ADS: Algorithmen und Datenstrukturen 2

Teil 3

Christian Kahmann

Institut für Informatik
Abteilung Automatische Sprachverarbeitung
Universität Leipzig

25. April 2018

[Letzte Aktualisierung: 14/05/2018, 07:08]

Randomisierte Algorithmen

Idee: viele zufällige Entscheidungen führen oft trotzdem zu sehr guten Ergebnissen.
Anwendung vor allem zur Optimierung.

Stochastische Optimierungsalgorithmen

Problemstellung

Lösungsmenge X, Bewertungsfunktion $f : X \to \mathbb{R}$.

Finde globales Minimum von f

Grundprinzip

- Start: endliche Menge A von erlaubten Lösungen ("Population") mit $|A| = n$.
- Schleife:
  1. Erzeuge erweiterte Lösungsmenge B, $|B| = m > n$.
  2. Selektiere eine neue Population $A' \subset B$ mit $|A'| = n$ in der die besseren Lösungen angereichert sind.
- Abbruchbedingung

Stochastische Optimierungsalgorithmen

- Motiviation durch Prozesse in der Natur
- Evolution als Wechselspiel von Mutation (allergemeiner: Erzeugung von Variabilität) und Selektion (aus den vorhandenen Varianten)
- Schrittweise Energie-Minimierung in physikalischen Systemen (z.B. Gefrieren von Flüssigkeiten ...)

Random Generate and Test

- $A = \{x\}$, eine einzelne Lösung
- $B = \{x, y\}$, wobei $y \in X$ eine **zufällig** gezogene Lösung ist.
- $A' = \{y\}$ if $f(y) < f(x)$, sonst $A' = \{x\}$.

Keine Garantie, dass ein globales Minimum gefunden wird.
FRAGE: wie kann man die *Struktur* von f ausnutzen?
IDEE: Gute Lösungen werden "irgendwie" ähnlich zu anderen guten Lösungen sein.

“Moves”

Lasse nur bestimmte “Schritte” zu.
Einfachster Fall: $A = \{x\}$
Formal: Definiere eine (erlaubte) Nachbarschaft $N(x)$ für jedes $x \in X$.
Nun waehle ein $y \in N(x)$ mit Wahrscheinlichkeit $1/|N(x)|$.
Impliziert eine Graph-Struktur Γ
Notwendige Bedingung: Γ muss stark zusammenhängend sein
sonst kann der Prozess auf einer Zusammenhangskomponente eingesperrt bleiben, die kein globales Minimum enthält.

Fitness-Landschaften

Formale Beschreibung des Optimierungsproblems als Funktion über der Knotenmenge des Graphen Γ

- Die Frage nach geometrischen Eigenschaften wie
  - Zahl von lokalen Minima
  - Tiefe und Breite von Tälern
  - Sattelpunkte zwischen Tälern
  - Rauhheits-Maße

  und deren Einfluss auf die Optimierung ist Gegenstand der Forschung
- Veränderung des Move-Sets verändert die Landschaft und damit das Optimierungsverhalten.
- Suche problemangepasste Movesets. Idee: einzelne Moves sollten die Qualität der Lösung (meistens) nicht zu drastisch verändern.

Adaptive Walk

- **Start:** wähle $x \in X$ zufällig
- **Schleife:**
  1. wähle $y \in N(x)$ gleichverteilt
  2. falls $f(y) < f(x)$, setze $x \to y$.
- **Abbruchbedingung**

- Folge der Lösungen enspricht einem Pfad in Γ, entlang dessen f strikt abnimmt. ("Adaptive Walk")
- Lokales Minimum: $f(y) \geq f(x)$ für alle $y \in N(x)$.
- Pfade bleiben in lokalen Minima stecken. Bei geeigneter Abbruchbedingung erreichen alle Pfade ein lokales Minimum.
- PROBLEM: Lösungen können sehr schlechter Qualität sein, d.h. $f(x) \gg f(\text{min})$.

Gradient Descent

- **Start:** wähle $x \in X$ zufällig
- **Schleife:**
  1. bestimme die Menge $Z(x)$ aller $z \in N(x)$ für die $f(z)$ minimal über $N(x)$ ist.
  2. wähle $y \in Z(x)$ gleichverteilt
  3. falls $f(y) < f(x)$, setze $x \to y$.
- **Abbruchbedingung**: stop falls $f(y) \geq f(x)$.

- Folge der Lösungen enspricht einem Pfad in Γ, entlang dessen f so steil wie möglich abnimmt. (Weg des steilsten Abstieg, "Gradient Descent")
- Endet immer in einem lokalen Minimum.
- Teurer als adaptive walk, weil $N(x)$ durchsucht werden muss.

Metropolis-Walks

Um lokalen Minima zu entkommen, lasse auch "aufwärts" Schritte zu:

- **Start:** wähle $x \in X$ zufällig
- **Schleife:**
  1. wähle $y \in N(x)$ gleichverteilt
  2. (a) falls $f(y) < f(x)$, setze $x \to y$
     (b) sonst setze $x \to y$ mit Wahrscheinlichkeit

$$\exp(-(f(y) - f(x))/T)$$

- **Abbruchbedingung**

- Steil bergauf = exponentiell selten
- Eigenschaft: Wenn man **lange** genug wartet, besucht man zu einem bestimmten Zeitpunkt t die Lösung x mit Wahrscheinlichkeit

$$p(x) = \exp(-f(x)/T)/Z$$

- Für kleine T: Lösungen mit kleinen Funktionswerten werden angereichert.

Simulated Annealing

- IDEE: Reduziere die "Temperatur" T im Laufe der Simulation
- Kirkpatrick, Gelatt and Vecchi (1983) and V. Černý (1985)
- WICHTIGE ROLLE:
  "cooling schedule" T_k, typischerweise eine monoton fallende Funktion der Schrittzahl k
- Anwendung auf harte Probleme, für die keine effizienten (polynomialen) Algorithmen bekannt sind.

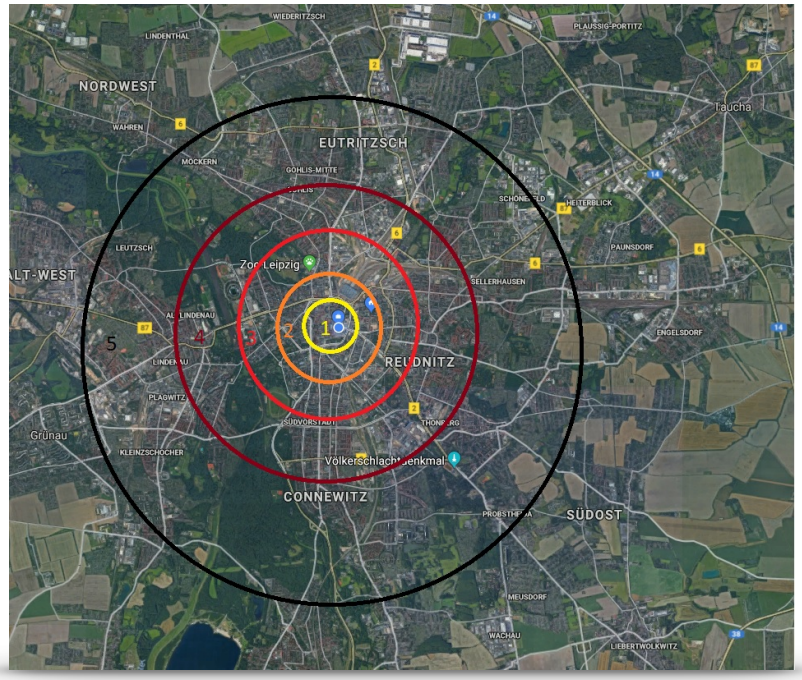
Beispiel

Ziel: finde eine optimale Wohnung

-Parameter: Fläche und Lage

-Eigenschaften einer Wohnung sind Preis und Prestige

- $Zufriedenheit\,(F, L) = \frac{Prestige\,(F, L)}{Preis\,(F, L)}$



Lage

Fläche / Preis	1	2	3	4	5
20m²	500	350	280	260	250
40m²	600	500	390	300	300
60m²	700	600	490	430	380
80m²	800	600	600	580	500
100m²	1200	1000	900	800	600

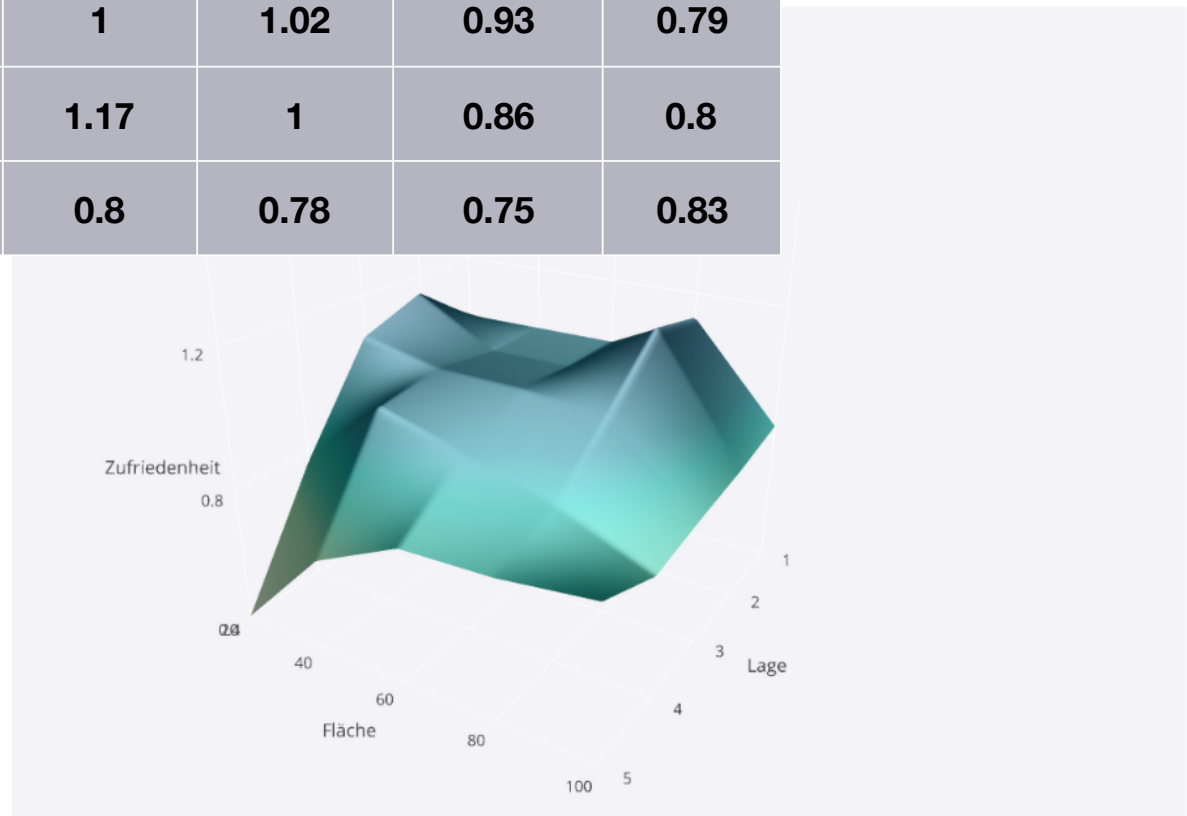
Lage

Fläche / Prestige	1	2	3	4	5
20m²	500	400	300	200	100
40m²	600	500	400	300	200
60m²	700	600	500	400	300
80m²	900	700	600	500	400
100m²	1000	800	700	600	500

