta(n)$

Totale Laufzeit $\mathcal{O}(n^2)$

Union-Find: Second Try

Speicher

Array $rep[u]$ = eindeutiger Repräsentant der ZHK von u

Array $members[r]$ = Liste der ZHK representiert von r

Operationen

make(V) setze $rep[u] \leftarrow u$ für alle $u \in V$ $\mathcal{O}(n)$

find(u) return $rep[u]$ $\mathcal{O}(1)$

union(u,v) für alle $i \in members[r]$, setze $rep[i] \leftarrow rep[r']$

$members[r'] \leftarrow members[r'] \cup members[r]$

$members[r] \leftarrow \emptyset$ $\mathcal{O}(|ZHK(u)|)$

Totale Laufzeit $\mathcal{O}(n^2)$

Union-Find: Third Try

Speicher

Array $rep[u]$ = eindeutiger Repräsentant der ZHK von u

Array $members[r]$ = Liste der ZHK representiert von r

Array $size[r]$ = Grösse der ZHK representiert von r

Operationen

make(V) setze $rep[u] \leftarrow u$ für alle $u \in V$ $\mathcal{O}(n)$

find(u) return $rep[u]$ $\mathcal{O}(1)$

union(u,v) für alle $i \in members[r]$, setze $rep[i] \leftarrow rep[r']$

Betrachte Grössen der ZHks von u und v

Mache Rep. der grösseren ZHK zum Rep. der Vereinigung.

$\mathcal{O}(\min\{|ZHK(u)|, |ZHK(v)|\})$

6.9 Amortisierte Analyse

Erzeugte Datenstruktur durch $make(V)$

Theorem 6.4

Jede Sequenz von $n - 1$ union Operationen hat Gesamtlaufzeit $\mathcal{O}(n \log n)$

Beachte: Einzelne Operationen können Laufzeit $\Omega(\log n)$ haben.

Proof 6.6

sei N_u = Anzahl der Änderungen von $rep[u]$
dann gilt

$$\sum_{i=1}^{n-1} \min\{|ZHK_i(u_i)|, |ZHK_i(v_i)|\} = \sum_{u=1}^n N_u \quad (49)$$

Jede Änderung von $rep[u]$ verdoppelt Grösse der ZHK von u (da wir immer nur Rep. der kleinere ZHK ändern.)

also gilt: $N_u \leq \log_2 n$ für alle u
Gesamtlaufzeit:

$$\mathcal{O}\left(\sum_{u=1}^n N_u\right) \leq \mathcal{O}(n \log n) \quad (50)$$

□

The following was presented in the lecture on 13. December 2018.

7 Datenstrukturen für Graphen

gerichteter Graph $G = (V, E)$

$|v| = n, |E| = m$

Note 6. Was ist eine Datenstruktur: (Prüfungsfrage)

Implementierung einer Struktur mit der man Daten darstellen kann.

1. Adjazenzmatrix: eine $n \times n$ Matrix mit allen Verbindungen = 1 sonst 0.
2. Adjazenzliste: Ein Array mit allen Knoten und da hat es je eine LinkedList mit allen Kanten.

7.1 Kürzeste Wege

Gegeben eine Karte (Graph) mit Gewichten

Gesucht: Kürzester Weg $s \rightarrow t$

$G = (V, E, c)$ $c : E \rightarrow \mathbb{R}^+$

Table 9:

Operationen	AM	AL
alle Nachbarn von einem $u \in V$	$\mathcal{O}(n)$	$\mathcal{O}(\deg^+(u))$
$(u, v) \in E$	$\mathcal{O}(1)$	$\mathcal{O}(\deg^+(n))$
Kante (u, v) einfügen	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Vorgänger von V		

Table 10:

Graph	one-to-all	all-to-all
$G = (V, E)$	BFS $\mathcal{O}(m + n)$	$n \times$ BFS $\mathcal{O}(mn + n^2)$
$G = (V, E, c) c : E \rightarrow \mathbb{R}^+$	Dijkstra $\mathcal{O}(m + n \log n)$	$n \times$ Dijkstra $\mathcal{O}(mn + n^2 \log n)$
$G = (V, E, c) c : E \rightarrow \mathbb{R}$	Bellman-Ford $\mathcal{O}(mn)$	Floyd-Warshall $\mathcal{O}(n^3)$

7.2 One-To-All (Distanzgraph)

Zusammenhängender Distanzgraph: $G = (V, E, c)$, $c : E \rightarrow \mathbb{R}^+$

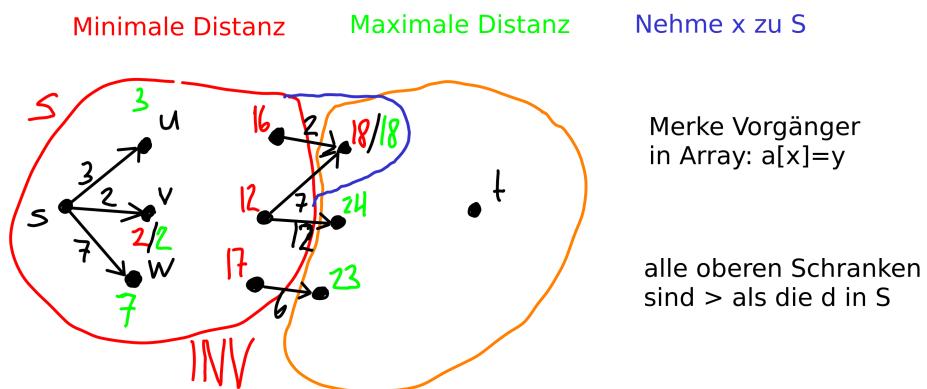
Gegeben ein Weg

Gesucht: Kürzester Weg $s \rightarrow t$

Länge: $\sum_{i=0}^{k-1} c((v_i, v_{i+1}))$

Kürzester Weg hat sicher kein Zyklus, da alle Kanten positiv sind. Weg hat $\leq n$ Knoten.

Algorithmus



Algorithmus 7.1

```

1  $S = \{s\}$  ,  $d(s) = 0$ 
2 while  $|S| < n$ 
3   choose  $(u, v)$  mit  $u \in S, v \in V \setminus S$  und  $d(u) + c(u, v)$  minimal
4    $S = S \cup \{v\}, d(v) = d(n) + c(u, v)$ 

```

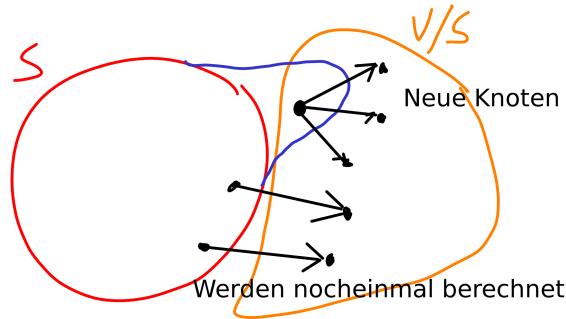
Wähle (u, v)

Schau alle Kanten an und prüfe $u \in V \setminus S$ $v \in V \setminus S$ Merke Minimum

$\Rightarrow \mathcal{O}(m)$

Gesamtlaufzeit: $\mathcal{O}(mn)$

Was ist noch nicht optimal?



Algorithmus 7.2: Dijkstra

1. Merke für alle Knoten obere Schranken, Anfang: $d(s) = 0$
 $d(n) = \infty$ für $u \neq S$
2. Finde neuen Knoten n mit Minimaldistanz
3. Update obere Schranken der Nachfolger von u

```

1  $S = \emptyset, d(s) = 0$ 
2  $d(n) = \infty$  for  $u \neq S$ 
3 while  $|S| < n$ :
4     finde  $v$  mit minimalem  $d$ 
5      $S = S \cup \{v\}$ 
6     for alle  $(u, w) \in E, w \in V \setminus S$ 
7          $d(w) = \min(d(w), d(v) + c(v, w))$ 

```

Analyse:

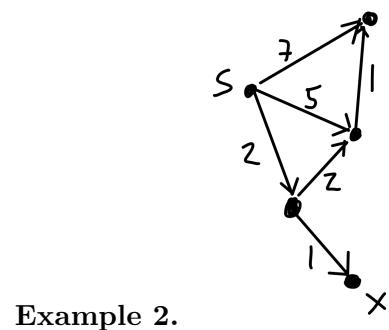
init: $\mathcal{O}(n)$

$n \times$ extract min $\mathcal{O}(n \log n)$ mit min Heap

$\leq m \times$ decrease dist $\mathcal{O}(m \log n)$

Gesamtlaufzeit: $\mathcal{O}((m+n) \log n)$

Mit Fibonacci Heap $\mathcal{O}(m + n \log n)$



It 1 $S = \{s\}$
 $d(n) = 7, d(v) = 5, d(w) = 2$

It 2 $S = \{s, w\}$
 $d(v) = 4, d(x) = 3$

It 3 $S = \{s, w, x\}$

It 4 $S = \{s, w, x, v\}$
 $d(n) = 5$

It 5 Fertig ✓

all-to-all, $c : E \rightarrow \mathbb{R}$

Idee: Mache alle Gewichte Positiv, dann Dijkstra

Idee 1: Subtrahiere kleinstes Gewicht. Funktioniert nicht, da die Distanzen verändert werden.

Algorithmus 7.3: Johnson

Idee 2: Definiere "Höhe" $h(n)$, $\forall n \in V$

neue Gewichte:

$$\hat{c}(u, v) = c(u, v) + h(n) - h(v) \quad (51)$$

Ziel für alle $\hat{c} \geq 0$ Dann gilt:

$$\hat{c}(s \rightsquigarrow t) = \sum_{i=0}^{k-1} \hat{c}(v_i, v_{i+1}) \quad (52)$$

$$= \sum_{i=0}^{k-1} c(v_i, v_{i+1}) + h(v_i) - h(v_{i+1}) \quad (53)$$

$$= \sum_{i=0}^{k-1} c(v_i, v_{i+1}) + \sum_{i=0}^{k-1} h(v_i) - h(v_{i+1}) \quad (54)$$

$$= c(s \rightsquigarrow t) + h(s) - h(t) \quad (55)$$

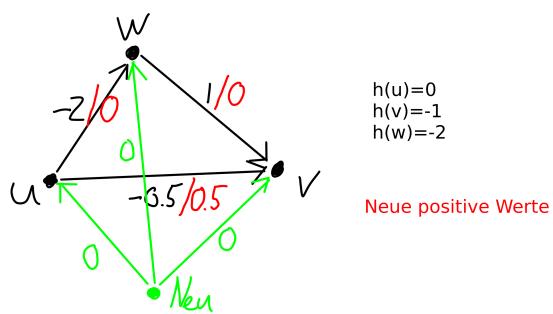
Es gilt auch, dass Zyklen ihr Gewicht behalten:

$$\hat{c}(s \rightsquigarrow s) = c(s \rightsquigarrow s) \quad (56)$$

$$h(u) = \text{kürzester Weg von neuem Knoten nach } u \quad (57)$$

$$h(v) \leq h(u) + c(u, v) \quad \forall u, v \quad (58)$$

$$\Leftrightarrow 0 \leq c(u, v) + h(u) - h(v) = \hat{c}(u, v) \quad (59)$$



Algorithmus 7.4

1. Neuer Knoten + Kanten $\mathcal{O}(n)$
2. h-Werte: Bellman-Ford $\mathcal{O}(m \cdot n)$
+ neue Gewichte
3. $n \times$ Dijkstra $\mathcal{O}(mn + n^2 \log n)$

The following was presented in the lecture on 20. December 2018.

8 Interessante Divide and Conquer Algorithmen (Nicht Prüfungsrelevant)

Example 3. Multiplikation ganzer Zahlen (n Ziffer) $\Theta(n^2)$

$$\begin{array}{r}
 \begin{array}{r} 62 \cdot 37 \\ \hline 14 \\ 14 \\ 16 \\ 18 \\ \hline 18 \\ \hline 2294 \end{array} \quad \begin{array}{l} bd \\ bd \\ (a-b)(d-c) \\ ac \\ ac \end{array}
 \end{array}$$

$$100ac + 10ac + 10bd + bd + 10(a-b)(d-c)$$

$$\left(\begin{array}{c|c} C_{11} & C_{12} \\ \hline C_{21} & C_{22} \end{array} \right) = \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array} \right) \left(\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array} \right)$$

$$T(n) = 3T\left(\frac{n}{2}\right) + c \cdot n \quad (60)$$

$$= 3^2 T\left(\frac{n}{4}\right) + \frac{3}{2}cn + c \cdot n \quad (61)$$

$$= 3^3 T\left(\frac{n}{8}\right) + \left(\frac{3}{2}\right)^2 cn + \frac{3}{2}cn + c \cdot n \quad (62)$$

$$= 3^{\log_2 n} T(1) + cn \cdot d \quad (63)$$

$$= 3^{\log_2 n} T(1) + \Theta(n) \quad (64)$$

$$= \Theta(n^{\log_2 3}) \approx \Theta(n^{1.7}) \quad (65)$$

(66)

$$p(x) = \sum_{i=0}^{n-1} a_i x^i = p_i(x)x^{\frac{n}{2}} + p_2(1) \quad (67)$$

$$q(x) = \sum_{i=0}^{n-1} b_i x^i = q_i(x)x^{\frac{n}{2}} + q_2(x) \quad (68)$$

8.1 Matrix Multiplikation

A,B $n \times n$ Matrizen $\Theta(n^3)$

$$C = AB \quad (69)$$

$$\begin{array}{c} U \begin{array}{|c|c|} \hline & & \\ \hline a & b & \\ \hline \end{array} \\ V \begin{array}{|c|c|} \hline & & \\ \hline c & d & \\ \hline \end{array} \end{array}$$

$$\begin{aligned} U &= a \cdot O^{\frac{n}{2}} + b \\ V &= c \cdot O^{\frac{n}{2}} + d \\ UV &= \underbrace{ac \cdot O^n}_{O^{\frac{n}{2}}} + \underbrace{ac O^{\frac{n}{2}}}_{(a-b)(d-c)} + \underbrace{bd O^{\frac{n}{2}}}_{bd} + bd \end{aligned}$$

$$T(n) = 3 T\left(\frac{n}{2}\right) + cn$$

$$C_{11} = A_{11}B_{11} + A_{12}B_{21} \quad (70)$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22} \quad (71)$$

$$T(n) = 8T\left(\frac{n}{2}\right) + c \cdot n^2 \Rightarrow \Theta(n^3) \quad (72)$$

Algorithmus 8.1: Strassen

$$M_1 = (A_{11} + A_{22})(B_{11} + B_{22}) \quad (73)$$

$$M_2 = (A_{21} + A_{22})B_{11} \quad (74)$$

$$M_3 = A_{11}(B_{12} + B_{22}) \quad (75)$$

$$\dots \quad (76)$$

$$M_7 = (A_{12} + A_{22})(B_{21} + B_{22}) \quad (77)$$

$$C_{11} + M_1 + M_4 - M_5 + M_7 \quad (78)$$

$$C_{12} = M_3 + M_5 \quad (79)$$

$$C_{21} = M_2 + M_4 \quad (80)$$

$$C_{22} = M_1 - M_2 + M_3 + M_6 \quad (81)$$

$$T(n) = 7T\left(\frac{n}{2}\right) + dn^2 \quad (82)$$

$$T(n) = \Theta(n^{\log_2 7}) \approx \Theta(n^{2.8}) \quad (83)$$

Bester Matrixmultiplikation Algorithmus, aber total unpraktisch:

$$\mathcal{O}(n^{2.37}) \quad (84)$$

Kleinste unterste Schranke

$$\Omega(n^2 \log n) \quad (85)$$

8.2 Auswahlproblem

Gegeben: Array $A[i], \dots, A[n]$ Zahlen

Gesucht: i-kleinstes Element Median $\frac{n}{2}$ -kleinstes

Idee 1: Sortieren, i-tes Ausgeben $\mathcal{O}(n \log n)$

Idee 2: i-Mal Minimum $\mathcal{O}(n \cdot i)$

Idee 3: Pivotieren

Pivotieren

1. Wähle Pivot p $\mathcal{O}(1)$
2. Zähle Elemente $\geq p$ $\mathcal{O}(n)$
3. Aufteilen $\mathcal{O}(n)$
4. $i = r$: return p
 $i < r$: Suche i-kleinstes links
 $i > r$: Suche (i-r)-kleinstes rechts

Laufzeit:

$$T(n) = T(n-1) + cn \quad (86)$$

Gesamtlaufzeit: $\mathcal{O}(n^2)$

Average case: $\Theta(n)$

Bei einem guten Pivot würde das Array immer kleiner werden. $0 < \epsilon < \frac{1}{2}$

$$T(n) \leq T((1 - \epsilon)n) + cn = T(qn) + cn \text{ mit } q < 1 \quad (87)$$

$$T(n) \leq T(1) + cn(1 + q + q^2 \dots) = \Theta(n) \quad (88)$$

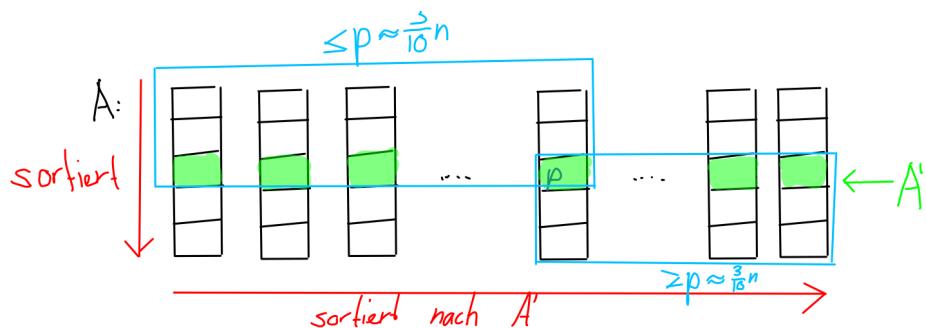
Algorithmus 8.2: Median der Mediane

Idee: Finde einen cleveren Pivot

- 1a) Betrachte Array in 5er Gruppen $\mathcal{O}(1)$
 - 1b) Berechne Median in jeder Gruppe (direkt) $\mathcal{O}(n)$
 - 1c) Sammle Gruppenmediane ($\lceil n \frac{n}{5} \rceil$) $\mathcal{O}(n)$
 - 1d) Berechne den Median von A' : Median ist Pivot
Rekursion: $\lceil \frac{1}{2} \lceil \frac{n}{5} \rceil \rceil$ - Element von A'
- 2-4 Aufteilen; Rekursion rechts oder links

Laufzeit:

$$T(n) \leq T\left(\frac{7}{10}n\right) + cn + T\left(\frac{n}{5}\right) \quad (89)$$



Zu zeigen $T(n) \leq an$, a ist konstant

Induktion nach n

Anfang: Wähle a so dass

$$T(n) \leq an \forall n \leq 100 \quad (90)$$

und $a \geq 10c$

Schritt A_{nn} : $T(k) \leq ak, \forall k < n$

$$T(n) \leq a \frac{7}{10}n + a \frac{2}{10}n + cn = a \frac{9}{10}n + cn \leq a \frac{9}{10}n + a \frac{1}{10}n = an \quad (91)$$

Algorithm and Data Structures - Exercise

Prof. Markus Püschel and Prof. David Steurer
TA: Tommaso Dorsi

Contents

0 Introduction	2
0.1 What is a graph	2
0.1.1 Connected Graph	2
1 Exercise (1/1, 1/2)	3
2 Exercise (1/1, 2/2)	5
3 Exercise (1/1, 2/2)	11
4 Exercise (1/2, 0/1)	19
5 Exercise (1/1, 2/2)	22
6 Exercise (1/1, 1/1, 1/1)	25
6.1 How to write correct pseudo code	30
6.2 Loop Invariants (6.4)	30
6.3 iPhone Drop Test (6.1.4)	30
7 Exercise (0/1, 0/1, 1/1)	31
8 Exercise (0/1, 2/2)	33
9 Exercise (1/1, 1/2)	38
9.1 LAS	44
9.2 Find longest common sequence	44
10 Exercise (0/1, 1/2)	44

11 Exercise (1/1, 1/1, 1/1)	49
12 Exercise (1/1, 2/2)	53
13 Programming Exercises	55
13.1 Wind Turbines	55
13.2 Algo Tower	56
13.3 Submatrix	56
13.4 Art Gallery	57
13.5 Dyno	57

The following was presented in the exercise on 24. September 2018.

0 Introduction

0.1 What is a graph

$$\begin{aligned} G &= (V, E) \\ (V', E') &= G' \subseteq G \text{ if } V' \subseteq V, E' \subseteq E \\ \forall u, v \in E', u, v \in V' \end{aligned}$$

0.1.1 Connected Graph

G is connected if $\forall u, v \in V$

$\exists u, v$ -path

(Maximal) Connected Component is a (maximal) connected subgraph

Acycle: Graph with no cycle.

Tree: Connected Acyclic graph.

Leaf: A vertex of degree 1 in a tree

1 Exercise (1/1, 1/2)

Exercise 1.1 a)

Base

$$A(n) = \sum_{i=1}^n i^3 = \frac{n^2(n+1)^2}{4} \quad (1)$$

$$A(1) = \sum_{i=1}^1 i^3 = \frac{1^2(1+1)^2}{4} = 1 \quad (2)$$

Hypothesis

We assume that $A(n)$ is true, then it is also true for $A(n+1)$

Induction Step

$$A(n+1) = \sum_{i=1}^{n+1} i^3 = \frac{(n+1)^2((n+1)+1)^2}{4} \quad (3)$$

$$A(n+1) = A(n) + (n+1)^3 = \frac{(n+1)^2((n+1)+1)^2}{4} \quad (4)$$

$$\frac{n^2(n+1)^2}{4} + (n+1)^3 = \frac{(n+1)^2((n+1)+1)^2}{4} \quad (5)$$

$$n^2 + 4(n+1) = (n+2)^2 \quad (6)$$

$$n^2 + 4(n+1) = (n+2)^2 \quad (7)$$

$$n^2 + 4n + 4 = n^2 + 4n + 4 \quad (8)$$

Exercise 1.1 b)

Base

$$A(n) = (1+x)^n = \sum_{i=0}^n \binom{n}{i} \cdot x^i \quad (9)$$

$$A(1) = (1+x)^1 = \sum_{i=0}^1 \binom{n}{i} \cdot x^i = 1+x \quad (10)$$

Hypothesis

We assume that $A(n)$ is true, then it is also true for $A(n+1)$

Induction Step

With the Binomial Coefficient:

$$\sum_{i=0}^n \binom{n}{i} \cdot x^i = \binom{n}{n} x^n + \binom{n}{n-1} x^{n-1} + \binom{n}{n-2} x^{n-2} + \dots + \binom{n}{0} x^0 \quad (11)$$

Our induction step:

$$A(n+1) = (1+x)^{n+1} = \sum_{i=0}^{n+1} \binom{n+1}{i} \cdot x^i \quad (12)$$

Insert the above formula:

$$A(n+1) = A(n) \cdot (1+x) = \binom{n+1}{n+1} x^{n+1} + \binom{n+1}{n} x^n + \binom{n+1}{n-1} x^{n-1} + \dots + \binom{n}{0} x^0 \quad (13)$$

Factor out $(x+1)$:

$$A(n+1) = A(n) \cdot (1+x) = (1+x) \cdot \left\{ \binom{n}{n} x^n + \binom{n}{n-1} x^{n-1} + \binom{n}{n-2} x^{n-2} + \dots + \binom{n}{0} x^0 \right\} \quad (14)$$

It is the same:

$$A(n+1) = A(n) \cdot (1+x) = (1+x) \cdot A(n) \quad (15)$$

The following was presented in the exercise on 01. October 2018.

1.1b

Auflösen von Binomialkoeffizienten in einer Summe

$$(1+x)^{k+1} = (1+x)(1+x)^k = (1+x) \sum_{i=0}^k \binom{k}{i} x^i = \sum_{i=0}^k \binom{k}{i} x^i + \sum_{i=0}^k \binom{k}{i} x^{i+1} \quad (16)$$

$$= \sum_{i=0}^k \binom{k}{i} x^i + \sum_{j=1}^{k+1} \binom{k}{j-1} x^j = \binom{k}{0} x^0 + \sum_{j=1}^k \binom{k}{j} x^j + \sum_{j=1}^k \binom{k}{j-1} x^j + \binom{k}{k} x^{k+1} \quad (17)$$

$$= \binom{k}{0} x^0 + \sum_{j=1}^k (\binom{k}{j} + \binom{k}{j-1}) x^j + \binom{k}{k} x^{k+1} = \binom{k+1}{0} x^0 + \sum_{j=1}^k \binom{k+1}{j} x^j + \binom{k+1}{k+1} x^{k+1} \quad (18)$$

$$(1+x)^{k+1} = \sum_{j=0}^{k+1} \binom{k+1}{j} x^j \quad (19)$$

1.2

- 1) We have a G-connected acyclic graph (tree) if $u \neq v$ - vertices, then there is a unique path between u and v.
- 2) If we add an edge, we get a cycle

The following was corrected in the exercise on 08. October 2018.

