

Operations Research

Methodik

Konstantin Kuchenmeister
ge57joq
03717439

Technische Universität München

1. Lineare Optimierung
2. Ganzzahlige IPs
3. Nichtlineare Optimierung
4. Spieltheorie

Stand: 17.07.2020 22:36

1. Lineare Optimierung

1.1 Konventionen

1.1.1 Normalform:

$$\begin{array}{ll} \text{Max } & c^T x \\ \text{s.t. } & Ax \leq b \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \text{Min } & c^T x \\ \text{s.t. } & Ax \geq b \\ & x \geq 0 \end{array}$$

1.1.2 Standardform: (mit $b \geq 0$)

$$\begin{array}{ll} \text{max } & c^T x \\ \text{s.t. } & Ax = b \\ & x \geq 0 \end{array}$$

1.1.3 Kanonische Form: Standardform + jede Gleichung (Constraint) enthält eine Variable mit Koeffizienten 1, welche einen Koeffizienten von 0 in allen anderen Gleichungen hat. (Umformung erfolgt bei Simplex durch Zeilensoperationen)

⇒ Umformung durch Einführung von Schlupfvariablen

1.2 Lösungswege für lineare Programme

1.2.1 Grafische Lösung: 1. Stelle für jeden Constraint eine lineare Gleichung der Form $y = mx + b$ auf (Ungleichungen werden dabei wie Gleichungen behandelt).

2. Zeichne für jede Nebenbedingung (d. h. die Nichtnegativitätsbedingung) die lineare Funktion in ein Koordinatensystem.

3. Markiere den zulässigen Bereich. (Kann durch Ungleichungszeichen abgespannen werden)

'Ungleichzeichen verändern' man x_2 mit -1 multipliziert habe muss.

4. Minimiere / Maximiere lineare Zielfunktionsfunktion: An Ecken des zulässigen Bereichs eingeschränkt und optimale Koordinaten ablesen.

1.2.2 Simplex-Tableau: 1. Forme LP in kanonische Form um.

1. Standardform
2. Rechte Seiten nichtnegativ
3. Kanonische Form?

2. Übernahme Koeffizienten in Simplex-Tafel

Base	Coefficients	Result	Beispielhaftes Simplex-Tafelau für ein LP mit 2 Constraints mit 2 Schlupfvariablen.
	x_1 x_2 x_3 x_4		
\leq		x	
x_3		1	
x_4		1	

	x_1	x_2	x_3	x_4	
\leq			1		x
x_3			1		
x_4			1		

Basisvariablen: Variablen, die für die kanonische Form "zuständig" sind. Sie sind 1 und der Rest 0 (in der Spalte)

Nichtbasisvariablen: "A - Basisvariablen"

3. Iteration: Finden einer Pivotzeile und Spalte

1. Pivotspalte: "negativer" Koeffizient der Zielfunktionszeile (Z-Zeile)

2. Pivotzeile: Elemente der Result-Spalte (=b) / Elemente der Pivotspalte

! Nur positive Elemente > 0 werden als Pivotzeile in Betracht gezogen!

3. Basistausch durchführen:
Eintretende Basisvariable: Variable der Pivotspalte
Austretende Basisvariable: Variable der Pivotzeile

4. LP durch Zeilenoperationen wieder in kanonische Form bringen.
(:= Spalte der neuen Basisvariable muss überall 0 sein bis auf die Zeile mit der Variable selbst)

5. Simplex terminiert wenn alle Elemente der Zi-Zeile > 0 sind.

6. Lösung ablesen:
Maximalwert = Resukt-Spalte der Zi-Zeile
Variablenwerte = Resukt-Spalte der Basisvariablen

1.2.3 Simplex mit Gross M-Methode: Wenn das LP trotz hinzufügen von Slackvariablen nicht in kanonischer Form ist.

1. Anfangsbasis:
- In der Zielfunktion wird $+M \cdot w_x$ hinzugefügt
- Im Constraint wird $+w_x$ hinzugefügt

2. Nun muss w_x aus der Zielfunktion eliminiert werden und der Simplex kann ganz normal durchgeführt werden.

1.2.4 Minimierungsproblem \rightarrow Maximierungsproblem

1.1 Zielfunktion mit -1 multiplizieren
2.1 Constraints $\Rightarrow 2$ Ungleichungen

Constraints, die schon in korrekter Form sind, können übernommen werden!

