

## 2.5 Auswertung von Simulationsläufen

Ziel der Simulation  $\Rightarrow$  Aussagen über ein System S

Erinnerung an Kapitel 1:

- Systemanalytiker betrachtet beobachtbare Größen P mit Werten  $w_P \in W_P$
- Größen P werden beeinflusst durch kontrollierbare /unkontrollierbare Größen C und U mit Werten  $w_C \in W_C$  und  $w_U \in W_U$
- Modell (hier Simulator) sollte den Einfluss  $(C,U) \rightarrow P$  „wie das reale System“ wiedergeben
- Simulator ist numerisches Modell zur „punktweisen“ Ermittlung der Funktion(?)

Welche Werte  $w_P$  werden bei Setzung von  $w_C$  und Annahme bestimmter Werte für  $w_U$  erreicht?

Ziel der simulativen Analyse: Ermittlung von Leistungsmaß  $Y$

Wir arbeiten meist mit stochastischen Simulatoren, einzelne

Parameter müssen durch Zufallsvariablen spezifiziert werden!

Damit ist zu erwarten, dass auch die beobachteten Größen wie  $Y$

„schwankende Verläufe“ haben

In der Simulation bedeutet dies, dass durch Setzen eines anderen

Startwertes (Saat) des Zufallszahlengenerators ein anderer

Verlauf entsteht

Ist dies realistisch?

- Ja, auch in der Realität verlaufen viele dynamische Abläufe immer „etwas anders“  
z.B. Verkehrsaufkommen, Fertigungsprozesse,  
Pflanzenwachstum, ...

⇒  $Y$  wird durch Realisierung des Zufalls beeinflusst

Generelle Aussagen müssten sich über alle möglichen Abläufe erstrecken!

**Ein stochastischer Prozess  $Y(t)$  ist eine Funktion über dem Parameterraum  $T$ , deren Resultate ZVs sind.**

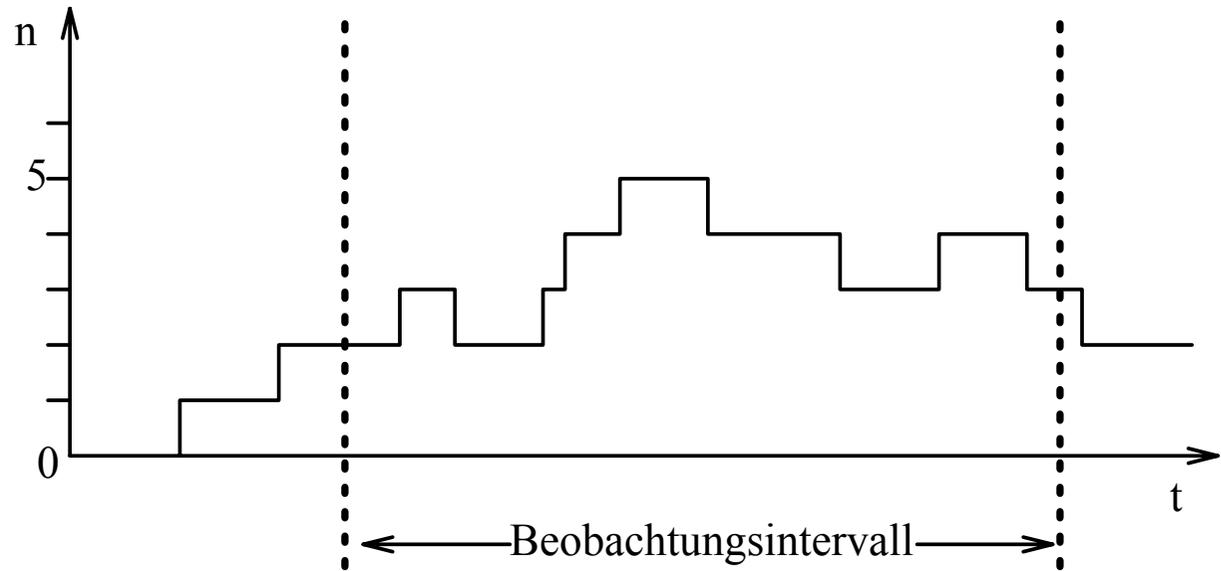
- Normalerweise wird Parameterraum  $T$  als Zeit interpretiert (diskret oder kontinuierlich)
- Für stochastische Prozesse kann man Momente, Dichte und Verteilungsfunktion zum Zeitpunkt  $t$  über  $Y(t)$  definieren
- Eine Realisierung von  $Y(t)$  über  $T$  bezeichnet man als **Trajektorie**

Beispiele:

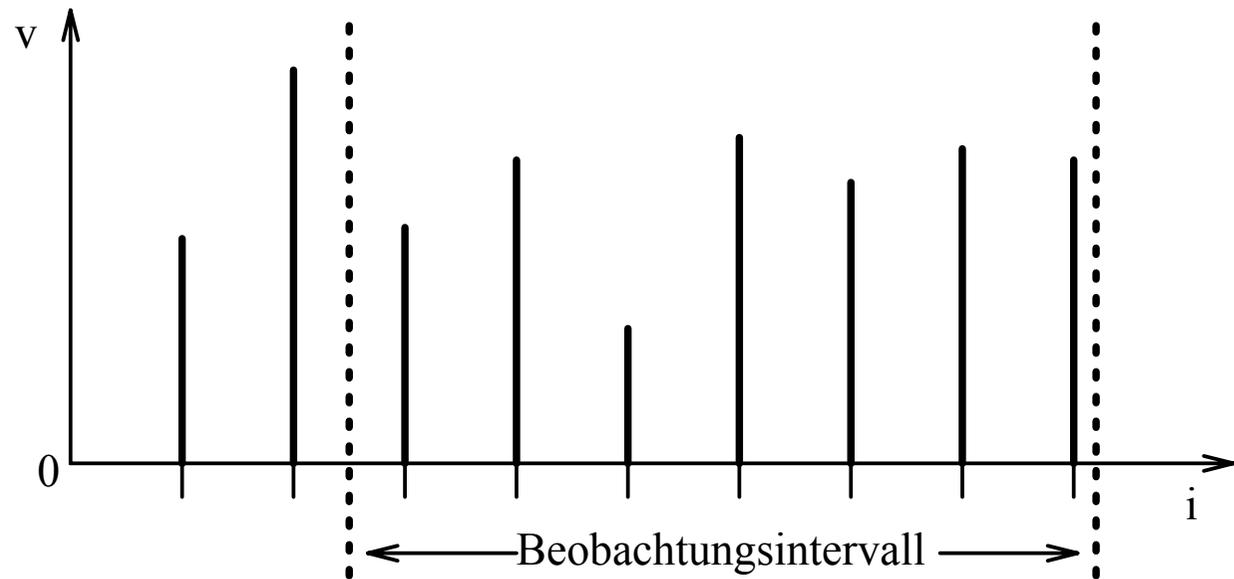
- Würfeln mit zwei unterscheidbaren Würfeln  
 $Y(t) := Y(t-1) + \text{Wert 1. Würfel} - \text{Wert 2. Würfel}$   
mit  $Y(0) = 0$
- Kundenzahl an einem Bankschalter  
 $Y(t)$  nimmt nicht negative ganzzahlige Werte an, die sich zu Ankunfts- und Abgangszeiten von Kunden ändern

# Typische Beobachtungsszenarien

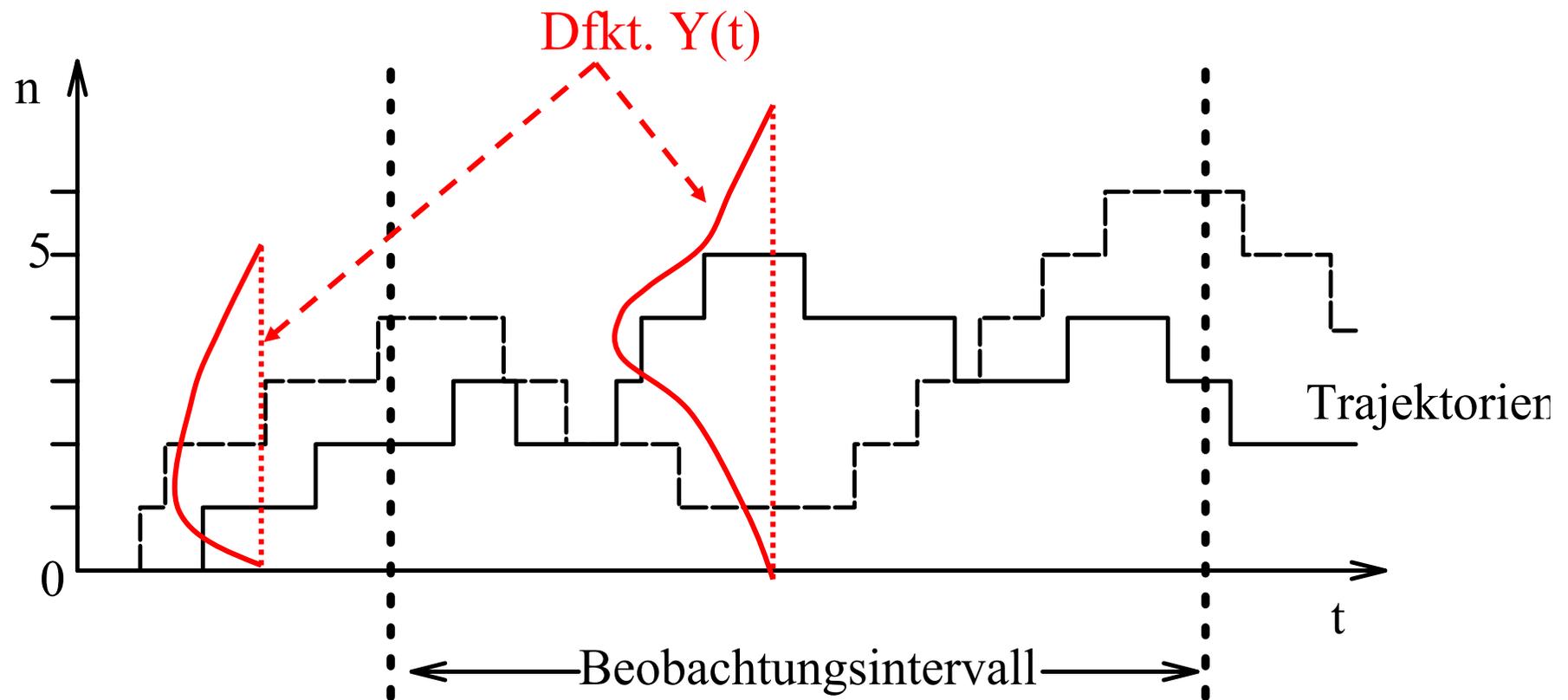
„Zustand“ über der Zeit  
(z.B. Anzahl anwesender Kunden:  $n$ )



„Schicksal“ temporärer Einheiten  
(z.B. Verweilzeit Kunden:  $v$ )



Auf Grund der stochastischen Einflüsse kann jede Trajektorie anders verlaufen



Beobachtung mehrerer Trajektorien liefert Aussagen über mögliche Abläufe (d.h. über die ZVs  $Y(t)$ )

Praktikabel Ermittlung einzelner Charakteristika von  $Y(t)$

(z.B.  $E(Y(t))$ ) und nicht vollständige Charakterisierung!

Beobachtungslage: Wir beobachten Realisierungen von  $Y$  während eines Simulationslaufs!

Sei  $Y_j$  ZVs der  $j$ -ten Beobachtung (von insgesamt  $m$  Beobachtungen)

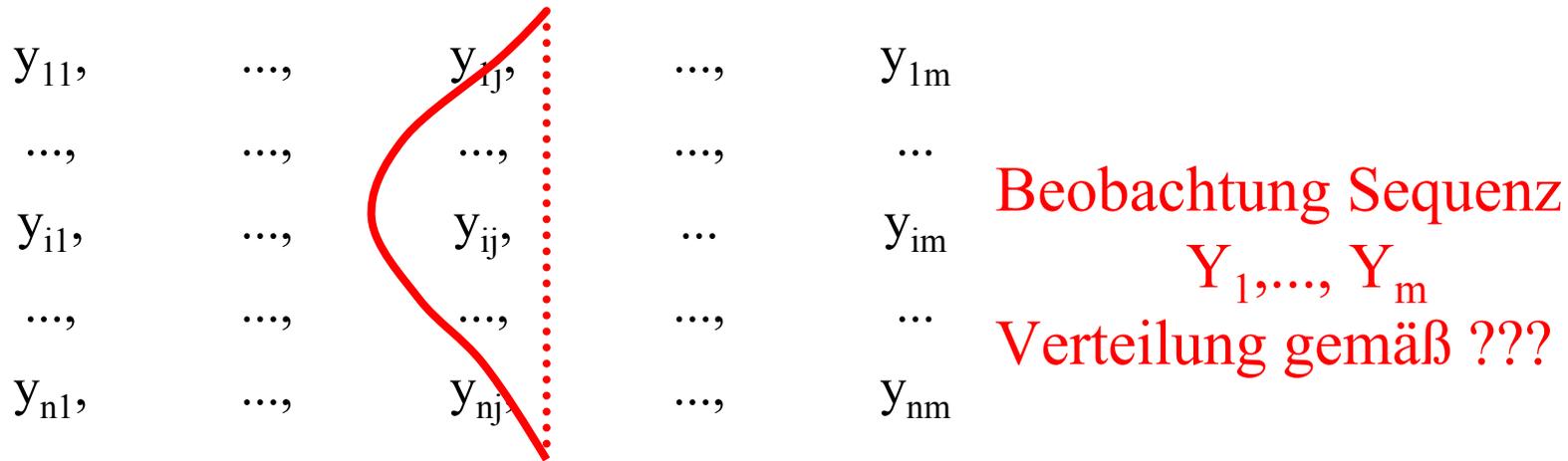
Beispiele:

- Verweilzeit des  $j$ -ten Kunden im System
- Durchsatz nach  $j$  Stunden
- Zeit des  $j$ -ten Ausfalls einer Komponente

Durch unterschiedliche Startwerte des ZZ-Generators erhalten wir unterschiedliche Trajektorien mit Beobachtungen  $y_{ij}$  für die Realisierung von  $Y_j$  in der  $i$ -ten Trajektorie

Beobachtung von  $n$  Trajektorien liefert  $n \cdot m$  Beobachtungswerte  
Auswertungsmöglichkeiten

- längs zu einer Trajektorie ( $y_{ij}$  mit  $i \in \{1, \dots, n\}$  und  $j = 1, \dots, m$ )
- quer zu einer Trajektorie ( $y_{ij}$  mit  $i = 1, \dots, n$  und  $j \in \{1, \dots, m\}$ )
- Kombination aus beiden Möglichkeiten

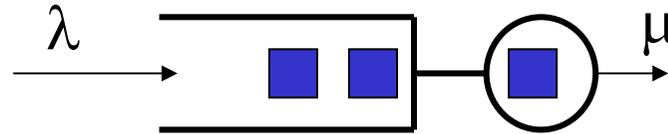


Beobachtung von  $Y_j$   
 Verteilung gemäß  $F_{Y_j}(y)$

Was gilt für die einzelnen Beobachtungen  $y_{ij}$  und  $y_{kl}$  ?

- Falls  $j=1$  und  $i \neq k$  beschreiben beide Beobachtungen Realisierung von  $Y_j$ 
  - Falls die verwandten ZZ unabhängig (nicht überlappende Sequenzen), dann sind auch  $y_{ij}$  und  $y_{kl}$  unabhängig
- Falls  $j \neq 1$  und  $i \neq k$  beschreiben die Beobachtungen Realisierungen von  $Y_j$  und  $Y_1$ 
  - Unabhängigkeit ?? identische Verteilung ??

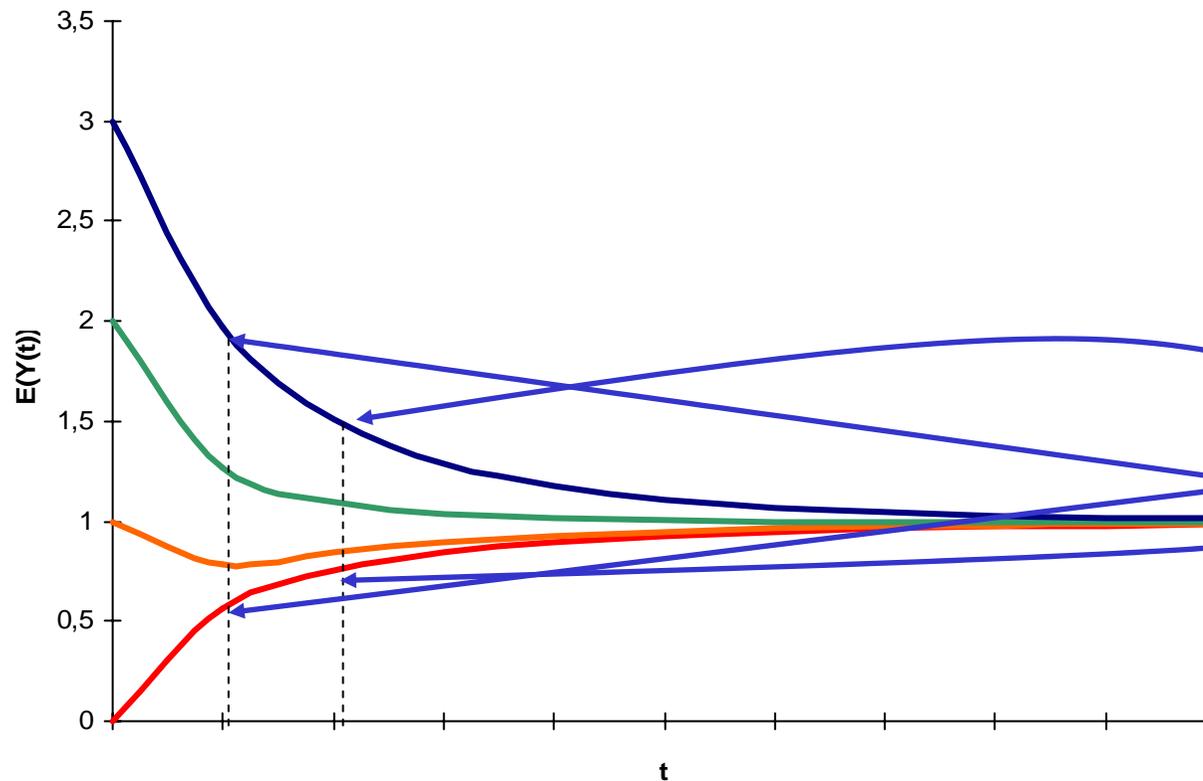
Beispiel:



M/M/1 System

(exponentiell verteilte Zwischenankunfts- und Bedienzeiten)

- mit  $n_0$  Aufträgen im System zu Beginn
- Ankunftsrate  $\lambda = 0.5$ , Bedienrate  $\mu = 1.0$
- $Y_j$  Kundenzahl im System nach  $j$  Zeiteinheiten



Unterschiedliche  
Verläufe, aber  
anscheinend  
Konvergenz der  
Erwartungswerte

$Y_j$  und  $Y_{i+1}$  sind  
offensichtlich nicht  
unabhängig!

Ziel der Simulationsauswertung ist es,  $Y$  auf Basis von Beobachtungen zu charakterisieren

In der Regel werden  $E(Y)$  und  $\sigma^2(Y)$  geschätzt, wir benutzen deshalb die Schreibweise  $\tilde{Y}$  für den Erwartungswertschätzer und  $\hat{Y}$  für den konkreten Schätzwert

Nehmen wir an, dass  $\hat{Y}_j$  aus  $n$  Beobachtungen ermittelt werden soll

Naheliegende Berechnung: 
$$\hat{Y}_j = \frac{1}{n} \sum_{i=1}^n y_{ij}$$

Aber  $\hat{Y}_j$  variiert offensichtlich in Abhängigkeit der Wahl der Startwerte des ZZ-Generators

- Welche Werte können wir für  $\hat{Y}_j$  erwarten?
- Wie weit ist  $\hat{Y}_j$  von  $E(Y_j)$  entfernt?

$\tilde{Y}_j$  ist offensichtlich ZVs.