2 Exercise (1/1, 2/2)

Exercise 2.1.1. How many edges does an undirected graph of n vertices maximally contain? How many edges does a directed graph of n vertices maximally contain? (In both cases, assume that the graph does not contain loops.) (loop=Schleife)

An undirected graph of n vertices can contain at most $(n - 1) + (n - 2) + \dots + 1 = \frac{n*(n-1)}{2}$ edges. Therefore, a directed graph of degree n has at most $n * (n - 1)$ edges.

Exercise 2.1.2. What is the maximum number of edges in an undirected k -partite graph with $n = \sum_{i=1}^k u_i$ vertices, where $u_i > 0$ is the number of vertices in the i -th subset of this partition?

For $k = 2$: $|E| = u_1 * u_2$

For $k = 4$: $|E| = u_1 * (u_2 + u_3 + u_4) + u_2 * (u_3 + u_4) + u_3 * u_4$

$$|E| = \sum_{i=1}^k (u_i \left(\sum_{j=i+1}^k \right) u_j)$$

Exercise 2.1.3. Given an undirected clique G of size n , where n is an odd prime number. How many pairwise edge-disjoint simple cycles (i.e. cycles that use every vertex at most once) of length n does G contain?

A cycle is a sequence of vertices v_0, \dots, v_l where $v_l = v_0$. So the question is how many distinct sequences exist, that contain each vertex at most once and are of length n , so how many distinct sequences exist that visit each vertex once. For the first vertex v_0 there are n options. For the second vertex there are $n - 1$ options, for the third $n - 2$ etc. The number of possible combinations can therefore be described by the statement $n * (n-1) * (n-2) * \dots * 2 * 1 = n!$

Correction:

$$\forall i \in \{1, \dots, \frac{n-1}{2}\} : E^{(i)} = \{\{v_j, v_k\} : k - j \equiv imodn\}$$

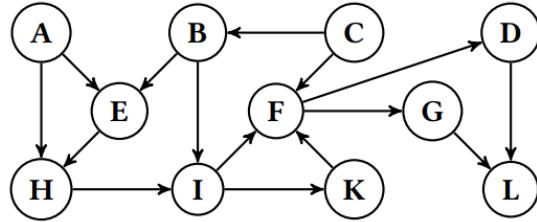
$$z_i(v_0, v_i, v_{2imodn}, \dots, v_{(n-1)imodn}, v_0)$$

$$\{\{v_j, v_k\} : k > j : k - j \leq \frac{n-1}{2} \rightarrow E^{(k-j)}$$

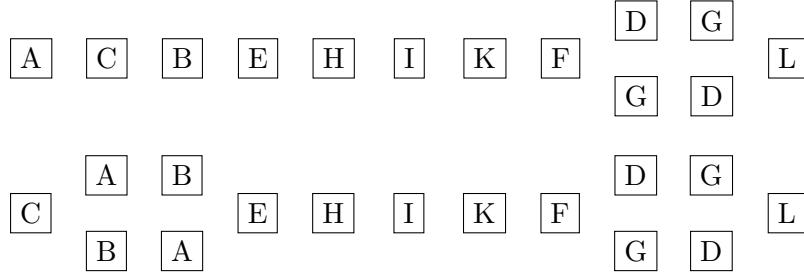
$$\text{Fall 2: } k - j > \frac{n-1}{2}$$

$$n - j + k \leq \frac{n-1}{2} \Rightarrow E^{(n-j+k)}$$

Exercise 2.2.1. How many topological orders does the following graph contain? List all topological orders.

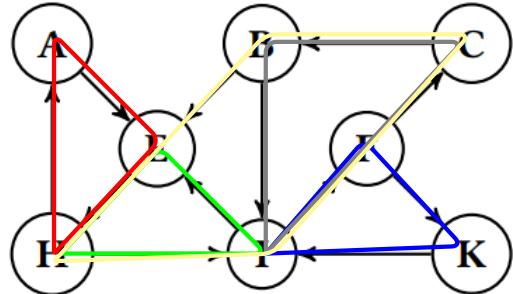
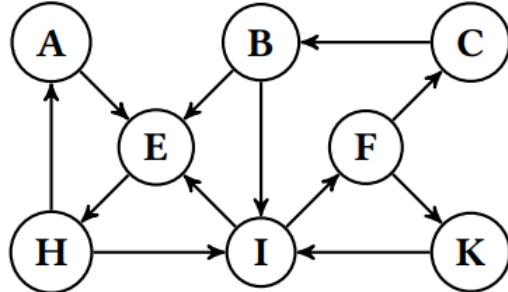


You have to start either at A oder at C. Therefore you have two options. If you start with A, then you have to choose C next, because E or H are not unlocked yet. And after B you can only do E. Then you are forced to choose H, I, K, F in this order. Then again you can chose either D or G, but are then forced to take the other one. And finally choose L. So you have the following topological orders:



So you have a total of 6 topological orders.

Exercise 2.2.2. Consider now the following graph $G = (V, E)$ and find a set $E' \subset E$ of minimum cardinality, such that $G' = (V, E \setminus E')$ can be topologically sorted. Justify your answer (i.e., why is it necessary to remove at least $|E'|$ edges?).



The total amount of circuits in this graph are 5 (not including those you can make in combination

of different circuits) The goal now is to cross out the least amount of edges, so you don't have any circuits anymore. If you only cross out one edge, you still have circuits left, because for example the red and the blue circuit have totally different edges. But if you cross the two Edges HE and IF, then it is not possible to create a circuit anymore.

Exercise 2.2.3. What is the maximum number of edges in a directed graph that can be topologically sorted? Formulate your claim for every positive integer n and prove your claim by using induction.

Hint: For each $n \in \{1, 2, 3, 4\}$ construct a graph with n vertices that is topologically sortable and contains the maximum number of edges. Then use these observations to derive your claim.)

Base

The maximum number of edges can be calculated as follows, with n being the number of vertices:

$$A(n) = \frac{n(n-1)}{2} \quad (20)$$

$$A(2) = \frac{2(2-1)}{2} = 1 \quad (21)$$

Hypothesis

If it is true for $A(n-1)$ then it is true for $A(n)$.

Correction:

Für ein $k \in \mathbb{N}_{>1} : G = (V, E)$ mit $|V| = n$ gilt: G hat topologische Sortierung: $|E| \leq \frac{k(k-1)}{2}$

Step

$$A(n-1) + (n-1) = A(n) \quad (22)$$

$$\frac{(n-1)(n-1-1)}{2} + (n-1) = A(n) \quad (23)$$

$$\frac{(n^2 - 3n + 2)}{2} + \frac{(2n-2)}{2} = A(n) \quad (24)$$

$$\frac{(n^2 - n)}{2} = A(n) \quad (25)$$

$$\boxed{\frac{n(n-1)}{2} = A(n)} \quad (26)$$

Correction:

$(k \rightarrow k+1)$: Betrachte $G = (V, E)$ mit $k+1 = |V|$ azyklisch: $\forall v \in V : \deg^-(v) = 0$ [Bemerke: $\deg^+(v) \leq R$]

Betrachte: $G' = (V \setminus \{v\}, E \setminus \{(v, u) : u \in \deg^+(v)\})$

$$|E| \leq \frac{k(k-1)}{2} = \frac{k(k+1)}{2}$$

Exercise 2.3.1. For any given number of vertices, you get the least amount of edges if you connect them to a tree. Then you can always remove a leaf and end up with a tree which has one edge less. At the end you have one vertex and zero edges.

Correction

Base

$\forall n$ zusammenhängende Graphen mit $|V| = n : |E| \geq n - 1$ $n = 1 : 0 \geq 1 - 1$

Hypothesis

$\forall l : 1 \leq l \leq k : G = (V, E)$ mit $|V| = l$ hat $\geq l - 1$ Kanten.

Step

$k \rightarrow k + 1$: Sei $G = (V, E)$ zusammenhängend mit $|V| = k + 1$

Wähle $v \in V$ und betrachte: $(V \setminus \{v\}, E')$

$C_1 = (V_1, E_1), \dots, C_m = (V_m, E_m)$ zusammenhängende Komponente. $|E| \geq \sum_{i=1}^m (|V_i| - 1) + \deg(v) = k - m + \deg(v) \geq k$ \square

Exercise 2.3.2. $G = (V, E), |V|$, Anzahl(zusammenhängende Komponenten) = m $2|E| \geq n - m$ $|E| = \sum_{i=1}^m |E_i| \geq \sum_{i=1}^m |V_i| - 1 = \sum_i |V_i| - m = n - m$ \square

Exercise 2.3.3. G: Wald mit m Bäumen

$$|E| = \sum_{i=1}^m |E_i| = \sum_{i=1}^m |V_i| - 1 = n - m$$

$G = (V, E), |V| = n$, Anzahl(zusammenhängende Komponenten) = $m, |E| = n - m$

Annahme $\exists G$ mit der Voraussetzung das nicht Teil des Waldes ist.

$B : C_1, \dots, C_m \Rightarrow \exists i \in \{1, \dots, m\} : |E_i| \geq |V_i|$

$$|E| = \sum_{j=1}^m |E_j| \geq |V_i| + \sum_{i \neq j} |V_j| - 1 = |V_i| + \sum_{j \neq i} |V_j| - (m - 1) = n + 1 - m \quad \text{Falsch}$$

Exercise 2.4.1. Let G be a directed acyclic graph which has a Hamiltonian path. What is the relationship between the set of Hamiltonian paths and the set of topological orderings of G? What are the sizes of these sets?

As proven in the lecture, every acyclic graph has a topological ordering.

Theorem From graphentheorie2.pdf, page 50. Jeder azyklische Graph hat eine topologische Sortierung.

So for the Graph $G = (V, E)$ there must exist some order of vertices v_1, v_2, \dots, v_n , where for every edge (v_i, v_j) $i < j$ and $V = \{v_1, \dots, v_n\}$. Since there exists a Hamilton path, in this topological ordering from every vertex v_i an edge has to go to the next vertex v_{i+1} , so $(v_i, v_{i+1}) \in E \forall i$ that satisfy $0 < i < n$. Otherwise a Hamiltonian Path would not be possible, since there is no way to go "backward" in a topological ordering, so no edge (v_i, v_j) with $i > j$ exists in G as we already stated above.

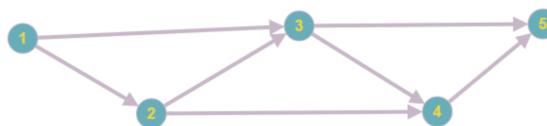


Figure 1: An example for a directed, acyclic graph that has a topological ordering

This in turn means that there can only be one single topological ordering, so the size of the set of topological orderings is 1. Because if we would change the position of any 2 vertices relative to each other, an edge that before satisfied the condition (v_i, v_j) $i < j$ would no longer fulfil

it, i.e. one edge would have to go "backwards", since the relative position of the nodes was changed. Therefore this new ordering would no longer be a topological ordering.

As we have already seen above there is only one Hamiltonian path for any topological ordering. So the size of the set of Hamiltonian paths is 1, which is equal to the size of the the set of topological sortings.

Exercise 2.4.2. Let G be a directed acyclic graph with no Hamiltonian paths. Can G have a unique topological ordering?

The answer to this question is no. As stated already in the last exercise, there is a theorem saying that any acyclic graph must have a topological ordering, so there exists at least one order of vertices v_1, \dots, v_n for this graph $G = (V, E)$ so that for every edge (v_i, v_j) $i < j$. But is this order unique?

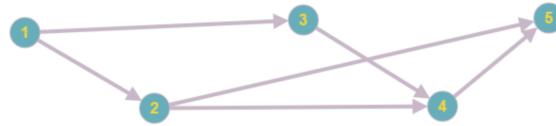


Figure 2: An example for a directed, acyclic graph that has no topological ordering

Unlike in the previous exercise, at least be at least one edge (v_i, v_{i+1}) cannot be an Element of E , or $\exists i (v_i, v_{i+1}) \notin E \wedge 0 < i < n$. Because of this, it's always possible to change an existing topological order, by swapping the position of any two vertices $v_i \& v_{i+1}$ where i satisfies the above condition. This is possible because there is no edge between these two vertices, so all edges still satisfy the condition (v_i, v_j) $i < j$, i.e. all precursors/successors of $v_i \& v_{i+1}$ are still before them/after them in this new topological order.

Eulerian tour

An Eulerian tour is a closed walk (Zyklus) that visits every edge exactly once. In this exercise, we ask you to prove that a connected graph contains an Eulerian tour if and only if it does not contain a vertex of odd degree.

Exercise 2.5.1. Prove that if a connected graph G contains an Eulerian tour, then G does not contain a vertex of odd degree.

A Eulerian Tour can therefore be described as an ordering of pairwise distinct edges $\{v_0, v_1\}, \{v_1, v_2\}, \dots, \{v_{l-1}, v_l\}$ of a Graph $G = (V, E)$, whereby every edge $\in E$ also has to be an Element of this order and $v_i \neq v_{i+2}$. Therefore, for every edge in this ordering $\{v_i, v_j\} \in E$ is true.

Proof. For every vertex in this graph it has to be true that incident edges are always found in pairs. As already stated, $\{v_0, v_1\}, \{v_1, v_2\}, \dots, \{v_{l-1}, v_l\}$ can describe an Eulerian tour. Now,

since an Eulerian can never contain the same edge twice, every vertex always needs to have an even number of edges. Looking at one vertex v_k , for every incident edge connecting this vertex to some vertex v_h , there also has to be an incident edge connecting it to some other vertex v_f . This is true also for more incident edges.

Put differently, there always needs to be an even number of incident edges so that Eulerian tour can go to this vertex and leave it again, else it would either not be possible to "leave" the vertex when all incident edges have been visited or it would not be possible to visit the last remaining edge. In both cases, an Eulerian tour therefore couldn't exist. \square

Exercise 2.5.2. An Eulerian tour is a closed walk (Zyklus) that visits every edge exactly once.

Prove that every connected graph without vertices of odd degree contains a Eulerian tour. Use mathematical induction on the number of edges.

Hint: Use the fact that every non-trivial connected graph without vertices of odd degree contains a cycle. Notice that this fact is a direct consequence of the fact that every non-trivial acyclic graph contains a leaf (which you proved in the previous exercise sheet).

Proof. Proof by Induction

Claim: $A(n)$: Every connected graph $G = (V, E)$ with at most n edges and without vertices of odd degree contains an Eulerian tour.

Base Case

Our Base Case is $A(3)$, since $A(0)$ doesn't contain any edges and therefore one can argue if an Eulerian tour really exists. Every connected graph with only one edge has 2 vertices of odd degree. Every connected graph with 2 Edges also has at least 2 vertices of degree 1, as long as we do not permit multiple edges between 2 vertices.

We prove the base case by a constructive existence proof: As shown by this image, there is a graph with 3 edges and only vertices of even degree

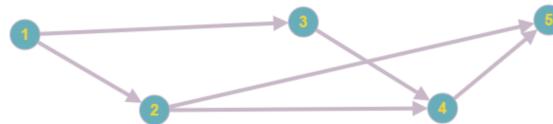


Figure 3: An example for a directed, acyclic graph that has no topological ordering

Induction Hypothesis

We assume that $A(n)$ is true for any n .

Induction Step

$$A(n-1) \rightarrow A(n) \forall n \leq 3$$

\square

The following was corrected in the exercise on 15. October 2018.

3 Exercise (1/1, 2/2)

Exercise 3.1.1. Write the following in Big \mathcal{O} Notation, simplifying them as much as possible.

$$1. \ 5n^3 + 40n^2 + 100$$

$$2. \ 1^2 + 2^2 + \dots + n^2$$

$$3. \ 2n \log_3(n^4)$$

1.

$$5n^3 + 40n^2 + 100 \leq 5n^3 + 40n^3 + 100n^3 = 145n^3 \leq O(n^3) \quad (C \geq 145) \quad (27)$$

2.

$$1^2 + 2^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6} = \frac{(n^2+n)(2n+1)}{6} \quad (28)$$

$$= \frac{2n^3 + 3n^2 + n}{6} \leq \frac{6n^3}{6} \leq O(n^3) \quad (C \geq 1) \quad (29)$$

3.

$$2n \log_3 n^4 = 8n \log_3 n = 8n \frac{\ln n}{\ln 3} = \frac{8}{\ln 3} n \ln(n) \quad (30)$$

$$\leq O(n \ln n) \quad (C \geq \frac{8}{\ln 3}) \quad (31)$$

Exercise 3.1.2. Place the following functions in order such that if f appears before g , it means that $f \leq O(g)$. If multiple functions have the same complexity, please indicate so.

$$2n^2 + n + n^5 \quad 2.1^n \quad \log n \quad n \quad n \log n \quad 2^n \quad (32)$$

$$\log n \quad n \quad n \log n \quad 2n^2 + n + n^5 \quad 2^n \quad 2.1^n \quad (33)$$

is the order according to the definition that

$$g(n) \leq O(f(n)) \Leftrightarrow \exists C > 0 \quad \forall n \geq 1 \quad g(n) \leq C * f(n)$$

Exercise 3.1.3. Prove that if $f_1(x), f_2(x) \leq O(g(x))$, then $f_1(x) + f_2(x) \leq O(g(x))$. Properties like this are useful with arguing about Big O Notation.

Proof. Direct Proof

If $f_1(x), f_2(x) \leq O(g(x))$, then $f_1(x) \leq C_1 * g(x)$ and $f_2(x) \leq C_2 * g(x)$. Therefore $f_1(x) + f_2(x) \leq C_1 * g(x) + C_2 * g(x) = (C_1 + C_2) * g(x) = C_3 * g(x) = O(g(x))$ \square

Exercise 3.1.4. Prove or disprove the following.

1. If $f_1(x), f_2(x) \leq O(g(x))$ then $\frac{f_1(x)}{f_2(x)} \leq O(1)$.

Assume $f_1(x), f_2(x), g(x) > 0$

2. If $f_1(x) \leq O(g(x))$ and $f_2(x) \leq O(\frac{1}{g(x)})$, then $f_1(x)f_2(x) \leq O(1)$.

Assume $f_1(x), f_2(x), g(x) > 0$

disproof. 1.

Disproof by counterexample

Consider $f_1(x) = x^2$ and $f_2(x) = x$. Then $f_1(x), f_2(x) \leq O(x^2)$. Therefore $\frac{f_1(x)}{f_2(x)} = \frac{x^2}{x} = x \notin O(1)$ \square

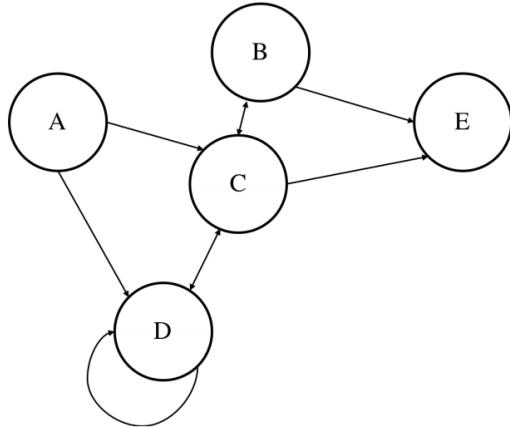
Proof. 2.

Direct proof

If $f_1(x) \leq O(g(x))$ and $f_2(x) \leq O(\frac{1}{g(x)})$, it holds that $f_1(x) \leq C_1 * g(x)$ and $f_2(x) \leq C_2 * \frac{1}{g(x)}$ according to definition. Therefore $f_1(x) * f_2(x) \leq C_1 g(x) * C_2 \frac{1}{g(x)} = C_1 C_2 * \frac{g(x)}{g(x)} = C_1 C_2 = C_3 = O(1)$ \square

Graph Representation

Exercise 3.2.1. For the given graph, express it as an adjacency matrix, and as adjacency lists



Adjacency Matrix

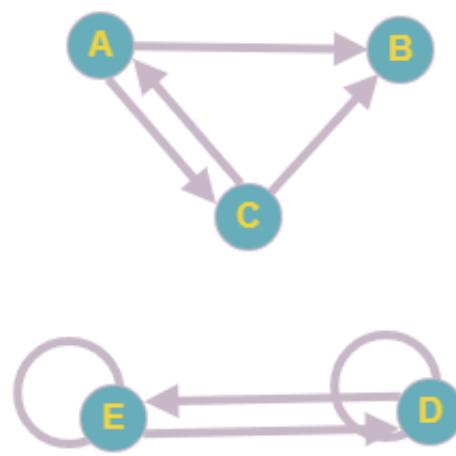
$$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Adjacency List

- A: C,D
- B: C,E
- C: B,D,E
- D: C,D
- E:

Exercise 3.2.2. For the given adjacency matrix, draw the graph that it represents, and also express it as adjacency lists.

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



Adjacency List

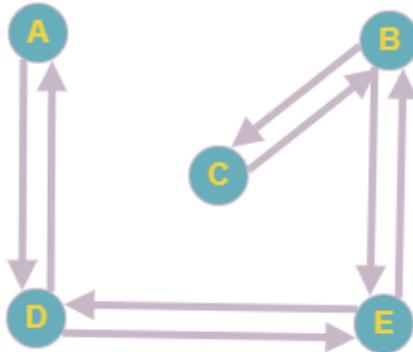
- A: B,C

- B:
- C: A,B
- D: D,E
- E: D,E

Exercise 3.2.3. For the given adjacency lists, draw the graph that it represents, and also express it as an adjacency matrix.

Adjacency List

- A: D
- B: C,E
- C: B
- D: A,E
- E: B,D



Adjacency Matrix

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Exercise 3.2.4. For the graphs in 3.2, answer the following:

- Is it directed or undirected?
- Is it a bipartite graph?

- Does the graph contain any loops?
- Does the graph contain any Euler cycles?
- Does the graph contain any Euler paths?

- 3.2.1

- Is it directed or undirected?
Directed
- Is it a bipartite graph?
No (The vertices C,B and E require at least 3 different colours)
- Does the graph contain any loops?
Yes (from D to D)
- Does the graph contain any Euler cycles?
No. Since A has 2 outgoing edges and no incoming edges, the path cannot contain both edges outgoing edges from A.
- Does the graph contain any Euler paths?
No. Since A has 2 outgoing edges and no incoming edges, the path cannot contain both edges outgoing edges from A.

- 3.2.2

- Is it directed or undirected?
Directed
- Is it a bipartite graph?
No (The vertices A,B and C require at least 3 different colours)
- Does the graph contain any loops?
Yes (from D to D and from E to E)
- Does the graph contain any Euler cycles?
No. The Graph is not connected
- Does the graph contain any Euler paths?
No. The Graph is not connected

- 3.2.3

- Is it directed or undirected?
It can either be seen as an undirected graph or a directed graph, were for every edge (vi, vj) also an edge (vj, vi) exists.
- Is it a bipartite graph?
Yes (Assign colour 1 to A,E and C and colour 2 to D and B)
- Does the graph contain any loops?
No
- Does the graph contain any Euler cycles?
Only if it is considered as a directed graph as described above. Then (A,D,E,B,C,B,E,D,A) is an Euler cycle. Else it's not possible to finish the Euler path on the same vertex.
- Does the graph contain any Euler paths?
Yes (A,D,E,B,C) or (A,D,E,B,C,B,E,D,A) respectively

Exercise 3.3

For this exercise, consider the greedy algorithm for graph coloring, where we assign a color to each vertex, and each vertex is assigned a different color than its neighbors. In the algorithm, we traverse the vertices in some order. For each vertex, inspect the color assigned to each adjacent vertex, taking note of which colors are already used. Finally, assign the first free color to the current vertex. If there are no free colors, add a new color to the list.

Exercise 3.3.1. Color the following graph using the greedy algorithm, visiting the vertices in the following order A, B, C, D, E . How many colors did you use? What is the minimum number of colors necessary to color the graph?