2.2 Constraints $\geq \rightarrow \leq$ (mit -1 multipliziert)

1.2.5 Normalform, Standardform, kanonische Form mit undefinierten Variablen

1. Neue Variable $-/+$ undefinierte Variable mit gleichen Koeffizienten zur Zielfunktion adden

2. siehe 1.2.4

1.2.6 Variablen zu einer unbeschränkten zusammenfassen?

1. Gleiche Koeffizienten, sowie in ZF und NB beide mit unterschiedlichem Vorzeichen.

1.2.4 Revidierter Simplex in Matrixschreibweise

$$\text{Formeln: } N' = B^{-1} \cdot N$$

$$b' = B^{-1} \cdot b$$

$$c'_N = (c_N - N'^T \cdot c_B)$$

$$\text{Anfangsbasis: } A = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ \text{(Koeffizienten der Constraints)} \end{pmatrix} \quad b = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad c = \begin{pmatrix} x_1 & x_2 & x_3 & \dots & x_n \\ \text{(Zielfunktionskoeffizienten)} \end{pmatrix}^T$$

1. Iteration:

$$\text{Basisvariablen: } x_B = \begin{pmatrix} x_{B1} \\ x_{B2} \\ \vdots \\ x_{Bn} \end{pmatrix} \quad \text{Nichtbasisvariablen: } x_N = \begin{pmatrix} x_{N1} \\ x_{N2} \\ \vdots \\ x_{Nn} \end{pmatrix} \quad \text{aufstellen}$$

$$1.1 \quad B \text{ aufstellen} = \begin{pmatrix} \text{(Koeffizienten)} \\ \text{der Basisvariablen} \end{pmatrix} \quad 1.2 \quad N \text{ aufstellen} = \begin{pmatrix} \text{(Koeffizienten)} \\ \text{der Nichtbasisvariablen} \end{pmatrix}$$

$$1.3 \quad c_B \text{ aufstellen} = \begin{pmatrix} \text{(Zielfunktionskoeffizienten)} \\ \text{der Basisvariablen} \end{pmatrix}^T \quad 1.4 \quad c_N \text{ aufstellen} = \begin{pmatrix} \text{(Zielfunktionskoeffizienten der NBV)} \end{pmatrix}^T$$

1.5 B^{-1} (Inverses) mit Gauß-Jordan berechnen

1.6 $B^{-1} \cdot N = N'$ berechnen

1.7 $c'_N = -(c_N - N'^T c_B)$ berechnen

1.8 $b' = B^{-1} \cdot b$ berechnen

1.9 "Negativstes" Element aus c'_N ist eintretende Basisvariable

1.10 Die Zeilen der eintretenden Basisvariable aus N' / die Spalten von b' teilen.

\Rightarrow Kleinstes Element tritt aus

[b/N]

2. Simplex terminiert, wenn alle Elemente aus $c_N \geq 0$

1.3 Sonderfälle

1.3.1 Keine Eckpunkte \Rightarrow Jede Lösung ist optimal (Wenn es keine Constraints gibt)
 \Rightarrow In Iteration 2 sind in der Pivotzeile nur Elemente ≥ 0

1.3.2 Prinzipale Unbeschranktheit, Prinzipale Unzulässigkeit

Es kann eine Pivotspalte gebunden werden, aber alle Elemente sind ≤ 0 . \Rightarrow Es kann keine Pivotzeile gefunden werden.

1.3.3 Mehrere optimale Lösungen

Wenn im Optimaltableau der Zielfunktionskoeffizient mindestens einer LBU den Wert 0 hat

1.3.4 Das Problem besitzt eine redundante Nebenbedingung

- Wenn eine \leq Nebenbedingung anderen Bedingungen eine nicht größere rechte Seite besitzt
- Wenn eine \geq Nebenbedingung anderen Bedingungen eine nicht kleinere rechte Seite hat

1.3.5 Prinzipale Unzulässigkeit / Prinzipale Unbeschranktheit

- Bei einem Constraint alle Koeffizienten positiv, abell negative Restwert-Spalte

1.4 Goal Programming

1.4.1 Weighted Goal Programming:

1. S^+ und S^- Variable für jeden Constraint einführen
2. Koeffizienten vor S -Variablen nach Gewichtung anpassen

1.4.2 Preemptive Goal Programming

1. "Wichtiges"-Ziel als Zielfunktion lösen
2. "Unwichtiges"-Ziel als Zielfaktor mit "wichtigem" Zielfaktor = Max (1.) als Constraint und lösen.