Damit können Aussagen nur auf Basis der Verteilungscharakteristika gemacht werden

## Schätzung von $E(Y)$

$\tilde{Y}_j$  ist ein erwartungstreuer Schätzer, d.h.  $E(\tilde{Y}_j) = E(Y_j)$

(dies gilt sogar, wenn die  $y_{ij}$  nicht unabhängig sind)

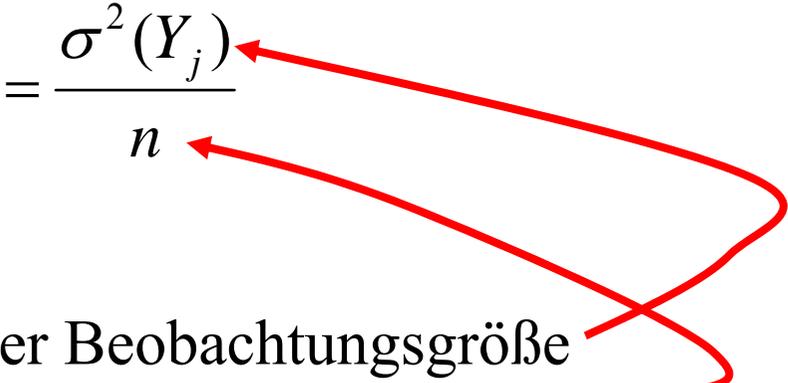
Erwartungstreue sagt aus, dass der Erwartungswert des Schätzers dem zu schätzenden Wert entspricht

Es ist aber keine Aussage über die Qualität des Schätzers nach  $n$  Beobachtungen darin enthalten

Betrachten wir die Varianz  $\sigma^2(\tilde{Y}_j)$  des Schätzers

$$\begin{aligned}\sigma^2(\tilde{Y}_j) &= E\left(\left(\tilde{Y}_j - E(\tilde{Y}_j)\right)^2\right) = \\ E\left(\left(\frac{1}{n} \sum_{i=1}^n Y_{ij} - E(Y_j)\right)^2\right) &= E\left(\left(\frac{1}{n} \left(\sum_{i=1}^n (Y_{ij} - E(Y_j))\right)\right)^2\right) = \\ \frac{1}{n^2} \left(\sum_{i=1}^n E\left(\left(Y_{ij} - E(Y_j)\right)^2\right) + \sum_{i=1}^n \sum_{k=1, k \neq i}^n E\left(\left(Y_{ij} - E(Y_j)\right)\left(Y_{kj} - E(Y_j)\right)\right)\right)\end{aligned}$$

Wir haben  $Y_{ij}$  als unabhängig und damit auch unkorreliert angenommen  $\Rightarrow$  Doppelsummen verschwinden

$$\begin{aligned}\sigma^2(\tilde{Y}_j) &= \frac{1}{n^2} \sum_{i=1}^n E\left(\left(Y_{ij} - E(Y_j)\right)^2\right) \\ &= \frac{1}{n^2} \cdot n \cdot \sigma^2(Y_j) &= \frac{\sigma^2(Y_j)}{n}\end{aligned}$$


Schwankungsbreite des Schätzers

- steigt mit der Schwankungsbreite der Beobachtungsgröße
- fällt mit der Anzahl Beobachtungen

Erwartungstreuer Schätzer für  $\sigma^2(Y_j)$ :  $\tilde{S}_j^2 = \frac{1}{n-1} \sum_{i=1}^n (Y_{ij} - \tilde{Y}_j)^2$

Damit ist  $\tilde{S}_j^2/n$  ein erwartungstreuer Schätzer für  $\sigma^2(\tilde{Y}_j)$

und  $\hat{S}_j^2/n$  der konkrete Schätzwert

## Praktische Berechnung der Schätzer:

### Berechnung von $\hat{Y}_j$ :

- Es ist unpraktisch alle Werte  $y_{ij}$  zu speichern und  $\hat{Y}_j$  am Ende zu berechnen (hoher Speicherplatzbedarf, keine Zwischenergebnisse)
- Besser ist die Nutzung der rekursiven Beziehung  
$$\hat{Y}_j(k) = (k-1)/k \hat{Y}_j(k-1) + y_{kj}/k \text{ für } k = 1, 2, \dots, n \text{ und } \hat{Y}_j(0) = 0$$

### Berechnung der Stichprobenvarianz:

- Erste Möglichkeit, direkte Auswertung der Formel

$$\hat{S}_j^2 = \frac{1}{n-1} \sum_{i=1}^n (y_{ij} - \hat{Y}_j)^2$$

dazu müssen alle Werte  $y_{ij}$  gespeichert werden

- Zweite Möglichkeit, Summen über  $y_{ij}$  und  $y_{ij}^2$  speichern und die Berechnung als

$$\hat{S}_j^2 = \frac{1}{n-1} \left( \sum_{i=1}^n (y_{ij})^2 - n \cdot (\hat{Y}_j)^2 \right) \text{ weniger Speicherbedarf, aber numerisch nicht stabil}$$

Bleibt immer noch die Frage:

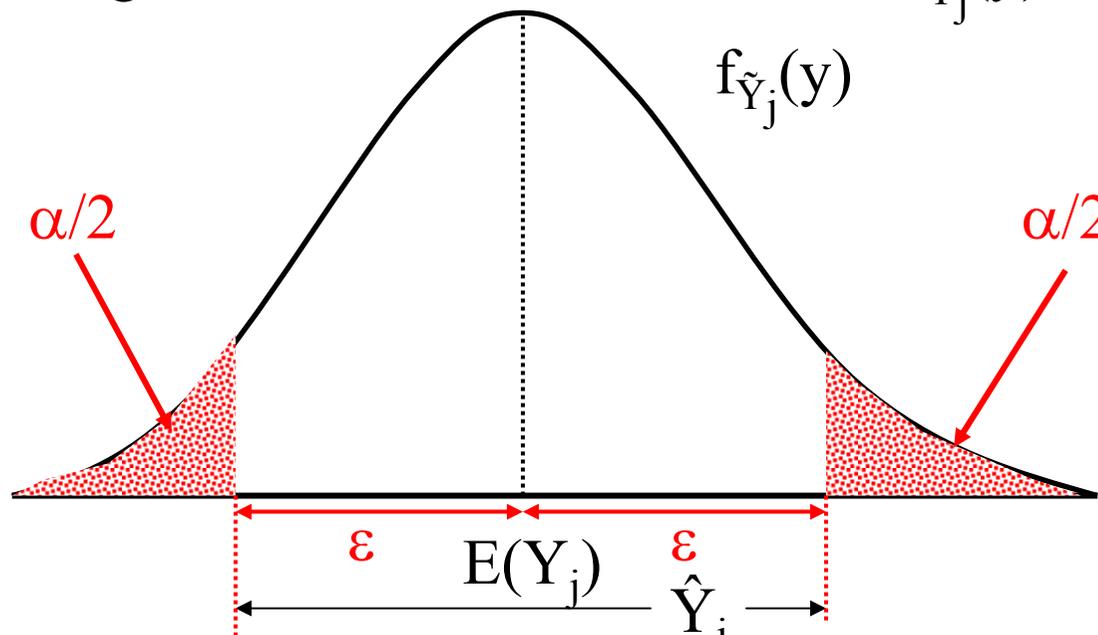
Wie weit ist der ermittelte Schätzwert  $\hat{Y}_j$  vom wahren Erwartungswert  $E(Y_j)$  entfernt?

Da wir es mit ZVs zu tun haben, muss die Frage lauten:

Wie groß ist die Wahrscheinlichkeit, dass  $\tilde{Y}_j$  um mehr als  $\varepsilon$  von  $E(Y_j)$  entfernt ist?

$$\text{d.h. } P[|\tilde{Y}_j - E(Y_j)| \geq \varepsilon] = \alpha$$

Angenommen die Dichtefunktion  $f_{\tilde{Y}_j}(y)$  sei bekannt



$\hat{S}^2$  ist ein Indiz für die Breite des Intervalls (die Größe von  $\varepsilon$ )

Möglicher Interpretationen der Fragestellung:

1. Wie groß ist die Wahrscheinlichkeit, dass  $\hat{Y}_j$  aus dem Intervall  $[E(Y_j)-\varepsilon, E(Y_j)+\varepsilon]$  herausfällt?
2. Wie groß ist die Wahrscheinlichkeit, dass das Intervall  $[\hat{Y}_j-\varepsilon, \hat{Y}_j+\varepsilon]$   $E(Y_j)$  nicht enthält?

Da  $E(Y_j)$  unbekannt (aber  $\hat{Y}_j$  bekannt), ist 2. hilfreicher  
Beantwortung ist aber aus 1. zu gewinnen

(zumindest prinzipiell, falls  $f_{\hat{Y}_j}(y)$  bekannt wäre)

Offensichtlich sind  $\alpha$  und  $\varepsilon$  wechselseitig abhängig

- bisherige Argumentation von  $\varepsilon$  in Richtung zugehöriges  $\alpha$
- übliche Fragestellung umgekehrt: Ausgehend von festem  $\alpha$  zugehöriges  $\varepsilon$  suchen

Wenn ich mit Wahrscheinlichkeit  $1-\alpha$  sicher sein will, dass  $E(Y_j)$  nicht mehr als  $\varepsilon$  von  $\hat{Y}_j$  abweicht, mit welchem  $\varepsilon$  ist dann zu rechnen?

**Man nennt  $\hat{Y}_j \pm \varepsilon$  ein  $(1-\alpha) \cdot 100\%$  Konfidenzintervall für  $E(Y_j)$**

## $[\hat{Y}_j - \varepsilon, \hat{Y}_j + \varepsilon]$ ist ein Intervallschätzer

Intervallschätzer sind ungleich wertvoller als Punktschätzer, da auf Grund der Variabilität von Punktschätzern diese allein nur wenig Information beinhalten!

Bleibt die konkrete Frage, welche  $(\varepsilon, \alpha)$ -Paare zusammengehören?

Bisherige Antworten basieren auf der Kenntnis der Dichtefunktion

Wir haben aber bisher „nur“ erwartungstreue Schätzer für Erwartungswert und Varianz!

⇒ Auf dieser Basis können höchstens Schranken für  $\varepsilon$  bei festem  $\alpha$  ermittelt werden

## Tschebyscheff'sche Ungleichung

Für eine beliebige ZV  $X$  mit  $E(X)$  und  $\sigma^2(X)$  gilt (für beliebiges  $c > 0$ ):

$$P[|X - E(X)| \geq c] \leq \sigma^2(X) / c^2$$

für unseren Erwartungswertschätzer gilt

$$P[|\tilde{Y}_j - E(Y_j)| \geq \varepsilon] \leq \sigma^2(Y_j) / (n \cdot \varepsilon^2)$$

Hantieren mit der Ungleichung (ein Beispiel):

Bestimmung des 90% Konfidenzintervalls für  $\hat{Y}_j$  also

$$\alpha = \sigma^2(Y_j) / (n \cdot \varepsilon^2) = 0.1$$

Habe die Stichprobe Umfang 10, dann  $\sigma^2(Y_j) / (10 \cdot \varepsilon^2) = 0.1$

d.h.  $\varepsilon^2 = \sigma^2(Y_j)$  bzw.  $\varepsilon = \sigma(Y_j)$

Bei errechnetem Punktschätzer  $\hat{Y}_j$  ist

$[\hat{Y}_j - \sigma(Y_j), \hat{Y}_j + \sigma(Y_j)]$  gesuchtes Konfidenzintervall

Beobachtung bei fester Varianz  $\sigma^2(Y_j)$ :

Halbierung der Konfidenzintervallbreite bei festem  $\alpha$

$\Rightarrow$  Vervierfachung der Anzahl Elemente in der Stichprobe

da 
$$\frac{\sigma^2(Y)}{n \cdot \varepsilon^2} = \frac{\sigma^2(Y)}{m \cdot (\varepsilon / 2)^2} \Rightarrow m = 4 \cdot n$$

Leider hat die Konfidenzintervallbestimmung nach Tschebyscheff zwei Nachteile:

- Bei Berechnung des Konfidenzintervalls wird die Varianz  $\sigma^2(Y_j)$  der beobachteten Größe benötigt, diese ist aber unbekannt  
Man könnte versuchen,  $\sigma^2(Y_j)$  durch  $\hat{S}_j^2$  zu ersetzen  
Damit ist die Voraussetzung von Tschebyscheff verletzt!
- Tschebyscheff liefert recht breite Konfidenzintervalle,
  - da ganz allgemein gültig
  - ohne weitere Informationen über die Verteilung von  $\tilde{Y}_j$  gewonnen

Bei Vorliegen von Verteilungsinformation sollten schmalere Konfidenzintervalle gewinnbar sein

Zusammenfassend:

- „Pessimistischer“ Charakter von Tsch.-Konfidenzintervall
- Gibt gewisse Berechtigung  $\sigma^2(Y_j)$  durch  $\hat{S}^2$  zu ersetzen
- Wir tolerieren Tsch. mit  $\hat{S}^2$  als Intervallschätzer

Beispiel Münzwurf mit fairer Münze ( $E(Y) = 0.5$ ,  $\sigma^2(Y) = 0.25$ )  
je Experimentserie 10 Würfe (Zahl=1, Kopf=0):

1. Experimentserie 0000101001:  $\hat{Y} = 0.3$ ,  $\hat{S}^2 = 0.23333$   
Konfidenzintervall für  $\alpha = 0.1$ :  $[-0.183, 0.783]$   
aus Kenntnis der Experimentsituation wissen wir, dass die untere Schranke 0.0 ist
2. Experimentserie 0110111001:  $\hat{Y} = 0.6$ ,  $\hat{S}^2 = 0.26667$   
Konfidenzintervall für  $\alpha = 0.1$ :  $(0.084, 1.116)$   
aus Kenntnis der Experimentsituation wissen wir, dass die obere Schranke 1.0 ist
3. Kombination aus 1. und 2.:  $\hat{Y} = 0.45$ ,  $\hat{S}^2 = 0.2605$   
Konfidenzintervall für  $\alpha = 0.1$ :  $[0.089, 0.811]$

Aussage der Intervallschätzer:

In 90% der Fälle liegt der gesuchte Wert im Konfidenzintervall!

Aber auch, in 10% kann der Wert außerhalb des Konfidenzintervalls liegen

Tschebyscheff berechnet Konfidenzintervalle ohne Information über die Verteilung der Schätzer

⇒ Information über die Verteilung verkleinert die Konfidenzintervalle

Welche Informationen über die Verteilung sind bekannt?

### **Zentraler Grenzwertsatz**

Seien  $Y_1, \dots, Y_n$  unabhängig identisch verteilte ZVs mit endlichem

$E(Y)=\mu$  und Varianz  $\sigma^2$  und sei  $\tilde{\mu} = \frac{1}{n} \sum_{i=1}^n Y_i$

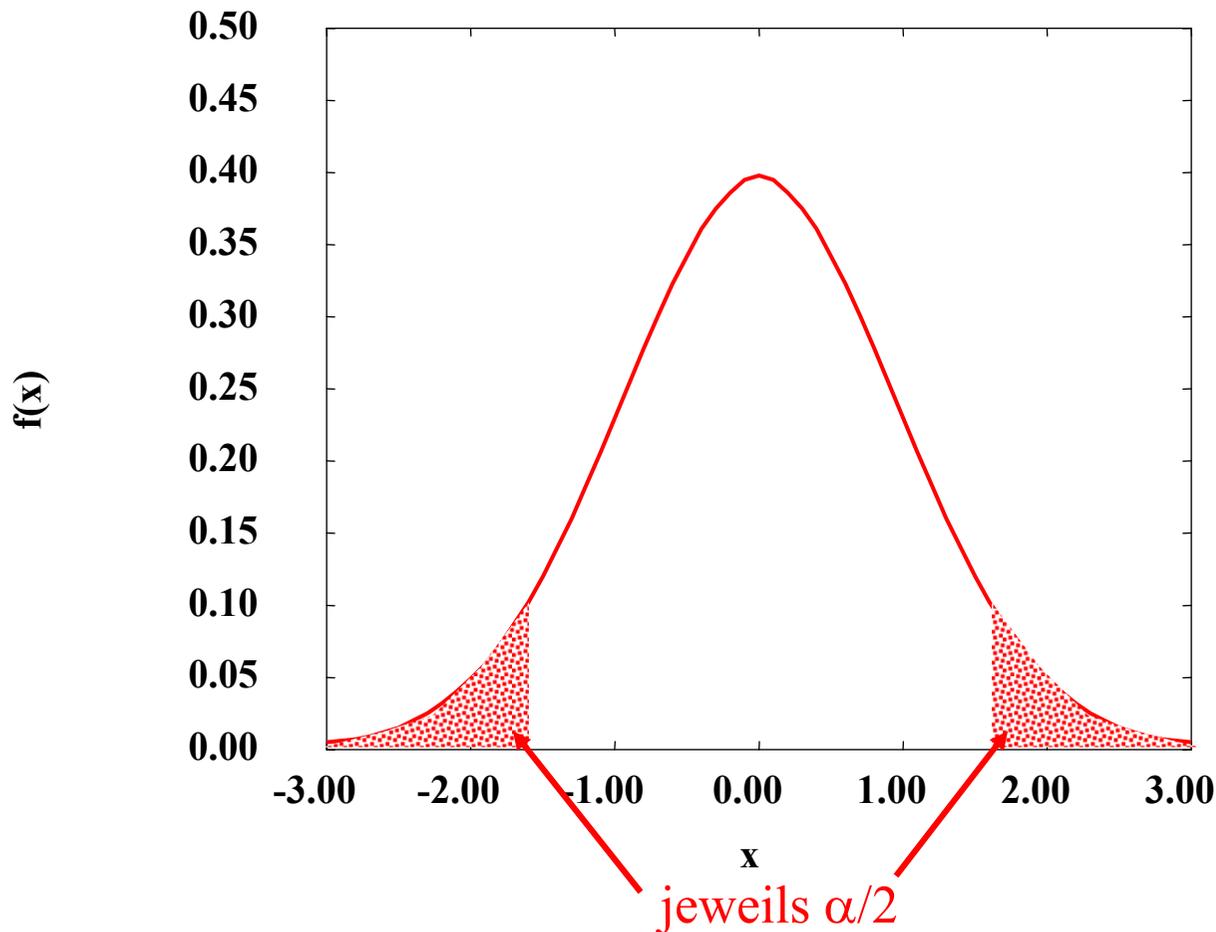
Sei eine ZV  $Z$  definiert als

$$Z_n = \frac{\tilde{\mu} - \mu}{\sigma / \sqrt{n}}$$

Dann approximiert die Verteilung von  $Z$  für große  $n$  die  $N(0,1)$ -Verteilung mit Vfkt  $\Phi(z)$  beliebig genau

(d.h.  $\lim_{n \rightarrow \infty} F_{Z_n}(z) = \Phi(z)$  wobei  $\Phi(z)$  die Vfkt. von  $N(0,1)$  ist)

Zentraler Grenzwertsatz gilt auch, wenn  $\sigma^2$  durch  $\tilde{S}^2$  ersetzt wird!  
Damit können Konfidenzintervalle aus der  $N(0,1)$ -Dichtefunktion gewonnen werden, falls  $n$  „groß genug“ ist



Kritische Werte der  
 $N(0,1)$ -Verteilung  
sind vertafelt!

Damit kann das  $(1-\alpha)\cdot 100\%$  Konfidenzintervall für  $\tilde{Y}$  berechnet werden als

$$P\left[|Z| \geq \varepsilon\right] = P\left[\left|\frac{\tilde{Y} - E(Y)}{\tilde{S} / \sqrt{n}}\right| \geq \varepsilon\right] = \alpha \Rightarrow P\left[|\tilde{Y} - E(Y)| \geq \varepsilon \cdot \tilde{S} / \sqrt{n}\right] = \alpha$$

damit ergibt sich folgendes Konfidenzintervall

$$\tilde{Y} \pm \varepsilon_{\alpha} \cdot \tilde{S} / \sqrt{n}$$

$\varepsilon_{\alpha}$  aus einer Tabelle für  $N(0,1)$ -Vert. ablesen

Konfidenzintervall für das Münzbeispiel (Folie 182):  $[0.262, 0.638]$   
(Tschebyscheff lieferte  $[0.089, 0.811]$ )

Konfidenzintervalle werden durch die Annahme der Normalverteilung für den Schätzer deutlich schmaler!

- Wann ist die Normalverteilungsannahme gerechtfertigt?
- Wie groß muss  $n$  sein?

Notwendige Größe von n hängt von der (unbekannten) Verteilung von Y ab!

Falls n zu klein gewählt wird, so kann die Wahrscheinlichkeit, dass der wahre Wert außerhalb des berechneten Konfidenzintervalls liegt deutlich größer als  $\alpha$  sein  
(die Abdeckung des Konfidenzintervalls ist kleiner als  $\alpha$ ).

Eines der wenigen bekannten Ergebnisse:

Falls Y normalverteilt ist, so ist  $\frac{(\tilde{Y} - E(Y))}{\tilde{S} / \sqrt{n}}$   
t-verteilt mit n-1 Freiheitsgraden

Damit ergibt sich das Konfidenzintervall  $\hat{Y} \pm t_{n-1, 1-\alpha/2} \cdot \frac{\hat{S}}{\sqrt{n}}$

Wobei  $t_{n-1, 1-\alpha/2}$  das (vertafelte) Quantil der t-Verteilung ist  
Die t-Verteilung liefert pessimistischere Konfidenzintervalle als die Normalverteilung und konvergiert für  $n \rightarrow \infty$  gegen eine Normalverteilung

Konfidenzintervalle für das Beispiel [0.252, 0.647] statt [0.262, 0.638]

In der Praxis ist  $Y$  meist nicht normalverteilt, trotzdem sollte für kleine Stichprobenumfänge die  $t$ -Verteilung wegen ihres pessimistischeren Charakters verwendet werden!

