Schätzen der Fehlerzahl

in Software-Dokumenten

Dr. Frank Padberg

Universität Karlsruhe

Mai 2003

Motivation

- beim Testen oder bei Inspektionen werden nicht alle Fehler gefunden
- Anzahl der unentdeckten Fehler ist nie genau bekannt
- Fehlerzahl ist wichtige Größe für Qualitätssicherung und Projektmanagement

© Dr. Frank Padberg 2003

Motivation

- beim Testen oder bei Inspektionen werden nicht alle Fehler gefunden
- Anzahl der unentdeckten Fehler ist nie genau bekannt
- Fehlerzahl ist wichtige Größe für Qualitätssicherung und Projektmanagement
- schätze die Fehlerzahl in einem Software-Dokument möglichst genau!

© Dr. Frank Padberg 2003

Lösungsansatz: Testreihen für Code

- nutze Information aus den Software-Tests zum Schätzen; z.B. Anzahl der Fehler je Test
- baue stochastisches Modell für Testreihen, das durch die Fehlerzahl m parametrisiert ist
- wende statistische Techniken an, um m aus konkreten Testergebnissen zu schätzen

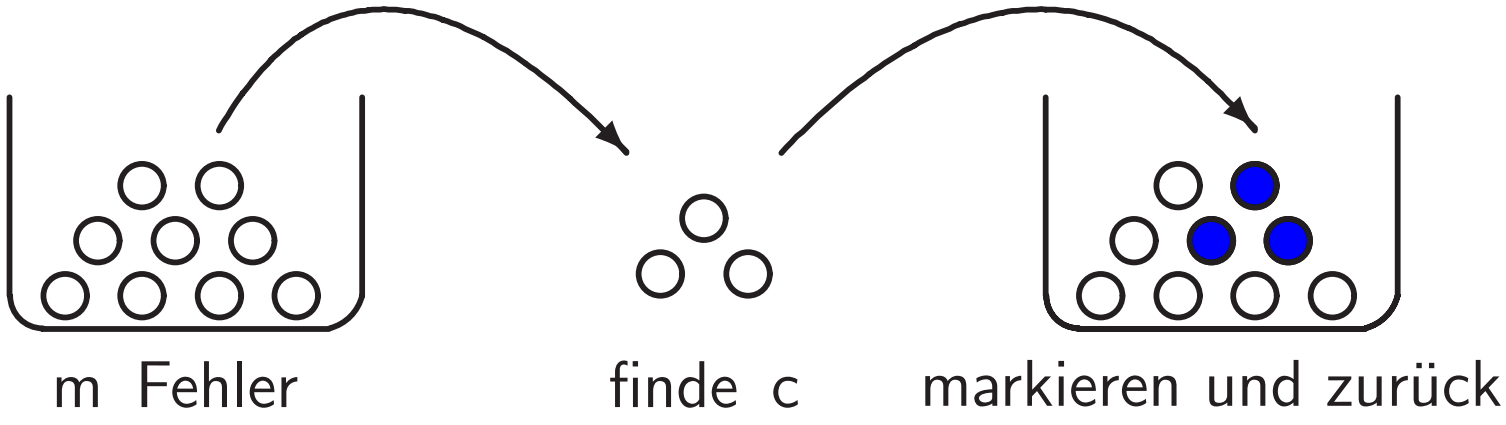
© Dr. Frank Padberg 2003

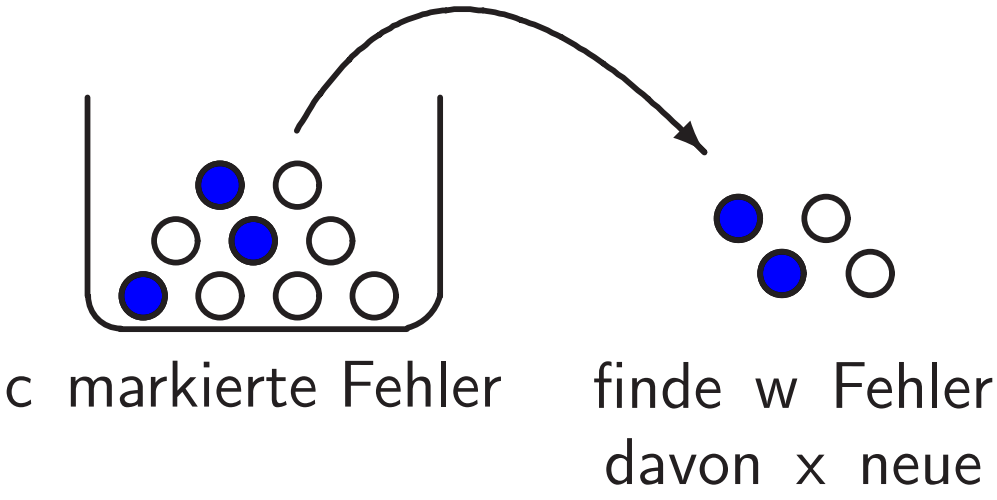
Capture-Recapture-Methoden

- kommen aus der Biologie: Schätzen der Größe einer Tierpopulation
- fange Tiere, markiere sie und lasse sie wieder frei
- fange nochmal Tiere und zähle, wieviele markiert sind
- schätze die Gesamtpopulation aus dem Verhältnis zwischen markierten und nicht markierten Tieren im zweiten Fang

© Dr. Frank Padberg 2003

Urnenmodell





© Dr. Frank Padberg 2003

Stochastisches Modell

- hypergeometrische Verteilung:

$$p_{m,w}(x, c) = \frac{\binom{m-c}{x} \cdot \binom{c}{w-x}}{\binom{m}{w}}$$

- im Beispiel: $c = 3,\ w = 4,\ x = 2$

$$p_{m,4}(2, 3) = \frac{\binom{m-3}{2} \cdot \binom{3}{4-2}}{\binom{m}{4}}$$

© Dr. Frank Padberg 2003

Likelihood-Funktion

- wie wahrscheinlich ist die Beobachtung (w, x) wenn vorher schon c Fehler gefunden wurden?
- die Antwort hängt von m ab
- definiere die Likelihood-Funktion

$$L(m) = p_{m,w}(x, c)$$

- w, x und c sind gemessen (also fest)

