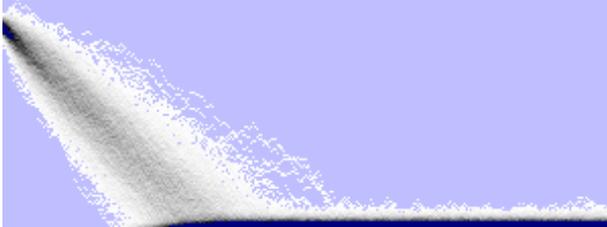


**Statistische Auswertung und Erkennung
der stationären Phase in der Simulation
zustands-diskreter Systeme**



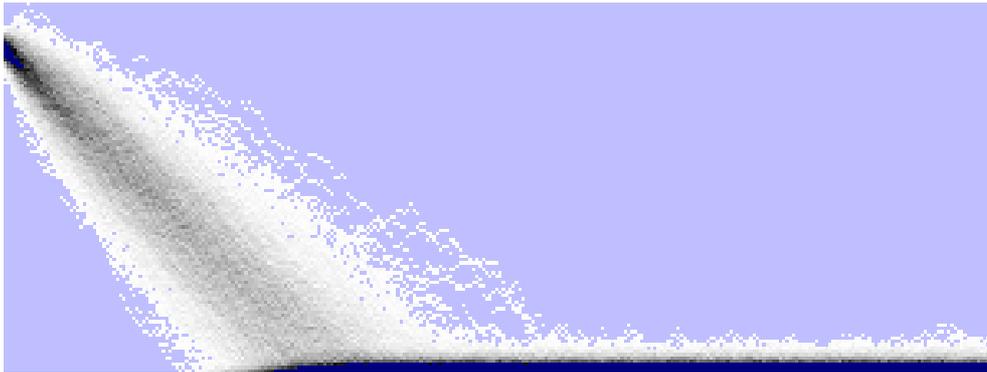
Diplomarbeit
von Mirko Eickhoff

Erklärung:

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Hilfsmittel angefertigt habe.

Dortmund, Januar 2002

Statistische Auswertung und Erkennung der stationären Phase in der Simulation zustands-diskreter Systeme



Diplomarbeit
von Mirko Eickhoff

13. Januar 2002

Lehrstuhl:

FB Informatik, Lehrstuhl IV

Universität Dortmund, Deutschland

Gutachter:

1. Dr. Falko Bause

2. Prof. Dr.-Ing. Heinz Beilner

Zusammenfassung

Deutsch

Viele Standard-Analyseverfahren für simulative Messreihen basieren auf der Berechnung des Mittelwertes und seinem Konfidenzintervall. Es existieren jedoch Simulationsmodelle, bei denen dieses Vorgehen zu grundsätzlich falschen Annahmen und Ergebnissen führt.

In dieser Diplomarbeit wird ein erweitertes Analyseverfahren vorgestellt, das zunächst prüft, ob ein stabiles Modell vorliegt. Durch diverse Methoden wird der Einfluss von dem Initialisierungszustand des Simulationsmodells auf die Ergebnisse eliminiert. Anschließend erfolgt die eigentliche statistische Auswertung.

Ausgewählte Methoden zur Auswertung von Simulationsdaten werden implementiert. Die entwickelten Software-Tools generieren aussagekräftige Ergebnisdarstellungen und ermöglichen in einigen Beispielen eine verbesserte Auswertung.

English

Many standard analysing methods for time series of simulation base on the determination of the mean and his confidence intervalls. But there are some simulation modells, where these methods lead to completely wrong suppositions and results.

In this diploma thesis, an expanded analysing-technic will be proposed. At first this technic will check the stability of the model. With the help of some methods, the influence of the initialization on the results (initialization bias) is eliminated. In the next step the statistical analysis follows.

Selected methods for analysing the simulationdata are implemented. The developed software generates detailed graphical results and allows in some examples a better analysis.

Inhaltsverzeichnis

| | |
|--|-----------|
| Zusammenfassung | iv |
| 1 Einleitung | 1 |
| 1.1 Güterverkehrszentrum | 2 |
| 2 Grundlagen der Analyse | 5 |
| 2.1 Modelle | 5 |
| 2.1.1 Definitionen | 5 |
| 2.1.2 Klassifizierung | 6 |
| 2.2 Allgemeine Analyse | 6 |
| 2.3 Simulative Analyse | 7 |
| 2.3.1 Ansätze und Ziele | 8 |
| 2.3.2 Realisierung | 10 |
| 2.3.3 Zufallszahlengenerator | 11 |
| 2.3.4 Unabhängigkeit von Simulationsläufen | 13 |
| 3 Grundlagen der Statistik | 15 |
| 3.1 Zeitreihen | 15 |
| 3.2 Zufallsvariablen | 16 |
| 3.2.1 Diskrete Zufallsvariablen | 16 |
| 3.2.2 Kontinuierliche Zufallsvariablen | 17 |
| 3.3 Parameter, Momente und empirische Momente | 18 |
| 3.3.1 Der Erwartungswert und das arithmetische Mittel | 18 |
| 3.3.2 Die Varianz | 19 |
| 3.3.3 Die Standardabweichung und der Variationskoeffizient | 20 |
| 3.3.4 Die Kovarianz und der Korrelationskoeffizient | 20 |
| 3.3.5 Median | 23 |
| 3.3.6 Modus | 23 |
| 3.3.7 Quantile | 24 |
| 3.3.8 Konfidenzintervalle | 24 |
| 3.4 Stationarität | 26 |
| 3.4.1 Mathematische Sichtweise | 26 |

| | | |
|----------|--|-----------|
| 3.4.2 | Simulative Sichtweise | 28 |
| 3.4.3 | Analytische Sichtweise | 29 |
| 3.5 | Ergodizität | 30 |
| 3.6 | Stichprobenvergleich | 31 |
| 3.6.1 | χ^2 -Test | 31 |
| 3.6.2 | Kolmogoroff-Smirnoff-Test | 32 |
| 3.6.3 | Mann-Whitney U-Test | 32 |
| 3.6.4 | Wilcoxon Signed-Rank Test | 32 |
| 3.6.5 | Differenz-Test | 33 |
| 3.7 | Unabhängigkeitstest | 33 |
| 3.7.1 | Run-Test (steigend/fallend) | 33 |
| 3.7.2 | Run-Test (größer/kleiner) | 34 |
| 4 | Allgemeine Auswertungsstrategie | 35 |
| 4.1 | Simulationsdaten | 36 |
| 4.1.1 | Disjunkte Intervalle | 36 |
| 4.1.2 | Wandernde Intervalle | 37 |
| 4.2 | Ensemble-Analyse | 38 |
| 4.3 | Zeitreihen-Analyse | 39 |
| 4.4 | Auswertungsbeispiel | 40 |
| 4.4.1 | Theoretische Betrachtung | 40 |
| 4.4.2 | Standard-Methoden | 41 |
| 5 | Chaos - eine andere Sichtweise | 45 |
| 5.1 | Ein Modell des Bender-Oszillators | 45 |
| 5.2 | Parallelen zu stochastischen Prozessen | 47 |
| 5.3 | Schlussfolgerungen | 50 |
| 6 | Verbesserte Auswertungsstrategie | 52 |
| 6.1 | Theoretische Überlegungen | 52 |
| 6.2 | Eine lange Simulation | 54 |
| 6.2.1 | Erkennung der stationären Phase | 55 |
| 6.2.2 | Auswertung der stationären Phase | 69 |
| 6.3 | Viele kurze Simulationen | 75 |
| 6.3.1 | Erkennung der stationären Phase mit Reduktion | 75 |
| 6.3.2 | Erkennung der stationären Phase ohne Reduktion | 79 |
| 6.3.3 | Auswertung der stationären Phase | 84 |
| 7 | Implementierte Software-Tools | 88 |
| 7.1 | ssa: "steady-state" Analyse | 88 |
| 7.1.1 | Voraussetzungen | 89 |