Experimentelle Untersuchung der Abdeckung eines 90% Konfidenzintervalls auf Basis der  $t$ -Verteilung für unterschiedliche Verteilungen von  $Y$ .

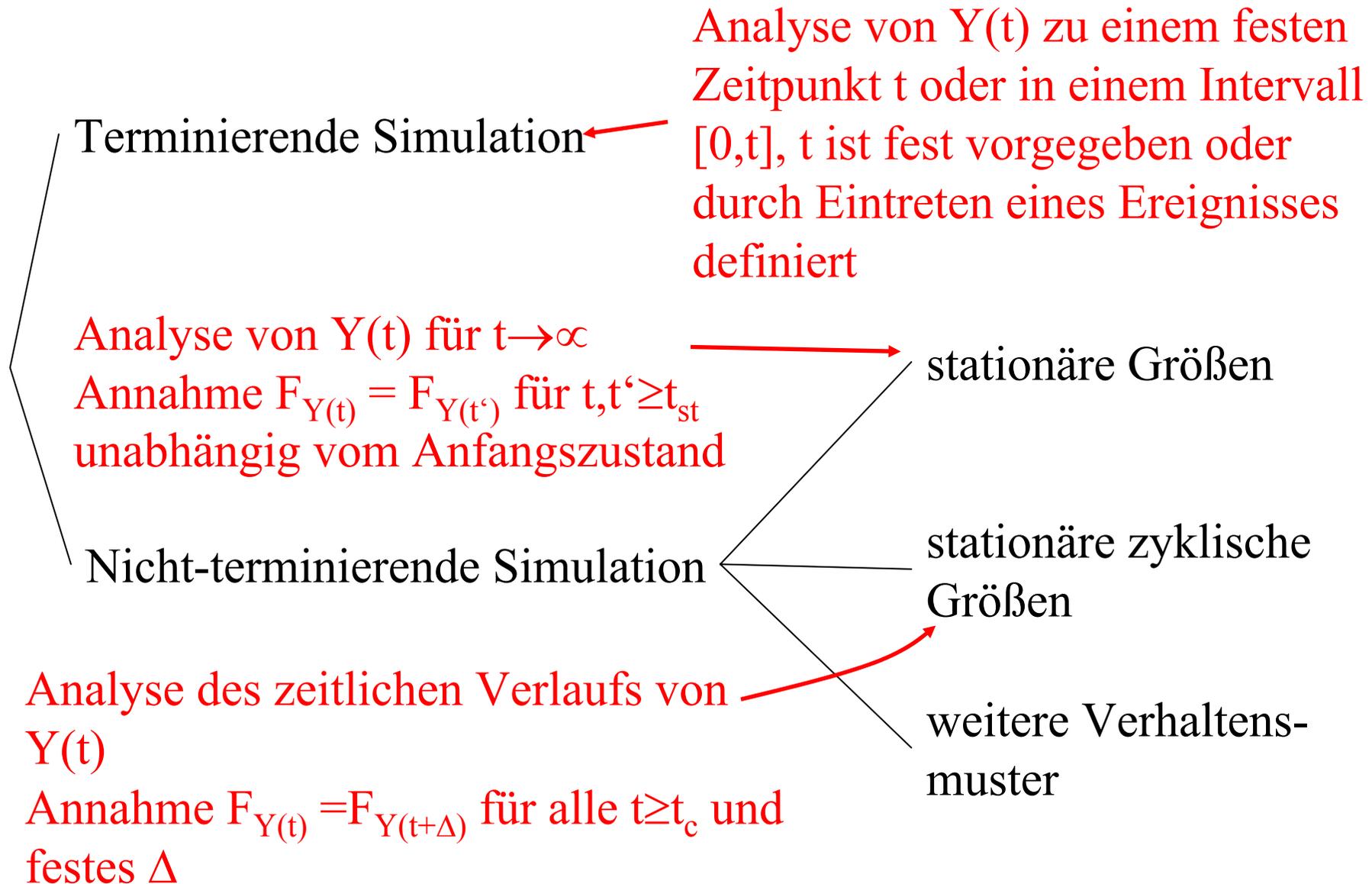
Resultate basieren auf jeweils 500 Experimenten

Verteilung	n=5	n=10	n=20	n=40
Normal	0.910	0.902	0.898	0.900
Exponential	0.854	0.878	0.870	0.890
Lognormal	0.758	0.768	0.842	0.852
Hyperexp.	0.584	0.586	0.682	0.774

Typische Problemstellungen, die mittels Simulation gelöst werden:

1. Durchsatzbestimmung einer Fertigungsstraße im Dreischichtbetrieb
  - System läuft über einen längeren Zeitraum ohne Unterbrechung und Belastungen sind zeitunabhängig
2. Kapazitätsplanung eines geplanten Kommunikationsnetzes
  - Schätzung homogener Ankunfts- und Bedienzeiten
3. Belieferung von Einzelhändlern mit Speiseeis
  - Belieferungsmenge abhängig vom Verbrauch, der vom Wetter (und damit auch von der Jahreszeit) abhängt
4. Bestimmung der mittleren Zeit bis zum Ausfall eines Computersystems
  - bei gegebenen Ausfallzeitverteilungen der Komponenten
5. Bestimmung der Kassenauslastung eines Supermarktes zwischen 9 und 12 Uhr
  - bei einem einem zeitlich variierendem Kundestrom

# Simulationstypen bzgl. Auswertung



Analyse des zeitlichen Verlaufs von  $Y(t)$   
Annahme  $F_{Y(t)} = F_{Y(t+\Delta)}$  für alle  $t \geq t_c$  und festes  $\Delta$

# Terminierende Simulation

Ziel einer terminierenden Simulation:

Analyse eines Systems im Intervall  $[0, T_E]$

$T_E$  kann

- als feste Zeit vorgegeben sein oder
- selbst ZV sein, die durch das Eintreten eines Ereignisses definiert ist

Resultat beschrieben durch ZV  $Y$  deren Erwartungswert ermittelt werden soll

$Y$  kann

- zum Ende des Intervalls d.h. zum Zeitpunkt  $T_E$  gemessen werden (kann z.B. auch der Wert  $T_E$  selbst sein)
- oder aus mehreren Beobachtungen im Intervall  $[0, T_E]$  berechnet werden

(z.B. als zur Bestimmung der mittleren Population durch  $\int_0^{T_E} Q(t)dt$  wobei  $Q(t)$  die Population zum Zeitpunkt  $t$  ist)

Übliches Vorgehen bei der Schätzung von  $E(Y)$ :  
n-malige Wiederholung der Simulation mit unterschiedlicher  
und unabhängiger Realisierung der ZZs

Die Wiederholungen werden als **Replikationen** bezeichnet.

Erweiterung der Notation:  $\tilde{Y}(n)$ ,  $\hat{Y}(n)$ ,  $\tilde{S}^2(n)$ ,  $\hat{S}^2(n)$  bezeichnen die  
Schätzer bzw. Schätzwerte bei n Replikationen und  
 $y_i$  sei das Resultat der i-ten Replikation

$$\text{Konfidenzintervall } \hat{Y}(n) \pm t_{n-1, 1-\alpha/2} \cdot \sqrt{\frac{\hat{S}^2(n)}{n}}$$

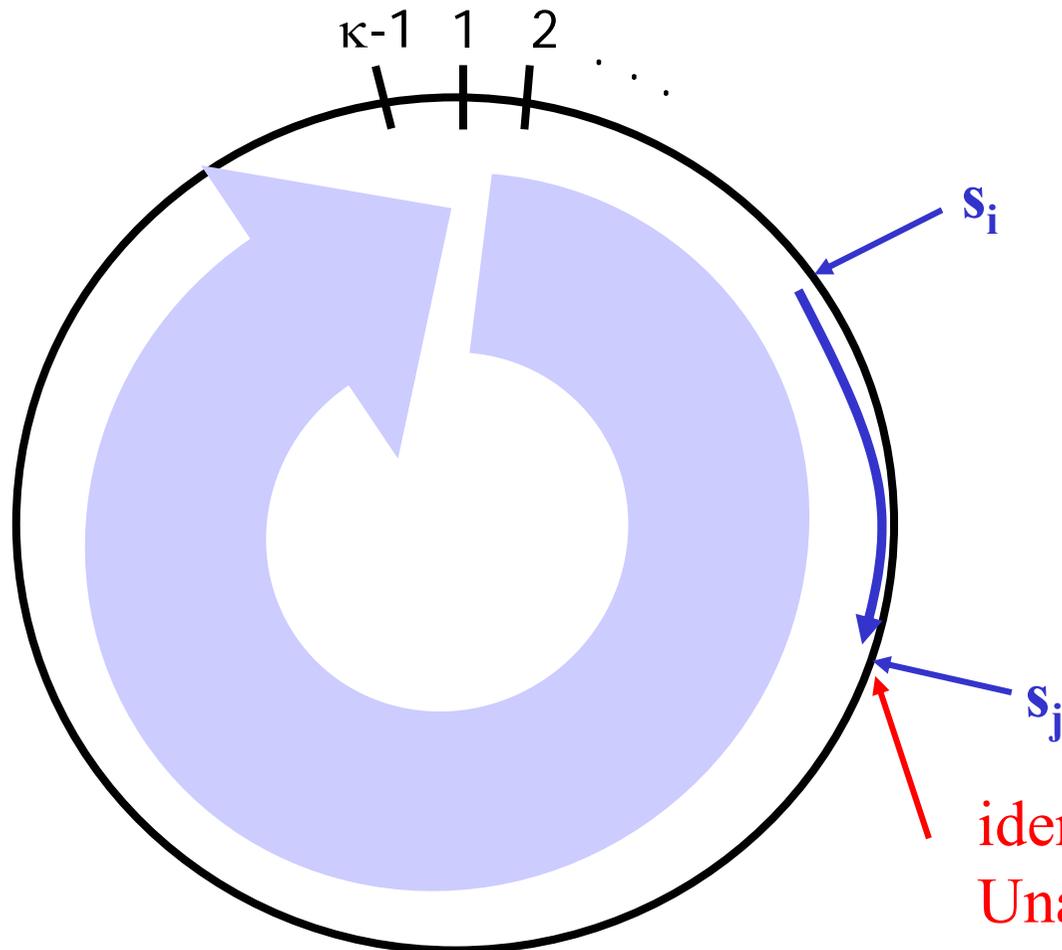
Qualität bzw. Abdeckung des Konfidenzintervalls hängt ab von

- der Nähe der Verteilung  $Y_i$  zu einer Normalverteilung
- der Unabhängigkeit der  $Y_i$

## Unabhängigkeit der $Y_i$

Sei  $s_i$  die Saat des ZZ-Generators der  $i$ -ten Replikation:

- $Y_i$  und  $Y_j$  sind unabhängig, falls die durch  $s_i$  und  $s_j$  definierten Sequenzen von ZZ sich nicht überlappen



Manche ZZ-Generatoren erlauben die Definition von Strömen mit einer vorgegebenen Zahl unabhängiger ZZs (durch implizite Definition der zugehörigen Saaten)

**identische ZZs  
Unabhängigkeit geht verloren**

# Gültigkeit der Annahme normalverteilter Beobachtungen

Annahme ist grundsätzlich modellabhängig

M/M/1-System (siehe Folie 172)

- Parameter  $\lambda=0.9$ ,  $\mu=1.0$ ,
- initial leeres System
- Bestimmung der mittleren Verweilzeit der ersten 25 Aufträge 90% Konf.Int.
- Resultate basieren auf 500 Exp. mit jeweils n Replikationen

n	Abdeckung	Breite rel. zu $\hat{Y}(n)$
5	0.880±0.024	0.67
10	0.864±0.025	0.44
20	0.886±0.023	0.30
40	0.914±0.021	0.21

System mit 3 Komponenten

- Ausfallzeiten Weibull-verteilt
- initial arbeiten alle Komp.
- Ausfall von 1 und (2 oder 3) führt zu Systemausfall
- Bestimmung Ausfallzeit
- Resultate basieren auf 500 Exp. mit jeweils n Replikationen

n	Abdeckung	Breite rel. zu $\hat{Y}(n)$
5	0.708±0.033	1.16
10	0.750±0.032	0.82
20	0.800±0.029	0.60
40	0.840±0.027	0.44

Nachteil des bisherigen Vorgehens:

Durch Vorgabe der Anzahl Replikationen ist unklar, welche Breite des Konfidenzintervalls erreicht wird

Eigentlich möchte man aber Resultate mit einer vorgegebenen Genauigkeit  $\varepsilon^*$  ermitteln  $\Rightarrow$   $n$  muss so gewählt werden, dass  $\varepsilon \leq \varepsilon^*$

$$\begin{aligned} \text{Es gilt } 1 - \alpha &\approx P \left[ \tilde{Y} - \varepsilon \leq E(Y) \leq \tilde{Y} + \varepsilon \right] \\ &= P \left[ |\tilde{Y} - E(Y)| \leq \varepsilon \right] \\ &\leq P \left[ |\tilde{Y} - E(Y)| \leq \varepsilon^* \right] \end{aligned}$$

Sei  $n(\varepsilon^*)$  die Anzahl Replikationen, die notwendig ist, damit die halbe Breite des Konfidenzintervalls kleiner gleich  $\varepsilon^*$  ist

Beachte hier  
wird die halbe  
Breite  
definiert!

$$\text{Es gilt: } n(\varepsilon^*) = \min \left\{ n : t_{n-1, 1-\alpha/2} \sqrt{\frac{\hat{S}^2(n)}{n}} \leq \varepsilon^* \right\}$$

Iterative Bestimmung von  $n(\varepsilon^*)$  bricht Simulation ab, wenn Genauigkeit erreicht!

In der Regel wird Genauigkeit in Abhängigkeit von  $\hat{Y}(n)$  definiert also Abbruch falls  $t_{n-1,1-\alpha/2}(\hat{S}^2(n)/n)^{1/2} / |\hat{Y}(n)| \leq \gamma$   
(Voraussetzung  $E(Y) \neq 0$ )

Falls  $E(Y) \approx 0$  sollte weiter mit absoluten und nicht mit relativen Werten gearbeitet werden!

Damit ergibt sich  $P\left[|\tilde{Y}(n) - E(Y)| \leq \gamma \cdot |E(Y)|\right] \approx 1 - \alpha$

Da wir nur  $\hat{Y}(n)$ , nicht aber  $E(Y)$  kennen, wählen wir  $\varepsilon^* = \gamma \cdot \hat{Y}(n)$  und geben  $\gamma$  vor

Abschätzung von  $n(\gamma)$  aus den Ergebnissen nach  $n$  ( $< n(\gamma)$ )

Replikationen :

$$t_{n-1,1-\alpha/2} \sqrt{\frac{\hat{S}^2(n)}{n}} \leq |\hat{Y}(n)| \cdot \gamma \Rightarrow n(\gamma) \approx \frac{t_{n-1,1-\alpha/2} \cdot \hat{S}^2(n)}{(\hat{Y}(n) \cdot \gamma)^2}$$

Ermittlung anderer Maße von  $Y$

Schätzung des Erwartungswerts ist oft nicht ausreichend

Weitere Maße:

1. Höhere Momente  $E(Y^i)$
2. Wert der Vfkt. an der Stelle  $y$ :  $P[Y \leq y] = F(y) = p_y$
3.  $p$ -Quantil ( $p \in (0,1)$ ) Wert  $y_p$ , so dass  $P[Y \leq y_p] = F(y_p) = p$

1. und 2. sind durch Erweiterungen des bisherigen Vorgehens behandelbar, 3. erfordert zusätzliche Konzepte (hier nicht vorgestellt) oder muss offline für jede Replikation ermittelt werden

Idee des Vorgehens: Definiere neue ZV  $X$ , so dass  $E(X)$  dem gesuchten Maß entspricht

z.B.  $X = Y^2$  (2. Moment) oder

$X = 1$  falls  $Y \leq y$  und  $0$  sonst (Wert der Vfkt. an Stelle  $y$ )

$\hat{X}$  und das zugehörige Konfidenzintervall liefern Resultate für das gewünschte Maß. (Vorsicht bei gleichzeitiger Def. mehrerer Maße)

## Bestimmung des initialen Zustands

In vielen Fällen ist der initiale Zustand fest vorgegeben

z.B. alle Komponenten lauffähig, wenn System startet,  
alle Puffer leer wenn System startet

Manchmal ist der initiale Zustand selbst ZV  $X_0$

z.B. Analyse eines Bankschalters zwischen 10.00 und 12.00 Uhr,  
Wahrscheinlichkeit einer Pufferüberlaufs in einem  
Fertigungssystem während einer Schicht

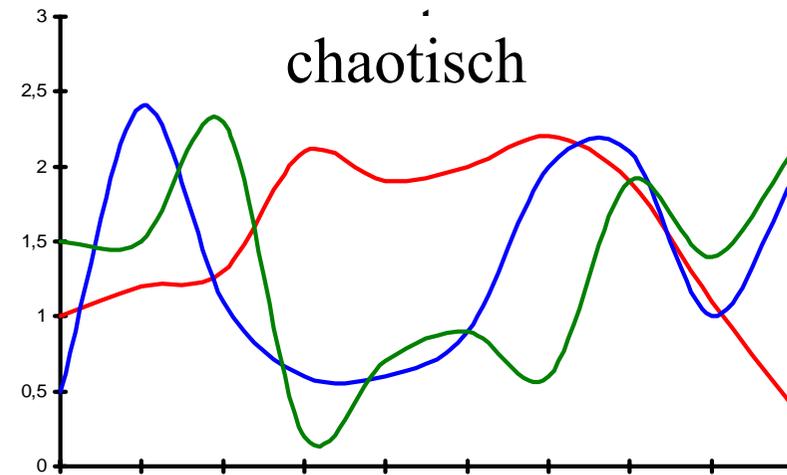
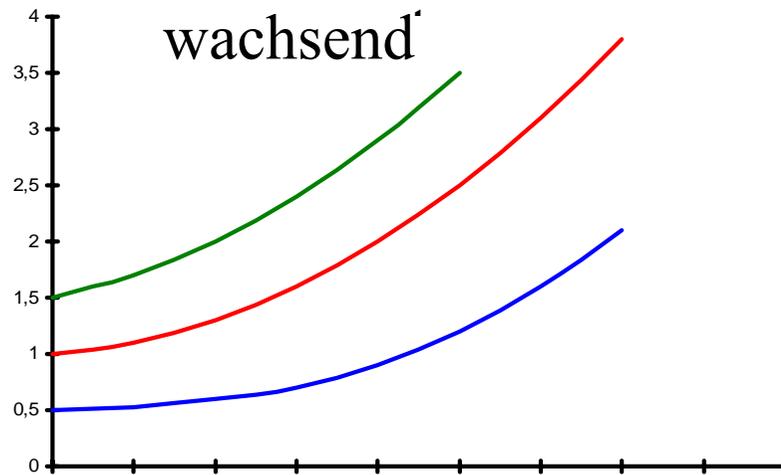
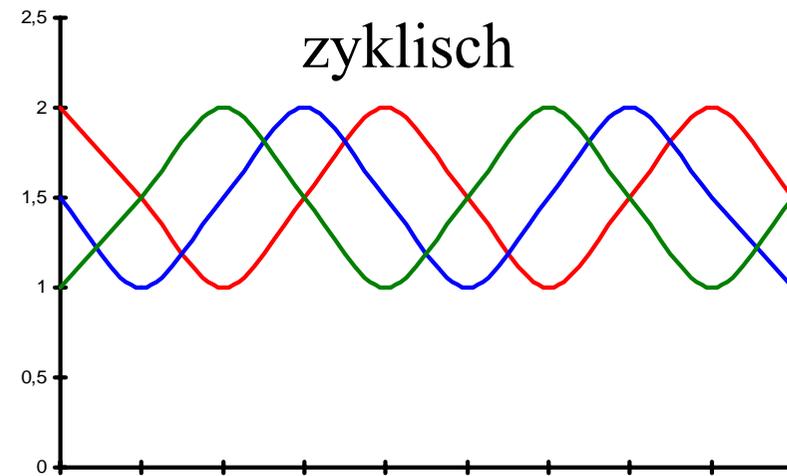
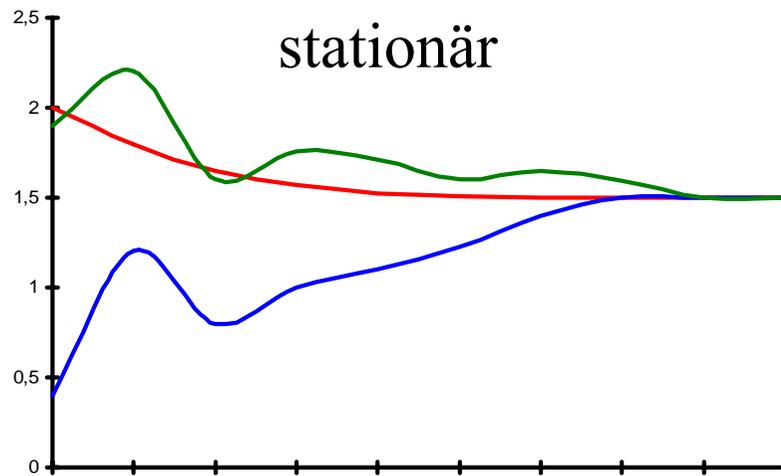
In diesen Fällen hängt das ermittelte Resultat vom (unbekannten)  
initialem Zustand ab

Möglichkeiten der Bestimmung

- Starte System früher in einem „natürlichen“ Anfangszustand und simuliere bis zum Startzeitpunkt
  - wann und in welchem Zustand ist das System zu starten?
- ermittle per Messung mögliche Anfangszustände und deren Wahrscheinlichkeit, ziehe Anfangszustand aus  $F_{X_0}(x)$

# Nicht-terminierende Simulation

Mögliche Verhaltensweisen bei unendlicher/sehr langer Laufzeit



# Statistische Analyse stationärer Simulationen

## Annahme:

Modell erreicht stationären Zustand unabhängig vom Anfangszustand:

$$\lim_{t \rightarrow \infty} E(Y(t)|s_0) = E(Y) \text{ für alle Anfangszustände } s_0$$

wobei  $E(Y(t)|s_0)$  der Erwartungswert von  $Y$  zum Zeitpunkt  $t$  ist, wenn das Modell zum Zeitpunkt 0 im Zustand  $s_0$  gestartet wurde

Obige Annahme gilt für unendliches Beobachtungsintervall,  
in der Praxis ist aber jedes Beobachtungsintervall endlich

Damit kann (und wird meistens) auch gelten

- $E(Y(t)|s_0) \neq E(Y(t)|s'_0)$  und  $E(Y(t')|s_0) \neq E(Y(t)|s_0)$

Wir nehmen an, dass ein Zeitpunkt  $t_s$  existiert, so dass für alle  $t \geq t_s$ :

- $E(Y(t)|s_0) \approx E(Y(t)|s'_0) \approx E(Y)$

In der Regel wird  $t_s$  von  $s_0$  abhängen, da  $s_0$  beliebig wählbar ist,  
nehmen wir an, dass  $s_0$  fest vorgegeben wird (gleich mehr dazu)

Ziel ist die Bestimmung von  $t_s$ , so dass  $E(Y(t)) \approx E(Y)$  für alle  $t \geq t_s$

Vorgehen bei der Simulation:

- Starte Simulator im Zustand  $s_0$  zum Zeitpunkt  $t=0$
- Simuliere bis zum Zeitpunkt  $t_s$  ohne Daten zu erheben
- Starte Beobachtung der Simulation zum Zeitpunkt  $t_s$

Wie ist  $t_s$  zu wählen? Zwei widersprüchliche Argumente:

- Sicherheitsargument: Wähle  $t_s$  möglichst groß.
- Aufwandsargument: Wähle  $t_s$  möglichst klein.