© Dr. Frank Padberg 2003

Max-Likelihood-Schätzung

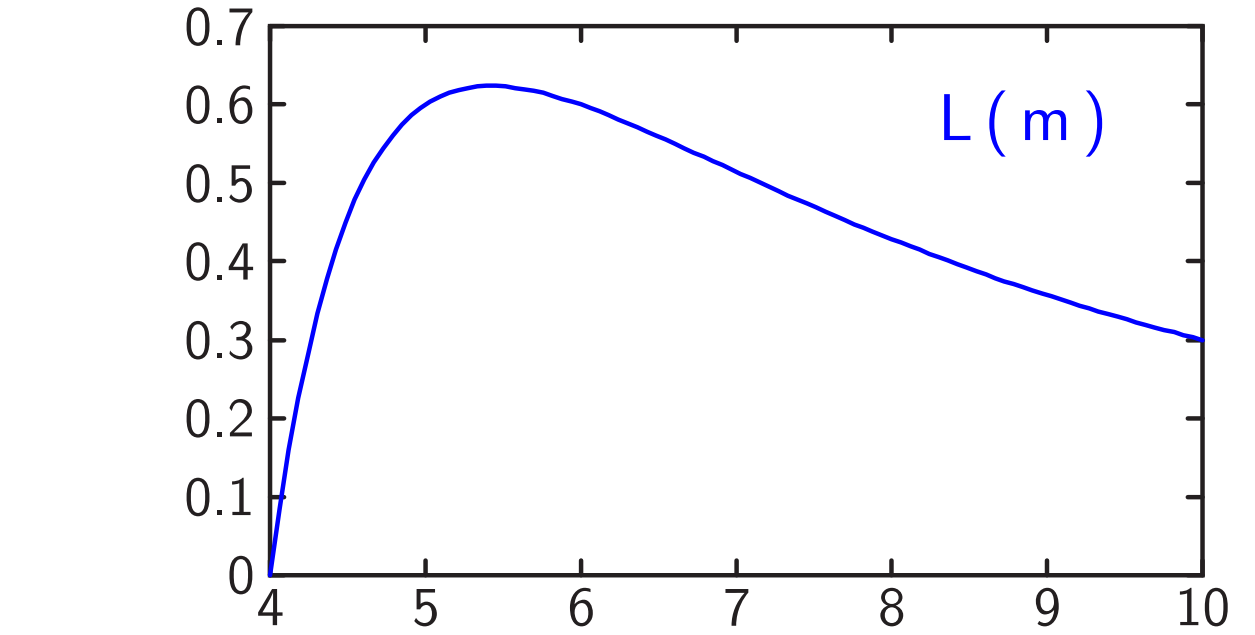
- Schätzwert ist dasjenige m, für welches die Beobachtung (w, x) am wahrscheinlichsten ist

- ein ML-Schätzwert $\widehat{m}$ maximiert also die Likelihood-Funktion: $L(\widehat{m}) \geq L(m)$

- $\widehat{m}$ muß nicht eindeutig sein

© Dr. Frank Padberg 2003

Beispiel einer Likelihood-Funktion



L (m) = 0.0 0.60 0.60 0.51 0.43 0.36 0.30

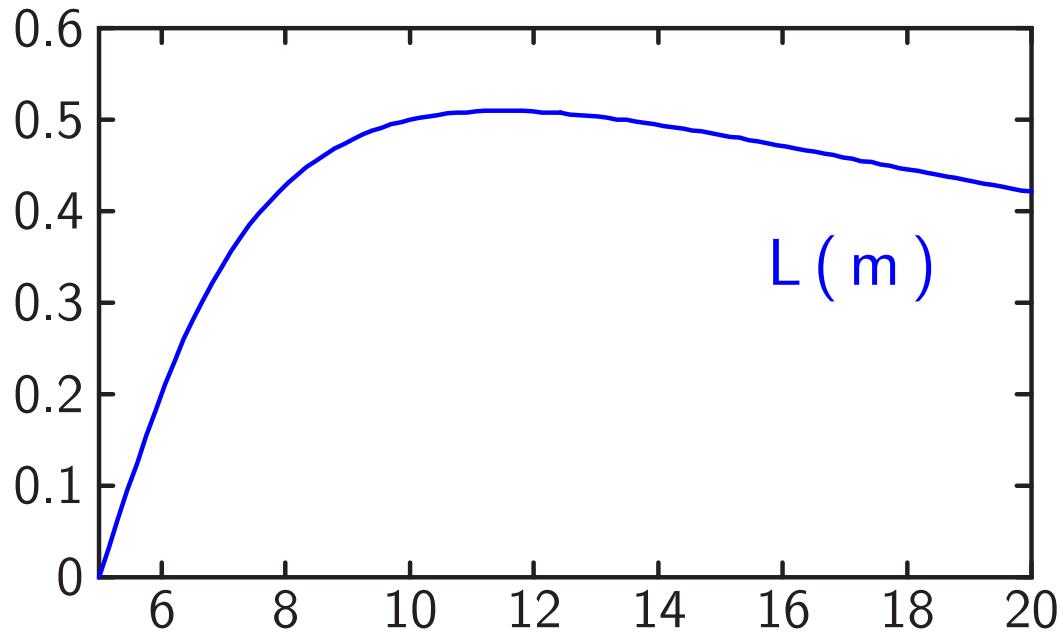
zwei markierte Fehler im zweiten Test

$(c = 3,\ w = 4,\ x = 2)$

Max-Likelihood-Schätzwert: 5 oder 6

© Dr. Frank Padberg 2003

Anderes Beispiel



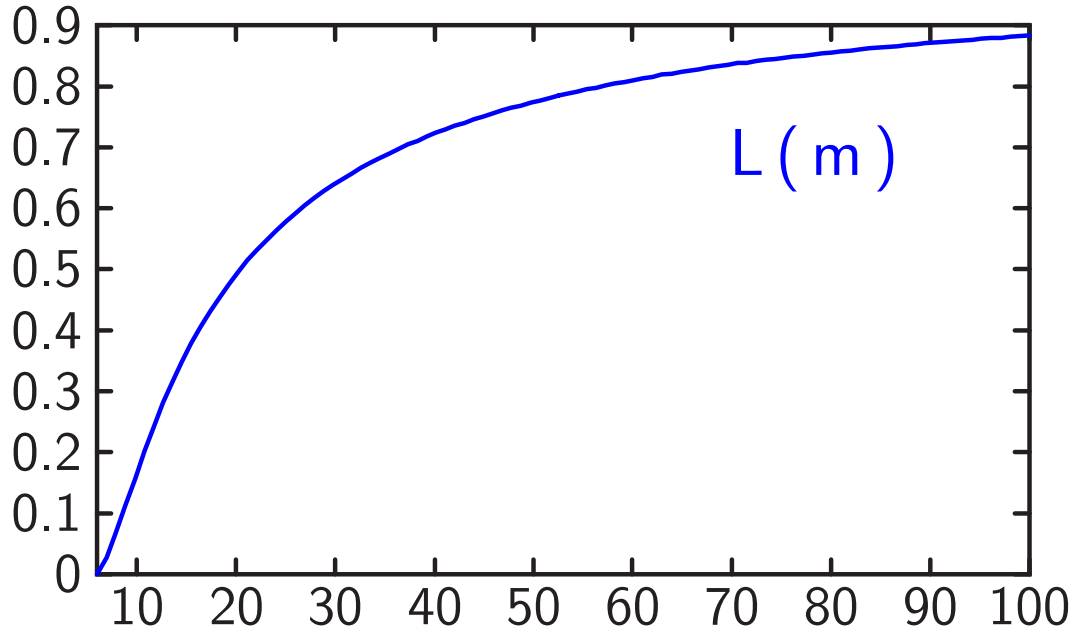
nur ein markierter Fehler im zweiten Test

$$(\, c = 3,\ w = 4,\ x = 3\,)$$

Max-Likelihood-Schätzwert: 11 oder 12

© Dr. Frank Padberg 2003

Noch ein Beispiel



kein markierter Fehler im zweiten Test

$$(c = 3,\ w = 4,\ x = 4)$$

Max-Likelihood-Schätzwert: “unendlich“

© Dr. Frank Padberg 2003

Verallgemeinerung auf Testreihen

Y. Tohma, R. Jacoby, Y. Murata, 1989 – 1995
M. Yamamoto, ...

R.-H. Hou, S.-Y. Kuo, Y.-P. Chang, 1994 – 1997
I.-Y. Chen

Hypergeometric Reliability Growth Model

© Dr. Frank Padberg 2003

Notation

w_k : Anzahl der gefundenen Fehler in Test k

x_k : Anzahl der neu entdeckten Fehler in Test k

c_k : Summe der in den ersten k Tests gefundenen verschiedenen Fehler

$$c_k = x_1 + x_2 + \ldots x_k$$

© Dr. Frank Padberg 2003

Allgemeines Hypergeometrisches Modell

- einzelner Test k

$$p_{m,\,w_k}\left(x_k\,,\,c_{k-1}\right) \;=\; \frac{\binom{m\;-\;c_{k-1}}{x_k}\;\cdot\;\binom{c_{k-1}}{w\;-\;x_k}}{\binom{m}{w_k}}$$