| | | |
|----------|--|------------|
| 7.1.2 | Aufruf und Ablauf | 91 |
| 7.1.3 | Erkennung der stationären Phase | 94 |
| 7.1.4 | Auswertung der stationären Phase | 96 |
| 7.1.5 | Verkehrsflussgleichgewicht | 97 |
| 7.1.6 | Ergebnisdarstellung | 97 |
| 7.1.7 | Parallelisierung | 100 |
| 7.1.8 | Implementierung | 101 |
| 7.2 | steady | 102 |
| 7.2.1 | Aufruf und Ergebnisse | 102 |
| 7.2.2 | Leistungsgrößen | 104 |
| 7.2.3 | Implementierung | 105 |
| 7.3 | hit2steady | 106 |
| 7.3.1 | Aufruf und Ergebnisse | 106 |
| 7.3.2 | Implementierung | 110 |
| 7.4 | steady_hit | 111 |
| 7.5 | visual | 112 |
| 7.5.1 | Aufruf | 113 |
| 7.5.2 | HISLANG-Modelle als Datenquelle | 115 |
| 7.6 | seed | 116 |
| 7.7 | oszillator | 117 |
| 8 | Testmodelle | 119 |
| 8.1 | Güterverkehrszentrum | 119 |
| 8.1.1 | Test auf Stationarität | 120 |
| 8.1.2 | Bewertung | 122 |
| 8.1.3 | Lagerbestand | 122 |
| 8.2 | M/M/1-Server: schlecht initialisiert | 123 |
| 8.2.1 | Zufallsverteilungsmethode und direkte Analyse | 124 |
| 8.2.2 | Balance-Regel und Batch Means für viele Simulationen | 127 |
| 8.2.3 | Spaced Batch Means für viele Simulationen | 129 |
| 8.2.4 | Balance-Regel und Batch Means für eine Simulation | 129 |
| 8.2.5 | Ergebnisüberblick | 130 |
| 8.3 | M/M/1-Server: normal ausgelastet | 131 |
| 8.3.1 | Analyse ohne Ausschluss der transienten Phase | 132 |
| 8.3.2 | Analyse mit Ausschluss der transienten Phase | 133 |
| 8.3.3 | Bewertung | 134 |
| 8.4 | M/M/1-Server: überlastet | 134 |
| 8.4.1 | Test auf Stationarität | 135 |
| 8.4.2 | Bewertung | 135 |
| 8.5 | Lagerverwaltung | 136 |

| | | |
|----------|---|------------|
| 8.5.1 | Test auf Stationarität | 136 |
| 8.5.2 | Bewertung | 137 |
| 8.6 | Warteschlangennetz mit mehreren Kundenklassen. | 138 |
| 8.6.1 | Test auf Stationarität | 139 |
| 8.6.2 | Bewertung | 140 |
| 9 | Schlussfolgerung und Ausblick | 142 |
| 9.1 | Erkennung der Stationarität | 142 |
| 9.2 | Erkennung der stationären Phase | 143 |
| 9.3 | Statistische Auswertung der stationären Phase | 145 |
| 9.4 | Implementierung | 146 |
| 9.5 | Fazit | 148 |

Kapitel 1

Einleitung

Im Alltag begegnen wir vielen Abläufen, deren Regeln und Zusammenhänge so komplex sind, dass eine intuitive Bewertung nur noch bedingt möglich ist. Gerade in der Produktionswirtschaft, in logistischen Systemen oder auch im Supply Chain Management ist es jedoch wichtig, die Abläufe möglichst effizient zu gestalten. Um komplexe Abläufe effizienter gestalten zu können, muss zunächst eine Bewertung der Abläufe durchgeführt werden. Hier gibt es viele verschiedene Vorgehensweisen, wie zum Beispiel die Simulation. Die Simulation besitzt viele Vorteile gegenüber anderen Analyseverfahren. Es gibt jedoch auch Nachteile, die bei einer unüberlegten Anwendung schnell zu falschen Ergebnissen führen.

Diese Diplomarbeit ist im Rahmen des Sonderforschungsbereiches 559 “Modellierung großer Netze in der Logistik” angesiedelt. Im Vordergrund steht eine Optimierung der Datenauswertung bei der Simulation, um qualitativ bessere Ergebnisse zu ermöglichen.

Bei der Bewertung von Abläufen interessiert meistens deren typisches Verhalten. Dies muss bei der Simulation berücksichtigt werden. Zunächst sollte die Frage geklärt werden, ob das betrachtete System überhaupt ein typisches Verhalten besitzt, das bewertet werden kann. Wenn zum Beispiel eine Warteschlange stetig wächst, macht es keinen Sinn, von einer typischen Warteschlangenlänge zu sprechen. Die Stabilität (siehe Kapitel 3.4) des Systems ist eine Voraussetzung für die Analyse des typischen Systemverhaltens.

Um eine Simulation durchführen zu können, wird von dem realen System ein Simulationsmodell abgeleitet (siehe Kapitel 2.1). Das Simulationsmodell wird zu Beginn der Auswertung mit einem ausgewählten Zustand initialisiert. Dieser Initialisierungszustand muss nicht im Zusammenhang mit dem typischen Systemverhalten stehen. Meistens braucht die Simulation eine Anfangsphase, um sich von dem untypischen Initialisierungszustand zu erholen und sich auf das typische Systemverhalten einzupendeln. Diese Anfangsphase wird transiente Phase genannt. Die Phase des typischen Systemverhaltens nennt sich stationäre Phase

(siehe Kapitel 3.4.2). Da die Messwerte, die während der transienten Phase beobachtet werden, für eine Bewertung des typischen Systemverhaltens nicht geeignet sind, ist es wichtig, den Beginn der stationären Phase zu ermitteln.

Die Erkennung der stationären Phase beinhaltet also zwei unterschiedliche Fragestellungen:

1. Ist das zu analysierende System stabil?
2. Ab wann liegt ein typisches Systemverhalten vor?

In den Kapiteln 6.2.1, 6.3.1 und 6.3.2 werden Verfahren und Regeln vorgestellt, die bei der Beantwortung der Fragen behilflich sind.

Im Anschluss müssen die beobachteten Messwerte der stationären Phase geeignet aufbereitet werden. Ansonsten wäre der Benutzer auf Grund der Größe der Urliste (siehe Kapitel 3) im höchsten Maße überfordert. Eine statistische Auswertung ist erforderlich. Die Grundlagen für die statistische Auswertung werden im Kapitel 3 vorgestellt. Auf die Situation bei der Auswertung von Simulationsdaten wird in den Kapiteln 6.2.2 und 6.3.3 eingegangen.

1.1 Güterverkehrszentrum

Im Rahmen des Sonderforschungsbereichs 559 “Modellierung großer Netze in der Logistik” wurde die Analyse eines Güterverkehrszentrums unternommen. Die hierbei ermittelten Ergebnisse sind in dem Bericht [3] und dem Artikel [6] veröffentlicht worden. Im Bereich dieser Diplomarbeit wird ein vereinfachtes Modell des Güterverkehrszentrums betrachtet, welches jedoch dieselben Schwierigkeiten bei der Analyse beinhaltet.

Das entworfene Modell besteht aus einem Lager, das eine begrenzte Anzahl an Gütern aufnehmen kann. LKW und Züge besuchen das Lager und laden Güter auf und ab. Im Durchschnitt werden gleichviele Güter ein- wie ausgelagert. Um die eigentlichen Auswertungsschwierigkeiten hervorzuheben, benötigen die Auf- und Abladevorgänge in diesem vereinfachten Modell keine Modellzeit. Für die LKW und Züge entstehen nur dann Wartezeiten, wenn das Lager an seine Kapazitätsgrenze stößt. In diesem Fall verzögern sich die Be- und Entladevorgänge. Die Abbildung 1.1 zeigt das vereinfachte Modell des Güterverkehrszentrums im B1-Modellformalismus (siehe [11]).

Das Modell des Güterverkehrszentrums ist nicht stabil und besitzt daher keine stationäre Phase. Eine Aussage über das typische Systemverhalten ist daher sinnlos. Dies wird mit Hilfe von Markov-Ketten in [3] und [6] bewiesen. Eine besondere Eigenschaft gilt ausschließlich in der Anfangsphase der Simulation: Im Durchschnitt ist die Anzahl der LKW und Züge im Lager konstant. Im Folgenden

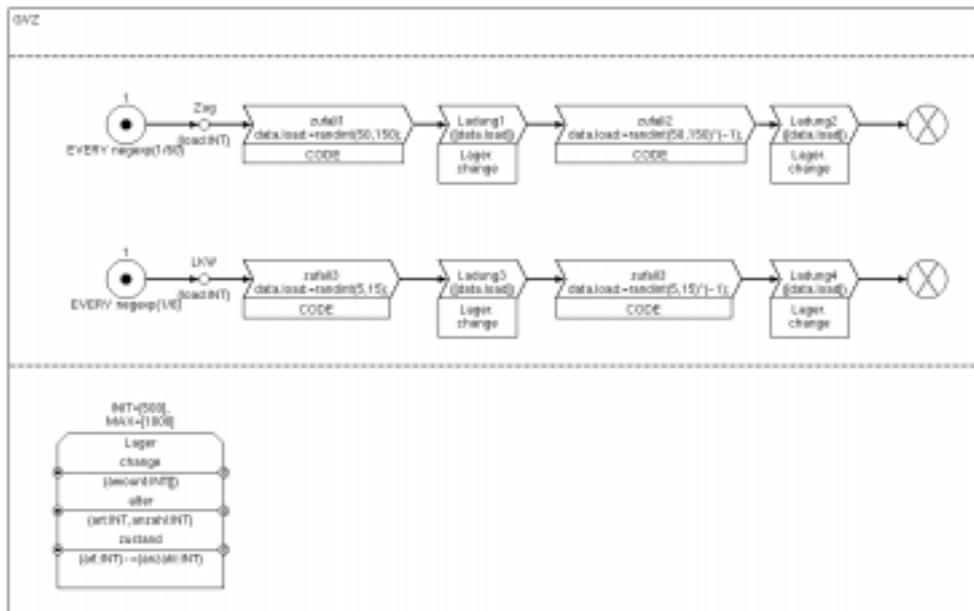


Abbildung 1.1: Das Modell des Güterverkehrszentrums im B1-Modellformalismus.

wird deutlich, dass auf Grund dieser Eigenschaft die Standard-Analyseverfahren (siehe Kapitel 4) falsche Ergebnisse liefern.