1.5 Sensitivitätsanalyse

1.5.1 In welchem Bereich kann ein Zielfunktionskoeffizient schwanken? (ohne Veränderung Optimaltableau)

0. Dem Koeffizienten der Variable x_i ein "+ Δ " hinzufügen der Form $Cx_i \rightarrow (C+\Delta)x_i$

1. Dem Koeffizienten c von x_i im Optimaltableau ein $-\Delta$ hinzufügen $\begin{array}{c|c} x_i & x_i \\ \hline C & C-\Delta \end{array} \rightarrow \begin{array}{c|c} z & c-\Delta \end{array}$
(Wenn $c=0$ dann nur $-\Delta$)

Fall 1: x_i ist eine Basisvariable

1. Den Zielfunktionskoeffizienten (meist $-\Delta$) durch Zeilenumperation $\oplus \Delta$ wieder auf 0 bringen

2. Constraints mit anderen neuen Zielfunktionskoeffizienten aufstellen, sodass alle Zielfunktionskoeffizienten $\pm \Delta \geq 0$ sind

3. Schwankungsbereich für Δ ablesen

4. 3. Lin. Zielfunktion einsetzen

Fall 2: x_i ist keine Basisvariable

1. Constraint aus Koeffizient aufstellen, sodass $z \geq 0$ damit er nicht in die Basis aufgenommen wird

2. Schwankungsbereich ablesen

3. 2. Lin. Zielfunktion einsetzen

1.5.2 Grenzen der Nebenbedingungen, ohne dass sich optimale Basis verändert

1. Spalte mit Schlupfvariante für i -te Nebenbedingung $\oplus \Delta_i$ auf die Result Spalte

2. Constraints aus Result-Spalte aufstellen, sodass $z_i \pm \Delta_i \geq 0$

3. Grenzen aufstellen

(4. Verfügbarer Bestand \pm Grenzen) falls nicht nach Änderung im Optimaltableau sondern erlaubt gefragt ist
Wenn zusätzliches Angebot $>$ Grenze:

1. Zugehörige Zeile t -mal auf z -Zeile (Zeile die negativ wird)

2. Constraints $z \geq 0$ aufstellen und Intervall festlegen

3. Neuer Schattenpreis = obere Grenze von Intervall in z einsetzen.

4. Obere Grenze von 1.5.2.3. Schattenpreis + obere Grenze aus 2. = neuer Schattenpreis

1.5.3 Schattenpreise sind die Koeffizienten der Nichtbasisvariablen und stellen die marginalen Kosten für eine weitere Einheit dieses Gutes x_i dar. Ergebnisse von c_N^T im Optimalitätsfall

1.5.3 Tauschen: "s₃ mit s₁ im Verhältnis 2:1"

1. ZJ · s₃ von der Ergebnisspalte abziehen und Abhängigkeiten aufstellen
2. J · s₁ auf 1 addieren
3. Intervall für J aufstellen
4. Obere Grenze = #s₃

1.5.4 Wenn Grenzen verletzt werden

- (1. Austrittende BV: Rechte Seite = 0 bei Intervallgrenze
(2. + Zeile auf Zielfunktion und Intervallgrenzen aufstellen)

1.5.5 Hinzufügen einer neuen Variable

Geg.: Neue Variable v mit Koeffizientenvektor (a₁, a₂, a₃)
Ges.: Zielfunktionskoeffizient c_v

$$1. c_v \geq \sum_{i \in I} \text{Schattenvariable}_i \cdot a_i$$

1.5.6 Theorie - Was bedeuten Zielfunktionskoeffizienten der Strukturvariablen im optimalen Tableau?

1. Wenn Koeffizient > 0 sollte diese Strukturvariable nicht produziert werden, da sie nicht in der optimalen Basis ist
2. Wenn sie fälschlicherweise doch produziert wird, verringert sich das Ergebnis um den Strukturvariable:

1.6 Dualität

1.6.1 Umformung von primalem zu dualem Programm

$$\begin{array}{ll} \text{Max } F(x) = c^T x & \text{Min } F^*(w) = b^T w \\ \text{s.t. } Ax \leq b & \Rightarrow \text{s.t. } A^T w \geq c \\ x \geq 0 & w \geq 0 \end{array}$$

Beispiel:

Max s.t.	$10x_1 + 20x_2$ $x_1 + x_2 \leq 10$ $6x_1 + 9x_2 \leq 72$ $x_1, x_2 \geq 0$	Min s.t.
	$10x_1 + 20x_2$	$100y_1 + 720y_2$
	\leq	$y_1 + 6y_2 \geq 10$
	\leq	$y_1 + 9y_2 \geq 20$
		≥ 0

Dualisierungsregeln:

1. Ein primales Maximierungsproblem ist ein duales Minimierungsproblem und umgekehrt.
2. Normalform bleibt Normalform ($\text{Min-Normalform} \rightarrow \text{Max-Normalform}$ und umgekehrt)
3. Aus einer unbeschränkten Variable folgt eine Gleichheitsrestriktion

1.6.2 Komplementärer Slackspace

$$(Ax^* - b)^T y^* = 0 \quad (A^T y^* - c)^T x^* = 0$$

(Nebenbedingung Zeile - Rechte Seite) $x_i y_{2i+1} \text{ nehmen} = 0$

1. Dual zulässige Lösung?
(In Constraints einsetzen)

2. Slackspace

3. Primal zulässig?
(In Constraints einsetzen)

1.7 Payoff Matrix

$$A:$$

Sch	S1	S2	V1	V2
Sch	x_1	\dots	x_n	
S1	\vdots	\ddots	\vdots	\vdots
S2	\vdots	\ddots	\vdots	\vdots
V1	y_1	y_2	y_3	
V2				

$$\begin{aligned} \text{Max } g &= y_1 x_1 + y_2 x_2 + \dots + y_n x_n \\ &\leq g \\ x_1, x_2, \dots, x_n &\geq 0 \end{aligned}$$

Normalform

$$\begin{aligned} \text{Max } g^+ - g^- &= g \\ x_1, x_2, \dots, x_n, g^+, g^- &\geq 0 \\ x_1 - x_2 - x_3 &\leq -1 \\ x_1 + x_2 + x_3 &\geq 1 \\ y_1, y_2, y_3, g^+, g^- &\geq 0 \end{aligned}$$

$$B: \begin{aligned} \text{Max } \pi^+ - \pi^- &= \pi \\ y_1, y_2, \dots, y_n, \pi^+, \pi^- &\geq 0 \end{aligned}$$

1.6.3 Schwache Dualität

Sei x die zulässige Lösung des primalen Problems und y die zulässige Lösung des dualen Problems.
Dann gilt: Primal- Z_f -Wert \leq Dual- Z_f -Wert

- Primal: Max

Die optimale duale Lösung ist obere Schranke der primalen.

- Primal: Min

Die optimale duale Lösung ist untere Schranke der primalen Lösung

1.6.4 Starke Dualität

Optimale Primallösung = Optimale Duallösung \rightarrow komplementärer Schluss

2. Ganzzahlige Lineare Optimierung

2.1 Boolesche Logik vs. Mathematische Gleichungen

- Logisches ODER = $x_1 \vee x_2 \vee x_3 = x_1 + x_2 + x_3 \geq 1$

- Logisches UND = $x_1 \wedge x_2 \wedge x_3 = \begin{cases} x_1 \geq 1; \\ x_2 \geq 1; \\ x_3 \geq 1; \end{cases}$

- Implikation = $x_1 \rightarrow x_2 = \neg x_1 \vee x_2 = x_1 \leq x_2 = x_2 - x_1 \geq 0$

- Äquivalenz = $x_1 \Leftrightarrow x_2 = x_1 - x_2 \leq 0 = x_2 - x_1 \leq 0$

- Negation = $\neg_{1A} = 1 - A$

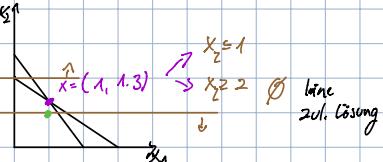
2.1 Lösen von ganzzahligen LPs

2.1.1 Branch-and-Bound

1. Optimales Tableau aus LP berechnen

2. Für nicht ganzzahlige Variable verzweigen mit $x \leq \text{Wert}_1$ $x \geq \text{Wert}_2$ und constraint hinzufügen

Grafisch:



Rechnerisch: Max ...

st. ...