- Likelihood-Funktion für Serie von n Tests

$$L\left(m\right) \;=\; p_{m,\,w_1}\left(x_1\,,\,0\right)\;\cdot\;p_{m,\,w_2}\left(x_2\,,\,c_1\right)\;\cdot$$

$$p_{m,\,w_3}\left(x_3\,,\,c_2\right)\;\cdot\;\;\ldots\;\;\cdot\;p_{m,\,w_n}\left(x_n\,,\,c_{n-1}\right)$$

© Dr. Frank Padberg 2003

Ableiten der Likelihood–Funktion?!

$$p_{m,w}\,(\,x\,,\,c\,) \quad =$$

$$\frac{(\,m-c\,)!\;c!\;w!\;(\,m-w\,)!}{x!\;(\,m-c-x\,)!\;(\,w-x\,)!\;(\,c-w+x\,)!\;m!}$$

$$\frac{\mathrm{d}\;p_{m,w}\,(\,x\,,\,c\,)}{\mathrm{d}\,m} \quad = \quad \ldots \text{ viel zu kompliziert!}$$

© Dr. Frank Padberg 2003

Wachstums-Quotient

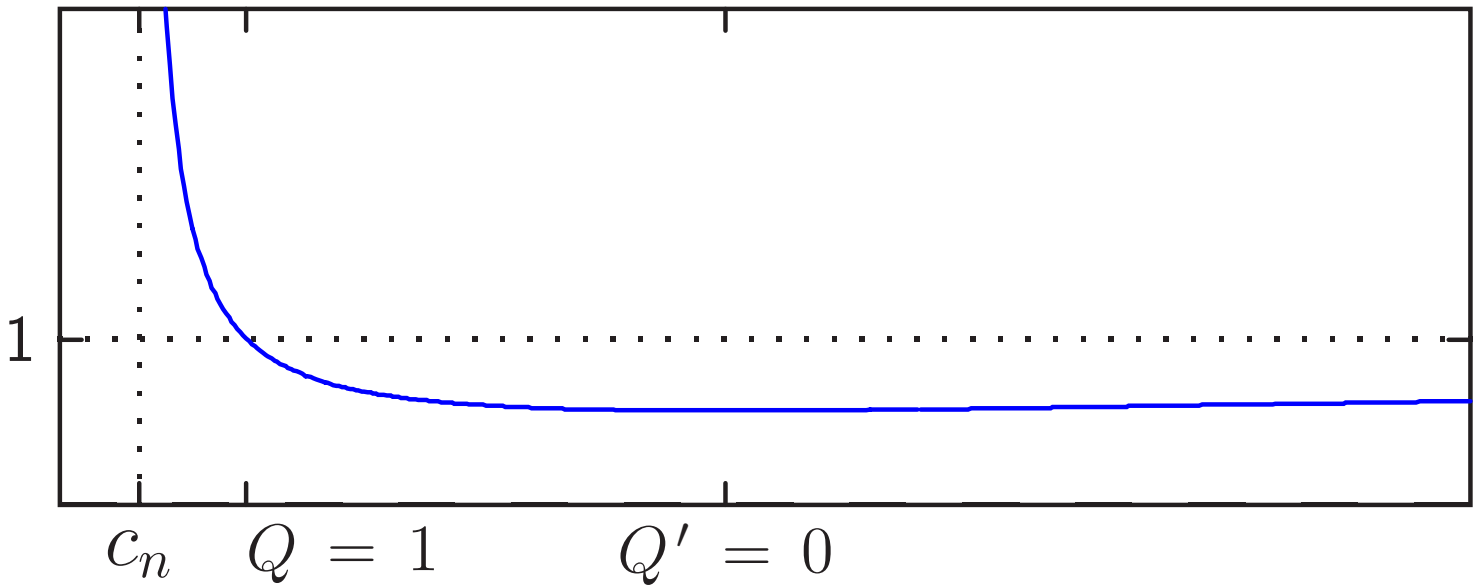
$$Q(m) = \frac{L(m)}{L(m-1)}$$

$$= \frac{(m-w_1) \cdot (m-w_2) \cdot \ \ldots \ \cdot (m-w_n)}{m^{n-1} \cdot (m-c_n)}$$

- $L(m)$ wächst genau dann wenn $Q(m) > 1$ ist
- $Q(m)$ hängt nicht von den einzelnen x_k ab, und nicht von der Reihenfolge der Tests

© Dr. Frank Padberg 2003

Typischer Graph des Quotienten



der Graph schneidet die Gerade $y = 1$

genau einmal für $x > c_n$ (Padberg 2001)

© Dr. Frank Padberg 2003

Algorithmus

- Q schneidet $y = 1$ einmal für $x > c_n$
- Schätzwert ist die größte ganze Zahl links vom Schnittpunkt
- probiere $Q(c_n + 1)$, $Q(c_n + 2)$, $\ldots$
- einfach, schnell, genau, numerisch stabil