Insgesamt, wähle  $t_s$  so klein wie möglich, um korrekte Resultate zu ermitteln. Methoden zur Ermittlung/Schätzung von  $t_s$

- Heuristiken
- statistische Tests

Oft mit wenig Substanz ,

Zitat von G.S. Fishman: „*No completely satisfactory procedure for resolving the problem has appeared yet*“

Ermittlung von  $t_s$  auf Grund der Beobachtung der Messwerte Stichprobe  $(y_1, \dots, y_n)$  liege vor, die ersten  $i$  ( $\leq n$ ) Werte sollen unberücksichtigt bleiben, da sie vor  $t_s$  erhoben wurden

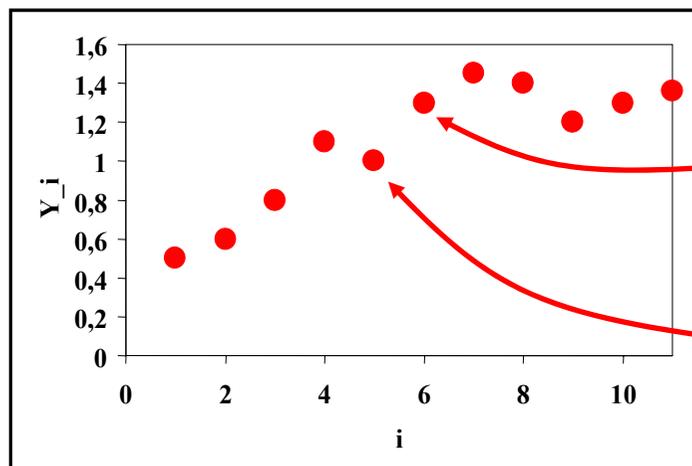
Ziel: Bestimmung von  $i$

Regel von Conway (1963):

- $y_k^+ = \max(y_k, \dots, y_n)$  und  $y_k^- = \min(y_k, \dots, y_n)$
- wähle kleinstes  $k$  mit  $y_k^- < y_k < y_k^+$

Regel von Conway (1978):

- $y_k^+ = \max(y_1, \dots, y_k)$  und  $y_k^- = \min(y_1, \dots, y_k)$
- wähle kleinstes  $k$  mit  $y_k^- < y_k < y_k^+$



Bemerkungen:

Beide Regeln funktionieren nur, wenn  $y_i$  Mittelwerte über mehrere Messungen, auch dann wird  $t_s$  meist gravierend unterschätzt

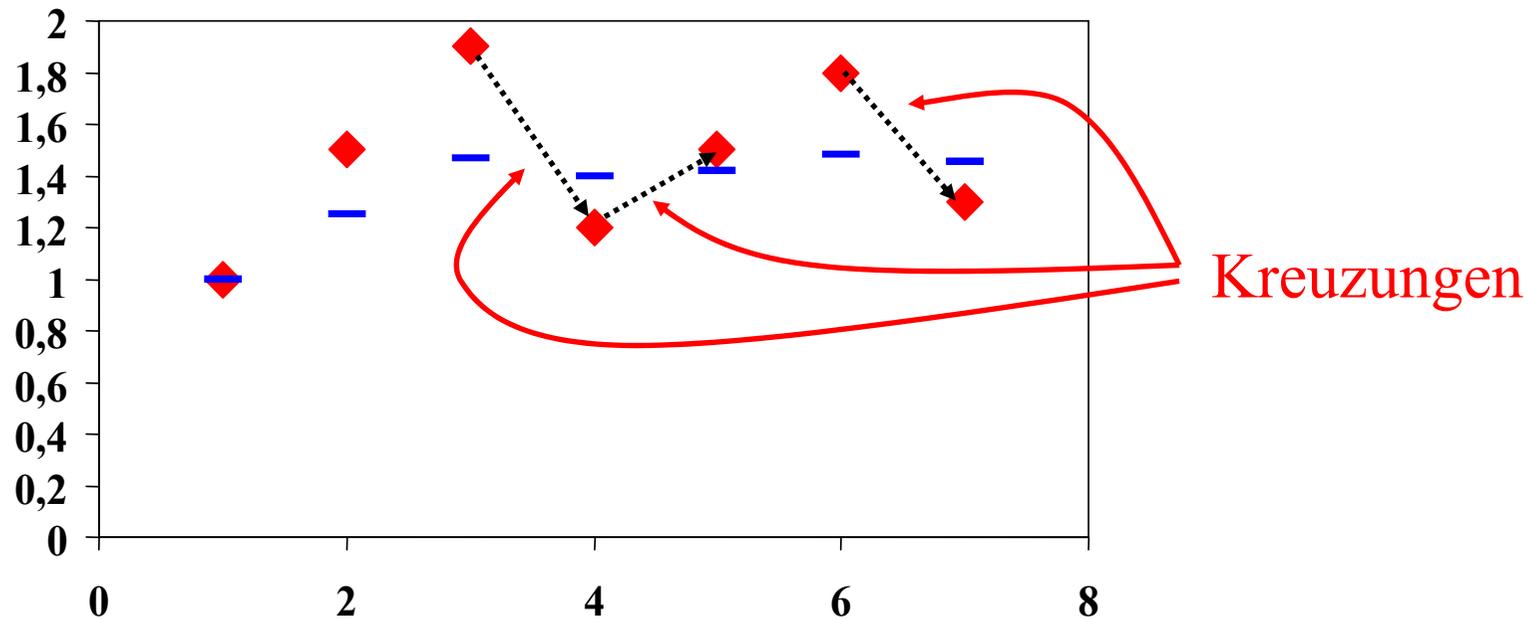
„Crossing the mean“ Regel (1973):

$$\bar{Y}_k = \frac{1}{k} \sum_{j=1}^k y_j \text{ Mittelwert der ersten } k \text{ Werte}$$

Wähle  $k$  so, dass  $\bar{Y}_k$  von den Werten  $(y_1, \dots, y_k)$   $K$ -mal gekreuzt wurde ( $K$  zwischen 3 und 5 wählen)

Eine Kreuzung liegt vor, wenn

$$y_{k-1} < \bar{Y}_k < y_{k+1} \text{ oder } y_{k-1} > \bar{Y}_k > y_{k+1}$$



## „Batch means“ Methode (1982)

- Unterteile Stichprobe in Gruppen (batches) der Größe  $i$  ( $i = 5, 10, \dots, n$ )
- Berechne Gruppenmittelwerte  $\bar{Y}_j(i) = \frac{1}{i} \sum_{k=(j-1)\cdot i+1}^{j\cdot i} y_k$
- Berechne Gesamtmittelwert  $\bar{Y}(i) = \frac{1}{m} \sum_{k=1}^m \bar{Y}_k(i)$  (m Anzahl batches)
- Mit wachsendem  $i$  (steigender batch-Größe), schwindet der Einfluss der ersten Beobachtungen
- Falls Sequenz  $\bar{Y}_2(i), \dots, \bar{Y}_m(i)$  keinen Trend erkennen lässt, werden die ersten  $i$  Beobachtungen gelöscht

### Weitere Alternative

- Pilotläufe (u.U. mit unterschiedlichen Anfangszuständen)
- Schätzung von  $t_s$  durch Vergleich der Replikationen (kann sehr aufwändig sein)

Hinweise zur Wahl von  $s_0$ :  $t_s$  hängt von  $s_0$  ab, z.B.

- Bei einem hoch ausgelasteten System wird es länger dauern bis der stationäre Zustand erreicht, wenn mit leeren Puffern gestartet wird, als wenn mit gefüllten Puffern gestartet wird,
- Bei einem System mit niedriger Auslastung ist es gerade umgekehrt

Wie also  $s_0$  wählen? (Viele Heuristiken in der Literatur)

- Starte im Zustand „empty and idle“
  - Begründung unklar, nicht immer definiert, aber einfacher Standardfall
- Starte „at steady state mode“  
(also Zustand mit größter stationärer Wahrscheinlichkeit)
  - sehr typischer Zustand, aber a priori leider unbekannt
- Starte „at steady state mean“ (also mit mittlerer Belegung)
  - Motivation und Probleme dito, oft nicht definiert  
z.B. Population 3.71

Pragmatische Vorgehensweise: Wähle einfach implementierbaren und erreichbaren Zustand

# Unabhängigkeit der Beobachtungen

Methoden zur Berechnung von Konfidenzintervallen beruhen alle auf der Annahme unabhängiger und identisch verteilter Beobachtungen

Bisherige Analyse untersuchte  $t_s$

- Unterteilung  $[0, t_s)$  transiente Phase,  $[t_s, \infty]$  bzw.  $[t_s, t_E]$  stationäre Phase
- Datenerhebung erst in der stationären Phase
- Implizite Annahme hier: Werte, die nach  $t_s$  erhoben wurden stammen aus einer Verteilung

Damit identische Verteilung sichergestellt, aber keine Aussagen über Unabhängigkeit

Alternativen der Datenerhebung:

1.  $n$  Simulationsläufe mit je  $m$  Beobachtungswerten
2. ein Simulationslauf mit  $n \cdot m$  Beobachtungswerten

Identischer Aufwand, aber im ersten Fall  $n$  statt einer transienten Phase

⇒ Ein einzelner Lauf liefert weniger verwertbare Daten

## Zur Erinnerung:

Seien  $X$  und  $Y$  ZV

- $X$  und  $Y$  unabhängig  $\Leftrightarrow P[X \leq x, Y \leq y] = P[X \leq x] \cdot P[Y \leq y]$
- $C(X, Y) = E((X - E(X)) \cdot (Y - E(Y))) = E(X \cdot Y) - E(X) \cdot E(Y)$  Kovarianz
- $X$  und  $Y$  unabhängig  $\Rightarrow C(X, Y) = 0$   
(Umkehrung gilt nicht zwangsläufig, für uns ist  $C(X, Y) = 0$  wichtig)

Seien  $Y_1, \dots, Y_n$  identisch verteilt und  $\tilde{Y} = 1/n \cdot \sum Y_i$

Dann gilt  $\sigma^2(\tilde{Y}) = \frac{1}{n^2} \cdot \sum_{i=1}^n \sum_{j=1}^n C(Y_i, Y_j)$  mit  $C(Y_i, Y_i) = \sigma^2(Y_i)$

Bisher sind wir davon ausgegangen, dass die Kovarianzen 0 sind.  
Ist dies nicht der Fall, so ändert sich der Varianzschätzer.  
Bei positiven Kovarianzen ist die geschätzte Varianz zu klein  
 $\Rightarrow$  Die ermittelten Konfidenzintervalle sind zu schmal!!  
(nicht vorhandene Sicherheit wird suggeriert!)

## Möglichkeiten der Simulationsausführung:

1. Ein langer Lauf mit  $n \cdot m$  Beobachtungen
2.  $n$  kurze Läufe mit jeweils  $m$  Beobachtungen

Im zweiten Fall sind ZV  $Y_{ij}$  und  $Y_{kj}$  unabhängig  
(bei geeigneter Wahl der Saat des ZZ-Generators)

## Naheliegenderes Vorgehen

- Führe  $n$  Replikationen durch, wobei die Länge  $m$  so gewählt wird, dass  $y_{im}$  nach  $t_s$  erhoben wird
- Verwende  $y_{im}$  ( $i = 1, \dots, n$ ) zur Berechnung von  $\hat{Y}(n)$

## Realisierung und Erweiterungen:

- Zur Ermittlung von  $t_s$  können die vorgestellten Ansätze verwendet werden
- Bei Verwendung der batch means-Methode wird  $Y_{im}$  durch den Mittelwert der Batches 2, ...,  $m$  ersetzt
- Verfahren zur parallelen Ausführung der Replikationen und zur Schätzung von  $t_s$  von existieren

Problem des vorgestellten Vorgehens bleibt die mehrfache Wiederholung der transienten Phase und der damit verbundene Aufwand

Betrachten wir die ZVs  $Y_j (=Y_{ij})$  und  $Y_k (=Y_{ik})$  in einer Replikation

- Falls beide nach  $t_s$  erhoben wurden, so können sie als identisch verteilt angenommen werden
- Unabhängigkeit ist zumindest fragwürdig

Beispiele (sei  $j < k$ )

- $Y_j$  Wartezeit des  $j$ -ten Kunden, falls  $k$  nur wenig größer als  $j$  ist, so haben der  $j$ -te und  $k$ -te Kunde mit großer Wahrscheinlichkeit gemeinsam gewartet (positive Korrelation)
- $Y_j$  Pufferpopulation zum Zeitpunkt  $j \cdot \Delta$ , falls  $k$  nur wenig größer als  $j$  ist, so wird ein Teil der Anwesenden in beiden Beobachtungen erfasst (positive Korrelation)

Daten aus einer Replikation sind oft (positiv) korreliert

⇒ Mit den bisherigen Methoden geschätzte Varianz ist zu klein

⇒ Konfidenzintervalle sind zu optimistisch

(Abdeckung zum Teil wesentlich geringer als  $1-\alpha$ )

Methoden zur Berücksichtigung der Kovarianz

(oder Autokovarianz, da Realisierung aus einem Prozess)

nur bei weiteren Annahmen:

Prozess sei schwach stationär

- $\sigma^2(Y_i) = \sigma^2(Y) = \sigma^2$

- $C(Y_i, Y_j) = C_F(Y, |i - j|)$

d.h. Kovarianz ist nur vom Abstand der Werte abhängig

- Abstand wird je nach Maß unterschiedlich definiert

i-ter und j-ter Kunde oder Zeiten  $t_i$  und  $t_j$ , dann  $|t_i - t_j|$

$$\rho(s) = \rho(|i-j|) = C_F(Y, |i - j|) / \sigma^2(Y)$$

**Autokorrelationskoeffizient der Ordnung s**

Unter Nutzung der Autokorrelationskoeffizienten erhalten wir

$$\sigma^2(\tilde{Y}) = \frac{\tilde{S}^2}{n} \cdot \left( 1 + 2 \cdot \sum_{s=1}^{n-1} \frac{n-s}{n} \cdot \rho(s) \right)$$

Einbeziehung der Autokorrelation durch Schätzung der  $\rho(s)$

Schätzer  $\tilde{\rho}(s)$  sind zwar bekannt, aber

- relativ aufwändig zu ermitteln
- wechselseitig abhängig

Weitere Methoden, um  $\rho(s)$  zu ermitteln

- Spektralanalyse
- Schätzung autoregressiver Modelle

beide nicht „blind“ anwendbar und hier nicht weiter verfolgt

Unterschätzung der Konfidenzintervallbreite sehr gefährlich,  
da zu große Sicherheit vorgespiegelt

Ausweg unter zusätzlicher Annahme:

$\rho(s)$  bzw.  $|\rho(s)|$  ist monoton fallende Funktion mit Grenzwert 0

Also fallender Einfluss mit wachsendem Abstand der Beobachtungen

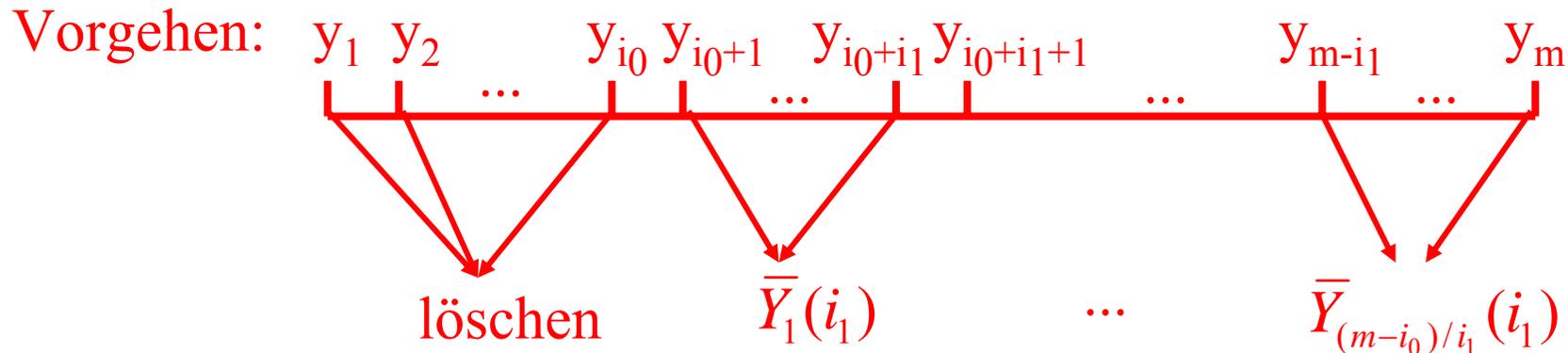
Plausibel für unsere Beispiele:

- Wartezeit  $k$ -ter Kunde wird nicht von Wartezeit des  $j$ -ten Kunden beeinflusst, wenn  $j$  lange vor  $k$  im System war
- Population zum Zeitpunkt  $k \cdot \Delta$  ist unabhängig von Population zum Zeitpunkt  $j \cdot \Delta$ , wenn  $|k \cdot \Delta - j \cdot \Delta|$  genügend groß

Falls die Annahme gilt, kann sie ausgenutzt werden, indem  $\rho(s) = 0$  gesetzt wird, falls  $s \geq s_c$  angenommen wird!

Naheliegender Ansatz „batch means“-Methode, falls die batches groß genug gewählt werden, sind ihre Mittelwerte unkorreliert!