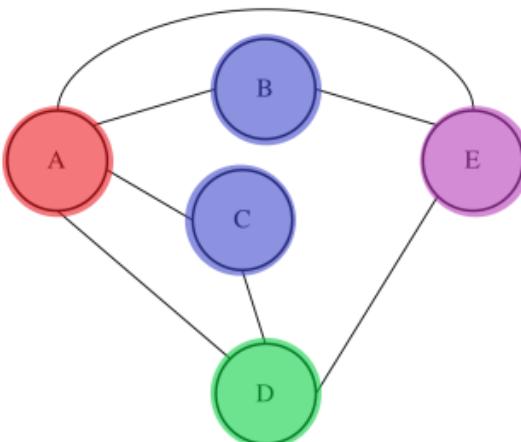
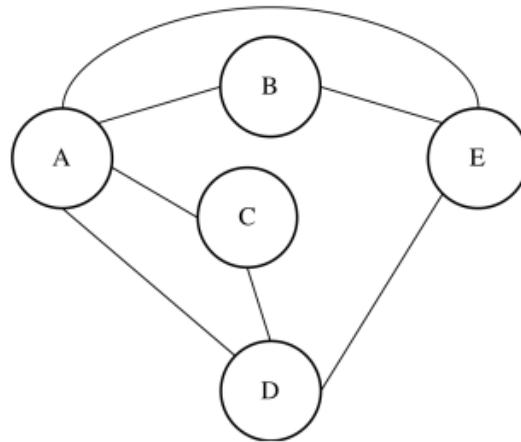


Figure 4: When executing the greedy algorithm, 4 colours were used

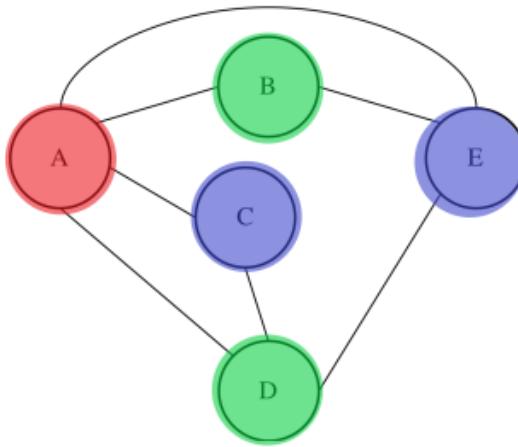


Figure 5: However the graph is actually tripartite, as can be seen from this solution. It can be easily seen that 3 is the smallest number of colours necessary, since the vertices A, C and D are all connected to each other

Exercise 3.3.2. Construct an undirected graph that, when visiting the vertices in some particular order, the greedy algorithm uses at least twice as many colors as the optimal coloring. Give that ordering of the vertices.

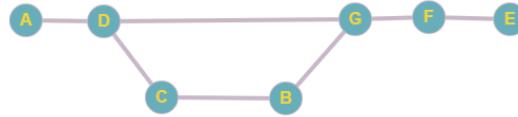


Figure 6: Consider the above graph being coloured using the greedy algorithm, which colours the vertices in alphabetical order

Exercise 3.3.3. Show that if every subgraph of an undirected graph has at least one vertex with degree at most k , then the graph can be colored with at most $k+1$ colors. Hint: use induction on the number of vertices in the graph.

Proof. Proof by induction

Consider some undirected Graph $G(V, E)$ with n vertices, where every subgraph of G has at least one vertex with degree at most k . The claim to be proven is that then the graph can be colored with at most $k+1$ colors. We proof this using induction on the number of vertices of G .

- **Base Case**

Let $n = 1$. Then G and every subgraph of G trivially have at least one vertex of degree 0, so $k = 0$. Therefore the graph should be $k + 1 = 1$ -colourable, which is true, since one colour is enough for a graph containing only 1 vertex.

- **Induction Hypothesis**

Assume that the property holds for any positive integer n . That is, every undirected graph with n vertices, of whom every subgraph has at least one vertex with degree at most k , can be colored with at most $k + 1$ colors.

- **Inductive Step**

We must show that the property holds for $n + 1$ if it holds for n . Consider an undirected Graph $G(V, E)$ with $n + 1$ vertices, where every subgraph has at least one vertex with degree at most k . Take any vertex v of G that is of the lowest degree. Let's remove v , with all its incident edges, from G . The resulting Graph G' is a subgraph of G . Therefore, all subgraphs of G' are also subgraphs of G and therefore have at least one vertex with degree at most k .

Therefore we can now use our induction hypothesis on G' and assume that G' is $(k + 1)$ -colourable. We now add the vertex v that was removed back in, together with all its incident edges. Since v is a vertex of lowest degree of G , it can be of degree at most k (since G has to contain at least one vertex of degree at most k). Therefore v can have at most k neighbours. Since we have $k + 1$ colours available, we therefore have at least 1 colour left to colour v . Therefore a valid $(k + 1)$ -colouring exists for G .

By the principle of mathematical induction, the claim is therefore true for any Graph G with n vertices.

□

Exercise 3.3.4. Suppose that every subgraph of an undirected graph $G = (V, E)$ has at least one vertex with degree at most k . Provide an algorithm to color G with at most $k + 1$ colors.

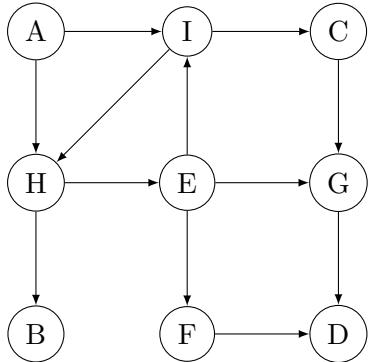
Start with an empty stack S and an undirected graph $G = (V, E)$, with the properties described above.

1. Repeat as long as there is more than one vertex in G :
 - (a) Let w be any vertex of G that is of minimal degree, together with all its incident edges
 - (b) remove w from G
 - (c) put w on S
2. Colour the last remaining vertex in G with 1 colour
3. Repeat as long as S is not empty:
 - (a) Let w be the top-most element of S
 - (b) remove w from S
 - (c) add w to G
 - (d) colour w with the first colour that is not used by any neighbour of w in G

The following was corrected in the exercise on 22. October 2018.

4 Exercise (1/2, 0/1)

Exercise 4.2



Exercise 4.2.1

The vertices are dequeued in the order $(A, H, I, B, E, C, F, G, D)$.

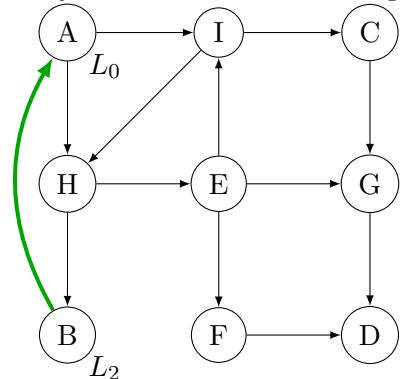
Exercise 4.2.2

$$\begin{aligned}
 L_0 &= \{A\} \\
 L_1 &= \{H, I\} \\
 L_2 &= \{B, C, E\} \\
 L_3 &= \{F, G\} \\
 L_4 &= \{D\}
 \end{aligned}$$

Exercise 4.2.3

- There can be an edge from level set L_i with $i \geq 2$ to a level set L_j with $j \leq i - 2$ in a directed graph.

Proof. Constructive existence proof by example:



□

- There can not be an edge from a level set L_i with $i \geq 2$ to a level set L_j with $j \leq i - 2$ in an undirected graph $G = (V, E)$.

Proof. Proof by contradiction: Assume that the statement is false, i.e. let $v_j \in L_j$ and $v_i \in L_i$ be vertices such that $\{v_j, v_i\} \in E$. Since $j \leq i - 2$, the shortest distance $d(v_j, v_i)$ is at least $i - j \geq 2$, and hence v_i and v_j cannot be neighbours. But this contradicts the assumption that $\{v_j, v_i\} \in E$, and hence the statement must be true. \square

- There can not be an edge from a level set L_i with $i \geq 0$ to a level set L_j with $j \geq i + 2$ in a directed graph $G = (V, E)$.

Proof. The proof is analogous to the proof for the previous point. It goes as follows: Assume that the statement is false, i.e. let $v_i \in L_i$ and $v_j \in L_j$ be vertices such that $(v_i, v_j) \in E$. Since $j - i \geq 2$, the shortest distance $d(v_i, v_j)$ is at least $j - i \geq 2$. But this contradicts the assumption that $(v_i, v_j) \in E$, and hence the statement must be true. \square

Exercise 4.2.4

Let $G = (V, E)$ be an undirected graph and L_0, L_1, \dots be the level sets computed by breadth-first search. Assume that there exists an edge between vertices within the same level set, i.e. let $u, v \in L_i$ be vertices such that $\{u, v\} \in E$. Since clearly $|L_0| = 1$, there is a path from the vertex $v_s \in L_0$ to u and v . And since $\{u, v\} \in E$, there exists a cycle in G of length $2i+1$, which is an odd number. Hence G cannot be bipartite when there is an edge between two vertices in the same level set, as a graph is bipartite if and only if it does not contain a cycle of odd length.

Correction:

Base case: L_0 v has color 1

Induction hyp: Every v at distance k has the same color

Induction step: If Level $k-1$ has the same color then Level k has the same color. Then (pseudo) proof by drawing

Exercise 4.3

$$\exists C : \forall n \geq 1 \quad D + g(u) \leq C \cdot F(u) + D \cdot f(u) \text{ and } D \leq D \cdot f(u) \quad \forall u \geq 1 \quad (34)$$

$$g(u) \leq C \cdot f(u) \text{ and } D \leq D \cdot f(u) \Rightarrow D + g(u) \leq (D + C) \leq f(u) \quad (35)$$

Exercise 4.3.2

It is not true

If n grows to ∞ then $f(n)$ would be 0 and $f(n)$ would be 1.

Exercise 4.3.3

(1) $\mathcal{O}(f)$ consists of all functions
 $g(n)$ so that $\exists C > 0 \forall n \geq 1$

$$g(n) \leq C \cdot f(n)$$

(2) $\mathcal{O}(f)$ consists of all functions
 $g(n)$ so that $\exists C > 0 \forall n_0 \geq 1$
so that $\forall n \geq n_0 \quad g(n) \leq C \cdot f(n)$

if $F(n) > 0 \forall n \geq 1$ then (1), (2) are equivalent.

$$(1) \Rightarrow (2) \rightarrow \exists C : g(n) \leq C \cdot f(n) \forall n \geq 1 \quad (36)$$

$$(1) \Leftarrow (2) \quad \text{Let } n_0 = C \quad g(n) \leq C \cdot f(n) \forall n \geq C \geq 1 \quad (37)$$

$$1 \Leftarrow \exists n_0 \geq 1, C > 0 : \quad \forall n \geq n_0 \quad g(n) \leq C \cdot f(n) \quad (38)$$

$$\forall n : 1 \leq n \leq n_0 \quad \frac{\max(f(n))}{g(n)} = D \quad f(n) \leq D \cdot g(n) = \frac{\max(f(n))}{g(n)} \cdot g(n) \quad (39)$$

$$C^* = \max(C, D) \quad \forall n \geq 1 \quad F(n) \leq C^* g(n) \quad (40)$$

Exercise 4.3.4

(1) and (2) are not equivalent.

Proof by counterexample: (We assume that (2) \Rightarrow (1))

$$f(n) = n - 1$$

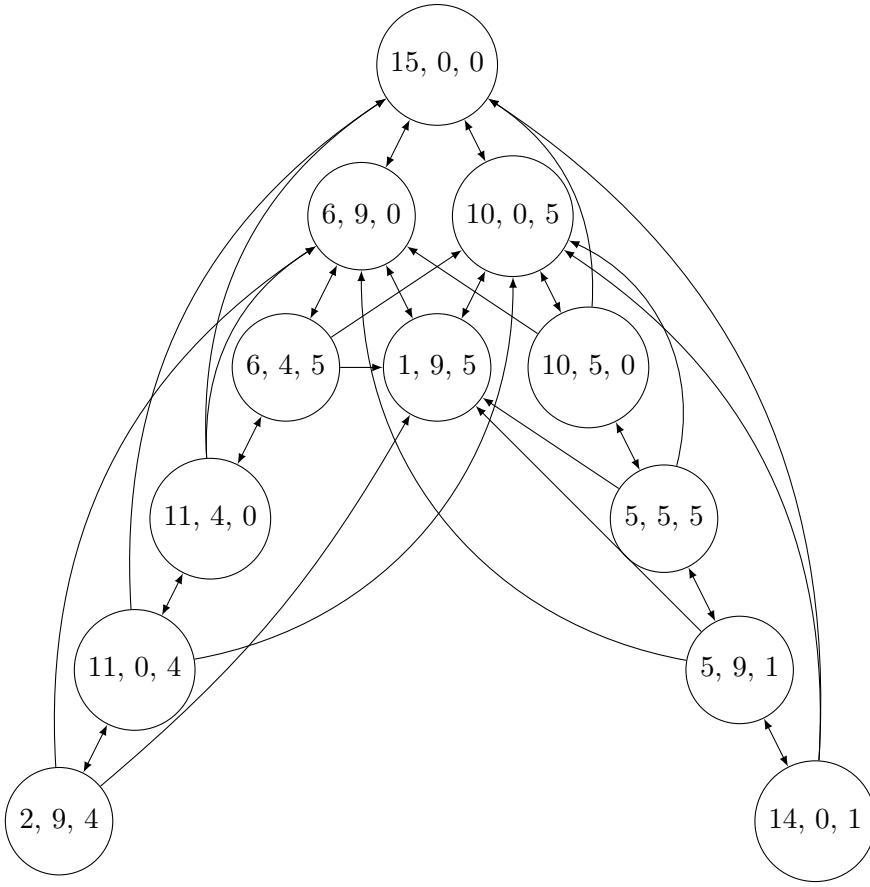
$$g(n) = n$$

for $n = 1$

$$1 \leq 2 \cdot 0 = 0$$

Exercise 4.4

1. Let $G = (V, E)$ be a graph of all valid states and the valid transitions between them. That is $V = \{(a, b, c) \in C_{15} \times C_9 \times C_5 \mid a+b+c = 15\}$ with $C_i = [0 \dots i]$, and $E = \{(u, v) \in V \mid [(a_u - a_v = b_v - b_u \wedge (a_u = 0 \vee a_v = 15 \vee b_u = 0 \vee b_v = 9)) \vee (a_u - a_v = c_v - c_u \wedge (a_u = 0 \vee a_v = 15 \vee c_u = 0 \vee c_v = 5))] \vee (b_u - b_v = c_v - c_u \wedge (b_u = 0 \vee b_v = 9 \vee c_u = 0 \vee c_v = 5))] \wedge (a_u - a_v \neq 0 \vee b_u - b_v \neq 0 \vee c_u - c_v \neq 0)\}$.
2. The following graph shows all valid states and the transitions between them up to the vertex labeled $(2, 9, 4)$, which is clearly the first vertex exhibiting the desired state, i.e. with the shortest distance from the starting node $(15, 0, 0)$.



It can be trivially verified that the graph shown exhibits all possible edges, namely by verifying $\forall v \in V : \deg_G^+(v) = 6 - |\{(v_i, v_j) \mid v_i \neq v_j \wedge (v_i = 0 \vee v_i = i)\}|$ where v_i denotes the i th value of the 3-tuple v .

Therefore it shows the shortest possible solution to reach (2,9,4), using 5 moves.

5 Exercise (1/1, 2/2)

Exercise 5.3.2

$$f(n) = 2f\left(\frac{n}{2}\right) + an^2, \quad f(1) = a, \quad n = 2^k$$

Which yields the following two cases:

$$f(n) = \begin{cases} a & \text{if } n = 1, \text{ so } k = 0 \\ 2f\left(\frac{n}{2}\right) + an^2 & \text{if } n = 2^k, \text{ with } k \geq 1 \end{cases}$$

Telescoping

$$\begin{aligned}
f(2^k) &= 2f(2^{k-1}) + a2^{2k} \\
&= 2(2f(2^{k-2}) + a2^{2(k-1)}) + a2^{2k} \\
&= 2(2(2 \dots) + a2^{2(k-1)}) + a2^{2k} \\
&= 2^k f(1) + \sum_{i=1}^k 2^{k-i} 2^{2i} a \\
&= 2^k a + \sum_{i=1}^k 2^{k+i} a \\
&= 2^k a + 2^k \sum_{i=1}^k 2^i a \\
&= na + n(2^{k+1} - 2)a \quad (\text{Recall that } n = 2^k, \text{ and that } 2^0 + 2^1 + \dots + 2^n = 2^{n+1} - 1) \\
&= na + na(2n - 2) \\
&= 2an^2 - na
\end{aligned}$$

Proof by Induction

To prove is that $A(n) := f(n) = 2an^2 - an$ is true for all $n \in \{n \mid \exists k > 0 (2^k = n)\}$.

Proof.

Base case : Since $f(n)$ is defined differently for $n > 1$ than for $n = 1$, let $n > 1$ be $n = 2^1 = 2$.
 $A(2)$ is trivially true: $f(2) = 2f(1) + a2^2 = 2a + 4a = 6a = 2a2^2 - 2a = 8a - 2a = 6a$.

Induction hypothesis : $A(\frac{n}{2}) \implies A(n)$

Inductive step :

$$\begin{aligned}
f(n) &= 2f\left(\frac{n}{2}\right) + an^2 && \text{(by definition of } f\text{)} \\
&= 2\left(2a\left(\frac{n}{2}\right)^2 - a\frac{n}{2}\right) + an^2 && \text{(by the induction hypothesis } A\left(\frac{n}{2}\right)\text{)} \\
&= 4a\frac{n^2}{4} - 2a\frac{n}{2} + an^2 \\
&= an^2 - na + an^2 \\
&= 2an^2 - na
\end{aligned}$$

Hence $A(n)$ is true for all n where $n = 2^k$ for some $k \geq 1$.

□

Exercise 5.5

Algorithm

The algorithm works by combining two algorithms studied in the lecture: Calculating prefix sums in an array, and finding the maximum subarray using an inductive algorithm. Ported into two dimensions, the first algorithm is used to calculate the vector sums of a row and all its neighbours individually, and the second algorithm is used to compute maximum subarray of each of these sums. The following is an implementation of the algorithm in python (2.7), which is close enough to pseudocode:

```
1 import numpy as np # for vector sum shorthand and better constant performance for the
2   addition (still O(n))
3
4 def max_subarray(A):
5     n = len(A)
6     accum = 0 # sum accumulator
7     accum_i = 0 # index tracker
8     rmax = (0, -1, -1) # rolling max tracker
9     for i in range(n): # O(n) as proved during the lecture
10         if accum <= 0:
11             accum_i = i
12         accum += A[i]
13         if accum < 0:
14             accum = 0
15         elif accum > rmax[0]:
16             rmax = (accum, accum_i, i)
17
18 return rmax
19
20
21 def max_submatrix(A):
22     n = len(A)
23     rmax_subm = (0, (-1, -1), (-1, -1)) # rolling max tracker, -1 indicates not found
24     for i in range(n):
25         accum = np.zeros(n) # sum accumulator
26         for j in range(i, n):
27             accum = np.add(accum, A[j]) # O(n)
28             m_subarr = max_subarray(accum) # O(n)
29             if m_subarr[0] > rmax_subm[0]: # O(1)
30                 rmax_subm = (m_subarr[0], (i, m_subarr[1]), (j, m_subarr[2]))
31
32 return rmax_subm
33
34 print(max_submatrix([[ -3,  8,  0,  1], [ 2, -3,  0,  0], [-4, -7,  0,  2], [ 1,  7,  2,  3]]))
```

Runtime

The runtime can easily be derived from looking at the loops in the program (the term c_2n for the submatrix sum was proved during the lecture):

$$\sum_{i=1}^n \sum_{j=i}^n c_1 n + c_2 n + c_3 \quad (41)$$

To prove is that (41) is in $\mathcal{O}(n^3)$.

Proof.

$$\begin{aligned}
\sum_{i=1}^n \sum_{j=i}^n c_1 n + c_2 n + c_3 &= \sum_{i=1}^n (c_1 n + c_2 n + c_3) \sum_{j=i}^n 1 \\
&= \sum_{i=1}^n (n - i)(c_1 n + c_2 n + c_3) \\
&= \sum_{i=1}^n n(c_1 n + c_2 n + c_3) - \sum_{i=1}^n i(c_1 n + c_2 n + c_3) \\
&< \sum_{i=1}^n n(c_1 n + c_2 n + c_3) \quad (\text{as } c_1, c_2, c_3, n > 0) \\
&= n^2(c_1 n + c_2 n + c_3) \\
&= n^3 c_1 + n^3 c_2 + n^2 c_3 \\
&\leq n^3(c_1 + c_2 + c_3) \\
&\leq \mathcal{O}(n^3)
\end{aligned}$$

□

The following was presented in the exercise on 29. October 2018.

5.4.1

1. $\rho_{\mathcal{O}}$: equivalence relation? It is not symmetric! ($n \leq \mathcal{O}(n^2)$ but $n^2 \notin \mathcal{O}(n)$)
2. ρ_{θ} : equivalence relation? Reflexiv, transivite and symmetric, so yes it is a equivalence relation.
3. ρ_{ω} : equivalence relation? No it is not symmetric.

5.4.2

1. Graph is not transitive.
2. Graph is not symmetric, but it is antisymmetric.

6 Exercise (1/1, 1/1, 1/1)

Exercise 6.1.4

Making two assumptions (derived intuitively from the preceding exercises), a simple algorithm can be derived:

- The algorithm must search from the bottom upwards, increasing by some step size
- The step size can be solely dependent on n , i.e. a constant for every n such that the current floor does not change the step size to calculate the next floor.

These assumptions result in the following algorithm of $\Theta(\sqrt[3]{n})$ complexity:

First, step upwards by $s_1(n) = n^{\frac{2}{3}}$, yielding a first lower and upper bound (when the phone breaks) after at most $\lfloor \frac{n}{s_1(n)} \rfloor = \lfloor n^{\frac{1}{3}} \rfloor$ steps. Then repeat on the resulting range with step size $s_2(n) = \sqrt{s_1(n)} = n^{\frac{1}{3}}$, yielding a second lower and upper bound after at most $\lfloor \frac{s_1(n)}{s_2 \circ s_1(n)} \rfloor = \lfloor n^{\frac{1}{3}} \rfloor$ steps. Then, on the final region, perform a linear search (step size 1), which trivially requires at most $s_2 \circ s_1(n) - 1 = n^{\frac{1}{3}} - 1$ steps. Hence the algorithm runs in $\Theta(\sqrt[3]{n})$. More precisely, the algorithm proceeds as follows:

Algorithm 1 Simple Breaking Point Search

```

procedure SEARCH( $n$ )
  for  $f \leftarrow n^{\frac{2}{3}}$ ;  $f \leq n$ ;  $f \leftarrow f + n^{\frac{2}{3}}$  do                                 $\triangleright$  First narrowing of bounds
    if not SURVIVES( $f$ ) then           $\triangleright$  Bounds are  $]f - n^{\frac{2}{3}}, f[$ 
      for  $g \leftarrow f - n^{\frac{2}{3}} + \sqrt[3]{n}$ ;  $g < f$ ;  $g \leftarrow g + \sqrt[3]{n}$  do       $\triangleright$  Second narrowing
        if not SURVIVES( $g$ ) then           $\triangleright$  Bounds are  $]g - \sqrt[3]{n}, g[$ 
          for  $i \leftarrow g - \sqrt[3]{n} + 1$ ;  $i < g$ ;  $i \leftarrow i + 1$  do       $\triangleright$  Exhaustive linear search
            if not SURVIVES( $i$ ) then
              return  $i$ 
    return  $f - 1$ 
return  $n$ 

```

However, the second assumption (namely that step size need not to change as the search proceeds) from which the above algorithm was derived does not lead to the optimal algorithm, although it does yield optimal worst-case complexity $\Theta(\sqrt[3]{n})$.

Optimal Algorithm

The optimal algorithm (still in $\Theta(\sqrt[3]{n})$, but with optimised constants) can be derived by closely examining the search tree. The following aims to explain how this algorithm can be discovered, without aspirations to a formal proof of correctness.

In the shown binary tree (Figure 7), the red edges denote where the phone breaks (these edges go to the left), whereas the black edges preserve the phone and hence allow it to be reused. An optimal algorithm must try to optimise for the height of this tree.