...

$$x_1 \leq 1$$

$$x_2 \geq 2$$

2.1.2 Schnittebenenverfahren:

- 1. Lösung des LP bestimmen
- 2. Gleichung aus aus nicht ganzzahliger Zeile im Optimaltableau aufstellen
- 3. Alle ganzen Teile nach links, reelle Teile nach rechts sortieren und aufspalten in eine ganze und reelle Zahl
- 4. Linker Seite ≤ 0 mit in Constraints aufnehmen

! lauert zur nächstkleineren ganzen Zahl!

$$y = \lfloor y \rfloor + (y - \lfloor y \rfloor)$$

2.2 Wichtige IPs

Zuordnungsproblem: N Arbeiten auf N Maschinen verteilen wodurch Verbrauch minimiert wird

$$\min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\text{s.t. } \sum_{i \in I} x_{ij} = 1 \quad \text{für alle Jobs } j \in J$$

$$\sum_{j \in J} d_{ij} x_{ij} \leq s_i \quad \leftarrow \text{Maschinenkapazität für alle Maschinen } i \in I$$

$$x_{ij} \in \{0,1\}$$

Rucksackproblem: Menge an Projekten um höchstem Nutzen mit gegebenem Budget

- n Projekte, Budget W

- w_j : Kosten bei Annahme Projekt j

- p_j : Ertrag des Projekts

- $x_j = 1$ wenn Projekt j ausgewählt wird, sonst 0

$$\max \sum_{j \in J} p_j x_j$$

$$\text{s.t. } \sum_{j \in J} w_j x_j \leq W$$

$$x_j \in \{0,1\} \text{ für alle } j \in J$$

Behälterproblem: Minimiere die Anzahl der Behälter i

$$\min \sum_{i \in I} y_i$$

$$\text{s.t. } \sum_{i \in I} x_{ij} = 1 \quad \text{für alle Gegenstände} \quad (\text{jeder Gegenstand darf nur einmal zugewiesen werden})$$

$$\sum_{j \in J} d_{ij} x_{ij} \leq s_i \quad \text{für alle } i \in I \text{ Behälter}$$

$$x_{ij}, y_i \in \{0,1\}$$

2.3 Column Generation

1. LP aufstellen (In der Zielfunktion Kosten abziehen)

2. B_0, b_1, c_B aus Angabe aufstellen (siehe revidierter Simplex)

Iteration i :

1. B^{-1} berechnen

2. $C_B^T B^{-1}$ berechnen (Kostenvektor für Basis)

3. LP mit 2. Lösung (≥ 1)

4. $b' = B^{-1} \cdot b$ berechnen

5. $y' = B^{-1} \cdot y$ berechnen

6. b'_i / y'_i von austretende Basisvariable zu bestimmen

2.4 Approximationsalgorithmen: Rucksackproblem

1. Tabelle aufstellen der Form

Gegenstand	1	2	\dots	n
Gewicht	2	x	\dots	n
Importance	1	2	x	n

2. Tabelle aufstellen $\begin{array}{c|ccccc} & 1 & \dots & n \\ \hline 1 & & & & & \\ \vdots & & & & & \\ n & & & & & \end{array}$ und ausfüllen

Approx. Algorithmus berechnet für jedes $\epsilon > 0$ eine möglich Lösung die höchstens um Faktor $(1+\epsilon)$ vom Optimum entfernt ist. $1/(1-\epsilon) = \text{Approximationsfaktor}$

$$\theta = \text{Skalierungsfaktor} < \frac{\epsilon \cdot v_{\max}}{n} \Rightarrow \text{Runde } v_i^* = \left\lfloor \frac{v_i}{\theta} \right\rfloor \text{ und lösse diese Instanz}$$

2.5 Algorithmus von Ford-Fulkerson

1. Startfluss wählen (Nullfluss)

2. Finde einen augmentierenden Pfad von s nach t und bestimme den Fluss
 3. Ersetze jede Kante durch eine Hin- und Rückkante
 3.1 Kapazität Hin-Kante: Restkapazität
 3.2 Kapazität Rück-Kante: aktueller Fluss im augm. Netzenwerk