Sei  $i_0$  Größe des ersten batches (transiente Phase) und  
 $i_1$  Größe der weiteren batches



- Bestimmung von  $i_0$  mit den beschriebenen Ansätzen/Heuristiken
- Bestimmung von  $i_1$  ebenfalls schwierig, nur Heuristiken bekannt

Einige allgemeine Ratschläge:

- Anzahl der batches nicht zu groß wählen (zwischen 10 und 30)
  - Weitere batches erhöhen Anzahl Beobachtungen
  - Größere batches sorgen für eine geringe Varianz und größere Nähe zur Normalverteilung
- Schätzung von  $\rho(s)$  auf  $\rho(1)$  beschränken:  
 Falls  $\rho(1) \approx 0$ , dann oft auch  $\rho(s) \approx 0$  ( $s > 1$ )

## Heuristische Methode

1. Ermittle die Daten aus einer Replikation und bestimme  $i_0$
2. Wähle initiales  $m \geq 10 \cdot i_0$
3. Unterteile die Daten in  $K$  ( $100 \leq K \leq 400$ ) batches und bestimme deren Mittelwerte  $\bar{Y}_k$  ( $k=1, \dots, K$ )
4. Schätze den Autokorrelationskoeffizienten der Ordnung 1:

$$\hat{\rho}(1) = \frac{\sum_{k=1}^{K-1} (\bar{Y}_k - \hat{Y}) \cdot (\bar{Y}_{k+1} - \hat{Y})}{\sum_{k=1}^{K-1} (\bar{Y}_k - \hat{Y})^2}$$

5. Teste die Autokorrelation
  - a. falls  $\hat{\rho}(1) \leq 0.2$  reduziere die Anzahl batches auf  $L$  ( $20 \leq L \leq 30$ ) durch Zusammenfassung der bisherigen Batch-Mittelwerte und bestimme das Konfidenzintervall aus einer  $t$ -Verteilung mit  $L-1$  Freiheitsgraden
  - b. andernfalls verdopple  $m$  und fahre bei 3. fort

Schätzung mehrerer Leistungsmaße aus einem Simulationslauf:

- oft sollen aus einem Simulationslauf gleichzeitig mehrere Leistungsmaße bestimmt werden  
(z.B. Verweilzeiten an verschiedenen Stationen, Verweilzeiten + Populationen etc.)
- für jedes einzelne Leistungsmaß sind Konfidenzintervalle bestimmbar (wie bereits gezeigt)

**Was kann man über die Qualität aller Resultate aussagen?**

- Sei  $\alpha_i$  das Signifikanzniveau für das  $i$ -te Leistungsmaß
- Sei  $1-\alpha$  die Wahrscheinlichkeit, dass die wahren Werte aller Leistungsmaße innerhalb der Konfidenzintervalle liegen
- Falls alle Beobachtungen unabhängig sind gilt:

$$1 - \alpha = \prod_{i=1}^k (1 - \alpha_i) \quad \text{wobei } k \text{ die Anzahl der Leistungsmaße ist}$$

- mit wachsendem  $k$  fällt die Wahrscheinlichkeit relativ schnell ab

## Ist die Annahme unabhängiger Beobachtungen über unterschiedliche Leistungsmaße realistisch?

In der Regel nein! Bsp.

- Beobachtung von Verweilzeit und Warteschlangenlänge
- Beobachtung von Verweilzeiten an aufeinander folgenden Warteschlangen
- ...

Für den allgemeinen Fall gilt  $1 - \alpha \geq 1 - \sum_{i=1}^k \alpha_i$  Bonferroni-Ungleichung

Offensichtlich erreicht man schnell den Fall  $1 - \alpha \geq 0$  (trivial)

Vorgehen (bei kleinerem  $k$ ) und vorgegebenen  $\varepsilon_i$  und  $\alpha$ :

- bestimme während der Simulation  $\alpha_i$ , so dass alle Konfidenzintervallbreiten  $\leq \varepsilon_i$
- falls  $(1 - \sum \alpha_i) \leq 1 - \alpha$  beende Simulation, ansonsten fahre mit Simulation fort

## Schätzung stationärer zyklischer Resultate

Falls der beobachtete Prozess kein stationäres, sondern zyklisches

Verhalten zeigt, so existiert eine Zykluszeit  $\Delta_C$ , so dass

$$F_{Y_i}(y) = F_{Y_{i+\Delta_C}}(y) \text{ bzw. } F_{Y_t}(y) = F_{Y_{t+\Delta_C}}(y) \quad (i \in \mathbb{N}, t \in \mathbb{R}_+)$$

Auswertung in zwei Schritten:

- Schätzung von  $\Delta_C$ 
  - Bestimmung  $t_s$  (Simulationszeit) und anschließend
  - Bildung von batches unterschiedlicher Länge und Test der batch-Mittelwerte auf Gleichheit (Testverfahren hier nicht vorgestellt)
- Schätzung von  $E(Y_{i+\Delta_C})$  und  $E(Y_{t+\Delta_C})$  (für  $i, t \in [0, \Delta_C)$ )
  - aus den Beobachtungen  $y_{i+k\Delta_C}$  und  $y_{t+k\Delta_C}$  ( $k=0, 1, \dots$ )
  - Vorgehen wie bei stationärer Analyse

## 2.6 Simulationssoftware

Basisfunktionalität für die ereignisdiskrete Simulation:

- Generierung von ZZs aus vorgegebener Verteilung
- Simulationszeitverwaltung
- Listenverarbeitung und dynamische Speicherverwaltung
- Statistische Verfahren zur Auswertung

Wünschenswert und notwendig für komplexere Probleme

- Prozessorientierte Beschreibungsmöglichkeit in Form von threads, Koroutinen etc.

Nach heutigem Stand der Technik weitergehende Forderungen

- Graphische Spezifikation (modellieren nicht programmieren)
- Graphische Ausgaben inkl. Animation
- Komponentenbibliotheken
- Integration von Methoden zur Datenmodellierung
- Experimentsteuerung

## Historische Entwicklung

- seit 1960 erste Simulationsprogramme in FORTRAN
- 1961-1965 Erweiterung von Programmiersprachen um Simulationsfunktionalität

## Beispiele

- GPSS (General Purpose Simulation System)  
IBM-Entwicklung, Blockdiagramm-Sicht auf Systeme
- SIMSCRIPT als FORTRAN-Erweiterung 1965
- GASP als ALGOL später FORTRAN-Erweiterung
- 1966-1975 Entwicklung von Simulationsprachen
  - Simula67 als erste „objektorientierte“ Sprache
  - GPSS/H als Erweiterung von GPSS
- 1976-1985 Simulation auf Mikro-Rechnern und PCs
  - SIMAN erste Sprache unter DOS auf IBM PC mit Realisierung event scheduling in FORTRAN
- seit 1986 Entwicklung integrierter Simulationsumgebungen

# Heutige Situation

Simulatoren in  
üblichen  
Programmiersprachen  
C/C++/Java u.U. unter  
Verwendung zus.  
Bibliotheken zur  
Realisierung von  
Simulations-  
funktionalität

- + Effizienz,  
Flexibilität,  
Kenntnis der  
Basissprache
- Fehleranfälligkeit  
Aufwand für  
komplexe Probleme

Verwendung  
etablierter  
Simulationssprachen  
insbes. GPSS  
manchmal auch  
SIMAN, Simula67

- + Effizienz,  
Flexibilität,
- Fehleranfälligkeit  
Aufwand für  
komplexe Probleme

Objektorientierte  
Simulationsansätze,  
als erweiterbare  
Baukästen

- + Kompatibilität zu  
heutigen  
Programmier-  
ansätzen, Flexibilität,  
Erweiterbarkeit
- Teilweise mangelnde  
Effizienz,  
Wiederverwend-  
barkeit nur  
eingeschränkt  
möglich

## Heutige Situation 2

```
graph TD; A[Heutige Situation 2] --> B[Integrierte Simulationsumgebungen für spez. Anwendungsgebiete]; A --> C[Integration von Simulationsfunktionalität in Spezifikationsansätze wie UML, SDL, PN, Prozessketten]; A --> D[Agentenorientierte Simulation, Simulation besteht aus einer Menge autonomer interagierender Agenten];
```

**Integrierte Simulationsumgebungen für spez. Anwendungsgebiete**  
+ Anwendungsnahe Modellierung, Unterstützung bei Modellerstellung, Modellverwaltung, Parametrisierung, Experimentdesign & -auswertung, Animation  
– Durch Szenario mangelnde Flexibilität, oft mangelnde Effizienz

**Integration von Simulationsfunktionalität in Spezifikationsansätze wie UML, SDL, PN, Prozessketten**

+ Simulation wird Teil der Spezifikation  
– proprietärer Ansatz, fehlende Ausdrucksfähigkeit, Qualität der verwendeten Simulatoren unklar

**Agentenorientierte Simulation, Simulation besteht aus einer Menge autonomer interagierender Agenten**  
+ Neue Möglichkeiten der Spezifikation dynamischer Abläufe mit komplexen Interaktionen  
– Auswertung und Interpretation bisher noch nicht beherrscht

Wir betrachten etwas detaillierter

- GPSS als historische Simulationsprache
- Demos als objektorientierten Ansatz mit historischen Wurzeln
- Arena als kommerzielles Werkzeug mit den Anwendungsgebieten Fertigungstechnik, Geschäftsprozesse
- HIT als universitäre Entwicklung mit den Anwendungsgebieten Rechensysteme, Rechnernetze
- Prozessketten als Erweiterung einer deskriptiven Beschreibungsform um Simulationsfunktionalität

# GPSS

Sprache+System zur Simulation warteschlangennetzartiger Modelle

- Geschichte reicht bis in die 50er Jahre zurück
  - Entwicklungsschritte GPS/GPSS/GPSS II/ III .., IBM geprägt
- Trotz Alter (und Steinzeit-Erscheinungsbild) immer noch akzeptiert und weit verbreitet/genutzt

Heute in unterschiedlichen Versionen verfügbar

- Mit z.T. großen Unterschieden, aber identischer Weltsicht (i.d.R. material-orientiert)
  - Einheiten (hier transactions) bewegen sich über
  - Ressourcen (hier equipment entities)
- Modellspezifikation
  - Beschreibung verschiedener Verkehrsmuster (transactions)
  - Generierung neuer Objekte im Objekt selbst
  - Benötigte Ressourcen werden in Verkehrsobjekten genannt und nicht eigens deklariert

GPSS als Sprache stark von Lochkarten-/Flussdiagrammen geprägt

Entwurf von transactions empfohlen in Form

- bestimmten Typs von Flussdiagrammen
- bestehend aus (unterschiedlichen) Kästchen blocks und
- Pfeilen zur Darstellung der Kontrollflussrichtung

Flussdiagramme ursprünglich als „Abbild eines Lochkartenstapels“  
gedacht, heute initiale Form einer graphischen  
Modellspezifikation

Jede GPSS-Version definiert durch

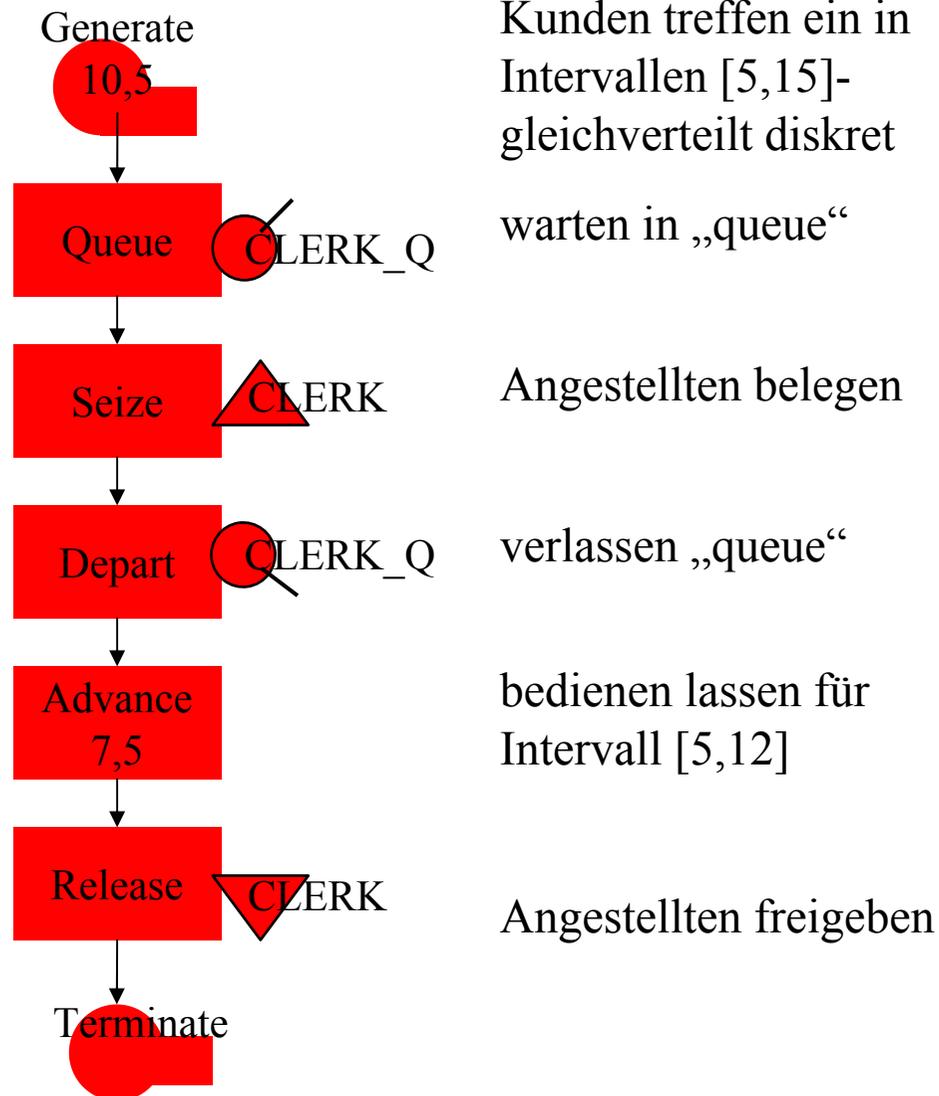
- ihre entities (Einheiten)
- deren attributes (Eigenschaften: Konstanten, Variablen)

Jedes entity (Typ) besitzt (per default) charakteristische attributes

- die im Verlauf der Simulation automatisch aktualisiert werden
- auf deren Werte zugegriffen werden kann

Identifikation von entities über Nummern ,  
von attributes über Standardnamen

# Beispiel Bankschalter



- Generelles wait\_until wird ersetzt durch warten auf Ressource
- Implizite Deklaration von CLERK und CLERK\_Q als Bediener und Warteschlange explizite Belegung-Freigabe von Bedienern durch SEIZE und RELEASE
- Zusätzlich vorhanden Strukturen zur Datensammlung (queues (!), frequency table)

# GPSS-Programm

SIMULATE

GENERATE 10,5

QUEUE 1

SEIZE 1

DEPART 1

ADVANCE 7,5

RELEASE 1

TERMINATE 1

START 500

END

Verweis auf  
Ressource #1

Start +  
Dauer  
Simulation

- Einfache Notierbarkeit einfacher Modelle
  - Automatische Statistiken
  - Gute Debugging-Hilfen
  - Wenn Compiler vorhanden hinreichend schnell (oft nur interpretiert)
  - Echte Ein-/Ausgabe erst in neueren Versionen (vorher Umweg über FORTRAN)
  - GPSS/H mit Anbindung an Animationstool PROOF
- Insgesamt Konzepte überholt!

# DEMOS

DEMOS (Discrete Event Modeling On Simula) wurde 1979 von

Birwistle als spezielle Simula-Klasse entwickelt

Im Gegensatz zu vielen anderen Szenariosprachen

- werden Szenario-spezifische Konzepte verfügbar gemacht
- ohne auf moderne Sprachkonzepte zu verzichten  
(durch Übernahme des Klassenkonzepts aus Simula)

DEMOS wird allgemein relativ gut bewertet und wird an

zahlreichen Universitäten in der Lehre eingesetzt

Aber Simula heute nur noch wenig verbreitet, deshalb

- Realisierung von Demos auf Basis anderer Sprachen  
(Modula, C++, Java)
- interessant insbesondere Java-Realisierungen  
(Unis Essen, Hamburg)

## Java-Demos vereint

- Konzepte von Demos
- Möglichkeiten der Sprache Java

Demos erlaubt (wie GPSS) graphische Darstellung der Modellbeschreibung (explizit als „Spezifikation“)

Konkret: (eine Form von) activity-diagrams  
(existieren auch in anderen Bereichen in verschiedenen Formen)

## Prinzip activity-diagram

- dynamische Einheiten (entities)
  - verfolgen einen Plan / ein Muster über Aktivitäten
- zur Ausführung von Aktivitäten werden passive Einheiten / Ressourcen (Resources) benötigt,
  - die nur in begrenzter Anzahl vorhanden sind, so dass auf sie gewartet werden muss

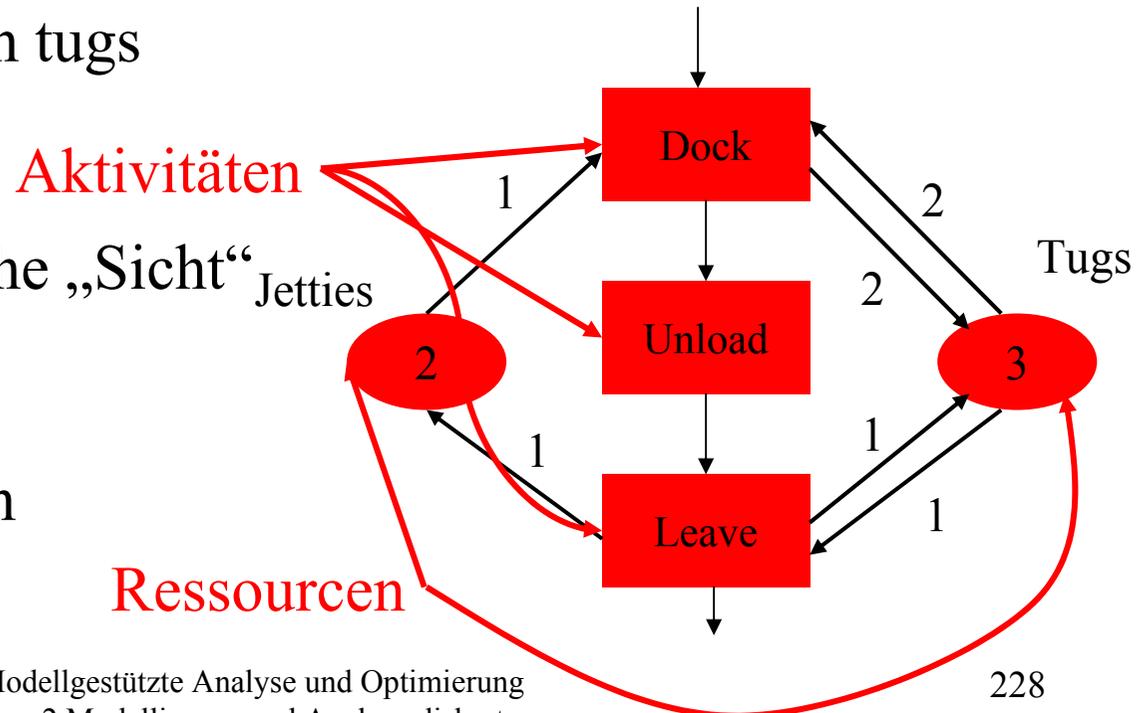
Zu Demos existieren zahlreiche Standard-Beispiele

Eines davon Hafen und Schiffe (hier in der einfachsten Form)

- Mit gewissen zeitlichen Abständen fahren Schiffe zum Entladen einen Hafen an
- Zur Entladung wird ein Pier (jetty) benötigt, zum Einlaufen sind zwei Schlepper (tugs), zum Auslaufen ein Schlepper notwendig
- Einlaufen, Entladen, Auslaufen dauern spezifische Zeiten
- Der Hafen weist eine begrenzte Zahl von jetties auf und verfügt über begrenzte Zahl von tugs

Auch hier wieder die gleiche „Sicht“

- Transaktionen
  - Folge von Aktivitäten
  - Nutzung von Ressourcen
- material-orientiert



## Demos-Programm

```
public class Harbour extends Entity {
```

```
  Res tugs, jetties ;
```

```
  public class boat extends Entity {
```

```
    public void body() {
```

```
      new boat().schedule(10.0) ;
```

```
      jetties.acquire(1) ;
```

```
      tugs.acquire(2) ;
```

```
      hold(1.0) ;
```

```
      tugs.release(2) ;
```

```
      hold(5.0) ;
```

```
      tugs.acquire(1) ;
```

```
      hold(1.0) ;
```

```
      tugs.release(1) ;
```

```
      jetties.release(1) ; } } // End boat
```

```
... // Methode body mit Deklartionen der Ressourcen
```

Deklaration Ressourcen

Klasse boat zur  
Beschreibung der  
Entladung eines Schiffs

Methode body wird in  
der Simulation  
ausgeführt und muss  
vorhanden sein

Erzeugung des nächsten  
Bootes zum Zeitpunkt  
jetzt+10.0

Belegung & Freigabe von  
Ressourcen

Zeitverzögerung

## Auch bei Demos

- ist die tatsächliche Nutzung der Graphik zur Modellspezifikation die Ausnahme
- lassen sich nicht alle programmiersprachlichen Konstrukte per Graphik beschreiben (Abhilfe code-nodes)