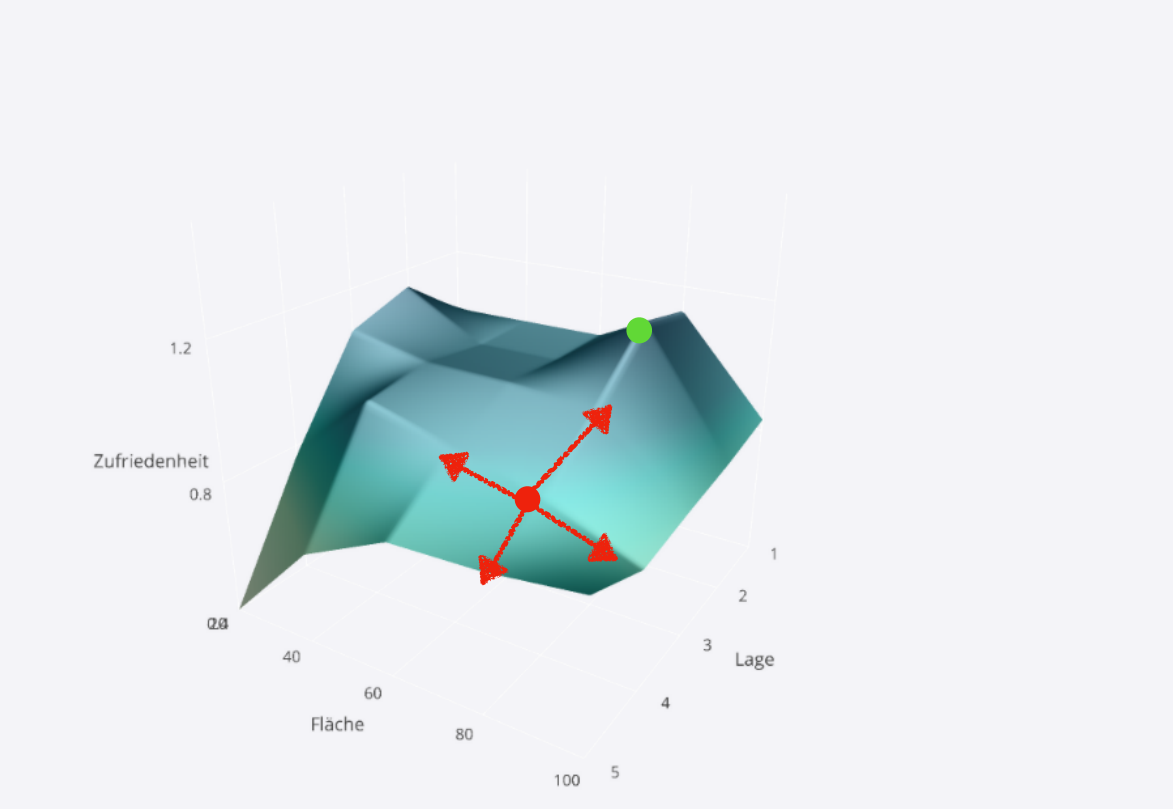
Beispiel

Bewertungsfunktion f: Zufriedenheit(F, L)

Zufriedenheit	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



Beispiel - Moves



Startpunkt

Zielpunkt

erlaubter Move

erlaubter Move: schrittweise Änderung von entweder Lage oder Fläche

versuche mit Abfolge von Moves Zielpunkt zu erreichen

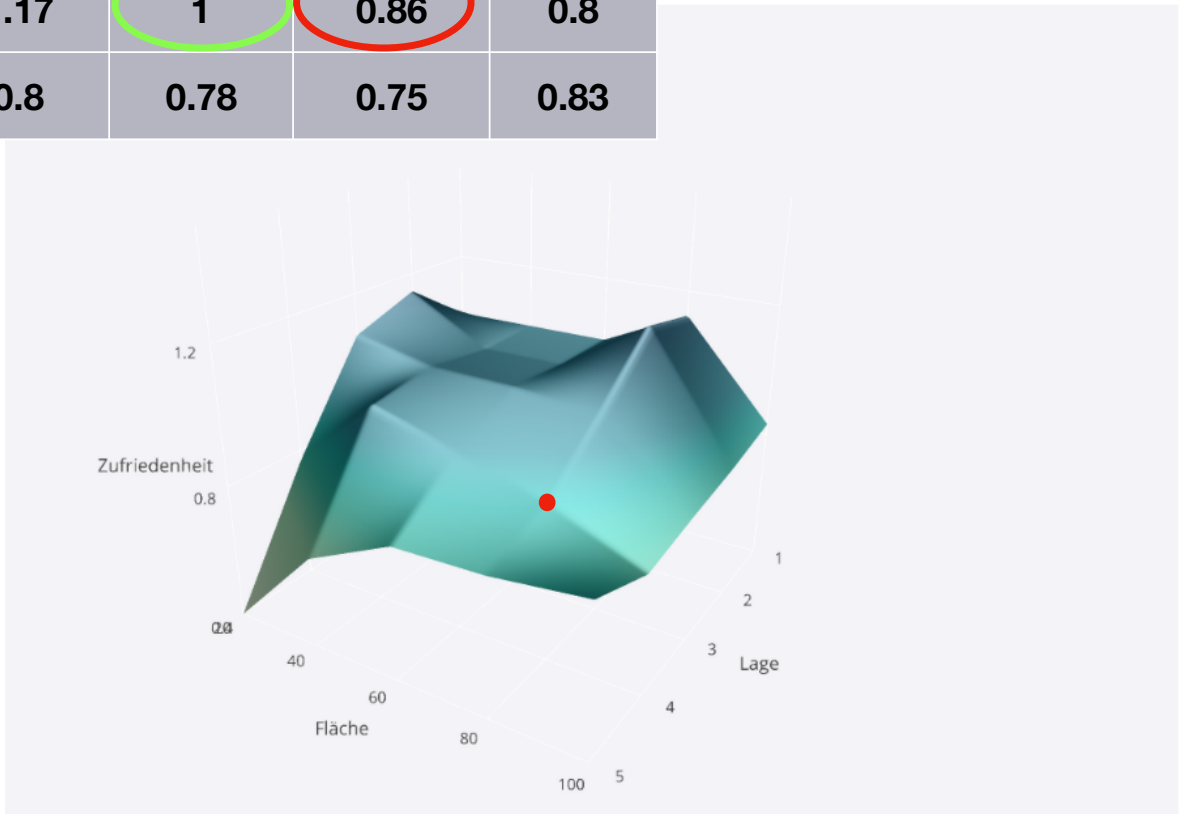
Adaptive Walk - Maximierung

- **Start:** wähle $x \in X$ zufällig
- **Schleife:**
  1. wähle $y \in N(x)$ gleichverteilt
  2. falls $f(y) > f(x)$, setze $x \to y$.
- **Abbruchbedingung**

- Folge der Lösungen enspricht einem Pfad in Γ, entlang dessen f strikt zunimmt. ("Adaptive Walk")
- Lokales Maximum: $f(y) \leq f(x)$ für alle $y \in N(x)$.
- Pfade bleiben in lokalen Maxima stecken. Bei geeigneter Abbruchbedingung erreichen alle Pfade ein lokales Maximum.
- PROBLEM: Lösungen können sehr schlechter Qualität sein, d.h. $f(x) \ll f(\min)$.