Die Standard-Analyseverfahren benutzen Konfidenzintervalle (siehe Kapitel 3.3.8), um ein Abbruchkriterium für die Simulation zu bestimmen. Ein übliches Abbruchkriterium ist, dass das 95%-Konfidenzintervall nur 5% des Mittelwertes betragen darf.

Wird ein Standard-Analyseverfahren mit diesem Abbruchkriterium auf das vereinfachte Modell des Güterverkehrszentrums angewendet, werden grundsätzlich falsche Ergebnisse ermittelt. Durch die konstante Anfangsphase des Modells ist das Abbruchkriterium schnell erfüllt. Die Standard-Analyse liefert daher einen Mittelwert und ein passendes Konfidenzintervall, obwohl das analysierte Modell nicht stabil ist. Die Abbildung 1.2 zeigt den zeitlichen Verlauf der Anzahl der Züge und LKW, die durch die Kapazitätsgrenze des Lagers blockiert werden. Nachdem ungefähr 600 LKW und Züge das Lager besucht und verlassen haben (Modellzeit 3500) ist die Instabilität des Güterverkehrszentrums zu erkennen.

In einer beispielhaft durchgeführten Simulation hat die Standard-Analyseverfahren des Simulationstools HIT (“Hierarchical Evaluation Tool”, siehe [9]) schon zur Modellzeit 278 ein befriedigendes Konfidenzintervall bestimmt, obwohl erst ungefähr 50 LKW und Züge das Lager betreten und verlassen haben. Hierdurch wurde die Instabilität des Modells bei dieser Analyse nicht erkannt. Würde sich ein Benutzer in diesem Fall auf das Ergebnis der Standard-Analysemethoden verlassen, hätte er die falsche Schlussfolgerung gezogen, dass ein stabiles Modell

vorliegt. In der Abbildung 1.2 ist ein deutlicher Anstieg nach der langen konstanten Anfangsphase zu erkennen. Dieser Anstieg bestätigt die Instabilität des Modells. Die Auswertung des Systemverhaltens durch die Standard-Analyseverfahren ist somit fehlgeschlagen.

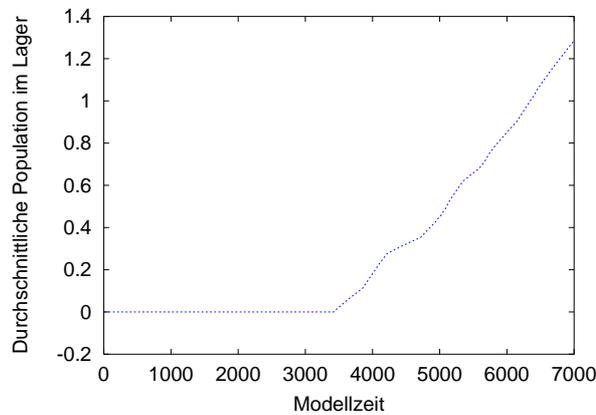


Abbildung 1.2: Anzahl der Züge und LKW im Lager des Güterverkehrszentrums bis zur Modellzeit 7000 (ungefähr 1200 Ankünfte).

In dieser Diplomarbeit soll eine erweiterte Auswertungsstrategie verfolgt werden, die eine verbesserte Analyse ermöglicht. In diesem Zusammenhang werden einige Software-Tools (siehe Kapitel 7) entwickelt. Das Modell des Güterverkehrszentrums wird in Kapitel 8.1 erneut betrachtet. Mit den neuen Software-Tools wird eine verbesserte Analyse durchgeführt, wodurch die Instabilität des Güterverkehrszentrums erkannt wird.

Kapitel 2

Grundlagen der Analyse

Die Analyse eines Systems kann nicht nur durch eine Simulation erfolgen. Es gibt sehr viele Möglichkeiten der Analyse. Die geeignete Wahl der Analysemethode hängt von dem erstellten Modell ab. In diesem Kapitel soll ein allgemeiner Überblick über die Analysemöglichkeiten gegeben werden. In dem folgenden Abschnitt wird deutlich gemacht, welche Schwierigkeiten bei der Erstellung eines geeigneten Modells entstehen.

2.1 Modelle

Es ist meistens nicht möglich mit realen Abläufen zu experimentieren. Daher sind Modelle die Basis vieler Simulationen. Ihre Aussagekraft in Bezug auf die Realität entscheidet über den Erfolg oder Misserfolg einer Analyse.

2.1.1 Definitionen

Ein reales *System* ist eine Menge von Objekten mit gewissen Eigenschaften. Die Objekte agieren in Abhängigkeit voneinander und bestimmen damit das Systemverhalten. Die abstrakte Funktion $f : (K, U) \rightarrow E$ sei das Systemverhalten mit den kontrollierbaren Eingangsgrößen K , den unkontrollierbaren Eingangsgrößen U und den Ergebnisgrößen E . Leider ist die Funktion f nicht direkt am System ablesbar. Sie muss durch Beobachten, Ausprobieren oder Experimentieren bestimmt werden.

Ein reales System kann beliebig komplex und unhandlich sein. Experimentieren mit dem realen System ist daher in vielen Fällen unmöglich. Alternativ wird ein Ersatzsystem benutzt: das *Modell*. Das Modell muss komplex genug sein, um das reale System zu repräsentieren, aber auch genügend einfach, damit es leicht zu manipulieren ist. Dieser Gegensatz macht deutlich, dass die Herleitung eines Modells kein triviales Problem ist.

2.1.2 Klassifizierung

Modelle lassen sich auf vielfache Weise klassifizieren. Es gibt die Unterteilung in physikalische und mathematische Modelle, sowie in deterministische und stochastische Modelle oder in offene und geschlossene Modelle. Diese Liste der Unterteilungen lässt sich noch beliebig erweitern. Eine genaue Diskussion ist in [27] Kapitel 1.2, in [25] Kapitel 24.3 oder in [8] Kapitel 1 zu finden.

Im Mittelpunkt des Interesses dieser Diplomarbeit stehen die stochastischen, dynamischen und zustands-diskreten Modelle. In stochastischen (oder probabilistischen) Modellen werden Zufallsvariablen benutzt, um einige Sachverhalte oder Abhängigkeiten zu beschreiben. Daher können verschiedene Simulationsläufe eines Modells unterschiedliche Ergebnisse liefern. Allerdings kann man auf einem Standard-Computer keinen echten Zufall erzeugen, daher sind alle stochastischen Computer-Modelle im gewissen Sinne doch deterministisch. In dynamischen Modellen ist der Modellzustand, im Gegensatz zu statischen Modellen, abhängig von der Zeit. Zustands-diskrete Modelle lassen sich über ihre diskreten Zustände definieren, von denen es abzählbar unendlich viele geben kann. Wenn ein Übergang von einem Zustand in einen anderen Zustand nur zu einem absoluten Zeitpunkt stattfindet, ist das Modell ereignis-diskret. Der Übergang wird *Ereignis* genannt. Theoretisch besteht ein Unterschied zwischen zustands-diskreten und ereignis-diskreten Modellen. Meistens wird in der Literatur jedoch nicht zwischen diesen Klassen unterschieden (siehe [25], Kapitel 24.3 unter “discrete-state” und “discrete-event”). In dieser Diplomarbeit werden nur Modelle behandelt, die sowohl zustands-diskret als auch ereignis-diskret sind, daher wird auch hier keine weitere Unterscheidung vorgenommen.

Beispiel 1: *Ein Bankschalter ist ein zustands-diskretes Modell, wenn sein Zustand durch die Anzahl wartender Kunden definiert wird. Da die Anzahl der Kunden ganzzahlig ist, hat das Modell diskrete Zustände und da die Warteschlange (prinzipiell) nicht beschränkt ist, gibt es abzählbar unendlich viele Zustände.*

2.2 Allgemeine Analyse

Abhängig von der Klassifizierung des Modells lassen sich verschiedene Analysetechniken anwenden. Zum Beispiel sind *Markov-Modelle* (siehe [25], Kapitel 6.7) eine Modellwelt, die nur einen Teil der ereignis-diskreten Modelle beschreiben kann. Markov-Modelle sind alle gedächtnislos, d. h. die Zukunft (spätere Zustände) hängt nur von dem gegenwärtigen Zustand ab und nicht von Zuständen, die das Modell in der Vergangenheit angenommen hatte. Die Markov-

Bedingung lässt sich mathematisch so ausdrücken:

$$\begin{aligned} & \Pr[\mathcal{X}(t) = x | \mathcal{X}(t_n) = x_n, \mathcal{X}(t_{n-1}) = x_{n-1}, \dots, \mathcal{X}(t_0) = x_0] \\ = & \Pr[\mathcal{X}(t) = x | \mathcal{X}(t_n) = x_n], \quad \text{mit } t > t_n > t_{n-1} > \dots > t_0 \end{aligned} \quad (2.1)$$

Die Interpretation der Gleichung: Die Wahrscheinlichkeit zum Zeitpunkt t im Zustand x zu sein, hängt nur von dem letzten Zustand ab. Die Angabe aller vorherigen Zustände ist nicht nötig.