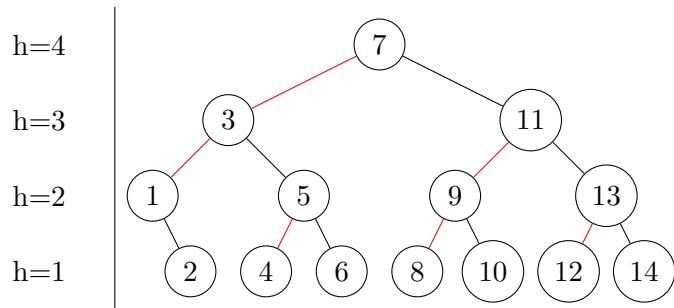


Figure 7: Tree for $n = 14$

Such an optimisation routine can be found by realising that the structure is inherently recursive:

When the phone does not break, the left side from the root can be discarded, yielding a new tree of height $h - 1$. Moreover, the height h can be computed from n by solving the equation $6n = h^3 + 5h$ for h and rounding up, i.e. $h \leftarrow \lceil h \rceil$ (this can be derived from the structure of the tree). Thus the task is to proceed in reverse, finding the root of the tree (i.e. where to drop the phone) from the precomputed height of the remaining tree and the number of phones left. In the following algorithm (Listing 1), this is implemented in the method `getRoot`.

From the Figure 6 one can observe the following formula, in case there are still three phones to use: Which corresponds to the following equation that yields the root vertex for a given height:

Table 1: Number of vertices on tree depth d

d	1	2	3	4	5	6	7	8	9
Vertices on this depth d	1	2	4	7	11	16	22	29	37

$$\text{getRoot}(h) = \frac{h^2 - h}{2} + 1$$

Thus the following algorithm optimally constructs and navigates such a tree for any n :

```

1 public class Phone {
2
3     static int nFloors = 41;
4     static int breakFloor = 37;
5
6     static boolean doesBreak(int floor) {
7         if(floor >= breakFloor) {
8             System.out.println("Phone broke from floor "+Math.min(floor, nFloors));
9             return true;
10        }
11        else {
12            System.out.println("Phone didn't break from floor "+floor);
13            return false;
14        }
15    }
16
17    /**
18     * Returns height from number of floors n. h = O(n^(1/3))
19     * @return The height h of the tree.
20     */
21    static int getHeight() {
22        int h = 1;
23        while(nFloors*6 > h*h*h + 5 * h) h++; // analytical computation would be very
24        ugly
25        return h;
26    }
27    /**
28     * Finds the root (top element) of the current subtree.
29     * @return Root vertex number of the current subtree
30     */
31    static int getRoot(int h, int phonesLeft) {
32        switch(phonesLeft) {
33        case 3:
34            return (h*h - h) / 2 + 1;
35        case 2:
36            return h;
37        }
38    }
39}
```

```

36     default:
37         return 1;
38     }
39 }
40
41 static int getBreakHeight() {
42     int phonesLeft = 3;
43     int treeHeight = getHeight();
44     int lowBound = 0;
45
46     while(phonesLeft > 0 && treeHeight > 0) {
47         int currentFloor = lowBound + getRoot(treeHeight, phonesLeft);
48         if(doesBreak(currentFloor)) {
49             phonesLeft -= 1;
50         }
51         else {
52             lowBound = currentFloor;
53         }
54         treeHeight -= 1;
55     }
56
57     return lowBound + 1 + treeHeight;
58 }
59
60 public static void main(String[] args) {
61     System.out.println(getBreakHeight());
62 }
63
64 }
```

Exercise 6.4

$\text{INV}(i)$:

1. $\text{randmax} := 0 \iff k = i$, and $\alpha[0 \dots -1] := \emptyset$. randmax is the sum $\sum_{j=k}^{i-1} \alpha[j]$ for the smallest k with $0 \leq k < i$ such that there exists no $l < i$ for which $\sum_{j=k}^l \alpha[j] < 0$ (which implies $\text{randmax} \geq 0$).
2. max is the maximum subarray sum for $\alpha[0 \dots i-1]$

$\text{INV}(n)$ implies the correct solution.

Proof. Proof by induction:

- $\text{INV}(0)$ is trivially true if $\alpha[0 \dots 0-1] := \emptyset$: $\sum \emptyset = 0 = \text{randmax} = \text{max}$.
- $\text{INV}(i) \implies \text{INV}(i+1)$:
 - (1) is true: By $\text{INV}(i)$, $\text{randmax} \leftarrow \sum_{j=k}^{i-1} \alpha[j] = \sum_{j=k}^i \alpha[j]$. There are then two cases: If $\text{randmax} < 0$, then $\text{randmax} \leftarrow 0 \implies k = i+1$, else randmax remains unmodified. Hence $\sum_{j=k}^i \alpha[j] = \text{randmax} \geq 0 \implies \text{INV}(i+1)$ is true in (1).

- (2) is true: By $\text{INV}(i)$, \max is the maximum subarray sum for $\alpha[0 \dots i - 1]$: Assume that there is new a maximum $m \neq \max$ for $\alpha[0 \dots i]$. Hence m cannot lie in solely in $\alpha[0 \dots i]$ (as $m \neq \max$ and $\text{INV}(i)$). Thus one term of m must be $\alpha[i]$, i.e. $m = \sum_{j=k}^i \alpha[j]$ with $k < i$ such that m is maximal. Thus there cannot be some l such that $\sum_{j=k}^l \alpha[j] < 0$, which is precisely the definition of randmax . Thus if there is a new maximum, it must be randmax , i.e. exactly when $\text{randmax} > \max$. Hence by lines 8–9, $\text{INV}(i + 1)$ is true in (2).
- Since $\text{INV}(i + 1)$ is true in both (1) and (2) whenever $\text{INV}(i)$ is true, $\text{INV}(i) \implies \text{INV}(i + 1)$.
- Since $\text{INV}(0)$ is true and $\text{INV}(i) \implies \text{INV}(i + 1)$, $\text{INV}(n)$ is true for any $n \in \mathbb{N}^+$. Hence $\text{INV}(n)$ implies the correct solution for any array $\alpha[0, 1, \dots, n - 1]$.

□

Exercise 6.5

Algorithm 2 TCP Congestion Avoidance

```

procedure FINDBANDWIDTH
     $(l, r) \leftarrow (1, 2)$                                  $\triangleright$  Assuming non-zero bandwidth
    while  $\text{ACK}(r)$  do                                     $\triangleright$  Check if packet gets ACK
         $(l, r) \leftarrow (r, 2r)$ 
    while  $l < r$  do                                          $\triangleright$  Binary Search
         $p \leftarrow \lfloor \frac{l+r}{2} \rfloor$ 
        if  $\text{ACK}(p)$  then
             $(l, r) \leftarrow (p + 1, r)$ 
        else
             $(l, r) \leftarrow (l, p - 1)$ 
    return  $l$ 

```

The first loop find the upper bound in $\mathcal{O}(\log b)$ as r is multiplied by 2 in each iteration until the bound is reached, i.e. $r = 2^c = b \implies c = \lceil \log_2 b \rceil$. The second loop is a binary search in the breadth $r - l = 2^c - 2^{c-1} = b - \frac{b}{2} = \frac{b}{2}$, hence it runs in $\mathcal{O}(\log b)$. Thus the overall upper bound is also $\mathcal{O}(\log b)$.

The same algorithm implemented in python 3.6:

```

1 def check_bandwidth():
2     bandwidth = 1 # current bandwidth guess
3
4     # Find upper bound
5     while ACK(bandwidth): # While packet goes through
6         bandwidth *= 2
7
8     bandwidth /= 2
9     pivot = bandwidth
10    while pivot > 1:
11        pivot /= 2
12        if ACK(bandwidth):
13            bandwidth += pivot

```

```

14     else:
15         bandwidth -= pivot
16
17     if not ACK(bandwidth):
18         bandwidth -= 1
19
20 return bandwidth

```

The following was presented in the exercise on 05. November 2018.

6.1 How to write correct pseudo code

Pseudo Code

$m \leftarrow 0$

if $m = 0$

6.2 Loop Invariants (6.4)

$INV(i = (m(i), rm(i))$

$m(i)$ max subarray sum of $a[0, 1, \dots, i - 1]$

$m(0) = 0$

$$rm(i) = \max \left\{ 0 \leq j < i \sum_{k=j}^{i-1} a[k], 0 \right\}$$

$rm(0) = 0$

1. $INV(0) = (m(0), rm(0)) \checkmark$

2. Suppose $INV(i)$ hold then it is true for $INV(i + 1)$

$$rm = \max \left\{ 0 \leq j < i \left\{ \sum_{k=j}^{i-1} a[k], 0 \right\} + a[i], 0 \right\} = \max \left\{ 0 \leq j < i \left\{ \sum_{k=j}^{i-1} a[k] + a[i], a[i], 0 \right\} \right\} \quad (42)$$

$$= \max \left\{ 0 \leq j \leq i \left\{ \sum_{k=j}^i a[k] + a[i], a[i], 0 \right\} \right\} \quad (43)$$

$$m(i+1) = \max \left\{ m(i), \max \left\{ 0 \leq j \leq i \sum_{k=j}^{i-1} a[k], 0 \right\} \right\} = \max \left\{ m(i), \max \left\{ 0 \leq j \leq i \sum_{k=j}^{i-1} a[k], 0 \right\} \right\} \quad (44)$$

6.3 iPhone Drop Test (6.1.4)

$$f_3(d) = 1 + f_2(d - 1) + f_3(d - 1) = 1 + \frac{d(d - 1)}{2} + f_3(d - 1) = \frac{d(d^2 + 5)}{6} \geq n \quad (45)$$

7 Exercise (0/1, 0/1, 1/1)

Exercise 7.2.2

To prove is that quicksort performs at most $2n^2$ comparisons.

Proof. Let $f(n)$ denote the number of comparisons performed by quicksort. As shown in the lecture, we get the following from telescoping:

$$f(n) = c \frac{n(n+1)}{2} \leq 2n^2$$

Base case :

$$f(0) = c \frac{0(0+1)}{2} = 0 \leq 0$$

Induction hypothesis : We assume $f(n) \leq 2n^2$ for all n .

Inductive step :

$$\begin{aligned} f(n+1) &= f(n) + cn && (\text{As the additional iteration performs } n \text{ comparisons}) \\ &= c \cdot \frac{n(n+1)}{2} + cn \\ &= cn \cdot \frac{n+3}{2} \\ &= c \cdot \frac{(n+1)(n+2)-2}{2} \\ &\leq 2(n+1)^2 \end{aligned}$$

Since $f(0) \leq 2(0^2)$ and $f(n) \leq 2n^2 \implies f(n+1) \leq 2(n+1)^2$, $f(n) \leq 2n^2$ for all $n \geq 0$.

□

Exercise 7.3

The key idea is to iteratively approach the solution from the left and the right (similar to how quicksort partitions an array), aborting if the pointer to the left steps past the pointer to the right.

Algorithm 3 Find resistors a and b such that $a + b = c$

```
procedure FINDRESISTORS( $A, c$ )                                ▷  $A$  is the box, given as an array of resistors
    HEAPSORT( $A$ )                                              ▷  $\mathcal{O}(n \log n)$ 
     $i \leftarrow 1$ 
     $j \leftarrow |A|$ 
    do
        while  $i < j \wedge A[i] + A[j] < c$  do
             $i \leftarrow i + 1$ 
        while  $j > i \wedge A[i] + A[j] > c$  do
             $j \leftarrow j - 1$ 
        if  $A[i] + A[j] = c$  then
            return True
        until  $i \leq j$ 
    return False

```

 ▷ Not Found

Exercise 7.4

The key idea is to always maintain a "bucket" (min-heap) of the m oldest elements, with the newest element p on top. When an element greater than this p enters the bucket, p is pushed out of the bucket (min-heap) and the new newest element in the heap makes its way to the top.

Algorithm 4 Find m oldest elements

```
procedure FINDOLDEST( $A, m$ )                                         ▷  $A$  is the box of coins
    heap  $\leftarrow$  MINHEAP( $m$ )                                         ▷ New, empty min heap of size  $m$ 
    for coin in  $A$  do
        if AGE(coin) > PEEK(heap) then
            if SIZE(heap)  $\geq m$  then
                POPTOP(heap)                                           ▷  $\mathcal{O} \log m$ 
                PUSH(heap, coin)                                       ▷  $\mathcal{O} \log m$ 
    return ASARRAY(heap)

```

The operations on the heap take $\mathcal{O} \log m$ time as proved during the lecture. Hence the total complexity of the algorithm is at most $\sum_{i=1}^n c_1 \cdot 2 \cdot \log m + c_2 \leq \mathcal{O} n \log m$.

Note that the heap structure used here is initialised as an empty min-heap. Hence if it is represented as an array (as shown during the lecture), it must track its used size (which is $< m$ until the heap is filled after the outer loop ran m times) to avoid representing invalid children.

The following was presented in the exercise on 12.11.18.

Note 1. Hint: Do MergeSort as Exam preparation like 7.1.1

Correction: 7.2.2

OBS1

OBS2

OBS3 If 2 does not happen the element gets swapped \Rightarrow 2 comparisions will be compared again

OBS4 The last comparison in either loop may involve an element already compared.

$$2n + 2x^2 + 2y^2 \leq 2n^2 \quad (46)$$

$$2n + 2x^2 + 2(n-x-1)^2 = 2n^2 + 2x^2 + 2 - 4n - 4xn + 4x = 2n^2 + \underbrace{4x(x+1-n-4n+2)}_{\leq 0} \leq 2n^2 - 4n + 2 = 2(n-1)^2 \leq 2n^2$$

$$(47)$$

Correction: 7.2.3

To maximize comparisions we ensure that the maximum element is always chosen as pivot.

$$A_0 = [3, 5, 7, 9, 10, 2, 6, 1, 8, 4] \rightarrow [3, 5, 7, 9, 4, 2, 6, 8, 10] \quad (48)$$

Recursive construction

$$A_0 = [3, 5, 7, 9, 11, \dots, n-1, n, \dots, n-4, x_{j-1}, n-2, x_j] \quad (49)$$

Correction: 7.4

Maintain a priority queue of size m

1. $\mathcal{O}(m)$
2. $\mathcal{O}(m)$
3. $\mathcal{O}((n-m)\log m)$

$$\Rightarrow \mathcal{O}((n \log m))$$

8 Exercise (0/1, 2/2)

Exercise 8.3

We create a binary tree with all multiples of three left of each-other in descending order (starting from the root). The numbers smaller than the smallest multiple of three are inserted to the left of this element. All non multiples of three will be found in a right sub-tree of a multiple of three (unless the tree doesn't contain any multiples of three). this ensures that all elements of the search tree are ordered not only from top to bottom but also from right (biggest numbers) to left(smallest number);

Therefore it is possible to answer the two questions pretty efficiently:

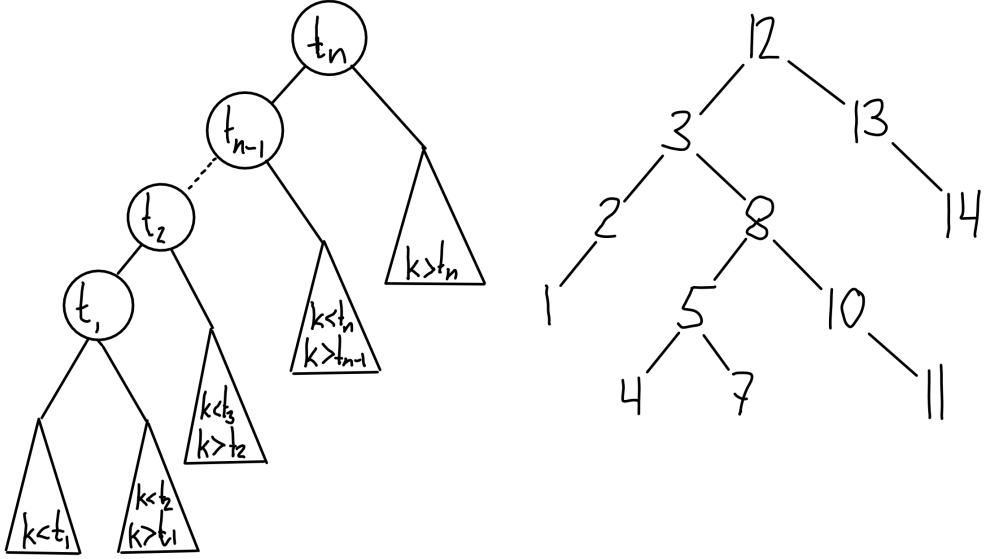


Figure 8: Advanced Search Tree

t_1 is the smallest multiple of three and t_n the biggest element multiple of three. Attached to each multiple of three is a sub-tree, whose elements are not multiple of three.

Bigger than k

The algorithm just goes to the left until it hits a number not a multiple of three and returns the counter.

Between k_1 and k_2

Almost identical to the above one.

Insertion

There are two cases to consider when inserting a number into the tree: the first case, is when the number isn't divisible by three, while the second case to consider is when it is divisible by three.

Let's first consider the case in which it is not. Just find the sub-tree (All numbers not divisible by three between t_i and t_{i+1}) where the number needs to be in and insert it.

If it is a multiple of three divide the sub-tree where the element would belong in two new sub-trees. Now insert the element divisible by three on the left branch with all the multiples of three. Attach the newly created sub-tree with the bigger numbers to this new element. Attach the newly created sub-tree with the lower numbers to the element where the old sub-tree was from.

A possible algorithm for insertion would be to attach a the number to be inserted to the left-most leaf of the tree and iterate up the tree until the parent is greater than the inserted value, then depending on weather the value is multiple of three or not, go down to right of the parent until the vertex is in the right position. This unfortunately gives us a run-time of $\mathcal{O}(n)$, which is really inefficient compared to the run-time $\mathcal{O}(\log n)$ of a normal search tree.

Removal

Again, there are two cases two consider, if it is a multiple of three or not.

In the second case, remove the element of the sub-tree the same way as in a normal binary search tree.

In the other way we need to combine the sub-tree of the element which gets removed to the sub-tree of the next smaller multiple of three.

To remove a value we must first find it, which is also very inefficient (worst case $\mathcal{O}(n)$). After a removal the tree must be reordered making the removal of a value even more costly.

Running time analysis

The worst case run time to find an element or to count all the multiples of three is $\mathcal{O}(n)$. This is, when every single number is divisible by three and they are all on the left branch. However, in practice the average run time of this algorithm is much faster, because we know exactly where the multiples of three are (to the left of each-other). This means that in a tree with both multiples and non-multiples of three it would give us a run time of $\mathcal{O}(m)$ compared to $\mathcal{O}(n)$ of a normal binary tree (where m is the number of multiples of three). However, the run-times of insertion and removal are worse compared to those of a normal search tree.

Exercise 8.5

Algorithm 5: OneColumnCandyCrush(S, T)

Input : A Stack S with n candies and an empty Stack T .
Output : The amount of uncrushed candies.

```

1  candy ← null;                                // the current candy type
2  triple ← 0;                                  // check if we had a triple
3  while  $S$  is non empty do
4      candy ←  $S.pop()$ ;
5      if candy equals  $T.top()$  then
6          |  triple ← triple + 1;
7      else
8          |  triple ← 1;
9      end
10     if triple equals 3 then
11         while candy equals  $S.top()$  do
12             |   $S.pop()$ ;
13         end
14         |   $T.pop()$ ;           // Before implementing consider edge cases!
15         |   $T.pop()$ ;           // for example when the stack  $T$  is empty.
16         |   $S.push(T.pop())$ ;
17         |   $S.push(T.pop())$ ;
18         |  triple ← 1;
19     else
20         |   $T.push(candy)$ ;
21     end
22 end
23 return  $T.size()$ ;

```

Running time analysis

In each iteration an element gets popped from the stack S . The only case the stack S can grow again, is when the triple counter detects three adjacent candies. In this case the stack S grows by two elements. In the worst case (e.g. DDCCBAAABCD) The algorithm would detect the triple A and then would push all the elements of stack T back to Stack S . Finally it would recognise the other triples and remove them. Therefore it would look at every element before the triple A three times. This means the algorithm iterates through the stack a maximum of three times, giving us a run-time of $\mathcal{O}(n)$.

For any given n the algorithm will take a maximum of $\frac{20}{3} \cdot n - 9 \quad \forall n \geq 3$, pop, top, push and size operations.

Iteration #	1	2	3	4	5	6	7	8	9	10	11	12	13
TOP	1	1	1	1	1	1	1	1	1	1	1	1	1
POP	1	1	1	1	1	1	3	2	1	1	3	2	1
PUSH	1	1	1	1	1	1	2	1	1	2	1	1	

20 push pop and TOP operations on A.

20 push pop and TOP operations on B .

10 push pop and ToP operations on C.

Figure 9: Example run-time

Exercise 8.4

Base Case

$$D(2) = 1 + D(0) \quad (50)$$

$$D(3) = 1 \quad (51)$$

Induction Hypothesis

Aussume it holds for some h-2 $D(h - 2) = \lfloor \frac{(h-2)}{2} \rfloor$

Induction Step

$$D(h) = \lfloor \frac{h}{2} \rfloor \quad (52)$$

$$D(h) = 1 + D(h - 2) = 1 + \lfloor \frac{(h-2)}{2} \rfloor = 1 + \lfloor \frac{h}{2} - 1 \rfloor = \lfloor \frac{h}{2} \rfloor \quad (53)$$

The following was presented in the exercise on 26. November 2018.

9 Exercise (1/1, 1/2)

Exercise 9.1

19	3	7	1	4	15	18	16	14	6	5	10	12	19	13	17	20	8	14	11
1	2	3	4	5	6	7	8												
19	7	15	18	12	19	17	20												
2	4	16	10	11	13	14													
1		14	8																
			6																
			5																

The longest ascending Subsequence is [1, 4, 5, 10, 12, 13, 17, 20].

Exercise 9.2

	-	7	6	3	2	8	4	5	1
-	0	0	0	0	0	0	0	0	0
3	0	0	0	1	1	1	1	1	1
9	0	0	0	1	1	1	1	1	1
10	0	0	0	1	1	1	1	1	1
8	0	0	0	1	1	2	2	2	2
7	0	1	1	1	1	2	2	2	2
1	0	1	1	1	1	2	2	2	3
2	0	1	1	1	2	2	2	2	3
6	0	1	2	2	2	2	2	2	3
4	0	1	2	2	2	3	3	3	3
5	0	1	2	2	2	3	4	4	4

All possible entries are :

1. 3845
2. 3245
3. 7645
4. 7245

Exercise 9.3

Explanation

1. *Definition of the DP-Table:* We create a two-dimensional table T that is indexed from 1 to n in one dimension and from 0 to 2 in the other. The entries $T[i, 0]$ (where $0 < i \leq n$) represent all the possible schedules where Alice does not go to the gym on the last day, the entries $T[i, 1]$ represent all possible schedules where Alice does go to the gym on the last

indexes	1	2	3	4	5	6	7	8	9	10	...	n
row 0	1	2	3	5	8	13	21	34	55	89	...	$T[n, 0]$
row 1	1	1	2	3	5	8	13	21	34	55	...	$T[n, 1]$
row 2	2	3	5	8	13	21	34	55	89	144	...	$T[n, 2]$

Table 2: This table represents the DP-table described below. The red arrows represent the calculation of the $T[i, 1]^{th}$ entry and the green arrows represent the calculation of the $T[i, 0]^{th}$ entry.

day, finally the entries $T[i, 2]$ represent the sum of the two aforementioned possibilities (it is not necessary to store this last row, but it makes the explanation easier).

2. *Calculation of an entry:* The entry $T[i, 0]$ will simply be the $T[i - 1, 2]^{th}$ entry. The $T[i, 0]^{th}$ entry as mentioned before, represents the schedules where Alice does not go to the gym on the last day, because Alice can't go the gym twice in a row, the $T[i, 0]$ will be all previous possibilities of the $i - 1^{th}$ day plus an extra day where Alice does not go to the gym.

We use a similar logic to calculate the $T[i, 1]^{th}$ entry. The $T[i, 1]^{th}$ entry will simply be $T[i - 1, 0]$. This is because Alice can't go to the gym twice in a row therefore all schedules where she goes to the gym on the last day will simply be the possibilities where she doesn't go the gym on the last day of the $i - 1^{th}$ day plus the extra day where she does go to the gym.