Graphik ist trotzdem hilfreich

- als Erklärungsinstrument
- zum Programmentwurf

## Demos

- benutzt konsequent das Klassenkonzept von Java (d.h. Demos ist selbst eine Klasse)
- erlaubt die Nutzung aller in Java vorhandenen Methoden
- beinhaltet eine Reihe von Unterklassen, in denen jeweils Datenstrukturen und Methoden für die Durchführung von Simulationen bereitgestellt werden
- beinhaltet zusätzlich einige Methoden zum geordneten Zugriff auf globale Datenstrukturen

# Einige wesentliche Komponenten aus der Demos-Welt

## **Entities**

- sind handelnde, aktive Einheiten
- bilden Prozesse
- konkurrieren um Betriebsmittel (Ressourcen)
- können zusammenarbeiten
- können sich gegenseitig unterbrechen
- können dynamisch erzeugt und zerstört werden
- können mittels schedule für einen Zeitpunkt zur Aktivierung vorgemerkt werden
- existieren für eine (Modell)-Zeit im Modell
- können mittels hold Zeitverbräuche modellieren
- werden durch Java-threads realisiert
- können einen Namen haben
- können Parameter und lokale Variablen haben

## Ressourcen res und bin

- beschreiben eine (bei Initialisierung festzulegende) Anzahl ununterscheidbarer Ressourcen
- können belegt und freigegeben werden
- blockieren Prozesse falls diese zu viele Ressourcen belegen oder zurückgeben (nur res) wollen
  - Implizite Warteschlange, so dass Prozesse automatisch deblockiert werden
- Keine direkte Bindung von Ressourcen an Prozesse

## Synchronisation über waitq und condq (auch hier kein wait\_until)

- **waitq** master-slave Synchronisation mit zwei Warteschlangen (WS)
  - jeweils zwei Prozesse (einer pro WS) synchronisieren
  - master-Prozess bestimmt den weiteren Ablauf bis zur expliziten Freigabe des slave-Prozesses
- **condq** erlaubt beliebige Synchronisationen
  - Prozesse können eine allgemeine Bedingung formulieren und bleiben in der condq, bis diese erfüllt ist
  - Bedingungen werden nur überprüft, wenn
    - sich ein Prozess in die condq einordnet (nur für diesen Prozess)
    - ein Prozess ein signal der condq initiiert (für alle Prozesse in der WS)

# Arena

Umfangreiche Simulationsumgebung für Produktionssysteme und Geschäftsprozesse

- Modelle werden graphisch spezifiziert und
- in SIMAN-Programme umgewandelt

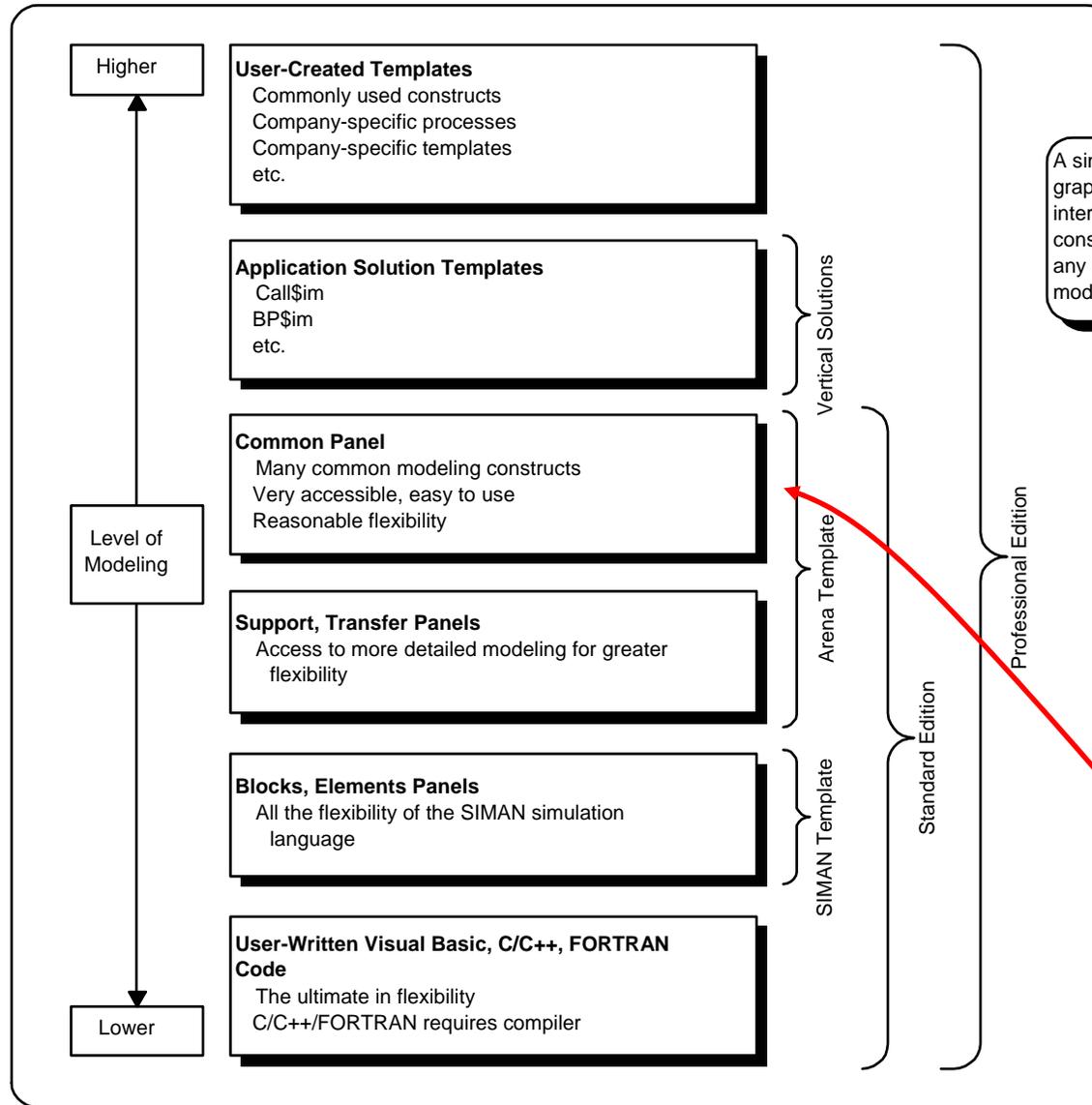
Bausteinorientierter und modularer Modellaufbau

- Vordefinierte Bausteine
- Möglichkeit der Definition von templates für spezielle Anwendungen
- Komplexe Strategien erfordern u.U. Rückgriff auf SIMAN und Verwendung globaler Variablen

Arena ist ein kommerzielles Werkzeug

- Freie Studentenversion  
(mit eingeschränkter Funktionalität, Modellgröße)

# Modellierungsebenen in Arena



- Mischung der Ebenen in einem Modell ist möglich
- Üblicherweise Modellierung im common panel mit einer Art von Flussdiagrammen

Modellspezifikation Mischung aus graphischer Beschreibung und Menüeingaben

Möglichkeit der Definition verschiedener Sichten auf ein Modell inkl. Animationssicht (2D, neuere Version auch 3D)

Zahlreich Zusatzwerkzeuge

- input analyzer zur Modellierung von Eingabedaten
  - Anpassung unterschiedlicher Verteilungen inkl. Parameterschätzung
  - Ausführung von Anpassungstests (inkl. K.S. und  $\chi^2$  – Test)
  - Direkte Übertragung der Verteilungen in eine Modell inkl. empirischer Verteilungen
- output analyzer zur Analyse von Simulationsresultaten
  - Konfidenzintervallberechnung (auch online während der Simulation)
  - Schätzung Autokorrelationskoeffizienten
  - Graphische Aufbereitung

# HIT

Universitäre Entwicklung, die auch in der Industrie eingesetzt wird  
(Siemens, ESA, ..)

Einsatzgebiete Rechensysteme, Rechnernetze aber auch in anderen  
Bereichen wie Materialflusssysteme, Logistik

Auch hier wieder hin zur Modellierungsumgebung mit

- Modelldatenbank
- Experimentsteuerung
- gemischt graphisch/textueller Eingabeschnittstelle
- hierarchischer/modularer Modellaufbau
- statistischer Auswertung

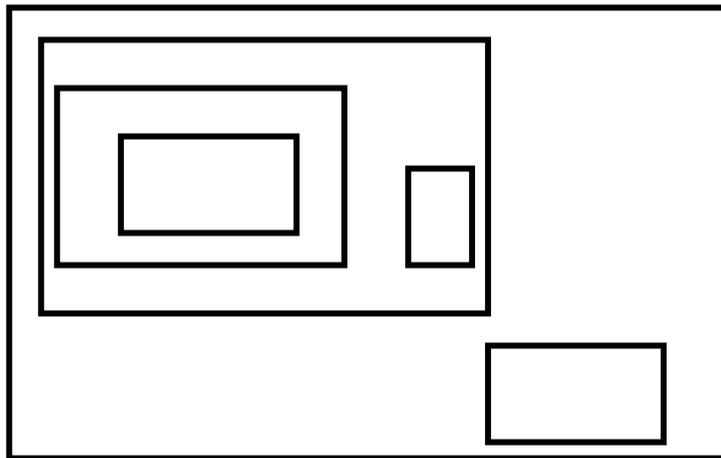
Aber zusätzlich (Forschungsaspekt!)

- konsequente Umsetzung eines hierarchischen Modellaufbaus durch das Konzept der virtuellen Maschine
- Integration weiterer Analysetechniken (neben Simulation)

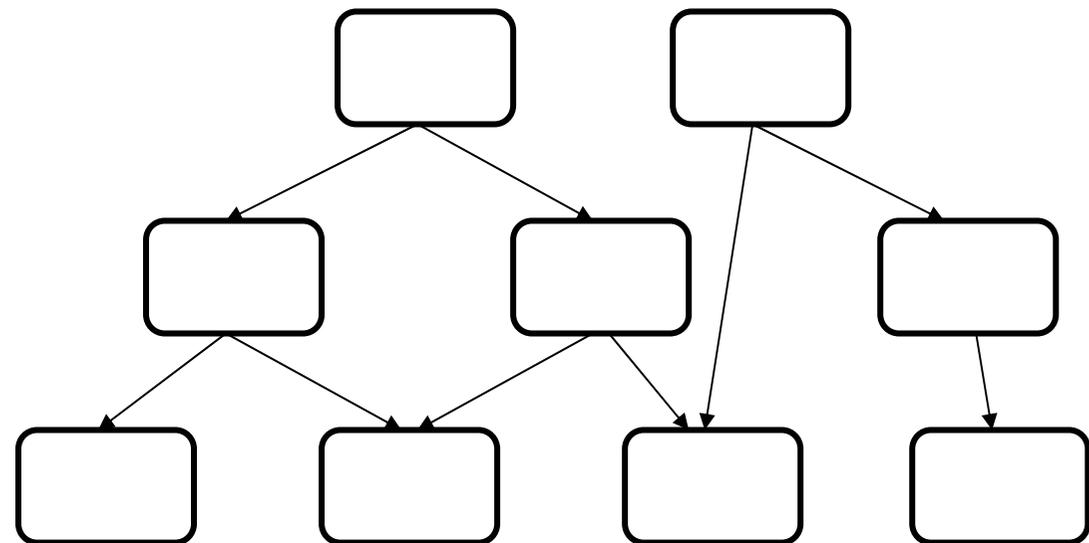
# Gängige Strukturierungsprinzipien in der Informatik

- Modularisierung (Abschottung, Autonomie)
- Hierarchisierung (unterschiedliche Möglichkeiten)  
hier insbesondere Hierarchie virtueller Maschinen
  - obere Ebene benutzt Dienste
  - untere Ebene bietet Dienste

Enthaltensein



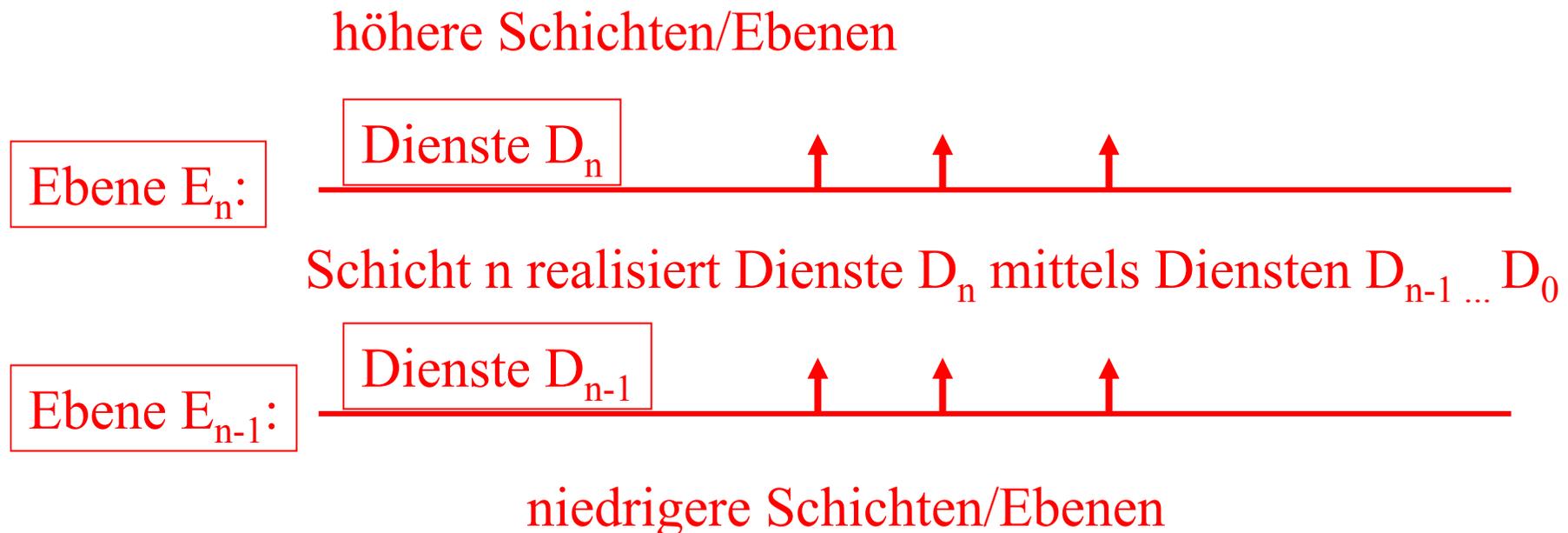
Benutzung/Aufruf



azyklischer Graph der Aufrufe

## Prozessorientierung:

- Prozessmuster: (Halb-)Ordnung von Aktivitäten
- Aktivität: lokale Aktion oder Benutzung des Dienstes einer Ressource
- Prozess: instantiiertes Prozessmuster
- durchgängige Last/Maschine-Sicht (üblich in der Informatik)  
virtuelle Maschinen exportieren Dienste an Andere und importieren Dienste von Anderen



## Globale Sicht: Hierarchie virtueller Maschinen

- oberste Schicht: Umgebungsdynamik
- unterste Schicht Standard-Komponenten (Raum-, Zeit-Verbrauch)

Beschreibung von Prozessmuster  
programmiersprachliche Form  
mit zusätzlichen Konstrukten

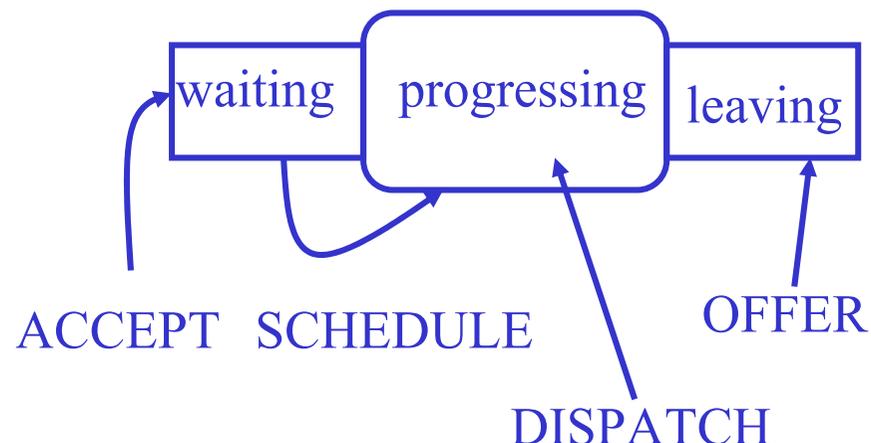
- zum Ausdruck von Parallelität
- zur Darstellung von Zeitverzögerungen
- zur Nutzung von Diensten

➤ HI-SLANG

Standardkomponenten

- Server (Zeitverbrauch)
- Counter (Raumverbrauch)

Komponenten als autonome  
Objekte mit parametrisierbarer  
Steuerung durch  
Kontrollprozeduren



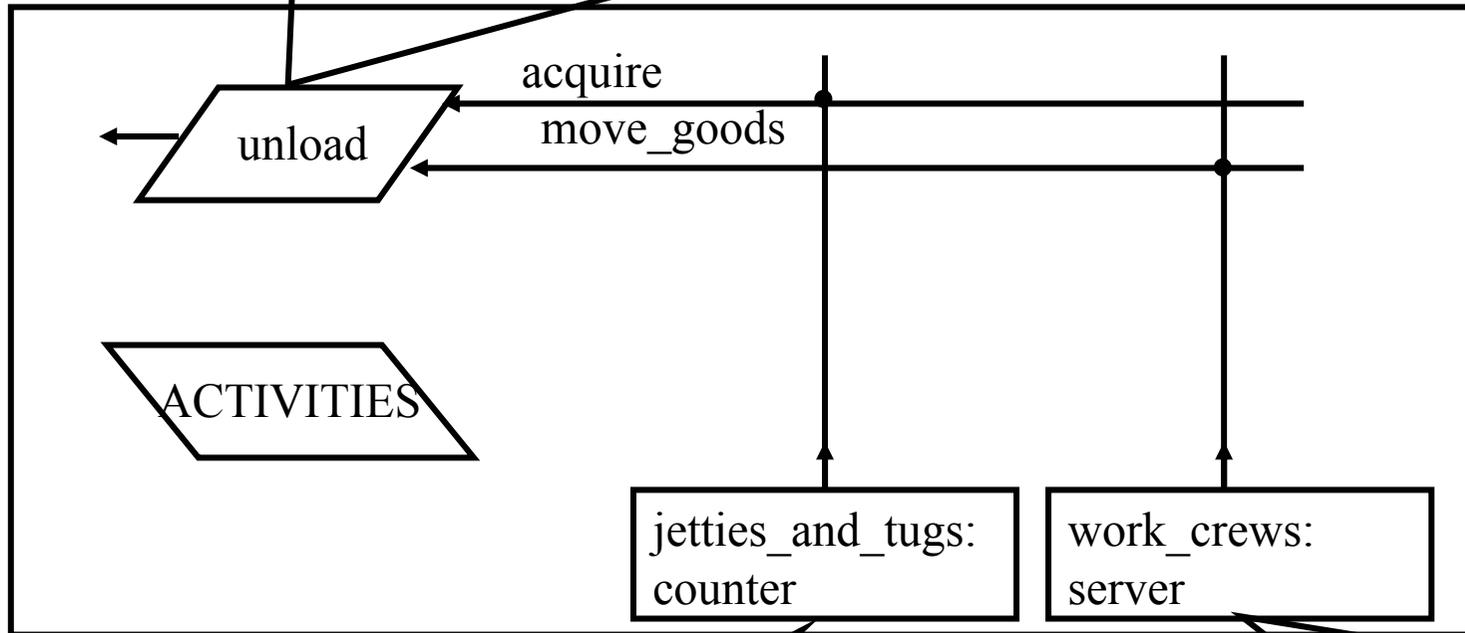
Für alle 4 Typen sinnvolle defaults  
vorhanden oder in HI-SLANG

programmierbar

# Gemischt graphisch – textuelle Beschreibung

```

acquire([-1,-2]) ;   move_goods(UNIFORM(3,7)) ;   acquire([1,1]) ;
HOLD(1.0) ;         acquire([0,-1]) ;
acquire([0,2]) ;   HOLD(1.0) ;
    
```