Auf Grund dieser Bedingung lassen sich die Markov-Modelle durch mathematische Methoden direkt analysieren. Es gibt z. B. mathematische Sätze, die genau bestimmen, ob sich ein Markov-Modell einer bestimmten Zustandsverteilung (siehe 3.4.2) nähert, oder ob es solch eine Grenzverteilung gar nicht gibt (siehe [7], Kapitel 2). Wenn sich die Zustandsverteilungen einer bestimmten Verteilung annähert, ist diese Verteilung (bei unendlicher Modellzeit) die Grenzverteilung.

Eine Zustandsverteilung legt fest, mit welcher Wahrscheinlichkeit ein Zustand angenommen wird. Zu jedem Zeitpunkt der Modellzeit kann es verschiedene Zustandsverteilungen geben. Wenn die Zustandsverteilungen ab einem bestimmten Zeitpunkt aber konstant sind, wird von der stationären Zustandsverteilung (siehe 3.4) gesprochen. Wenn eine stationäre Zustandsverteilung existiert, ist sie auch die Grenzverteilung.

Die Markov-Bedingung gilt nicht für jedes Modell, daher kann leider nicht für jedes zustands-diskrete Modell theoretisch bestimmt werden, ob es eine Grenzverteilung besitzt. Hier bietet die Simulation den Vorteil, dass sie auf jedes Modell angewendet werden kann. Sie hat jedoch den Nachteil, dass sie immer nur ein mögliches Verhalten "nachspielt". Allgemeingültige Ergebnisse liefert die Simulation daher nur unter bestimmten Bedingungen, die schon im Einleitungskapitel angesprochen wurden.

2.3 Simulative Analyse

Die *Simulation* ist eine Technik zur Analyse eines Modells. Durch Experimente mit einem vom System abgeleiteten Modell werden Informationen gesammelt, um die Funktionsweise und das Verhalten des Systems beurteilen zu können. Darüber hinaus kann analysiert werden, wie sich das System verhält, wenn es mit anderen Umweltbedingungen konfrontiert wird. Zum Beispiel ist die Erhöhung der Systemlast eine Veränderung der Umweltbedingungen. Ebenfalls kann analysiert werden, wie sich eine Veränderung im System auswirkt. Dies ist für praktische Probleme von großer Bedeutung. Wenn zum Beispiel eine Firma eine Änderung in ihrer Produktionskette plant, kann sie mit Hilfe eines Modells und einer

Simulationsanalyse überprüfen, ob die geplante Veränderung den gewünschten Effekt hat, bevor teure Umbauten vorgenommen werden.

Der wichtigste Vorteil der Simulation gegenüber anderen analytischen Methoden ist, dass die Simulation bei einer weitaus größeren Klasse von Modellen einsetzbar ist. Das liegt daran, dass die Simulation weniger Anforderungen, wie z. B. die Markov-Bedingung, an die Modelle stellt, als die meisten anderen Verfahren. Es gibt jedoch auch zwei große Nachteile. Bei der Simulation ist nur eine begrenzte Anzahl an Beobachtungsdaten abrufbar, die mit geeigneten statistischen Verfahren bearbeitet werden müssen. Das hierdurch bestimmte Ergebnis hat immer einen Fehler in Bezug auf den realen Wert. Er lässt sich nur durch Wahrscheinlichkeiten eingrenzen. Die Simulation liefert folglich nur statistische Ergebnisse. Der zweite Nachteil ist, dass die Simulation häufig mit viel Rechenzeit bzw. Rechenleistung verbunden ist. Analytische Verfahren sind deshalb immer der Simulation zu bevorzugen, wenn dies möglich ist.

Die zustand-diskrete (bzw. ereignis-diskrete) Simulation ist die Simulation mit zustand-diskreten (bzw. ereignis-diskreten) Modellen.

2.3.1 Ansätze und Ziele

Es gibt zwei grundsätzlich verschiedene Ansätze der zustands-diskreten Simulation: die Simulation mit begrenztem und die Simulation mit unbegrenztem Horizont. Im folgenden Abschnitt soll deutlich werden, dass die Simulationsansätze nicht miteinander konkurrieren, sondern verschiedene Arten von Problemen lösen bzw. verschiedene Fragestellungen beantworten können.

Die Simulation mit *begrenztem Horizont* ("finite-horizon simulation" oder "terminating simulation") endet zu einer bestimmten Modellzeit oder in einem bestimmten Zustand, z. B. wenn sich kein Kunde mehr im Modell befindet oder wenn alle Aufträge abgearbeitet wurden. Gestartet wird die Simulation in einem beliebigen Zustand. Dies kann der Zustand "empty & idle" sein, wenn ein Systemanlauf beobachtet werden soll. Im Allgemeinen kann aber festgehalten werden, dass die gemessenen Ergebnisgrößen vom Anfangszustand abhängig sind. Daher kann kein stationäres Systemverhalten beobachtet werden.

Die Simulation mit *unbegrenztem Horizont* (steady-state simulation oder non-terminating simulation) wird benutzt, wenn die normale bzw. typische Zustandsverteilung interessiert. Da diese stationäre Zustandsverteilung auch im realen System nicht von den Anfangsbedingungen abhängig ist, sollte das Modell zu einem Zeitpunkt beobachtet werden, der sich nicht in der vorübergehenden Anlaufphase befindet. An dieser Stelle wird deutlich, dass möglichst lange Simulationsläufe durchgeführt werden sollten, um sicherzustellen, dass die Ergebnisgrößen nicht mehr durch das Anlaufverhalten beeinflusst werden. Abbruchkrite-

rien können durch Konfidenzintervalle bestimmt werden.

Ein System hat allerdings nicht immer eine stationäre Zustandsverteilung. Das ist zum Beispiel der Fall, wenn ein System überlastet ist. Dann wird die Anzahl wartender Kunden im System mit der Zeit immer mehr steigen. Des Weiteren werden an einem realen System oft Änderungen durchgeführt, die das stationäre Verhalten beeinflussen. Dies muss in der Simulation natürlich berücksichtigt werden, da sonst die gemessene stationäre Zustandsverteilung des (unge- nauen) Modells keine Aussagekraft hat.

Bisher wurde das Langzeitverhalten eines Systems immer mit einer stationären Zustandsverteilung in Verbindung gebracht. Für Systeme mit einem *periodischen Verhalten* ist dies aber nicht sinnvoll. Von einer stationären Zustandsverteilung kann in diesem Fall nur gesprochen werden, wenn von der Periode des Systems abstrahiert wird. Der Mittelwert der Daten einer Periode könnte als Stellvertreter der ganzen Periode genutzt werden. (Die Aussagekraft des Mittelwertes müsste erst bestätigt werden.) Die ermittelte stationäre Zustandsverteilung hat dann natürlich auch nur eine eingeschränkte Aussagekraft. Ist jedoch nur eine Periode des Systems von Interesse, so wäre eine Simulation mit begrenztem Horizont sinnvoll.

Die Wahl des Simulationsansatzes hängt also in erster Linie davon ab, was beobachtet werden soll. Dies kann wiederum von dem gegebenen System abhängen. Es gibt deshalb Systeme, für die es durchaus Sinn macht, beide Simulationsansätze zu verfolgen. Es gibt aber auch Systeme, für die der unbegrenzte Simulationsansatz sinnlos ist.

Beispiel 1: *Es sei von Interesse, wann ein Kunde das erste Mal länger als 15 Minuten in einer Warteschlange einer überlasteten Bedienstation wartet. Hier kann nur eine Simulation mit begrenztem Horizont angewendet werden. Da das System wegen der Überlastung keinen stationären Zustand besitzt, ist der unbegrenzte Simulationsansatz sinnlos.*

Beispiel 2: *Angenommen, das System aus dem vorherigen Beispiel sei nicht überlastet und der Anfangszustand sei "empty & idle", ist sowohl die Frage "Wann wartet ein Kunde das erste Mal länger als 15 Minuten in einer Warteschlange einer Bedienstation?" als auch die Frage "Was ist die typische Warteschlangenlänge?" sinnvoll. Beide Simulationsansätze werden hier benötigt.*

Beispiel 3: *Eine Maschine sei ohne Unterbrechung arbeitsbereit. Sechs Tage in der Woche werden Aufträge an die Maschine gestellt. Am siebten Tag sei genügend Zeit vorhanden, angestaute Aufträge abzuarbeiten. Diese Abfolge setze sich ohne weitere Störungen*

fort. Dieses System ist periodisch, da es nach sieben Tagen immer wieder im Anfangszustand "empty & idle" ist. Wenn die Abbruchbedingung für die Simulation mit begrenztem Horizont das Erreichen des Zustandes "empty & idle" ist, sind in diesem Fall beide Simulationsansätze äquivalent. Der Systemanlauf ist in jeder Periode enthalten.

Weitere Anregungen zu diesen beiden Simulationsansätzen sind in [27] Kapitel 9.3 und in [4] Kapitel 7.1.3 zu finden.