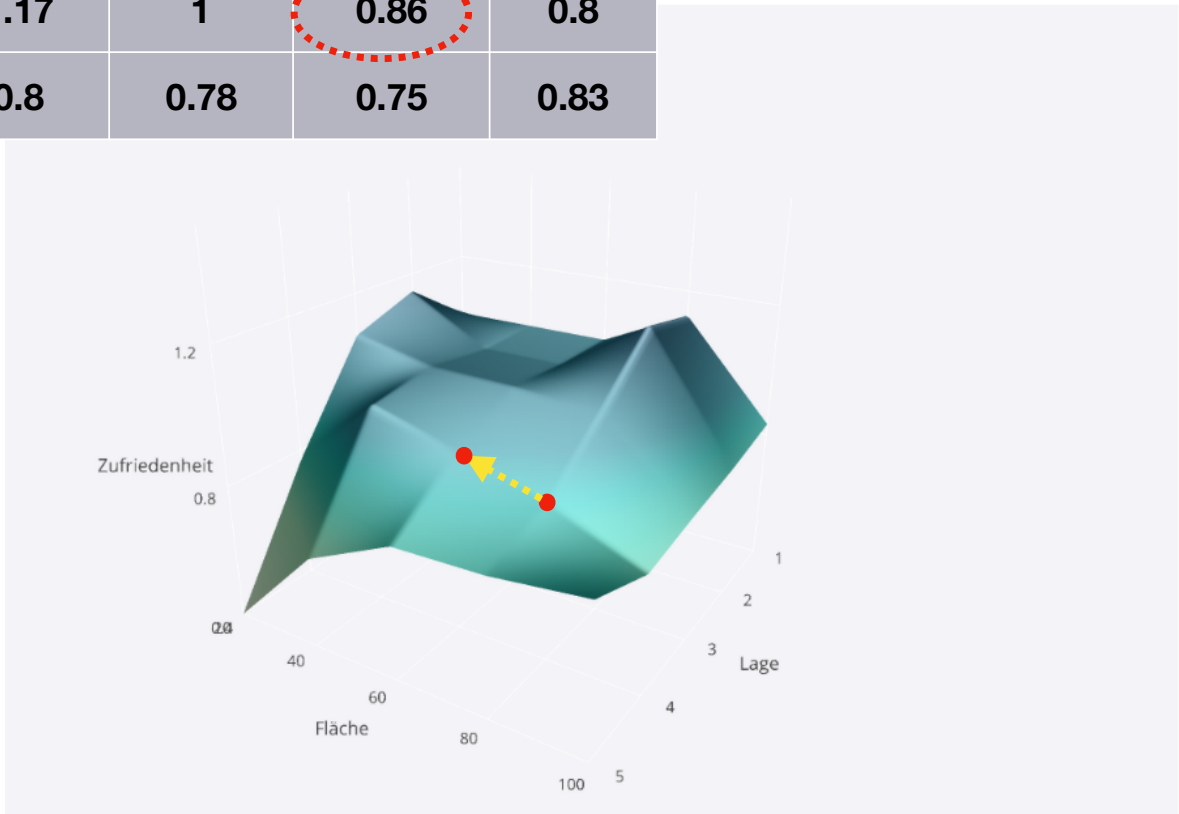
Beispiel - Adaptive Walk

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



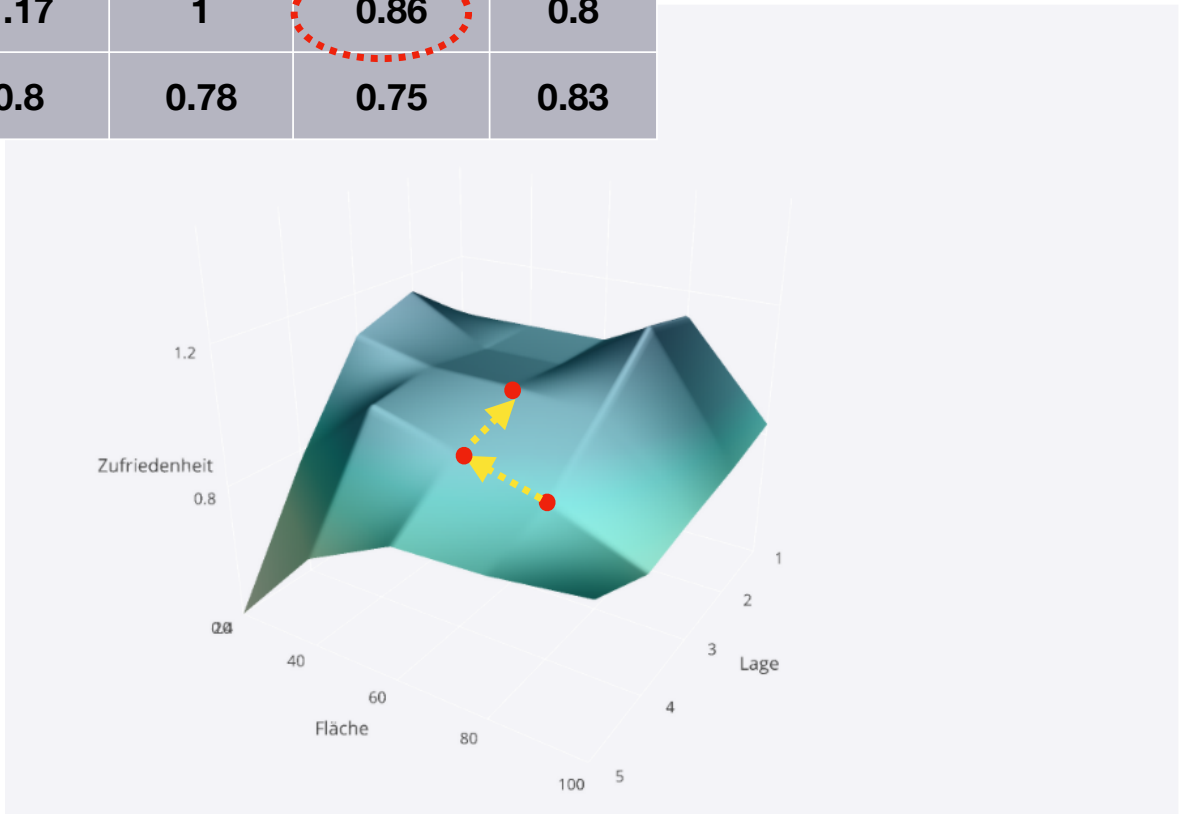
Beispiel - Adaptive Walk

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



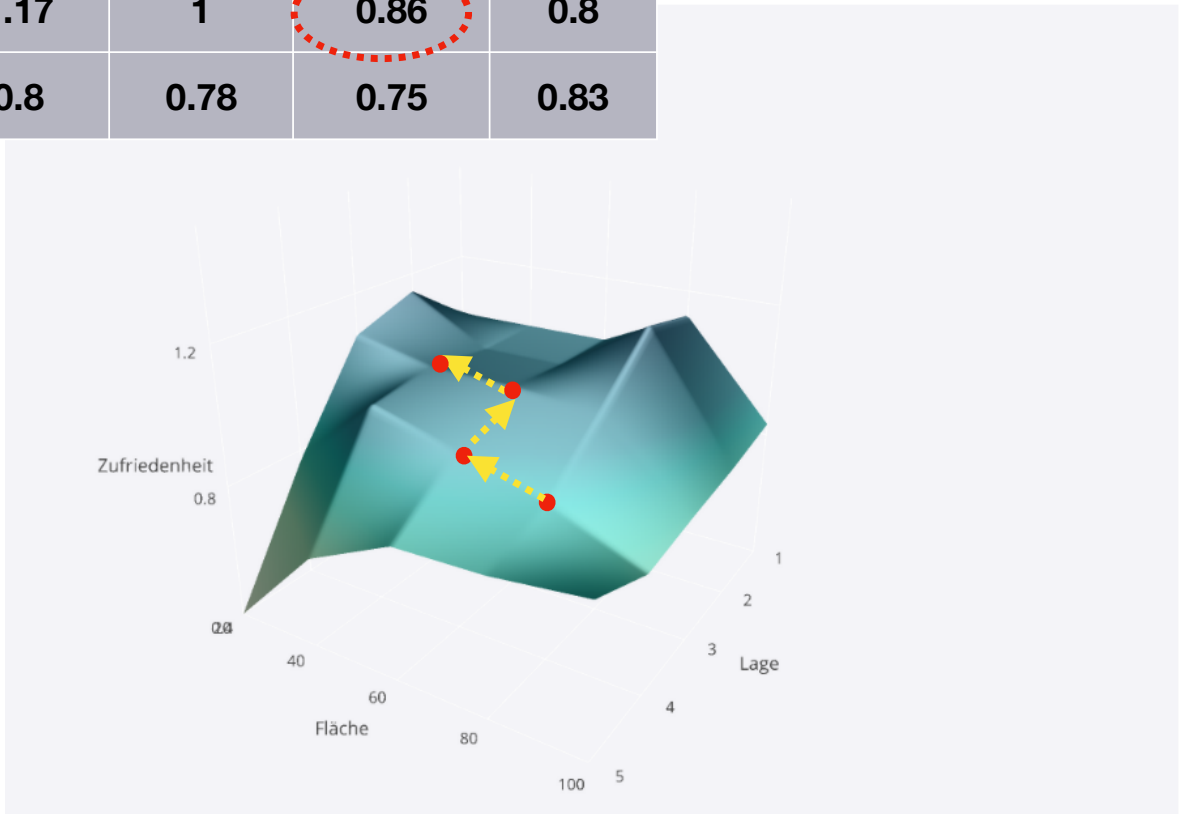
Beispiel - Adaptive Walk

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



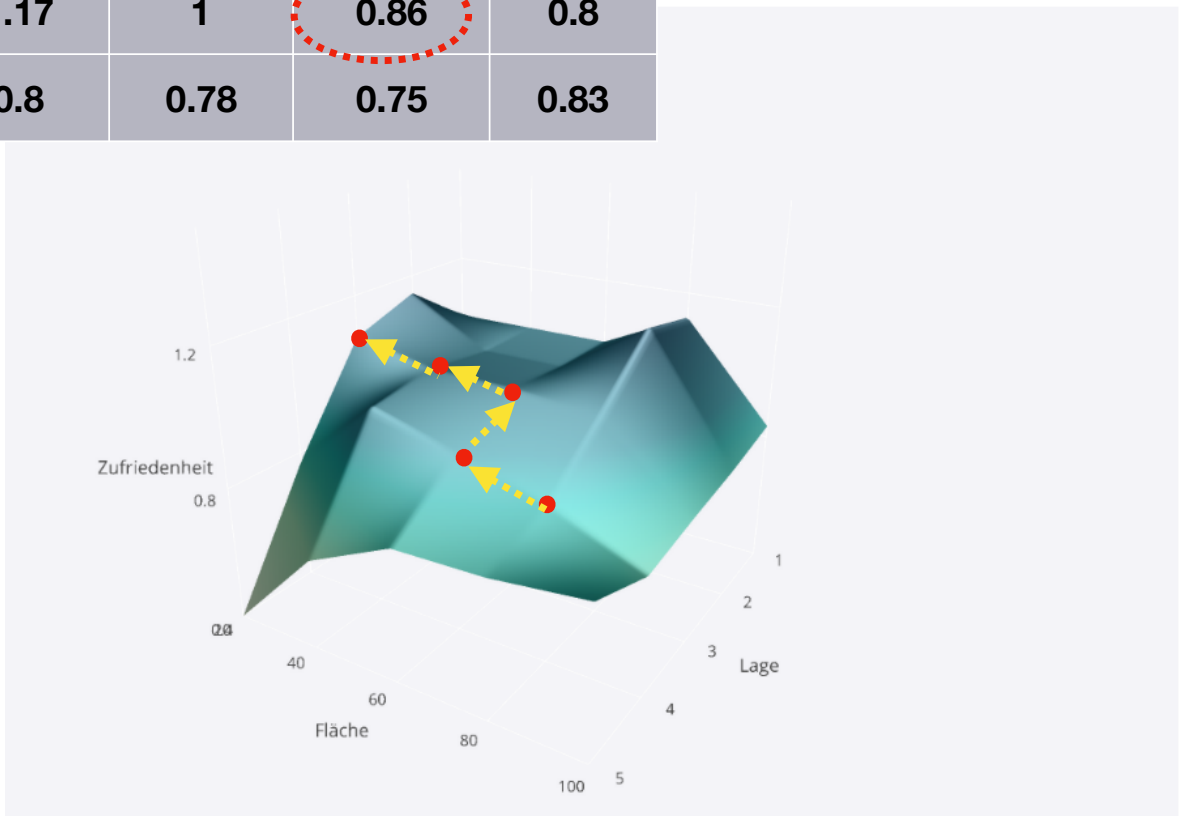
Beispiel - Adaptive Walk

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



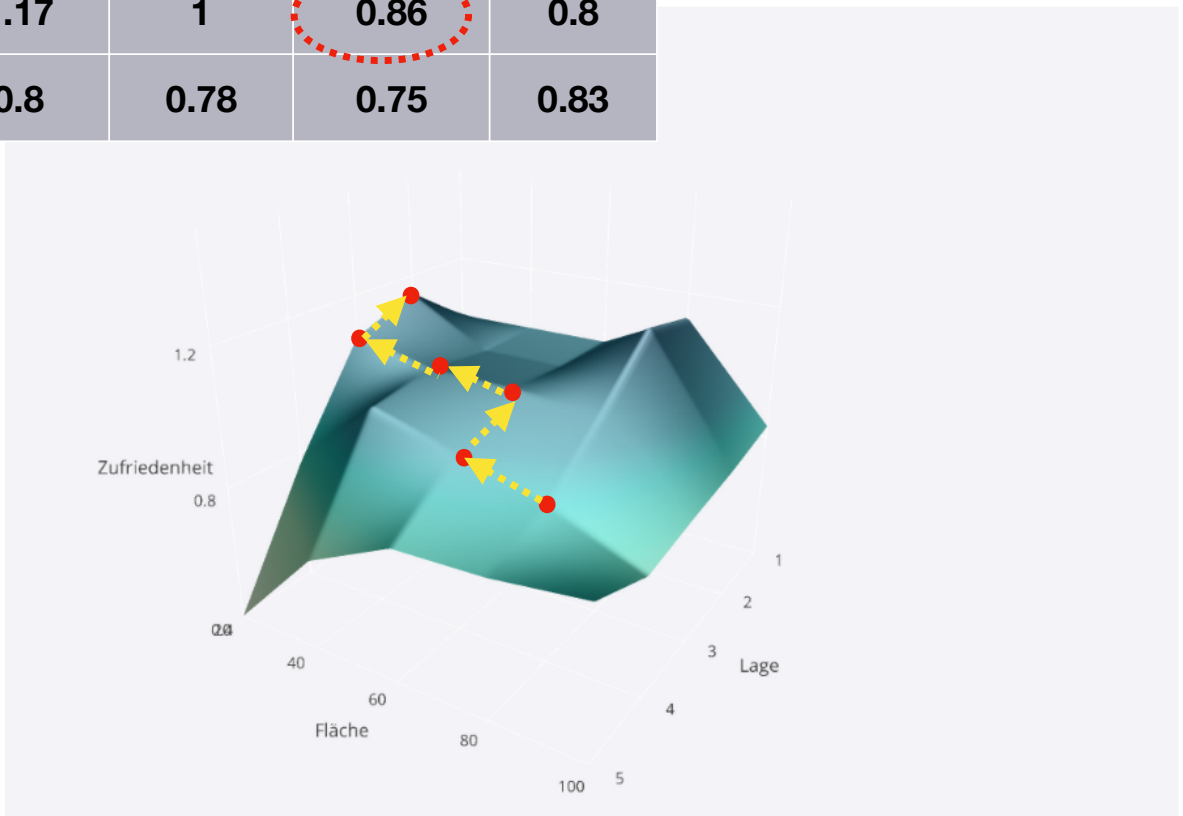
Beispiel - Adaptive Walk

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



Beispiel - Adaptive Walk