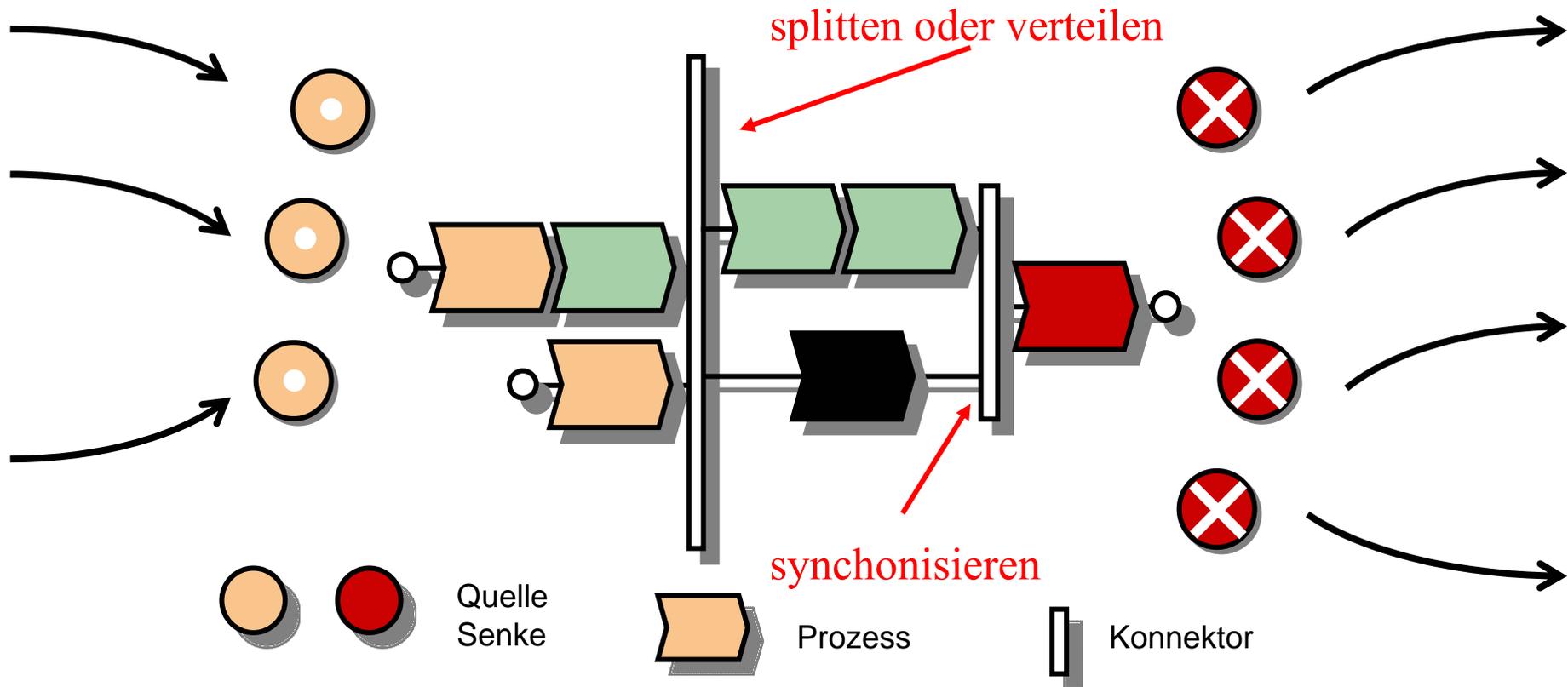


parametrisiert mit  
 min = [0,0] ;  
 max = [2,3] ;  
 int = [2,3] ;

control durch  
 ACCEPT = ALWAYS ;  
 SCHEDULE = IMMEDIATE ;  
 DISPATCH = SHARED(SPEED = 2.0)

# Prozessketten

Ursprünglich deskriptives Modell zur Spezifikation von Abläufen  
(Bearbeitungsprozessen, Workflows, Transportprozessen, ...)



## Hierarchische und modulare Beschreibung

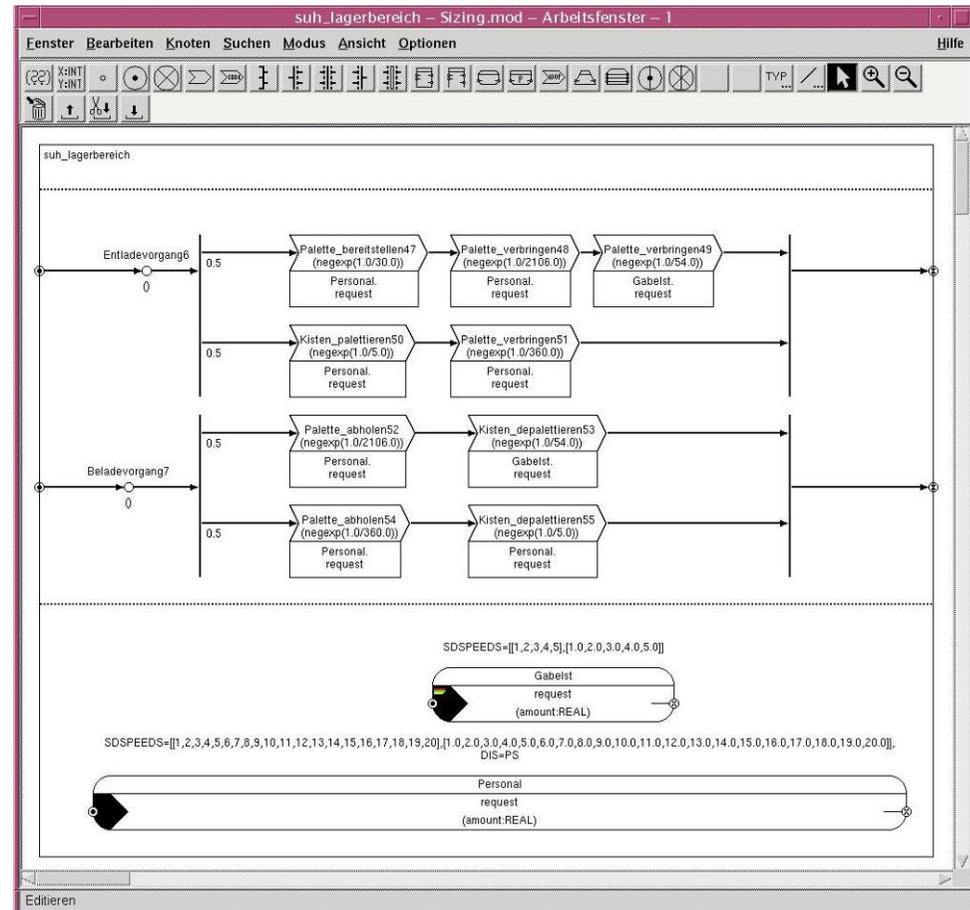
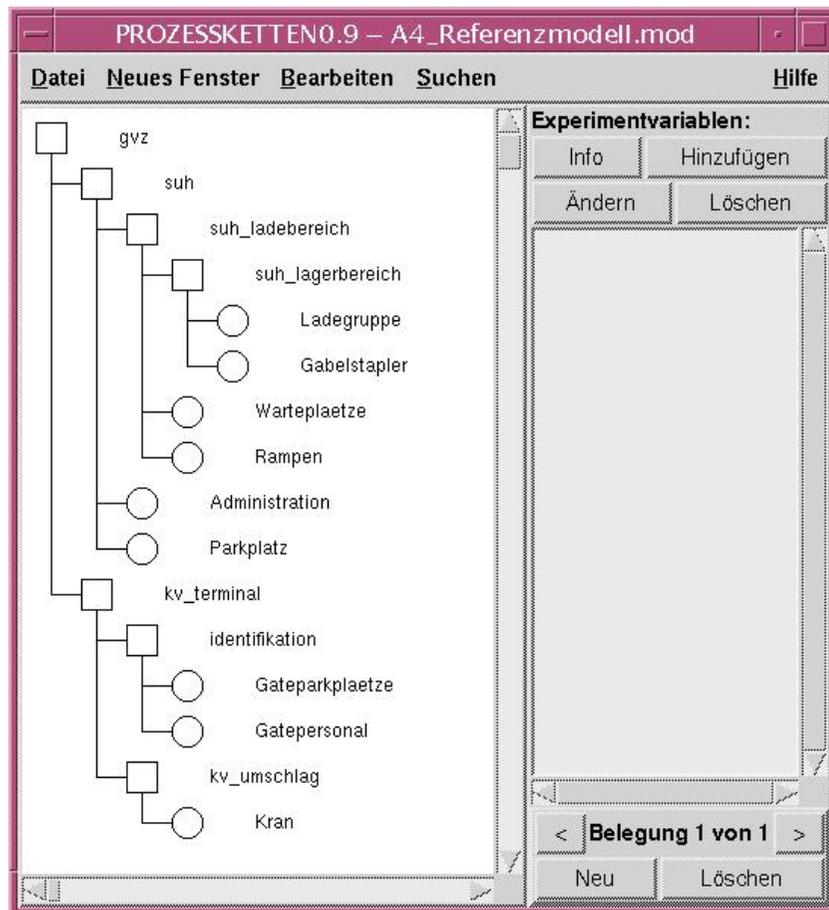
- Prozesselemente können Prozessketten enthalten
  - Bindung der Ebenen über Quellen/Senken
- Prozessketten können sequentiell komponiert werden durch Bindung der Senken der ersten Prozesskette an die Quellen der zweiten

## Zusätzliche Informationen zur Erstellung ablauffähiger Modelle

- Generierungszeiten von Prozessen
- Quantifizierung von Entscheidungen
- Ressourcennutzung und zugehörige Ressourcen
  - Auch hier wieder Bindung von Aktivitäten an Maschinen
    - Maschinen können elementarer Natur sein oder
    - virtuelle Maschinen definiert als Prozessketten
- Zur Resultatmessung müssen Maße auf Basis von Aktivitäten definiert werden

# Beispiel einer hierarchischen Beschreibung

- Links hierarchische Struktur
- Rechts ein Ausschnitt aus der Hierarchie



## 2.7 Möglichkeiten und Grenzen der Simulation

### Vorteile der Simulation

- Viele reale Systeme können nicht akkurat in einem analytischen Modell beschrieben werden
- Simulation kann Systeme unter unterschiedlichen (realen oder unrealen) Bedingungen analysieren
- Modifikationen am System sind einfach durchführbar
- Experimente können in der Simulation einfach gesteuert und ausgeführt werden
- Simulation erlaubt es, Systeme in sehr kurzen oder sehr langen Zeitintervallen zu analysieren, die in der Realität nicht beobachtbar sind
- Durch heutige Rechnerkapazität lassen sich ohne großen Aufwand viele Experimente parallel durchführen

## Nachteile der Simulation

- Stochastische Simulation produziert stochastische Ausgaben
  - damit können Größen nur geschätzt werden
  - Genauigkeit der Schätzung basiert auf kaum beweisbaren Annahmen
- Simulationsmodelle haben einen hohen Datenbedarf
- Simulationsmodelle sind aufwändig und teuer in der Entwicklung
  - neben den üblichen Problemen bei der Realisierung komplexer Programme
  - existieren simulationsspezifische Probleme, die vom dynamischen Verhalten abhängen
- Simulationsmodelle liefern extreme Datenmengen, so dass die Genauigkeit und Gültigkeit der Resultate oft nicht überprüft oder überbewertet wird

## Fehler bei der Anwendung der Simulation

- Modellerstellung ohne konkrete Zielsetzung
- Falscher Detaillierungsgrad des Modells
  - oft Abbildung unnötiger Details
- Aufwand für Datenerhebung und Validierung wird unterschätzt
  - Einsatz von Mittelwerten statt Verteilungen
  - keine Validierung (oft nicht einmal Plausibilitätsprüfung) der Resultate
- Überbewertung von Animation
- Kein oder zu geringer Einsatz statistischer Methoden
  - falsche Modellierung der Eingabedaten
  - keine Auswertung der Resultate (nur Mittelwertbildung)
- Unabhängigkeitsannahme bei Ein- und Ausgabe
- Keine Sensitivitätsanalyse
- Keine Abschätzung des Gültigkeitsbereichs

## 2.8 Validierung von Modellen

Zur Erinnerung:

Modelle werden erstellt,

- um Experimente mit realen Systemen zu vermeiden
- um Aussagen über (potenzielle) Objekt-Systeme zu erhalten, die (noch) nicht existieren

Folgerungen aus der Modellanalyse

(hier aus Simulationsexperimenten) sollten weitestgehend gleich sein zu Folgerungen, die aus entsprechenden (potenziell irgendwann möglichen) Objekt-Analysen/-Experimenten gewonnen würden

⇒ Es muss zumindest plausibel gemacht werden, dass gleiche/ähnliche Folgerungen aus Modell- und System-Experiment gezogen werden

Gleiche Beobachtungsschemata vorausgesetzt, sind Folgerungen (zumindest) dann identisch, wenn unmittelbare Beobachtungen/Resultate von Simulations- und Objekt-Experimenten identisch sind!

Ist Identität der Resultate zu erwarten?

Sicher nicht immer, nicht allgemein, nicht in jedem Detail

Objekt- und Modell-System sind nicht identisch,

Objekt- und Modell-System zeigen stochastisches Verhalten

⇒ Verhaltensunterschiede sind zu erwarten

Zentrale Frage:

Welche Verhaltensunterschiede sind tolerierbar?

## Grundlegende Definitionen

Benötigt wird eine vernünftige Definition von

**Realitätstreue, Gültigkeit, validity**

Aufbauend auf einem Maß für Verhaltenunterschiede

$$D(V_R, V_S) \quad (\text{R: real, S: simuliert})$$

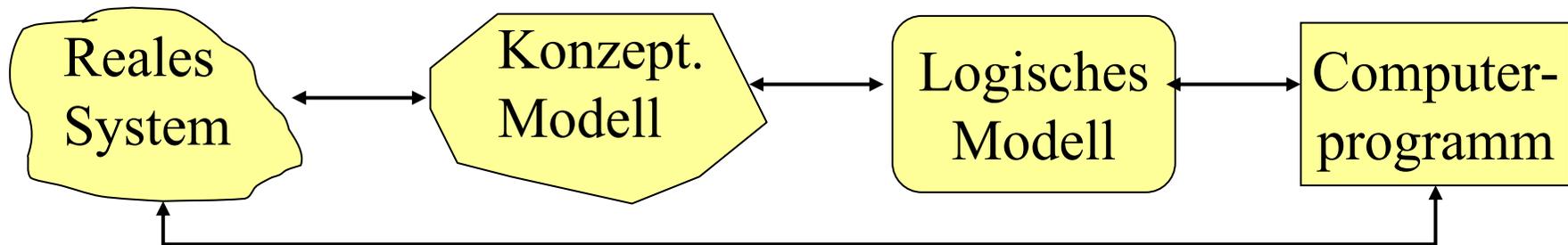
Eine solche Definition müsste

- Realitätstreue bejahen, falls Abweichungsmaß unter einem bestimmten, festzulegendem (problemabhängigem, Fragestellung-abhängigem) Grenzwert liegt
- den Grenzwert so festlegen, dass (unvermeidlich existierende) Verhaltensunterschiede, die zu treffende Folgerungen nicht beeinflussen (d.h. nicht zu unterschiedlichen Folgerungen führen) toleriert werden

## Ursachen für Verhaltenunterschiede

- Strukturelle Unterschiede
  - Abstraktion, Aggregation (beabsichtigt)
  - Ungenauigkeiten (bugs) (unbeabsichtigt)
  - Fehler (faults) (unbeabsichtigt)
- Parameterwerte
  - Deterministische Werte + Abhängigkeiten
  - Verteilungsfunktionen
- dazu stochastische Schwankungen

## Transformationsschritte im Rahmen der Modellbildung:



Transformationsprozess von vage definierter Problemstellung  
(schlecht strukturiertem Realsystem)  
zu wohl-definiertem Computerprogramm

Jeder angegeben Transformationsschritt kann zu neuen  
Fehlern / Irrtümern / Verzerrungen  
führen!

Oftmals uneinheitliche Terminologie, wir unterscheiden hier:

- Verifikation  
Bestätigung aller Modell-Eingabegrößen/-Annahmen inkl. Struktureller Annahmen, Programmverifikation, Parameterwerte (mittels Testverfahren), ...
- Validierung  
Bestätigung der Modellresultate
- Kalibrierung  
Reduktion der Verhaltensunterschiede  
(d.h. Reduktion von  $D(V_R, V_S)$ )  
falls Unterschiede als zu groß bewertet wurden  
Anpassung i.a. durch Änderungen am Modell

Auch wenn große Mühe auf eine „gute“ Wahl der Eingangsgrößen gelegt wurde (d.h. Modellierung der Eingabegrößen, Testverfahren, ...), steigt zwar die „Hoffnung“ auf ein gültiges Modell, eine „Garantie für ein gültiges Modell ist aber nicht gegeben!

Positivistischer Blick: Ergebnisse ok, alles ok reicht oft nicht aus!

⇒ Kalibrierung ohne Verifikation / Validierung ist gefährlich! (wird aber leider oft bei induktiver Modellierung verwendet)

In allen Schritten Beachtung des Kosten/Nutzen-Aspektes:

- Nutzen steigt mit dem Erkenntniswert
- Kosten steigen mit dem Aufwand der Erstellung

➤ Kompromiss zwischen Resultat und Aufwand ist notwendig

# Verifikation

Oft Einsatz von (semi-)formalen/automatischen Techniken

Ein wichtiger Aspekt:

Verifikation des Simulationsprogramms am logischen/konzeptuellen Modelle (also Programmverifikation) hier nicht weiter behandelt!

Weitere Ansätze und Aspekte:

Zum Programmentwurf / zur Programmerstellung:

- Strukturierte Programmierung mit Test/Debugging von Modulen/Subprogrammen
- Code-review durch andere Mitarbeiter
- Bei Verwendung von Spezifikationstechniken (semi-) automatische Generierung von Programmcode
- Inkrementeller Entwurf und Test

## Simulationsspezifischere Ansätze:

- Test des Simulators für unterschiedliche Eingabeparameter (unterschiedliche Umgebungsbedingungen)
- Erstellen und analysieren von traces
- Simulationsläufe unter vereinfachenden Annahmen (z.B. on ZVs), so dass Verhalten vorhersagbar
- „Durchspielen“ typischer Anwendungen
- Beobachtung von Animationen
- Verwendung von zuverlässigen Simulationswerkzeugen
- Verifikation der Eingabegrößen  
i.d.R. durch statistische Testverfahren  
(Verwendung des Begriffs Verifikation ist an dieser Stelle umstritten!)

## Allgemeine Überlegungen zur Validierung

Simulationsmodell soll nützlich sein, d.h. mit sinnvoller Genauigkeit Aussagen über das System erlauben

⇒ **Es gibt keine absolute Validität von Simulatoren!**

Zu beachten ist

- Validierung ist modell-individuell
- Validierung ist graduell
- Validität ist oft Ergebnis eines (Verhandlungs-) Prozesses
- Validierung ist ein Projekt-begleitender Prozess

Man unterscheidet:

- Funktionsbezogene Validierung  
Test auf Plausibilität
- Ergebnisbezogene Validierung  
„Übereinstimmung“ der Ergebnisse System / Modell
- Theoriebezogene Validierung  
„Übereinstimmung“ der Ergebnisse analytisches /  
simulatives Modell

Wir betrachten im Folgenden primär die ergebnisbezogene  
Kalibrierung und Validierung !

Einsatz von Kalibrierung und Validierung setzt voraus, dass Daten für das Realsystem vorhanden sind oder ermittelt werden können, damit  $V_R$  bestimmbar ist

- Daten resultieren aus Messungen am Realsystem
  - Daten sind oft „gestört“ oder müssen aufgearbeitet werden
  - Daten können nur in aggregierter Form oder für nicht relevante Situationen gewonnen werden
  - ⇒ Probleme ähnlich zur Datenermittlung zur Spezifikation quantitativer Modellgrößen (dort vorgestellte Ansätze auch hier verwenden)
- Daten resultieren aus anderen Systemen, anderen Modellen, Schätzungen, Vorhersagen
  - Im Prinzip ähnlich wie im ersten Fall, aber oft noch ungenauer

## Schritte zur Kalibrierung von Modellen

- Identifikation der Ursachen von Verhaltensunterschieden
- entsprechende, gezielte Änderungen am Modell
  - Struktur, d.h. Code-Änderungen
  - Parameter, d.h. Werte-Änderungen

Unterstützung beim Finden der Ursachen:

- Strukturelle Ungenauigkeiten / Fehler sind meistens aus qualitativen Abläufen (Traces, Animationen) ableitbar
- Parameter Ungenauigkeiten / Fehler erfordern den Vergleich quantitativer Größen  
hilfreich zur Eingrenzung, Vergleich von Zwischenergebnissen  
z.B: Verweilzeit an einer Station statt Gesamtverweilzeit

- Bei stochastischen Modellen zwei spezifische Testprobleme
- Auswahl eines aus mehreren Modellen ( $S_1, S_2, \dots, S_K$ ) auf Basis der zugehörigen  $D(V_R, V_{S_k})$  ( $k = 1, \dots, K$ )
  - $D$  (wie gewohnt) Zufallsvariable oder stochastischer Prozess
  - Unterschiede der  $D$ 's müssen zur Auswahl einer Konfiguration „das normale Schwankungsmaß“ übersteigen  
 $\Rightarrow$  Unterschiede müssen signifikant sein  
 Aufgabe: **Test auf Signifikanz**
- „Tuning“ durch Parameterveränderungen, also Suche nach Parametervektor  $p_{\text{opt}}$  so dass  

$$p_{\text{opt}} = \operatorname{argmin}_p D(V_R, V_S(p))$$
 Aufgabe: **Stochastische Optimierung**

Methodisch gesehen ist Kalibrieren eines stochastischen Simulators identisch zu Experimentieren mit einem stochastischen Simulator

Also: Für gegebenes Realsystem mit Verhalten  $V_R$  wird über Experimente versucht, ein System (Modell)  $E$  mit Verhaltensgüte  $V_E$  zu finden, so dass  $D(V_R, V_E)$  kleiner als vorgegebene Schranke

Möglichkeiten der Veränderung zur „besten/akzeptablen“ Alternative

- Ausprobieren struktureller Alternativen
- Tuning von System-Parametern

⇒ Methoden zum qualitativen und quantitativen Experimentieren mit stochastischen Systemen sollten hilfreich sein

Angenommen wir hätten  $D$  kleiner als erträgliches  $D_{\text{ertr}}$   
erreicht

Ist dann das inhaltliche Ziel erreicht: Folgerung aus Objekt-  
und Modell-Experiment sind identisch

Was wurde getan?