2.3.2 Realisierung

Bei der Realisierung einer zustands-diskreten Simulation besteht die Wahl zwischen drei gleichwärtigen *Sichtweisen*. Es gibt die ereignisorientierte, die prozessorientierte und die aktivitätsorientierte Sichtweise. Das Vorgehen der drei Sichtweisen betont den jeweiligen Charakter. Zum Beispiel werden bei der prozessorientierten Sichtweise alle Abläufe und Zusammenhänge durch Prozesse modelliert. Ein Prozess ist ein Ablaufschema einer Modellkomponente und besteht aus vielen Ereignissen. Das genaue Vorgehen ist in [8] Kapitel 2 und 3 beschrieben. Da aber alle drei Sichtweisen gleichwertig sind, ähneln sich die wichtigsten Komponenten der Realisierungen und werden in jeder Sichtweise genutzt:

Plan Der Plan beschreibt das Verhalten eines Ereignisses, Prozesses oder einer Aktivität.

Modellzeit Es muss eine Variable für die Modellzeit existieren.

Fortschritt Der zeitliche Fortschritt kann entweder durch kontinuierliches Erhöhen oder durch Sprünge von Ereignis zu Ereignis durchgeführt werden. Prozesse und Aktivitäten lassen sich durch Ereignisse beschreiben, somit ist das Springen von Ereignis zu Ereignis in allen Ansätzen möglich.

Zustand Der Zustand des Systems muss in einer Datenstruktur (bzw. Variablen) festgehalten werden.

Ereignisliste Eine chronologisch sortierte Liste von Ereignissen, die in der weiteren Simulation eintreten werden.

Diese Aufzählung ist nicht vollständig, sie soll nur einen kleinen Überblick geben. Eine umfassendere Darstellung ist in [8] Kapitel 3, in [25] Kapitel 24.5.3 und in [27] Kapitel 1.3 zu finden.

2.3.3 Zufallszahlengenerator

Da Computer rein deterministisch arbeiten, muss der Zufall bei der Computersimulation von stochastischen Modellen künstlich erzeugt werden. Unterschieden wird zwischen 3 Arten von Zufallszahlengeneratoren: Zufallszahlengeneratoren, die

1. gleichverteilte und diskrete Zufallszahlen realisieren.
2. gleichverteilte und kontinuierliche Zufallszahlen realisieren.
3. beliebig verteilte Zufallszahlen realisieren.

In [8] wird beschrieben, wie die Fälle zwei und drei auf den ersten Fall reduziert werden können. Daher wird hier nur ein Zufallszahlengenerator der ersten Art vorgestellt.

Ein multiplikativer Kongruenz-Generator ist relativ einfach und besitzt daher besondere Eigenschaften, die im weiteren Verlauf der Diplomarbeit benötigt werden. Er ist eine vereinfachte Form der allgemeinen Kongruenz-Generatoren, da er rein multiplikativ arbeitet und keinen additiven Term besitzt:

$$Z_i = (a \cdot Z_{i-1}) \bmod 2^n \quad . \quad (2.2)$$

Die aktuelle Zufallszahl berechnet sich durch ihren Vorgänger. Der Startseed ist der Wert Z_0 . Mit der Wahl des Startseeds werden alle weiteren Zufallswerte festgelegt, daher hat seine Wahl eine große Bedeutung. n sei die Wortgröße, sie ist abhängig von dem Betriebssystem und der Rechnerarchitektur (in UNIX gilt $n=32$). a sei ein Parameter der Form

$$a = 5^{2p+1} \quad . \quad (2.3)$$

Mit $p = 6$ ergibt sich für $a = 1220703125$. Die Periode eines Zufallszahlengenerators ist die maximale Anzahl an Zufallswerten, bei denen sich keine Zufallszahl wiederholt. Nach [8] hat dieser Zufallszahlengenerator die Periode 2^{n-2} (siehe auch [9], Anhang D.2.3). Er kann somit 1073741824 verschiedene Zahlen erzeugen, bevor er sich wiederholt. Auf Grund der Modulo-Operation liegen die Zufallswerte im Bereich von 0 bis $2^n - 1$. Ist der Startseed Z_0 ungerade, sind alle weiteren Zufallswerte ungerade, da der Parameter a ungerade ist. Ist der Startseed hingegen gerade, sind alle weiteren Zufallswerte auch gerade Zahlen. Der Bildbereich des Zufallszahlengenerators ist viermal so groß wie die Periode.

Die besondere Eigenschaft dieses Zufallszahlengenerators ist, dass die einzelnen Zufallswerte auch direkt bestimmt werden können. Genauer gesagt: Die Iterationsgleichung kann in eine geschlossene Form gebracht werden.

$$\begin{aligned}
Z_i &= (a \cdot Z_{i-1}) \bmod 2^n \\
&= (a^2 \cdot Z_{i-2}) \bmod 2^n \\
&= \dots \\
&= (a^i \cdot Z_0) \bmod 2^n
\end{aligned} \tag{2.4}$$

Bei dieser Herleitung wird ausgenutzt, dass ein multiplikativer Kongruenz-Generator keinen additiven Term besitzt. Für allgemeine Kongruenz-Generatoren kann diese geschlossene Form nicht benutzt werden. Bei der parallelen Durchführung von mehreren Simulationen wird sich diese Eigenschaft als sehr nützlich erweisen (siehe Kapitel 6.3).

Probleme durch Bereichsüberschreitungen

Die geschlossene Gleichung 2.4 kann bei der Berechnung zu Bereichsüberschreitungen führen. Durch die Potenzierung entstehen Zahlen, welche die Wortgrenze n überschreiten. Sollte die Potenzierungsfunktion der verwendeten Programmiersprache nicht automatisch “Modulo n ” rechnen, muss dies durch den Programmierer sichergestellt werden.

Potenzierung: Die Potenzierung kann mit dem rekursiven Algorithmus durchgeführt werden, der in [1] Kapitel 11.2 beschrieben ist. Zu beachten ist jedoch, dass nur Multiplikationsalgorithmen verwendet werden, die im Zahlenbereich $[0, 2^n - 1]$ arbeiten und damit automatisch “Modulo n ” rechnen. Ein alternativer Ansatz wird in [2] durch Algorithmus 10.1 (Binäre modulo-Exponentiation) verfolgt. Auch bei diesem Ansatz muss ein geeigneter Multiplikationsalgorithmus verwendet werden.

Multiplikation: Die Multiplikation von binären Zahlen im Bereich $[0, 2^n - 1]$ kann durch Veränderung der “Divide-and-Conquer” Methode (siehe [39], Kapitel 3.9) der allgemeinen Multiplikation hergeleitet werden. Seien a und b Bitfolgen der Länge n . $a = (a_1, a_2)$ sei die Zerlegung einer Bitfolge in zwei Bitfolgen mit je $\frac{n}{2}$ Bits. $|a|$ sei die dezimale Zahleninterpretation von a im Bereich $[0, 2^n - 1]$.

Es gilt:

$$\begin{aligned}
|p| &= (|a| \cdot |b|) \bmod 2^n \\
&= ((|a_1| \cdot 2^{\frac{n}{2}} + |a_2|) \cdot (|b_1| \cdot 2^{\frac{n}{2}} + |b_2|)) \bmod 2^n \\
&= (|a_1 \cdot b_1| \cdot 2^n + |a_2 \cdot b_1| \cdot 2^{\frac{n}{2}} + |a_1 \cdot b_2| \cdot 2^{\frac{n}{2}} + |a_2 \cdot b_2|) \bmod 2^n
\end{aligned} \tag{2.5}$$

Durch die Modulo-Operation sind nur die niedrigen n Bits von Interesse. Höherwertige Bits können abgeschnitten werden. Die einzelnen Summanden müssen überprüft werden:

- $|a_2 \cdot b_2|$: Ist immer kleiner als 2^n .
- $|a_1 \cdot b_1| \cdot 2^n$: Kann komplett außer Acht gelassen werden.
- $|a_2 \cdot b_1| \cdot 2^{\frac{n}{2}}$: $|a_2 \cdot b_1| = |x| = |x_1| \cdot 2^{\frac{n}{2}} + |x_2| \Rightarrow$ nur x_2 ist von Interesse.
- $|a_1 \cdot b_2| \cdot 2^{\frac{n}{2}}$: analog \Rightarrow nur y_2 ist von Interesse.

Aus diesen Ergebnissen folgt:

$$|p| = (|x_2| \cdot 2^{\frac{n}{2}} + |y_2| \cdot 2^{\frac{n}{2}} + |a_2 \cdot b_2|) \bmod 2^n \quad (2.6)$$

Die Multiplikationen in Gleichung 2.6 verursachen keine Bereichsüberschreitung. Dies gilt nicht für die Additionen. Aus diesem Grund muss auch eine spezielle Addition verwendet werden.