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



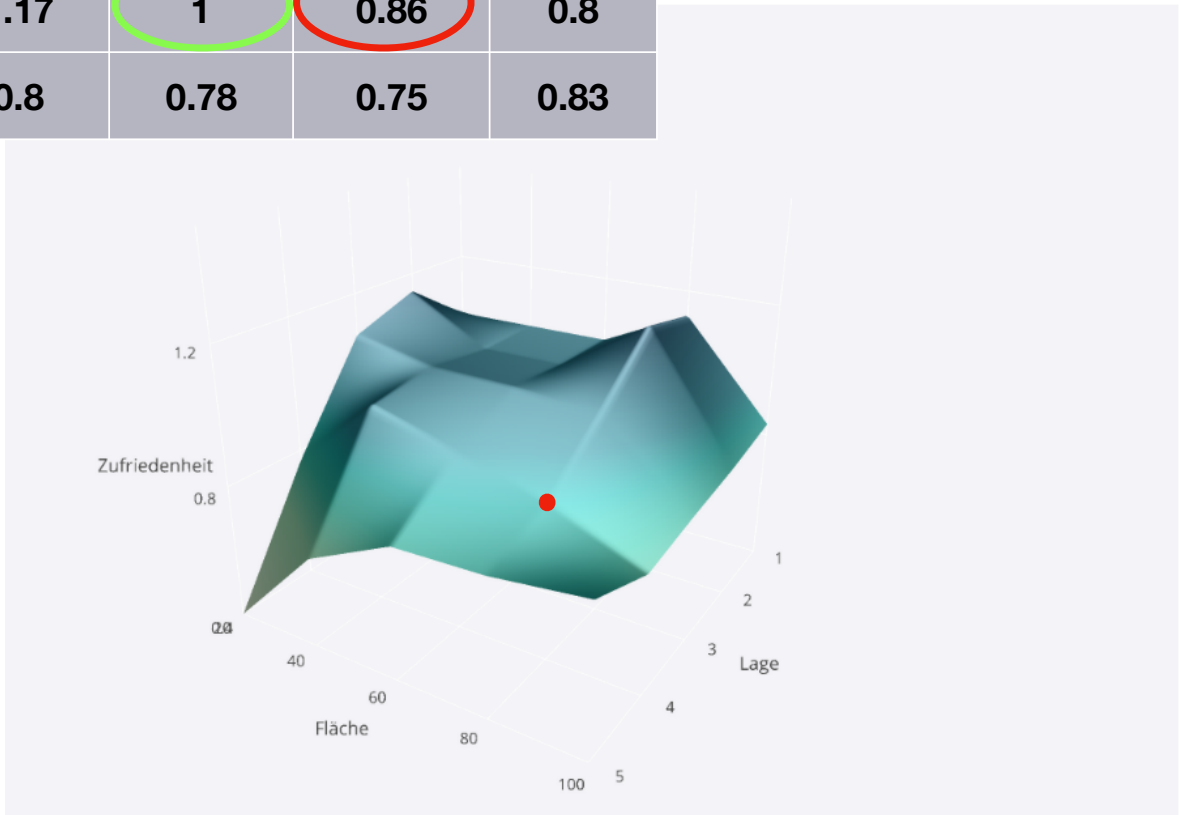
Gradient Descent - Maximierung

- **Start:** wähle $x \in X$ zufällig
- **Schleife:**
  1. bestimme die Menge $Z(x)$ aller $z \in N(x)$ für die $f(z)$ maximal über $N(x)$ ist.
  2. wähle $y \in Z(x)$ gleichverteilt
  3. falls $f(y) > f(x)$, setze $x \to y$.
- **Abbruchbedingung**: stop falls $f(y) \leq f(x)$.

- Folge der Lösungen enspricht einem Pfad in Γ, entlang dessen f so steil wie möglich zunimmt. (Weg des steilsten Anstiegs)
- Endet immer in einem lokalen Maximum.
- Teurer als adaptive walk, weil $N(x)$ durchsucht werden muss.