© Dr. Frank Padberg 2003

Ein größeres Beispiel: T_{19}

k	x_k	w_k
1	15	15
2	29	29
3	22	22
4	37	37
5	2	21
6	5	43
7	36	56

k	x_k	w_k
8	29	48
9	4	15
10	27	30
11	27	46
12	22	24
13	21	57
14	22	69

k	x_k	w_k
15	6	72
16	7	45
17	9	36
18	5	41
19	3	64

Testreihe der Länge $n = 19$

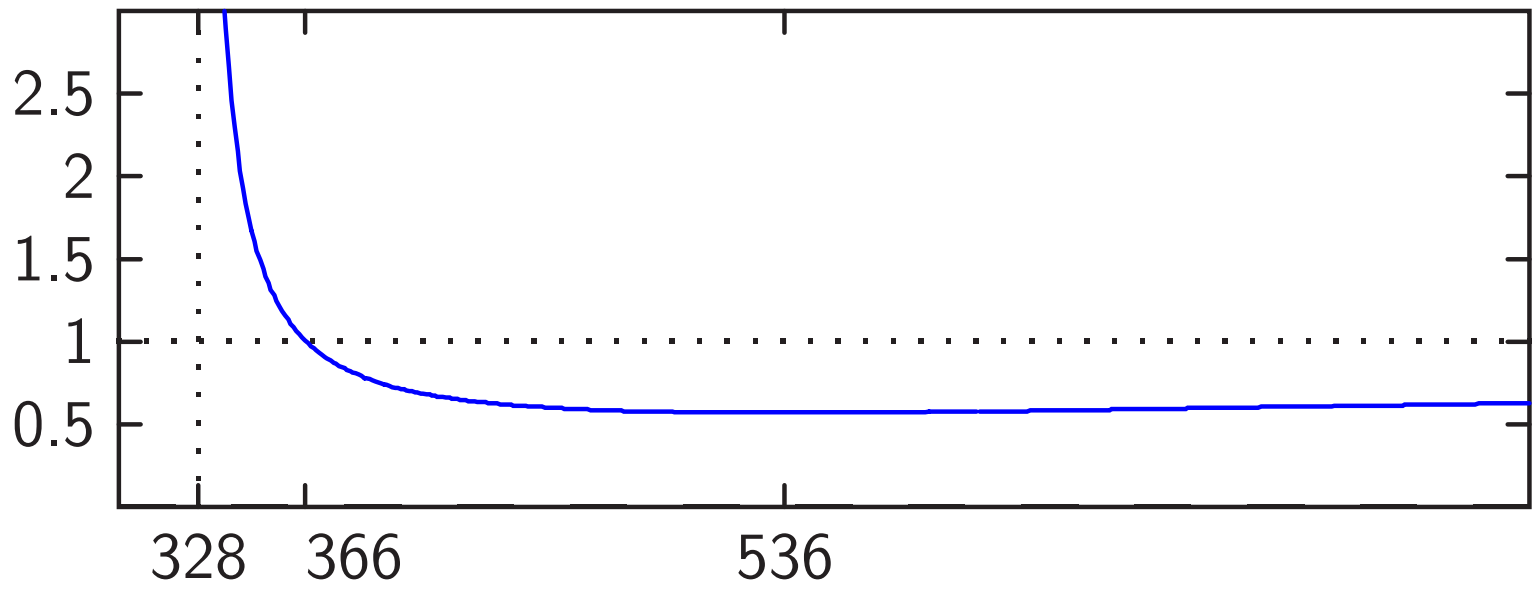
© Dr. Frank Padberg 2003

Schätzung für T_{19}

- Testreihe der Länge $n = 19$
- beim Testen entdeckte Fehler: $c_{19} = 328$
- erst später entdeckte Fehler: 30
- Gesamtzahl enthaltener Fehler: 358
- hypergeometrischer Schätzwert: $\widehat{m} = 366$

© Dr. Frank Padberg 2003

Graph des Quotienten für T_{19}



Schätzwert 366, wahrer Wert 358

© Dr. Frank Padberg 2003

Vergleich mit anderen Schätzmodellen

Modell	Schätzwert	Fehler (%)
exponential MLE	455	+27.1
delayed S-shaped MLE	351	−2.0
inflection S-shaped MLE	347	−3.1
hypergeometric MLE	366	+2.2
hypergeometric LSE	388	+8.4

Max-Likelihood-Schätzer für das hypergeometrische Modell schneidet gut ab

© Dr. Frank Padberg 2003

Eigene Veröffentlichungen zum Hypergeometrischen Modell

- *A Fast Algorithm to Compute Maximum Likelihood Estimates for the Hypergeometric Software Reliability Model*
  Asia-Pacific Conference on Quality Software APAQS (2001) 40–49, IEEE Computer Society Press

- *Maximum Likelihood Estimates for the Hypergeometric Software Reliability Model*
  International Journal of Reliability, Quality and Safety Engineering IJRQSE (2003) erscheint in Kürze, World Scientific Publishing

© Dr. Frank Padberg 2003

Software–Inspektionen [1]

- mehrere Inspektoren (2 bis 8)
- untersuchen unabhängig voneinander dasselbe Software–Dokument
- gefundene Defekte (Fehler, Schwachstellen, Unklarheiten) werden aufgeschrieben
- gemeinsames Durchsprechen der Defekte
- Probleme identifizieren, nicht lösen

© Dr. Frank Padberg 2003

Software–Inspektionen [2]

- anwendbar auf alle Software–Dokumente:
  Pflichtenhefte, Spezifikationen,
  Entwürfe, Code, Testfälle, ...
- jederzeit und frühzeitig durchführbar
- sehr effektiv in der industriellen Praxis
- aufwendig von Hand, also teuer
- Schätzen der Fehlerzahl?

© Dr. Frank Padberg 2003

Lösungsansatz: Inspektion als "Testreihe"

- jeder Inspektor entspricht einem Test
- w_k ist die Anzahl der Defekte, die Inspektor k gefunden hat
- c_n ist die Anzahl der verschiedenen Defekte, die das Inspektionsteam insgesamt gefunden hat
- kann alles aus der 0/1–Matrix abgelesen werden

© Dr. Frank Padberg 2003

0/1–Matrix

zeigt, welcher Inspektor welchen Defekt gefunden hat

	Inspektor				
Defekt	A	B	C	D	Meet
5:40	1				
6:29			1	1	
9:32		1			
4:17					1

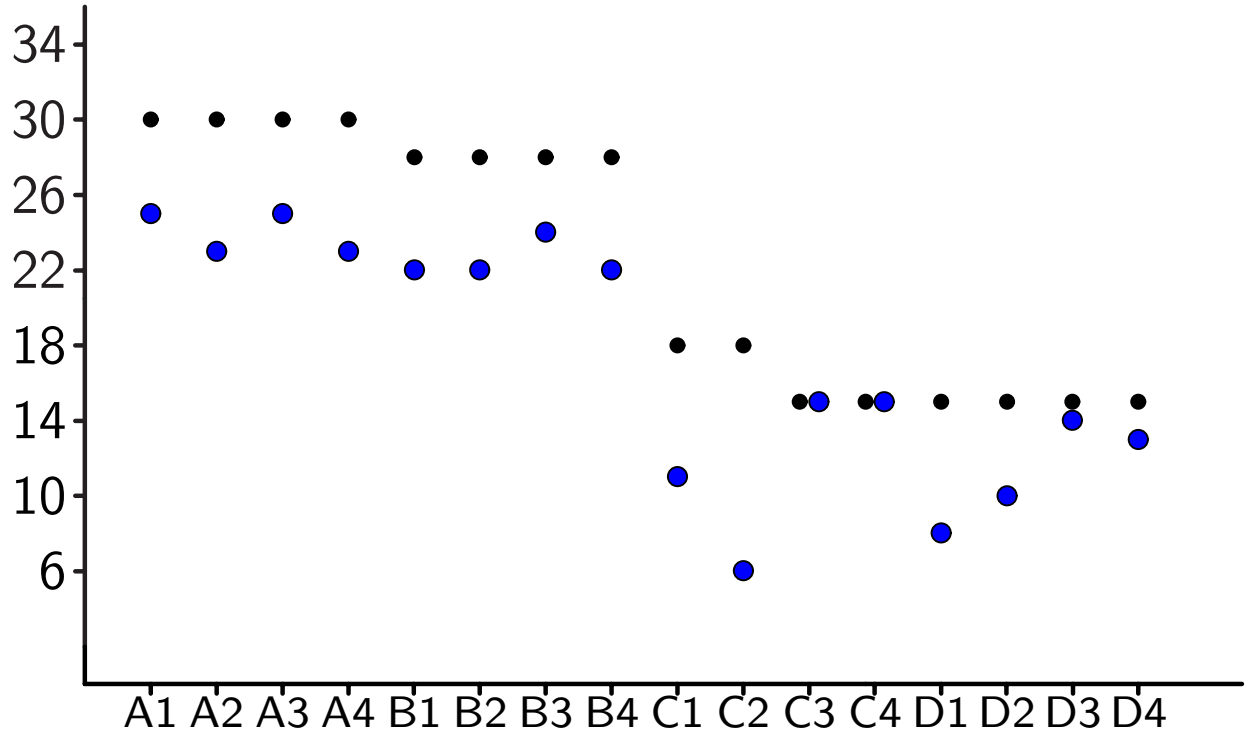
© Dr. Frank Padberg 2003

Empirischer Datensatz

- 16 Inspektionen aus kontrollierten Experimenten bei NASA SEL (Basili e.a. 1994/95)
- jeweils 6 bis 8 Inspektoren
- Spezifikationen als untersuchte Dokumente
- wahre Zahl der enthaltenen Fehler genau bekannt
- Datensatz ist “Standard-Benchmark“