Addition: Die Addition von binären Zahlen im Bereich $[0, 2^n - 1]$ kann durch geschickte Überprüfung der Summanden erfolgen. Sei $|p| = (|a| + |b|) \bmod 2^n$. Wenn $2^n - |a| \geq |b|$ gilt, dann ist $|p| = |a| + |b|$. Sonst sei $|x| = 2^n - |a|$ und es ist $|p| = |b| - |x|$. Es wird hierbei vorausgesetzt, dass $|a| < 2^n$ und $|b| < 2^n$ ist.

Bemerkung: Auf diese Weise kann die Potenzierung auf die Multiplikation, und die Multiplikation auf die Addition zurückgeführt werden. Da alle Operationen im Zahlenbereich $[0, 2^n - 1]$ erfolgen, kann keine Bereichsüberschreitung stattfinden.

2.3.4 Unabhängigkeit von Simulationsläufen

In vielen Situationen ist es sinnvoll, mehrere vergleichbare Simulationen eines Modells durchzuführen. Die Vergleichbarkeit wird durch gleiche Modellparameter und gleiche Startzustände erreicht. Um unterschiedliche Simulationsläufe zu erzwingen, muss der Startseed des Zufallszahlengenerators variiert werden. Hierbei muss der Startseed so gewählt werden, dass den Simulationsläufen eine möglichst große Anzahl an verschiedenen Zufallszahlen Z_i zur Verfügung steht. Im Optimalfall sollte jede Zufallszahl Z_i höchstens einmal verwendet werden.

Werden alle Simulationsläufe nacheinander durchgeführt, sollte der Startseed eines Simulationslaufes der Nachfolger der zuletzt verwendeten Zufallszahl sein. Erst wenn die Periode des Zufallszahlengenerators vollständig durchlaufen ist, wiederholen sich die Zufallszahlen. Auf diese Weise wird die Periode des Zufallszahlengenerators optimal ausgenutzt.

Wenn die Simulationsläufe parallel durchgeführt werden, kann dieses Verfahren nicht angewendet werden, da zu Beginn der Simulationen im Allgemeinen nicht bekannt ist, wieviele Zufallszahlen jeder Simulationslauf benötigt. Bei k parallelen Simulationsläufen sollte die Periode des Zufallszahlengenerators in k gleichgroße Intervalle unterteilt werden. Wenn n die Größe der Periode ist, enthält jedes Intervall $\frac{n}{k}$ Zufallszahlen. Der Startseed der Simulation i ist der Beginn der Periode i . Der Beginn der Periode i ist die Zufallszahl $Z_{(i\frac{n}{k})}$ für $0 \leq i \leq k - 1$. Wenn jeder Simulationslauf nicht mehr als $\frac{n}{k}$ Zufallszahlen benötigt, wird jede Zufallszahl höchstens einmal verwendet. Auf diese Weise wird die Periode des Zufallszahlengenerators ebenfalls optimal ausgenutzt. Für dieses Verfahren ist es vorteilhaft, wenn sich die Zufallszahlen Z_i direkt bestimmen lassen, ohne die gesamte Periode zu durchlaufen. Ein Beispiel zur Einteilung der Periode in k Intervalle wird in [13] Kapitel 6.8.3 vorgestellt.

Kapitel 3

Grundlagen der Statistik

Bei der Durchführung von Simulationen erhält man im Allgemeinen Zeitreihen als Ergebnis. Zeitreihen bestehen aus Messwerten, welche zu bestimmten Zeitpunkten als Beobachtungsgrößen von der Simulation ausgegeben und in der *Urliste* gespeichert werden. Diese Urliste kann eine riesige Datenmenge sein, die auf den ersten Blick keine Interpretation oder Aussage zulässt. Ziel bei der Auswertung der Zeitreihen ist, die riesige Datenmenge zu einigen wenigen Informationen zusammenzufassen, die ein möglichst intuitives Verständnis des Modells nahelegen. Diese wenigen Informationen sind der Aussagekern und sollten in geeigneter Weise von den riesigen Datenmengen abstrahieren. Die Statistik bietet hier die *empirischen Momente* an. Diese müssen allerdings, abhängig vom Modell und den Modellparametern, immer auf ihre Gültigkeit überprüft werden.

3.1 Zeitreihen

Eine *Zeitreihe* ist eine zeitlich geordnete Folge $(x_t)_{t \in T}$ von Beobachtungen einer Größe. Für jeden Zeitpunkt t einer *Parametermenge* T von Beobachtungszeitpunkten liegt dabei genau eine Beobachtung vor. Die Parametermenge ist im Allgemeinen eine endliche, diskrete Menge von gleichabständigen Zeitpunkten. Die Zeitreihe kann durch den *Plot* visualisiert werden. In dem Plot werden die Beobachtungen über die Zeit aufgetragen. Weitere Anregungen und Beispiele sind in [31] Kapitel 1.1 aufgeführt.

Die Abbildung 3.1 ist ein Plot, der die täglichen Kundenankünfte einer Tankstelle darstellt. Es ist zu erkennen, dass sich die Kundenankünfte wochenweise ähneln. Der Plot lässt schon eine gewisse Bewertung zu und ist damit ein erster Schritt zur Analyse einer Zeitreihe.

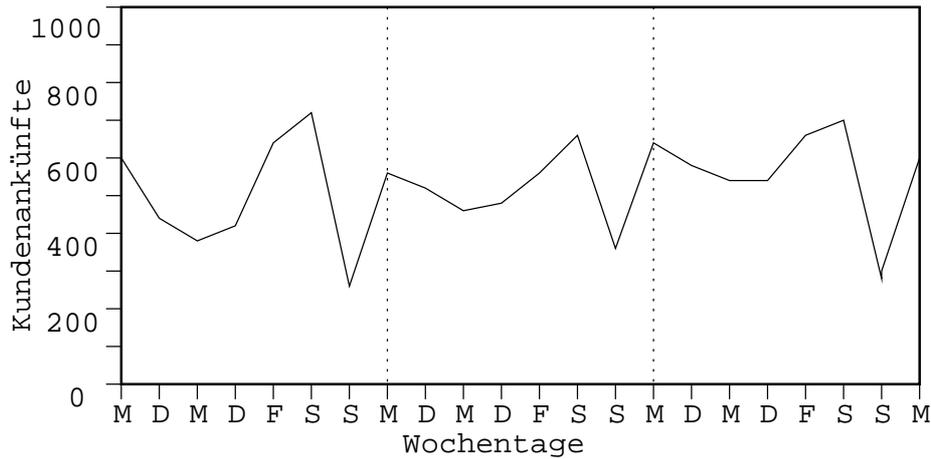


Abbildung 3.1: Tägliche Kundenankünfte an einer Tankstelle.

3.2 Zufallsvariablen

Eine *Zufallsvariable* X ist eine Abbildung, die jedem Ergebnis ω der Ergebnismenge Ω eine reelle Zahl $X(\omega) = x$ zuordnet:

$$X : \Omega \longrightarrow \mathbb{R} \quad . \quad (3.1)$$

Meistens werden Zufallsvariablen durch Großbuchstaben bezeichnet, wie X , Y oder Z . Mit Kleinbuchstaben, wie x , y oder z , werden die tatsächlich angenommenen Werte bezeichnet.

Zufallsvariablen können diskret oder stetig sein. Diskret bedeutet, dass die Zufallsvariable nur abzählbar unendlich viele Werte annehmen kann. Stetig bedeutet demnach, dass der Wertebereich der Zufallsvariablen kontinuierlich ist.

3.2.1 Diskrete Zufallsvariablen

Eine diskrete Zufallsvariable X lässt sich auf unterschiedliche Weise beschreiben.

$$\text{Wahrscheinlichkeitsverteilung : } \Pr(X = x_i) = p_i \quad (3.2)$$

$$\text{Wahrscheinlichkeitsfunktion : } f(x) = \begin{cases} p_1 & : x = x_1 \\ p_2 & : x = x_2 \\ \dots & : \dots \\ 0 & : \text{sonst} \end{cases} \quad (3.3)$$

$$\text{Verteilungsfunktion : } F(x) = \Pr(X \leq x) = \sum_{i: x_i \leq x} f(x_i) \quad (3.4)$$

Hierbei gilt für die Wahrscheinlichkeiten p_i zum einen $0 \leq p_i \leq 1$ und zum anderen $\sum p_i = 1$. Die Verteilungsfunktion $F(x)$ ist eine Treppenfunktion. Ist die diskrete Zufallsvariable endlich (k Realisierungen), dann gilt für $x < x_1$ $F(x) = 0$

und für $x > x_k$ gilt $F(x) = 1$. Sofern die diskrete Zufallsvariable nicht endlich ist, gilt $\lim_{x \rightarrow \infty} F(x) = 1$. Genauere Hinweise auf die Eigenschaften der Wahrscheinlichkeitsfunktion und der Verteilungsfunktion diskreter Zufallsvariablen sind in [16] Kapitel 5 und in [27] Kapitel 4.2 zu finden.