Fluss $^h_a =$ Fluss + a falls die Hin-Kante enthalten

Fluss $^r_a =$ Fluss - a falls die Rück-Kante enthalten

4. Abbruchbedingung: Es gibt keine augmentierenden Pfade mehr

Minimales s-t-Schnittr: Partition sodass $S = [X, V \setminus X]$ sodass $s \in S$ und $t \in V \setminus X$

- Minimales Schnitt = Maximaler Fluss

- Nach Schnitt existieren keine Kanten mehr von s nach t

2.6 Edmonds-Karp-Algorithmus

1. Finde \min_{weg} von s nach t die minimal sind (Kreistensuche)
2. Berechne max Flow des paths (minimale Kanten) und passe Residualnetzwerk an (siehe 2.5)
3. Abbruchbedingung: keine \min_{weg} existierende Kante mehr

2.7 Optagische Methode

0. Beachten: 1. Matrix muss quadratisch sein, also ggf. Dimensionstausch (0) hinzufügen
2. Muss ggf durch Multiplikation mit -1 in Minimierungsaufgabe gewandelt werden
1. Zeilenminimum abziehen } (Von allen Einträgen)
 2. Spaltenminimum abziehen }

3. Minimale Abdeckung überprüfen

- 3.1 Minimale Überdeckung aller σ durch Geraden
- 3.2 Wenn Anzahl geraden < Anzahl Spalten \Rightarrow sonst σ .

4. Kleinstes nicht überdecktes Element bestimmen

- 4.1 Von allen nicht überdeckten Elementen abziehen
- 4.2 Auf alle doppelt überdeckten Elemente aufzufaddieren

5. Für jede Spalte und Zeile 1 Null auswählen und mit Anfangsmatrix vergleichen

2.8 Unimodularität: enthält keine quadratische Submatrix deren Determinante keinen Wert aus $\{-1, 0, 1\}$ annimmt.

1. Alle Einträge $\in \{-1, 0, 1\}$
 2. Jede Spalte enthält maximal 2 Koeffizienten $\neq 0$
 3. Es existiert eine Partition (N_1, N_2) der Zeilen, sodass
- hinterher
bedingt,
- 3.1 Wenn 2 Nicht-Null-Einträge einer Spalte dasselbe Vorzeichen haben, dann ist ein Eintrag in Zeilen N_1 das andere in N_2
3.2 Wenn 2 Nicht-Null-Einträge einer Spalte unterschiedliche Vorzeichen haben, dann sind beide in einer Partition.
 \rightarrow Wenn A vollständig unimodular, dann auch $-A$ und $[A, I]$
- if (true) = unimodular
if (false) = kann trotzdem unimodular sein

2.9 Matroide: $M(E, I)$

1. $\emptyset \in I$
2. Heforditärbedingung Wenn $I \subseteq I$ und $|I| \leq |I'|$ dann ist $|I'| \subseteq I$
3. $A, B \in I$ $|A| \leq |B|$, $\exists X \in \mathcal{P}^A$, sodass $A \cup X \subseteq B$

2.10 Nearest Neighbor Algorithm

1. Beliebigen Knoten als Startknoten auswählen
2. Gerade Kante zum jeweils nächsten Knoten ziehen

2.11 Kruskal - Algorithmus

1. Alle Kanten entfernen
2. Alle Kanten nach Gewicht/Mitanz aufsteigend hinzufügen bis minimaler Spannbaum
! Kanten, die einen Kreis erzeugen würden, werden "überspringen" / nicht hinzugefügt

2.11 Algorithmus von Christofides (2-Approximation)

1. Verdopple alle Kanten und finde Euler-Tour (Aus MST = minimalem Spannbaum)
2. Laufe Tour entlang und überspringe alle Knoten die bereits besucht werden.