© Dr. Frank Padberg 2003

Schätzung mit Hypergeometrischem Modell

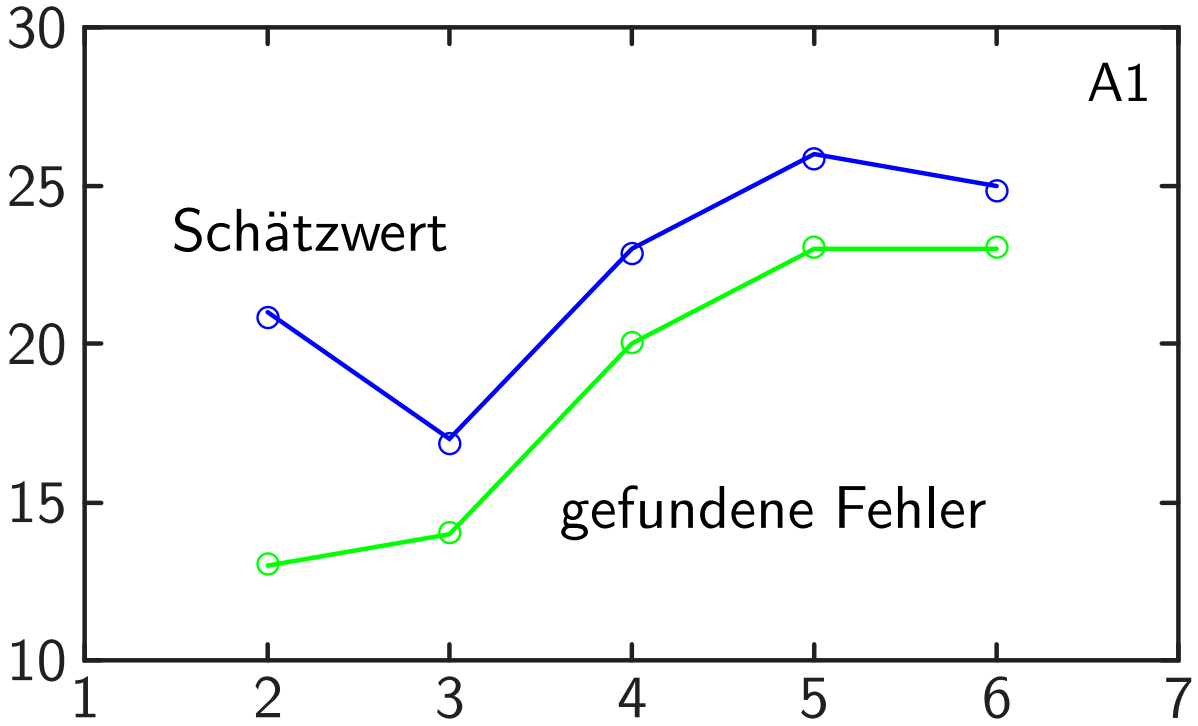


durchschnittlicher Fehler von 24 Prozent;

maximaler Fehler von -67 Prozent

© Dr. Frank Padberg 2003

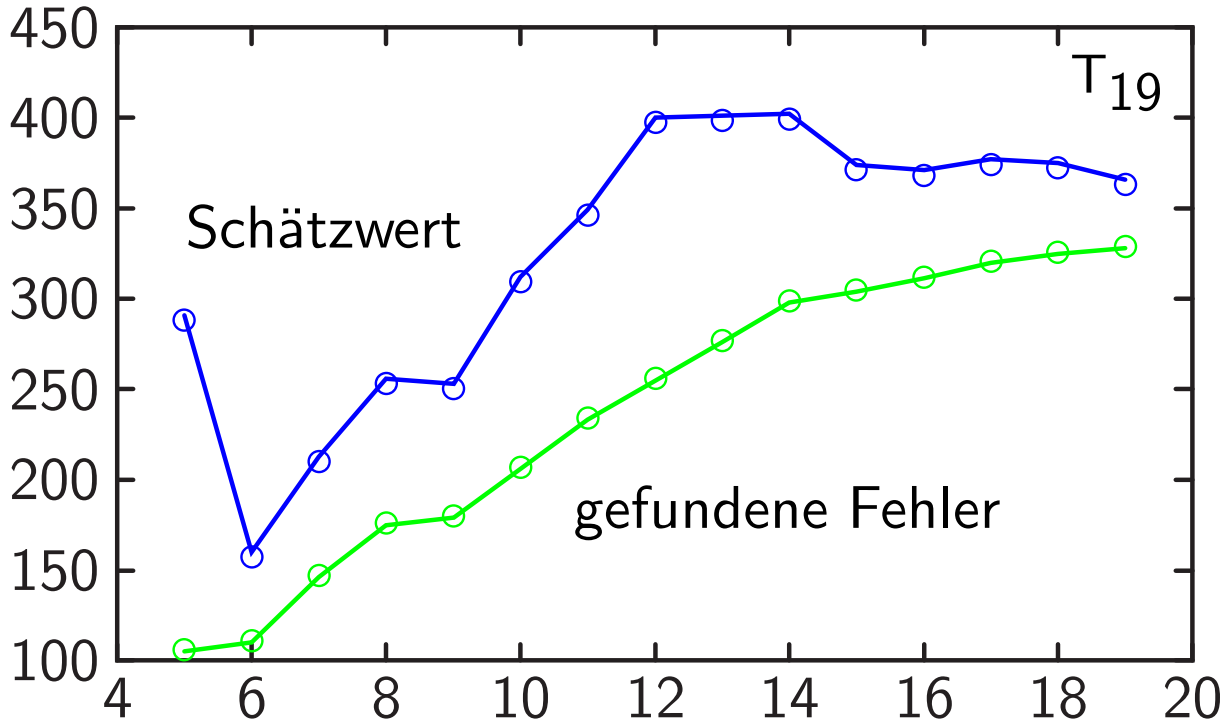
Schätzwert versus Anzahl der Inspektoren



Werte schwanken mit Anzahl der Inspektoren;
Schätzwert deutlich zu niedrig (25 statt 30)

© Dr. Frank Padberg 2003

Schätzwert versus Länge der Testreihe



Wert “stabilisiert“ sich bei längerer Testreihe;
deutliche Schwankung bei den ersten paar Tests

© Dr. Frank Padberg 2003

Warum Capture–Recapture fehlschlägt

- Mathematik: “Testreihe“ ist zu kurz
- nur das Ergebnis dieser Inspektion geht in die Schätzung ein
- mit anderen Worten: kein Lernen aus der Erfahrung

© Dr. Frank Padberg 2003

Neuer Ansatz

- nutze Wissen aus früheren Inspektionen zum Schätzen der Fehlerzahl im neuen Dokument

© Dr. Frank Padberg 2003

Neuer Ansatz

- nutze Wissen aus früheren Inspektionen zum Schätzen der Fehlerzahl im neuen Dokument
- "lerne" Zusammenhang zwischen Merkmalen einer Inspektion und der Zahl der tatsächlich im Dokument enthaltenen Fehler