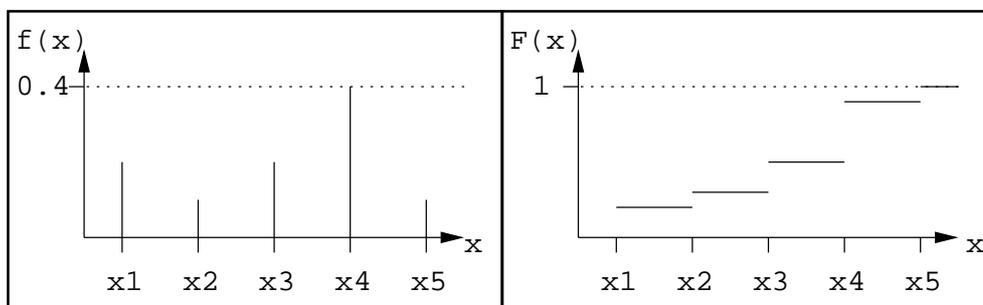


Abbildung 3.2: Das Stabdiagramm einer Wahrscheinlichkeitsfunktion und die dazugehörige Verteilungsfunktion.

3.2.2 Kontinuierliche Zufallsvariablen

Eine kontinuierliche (stetige) Zufallsvariable X besitzt eine (Wahrscheinlichkeits-) Dichtefunktion $f(x) \geq 0$. Für die Dichtefunktion $f(x)$ gilt:

$$\Pr(a \leq X \leq b) = \int_a^b f(x) dx \quad \text{und} \quad (3.5)$$

$$1 = \int_{-\infty}^{+\infty} f(x) dx \quad . \quad (3.6)$$

Die Verteilungsfunktion $F(x)$ einer kontinuierlichen Zufallsvariable ist definiert durch

$$F(x) = \Pr(X \leq x) = \int_{-\infty}^x f(t) dt \quad . \quad (3.7)$$

Die Ähnlichkeit der Verteilungsfunktionen diskreter und kontinuierlicher Variablen sind offensichtlich. Daher haben beide Verteilungsfunktionen auch ähnliche Eigenschaften. Die kontinuierliche Verteilungsfunktion $F(x)$ ist stetig und monoton wachsend mit Werten im Intervall $[0, 1]$. Für die Grenzen gilt

$$\lim_{x \rightarrow +\infty} F(x) = 1 \quad \text{und} \quad \lim_{x \rightarrow -\infty} F(x) = 0 \quad . \quad (3.8)$$

Eine detailliertere Einführung wird in [16] Kapitel 6 und in [27] Kapitel 4.2 gegeben.

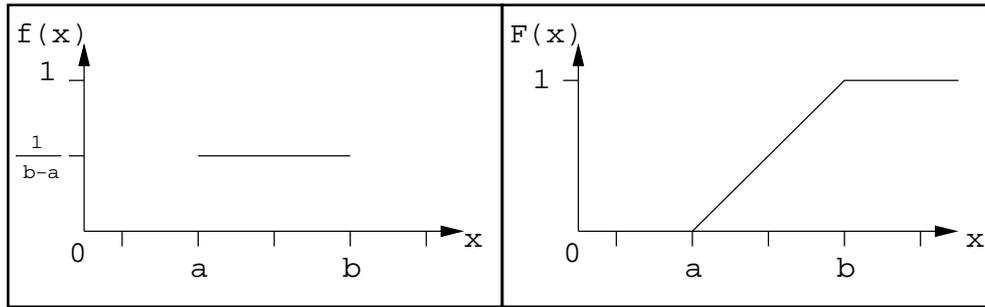


Abbildung 3.3: Die Dichte und Verteilungsfunktion einer kontinuierlichen Zufallsvariable ($[a, b]$ -gleichverteilt).

3.3 Parameter, Momente und empirische Momente

Die Daten x_1, \dots, x_n seien bei der Ziehung aus einer Zufallsvariablen X entstanden. Die Parameter seien hier die Kennwerte einer beliebigen, unbekanntem Verteilung der Zufallsvariablen. Die Zufallsvariable X kann durch ihre Momente beschrieben werden. Die Momente der Ordnung r der Zufallsvariablen X sind definiert durch $\mu_r = E[X^r]$. Die wichtigsten Momente sollen nun anhand der Daten x_1, \dots, x_n bestimmt werden. Die empirischen Momente sind eine konkrete Realisierung (Schätzung) der eigentlichen Momente. Das Wort empirisch bedeutet, dass es sich um Maßzahlen handelt, die aus konkreten Daten berechnet werden. In diesem Kapitel sollen nur die wichtigsten Momente genannt werden. In [31] werden die für die Praxis wichtigsten Methoden und Definitionen genannt. Kapitel 1.2 geht auf die besondere Situation bei der Auswertung von stationären Zeitreihen ein. In [16] werden die Momente wesentlich theoretischer vorgestellt. Auf die Parameter (bzw. Momente) diskreter Zufallsvariablen geht das Kapitel 5.2.3 ein. Kapitel 6.2 stellt die Parameter (bzw. Momente) stetiger Zufallsvariablen dar. Auf den Zusammenhang zwischen zwei Zufallsvariablen gehen die Kapitel 3.4 und Kapitel 8.5 ein. Hinweise zur praktischen Berechnung der Parameter hält Kapitel 2.2 fest. Kapitel 9 geht allgemeiner auf die Parameterschätzung ein. Ergänzende Literatur, die den Zusammenhang mit der Simulation deutlich macht, ist [25], [27] und [10].

3.3.1 Der Erwartungswert und das arithmetische Mittel

Das *arithmetische Mittel* einer Zeitreihe $(x_t)_{t=1, \dots, n}$:

$$\bar{x} = \frac{1}{n} \sum_{t=1}^n x_t \quad (3.9)$$

ist wohl das bekannteste empirische Moment. Mit dieser Definition lässt sich das arithmetische Mittel aus der Urliste bestimmen. Das arithmetische Mittel be-

schreibt den Schwerpunkt von Daten einer Zeitreihe. Wenn jedem Messwert eine Gewichtseinheit zugeordnet wird und die Messwerte entsprechend ihrer Größe auf einer Geraden angeordnet sind, ist das arithmetische Mittel der Schwerpunkt dieser Geraden. Eine andere Formulierung in diesem Sinne ist: Das arithmetische Mittel ist die zentrale Lage der Messwerte.

Der *Erwartungswert* einer Zufallsvariablen ist eine Maßzahl für das Zentrum einer Verteilung. Bei einer diskreten Zufallsvariablen X mit der Wahrscheinlichkeitsfunktion $f(x)$ ist er durch

$$\mu = E(X) = \sum_{i \geq 1}^n x_i f(x_i) \quad (3.10)$$

definiert. Bei einer stetigen Zufallsvariablen X mit der Dichte $f(x)$ ist der Erwartungswert durch

$$\mu = E(X) = \int_{-\infty}^{+\infty} x f(x) dx \quad (3.11)$$

definiert.

Obwohl das arithmetische Mittel und der Erwartungswert durch den (gewichteten) Mittelwert bestimmt werden, ist deren Funktion deutlich zu unterscheiden. Das arithmetische Mittel ist eine konkrete Realisierung und basiert auf den tatsächlich vorhandenen Daten. Der Erwartungswert ist jedoch für eine Zufallsvariable definiert und nicht für die Daten einer Zeitreihe. Konkret: Der Erwartungswert $E(X)$ charakterisiert das Verhalten eines Zufallsexperiments. Das arithmetische Mittel \bar{x} beschreibt den Schwerpunkt von Daten einer Zeitreihe. Das arithmetische Mittel ist der empirische Erwartungswert.

3.3.2 Die Varianz

Neben dem arithmetischen Mittel interessiert ebenfalls ein Maß, das die Abweichung der Messwerte von diesem Mittelwert beschreibt. Mit $x_i - \bar{x}$ lässt sich die Abweichung eines einzelnen Messwertes bestimmen. Allerdings werden sowohl positive als auch negative Abweichungen erwartet. Damit sich diese Abweichungen in der Summe nicht gegenseitig aufheben, werden die Abweichungen erst quadriert und dann summiert. Nun sollte diese Summierung allerdings noch auf die Anzahl der Messwerte normiert werden. Eine genauere Betrachtung zeigt jedoch, dass es besser ist, die Summe auf die Anzahl der Freiheitsgrade zu normieren. Da $\sum(x_i - \bar{x}) = 0$ ist, gibt es bei n Messwerten nur $n - 1$ Freiheitsgrade. Daher lautet die Formel zur Berechnung der *Stichprobenvarianz*:

$$s^2 = \frac{1}{n - 1} \sum_{i=1}^n (x_i - \bar{x})^2 \quad (3.12)$$

Eine mathematische Bestätigung dieses gedanklichen Ansatzes ist in [10] Kapitel 3.2 beschrieben. Die Stichprobenvarianz ist eine Realisierung der empirischen Varianz.

Die *Varianz* einer Zufallsvariablen lässt sich als die erwartete quadratische Abweichung beschreiben:

$$\sigma^2 = \text{Var}(X) = E(X - E(X))^2 = E(X^2) - (E(X))^2 \quad (3.13)$$

Der hintere Teil der Formel eignet sich besonders gut für eine iterative Berechnung.