The edge case $i = 1$ is very easy to solve. We need to return all possible schedules for one day, therefore Alice can either go to the gym or not go to the gym meaning we initialize the table by setting $T[1, 0] = 1$, $T[1, 1] = 1$ & $T[1, 2] = T[1, 1] + T[1, 0]$.

3. *Calculation order:* To calculate the i^{th} column we must first calculate the $i - 1^{th}$ column therefore we calculate the entries by ascending i , we calculate the columns from top to bottom (from index 0 to index 2).
4. *Reading the solution:* the solution will be found in the $T[n, 2]^{th}$ entry.

Observations

If we look at the rows we notice that the possibilities follow the fibonacci sequence.

Pseudocode & running-time

Algorithm 5 ALICE-GYM-SCHEDULE(n)

```

 $F[1] \leftarrow 2$ 
 $F[2] \leftarrow 3$ 
for  $i \leftarrow 3, \dots, n$  do
     $F[i] \leftarrow F[i - 1] + F[i - 2]$ 
return  $F[n]$ 

```

As already stated during the lecture the running-time of this algorithm is $\mathcal{O}(n)$. Because the last row in the table is shifted we set $F[1]$ to 2 and $F[2]$ to 3.

Exercise 9.4

Algorithm 6 Black and White Stones

```

 $factorial1plus2 \leftarrow 1, factorial2 \leftarrow 1, counter1plus2 \leftarrow 1, counter2 \leftarrow 1$   $\triangleright \mathcal{O}(1)$ 

procedure PRECALCULATION( $m$ )  $\triangleright \mathcal{O}(m)$ 
     $factorial1 \leftarrow 1$ 
    for  $i = 1$  until  $m - 1$  do
         $factorial1 \leftarrow factorial1 \cdot i$ 
         $counter1plus2 \leftarrow counter1plus2 + 1$ 
     $factorial1plus2 = factorial1$ 
    return  $factorial1$ 

procedure FACTORIALCHECK( $p, q$ )  $\triangleright \mathcal{O}(1)$ 
    if  $counter1plus2 \leq p + q$  then
         $factorial1plus2 \leftarrow factorial1plus2 \cdot counter1plus2$ 
         $counter1plus2 \leftarrow counter1plus2 + 1$ 
    if  $counter2 \leq q$  then
         $factorial2 \leftarrow factorial2 \cdot counter2$ 
         $counter2 \leftarrow counter2 + 1$ 

procedure CALCPOSSIBILITIES( $p, q, factorial1$ )  $\triangleright \mathcal{O}(1)$ 
    FACTORIALCHECK( $p, q$ )
    FACTORIALCHECK( $p, q$ )
    if  $q < 0$  then
         $counter1plus2 \leftarrow counter1plus2 - 1$ 
         $factorial1plus2 \leftarrow \frac{factorial1plus2}{counter1plus2}$ 
    if  $p + q = 0$  then
        if  $p = 1$  then
            return 0
        else
            return 1
    else
        return  $\frac{factorial1plus2}{factorial1 \cdot factorial2}$ 

procedure TIMPOSSIBILITIES( $m, n, factorial1$ )  $\triangleright \mathcal{O}(n)$ 
     $postTim \leftarrow 1$ 
    if  $m \bmod 2 = 0$  then
        if  $n \bmod 2 = 0$  then
             $i \leftarrow -1$ 
        else
             $i \leftarrow 1$ 
    else
         $i \leftarrow 0$ 
    while  $i \leq n$  do
         $postTim \leftarrow postTim + CALCPOSSIBILITIES(m - 1, i, factorial1)$ 
         $i \leftarrow i + 2$ 
    return  $postTim$ 

procedure TIPERCENTAGE( $m, n$ )  $\triangleright \mathcal{O}(m + n)$ 
     $factorial1 \leftarrow PRECALCULATION(m)$ 
    return  $TIMPOSSIBILITIES(m, n, factorial1) / CALCPOSSIBILITIES(m, n, factorial1 \cdot m)$ 

```

White Stones	Total Possibilities					Tim Possibilities					For Calculation				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
0	1	1	1	1	1	(1)	(0)	(1)	(0)	1	1	0	1	(0)	1
1	2	3	4	5	6	1	2	1	4	1	1	2	1	4	1
2	3	6	10	15	21	2	2	7	(4)	16	1	2	6	(4)	15
3	4	10	20	35	56	2	6	7	24	16	1	4	6	20	15
4	5	15	35	70	126	3	6	22	(24)	86	1	4	15	(20)	70
5	6	21	56	(126)	252	3	12	22	(80)	86	1	6	15	(56)	70

Table 3: The function PreCalculation is responsible for the circles in the first row. The function TimPossibilities calculates 80.

Explanation

The algorithm is called by TimPercentage(numberOfBlackStones, numberOfWhiteStones). Then the algorithm keeps track of 3 factorials: factorial1, factorial2 and factorial1and2.

The idea behind the algorithm is that one can compute the number of possibilities by $\frac{(m+n)!}{m! \cdot n!}$, which is equivalent to $\frac{factorial1and2}{factorial1 \cdot factorial2}$. For example the total possibilities for Tim and Gordon to draw the stones if you have 4 black stones and 5 white stones is $\frac{(4+5)!}{4! \cdot 5!} = 126$. This is exactly the number which gets returned by the CalcPossibilities call in the TimPercentage function.

To calculate the total possibilities for Tim to win the game is a bit trickier. It first calculates the number for 4 black stones and 0 white stones, which is zero. Next it calculates the possibilities for 4 black stones and 2 white stones which is $\frac{((4-1)+(2-1))!}{(4-1)!(2-1)!} = 4$. Then it adds it to the previous calculation of 4 black stones and 0 white stones. Now it does the same for 4 white stones and finally 6 white stones. Now it returns the result, because the possibility for 6 white stones is the same as for 5 white stones.

Running Time Analysis

As one can see from the pseudo code the run time is $\mathcal{O}(m + n)$. The reason why this algorithm is so fast, is because it keeps track of the 3 different factorials as explained above.

The memory utilization is $\mathcal{O}(1)$.

Appendix: Python Code for Black and White Stones (Does not need to be corrected)

```

1 factorial1plus2 = factorial2 = counter1plus2 = counter2 = 1
2
3 def pre_calculation(p):
4     global factorial1plus2, counter1plus2
5
6     factorial1 = 1

```

```

7     for i in range(1, p):
8         factorial1 *= i
9         counter1plus2 += 1
10
11    factorial1plus2 = factorial1
12    return factorial1
13
14 def factorial_check(p, q):
15     global factorial1plus2, factorial2, counter1plus2, counter2
16
17     if counter1plus2 <= p + q:
18         factorial1plus2 *= counter1plus2
19         counter1plus2 += 1
20
21     if counter2 <= q:
22         factorial2 *= counter2
23         counter2 += 1
24
25 def old_current_possibilities(p,q):
26     k1=k2=k3=1
27
28     for i in range(1,p+q+1):
29         k1*=i
30
31     if p+q==0:
32         k1=0
33
34     for i in range(1,p+1):
35         k2 *= i
36
37     for i in range(1,q+1):
38         k3 *= i
39
40     return k1,k2,k3
41
42 def current_possibilities(p, q, factorial1):
43     global factorial1plus2, factorial2, counter1plus2, counter2
44
45     factorial_check(p, q)
46     factorial_check(p, q)
47
48     if q<0:
49         counter1plus2-=1
50         factorial1plus2 = int(factorial1plus2/counter1plus2)
51
52     if p + q == 0:
53         if p==1:
54             return 0
55         else:
56             return 1
57     else:
58         return int(factorial1plus2 / (factorial1 * factorial2))
59
60 def tim_possibilities(p, q, factorial1):
61     possibilities_tim = 0
62     if p % 2 == 0:
63         if q % 2 == 0:
64             i = -1

```

```

65     else:
66         i = 1
67     else:
68         i = 0
69
70     while i <= q:
71         possibilities_tim += current_possibilities(p - 1, i, factorial1)
72         i += 2
73
74     return possibilities_tim
75
76 def tim_percentage(m, n):
77     factorial1 = pre_calculation(m)
78     return 1.0 * tim_possibilities(m, n, factorial1) / current_possibilities(m, n,
79                                 factorial1 * m)
80 print(tim_percentage(4, 5))

```

1. Definition of DP table
2. Calculation of an entry
3. Calculation order
4. Reading the solution

9.1 LAS

DP 1 Contains the length of a minimum LAS ending at index i.

DP 2 Contains the index of the previous entry in the longest subsequence.

9.2 Find longest common sequence

$$L(i, j) = \max\{\delta_{lm} + L(i-1, j-1), L(i-1, j), L(i, i-1)\} \quad (54)$$

The following was presented in the exercise on 03. December 2018.

10 Exercise (0/1, 1/2)

10.4 Game with coins

Definition of the DP table

$$c = [20, 20, 500, 20, 20, 20, 20, 100, 100, 100, 5, 5, 5, 5, 5, 1, 1, 1, 1, 1] \quad (55)$$

	r																				
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	20	X	500	X	X	520	X	X	540	X	X	705	X	X	740	X	X	760	X	X	803
2	0	20	X	20	X	X	520	X	X	540	X	X	720	X	X	740	X	X	680	X	X
3	0	500	X	500	X	X	520	X	X	700	X	X	720	X	X	660	X	X	802	X	
4	0	20	X	20	X	X	200	X	X	220	X	X	160	X	X	302	X	X	341		
5	0	20	X	20	X	X	200	X	X	140	X	X	225	X	X	321	X	X			
6	0	20	X	100	X	X	120	X	X	140	X	X	301	X	X	340	X				
7	0	20	X	100	X	X	120	X	X	220	X	X	320	X	X	325					
8	0	100	X	100	X	X	200	X	X	300	X	X	305	X	X						
9	0	100	X	100	X	X	200	X	X	205	X	X	200	X	X						
10	0	100	X	100	X	X	105	X	X	100	X	X									115
11							0	5	X	5	X	X	10	X	X	15	X	X			
12								0	5	X	5	X	X	10	X	X	15	X			
13									0	5	X	5	X	X	10	X	X	15			
14									0	5	X	5	X	X	10	X	X				
15										0	5	X	5	X	X	10	X				
16											0	5	X	5	X	X	6				
17											0	1	X	1	X	X					
18												0	1	X	1	X					
19													0	1	X	1					
20														0	1	X					
21															0	1					

Table 4: Game with coins

The DP-table in this exercise is an $(n \times n)$ table where each initialized entry gives you the maximum value the player could have gotten in the game of coins until that point.

Calculation of an entry

$$T(l, r) = \max\{T(l + 1, r - 2) + c(l), T(l + 2, r - 1) + c(r)\} \quad (56)$$

In each entry you have to choose the maximum number between two values of the DP table, those two values are the maximum value of coins you could have gotten on the game until that point adding one of the possible coins to be picked at the current turn. We then choose the maximum possible value and initialize our entry as that value.

Calculation order

We start by initializing all values $T(i,i)$ as $c(i)$ and from there we start calculating and for each initialized entry we calculate the value of each entry on the DP-table that has an initialized value in $T(l+1,r-2)$ and $T(l+2,r-1)$, until we have finally calculated the value of the top right corner, which would be our final result.

Reading the solution

The solution can easily be read from the top right of the DP-table, it displays the maximum possible amount gained through playing the game.

Running time and memory analysis

To calculate an entry of the table you only need a running time of $\mathcal{O}(1)$, since you always have to compare two entries on the table, taking then the maximum value. In total there are up to $\sum_{i=1}^n \frac{i}{3} \approx \frac{\frac{n}{3} + (\frac{n}{3} + 1)}{2}$ possible entries to compute and therefore the total running time is $\mathcal{O}(n^2)$. For the memory since we need a DP-table of $(n \times n)$ we would need a memory space of $\mathcal{O}(n^2)$.

2. Report the order of coins

If we want to report the order of coins we would have to pick in order to get the maximum value possible we would have to start at the top right corner of the DP-table, where our maximum value lies, from there we check if $T(l,r) - c(l) = T(l+1,r-2)$ or $T(l,r) - c(r) = T(l+2,r-1)$ if it is true for $T(l+1,r-2)$ then return that you have to pick the coin on the right and repeat the process for $T(l+1,r-2)$, otherwise return that you have to pick the coin on the left and continue the process with $T(l+2,r-1)$, continue doing this until there are no more values to choose from, which would mean you have gotten to the last coin.

Since you have to go through 2 coins every time you have to choose and you have to choose $\frac{n}{3}$ times, you would have to perform $\frac{2n}{3}$ operations, which would give the process a running time of $\mathcal{O}(n)$.

10.5 Mechanical Computer

Definition of the DP table

We have the following list of breakpoints:

$$L = [1, 2, 5] \quad (57)$$

	r								
	1	2	3	4	5	6	7	8	9
1	0	2 (1)	5 (2)	6 (2)	7 (2)	12 (2)	14 (5)	15 (5)	16 (5)
2		0	2 (2)	3 (2)	4 (2)	9 (2)	10 (5)	11 (5)	12 (5)
3			0	0	0	4 (5)	5 (5)	6 (5)	7 (5)
4				0	0	3 (5)	4 (5)	5 (5)	6 (5)
5					0	2 (5)	3 (5)	4 (5)	5 (5)
6						0	0	0	0
7							0	0	0
8								0	0
9									0

Table 5: Mechanical Computer

Calculation of an entry

$$T(l, r) = \min\{T(l, b) + T(b + 1, r) + r - l - 1\} \quad \forall b \mid b \in L \text{ and } l \leq b < r \quad (58)$$

If no breakpoint exists between l and r then $T(l, r) = 0$

The number in the brackets behind each number is the b -value the min function has returned the value.

Calculation order

First, initialize the DP-table with $T(i,i)=0$. Next one calculates the entries for $T(i,i+1)$, then $T(i,i+2)$ and so on until the top right of the DP-Table is reached.

Reading the solution

The minimum number of units of time needed to complete the operation can be read from the top right of the DP-Table. To get the order of the operations one needs to follow back the number in the brackets. In this example (5) is the b -value for 16. So break at 5 first and then check for $r=5$ where we find 7 (2). So next break at two and finally at one.

Running time and memory analysis

To calculate an entry of the table $\mathcal{O}(n)$ time is needed, because it is possible to have n breakpoints. Therefore one would have to do a calculation for every single breakpoint and then take the minimum value. in total there are up to $\sum_{i=1}^n i \approx \frac{n^2}{2}$ possible entries to compute and therefore the total running time is $\mathcal{O}(n^3)$

As you can see in the DP-table the total space needed is $\mathcal{O}(n^2)$, because there are $\frac{n^2}{2}$ possible entries $+n$ to compute the minimum for each entry. $\Rightarrow \frac{n^2}{2} + n = \mathcal{O}(n^2)$

10.3

Suppose:

$$\sum_S \leq \sum_T \quad (59)$$

Claim: If (i) holds then

$$\sum_S +A[i] \leq \sum - \sum_S \quad (60)$$

$$\sum_S + \sum_T +A[i] \leq \sum \quad (61)$$

$$\sum_S +A[i] \leq \sum - \sum_T \leq \sum - \sum_S \quad (62)$$

We proved $*$ holds at each step i

10.5

One needs to Proof by contradiction that it is the smallest possibility to set the breakpoints.

The following was presented in the exercise on 10. December 2018.

Correction: Homework 10.2.2

$$T(j, l) := \min \text{ weight of } s \cdot j \text{ walk of length } \leq l \quad (63)$$

$$S \text{ source negative cycle} \Leftrightarrow \exists j \in V : T(j, n-1) \neq T(j, n) \quad (64)$$

(\Leftarrow) Suppose: $\exists j$ so that $T(j, n-1) \neq T(j, n)$

Claim: $T(j, n-1) < T(j, n)$ and \exists a $s - j$ path of length n and most $T(j, n)$

$$\underbrace{s \rightarrow v_1 \rightarrow \dots \rightarrow v}_{S_{sv}} \underbrace{\dots \rightarrow v}_{S_{vv}} \underbrace{\dots \rightarrow j}_{S_{vj}} \quad (65)$$

$$S_{sv} + S_{vj} > S_{sv} + S_{vj} + S_{vv} \quad (66)$$

(\Rightarrow) We have a negative cycle.

Show $\exists j \in V$ so that $T(j, n-1) \neq T(j, n)$

Suppose $\nexists j \in V$ so that $T(j, n-1) \neq T(j, n)$

Then $w(C) \geq 0$

$$v_0 \rightarrow \dots \rightarrow v_{i+1} \rightarrow \dots \rightarrow v_{k-1} \rightarrow \underbrace{v_k}_{=v_0} \quad (67)$$

walk $s - v_{i+1}$

$$T(v_i, n-1)w(v_i, v_{i+1}) \quad (68)$$

$$T(v_{i+1}, n) \leq T(v_i, n-1) + w(v_i, v_{i+1}) \quad (69)$$

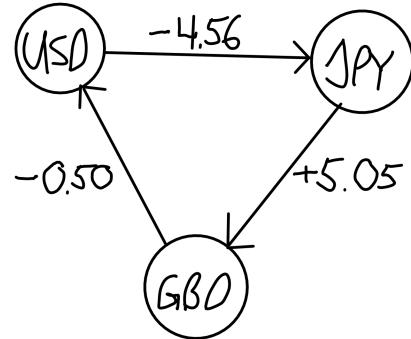
$$T(v_{i+1}, n-1) \leq T(v_i, n-1) + w(v_i, v_{i+1}) \quad (70)$$

$$\underbrace{\sum_0^{k-1} T(v_{i+1}, n-1)}_{= \sum_0^k T(v_i, n-1)} \leq \sum_0^{k-1} T(v_i, n-1) + \sum_0^{k-1} w(v_i, v_{i+1}) \quad (71)$$

$$0 \leq \sum_0^{k-1} w(v_i, v_{i+1}) \quad (72)$$

11 Exercise (1/1, 1/1, 1/1)

11.2.1 Arbitrage



	\$	¥	£
\$	1	95.729	-
¥	-	1	0.00638
£	1.65133	-	1

Modeling as a graph problem

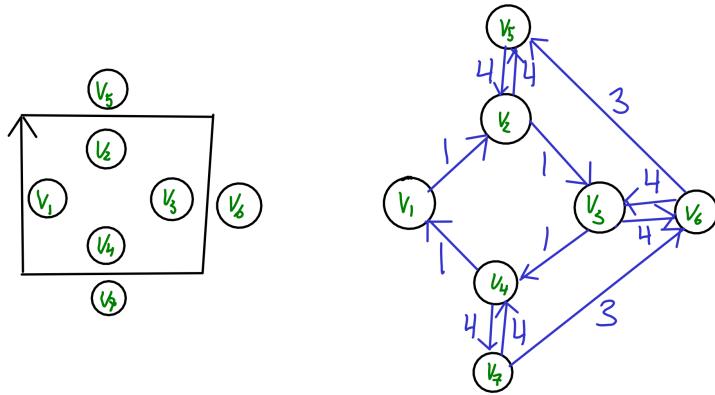
In this exercise in question the graph would be composed of n vertices, which would be the number of currencies, each one would be representing one currency. Then each of the vertices would have a directed edge with each of the vertices representing a currency for which we know

the exchange rate in respect to the currency of the vertex we are in, each of these edges would be representing the transition of value between the two currencies. To calculate the weight of the edge between two vertices u and v we have to use \ln , if we take the $\ln(\frac{u}{v})$ and we would get the weight that should be add for the exchange rate from u to v . If we add all the weights in a cycle we should get the value of our currency after the intended arbitrage. This would be the same as multiplying the different values in our exchange currency matrix. For example in our graph we would add the weight of edges $E(v_1, v_2) + E(v_2, v_3) + E(v_n, v_1) = 1.0086 \$$, which would be exactly the same as finding the values in our matrix $R(1,2) \times R(2,3) \times R(3,1) = 1.0086 \$$.

Negative cycles and arbitrages

We know an arbitrage is only possible in our particular problem if there is a negative cycle in our graph, an arbitrage happens when you can go through a cycle through the graph and the value at the end has decreased with respect to the starting value of the currency we started with, which can obviously only happen if the cycle in question is negative. If it was not then the value at the end would be exactly the same as the starting value, which would make our intended arbitrage nonexistent. Since the weight of an edge $E(u,v)$ reflects as a weight a value $R(i,j)$ from the matrix, therefore if there does not exist a sequence of values for which $R(a_1, a_2) \times R(a_2, a_3) \times \dots \times R(a_n, a_1) \geq 1$, then there can also not be a cycle of vertices where $E(v_2, v_2) \times E(v_2, v_3) \times \dots \times E(v_n, v_1) \geq 1$.

11.3.1 Side Gig



Graph description

For each road one creates two vertices one for the left side and one for the right side. If it is a one way street, then there is only one vertex. If we have two vertices on a street, there is an edge with weight 4 between them to represent the U-turn. To draw all the other edges one simply connects them as one would be able to drive on a regular road.

Fastest route

The fastest route can be found by running the Bellman-Ford algorithm. There are four possible shortest route as there are two start points and two end points. For example if the starting point is the street to the right, then the driver could start from vertex v_3 or v_6 . The same is true for the end points. Therefore one has to run the Bellman-Ford algorithm four times and then take the shortest output.

11.3.2

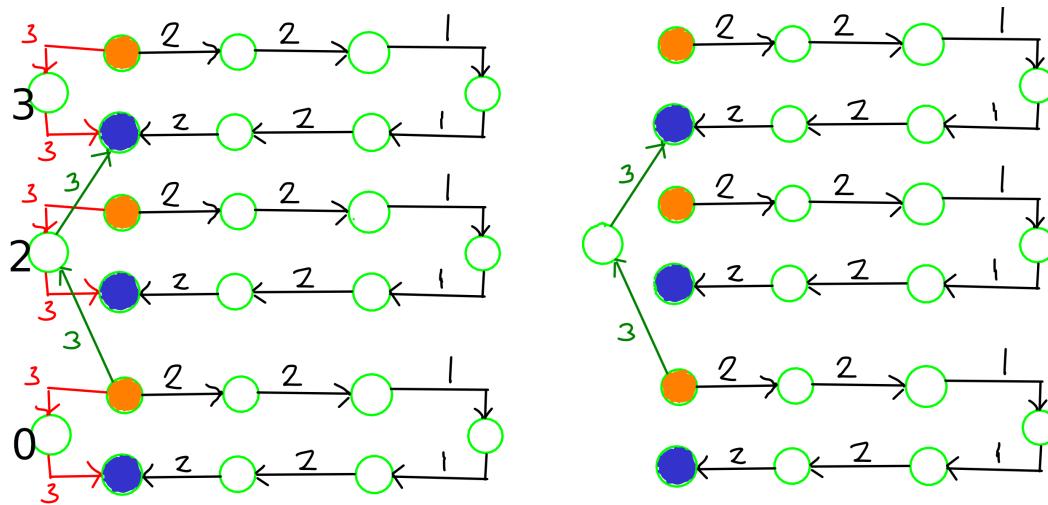


Figure 10: Taxi path with multiple layers. The orange point is the starting point and the blue one the end point. On the left side one sees the path three times with the redirections in green. On the right side is the final graph for the Bellman-Ford Algorithm.

Modelling Graph

We copy the graph L times and number them from 0 to L . Next we redirect all left turns and U-turns of the graph n to the graph $n + 1$. We do that for all graphs from 0 to $L - 1$. If there are still left turns in the L graph one crosses them out.

Fastest route

In this new bigger graph there is still one starting point, namely the starting point of the lowest subgraph. But there are now L possible end points. Therefore one has to run the Bellman-Ford Algorithm L times, for each end point once. The minimum of those L outputs is the shortest path.

Running time analysis

One has to run L times the Bellman-Ford algorithm therefore the running time is $\mathcal{O}(|V|^3 \cdot L)$. Alternatively one could implement the Floyd-Marhsall-Algorithm, which would work similarly.

11.3.2 DP Try (Wrong Approach)

Definition of the DP table

i The current iteration

l The maximum number of left turns.

		l				
		0	1	2	3	
i		0	∞	∞	∞	∞
		1	∞	∞	∞	∞
2		2	∞	∞	6	6
3		3	∞	∞	6	6
4		4	∞	∞	6	6
5		5	8	8	6	6

Table 6: Shortest Path

Calculation of an entry

To calculate each entry on the DP-table we do a simple recursion in which we select the minimum value between the values of the last iteration, selecting if it would be better to add a U-turn or left turn or remain right, we also have to take notice of the value corresponding to the way we are going and add it to our entry.

$$T(i, l) = \min\{T(i - 1, l) + 1, T(i - 1, l) + 2, T(i - 1, l - 1) + 3, T(i - 1, l - 1) + 4\} \quad (73)$$

Calculation order

We start by calculating all entries on the first row, then we start going from left to right in each of the subsequent rows, since each row is dependent on the result of the row that stands over it. Since you have to compare the value of the two fields that stand over the one we want to calculate.