© Dr. Frank Padberg 2003

Neuer Ansatz

- nutze Wissen aus früheren Inspektionen zum Schätzen der Fehlerzahl im neuen Dokument
- "lerne" Zusammenhang zwischen Merkmalen einer Inspektion und der Zahl der tatsächlich im Dokument enthaltenen Fehler
- Schätzen der Fehlerzahl als Regressionsaufgabe
- Padberg e.a. 2002

© Dr. Frank Padberg 2003

Kandidaten für Merkmale

- abgeleitet aus der 0/1–Matrix
- TDD, AVE, MIN, MAX, STD
- Beispiel Inspektion A1:

  (9, 7, 6, 13, 9, 6) und 23 ergibt

TDD	AVE	MIN	MAX	STD
23	8.3	6	13	2.4

© Dr. Frank Padberg 2003

Eingabedaten für Lineare Regression

- Korrelationsanalyse ergibt Rangfolge

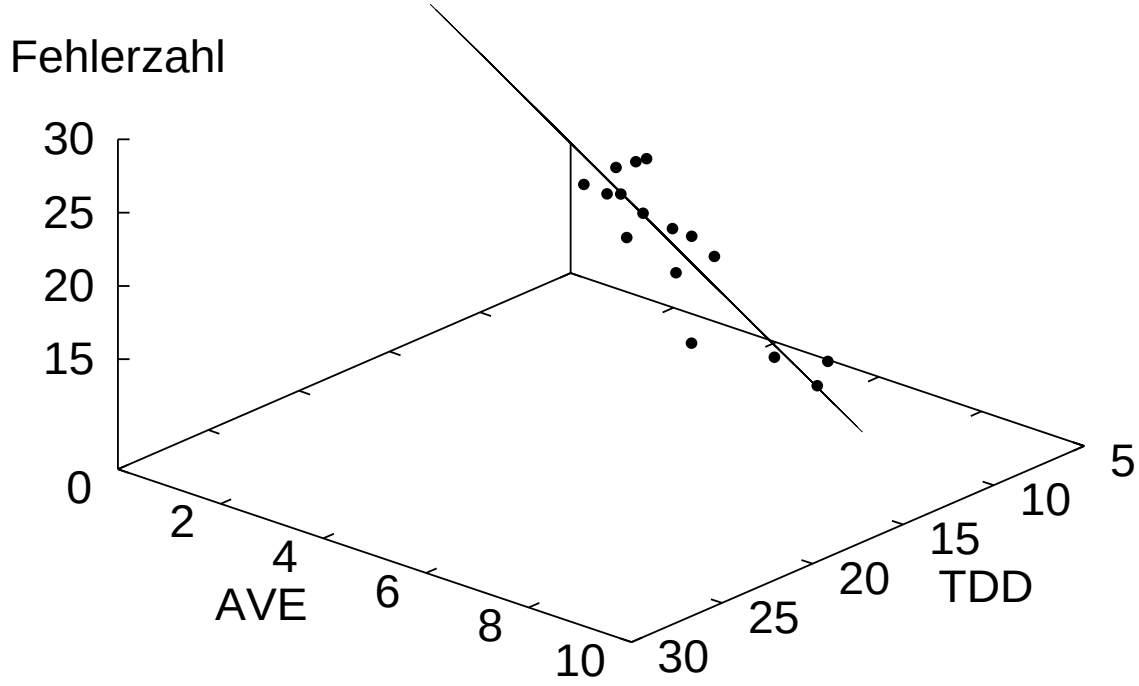
$$\text{TDD} > \text{AVE} > \text{MIN} > \text{MAX} > \text{STD}$$

- einige Stützpunkte:

Inspektion	TDD	AVE	Zielwert
A1	23	8.3	30
B1	20	6.0	28
C1	10	3.2	18
D1	6	1.3	15

© Dr. Frank Padberg 2003

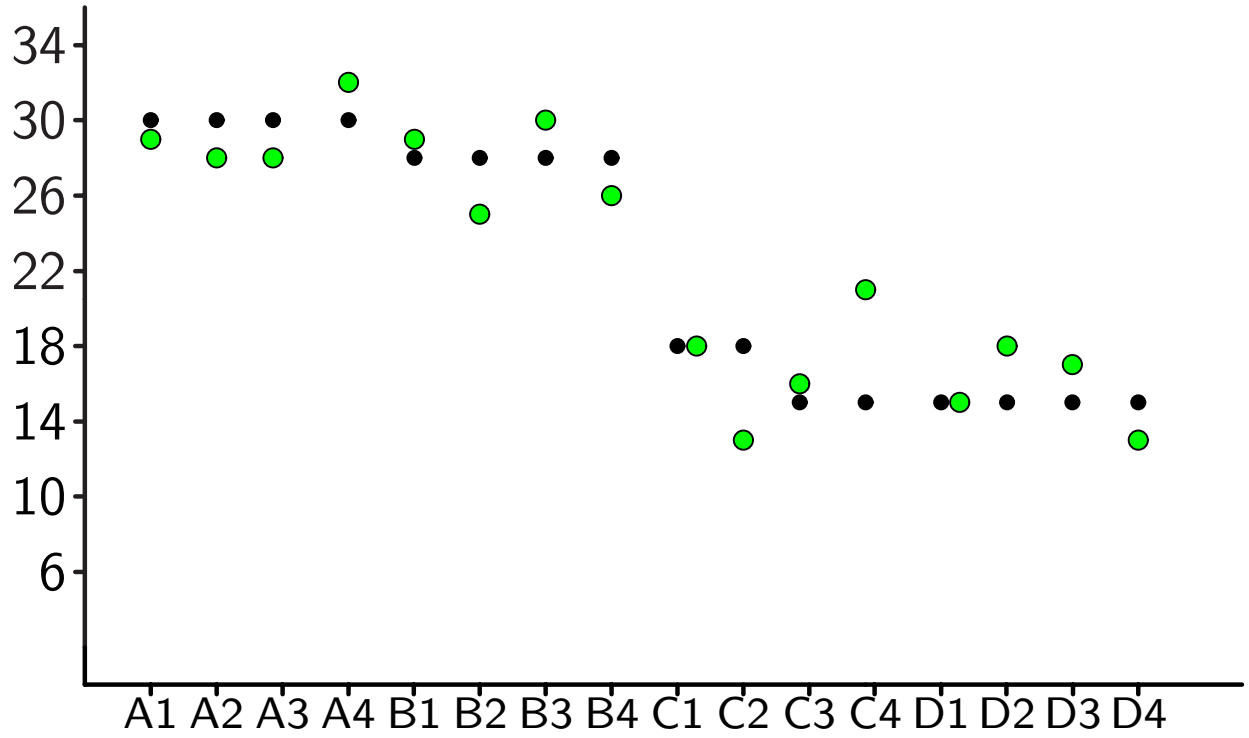
Regressionsebene



einige Datenpunkte weichen deutlich von der Ebene ab

© Dr. Frank Padberg 2003

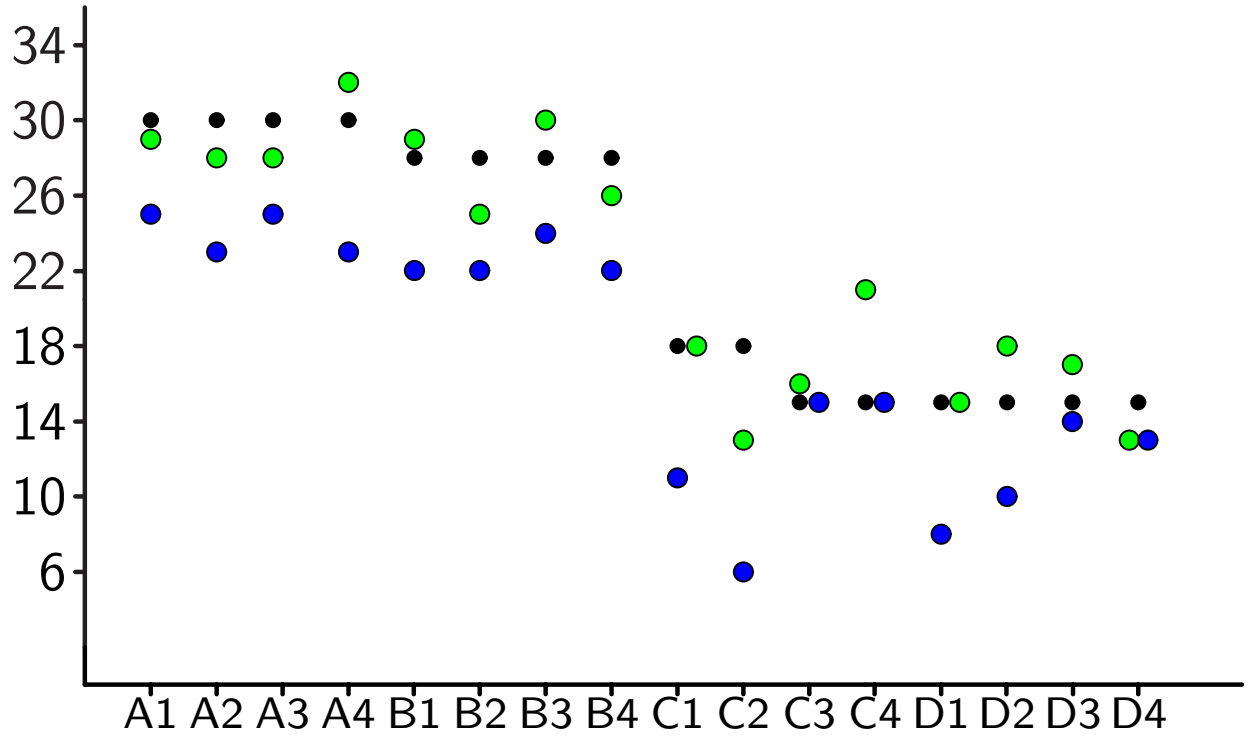
Schätzwerte mit Linearer Regression



Jackknife-Validierung ("leave one out")
durchschnittlicher Fehler von 11 Prozent;
maximaler Fehler von 40 Prozent

© Dr. Frank Padberg 2003

Schätzwerte mit Linearer Regression

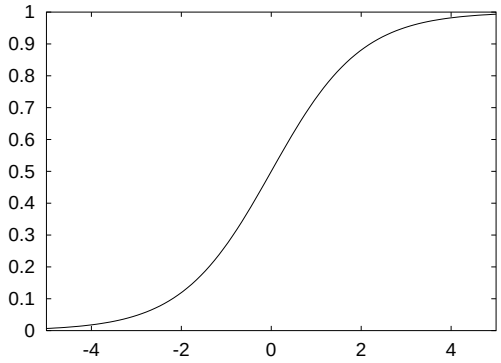


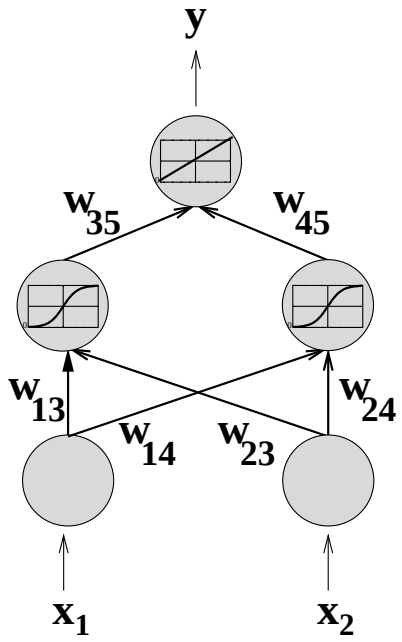
Jackknife-Validierung ("leave one out")
durchschnittlicher Fehler von 11 Prozent;
maximaler Fehler von 40 Prozent

© Dr. Frank Padberg 2003

Nicht-Lineare Regression: Neuronale Netze

$$\text{logist}\,(\,x\,) \;=\; \frac{1}{1\;+\;e^{-x}} \qquad\qquad s_i \;=\; \text{logist}\Big(\sum_j w_{ji}\cdot s_j\Big)$$





© Dr. Frank Padberg 2003

Eingabedaten für Nicht-Lineare Regression

- Informationsgehalt ergibt Rangfolge

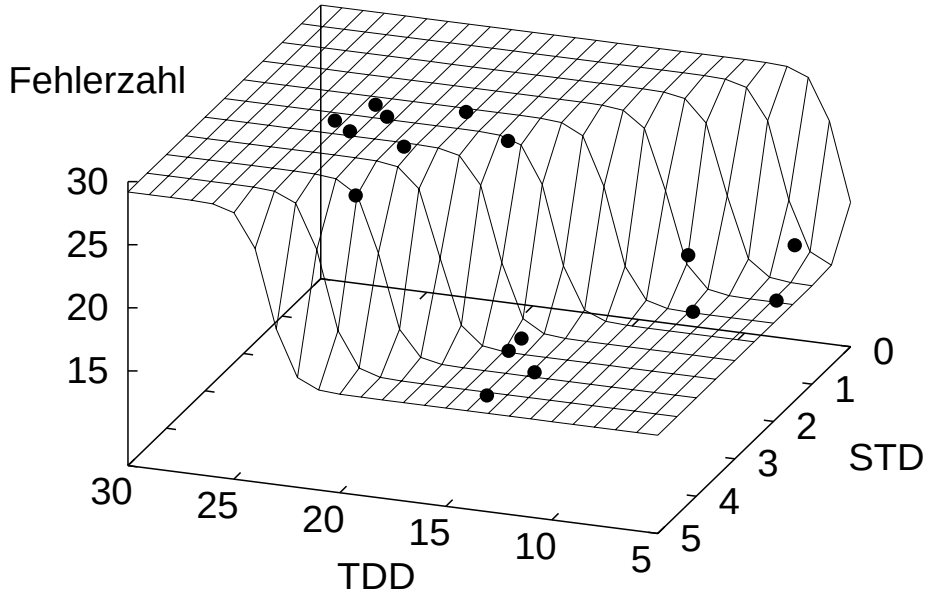
  TDD > STD > MAX > MIN > AVE

- einige Trainingspunkte:

Inspektion	TDD	STD	Zielwert
A1	23	2.4	30
B1	20	1.7	28
C1	10	1.5	18
D1	6	1.4	15

© Dr. Frank Padberg 2003

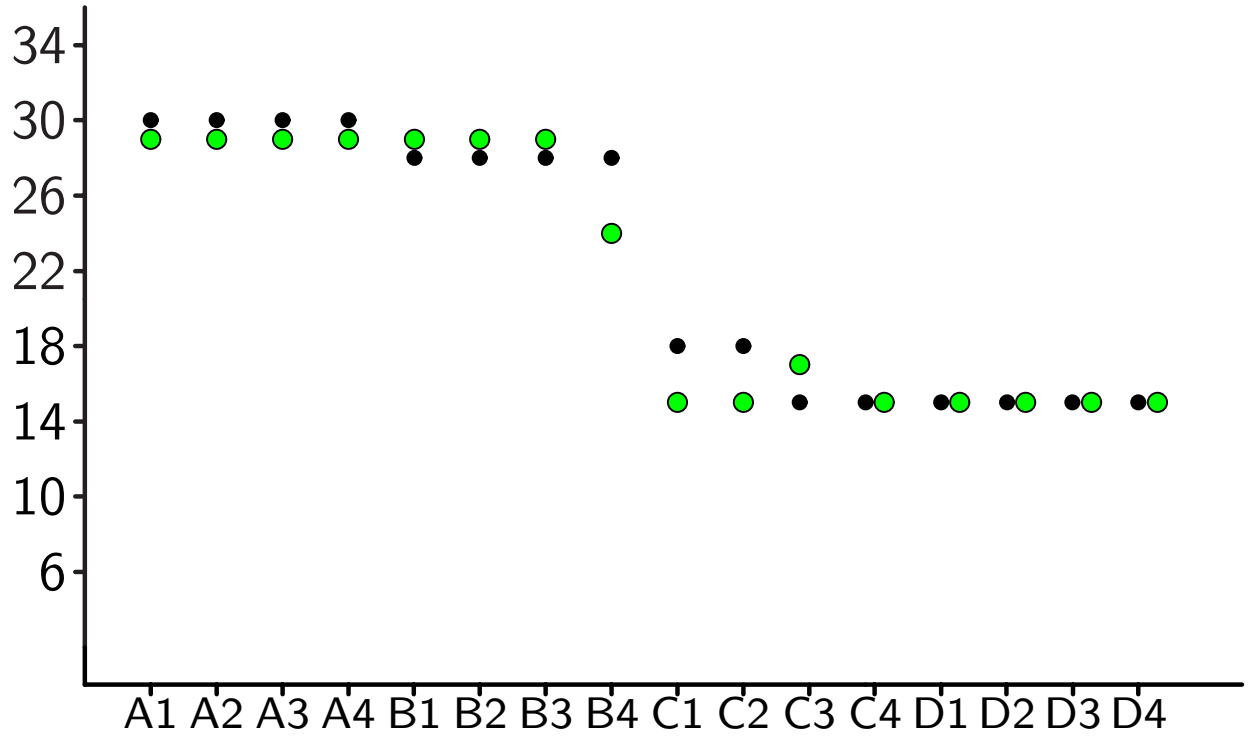
Nicht-Lineare Regressionsfläche



Netz mit zwei versteckten Neuronen;
Fläche paßt sich den Datenpunkten gut an

© Dr. Frank Padberg 2003

Schätzwerte mit Neuronalen Netzen



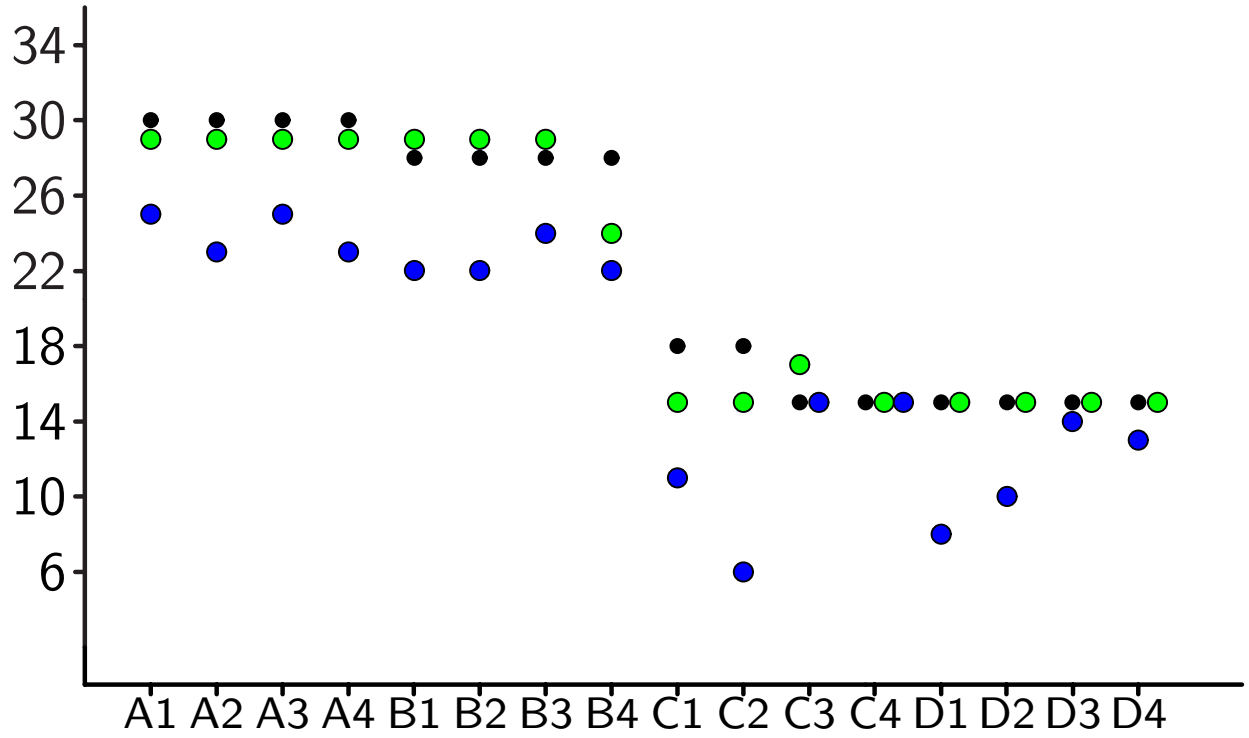
Jackknife-Validierung ("leave one out")

durchschnittlicher Fehler von nur 6 Prozent;

maximaler Fehler von -17 Prozent

© Dr. Frank Padberg 2003

Schätzwerte mit Neuronalen Netzen



Jackknife-Validierung ("leave one out")

durchschnittlicher Fehler von nur 6 Prozent;

maximaler Fehler von -17 Prozent

© Dr. Frank Padberg 2003

Vergleich mit anderen Schätzverfahren

Methode	mittlerer abs. Fehler	maximaler Fehler
Capture–Recapture	24 %	- 67 %
Detection Profile	36 %	+ 113 %
Lineare Regression	11 %	+ 40 %
Interval Estimates	7 %	+ 14 %
Neuronale Netze	6 %	- 17 %

der Regressionsansatz ist vielversprechend;

weitere Validierung mit empirischen Daten ist nötig

© Dr. Frank Padberg 2003

Eigene Veröffentlichungen zum Fehlerschätzen nach Inspektionen

- *Empirical Interval Estimates for the Defect Content After an Inspection*
  International Conference on Software Engineering ICSE (2002) 58–68, IEEE Computer Society Press

- *Applying Machine Learning to Solve an Estimation Problem in Software Inspections*
  International Conference on Artificial Neural Networks ICANN (2002) 516–521, Springer LNCS 2415
  (mit T. Ragg und R. Schoknecht)

© Dr. Frank Padberg 2003

Vielen Dank!

© Dr. Frank Padberg 2003