Simulator Verhalten wurde (mittels Änderungen) einem  
Objekt-Verhalten angepasst

- für einen Zustand der Umwelt
- für einen Zustand des Systems

d.h. für genau eine Situation

Ziel der Simulation sind aber

- Aussagen über andere Situation  
(das Resultat der Kalibriersituation ist ja bekannt)

## Skizze des Prinzips

### Situationenraum

- K
- E

- Kalibrierung für Situation K
- Einsatz des Simulators für Situation E

Zentrale Frage:

Ist der Unterschied  $D$  bei E gleich oder ähnlich dem Unterschied  $D$  bei K?

**Dies ist u.U fraglich!**

Wenn Verhaltensunterschiede existieren, dann (wahrscheinlich) abhängig von der jeweiligen Situation, zu erwarten

- Bereich mit hinreichend kleinen Unterschieden: Gültigkeitsbereich  $G$
- Bereich mit zu großen Unterschieden  $\text{not } G$



Frage reduziert sich zu  $E \in G$  oder  $E \notin G$

- Prinzipiell nur beantwortbar, wenn
- von Verhaltensunterschieden für gewisse Situationen
- auf Verhaltensunterschiede in anderen Situationen geschlossen werden kann

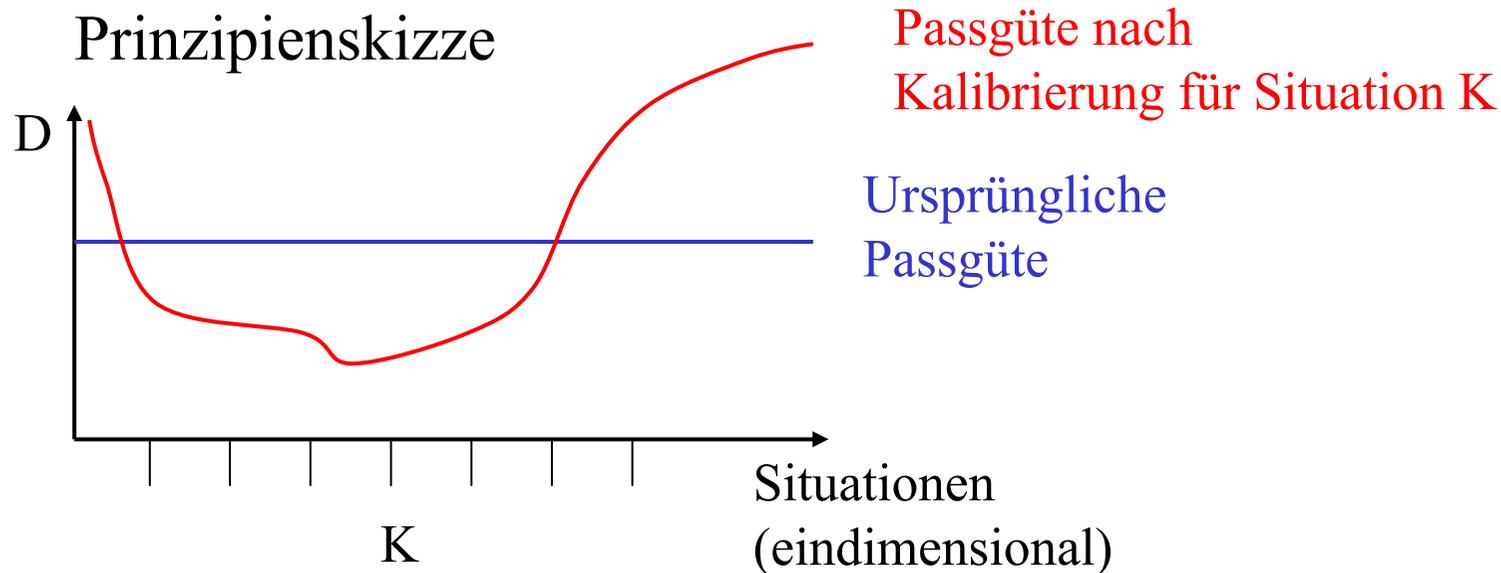
Dies gilt i.d.R. nicht, da dann Aussagen über das Systemverhalten für Situationen erforderlich, bei denen keine Experimente durchgeführt wurden

Damit (leider!) Übergang

- vom Beweis der Gültigkeit
- zur Zuversicht in Gültigkeit

bei jedem vorhersagendem Modellieren

# Kalibriervorgang kann Problem durch Überanpassung noch verschärfen

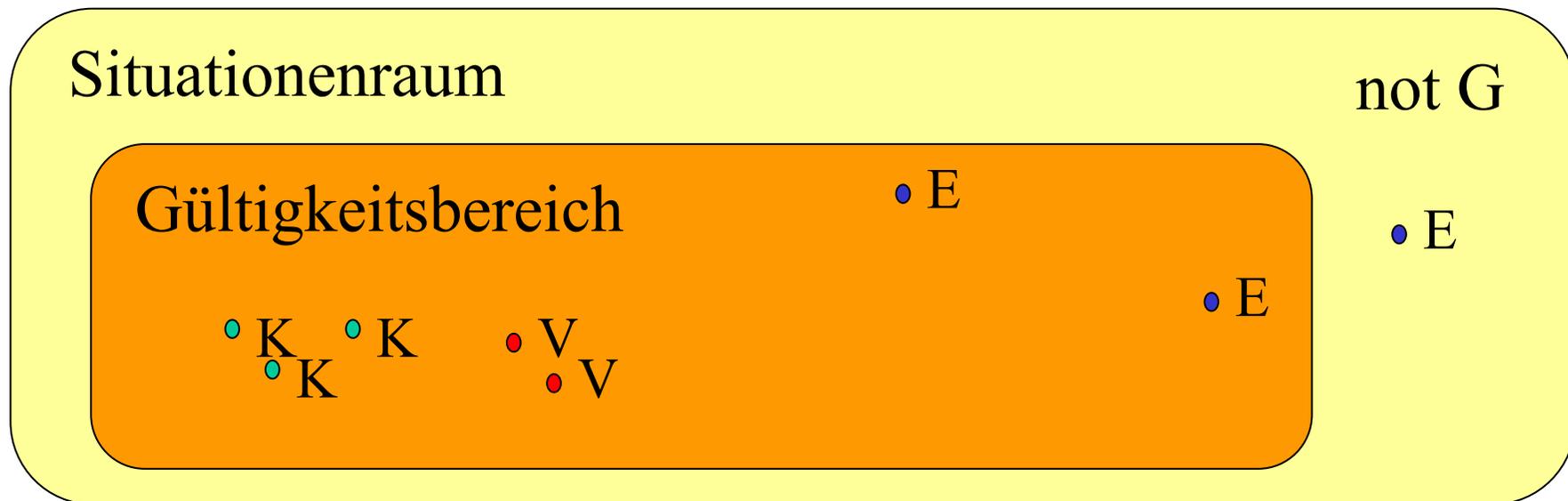


- Je nach Lage von E schlechtere Resultate nach der Kalibrierung
- Gefahr der Überanpassung kann reduziert werden, wenn für mehr als eine Situation kalibriert wird.

Dennoch ist formaler zu zeigen, dass Kalibrierung nicht zu Überanpassung führt!

Wie kann man plausibel machen, dass Model- und System-Verhalten hinreichend übereinstimmen?

- Spezielle und unabhängige Validierungsläufe für die Situationen, die nicht zur Kalibrierung genutzt wurden
- Prüfung, ob Verhaltensunterschiede für diese (Validier-) Situationen „in etwa so“ wie für bekannte (Kalibrier-) Situationen



K Kalibrier-Situation, V Validier-Situation, E Experiment-Situation

- Allerdings, ob eine Experiment-Situation E im (unbekannten) Gültigkeitsbereich G liegt oder nicht, ist nach wie vor unbeweisbar (in irgendeinem strengen Sinne) bzw. ist nur retrospektiv beweisbar, dann aber uninteressant für Vorhersagen!
- Aber das Vertrauen in ein Modell wächst durch erfolgreiche Validierung (auch durch erfolgreiche retrospektive Validierung)
- Weitere Unterschiede auch bzgl. Abstand zu Validier- / Kalibrier-Situationen
  - Relativ große Sicherheit für Situationen E nahe bei bekannten Situationen K oder V (aber meistens weniger interessant)
  - Kaum Sicherheit für Situationen E weit weg von bekannten Situationen K oder V (aber oft interessant)

# Messung von Verhaltensunterschieden

Bestimmung des Ausmaßes von Verhaltensunterschieden  
Objekt/Modell war wesentlich bei

- Kalibrierung: Realitätstreue des Modells verbessern durch Reduktion der Verhaltensunterschiede
- Validierung: Realitätstreue des Modells bestätigen durch Überprüfung von Verhaltensunterschieden in (von Kalibrierung) unabhängigen Situationen

Kalibrierung/Validierung soll/muss Vertrauen in  
hinreichende Realitätstreue unterstützen, denn Modell wird  
erstellt für Experimente

⇒ Modellgüte erhöhen durch Modellanpassung

## Zentrale Fragestellungen beim „Messen von Verhaltensunterschieden“:

1. Mit dem Verhalten welchen Systems soll das Verhalten des Simulators verglichen werden?
2. Welche Verhaltens-Aspekte sollen für den Vergleich gewählt werden?
3. Welche Vergleichs-Methoden und –Techniken sollen eingesetzt werden?

Zu 1.

- Verhalten eines realen Objekts wäre ideal, aber
  - System muss existieren und beobachtbar sein
  - dann Verhaltensbeobachtung in verschiedenen Situationen
    - betreibe System und Modell in identischer Situation
    - oder verwendet vorhandene Aufzeichnungen über Systemverhalten, auch Aufzeichnungen aus „ähnlichen Systemen“
    - Trace-getriebene Simulation kann unterstützend wirken
    - nicht alle Daten für die Kalibrierung verwenden, sondern Daten für Validierung zurückhalten

## Zu 1. (Fortsetzung)

- Verhalten analytischer Modelle
  - Analytisches Modell für Marginalsituationen entwickeln  
(z.B. ohne ZVs, bei sehr hoher Last, ..)
  - Ergebnisse der Simulation für diese Situationen mit analytischen Resultaten vergleichen
- Verhalten anderer Simulationsmodelle
  - Modelle ähnlicher Systemee
- Weitere Möglichkeiten (in der Praxis kaum verwendet)
  - Nutzung von Expertenwissen  
(z.B. kann Experte Ausgaben des Real-System von Ausgaben des Simulators unterschieden?)

Zu 2.

- Ziel ist es, das System auf Basis bestimmter (ausgewählter) Leistungskriterien zu bewerten
  - Einsatz dieser Kriterien für den Vergleich
  - bei mehreren Kriterien oft Zielkonflikte
  - deshalb wird oft die Analyse auf ein Kriterium beschränkt, dann
    - Kalibrierung und Validierung bzgl. dieses Kriteriums
    - andere Kriterien nur zur Unterstützung
  - oder Einsatz mehrerer Modelle für die unterschiedlichen Kriterien

### Zu 3. Fallunterscheidung bzgl. der eingesetzten Methoden

- Zum Vergleich Objektsystem – Simulator
  - Problemtyp: Vergleich zweier Stichproben
  - Entscheidung, ob zwei Stichproben hinreichend ähnlich
  - Oft beschränkt sich der Vergleich auf wenige Charakteristika (z.B. Mittelwerte)
  - Verfahren dazu auf den folgenden Folien
- Zum Vergleich analytisches Modell – Simulator
  - Problemtyp: Vergleich analytische Verteilung mit einer Stichprobe
  - Entscheidung, ob Stichprobe hinreichend ähnlich (d.h. aus analytischer Verteilung stammen könnte)
  - Verfahren kennen wir bereits ( $\chi^2$ -Test, K.S.-Test, ...)

# Testverfahren

Ziel: Vergleich zweier Stichproben

(z.B. Verweilzeit Kunden am Bankschalter)

$$V_R = (V_{R_1}, \dots, V_{R_n})$$

$$V_S = (V_{S_1}, \dots, V_{S_m})$$

(z.B. R aus Beobachtungen Realsystem, S aus Simulation)

Man unterscheidet

- Subjektive Verfahren  
(basierend auf graphischen Darstellungen)
- Objektive Verfahren  
(basierend auf statistischen Tests)

## Inspektionsansatz

Darstellung der Stichproben in graphischer Form durch

- Histogramme
- Punktdiagramme
- Box-Plots

Dargestellt werden

- alle Werte der Stichprobe oder
- eine zufällige Auswahl der Werte (bei Punktdiagrammen)  
oder
- die Stichprobe nach Elimination von Ausreißern

Anschließend Bewertung durch visuellen Vergleich

- Entwickler des Simulationsmodells
- Experten im Anwendungsbereich
- Turing Test

Probleme bei diesem Vorgehen

- Graphische Darstellung visualisiert nur einen Teil der Information
- Unterschiedliche Parametrisierung führt zu unterschiedlichen visuellen Eindrücken
- Keine „objektivierbaren“ Kriterien zur Entscheidungsfindung

Aber Inspektionsmethode ist ein erster wichtiger Schritt und kann mit statistischen Auswahlverfahren kombiniert werden

## Vergleich von Konfidenzintervallen

Ziel: Aussage ob Mittelwerte gleich oder unterschiedlich  
(deutlich restriktiver als bei der Inspektionsmethode!)

Voraussetzung:

Werte der beiden Stichproben seien unabhängig identisch verteilt!

Verschiedene Methoden existieren, im Wesentlichen  
Untersuchung, ob Konfidenzintervall für die Differenz der  
Mittelwerte 0 enthält

Wir betrachten hier zwei Verfahren

- Ungepaarte Stichproben (Welch-Verfahren)
- Gepaarte Stichproben (Paired t-Konfidenzintervalle)

## Ungepaarte Stichproben (Welch Verfahren)

Zusätzliche Voraussetzung:

Werte zwischen den Stichproben sind unabhängig  
aber keine Annahmen bzgl. identischer Varianz

Schätzer für die beiden Stichproben:

$$\begin{aligned}\tilde{\mu}_R &= \frac{1}{n} \sum_{i=1}^n v_{R_i} & \tilde{S}_R^2 &= \frac{1}{n-1} \left( \sum_{i=1}^n v_{R_i}^2 - n \cdot \tilde{\mu}_R^2 \right) \\ \tilde{\mu}_S &= \frac{1}{m} \sum_{i=1}^m v_{S_i} & \tilde{S}_S^2 &= \frac{1}{m-1} \left( \sum_{i=1}^m v_{S_i}^2 - m \cdot \tilde{\mu}_S^2 \right)\end{aligned}$$

Uns interessiert die Differenz  $\mu_{RS} = \mu_R - \mu_S$

Mit Hilfe eines Konfidenzintervalls für  $\mu_{RS}$  können wir entscheiden.

- Falls die 0 im Konfidenzintervall enthalten ist, so wird  $\mu_r = \mu_s$  angenommen,
- ansonsten wird  $\mu_r \neq \mu_s$  angenommen

Wenn  $n = m$  gilt, so ist  $v_{RS} = (v_{RS_1}, \dots, v_{RS_n})$   
mit  $v_{RS_i} = v_{R_i} - v_{S_i}$  eine Stichprobe der Differenz

Aus dieser Stichprobe kann  $\hat{\mu}_{RS}$  und  $\hat{S}_{RS}^2$  bestimmt werden und daraus mit den bekannten Methoden ein Konfidenzintervall

$$\hat{\mu}_{RS} \pm t_{n-1, 1-\alpha/2} \cdot \hat{S}_{RS} / (n)^{1/2}$$

berechnet werden.

## Grenzen des Vorgehens

- $n = m$  muss gelten, ansonsten müssen Beobachtungen weggelassen werden
  - Da in der Simulation einfach Daten erzeugt werden können, gilt aber oft  $m \gg n$
- Es werden gleiche Varianzen der beobachteten Stichproben vorausgesetzt
  - Annahme gleicher Varianzen ist unrealistisch
  - Experimente zeigen, dass die Resultate für  $n = m$  auch bei ungleichen Varianzen akzeptabel sind

Für den allgemeinen Fall ist die Varianz nicht exakt zu ermitteln, folgende Approximation von Welch (1938) liefert aber i.d.R. gute Resultate

$$\hat{\mu}_{RS} = \hat{\mu}_R - \hat{\mu}_S \quad \text{und} \quad \hat{S}_{RS}^2 = \hat{S}_R^2/n + \hat{S}_S^2/m$$

Schätzer für die Anzahl der Freiheitsgrade der  $t$ -Verteilung:

$$\hat{f} = \frac{(\hat{S}_{RS}^2)^2}{\frac{1}{n-1} \cdot (\hat{S}_R^2/n)^2 + \frac{1}{m-1} \cdot (\hat{S}_S^2/m)^2}$$

- Da  $\hat{f}$  i.a. keine ganze Zahl, Wert runden oder Werte der  $t$ -Verteilung interpolieren!

Konfidenzintervall:  $\hat{\mu}_{RS} \pm t_{\hat{f}, 1-\alpha/2} \cdot \hat{S}_{RS}$

Beispiel:	$i$	$V_{Ri}$	$V_{Si}$	$V_{Ri} - V_{Si}$
	1	126.97	118.21	8.76
	2	124.31	120.22	4.09
	3	126.68	122.45	4.23
	4	122.66	122.68	-0.02
	5	127.23	119.40	7.83

Schätzwerte:

$$\hat{\mu}_R = 125.57, \hat{\mu}_S = 120.59, \hat{S}_R^2 = 4.0 \text{ und } \hat{S}_S^2 = 3.76$$

$$\text{Sowie } \hat{\mu}_{RS} = 4.98, \hat{S}_{RS}^2 = 1.55 \text{ und } \hat{f} = 7.99$$

Konfidenzintervalle:

- [2.67, 7.29] zum Signifikanzniveau 0.1
- [0.81, 9.15] zum Signifikanzniveau 0.01

in beiden Fällen ist 0 nicht enthalten

## Gepaarte Stichproben (Paired t-Konfidenzintervalle)

Zusätzliche Voraussetzung

**n = m identische Beobachtungszahlen**

aber keine Annahmen bzgl. identischer Varianz oder  
Unabhängigkeit der Stichproben!

Bestimme  $\hat{\mu}_{RS}$  wie vorher und  $\hat{S}_{RS}^2 = \frac{\sum_{i=1}^n \left( (v_{R_i} - v_{S_i}) - \hat{\mu}_{RS} \right)^2}{n \cdot (n-1)}$

Berechnung des Konfidenzintervalls:

$$\hat{\mu}_{RS} \pm t_{n-1, 1-\alpha/2} \cdot \hat{S}_{RS}$$

Beispiel:

Daten aus dem vorherigen Beispiel (siehe Folie 283)

liefern  $\hat{\mu}_{RS} = 4.98$  und  $\hat{S}_{RS} = 1.56$

Konfidenzintervalle:

- $[1.66, 8.30]$  zum Signifikanzniveau 0.1
- $[-2.20, 12.16]$  zum Signifikanzniveau 0.01

Zum Signifikanzniveau 0.1 können die beiden Mittelwerte als ungleich angenommen werden, zum Signifikanzniveau 0.01 wird Gleichheit angenommen

Falls beide Verfahren anwendbar sind,

- ist nicht klar, welches schmalere Konfidenzintervalle liefert,
- sollte nach Datenlage entschieden werden

## Weitere Methoden

### Testverfahren

- Parameterisch (d.h. mit Verteilungsannahme) um  $\mu_R \neq \mu_S$  zu testen
- Nicht-parametrisch (d.h. ohne Verteilungsannahme), oft wird auf  $P[V_R < V_S] = 0.5$  getestet

Falls nur Mittelwerte verglichen werden sollen, sind Konfidenzintervalle zu bevorzugen, da sie quantitative Aussagen liefern, während Testverfahren eine binäre Antwort liefern.