Reading the solution

The solution is on the bottom right of the DP-table, which corresponds to the weight of the fastest route that you can take with the amount of left and U-turns that are allowed to you.

The following was presented in the exercise on 17. December 2018.

12 Exercise (1/1, 2/2)

12.1 Ancient Kingdom of Macedon

Paving the roman roads

We can model this problem as a graph problem, each city v in the kingdom of Macedon would be a vertex in our graph and each road connecting two cities u and v would be an edge $E(u,v)$, since the price to pave a road is directly proportional to the length of the road the weight of each of the vertices in the graph will correspond to the length of the road. We would then proceed to find the minimal span tree in our graph, since it would give us a sub graph where all of our cities are connected and we have the minimum possible overall weight, this would mean that all cities can be visited from any other city only using asphalt roads, while maintaining the overall cost as low as possible. To find the minimum spanning tree one would have to use Kruskal's algorithm, which works as follows:

First we sort all edges our graph in a list from smallest to largest. We create an empty graph, then we iterate through all edges in a our list, if they would not form a cycle with the edges already in the graph, then we add them to our graph, once we have iterated through the whole list we would have our minimum spanning tree as our new graph.

Should new roman roads be paved for the checkpoint?

Proof by contradiction: Let G be any graph with different weight on each of it's edges, let A be the minimal spanning tree subgraph of that graph. Getting the price you would have to pay for the checkpoints and the maintenance of the road would mean adding a number k , (which is constant and is not dependable of the weight of each of the edges) to each of the weights of any of the edges of the graph. To get the optimal price for our roads with checkpoints that connect every city we would have to find a new minimal spanning tree. If we assume that we have to pave a new road that was not on our previous minimal spanning tree, that would mean that our new tree would have to leave out one of the old roads. The road we are leaving out was part of the optimal solution for the spanning tree without checkpoints, which means it had a smaller weight than the road we are getting into the spanning tree now, since the constant number k has been added to each of the weights that would mean that the road left outside is still from a smaller weight than the new road, which would mean the new road is not the optimal solution, coming to a contradiction. This means no new roman roads should be converted into asphalt roads and the minimal spanning tree should stay exactly like it was when no checkpoints were on the roads.

12.2 The Swiss Federal Roads Office

Model the graph

We have N cities and therefore we create N vertices. We connect the vertices with the M roads as edges. The edge weight is the price of the road, except for those roads which are already paid for, we set the weight to zero.

Algorithm

The Kruskal's algorithm can find the minimum spanning tree. It first orders all the edges in ascending order and then finds and adds edge which does not create a cycle. The following algorithm is slightly modified. In the first for loop, it finds all the roads already paid for and orders the unpaid roads simultaneously, as it is more efficient. Then in the second for loop is the Kruskal's algorithm for the remaining unpaid roads.

Algorithm 7 Advanced Kruskal's algorithm

```

procedure ADVKRUSKALS( $V, E$ )  $\triangleright \theta(E \cdot \log V)$ 
    make( $V$ )
    roadsToPayOrdered  $\leftarrow$  emptylist
    totalPrice  $\leftarrow 0$ 
    for  $e$  in roads with  $e = \{u, v\}$  do  $\triangleright \theta(E)$ 
        if  $e$  is paid for then
            if  $find(u) \neq find(v)$  then  $\triangleright \theta(\log V)$ 
                union( $u, v$ )
            else
                roadsToPayOrdered.insertOrdered( $e$ )  $\triangleright \theta(\log E) = \theta(\log V)$ 
        for  $e$  in roadsToPayOrdered with  $e = \{u, v\}$  do  $\triangleright \theta(E)$ 
            if  $find(u) \neq find(v)$  then  $\triangleright \theta(\log V)$ 
                union( $u, v$ )
            totalPrice  $\leftarrow$  totalPrice + Weight of  $e$ 
    return totalPrice

```

Running time

The running time of both for loops in the above algorithm is $\theta(E \cdot \log V)$. Therefore the total running time is also $\theta(E \cdot \log V)$.

Correction: Homework 12.5

T be a MST wrt. w' ?

Is T a MST wrt. w ?

Suppose T, T' ST that wrt w so that

$$w(T) < w(T') \text{ then } w'(T) < w'(T') \quad (74)$$

$$w'(T') - w'(T) = nm(w(T') - w(T)) + \sum_{e_j \in T'} i + \sum_{e_j \in T} j \quad (75)$$

$$\geq nm + \underbrace{\sum_{e_j \in T'} i}_{>0} - \underbrace{\sum_{e_j \in T} j}_{<(n-1)\cdot m} \geq n \cdot m - (n-1) \cdot m \geq 0 \quad (76)$$

Correction: 12.5

Claim: For two different MST S, T , there are always $s \in E(S), t \in E(T)$ so that $S - s + t$ is a MST.

Suppose (\star) is not true, let S, T be a counterexample.

$$\exists s \in E(S), t \in E(T) \text{ so that } \underbrace{S - s + t}_{S'} \text{ is a MST} \Rightarrow w(s) = w(t) \quad (77)$$

$$S' \neq T$$

$$\exists s' \in E(S'), t \in E(T) \text{ so that } \underbrace{S' - s' + t}_{S'} \text{ is a MST} \quad (78)$$

Proof of Claim 12.5

Consider $S + T$

$$\exists s \in E(C) \text{ so that } w(s) = w(t) \quad (79)$$

$$S' = S + t - s \text{ for some } s \in t(c) \quad (80)$$

$$T - t$$

The cost for s must be the same as for tree as both are a minimal spanning tree and therefore the cost must be the same otherwise the both would'nt create a minimal spanning tree.

Correction: 12.5.4

It is not true.

Find a counterexample (Three vertices with 2 edges 1)

12.5.5

A SPT is a ST B with root r so that $\forall v \in V(B)$

$r - v$ is a shortest path in G .

1. Every MST T is A SPT
is false by counterexample
2. Every SPT is a MST
is false by counterexample

13 Programming Exercises

13.1 Wind Turbines

```

1 static int solve(int n, int D, int[] d, int[] e)
2 {
3     int[] T = new int[n+1];
4     int q=1;
5
6     for (int i=1; i<=n; i++) {
7         while (d[i] - d[q] >= D)
8             q++;
9         T[i]=Math.max(T[i-1], T[q-1]+e[i]);
10    }
11    return T[n];
12 }
```

13.2 Algo Tower

```

1 static int solve(int n, int[] l, int[] b, int[] h) {
2     int[] dp = new int[n + 1];
3     int max = 0;
4
5     for (int i = 1; i <= n; i++) {
6         dp[i] = h[i];
7         for (int prev = 1; prev < i; prev++) {
8             if ((l[prev] >= l[i] && b[prev] >= b[i]) || (l[prev] >= b[i] &&
9                 b[prev] >= l[i]))
10                dp[i] = Math.max(dp[prev] + h[i], dp[i]);
11        }
12        max = Math.max(dp[i],max);
13    }
14    return max;
15 }
```

13.3 Submatrix

```

1 import java.io.InputStream;
2 import java.io.PrintStream;
3 import java.util.Arrays;
4 import java.util.Scanner;
5
6 class Main
7 {
8     static int[][] matrix;
9
10    static int query(int a, int b, int c, int d) {
11        return matrix[b][d]-matrix[b][c-1]-matrix[a-1][d]+matrix[a-1][c-1];
12    }
13
14    public static void read_and_solve(InputStream in, PrintStream out) {
15        Scanner scanner = new Scanner(in);
16        int a; int b; int c; int d;
17
18        int n = scanner.nextInt();
19        matrix = new int[n+1][n+1];
```

```

20     for (int i=1; i<=n; i++) {
21         for (int j=1; j<=n; j++) {
22             matrix[i][j] =
23                 matrix[i][j-1]+matrix[i-1][j]-matrix[i-1][j-1]+scanner.nextInt();
24         }
25     }
26     int q = scanner.nextInt();
27     int[][] cases = new int[q][4];
28     for (int i=0; i<q; i++) {
29         for (int j=0; j<4; j++) {
30             cases[i][j] = scanner.nextInt();
31         }
32     }
33     for (int i=0; i<q; i++) {
34         a = cases[i][0];
35         b = cases[i][1];
36         c = cases[i][2];
37         d = cases[i][3];
38         out.println(query(a,b,c,d));
39     }
40 }
41 scanner.close();
42 }
43 }
44 public static void main(String[] args) {
45     read_and_solve(System.in, System.out);
46 }
47 }
48 }
```

13.4 Art Gallery

```

1 static int solve(int n, int V, int T, int[] volume, int[] time, int[] price) {
2     int[][][] dp = new int[n + 1][V + 1][T + 1];
3
4     for (int i = 1; i <= n; i++) {
5         for (int t = 1; t <= T; t++) {
6             for (int v = 1; v <= V; v++) {
7                 if (t >= time[i] && v > volume[i]) {
8                     dp[i][t][v] = Math.max(dp[i - 1][t][v], dp[i - 1][t -
9                         time[i]][v - volume[i]] + price[i]);
10                } else {
11                    dp[i][t][v] = dp[i - 1][t][v];
12                }
13            }
14        }
15    }
16    return dp[n][V][T];
}
```

13.5 Dyno

```
1 private static int solve(int L, int D, int C, int[] c) {
2     boolean[] cac = new boolean[L + 1];
3     for (int i : c) {
4         cac[i] = true;
5     }
6     boolean[] path = new boolean[L + 1];
7     path[0] = true;
8
9     int max = 0;
10    for (int i = 1; i <= L; i++) {
11        if (!cac[i] && (path[i - 1] || (i >= D && path[i - D]))) {
12            path[i] = true;
13            max = i;
14        }
15    }
16    return max;
17 }
```

Introduction to Programming - Lecture

Prof. Thomas R. Gross

Contents

1 Einführung in die Programmierung	6
1.1 Prüfung	6
1.2 Allgemeines	6
1.3 Warum Programmieren lernen?	6
2 EBNF	7
2.1 Control forms (zum Kombinieren)	7
2.1.1 Aufreihung (Sequence)	7
2.1.2 Auswahl	8
2.1.3 Option	8
2.1.4 Wiederholung (Repetition)	8
2.2 EBNF Beispiel	8
2.3 Überprüfung der LHS	8
2.4 Ableitungsbäume	9
2.5 Sonderzeichen	9
2.6 Äquivalente EBNF Beschreibung	9
2.7 Syntax und Semantik	9
2.8 EBNF Beschreibung <i>integer-set</i>	9
2.9 Graphische Darstellung von EBNF Regeln	10
2.10 Rekursion	10
3 Einfache Java Programme	11
3.1 EBNF	11
3.1.1 Bezeichner	11
3.2 Einführung	11

3.3	Methoden	11
3.3.1	static methods	11
3.3.2	Wie entstehen Methoden?	12
3.3.3	Ausführung einer Methode	12
3.4	Typen und Variablen	12
3.4.1	Welche Typen kann ein Java Programm verwenden?	12
3.4.2	Ausdrücke (Expressions)	12
3.4.3	Reelle Zahlen (Typ double)	13
3.5	String und Operationen	13
3.6	for loop syntax	13
3.7	Input und System.in	13
3.8	Verschachtelte for-Schleife	13
3.9	Übergabe von Werten	14
3.10	if-else Anweisung	14
3.11	Boolesche Ausdrücke	14
3.12	Typ boolean	14
3.13	Bedingte short-circuit Auswertung	14
3.14	Gartenzaun Analogie	14
3.15	while-Schleifen	14
3.16	do/while-Schleife	15
3.17	Ergebnis Rückgabe	15
3.18	Scope (Sichtbarkeitsbereich)	15
3.19	String, Math und Random	15
3.20	Math Klasse	16
3.21	Random Klasse	16
3.22	Arrays	16
3.23	Graphische Benutzeroberflächen (GUIs)	16

3.24 Input / Output Dateien	16
3.25 Javadoc	17
4 Klassen und Objekte	17
4.1 Reference Semantics	17
4.2 Value Semantics	17
4.3 Klassen selber entwickeln	18
4.4 Beispiel Point Objekte	18
4.5 Attribute	18
4.6 Array in Objekten	18
4.7 Methoden	19
4.8 Konstruktoren	19
4.9 Encapsulation	20
4.9.1 Private Attribute	20
4.9.2 this Keyword	20
4.10 Redundanz in Programmen	21
4.11 static	21
4.12 Type Case	21
5 Arbeiten mit Objekten und Klassen	21
5.1 Kombinationen von Referenzen auf Klassen und Arrays	21
5.2 Rekursive Methoden	22
5.3 Datenstrukturen mit Verknüpfungen	22
5.3.1 List Node	22
5.4 get-Methode	23
5.5 add-Methode	23
5.6 Program beenden	23
5.7 remove-Methode	23
5.8 addSorted-Methode	23

5.9 Vergleiche Array und Liste	24
5.10 Unterschied null und leere Liste	24
5.11 Java Programme Visualisieren	24
5.12 Neue Klasse aus existierenden Klassen	24
5.13 Details offen halten	25
5.14 Protected Attribute	25
5.15 Array mit verschiedenen Referenzvariablen	25
5.16 Klasse Object	25
5.17 Casts	26
5.18 @Override	26
5.19 Polymorphismus	26
6 Interfaces	26
6.1 Interface Repetition	27
6.2 Overloading	28
7 Exceptions	28
8 Generische Programmierung	29
8.1 Collection	29
8.2 ArrayList	29
8.3 Typeparameter	29
8.4 Wrapper Klasse	30
8.5 compareTo	30
8.6 Collections	30
8.7 Interface Comparable	30
8.8 Ternary Operator	31
8.9 Collections	31
8.10 Mengen (Sets)	31

8.10.1 HashSet	31
8.11 TreeSet	31
8.12 Linked Hash Set	31
8.13 Operationen in Sets	31
8.14 Ordnungsrelationen	32
8.15 For each Schleife	32
8.16 Map	32
8.17 KeySet	32
8.18 Values	32
8.19 Umkehrfunktion der Abbildung	33
8.20 Iteration	33
9 Abstrakte Datentypen	34
9.1 Stack	34
10 Systematisches Programmieren	34
10.1 Hoare Logik	34
10.1.1 Vorwärts schliessen	34
10.1.2 Rückwärts schliessen	35
10.2 Terminologie	35
10.2.1 If-Statements	35
10.3 Notation	35
10.4 Hoare Triple	35
10.5 Assert	36
10.6 Hoare Triple - Zuweisung	36
10.7 Hoare Triple - Folgen von Anweisungen	36
10.8 Hoare Triple - If-Statements	36
10.9 Aussagen über Zustände	37
10.10Vorbedingung	37

10.11Loop	37
10.12Hoare Logik - Invariante	38
10.13Methodologie um Invarianten zu finden	38
10.14Chiara Problem	38
10.15Abstract	38
10.16Inner Classes	38

The following was presented in the lecture on 18. September 2018.

1 Einführung in die Programmierung

1.1 Prüfung

24. Januar 2019
 ONA7 und ONA25 ab 9:00

1.2 Allgemeines

Laboratory for Software Technology

<https://lst.inf.ethz.ch>

<https://www.video.ethz.ch/lectures/d-infk/2018/autumn/252-0027-00L.html>

Username: gro-18w

Password: Ca3A7Zh

1.3 Warum Programmieren lernen?

- Problem analysieren
- Problem in Teilprobleme zerlegen
- Lösungen finden
- Ergebnisse kombinieren

Zwiespalt:

1. Programmiersprache einfach zu schreiben
2. Programmiersprache einfach zu lesen

EBNF:

- Extended
- Backus
- Naun or Normal
- Form

[https://en.wikipedia.org/wiki/Extended_Backus\Naur_Form](https://en.wikipedia.org/wiki/Extended_Backus%5C-Naur_Form)

Control Forms:

- Sequence
- Decision
- Repetition
- Recursion

EBNF rule

Left Hand Side, Symbol and Right Hand Side

The following was presented in the lecture on 21. September 2018.

2 EBNF

EBNF Beschreibung ist eine Menge von EBNF Regeln

LHS \Leftarrow RHS

\Leftarrow : bedeutet ist definiert als **RHS kann eines der folgenden enthalten:**

- Namen
- Buchstaben (Kann auch eine Zahl sein)
- Kombinationen der vier Kontrollelemente ("control forms")

digit kursiv gedruckt ist das Gleiche wie <digit>

Wenn etwas krusiv gedruckt ist, ist es definiert als etwas.

2.1 Control forms (zum Kombinieren)

2.1.1 Aufreihung (Sequence)

- Aufreihung von links nach rechts

- Reihenfolge ist wichtig
- Muss mit Abstand getrennt werden
- Jeder Name darf nur einmal definiert werden

Beispiel:

```
buchstabe_1 = D
buchstabe_2 = 2
buchstabe_3 = 8
raum = buchstabe_1  buchstabe_2  buchstabe_3
```

2.1.2 Auswahl

- Eine Menge von Alternativen (Reihenfolge unwichtig)
- Durch | (eng.: “Stroke”) getrennt
- Alternativen folgen den EBNF Regeln

2.1.3 Option

- Element in [und] (eng.: “square bracket”)
- Kann gewählt werden, muss aber nicht gewählt werden

2.1.4 Wiederholung (Repetition)

- Der zu wiederholende Ausdruck steht zwischen { und } (eng.: “curly braces”)
- Kann soviel wiederholt werden, wie ich will
- 0 Wiederholungen bedeutet es fehlt und ist auch möglich

2.2 EBNF Beispiel

Definition ganzer Zahlen:

```
digit <= 0|1|2|3|4|5|6|7|8|9
integer <= [+|-]  digit  { digit}
```

2.3 Überprüfung der LHS

Genaue Übereinstimmung: legal \Rightarrow EBNF Regel erfüllt

- Buchstaben im Symbol
- Es darf kein Buchstabe im Symbol übrig bleiben

- Es darf kein Element übrig bleiben

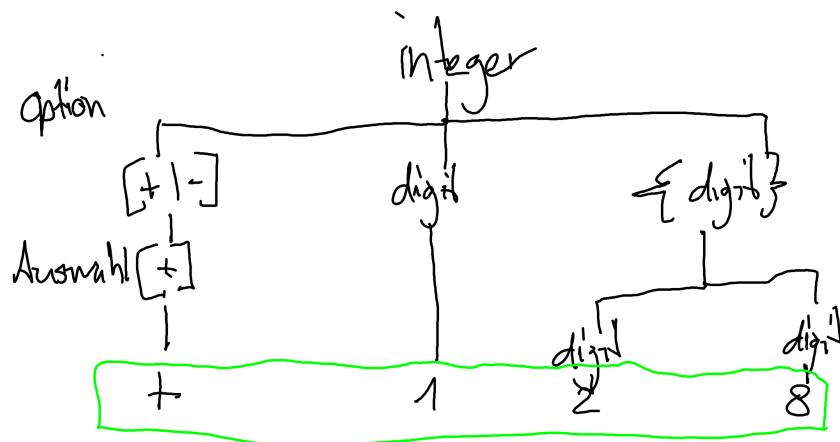
Sonst: Symbol nicht legal \Rightarrow illegal

2.4 Ableitungsbäume

Graphische Darstellung eines Beweises durch eine Tabelle

Oben: Name der EBNF Regel

Unten: Symbol



2.5 Sonderzeichen

Wenn ein Sonderzeichen wie {}, [], (), () geschrieben werden muss, wird ein Rahmen herumgezeichnet.

2.6 Äquivalente EBNF Beschreibung

Äquivalent = Gleichwertig

EBNF Beschreibungen erkennen die selben legalen und illegalen Symbole

2.7 Syntax und Semantik

EBNF beschreibt nur die Syntax

2.8 EBNF Beschreibung *integer_set*

- Mengen von Zahlen
- Anfang einer Beschreibung, Ende
- Zwischen {} und } keine oder mehr Zahlen, durch Komma getrennt

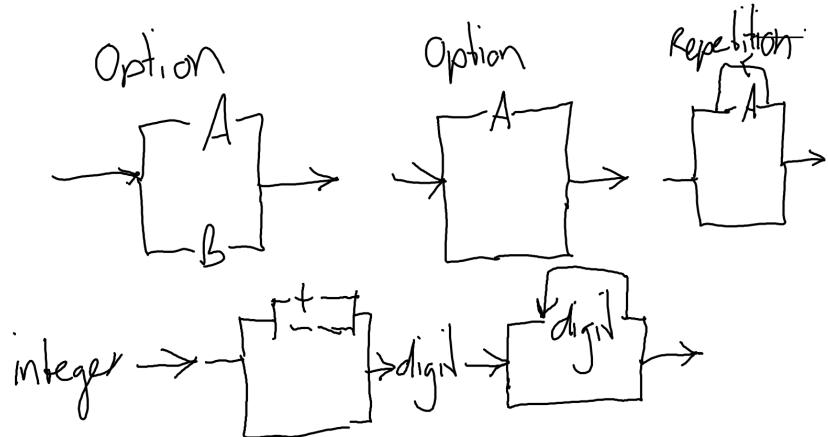
$$\text{integer_list} \Leftarrow \text{integer}\{, \text{integer}\} \quad (1)$$

$$\text{integer_set} \Leftarrow \boxed{\{\text{[integer_list]}\}} \quad (2)$$

Box bedeutet wir meinen genau dieses Zeichen.

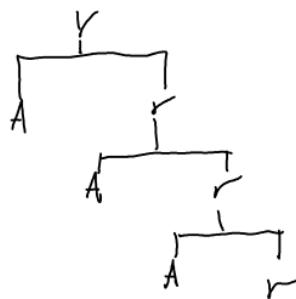
The following was presented in the lecture on 25. September 2018.

2.9 Graphische Darstellung von EBNF Regeln



2.10 Rekursion

Wenn der Name der Links auftrifft auch Rechts auftritt hat man eine rekursive Regel. $r \Leftarrow |Ar$



Das Problem ist, dass nicht jede Rekursion durch Wiederholungen ausgedrückt werden kann. Deshalb wird Rekursion eingeführt. Beispiel: Wenn die Anzahl A gleich von B sein sollte, z.B. AAABBB

$$\text{balance} \Leftarrow |A \text{ balance } B$$

1. Direkte Rekursion:

$$r \Leftarrow A|Ar$$

2. Indirekte Rekursion:

Folge von Regeln N_1, \dots, N_k sodass N_2 auf der RHS von N_1, N_3 auf der RHS von R_2, \dots und N_1 auf der RHS von N_k erscheint.

$name1 \Leftarrow A \quad name2$
 $name2 \Leftarrow B \quad name1|C$

3 Einfache Java Programme

3.1 EBNF

Hält die Syntax Regeln von Java Programmen fest

3.1.1 Bezeichner

$bezeichner \Leftarrow letter \quad \{letter|digit\}$

3.2 Einführung

class Name sollte gleich dem .java Namen sein.

Eine Java Methode muss main heissen; das ist der Code den wir ausführen möchten.

Konvention Programmnamen und class beginnt mit grossem Buchstaben.

Konvention method beginnt mit einem Kleinbuchstaben

Java Comments: // Text und /* Text */

The following was presented in the lecture on 28. September 2018.

3.3 Methoden

public static void helper() { } ist ein Beispiel für eine weitere Methode.

Ein guter Programmierer zerlegt ein Programm in verschiedene Methoden und hat so das Problem eigentlich schon gelöst.

3.3.1 static methods

static methods: Methode mit weiteren Eigenschaften

main ist eine static method

main wird automatisch aufgerufen

3.3.2 Wie entstehen Methoden?

Entwickeln des Algorithmus und Aufteilung in Teil-Probleme

```
1 public class PrintExample {  
2     public static void main(String[] args) {  
3         printWarning();  
4         System.out.println("Langer Text");  
5         printWarning();  
6     }  
7  
8     public static void printWarning() {  
9         System.out.println("\n-----\n");  
10        System.out.println("!!!Warning!!!");  
11        System.out.println("\n-----\n");  
12    }  
13 }
```

3.3.3 Ausführung einer Methode

Die Abfolge der Ausführung nennen wir Kontrollfluss (eng.: Control flow)

3.4 Typen und Variablen

Typen (eng.: "types") beschreiben Eigenschaften von Daten
Die Programmiersprache legt fest, wie ein Typ implementiert ist.