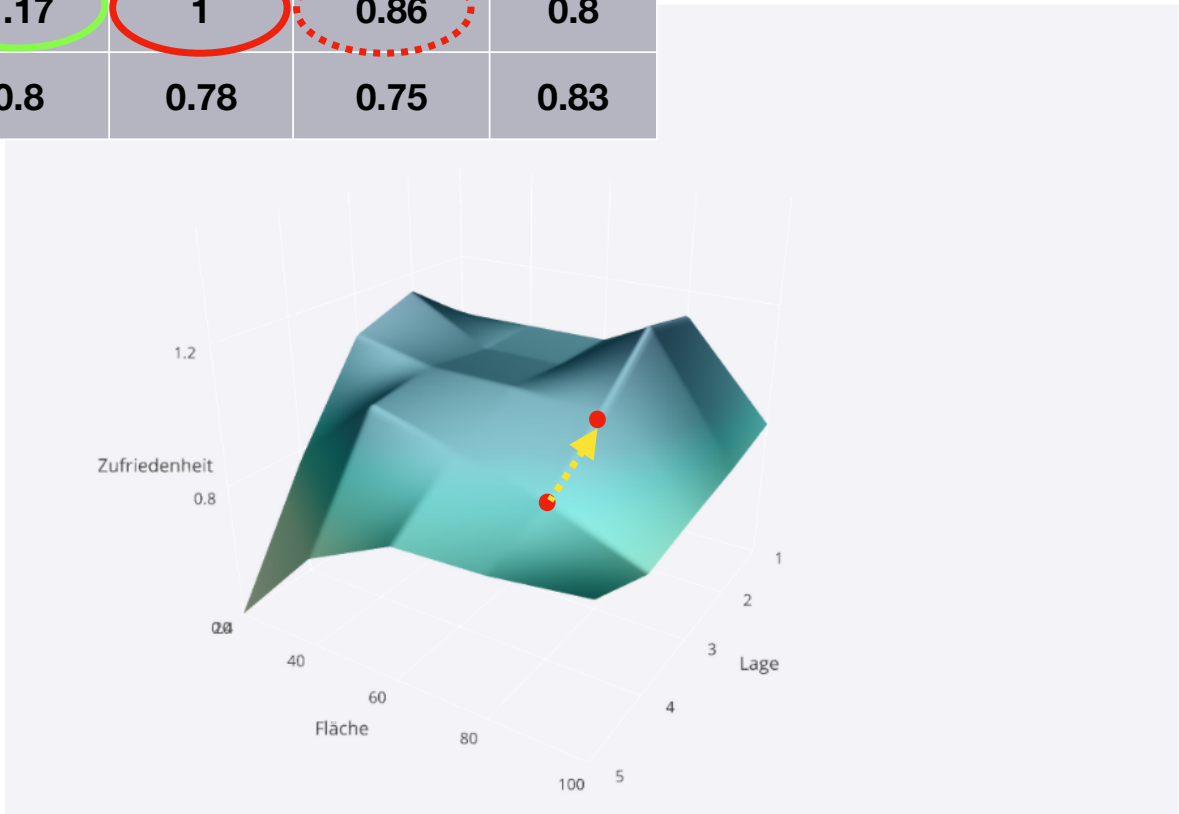
Beispiel - Gradient Descent

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



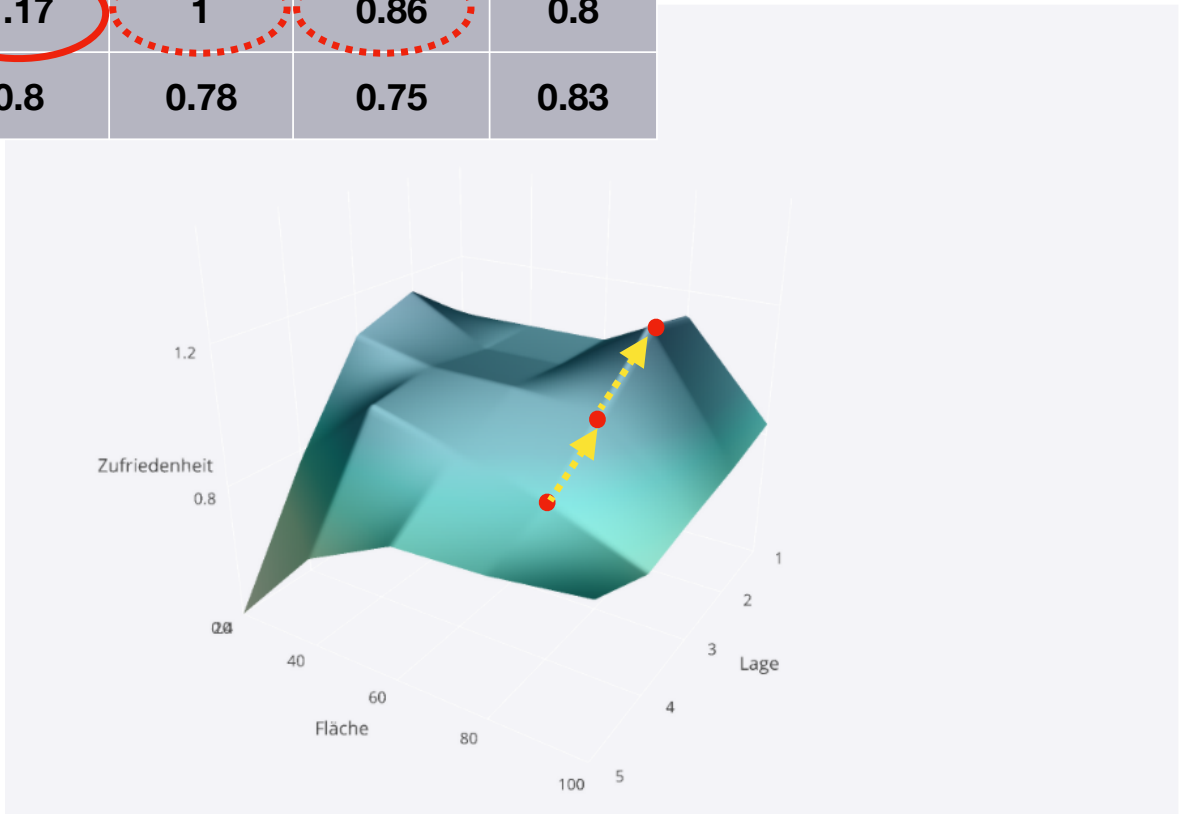
Beispiel - Gradient Descent

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



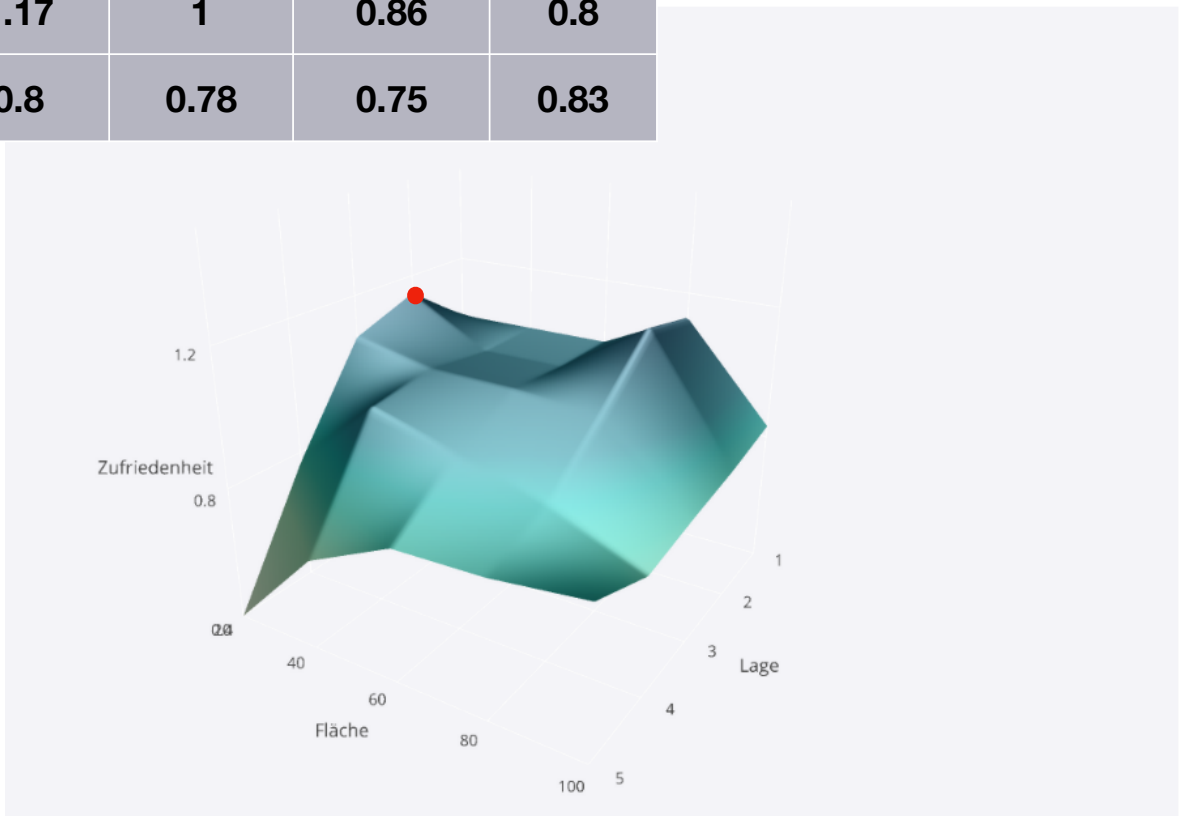
Beispiel - Gradient Descent

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



Beispiel - Gradient Descent

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



Metropolis-Walks - Maximierung

Um lokalen Maxima zu entkommen, lasse auch "abwärts" Schritte zu:

- **Start:** wähle $x \in X$ zufällig
- **Schleife:**
  1. wähle $y \in N(x)$ gleichverteilt
  2. (a) falls $f(y) > f(x)$, setze $x \to y$
     (b) sonst setze $x \to y$ mit Wahrscheinlichkeit

$$\exp(-(f(x) - f(y))/T)$$

- **Abbruchbedingung**

- Steil bergab = exponentiell selten
- Eigenschaft: Wenn man **lange** genug wartet, besucht man zu einem bestimmten Zeitpunkt t die Lösung x mit Wahrscheinlichkeit

$$p(x) = \exp(-f(x)/T)/Z$$

- Für kleine T: Lösungen mit großen Funktionswerten werden angereichert.

Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83

-für Adaptive Walk und Gradient Descent kein Erreichen der optimalen Lösung mehr möglich, da alle moves die Zufriedenheit verringern würden und somit nicht zulässig sind

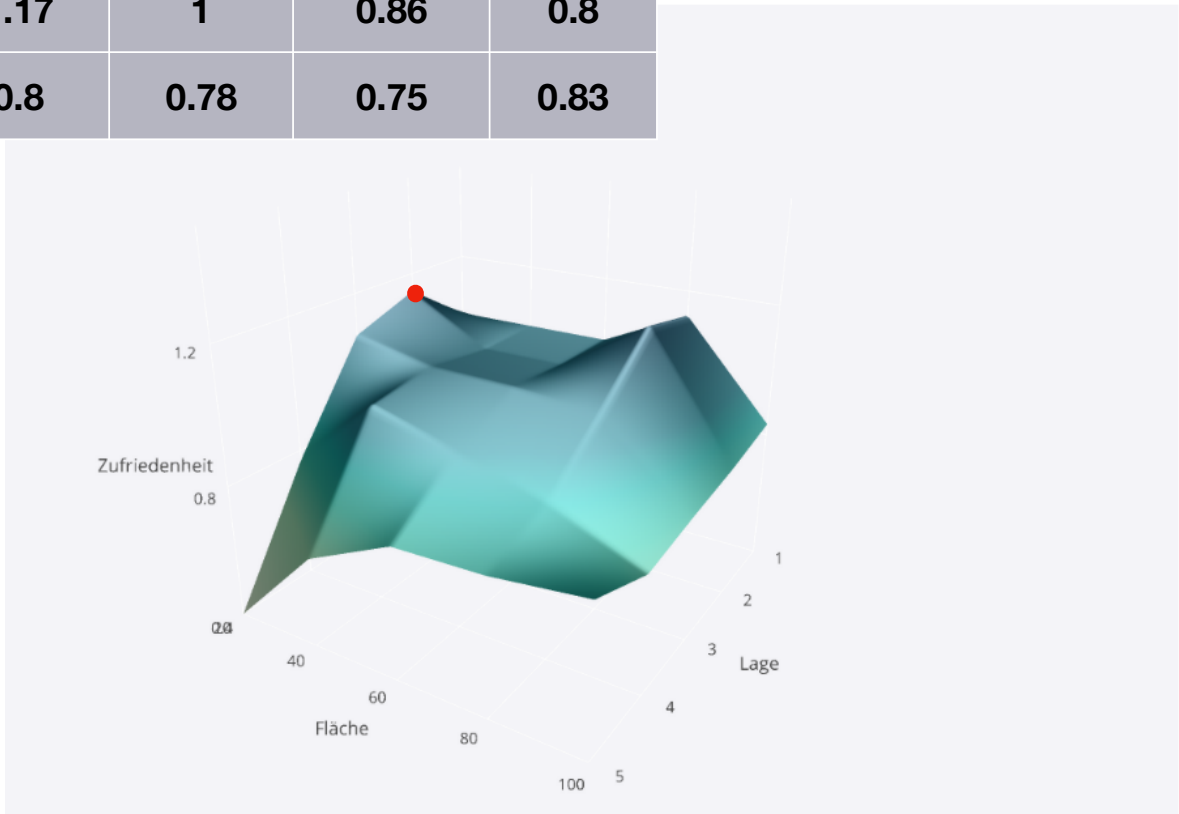
-für Metropolis sind auch moves erlaubt, durch welche die Zufriedenheit verringert wird

-Parameter T gibt dabei „Risikobereitschaft“ an in einen schlechteren Zustand zu wechseln

Move	T=1	T=10
(20, 2) —> (20, 3)	0.93	0.99
(20, 4) —> (20, 5)	0.69	0.96

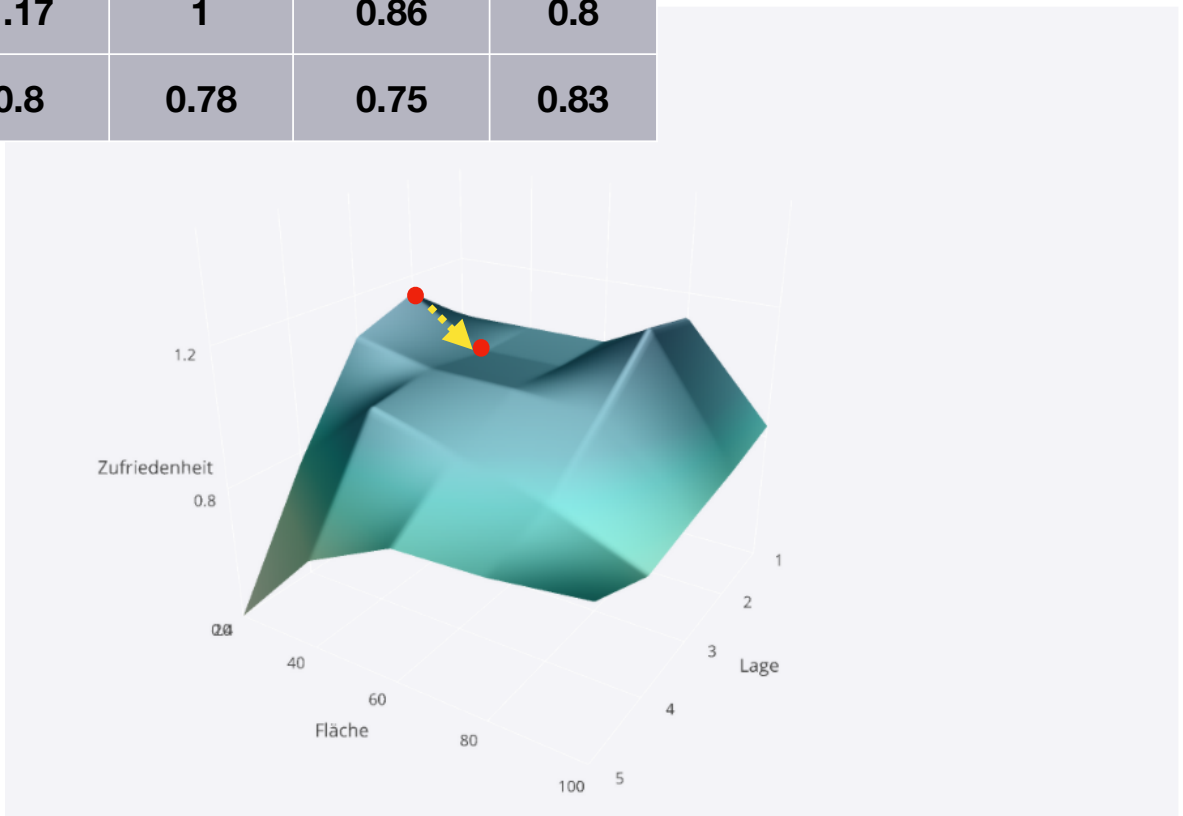
Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



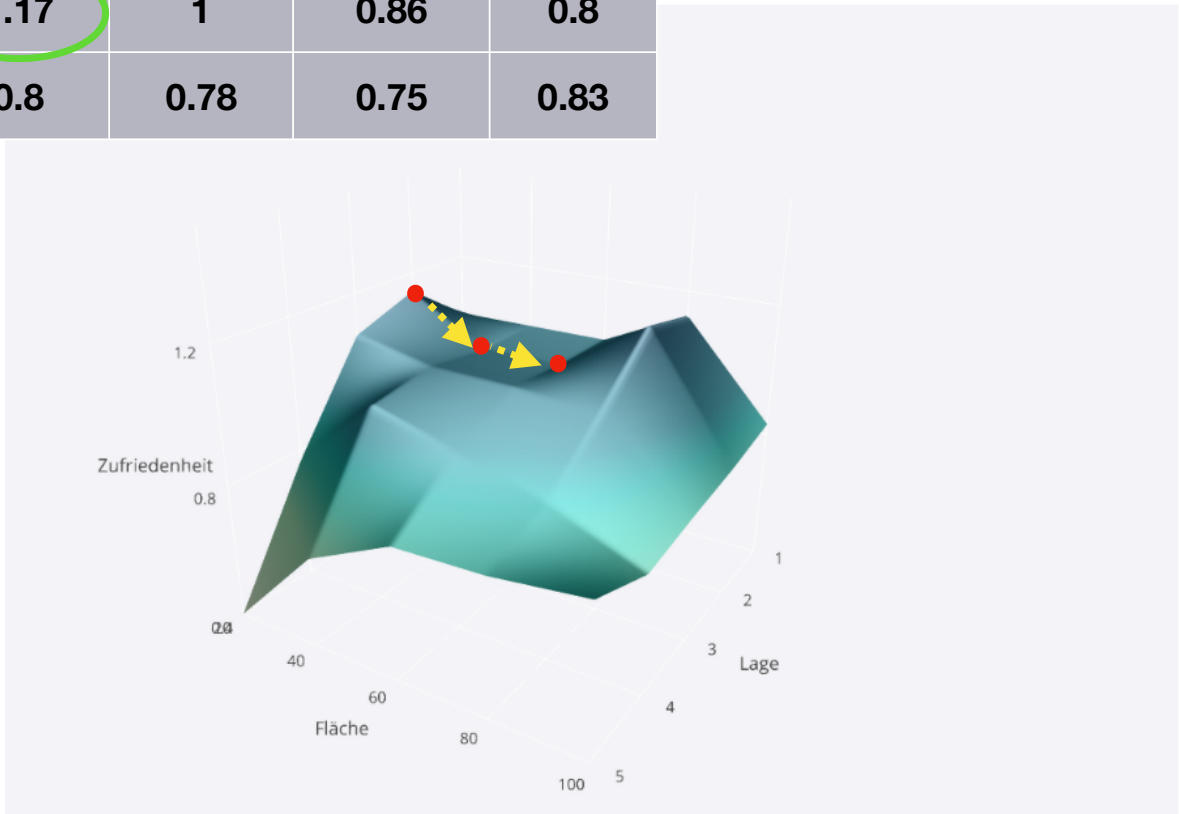
Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