2.12 Algorithmus von Christofides (1.5-Approximation)

1. MST (Graph) berechnen
2. berechne ein Matching mit minimalem Gewicht (M) zwischen allen Knoten mit ungeradem Grad.
Ein Matching ist eine Menge von Kanten in einem Graph, die keinen gemeinsamen Knoten haben
3. Kombiniere M und T (verdoppe alle Kanten in T , die Teil des Matchings sind)
4. Laufe Tour entlang und überspringe bereits besuchte Knoten

2.13 Nearest Insertion Algorithmus (2-Approximation)

1. starte mit einem beliebigen Kreis der Länge 3
2. Füge Knoten hinzu, der kleinste Abstand zu einem bereits bestehenden Knoten hat.
3. Lösche alle "inneren Kanten"

3. Nichtlineare Optimierung

3.1 finden lokaler Minima / Maxima

1. Gradienten aufstellen $\nabla f(x, y) = \begin{pmatrix} \text{partielle Ableitung } x \\ \text{partielle Ableitung } y \end{pmatrix} \stackrel{!}{=} 0$

2. $I=0$ setzen und Lösungen für x & y bestimmen

3. $II=0$ setzen und Lösungen mit Hilfe von einsetzen von 2. bestimmen

4. Lösungen aus 3. zu kritischen Punkten zusammenfassen (x -Werte aus I in II für y einsetzen)

5. $\nabla^2 f(x, y)$ bilden = $\begin{pmatrix} \text{partielle Ableitung } x_1 \text{ partielle Ableitung } y_1 \\ \text{partielle Ableitung } x_2 \text{ partielle Ableitung } y_2 \end{pmatrix}$

6. Kritische Punkte in $\nabla^2 f(x, y)$ einsetzen und Eigenwerte berechnen

7. switch (Eigenwerte)

case negativ definit \Rightarrow lokales Maximum (beide Vorzeichen negativ)

case positiv definit \Rightarrow lokales Minimum (beide Vorzeichen positiv)

Sei: ein Eigenwert = 0

case indefinit \Rightarrow Sattelpunkt (unterschiedliche Vorzeichen der Eigenwerte)

3.2 Karush-Kuhn-Tucker Bedingungen (KKT)

1. Lagrangefunktion bilden: Zielfunktion - λ^* Nebenbedingungen; !Nebenbedingungen müssen in KKT berücksichtigt werden!

2. Partielle Ableitungen $L_x(x, \lambda) = 0$

2.1. X_1 : partielle Ableitung nach $x_1 \leq 0$

2.2. X_2 : partielle Ableitung nach $x_2 \geq 0$

3. keine Richtung verbessert den Zielfunktionswert: $x^* L_x(x, \lambda) = 0$

3.1. X_1 : $x_1 \cdot (2.1) = 0$

3.2. X_2 : $x_2 \cdot (2.2) = 0$

4. Erstreckbarkeit der Lösung: $L_\lambda(x, \lambda) \geq 0$

4.1. Lagrangefunktion abgeleitet nach λ 1

4.2. Lagrangefunktion abgeleitet nach λ 2

5. Komplementarität: $\lambda \cdot L_2(\sigma, \lambda) = 0$

$$S.1 \quad \lambda_1: \lambda_1 \cdot (4.1) = 0$$

$$S.2 \quad \lambda_2: \lambda_2 \cdot (4.2) = 0$$

6. Prinzipal & Dualer Zulässigkeit: $x_1, x_2, \lambda_1, \lambda_2 \geq 0$

$$6.1 \quad x_1, x_2, \lambda_1, \lambda_2 \geq 0$$

3.3 Konvexität f Wenn $f''(x) \geq 0$ für alle $x \in \mathbb{R}$

1. Ist die Funktion reellstetig differenzierbar? (Nicht differenzierbar)

case (2a)

- Ist die Funktion im eindimensionalen Bereich? \rightarrow 2 mal Ableiten

- Ist die Funktion im mehrdimensionalen Bereich?

case (nein) 2.B. (x)

$$f(\lambda \cdot x_1 + (1-\lambda)x_2) = \lambda \cdot f(x_1) + (1-\lambda) \cdot f(x_2) \rightarrow$$

Punkte auswählen nach Graph: welche könnten nicht korrekt sein

4. Spieltheorie

4.1 Nash-Gleichgewicht mit parameter

NG: keiner will seine Strategie verändern

	C	D
A	2,1	1,a
D	1,a	2,2

Wenn Zeilenspieler A, dann wählt er A/C
⇒ dann ist Spaltenspieler auch A/C wählt muss $a \leq 1$ sein