3.4.1 Welche Typen kann ein Java Programm verwenden?

Name	Beschreibung	Table 1: Beispiele
int	ganze Zahlen	42, -3, 0, 926394
double	reelle Zahlen	3.1, -0.25, 9.4e3
char	(einzelne) Buchstaben	'a', 'X', '?', '\n'
boolean	logische Werte	true, false

3.4.2 Ausdrücke (Expressions)

Expression: Ein Wert oder Operanden und Operator, die einen Wert berechnen

wert \Leftarrow 0|1|2|3|4|5|6|7|8|9
operator \Leftarrow +|-|*|/|%
expr \Leftarrow *wert* | *expr operator expr*

Während der Ausführung eines Programms werden die Ausdrücke ausgewertet (eng.: evaluated)

3.4.3 Reelle Zahlen (Typ double)

The following was presented in the lecture on 02. October 2018.

3.5 String und Operationen

In Java gibt es automatische Konvertierung wo man zum Beispiel String und Integeres zusammenfügen kann.

3.6 for loop syntax

```
1 for (initialization; test; update) {  
2     statement;  
3 }
```

- Initialisierung wird einmal am Anfang ausgeführt und bestimmt, welche Variable für den Foor-Loop verwendet wird.
- Test
- Update

x=1

y=x++: x wird zuerst verwendet und erst dann ausgeführt. Deshalb ist y auch gleich 1.

3.7 Input und System.in

Scanner: erlaubt es Input von unterschiedlichen Quellen zu lesen. Scanner sind in der Bibliothek java.util definiert.

```
1 import java.util.*;  
2 Scanner console = new Scanner(System.in); //Statt console kann man auch einen anderen  
3 int alter = console.nextInt();
```

The following was presented in the lecture on 05. October 2018.

3.8 Verschachtelte for-Schleife

Auf Englisch nennt man es Nested Loop.

3.9 Übergabe von Werten

Auf Englisch nennt man es "Value semantics".

Die Werte werden an die Funktion übergeben und diese behandelt die Werte als neue Werte ausser sie wurden als Static definiert.

3.10 if-else Anweisung

Um if-else Schleifen zu vereinfachen können boolesche Ausdrücke verwendet werden.

3.11 Boolesche Ausdrücke

&& is And

|| is Or

! is not

Boolesche Operationen haben sehr kleine Präcedenz.

3.12 Typ boolean

Boolesche Werte können in Boolean abgespeicheret werden. Diese können den Wert true or false annehmen.

3.13 Bedingte short-circuit Auswertung

Java beendet die Auswertung eines booleschen Ausdrucks sobald das Ergebnis feststeht. Umbedingt aufpassen. Es kann ungewollte Nebenwirkungen haben, wenn zum Beispiel Variablen noch verändert werden in einer if-Schleife, jedoch der zweite Teil nicht immer ausgeführt wird.

De Morgan's Regeln, das sind Regeln für die Negation von booleschen Ausdrücken und können zum Teil solche Nebenwirkungen verhindern.

The following was presented in the lecture on 09. October 2018.

3.14 Gartenzaun Analogie

Füge eine Anweisung ausserhalb der Schleife hinzu.

3.15 while-Schleifen

while-Schleife: Führe den Schleifenrumpf so lange aus wie der boolesche Ausdruck den Wert true ergibt.

3.16 do/while-Schleife

Führe das do Zuerst aus und mache erst dann den while check.

```
1 do {  
2 # Statement  
3 } while (# check)
```

3.17 Ergebnis Rückgabe

```
1 public static type name(paramters) {  
2 statements;  
3 ...  
4 return expression;  
5 }
```

type void bedeutet, dass es kein Rückgabewert gibt.

Bei einer void Methode kann man auch ein return schreiben, einfach ohne dass ein Wert zurückgegeben wird.

3.18 Scope (Sichtbarkeitsbereich)

scope: Der Teil eines Programms in dem eine Variable sichtbar ist. Eine Variable die in einer Methode definiert wurde, existiert nur in der Methode. Definiert in for-Schleife existiert nur in for-Schleife.

3.19 String, Math und Random

Ein String ist KEIN array. String methoden:

- indexOf
- length
- substring
- toLowerCase
- toUpperCase

Diese Methoden ändern den String nicht. Man muss einen neuen String zuweisen.

The following was presented in the lecture on 12. Octobre 2018.

3.20 Math Klasse

Enthält viele Mathematik Methoden.
Die meisten geben double Werte zurück.

3.21 Random Klasse

```
1 Random rand = new Random();
2 int randomNumber = rand.nextInt(10);
```

3.22 Arrays

Ein Array erlaubt uns, mehrere Werte des selben Typs zu speichern.

```
1 type[] name = new type[length];
```

Arrays class hat eine equals Methode mit der man zwei Arrays vergleichen kann.
Sie enthält auch eine toString Methode für die Ausgabe von Arrays.

The following was presented in the lecture on 16. October 2018.

3.23 Graphische Benutzeroberflächen (GUIs)

```
1 int width=500, height = 300;
2 Window window = new Window("First GUI", width, height);
3 window.open();
4 window.fillCircle(30,30,20);
5 for (int i = 0; i < width;i++0 {
6     double x = 0.05 * i;
7     double y = Math.sin(x);
8     window.fillRect(i, y*height/4 + height/2,1,1);
9     window.refresh(10);
10 }
11 window.waitUntilClosed();
```

3.24 Input / Output Dateien

```
1 import java.io.*;
2 import java.util.*;
3 public static void main ([String[] args) throws FileNotFoundException {
4     File file = new File("example.txt"); // Only the handler; does not create a new
        file;
5     if (file.exists() && file.length() > 1) {
6         Scanner scanner = new Scanner(file);
7         int zahl = scanner.nextInt();
```

```
8     PrintStream fileOutput = new PrintStream(file);
9     fileOutput.println(zahl);
10    }
11 }
```

Note 1. Nicht mit dem selben Input und Output file arbeiten.

3.25 Javadoc

Kann man benützen um verschiedene Methoden über Klassen und Methoden holen. Sollte man unbedingt vor dem Test anschauen. Da man es verwenden kann während dem Test.

The following was presented in the lecture on 19. October 2018.

4 Klassen und Objekte

Klassen beschreiben einen Typ

Typen beschreiben Eigenschaften von Daten

Ein Typ beschreibt eine Menge oder Kategorie von Daten Werten. Der Begriff Objekt ist der Sammelbegriff für alle Datenwerte, die durch irgend eine Klasse beschrieben werden.

Begriff new initialisiert ein neues Objekt.

Strings sind auch Objekte deshalb könnte man auch mit dem new Operator ein neuer String konstruieren:

```
1 String s = new String("Hello World");
```

4.1 Reference Semantics

Eine Variable enthält eine Referenz auf einen Array.

Wenn man aber nun a den Array b zuweist, und nun a[0] verändere ist auch b[0] verändert.

Objekte verwenden auch Reference Semantics.

Wenn man den Array kopieren möchte muss man folgenden Command verwenden.

```
1 int[] src = {1,2,3,4,5};
2 int[] dest = new int[5];
3 System.arraycopy( src, 0, dest, 0, src.length ); // Professor
4 dest = src.clone(); // Eigene Erfahrung
```

4.2 Value Semantics

Die Basistypen verwenden Value Semantics, wo man jede Variable einzeln verändern kann.

4.3 Klassen selber entwickeln

In einer Klasse können wir:

1. Einen namenlosen Service implementieren
2. Eine neue Art von Objekten beschreiben

Klasse ist die Vorlage die Objekte beschreibt.

Ein Objekt ist ein Gebilde das Zustand und Verhalten verbindet (State and behavior)

Objekt-orientiertes Programmieren ist ein Programmiermodell das ein Programm als eine Menge von aufeinander einwirkenden Objekten organisiert.

Objekte sind Exemplare einer Klasse

- Wenn ein Objekt erschaffen wird spricht man von der Instanziierung
- Die Menge aller Objekte einer Klasse bilden einen Typ

4.4 Beispiel Point Objekte

Point Objekte können von anderen Programmen verwendet werden.

Programme die Point Objekte verwenden heissen Klienten (Client Programs) der Point Klasse.

The following was presented in the lecture on 23 .October 2018.

4.5 Attribute

Ein Objekt ist ein Gebilde das Zustand und Verhalten erbindet. Klasse ist die Vorlage die Objekte beschreibt.

Attribute (field): Eine Variable innerhalb eines Objektes die Teil des Objekt Zustandes ist
Ein Exemplar ist ein neu generiertes Exemplar dieser Klasse.

Die Klasse die ein neues Exemplar erstellt ist ein Klient der Klasse wo das Exemplar programmiert ist.

Mit einer Referenzvariable kann man auf ein Attribut eines Objektes zugreifen.

Für reference variables gelten die reference semantics Regeln.

Wenn man zwei Reference Variablen auf das gleiche Exemplar hat und ein Attribut des Exemplares verändert, ändert sich auch das Attribut für das Exemplar und nicht nur für die reference variable.

Mit "null" kann man eine reference Variablen zurücksetzen.

4.6 Array in Objekten

Initialisierung von Array Objekten:

```
1 Point[] ziel = new Point[3];
2 for (int j; j < ziel.length; j++) {
3     Point[j] = new Point();
4 }
```

Initialisierung eines mehrdimensionalen Arrays

```
1 int [] [] x = new int [5][5];
2 int [] [] x = {{1,0,0},{0,1,0},{0,0,1}};
```

The following was presented in the lecture on 26. October 2018.

4.7 Methoden

Methoden in Exemplaren habe keine static Methode. Normale Methoden, die sich in Objekten befinden.

AccessorMethoden Eine Methode die es erlaubt den Zustand eines Objektes anzusehen (bsp. getAdresse());

MutatorMethoden Eine Methode die den Zustand eines Objektes verändert (bsp. setAdresse());

Um ein Objekt zu drucken, kann man dafür eine `toString` Methode im Objekt schreiben.

```
1 public String toString() {
2     return name + Arrays.toString(myarray)
3 }
```

4.8 Konstruktoren

Konstruktor initialisiert den Zustand eines neuen Objektes.

Deshalb kann man dann, so Objekte erstellen:

```
1 Point p = new Point(3, 8); // Initialisierung von irgendwo
2
3 public class Point {
4     int x;
5     int y;
6
7     // Konstruktor
8     public Point(int initialX, int initialY) {
9         x = initialX;
10        y = initialY;
11    }
12
13    // Default Konstruktor
14    public Point() {
```

```

15     x = 0;
16     y = 0;
17     // this(0, 0) Besser
18 }
19
20 //Methods
21 public void setLocation(int newX, int newY {
22     // Statements
23 }
24 }
```

Eine Klasse kann mehrere Konstruktoren haben, jedoch muss jeder Konstruktor eine unverwechselbare Liste von Parametern haben. Also unterschiedliche typen.

Wenn wir einen Konstruktor haben, verschwindet der Default Konstruktor. Deshalb ist es empfohlen noch einen Default Konstruktor zu kreieren.

Im Konstruktor darf keine Variable deklariert werden.

Konstruktores sind keine Methoden, deshalb darf man auch nichts returnieren. (Auch kein void)

4.9 Encapsulation

Idee: Verstecken der Details der Implementierung einer Klasse vor den Klienten der Klasse.

Trennung von Verhalten (extern sichtbar) und Zustand (intern).

Encapsulation erlaubt Anpassung an Objekten, ohne dass der Klient etwas mitbekriegt und dadurch kaputt geht.

4.9.1 Private Attribute

```
1 private int id;
```

Private Mehoden ond types können nur von der jeweiligen Klasse aufgerufen werden

4.9.2 this Keyword

this.field Expliziter Zugriff auf ein Attribut

this.method() Expliziter Zugriff auf eine Methode

this verweist immer auf den impliziten Parameter einer Methode

this kann auch verwendet werden um einen Konstruktor aufzurufen: this(constructor)

The following was presented in the lecture on 30. October 2018.

4.10 Redundanz in Programmen

Modul: wiederverwendbare Software, in einer Klasse abgelegt.
So wird es aufgerufen:

```
1 class.method(parameters);
```

4.11 static

Eine Static Methode gehöhrt zur Klasse. Static Methoden sind nicht Teil eines Attributes. Und deshalb exisiteren sie nur einmal, unabhängig wieviele Exempalre erstellt wurden.

Im Englischen wird eine static Variable field genannt.

final bedeutet, dass es nicht mehr verändert werden kann.

The following was presented in the lecture on 02. November 2018.

4.12 Type Case

Eine explizite Umwandlung in einen anderen type wird type cast genannt. Er ist rechts-assoziativ, dass heisst er wandelt nur den Operanden direkt rechts daneben um.

```
1 (type) expression
2 double result = (double) 19 / 5;
3 int result2 = (int) result;
4 double x = (double) 1 + 1 / 2; // 1.0
5 double y = 1 + (double) 1 / 2; // 1.5
6 double y = 1 + 1.0 / 2; // 1.5 The same as above
```

5 Arbeiten mit Objekten und Klassen

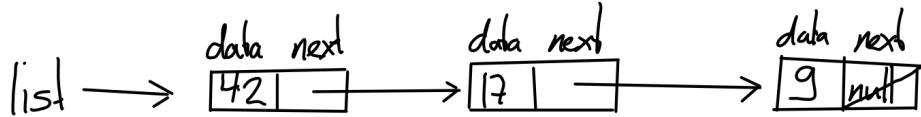
5.1 Kombinationen von Referenzen auf Klassen und Arrays

```
1 public static void someFct() {
2     SomeClass [] ca = new SomeClass[3]
3
4     for (int i=0, i<3; i++ {
5         ca[i] = new SomeClass();
6         ca[i].one = new OtherClass();
7         ca[i].two = new int[5];
8         ca[i].three = new SomeClass();
9     }
10
11     System.out.println(ca[2].three.two[3]);
12 }
```

5.2 Rekursive Methoden

1. Basis Fall finden. Wann hört die Rekursion auf? Muss immer gegeben sein.
2. Wie stellen wir sicher, dass wir auf die Basis hinarbeiten.

5.3 Datenstrukturen mit Verknüpfungen



5.3.1 List Node

```
1 class ListNode {
2     int data;
3     ListNode next;
4
5     public ListNode(int data) {
6         this.data = data;
7         this.next = null;
8     }
9
10    public ListNode(int data, ListNode next) {
11        this.data = data;
12        this.next = next;
13    }
14 }
15
16 public class ConstructList {
17     public static void main(String[] args) {
18         ListNode list = new ListNode(0);
19         for (int i=1; i<=10; i++) {
20             list = new ListNode(i,list);
21         }
22
23         ListNode list2 = new ListNode(0);
24         ListNode currentList = list2;
25         ListNode nextList;
26         for (int i=1; i<=10; i++) {
27             nextList = new ListNode(i);
28             currentList.next=nextList;
29             currentList = nextList;
30         }
31         System.out.println(list.data);
32         System.out.println(list2.next.next.next.data);
33     }
34 }
```

The following was presented in the exercise on 06. November 2018.

5.4 get-Methode

```
1 public int get (int index) {  
2     ListNode current = front;  
3     for (int i=0; i<index; i++) {  
4         current = current.next;  
5     }  
6     return current.data;  
7 }
```

5.5 add-Methode

```
1 public void add(int index, int value) {  
2     if (index==0) {  
3         front = new ListNode(value, front);  
4     } else {  
5         ListNode current = front;  
6         for (int i=0; i<index-1; i++) {  
7             current = current.next;  
8         }  
9         current.next = new ListNode(value, current.next);  
10    }  
11 }
```

5.6 Program beenden

```
1 System.exit(-1);
```

5.7 remove-Methode

```
1 public void remove (int index) {  
2     if (index==0) {  
3         front = front.next  
4     } else {  
5         ListNode current = front;  
6         for (int i=0; i<index-1; i++) {  
7             current = current.next;  
8         }  
9         current.next = current.next.next;  
10    }  
11 }
```

5.8 addSorted-Methode

```
1 public void addSorted() {  
2     if (front==null || value <= front.data) {
```

```

3         front = new ListNode(value, front);
4     } else {
5         ListNode current = front;
6         while (current.next != null && current.next.data < value) {
7             current = current.next;
8         }
9         current.next = new ListNode(value, current.next);
10    }
11 }

```

5.9 Vergleiche Array und Liste

Array: Konstante Zugriffszeit

Liste: Grösse ist flexibel

5.10 Unterschied null und leere Liste

null Liste `LinkedList<Integer> list = null;`

leere Liste: `LinkedList<Integer> list = new LinkedList<Integer>();`

The following was presented in the lecture on 09. November 2018.

5.11 Java Programme Visualisieren

https://cscircles.cemc.uwaterloo.ca/java_visualize/

The following was presented in the lecture on 13. November 2018. Double Int List

1. Den Wert einer Zahl
2. Verweis auf Vorgänger
3. Verweis auf Nachfolger

5.12 Neue Klasse aus existierenden Klassen

Vererbung (eng.: inheritance) erlaubt uns, eine Klasse als Erweiterung einer anderen Klasse auszudrücken.

Vererbungshierarchie (Inheritance Hierarchy): Eine Menge von Klassen, die durch eine Beziehung verbunden sind, die gemeinsamen Code verwenden.

Vererbung erlaubt es neue Klassen aus existierenden Klassen zu bilden, sodass die neue Klasse die Attribute beziehungsweise das Verhalten der alten Klasse übernimmt.

Eine Klasse kann eine andere erweitern (extend) und Daten und Zustand sowie Verhalten absorbieren.

Die subclass erweitert die superclass.

```
1 public class subclass extends superclass {  
2     // First get the Constructor of the superClass  
3     public subConstructor(int years) {  
4         super(years); // Called den Construcotr the SuperClass.  
5     }  
6  
7     public double getSalary() {  
8         return super.getSalary() + 5000;  
9     }  
10  
11    // Add extended code  
12    // It is possible to override methods  
13 }
```

Aufpassen bei Konstruktoren. Subclass erben keine Konstruktoren automatisch. Man muss sie explizit aufrufen.

The following was presented in the lecture on 16. November 2018.

5.13 Details offen halten

In der Superclass muss eine Methode nicht endgültig festgelegt werden - eine Subclass kann dann später diese Methode überschreiben.

Selbe Signatur jedoch kann das Verhalten mit override überschrieben werden.

5.14 Protected Attribute

Auf dieses Attribut kann nur innerhalb der Klasse und ihrer Subclasses zugegriffen werden.

5.15 Array mit verschiedenen Referenzvariablen

```
1 Angestellte [] array = new SuperClass [5];  
2 array[0] = new FristSubclass();  
3 array[1] = new SecondSubclass();  
4 array[2] = new ThirdSubclass();
```

5.16 Klasse Object

```
1 Object [] array = new Object [5];  
2 array[0] = new Point(5, 3);  
3 array[1] = "Hello World";  
4 array[2] = new Person();
```

```
5 array[3] = new Point(5, 3);
6 int len = array[1].length(); // Throws an error
7 if (array[0] == array[3]) {} // Is always false
```

5.17 Casts

```
1 public boolean equals(Object o) {
2     if (o instanceof Point) {
3         Point point = (Point) o; // Type Cast
4         return x == point.x && y == point.y;
5     } else {
6         return false;
7     }
8 }
```

5.18 @Override

Ist sinnvoll um Fehler zu vermeiden und nicht einfach neue Methoden zu erstellen. Falls man etwas falsch implementiert hat.

The following was presented in the lecture on 20. November 2018.

5.19 Polymorphismus

Ein Programm ,welches viele Formen und Gestalten annehmen kann. Es sollte so entwickelt werden, sodass es für unterschiedliche Objekttypen verwendet werden kann und sein Verhalten seinem Typen anpasst.

Gerebte Klasse werden Kursiv geschrieben oder unterstrichen. Man kann Vereberungen auch mit einer Tabelle darstellen, mit allen Methoden und Klassen.

Wenn eine Methode nur in einer UnterkLASSE definiert ist, kann man mittels cast darauf zugreifen.

```
1 ((Subclass) var).subclassmethod();
```

Casts ändert nicht die Darstellung oder das Verhalten eines Objektes. sondern nur die Menge der Methoden, die aufgerufen werden können.

Verwandlung geht nur nach unten und oben in der Inheritance Hierarchie.

6 Interfaces

Interfeaces geben uns eine ist-ein Beziehung ohne gemeinsamen Code und Zustand aber mit gemeinsamen Verhalten (Methoden) (Seit Java 8.0 gemeinsamen Code erlaubt)

```
1 public interface Shape {  
2     public double area();  
3     public double perimeter();  
4 }
```

Keine Aussagen über Attribute möglich.

The following was presented in the lecture on 23. November 2018.

6.1 Interface Repetition

Eine Gruppe von Methoden die eine Klasse implementieren kann. Keine Aussagen über Attribute möglich.

```
1 public interface Vehicle {  
2     public void start();  
3     public void move(double speed);  
4     public void stop();  
5 }
```

Die Interface Deklaration enthält abstrakte Methoden: Methoden Header ohne Implementation

```
1 public class Circle implements Shape {  
2     private double radius;  
3  
4     public double area() {  
5         return Math.PI * radius * radius  
6     }  
7  
8     public double perimeter() {  
9         return 2*Math.PI * radius  
10    }  
11 }  
12 }
```

```
1 public class Bisycle implements Vehicle {  
2     public void start() {  
3         ...  
4     }  
5  
6     public void move(double s) {  
7         ...  
8     }  
9  
10    public void stop() {  
11        ...  
12    }  
13 }
```

Ein Interface kann ein anderes Interface erweitern.

```
1 public interface Amphicar implements Vehicle, Boat, Property {  
2     ...  
3 }
```

Interfaces geben uns eine ist-ein Beziehung ohne gemeinsamen Code und Zustand.

Interfaces nützen nicht die Klasse sondern den Klienten.

Jedes Objekt, das das Interface implementiert, kann als Parameter übergeben werden.

Interfaces sind wichtig um grosse Programme zu strukturieren.

6.2 Overloading

Zwei oder mehrere Methoden mit dem selben Namen aber unterschiedlichen Parameter. Zum Beispiel unterschiedliche Typen oder Reihenfolge.

```
1 void foo(int i, double j) {}  
2 void foo(double i, int j) {}  
3 foo(1,2) // Throws an error! Because it does not know which function it should call.
```

7 Exceptions

Änderung der Ausführungsreihenfolge auf Grund eines aussergewöhnlichen Ereignisses.

Man muss dabei zwischen zwei unterschiedlichen Exceptions unterscheiden. Wenn das Programm damit umgehen kann und es weiter geht (FileNotFoundException, InputNotValid) oder das Programm muss gestoppt werden, da eine Weiterführung unmöglich ist (bsp. OutOfMemoryError, NullPointerException, NoBatteryLeft).

Mit throws kann man eine exception weitergeben.

Der Handler (try – catch) reicht das Problem nicht mehr weiter nach oben, sondern versucht das Problem zu lösen.

```
1 Scanner rd==null;  
2  
3 while (rd == null) {  
4     try {  
5         rd = new Scanner(new FileReader(input.next()));  
6     } catch (MyException) {  
7         System.out.println("File is too big!");  
8     } catch (IOException ex) { \\\ IOException is a SuperType of FileNotFoundException  
9         System.out.println("Cannot open file!")  
10    }  
11 }
```

Nur der Erste Catch Block wird ausgeführt, alle anderen werden übersprungen.

```
1 static Scanner getScanner(String n) throws FileNotFoundException, MyException {  
2     File f = new File(n);  
3     if (f.exists() && f.length() > 1000) {  
4         throw new MyException();
```

```
5     }
6     return new Scanner(f);
7 }
8
9 class MyException extends Exception {
10    // Here can be something but not mandatory.
11 }
```

The following was presented in the lecture on 27. November 2018.

8 Generische Programmierung

8.1 Collection

Ein Objekt das eine Ansammlung von Daten speichert.

8.2 ArrayList

Wie kann man ArrayList aufsetzen, dass wir diese Methoden für alle Arten von Objekten nutzen können?

Daraus folgt der Typ der Elemente sollte ein Parameter sein,
Java erlaubt dies mit generic types.

```
1 ArrayList<Type> name = new ArrayList<Type>();
```

Den Type kann man nun zwischen den Symbolen angeben.