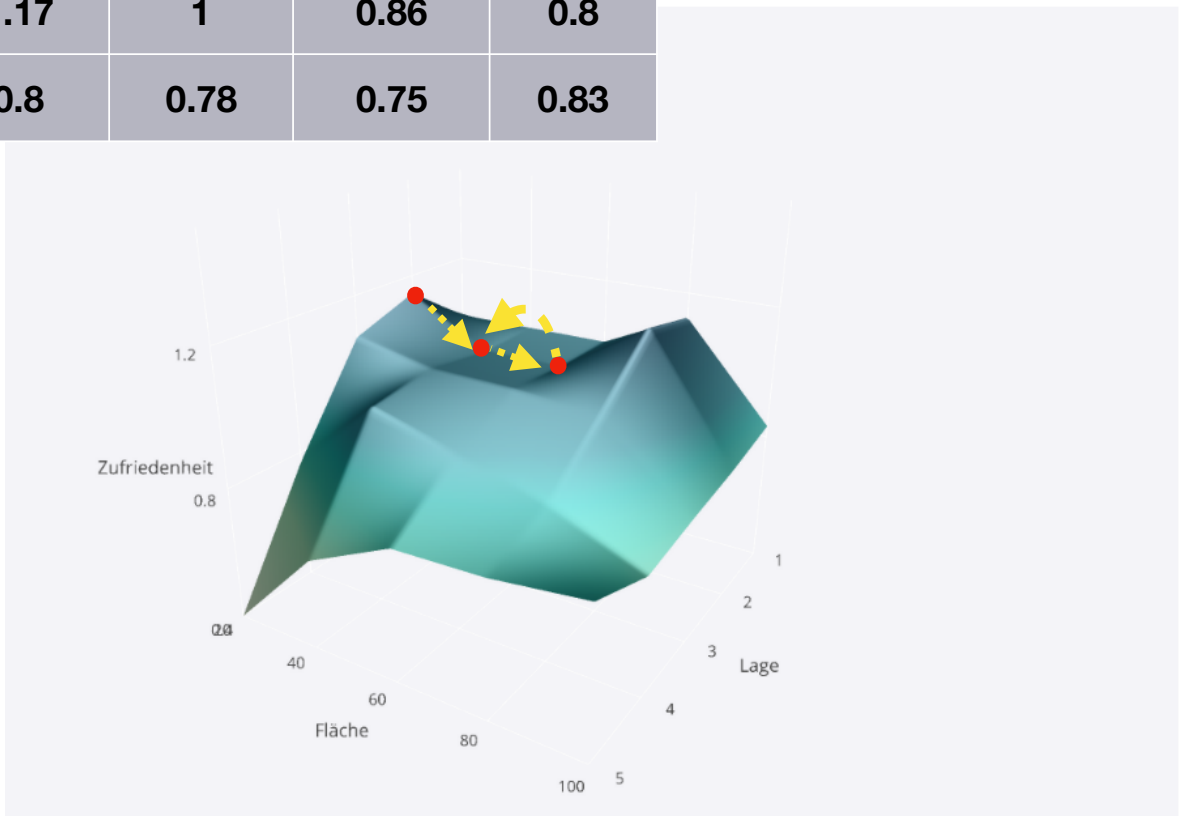
Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



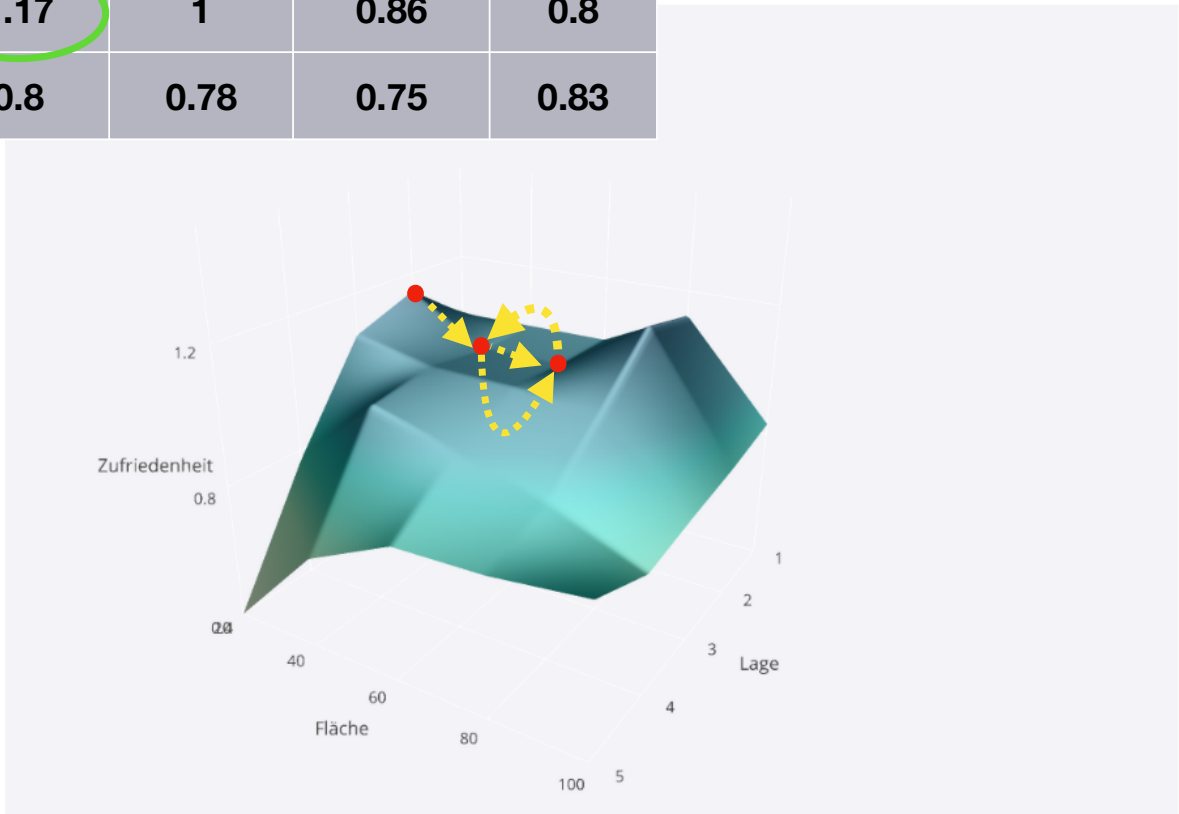
Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



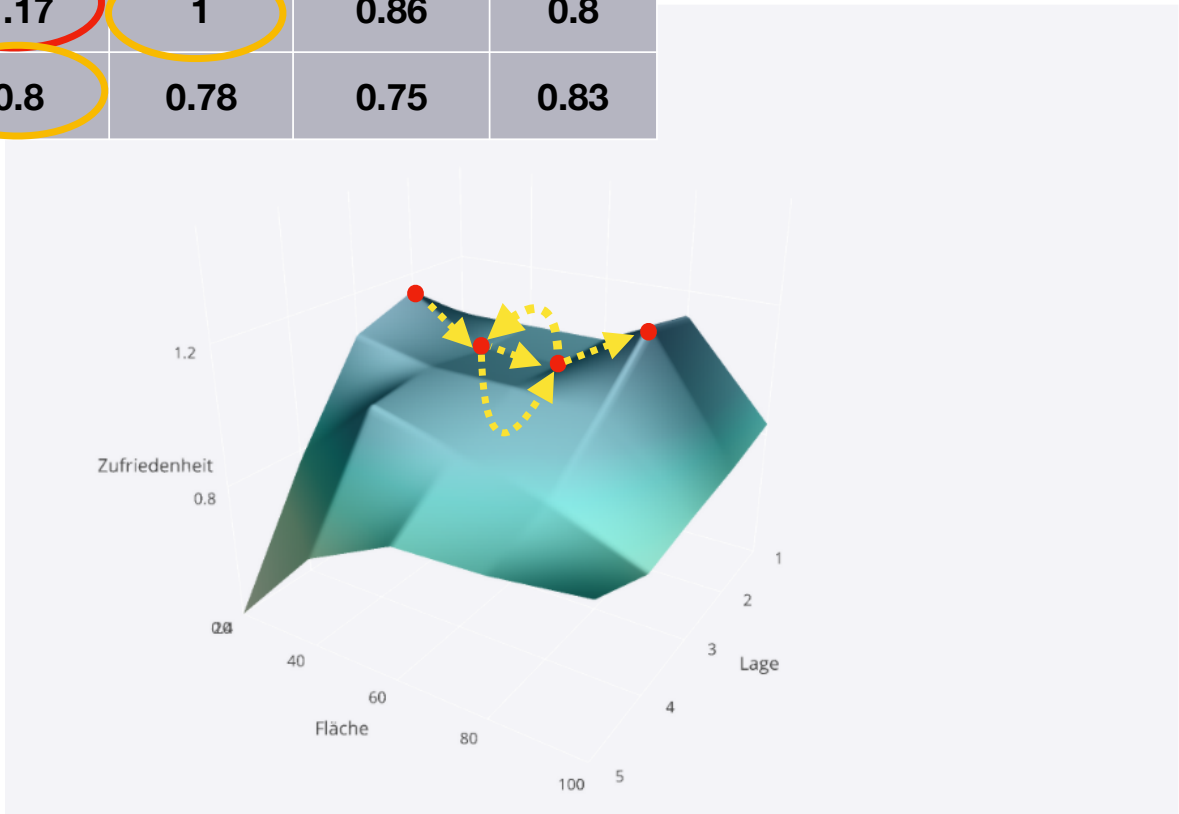
Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



Beispiel - Metropolis Walks

Zufriedenh.	1	2	3	4	5
20m²	1	1.14	1.07	0.77	0.4
40m²	1	1	1.03	1	0.67
60m²	1	1	1.02	0.93	0.79
80m²	1.13	1.17	1	0.86	0.8
100m²	0.83	0.8	0.78	0.75	0.83



Genetische Algorithmen

- inspiriert von Evolution natürlicher Lebewesen
- Ablauf:
  1. Initialisierung: Lege Elemente der Startgeneration fest (oft zufällig)
  2. Evaluation: Weise jedem Kandidat einen Wert der Fitnessfunktion zu
  3. Selektion: Wähle Individuen für die Rekombination (z.B. anhand Wert Fitnessfunktion)
  4. Rekombination (Crossover): Kombination der ausgewählten Individuen
  5. Mutation: Zufällige Veränderung der Nachfahren
  6. gehe zu 2.

Selektion:
viele Varianten:

- fitness-proportional selection.
  Ziehe aus B mit Wahrscheinlichkeiten proportional zu $-f(z)$ oder $\exp(-f(z))$ bis genau n Lösungen gezogen sind
- ziehe zwei Lösungen z und z'. Falls $z > z'$, akzeptiere z. Wiederhole bis n Lösungen akzeptiert sind.
- Wähle die besten n Lösungen.
- ...