8.3 Typeparameter

```
1 class MyType<T> {
2     T intern;
3     String s;
4
5     public MyType() {}
6
7     public MyType(T,i) {
8         intern = i;
9         s = i.toString();
10    }
11
12    public String toString() {
13        return s
14    }
15 }
16
17 myType<Point> mine = MyType<Point>(new Point(1,2));
```

8.4 Wrapper Klasse

```
1 ArrayList<int> list = new ArrayList<int>(); // Throws Error! Needs to be a Wrapper Class!
2 ArrayList<Integer> list = new ArrayList<Integer>();
3 ArrayList<Double> list = new ArrayList<Double>();
```

Das Umwandeln eines Basistyps in den entsprechenden Wrapper Typ wird als “boxing” bezeichnet. Das zurückwandeln nennt man unboxing.

The following was presented in the lecture on 30. November 2018.

8.5 compareTo

```
1 String a = "alice";
2 String b = "bob";
3 if (b.compareTo(a) > 0) {
4     System.out.println("The first is bigger than the second.");
5 }
```

8.6 Collections

```
1 ArrayList<String> list1 = new ArrayList<String>();
2 for (int i = 0; i < s.length; i++) {
3     list1.add(new String(s[i]));
4 }
5 Collections.sort(list1);
```

8.7 Interface Comparable

```
1 public interface Comparable<E> {
2     public int compareTo(E other);
3 }
```

Eine Klasse kann das Interface Comparable implementieren und so eine natürliche Ordnung für ihre Exemplare definieren.

```
1 class OwnClass implements Comparable<OwnClass> {
2     String item;
3     //Constructors
4     public int comareTo(Ownclass myobject) {
5         String compareItem = myobject.item;
6         return item.compareTo(compareItem);
7     }
8 }
```

8.8 Ternary Operator

```
1 max = (a>b) ? a : b;
```

The following was presented in the lecture on 04. December 2018.

8.9 Collections

Wichtige Java Service Klasse: Collections.

8.10 Mengen (Sets)

Eine Sammlung eindeutiger und einzigartiger Elemente. (Dublikate sind erlaubt). Folgende Operationen können ausgeführt werden: add, remove, contains.

Set kann in folgenden Klassen implementiert werden: HashSet und TreeSet.

8.10.1 HashSet

Bei einem HashTable wird einem String eine ganze Zahl zugeordnet und beim Index dieser ganzen Zahl eingetragen. Dadurch kann man immer in konstanter Zeit den String finden.

8.11 Tree Set

Basiert auf einem binären Baum und Abrufe können in $\mathcal{O}(\log n)$ gemacht werden.

8.12 Linked Hash Set

Speichert Elemente in der Reihenfolge in der sie in die Menge hinzugefügt werden. Dadurch kann man sie in konstanter Zeit wieder abrufen.

```
1 List<String> list = new ArrayList<String>();
2 Set<Integer> set1 = new TreeSet<Integer>(); // Da es ein Set ist koennte man es
   nachtraglich zu einem HashSet aendern => Grosse Flexibilitaet
3 Set<String> set2 = new HashSet<String>(list);
```

8.13 Operationen in Sets

Union addAll

Intersection retainAll

Difference removeAll

8.14 Ordnungsrelationen

HashSet: Elemente werden in beliebiger Reihenfolge gespeichert

TreeSet: Aufsteigend (nach compareTo) ArrayList: Werden in der Reihenfolge des Hinzufügens gespeichert.

8.15 For each Schleife

Folgender Code geht durch HashSet und TreeSet in derselben Reihenfolge durch.

```
1 for (type name: collection) {  
2     //statements;  
3 }  
4 for (String name: myMap.keySet()) {  
5     String name = myMap.get(name);  
6 }
```

8.16 Map

Hält eine Menge Schlüssel und eine Sammlung von Werten, wobei jeder Schlüssel mit einem Wert assoziiert ist. (Python: Dictionary)

```
1 Map<String, String> myMap = new HashMap<String, String>(); // Or TreeMap  
2 myMap.put("ETH", "is cool");  
3 myMap.get("ETH");  
4 myMap.remove("ETH");
```

8.17 KeySet

Die Methode keySet liefert die Menge (Set) aller Keys in der Abbildung.

```
1 myMap.keySet();
```

8.18 Values

Die Methode values liefert eine Ansammlung aller in der Map auftretenden Werte.

```
1 myMap.values();
```

8.19 Umkehrfunktion der Abbildung

Es ist erlaubt, eine Map von Mengen, oder eine Liste von Listen oder sonst zu definieren.
Muss eine Abbildung von Keys auf eine Menge von values sein.

```
1 Map<Integer, Set<String>> myMap = new HashMap<Integer, Set<String>>();
2 myMap.put(3, new TreeSet<String>());
3 myMap.get(3).add("FirstEntry");
4 myMap.get(3).add("SecondEntry");
5 myMap.get(3); // [FirstEntry, SecondEntry]
```

Example 1. WordCount

```
1 array = [allWords];
2 Map<String, Integer> wordCount= new HashSet<String, Integer>();
3 for (int i=0; i<array.length; i++) {
4     if (w.contains(array[i])) {
5         int currentCount = wordCount.get(array[i]) + 1;
6         wordCount.put(array[i], currentCount);
7     } else {
8         wordCount.put(array[i], 1);
9     }
10 }
11 for (String name: wordCount.keySet()) {
12     int count = myMap.get(name);
13     if (count >= 1000) {
14         System.out.println(name + " " + count);
15     }
16 }
```

The following was presented in the lecture on 07. December 2018.

8.20 Iteration

In einer For each Schleife durch ein Set darf das Set nicht verändert werden. Deshalb braucht man einen Iterator. Ein Objekt das einem Klienten erlaubt, die Elmente einer Ansammlung zu besuchen.

```
1 Set<Double> scores = new HashSet<Double>();
2 for (double score : scores) {
3     System.out.println(score);
4 }
```

Ein Iterator ist ein Objekt das einem Klienten erlaubt, die Elemente einer Ansammlung zu besuchen. Deshalb kann man an einer beliebigen Position ein Element ansehen oder entfernen. Sie hat eine hasNext und next Methode.

```
1 Iterator<Integer> itr = scores.iterator();
2 while (itr.hasNext()) {
3     int score = itr.next();
4     System.out.println(score);
```

```
5     if (score(<10) {  
6         itr.remove();  
7     }  
8 }  
9 System.out.println(scores);
```

Oder mit einem keySet

```
1 Iterator<String> itr = scores.keySet().iterator();
```

9 Abstrakte Datentypen

Spezifikation einer Ansammlung von Daten und der Operationen, die mit diesen Daten ausgeführt werden können. Wir wissen nicht wie der Datentyp implementiert ist, brauchen das jedoch auch nicht zu wissen.

Es ist eine gute Idee die Variablen für Ansammlungen als Variable des ADT Interface Typs zu deklarieren.

9.1 Stack

Eine Ansammlung zu der Elemente hinzugefügt werden können und aus der Elemente entfernt werden können. LIFO: Last in first out.

The following was presented in the lecture on 11. December 2018.

10 Systematisches Programmieren

Um rationale Zahlen darzustellen ist es wichtig die Zahlen richtig darzustellen, dafür ist der gcd hilfreich. Mit einem Zahlenpaar kann man rationale Zahlen genau darstellen. Ein Kommentar kann klären wer für die korrekte Implementierung verantwortlich ist.

10.1 Hoare Logik

Ein Ansatz wie man über ein Programm logische Schlüsse ziehen kann.

1. Vorwärts und rückwärts schliessen
2. Genauere Definition von Aussagen, Vor- und Nachbedingungen

10.1.1 Vorwärts schliessen

- Bestimmt was sich aus den ursprünglichen Annahmen herleiten lässt.

- Sehr praktisch wenn eine Invariante gelten soll.
- Simuliert die Ausführung des Programms
- Leicht zu verstehen, jedoch führt es dazu dass viele Details festgehalten werden müssen.

10.1.2 Rückwärts schliessen

- Bestimmt hinreichende Bedingungen die ein gegebenes Ergebnis garantieren.
- Wenn das Ergebnis erwünscht ist, dann folgt aus den Bedingungen die Korrektheit.
- Ist das Ergebnis unerwünscht, dann reichen die Bedingungen um einen Bug zu generieren.
- Oft von grossem praktischem Nutzen. Man muss verstehen was jede Anweisung zum Erreichen eines bestimmten Zustands beiträgt.
- Es gibt eine neue Sicht auf ein Programm

10.2 Terminologie

Die Annahme, die vor der Ausführung eines Statements gilt, ist die Precondition.
 Die Aussage, die nach der Ausführung gilt, ist die Postcondition

10.2.1 If-Statements

Die Precondition für den then-Block und den else-Blocks (eines If-Statements) beinhaltet das Ergebnis des Tests

Die Postcondition nach dem If-Statement ist die Disjunktion der Postconditions des then und -else Blocks

10.3 Notation

Statt die Pre/Postconditions in Kommentaren verwendet man geschweifte Klammern. Dazwischen steht eine logische Aussage.

The following was presented in the lecture on 14. December 2018.

10.4 Hoare Triple

Eine Hoare Triple besteht aus zwei Aussagen und einem Programmsegment

$$\{P\} \quad S \quad \{Q\} \tag{3}$$

$$\text{Precondition } \text{Programmsegment } \text{Postcondition} \tag{4}$$

Wenn P gilt und man dann S ausführt, dann gilt danach Q.
Andernfalls ist das Hoare Triple ungültig.

10.5 Assert

Eine Aussage in Java kann durch eine assert Statement ausgedrückt werden, Wenn zur Laufzeit expression nicht gültig ist, dann wird ein Laufzeitfehler generiert.

```
1 assert expression;
```

10.6 Hoare Triple - Zuweisung

Eine Variable wird durch eine andere Variable ersetzt. Q' ist gültig genau dann wenn $P \Rightarrow Q'$

10.7 Hoare Triple - Folgen von Anweisungen

Triple ist gültig wenn es eine Aussage R gibt sodass

1. $\{P\}S1\{R\}$ ist gültig und
2. $\{R\}S2\{Q\}$ ist gültig

10.8 Hoare Triple - If-Statements

Triple ist gültig wenn es eine Aussages Q1, Q2 gibt sodass

1. $\{P \wedge b\}S1\{Q1\}$ ist gültig und
2. $\{P \wedge !b\}S2\{Q2\}$ ist gültig und
3. $Q1 \vee Q2 \Rightarrow Q$

Beispiel

Example 2.

```
1 {x>0}
2 if(a>10) {y=2x;}
3 else {y=a*x;}
4 {y>0}
```

```
1 {(x > 0) \wedge (a < 10)}
2 y=2x
3 {y \geq 2}
4
5 {(x > 0) \wedge !(a < 10)}
```

```
6 y=ax
7 {y ≥ 10}
8
9 (y ≥ 2) ∨ (y ≥ 10) ⇒ y > 0
```

10.9 Aussagen über Zustände

Wenn gilt $P_1 \Rightarrow P_2$ dann gilt P_1 ist stärker und P_2 ist schwächer als P_1

Beim Rückwärtsschliessen sollte man immer die schwächste mögliche Precondition suchen. Da man so auch auf die Stärkeren schliessen kann.

Ohne Schleifen und Methoden gibt es für jedes Programmsegment S und jede Postcondition Q eine eindeutige schwächste Precondition. Abgekürzt $\text{wp}(S, Q)$

Die schwächste Precondition ist true.

Wenn wir einer Variable einen Wert zuweisen dann müssen wir eventuell die Variablennamen ändern, sonst kann es Komplikationen geben.

Eine swap Methode kann nützlich sein um Werte zu vertauschen.

The following was presented in the lecture on 18. December 2018.

10.10 Vorbedingung

Es interessiert uns immer die schwächste Vorbedingung. Das macht es leichter die Vorbedingung der Vorbedingung zu zeigen.

10.11 Loop

Example 3. Man muss eine Invariante suchen.

```
1 // assume: x >=x
2 y=0; i=0;
3 // invariant: y = sum(1,i)
4 while(i != x) {
5     // y = sum(1,i) and i != x
6     i++;
7     // y = sum(1,i-1)
8     y+=i;
9     // y = sum(1,i-1)+i
10 }
11 // i = x and y = sum(1,i)
12 // assert: y = sum(1,x)
```

Um Aussagen über die Ausführung des Loops zu machen brauchen wir eine Invariante. Invariante und der Schleifen Test müssen stark genug sein um zu zeigen, dass die Postcondition des Rumpfs auch die Invariante impliziert.

Invariante und der Schleifen Test müssen stark genug sein um die Postcondition der Schleife zu zeigen.

10.12 Hoare Logik - Invariante

Ein Hoare Triple ist gültig wenn es eine Invariante I gibt so dass:

1. Invariante gilt zu Beginn $P \Rightarrow I$
2. Nach Ausführen des Rumpfes gilt die Invariante wieder $\{I \wedge B\}S\{I\}$
3. Invariante impliziert Postcondition. $(I \wedge !B) \Rightarrow Q$

Invarianten dürfen weder zu stark noch zu schwach sein.

10.13 Methodologie um Invarianten zu finden

1. Bestimme zuerst die Invariante und lasse die anderen Schritte leiten.
Was bringt uns jede Iteration näher an das Ziel?
Was muss nach jeder Iteration gelten?
2. Schreibe einen Rumpf der die Invariante gültig macht
3. Bestimme den Loop Test so, dass Test-ist-false die Postcondition impliziert
4. Schreibe die Initialisierung so, dass dieser Code die Invariante sicher stellt.

10.14 Chiara Problem

Man hat rote, weisse und blaue Kugeln, die in einem Array sortiert werden müssen.
Es ist besser den beliebigen Teil in der Mitte zu haben. Links hat man die roten Kugeln und rechts die Blauen.

The following was presented in the lecture on 21. December 2018.

10.15 Abstract

Bedeutet dass es nur ein abstraktes Interface ist.
Muss von allen subclass implementiert werden.
Dadurch kann man Interfaces implementieren, ohne alle Methoden zu deklarieren, muss aber in subclasses gemacht werden.

10.16 Inner Classes

Innere Klasse ist eine Klasse die innerhalb einer anderen Klasse definiert sind.

Introduction to Programming - Exercise

Prof. Thomas R. Gross
TA: Mickey

Contents

1	Introduction	2
1.1	Organisation	2
1.2	Preview	2
1.3	Gerade Zahl - Good exam question	2
1.3.1	With recursion	2
1.4	EBNF - Graphical	2
1.5	Vereinfachung der EBNF Regel	2
1.6	Ternary Operator	3
1.7	Verzahnungen - Ugly Code	3
1.8	Test Cases	3
1.9	Rechner	4
2	Objects	6
2.1	Bienen	6
3	Compiler	8
4	Hoare Logic	8
4.1	Invariant	8
4.2	Palindrome	8
Getting help from Mickey		

- What am I trying to do?
- What am I doing?
- What am I expecting to happen?
- What is happening?

- Line number
- Any output/logs

The following was presented in the exercise on 25. September 2018.

1 Introduction

1.1 Organisation

git-scm.com/book/en/v2

1.2 Preview

blog.osteele.com/2008/05/my-git-workflow/

1.3 Gerade Zahl - Good exam question

```
ger_digit =≤ 0|2|4|6|8
ung_digit ≤ 1|3|5|7|9
all_digit ≤ ger_digit|ung_digit
gerade_zahl ≤ [+|-] {all_digit} ger_digit
```

1.3.1 With recursion

```
digit ≤ |all_digit digit
gerade_zahl ≤ [+|-] digit ger_digit
```

1.4 EBNF - Graphical

1. Ableitungsbaum
2. Syntax
3. Pathway

1.5 Vereinfachung der EBNF Regel

Simplify: $[A[A[A]]] \implies [A][A][A]$

The following was presented in the exercise on 09. October 2018.

1.6 Ternary Operator

```
1 int i = input > MAX ? MAX : input;
```

1.7 Verzahnungen - Ugly Code

```
1 import java.io.PrintStream;
2 import java.util.Scanner;
3
4 public class Verzahnung {
5
6     public static void main(String[] args) {
7         PrintStream output = new PrintStream(System.out);
8         Scanner scanner = new Scanner(System.in);
9         String s = scanner.nextLine();
10        String t = scanner.nextLine();
11        verzahnungen(s, t, output);
12
13        output.close();
14        scanner.close();
15    }
16
17    static StringBuffer __ = new StringBuffer("");
18
19    public static void verzahnungen(String __, String _, PrintStream $) {
20        if (_(__, 0xdeadbeef, _, $) & _(__, 0xcafebabe, __, $))
21            $.println(__);
22    }
23
24    static boolean _(String $, int 0, String _, PrintStream $$) {
25        if ((0 = $.length()) != 0) {
26            __.append($.charAt(0xf % 0b1111));
27            verzahnungen($.substring(1, 0), _, $$);
28            __.setLength(__.length() - 1);
29        }
30        return 0 == 0;
31    }
32 }
```

The following was presented in the exercise on 23. October 2018.

1.8 Test Cases

Always write good test cases.

The following was presented in the exercise on 06. November 2018.

public Überall Sichtbar

private Nur in der Klasse und Methoden sichtbar

no modifiers Also from Subclass sichtbar

1.9 Rechner

```
1 import java.util.Scanner;
2
3 public class Rechner {
4
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7         parse(scanner);
8     }
9
10    public static Expr parse(String input) {
11        return parse(new Scanner(input));
12    }
13
14    public static Expr parse(Scanner input) {
15        if (input.hasNext()) {
16            String next = input.next();
17            switch (next) {
18                case "+":
19                    return new Addition(parse(input), parse(input));
20                case "*":
21                    return new Multiplication(parse(input), parse(input));
22                case "-":
23                    return new Subtraction(parse(input));
24                default:
25                    return new Value(Integer.parseInt(next));
26            }
27        }
28        return new Value(0);
29    }
30
31 }
32
33 class Expression implements Expr {
34     Expr[] kids = new Expression[2];
35
36     @Override
37     public Expr[] children() {
38         return kids;
39     }
40
41     @Override
42     public int eval() {
43         return 0;
44     }
45
46     @Override
47     public String toString() {
48         return "" + this.eval();
49     }
50 }
```

```

51 }
52
53 class Addition extends Expression implements BinOp {
54
55     Addition(Expr expr, Expr expr2) {
56         kids[0] = expr;
57         kids[1] = expr2;
58     }
59
60     @Override
61     public int eval() {
62         return kids[0].eval() + kids[1].eval();
63     }
64
65     @Override
66     public Expr left() {
67         return kids[0];
68     }
69
70     @Override
71     public Expr right() {
72         return kids[1];
73     }
74
75 }
76
77 class Multiplication extends Addition implements BinOp {
78     Multiplication(Expr expr, Expr expr2) {
79         super(expr, expr2);
80     }
81
82     @Override
83     public int eval() {
84         return kids[0].eval() * kids[1].eval();
85     }
86 }
87
88 class Subtraction extends Expression implements UnaryOp {
89     public Subtraction(Expr parse) {
90         kids[0] = parse;
91     }
92
93     @Override
94     public Expr[] children() {
95         return new Expr[] { kids[0] };
96     }
97
98     @Override
99     public int eval() {
100         return -operand().eval();
101     }
102 }
103
104 @Override
105 public Expr operand() {
106     return kids[0];
107 }
108

```

```

109 }
110
111 class Value extends Expression implements Constant {
112     int data;
113
114     public Value(int parseInt) {
115         this.data = parseInt;
116     }
117
118     @Override
119     public int eval() {
120         return val();
121     }
122
123     @Override
124     public int val() {
125         return data;
126     }
127
128     @Override
129     public Expr[] children() {
130         return new Expression[0];
131     }
132 }
```

The following was presented in the exercise on 04. December 2018.

2 Objects

An object is always dynamic.

2.1 Bienen

```

1 import java.io.File;
2 import java.io.FileNotFoundException;
3 import java.io.PrintStream;
4 import java.util.HashMap;
5 import java.util.Map;
6 import java.util.Scanner;
7
8 public class Bienen {
9
10    public static void main(String[] args) throws FileNotFoundException {
11        String dateiName = "bienen.txt";
12        Scanner scanner = new Scanner(new File(dateiName));
13        PrintStream output = new PrintStream(System.out);
14
15        analyze(scanner, output);
16
17        output.close();
18        scanner.close();
```

```

19     }
20
21     public static void analyze(Scanner input, PrintStream output) {
22
23         Scanner item;
24         Person nextPerson;
25         Person maxIncome = new Person ("", "", 0, 0);
26         Person maxPart = new Person ("", "", 1, 0);
27         String maxCountry = "x";
28         Map<String, Integer> countryCount = new HashMap<String, Integer>();
29         countryCount.put(maxCountry, 0);
30
31         while(input.hasNextLine()) {
32             item = new Scanner(input.nextLine());
33             nextPerson = new Person(item.next(), item.next(), item.nextInt(),
34                         item.nextInt());
35             if (nextPerson.income > maxIncome.income) {
36                 maxIncome = nextPerson;
37             }
38             if (1.0*nextPerson.special/(nextPerson.income+nextPerson.special) >
39                 1.0*maxPart.special/(maxPart.income+maxPart.special)) {
40                 maxPart = nextPerson;
41             }
42             if (countryCount.get(nextPerson.country) == null) {
43                 countryCount.put(nextPerson.country, nextPerson.special);
44             } else {
45                 countryCount.put(nextPerson.country,
46                               countryCount.get(nextPerson.country)+nextPerson.special);
47             }
48         }
49
50         output.println(maxIncome.name+" "+(maxIncome.income+maxIncome.special));
51         output.println(maxPart.name+
52                         "+Math.round(100.0*maxPart.special/(maxPart.income+maxPart.special)));
53         output.println(maxCountry+" "+countryCount.get(maxCountry));
54     }
55
56     class Person {
57         String name;
58         String country;
59         int income;
60         int special;
61
62         public Person(String name, String country, int income, int special) {
63             super();
64             this.name = name;
65             this.country = country;
66             this.income = income;
67             this.special = special;
68         }
69     }

```

The following was presented in the exercise on 11. December 2018.

3 Compiler

CONST Pushes constant value c onto stack

LOAD Loads value of variable v and pushes it onto stack

STORE Pops a value from stack and stores it in variable v

OP Pops two values r and l, computes $l \oplus r$ and pushes the result back onto the stack

FUNC Pops a value x from the stack, computes $f(x)$ and pushes the result back onto the stack.

The following was presented in the exercise on 18. December 2018.

4 Hoare Logic

$\{P\}S\{Q\}$

S Program Code

P Precondition to the program S

Q Postcondition to the program S

4.1 Invariant

1. Invariant holds at the start $P \Rightarrow I$
2. Nach Ausführen des Rumpfes gilt die Invariante wieder $\{I \wedge B\}S\{I\}$
3. Invariante impliziert Postcondition. $(I \wedge !B) \Rightarrow Q$

4.2 Palindrome

```
1 import java.util.List;
2 import java.util.ArrayList;
3 import java.util.Arrays;
4 import java.util.HashSet;
5 import java.util.Set;
6
7 public class Palindrome {
8     public static void main(String[] args) {
9         List<Integer> list = Arrays.asList(0, 1, 2, 3, 3, 2, 1, 0);
10        Set<List<Integer>> set = maxPalindrome(list);
```

```

11     System.out.println(set.toString());
12 }
13
14 public static Set<List<Integer>> maxPalindrome(List<Integer> input) {
15     checkForError(input);
16     Set<List<Integer>> set = new HashSet<List<Integer>>();
17     int n = input.size();
18     for (int i = n; i > 0; i--) {
19         for (int j = 0; j + i <= n; j++) {
20             List<Integer> list = new ArrayList<Integer>();
21             for (int q = j; q < j + i; q++) {
22                 list.add(input.get(q));
23             }
24             if (checkIfPalindrome(list)) {
25                 set.add(list);
26                 i=0;
27             }
28         }
29     }
30     return set;
31 }
32
33 private static void checkForError(List<Integer> input) {
34     boolean error=false;
35     if (input==null)
36         error=true;
37     else if (input.isEmpty())
38         error=true;
39     else {
40         for (Integer i:input) {
41             if (i==null)
42                 error=true;
43         }
44     }
45     if (error) {
46         throw new RuntimeException("invalid list");
47     }
48 }
49
50 private static boolean checkIfPalindrome(List<Integer> list) {
51     int n = list.size();
52     for (int i = 0; i < n / 2; i++) {
53         if (!list.get(i).equals(list.get(n - 1 - i)))
54             return false;
55     }
56     return true;
57 }
58 }
```