Cross-Over

Für Strings:
1-Punkt Crossover

```
x1 = 01000010101000101010101101
x2 = 10101011101010100111100110101
```

Bestimme einen zufälligen *Bruchpunkt* k und vertausche die Suffixe:
$x_1' = x_1[1..k]x_2[k+1..n]$ und $x_2' = x_2[1..k]x_1[k+1..n]$
Für $k = 7$

```
x1' = 0100001.11010101001111100110101
x2' = 1010101.01010001010101010101101
```

Cross-Over

Uniformer Crossover

übernimm jede Position zufällig entweder unverändert oder vertausche die beiden Zeichen zwischen Eltern

```
    x1 = 01100011101010101011100100101
    x2 = 10001010101000100110101111101
tausch   --**-***--******---**---**-*
```

Übliche Forderung an Rekombination:

- $\mathcal{R}(x, y) = \mathcal{R}(y, x)$
- Für $z \in \mathcal{R}(x, y)$ gilt $d(x, z), d(y, z) \leq d(x, y)$

Crossover für Vektoren

Mittelwertbildung:

$$z = \frac{1}{2}(x + y)$$

Konvexe Kombination:

$$z = px + (1 - p)y \qquad p \in [0, 1]$$

Die Zufallszahl p wird typischerweise aus einer Gleichverteilung gezogen
Spezielle Verfahren für andere Typen von Objekten, wie z.B., Permutationen, Bäume ...

Abbruchbedingungen

- Normalerweise ist nicht bekannt, ob das Ziel bereits erreicht ist, d.h., ob ein globales Minimum bereits gefunden ist.
- Fixe Anzahl von Iterationen
- Keine Verbesserung seit K Iterationen
- Verbesserung in den letzten K Iterationen weniger als ϵ (im Fall von reell-wertigen Problemstellungen)

Beispiel - Genetische Algorithmen

- Ziel: Finde das perfekte Auto
- ein Auto habe folgende Attribute:
  - 1. Preis
  - 2. Leistung
  - 3. Verbrauch
  - 4. Reichweite
  - 5. Treibstoff
  - 6. Sicherheit (1-5 Sterne)
  - 7. Baujahr
  - 8. Sitzplätze
  - 9. Kofferraumvolumen
  - 10. Navi (0: keins; 1: Handy; 2: TomTom 3: integriert preiswert 4: integriert premium)

Beispiel - Genetische Algorithmen

Z1

PREIS IN €	Z
500	1
25000	2.5
50000	5
75000	7,5
100000	10

Z2

LEISTUNG IN PS	Z
50	0.1
100	0.2
200	0.4
300	0.6
500	1

Z3

VERBRAUCH IN L/100KM	Z
1	1
4	0.5
7	0.3
10	0.25
25	0.1

Z4

REICHWEITE IN KM	Z
200	0.1
300	0.2
400	0.3
600	0.6
1000	1

Z5

TREIBSTOFF	Z
DIESEL	0.1
BENZIN	0.2
ERDGAS	0.3
HYBRID	0.6
ELEKTRO	1

Z6

SICHERHEIT	Z
1	0.2
2	0.4
3	0.6
4	0.8
5	1

Z7

BAUJAHR	Z
1950	0.2
1980	0.9
2000	0.6
2010	0.8
2018	1

Z8

SITZPLÄTZE	Z
1	0.2
2	0.5
4	0.6
5	0.8
7	1

Z9

KOFFERRAUM VOLUMEN IN L	Z
0	0.2
200	0.5
400	0.6
500	0.8
700	1

Z10

NAVIGATION	Z
KEINS	0.2
HANDY	0.5
TOMTOM	0.6
INTEGRIERT PREISWERT	0.8
INTEGRIERT PREMIUM	1

Beispiel - Genetische Algorithmen

Bewertungsfunktion:

$$f(Auto)=\left(\frac{z_2+z_3+z_4+z_5+z_6+z_7+z_8+z_9+z_{10}}{z_1}\right)$$

Beispiel - Genetische Algorithmen

Bewertungsfunktion:

$$f(Auto)=\left(\frac{z_2+z_3+z_4+z_5+z_6+z_7+z_8+z_9+z_{10}}{z_1}\right)$$

	Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
x	500 €	50	10	300	Benzin	1	1950	4	0	keins
z(x)	10	0.1	0.25	0.2	0.2	0.2	0.2	0.6	0.2	0.2

$$f(VW\ Beetle)=\left(\frac{0.1+0.25+0.2+0.2+0.2+0.2+0.6+0.2+0.2}{1}\right)=\frac{2.15}{1}=2.15$$

Beispiel - Genetische Algorithmen

1. Initialisierung: zufällige Individuen für Startgeneration

	Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi	f(Auto)
	500 €	50	10	300	Benzin	1	1950	4	0	keins	
	2000 €	20	10	100	Benzin	1	2000	1	0	keins	
	50000 €	200	6	800	Diesel	5	2015	7	700	integriert preiswert	
	60000 €	350	20	500	Benzin	4	2012	2	100	integriert premium	

Beispiel - Genetische Algorithmen

	Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi	f(Auto)
	500 €	50	10	300	Benzin	1	1950	4	0	keins	**2.15**
	2000 €	20	10	100	Benzin	1	2000	1	0	keins	**0.85**
	50000 €	200	6	800	Diesel	5	2015	7	700	integriert preiswert	**1.24**
	60000 €	350	20	500	Benzin	4	2012	2	100	integriert premium	**0.78**

2. Evaluation : Bewerte Individuen anhand Fitnessfunktion

Beispiel - Genetische Algorithmen

	Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi	f(Auto)
	500 €	50	10	300	Benzin	1	1950	4	0	keins	**2.15**
	2000 €	20	10	100	Benzin	1	2000	1	0	keins	**0.85**
	50000 €	200	6	800	Diesel	5	2015	7	700	integriert preiswert	**1.24**
	60000 €	350	20	500	Benzin	4	2012	2	100	integriert premium	**0.78**

3. Selektion: Wähle Individuen für Rekombination anhand von Fitnessfunktion f aus

Beispiel - Genetische Algorithmen

4. Rekombination : kombiniere VW Beetle mit VW T6

Rekombination durch Mittelwertbildung

- Preis : Ziehe aus Verteilung mit Mittelwert 25250 €
- Leistung : Ziehe aus Verteilung mit Mittelwert 125 PS
- ...

	Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
	500 €	50	10	300	Benzin	1	1950	4	0	keins
	50000 €	200	6	800	Diesel	5	2015	7	700	integriert preiswert

Beispiel - Genetische Algorithmen

4. Rekombination : kombiniere VW Beetle mit VW T6

Rekombination durch Mittelwertbildung

- Preis : Ziehe aus Verteilung mit Mittelwert 25250 €
- Leistung : Ziehe aus Verteilung mit Mittelwert 125 PS
- ...

	Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
	500 €	50	10	300	Benzin	1	1950	4	0	keins
	50000 €	200	6	800	Diesel	5	2015	7	700	integriert preiswert

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
27000 €	150	8	500	Benzin	4	1980	5	400	TomTom
30000 €	100	7	600	Benzin	2	1990	6	200	Handy
20000 €	90	6	700	Diesel	3	1995	4	600	TomTom
25000 €	125	8	500	Diesel	3	1985	5	400	keins

Beispiel - Genetische Algorithmen

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
27000 €	150	8	500	Benzin	4	1980	5	400	TomTom
30000 €	100	7	600	Benzin	2	1990	6	200	Handy
20000 €	90	6	700	Diesel	3	1995	4	600	TomTom
25000 €	125	8	500	Diesel	3	1985	5	400	keins

5. Mutation: Verändere zufällig Attribute

Beispiel - Genetische Algorithmen

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
27000 €	150	8	500	Benzin	4	1980	5	400	TomTom
30000 €	100	7	600	Benzin	2	1990	6	200	Handy
20000 €	90	6	700	Diesel	3	1995	4	600	TomTom
25000 €	125	8	500	Diesel	3	1985	5	400	keins

5. Mutation: Verändere zufällig Attribute

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi
27000 €	150	8	**1000**	Benzin	4	1980	5	400	TomTom
15000 €	100	7	600	Benzin	2	1990	6	200	Handy
20000 €	90	6	700	Diesel	3	1995	**2**	600	TomTom
10000 €	125	8	500	**Elektro**	3	**2018**	5	400	**integriert**

Beispiel - Genetische Algorithmen

1. Evaluation

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi	f(Auto)
27000 €	150	8	**1000**	Benzin	4	1980	5	400	TomTom	1
15000 €	100	7	600	Benzin	2	1990	6	200	Handy	1.2
20000 €	90	6	700	Diesel	3	1995	**2**	600	TomTom	1.15
10000 €	125	8	500	**Elektro**	3	**2018**	5	400	**integriert pr.**	**2.9**

Beispiel - Genetische Algorithmen

1. Evaluation

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi	f(Auto)
27000 €	150	8	**1000**	Benzin	4	1980	5	400	TomTom	1
15000 €	100	7	600	Benzin	2	1990	6	200	Handy	1.2
20000 €	90	6	700	Diesel	3	1995	**2**	600	TomTom	1.15
10000 €	125	8	500	**Elektro**	3	**2018**	5	400	**integriert pr.**	**2.9**

anschließend Selektion, Rekombination, Mutation....

Beispiel - Genetische Algorithmen

1. Evaluation

Preis	Leistung	Verbrauch	Reichweite	Treibstoff	Sicherheit	Baujahr	Sitzplätze	Kofferraum	Navi	f(Auto)
27000 €	150	8	**1000**	Benzin	4	1980	5	400	TomTom	1
15000 €	100	7	600	Benzin	2	1990	6	200	Handy	1.2
20000 €	90	6	700	Diesel	3	1995	**2**	600	TomTom	1.15
10000 €	125	8	500	**Elektro**	3	**2018**	5	400	**integriert pr.**	**2.9**

anschließend Selektion, Rekombination, Mutation....

bestes Ergebnis: kaufen Sie einen Elektrofahrzeug mit 125 PS von 2018 für 10000€

Beispiel - Genetische Algorithmen

Beispiel - Genetische Algorithmen

Zusammenfassung

- randomisierte Algorithmen werden gerne für Probleme ohne "bekannte polynomielle" Lösung benutzt
- bei Problemen mit vielen lokalen Optima kann das Finden einer / der global optimalen Lösung nicht garantiert werden (in endlicher Zeit)
- Markov Chain Monte Carlo (MCMC) wird in der modernen Statistik sehr häufig benutzt – insbesondere für Probleme ohne analytische Lösung
- . . . derer es sehr viele gibt (analytische Lsgn sind selten!)