3.3.3 Die Standardabweichung und der Variationskoeffizient

Infolge des Quadrierens hat die Varianz nicht dieselbe Maßeinheit wie der Erwartungswert. Deshalb ist die *Standardabweichung* oder *Streuung* als die Wurzel aus der Varianz definiert.

$$\sigma = +\sqrt{\sigma^2} \quad \text{bzw.} \quad s = +\sqrt{s^2} \quad (3.14)$$

Um einen Vergleich mit unterschiedlichen Streuungen durchzuführen, wird ein maßstabsunabhängiges Streuungsmaß benötigt. Aus diesem Grund wird die Standardabweichung durch den Erwartungswert geteilt, damit eine geeignete Normierung entsteht – der *Variationskoeffizient*:

$$v = \frac{\sigma}{\mu} \quad \text{bzw.} \quad v = \frac{s}{\bar{x}} \quad (3.15)$$

3.3.4 Die Kovarianz und der Korrelationskoeffizient

In vielen Fällen wird bei statistischen Verfahren die Unabhängigkeit von Zufallsvariablen vorausgesetzt. Die Zufallsvariablen X_1, \dots, X_n heißen *unabhängig*, wenn für alle x_1, \dots, x_n gilt

$$\Pr[X_1 \leq x_1, \dots, X_n \leq x_n] = \Pr[X_1 \leq x_1] \cdot \dots \cdot \Pr[X_n \leq x_n] \quad (3.16)$$

Es existieren viele verschiedene Abhängigkeitsformen. Bei der Zeitreihenanalyse hat die *lineare Abhängigkeit* die größte Bedeutung. Die *Kovarianz* ist ein Maßzahl zur Beschreibung der linearen Abhängigkeit. Sie ist für die Zufallsvariablen X und Y bestimmt durch

$$\begin{aligned} \text{Cov}(X, Y) &= E([X - E(X)][Y - E(Y)]) \\ &= E(XY) - E(X)E(Y) \quad . \end{aligned} \quad (3.17)$$

Um aber, wie bei der Standardabweichung, verschiedene Kovarianzen vergleichen zu können, benötigt man eine geeignete Normierung. Dies wird durch den *Korrelationskoeffizienten* berücksichtigt:

$$\rho = \rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)}\sqrt{\text{Var}(Y)}} \quad (3.18)$$

Der Wertebereich des Korrelationskoeffizienten ist $-1 \leq \rho \leq 1$.

Eine Realisierung (bzw. eine Schätzung) der Kovarianz ist die *empirische Kovarianz* für n Beobachtungspaare (x_i, y_i) . Sie ist definiert durch

$$c = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) \quad . \quad (3.19)$$

Eine sehr intuitive Erklärung der Formel für die empirische Kovarianz ist in [16] Kapitel 3.4.1 dargestellt.

Der *empirische Korrelationskoeffizient* (auch Bravais-Pearson Korrelationskoeffizient) ist bestimmt durch

$$\begin{aligned} r &= \frac{c}{s_X s_Y} \\ &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}} \\ &= \frac{\sum_{i=1}^n x_i y_i - n \bar{x} \bar{y}}{\sqrt{(\sum_{i=1}^n x_i^2 - n \bar{x}^2)(\sum_{i=1}^n y_i^2 - n \bar{y}^2)}} \quad . \quad (3.20) \end{aligned}$$

Der letzte Teil der Formel ist zur Berechnung besonders günstig, da die einzelnen Summen und Mittelwerte bei nur einem Durchlauf der Ergebniswerte parallel bestimmt werden können. Der empirische Korrelationskoeffizient hat den Wertebereich $-1 \leq r \leq 1$.

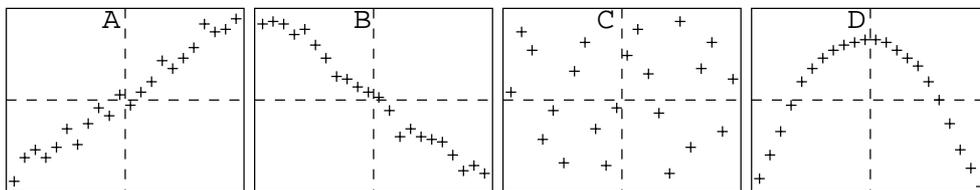


Abbildung 3.4: Streudiagramme bei verschiedenen Beobachtungen.

Wenn die Messwerte annähernd auf einer Geraden mit positiver Steigung angeordnet sind, gilt $r > 0$ und es besteht ein gleichsinniger linearer Zusammenhang: *positive Korrelation* (Abb.: 3.4 A). Wenn die Messwerte annähernd auf einer Geraden mit negativer Steigung angeordnet sind, gilt $r < 0$ und es besteht ein gegensinniger linearer Zusammenhang: *negative Korrelation* (Abb.: 3.4 B). Für $r = 0$ besteht kein linearer Zusammenhang und die Messwerte sind *unkorreliert* (Abb.: 3.4 C und D). Die Abbildungen 3.4 D verdeutlichen jedoch, dass unkorreliert nicht gleichbedeutend mit unabhängig ist. Die Messwerte sind eindeutig abhängig, aber nicht linear abhängig. Daher sind die Messwerte sowohl unkorreliert, als auch abhängig. Die Aussagekraft des Korrelationskoeffizienten muss daher immer hinterfragt werden.

Im Allgemeinen wird noch zwischen Autokovarianz (bzw. Autokorrelation) und Kreuzkovarianz (bzw. Kreuzkorrelation) unterschieden. Das Präfix “Auto”

wird benutzt, wenn man sich auf die lineare Beziehung innerhalb einer Zeitreihe bezieht. Das Präfix “Kreuz” wird dementsprechend benutzt, wenn man sich auf die lineare Beziehung zwischen verschiedenen Zeitreihen bezieht.

In [31] Kapitel 1.2 wird speziell auf die Kovarianz-Schätzung bei der Analyse von Zeitreihen eingegangen. In dieser Situation sind nicht zwei Zufallsvariablen gegeben, sondern die aufeinander folgenden Wertepaare (x_1, x_2) , (x_2, x_3) , \dots , (x_{n-1}, x_n) sollen verglichen werden. Es wird Stationarität (siehe 3.4) vorausgesetzt. Daher kann auch ein arithmetisches Mittel \bar{x} und eine Standardabweichung s vorausgesetzt werden und in beiden Subtraktionen in der Summe zur Berechnung der Kovarianz benutzt werden. Die Normierung durch n und nicht durch $n - \tau$ hat statistische Vorteile, auf die in [31] Kapitel 6 näher eingegangen wird. τ sei der Zeitabstand zwischen den zu vergleichenden Werten. Die Autokovarianzfunktion (c_τ) einer Zeitreihe ist somit

$$c_\tau = \frac{1}{n} \sum_t (x_t - \bar{x})(x_{t-\tau} - \bar{x}) \quad (3.21)$$

mit dem Zeitabstand $\tau = -(n-1), \dots, -1, 0, 1, \dots, (n-1)$. Die Autokorrelationsfunktion (r_τ) hat in der Praxis größere Bedeutung, da die Normierung wesentliche Vorteile bringt. Sie ist definiert durch

$$r_\tau = \frac{c_\tau}{c_0} \quad (3.22)$$

Der Graph der Autokorrelationsfunktion (r_τ über τ) ist das Korrelogramm (siehe Abb.: 3.5). An dem Korrelogramm lassen sich die zeitlichen (linearen) Abhängigkeiten der beobachteten Reihe ablesen.

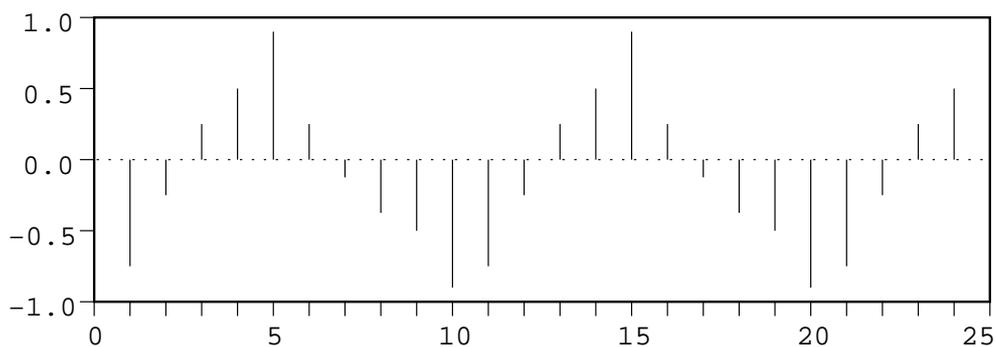


Abbildung 3.5: Das Korrelogramm einer periodischen Zeitreihe. Lineare Abhängigkeiten treten verstärkt alle 5 Zeiteinheiten auf.

Man sollte sich genau vor Augen halten, was die Kovarianz eigentlich aussagt bzw. was sie nicht aussagt. Die im folgenden aufgeführten Sätze sollen hierbei behilflich sein.

1. Wenn zwischen den Zufallsvariablen X und Y eine lineare Transformation möglich ist, genau dann ist deren Korrelationskoeffizient $\rho = 1$ oder $\rho = -1$.

$$Y = aX + b \Leftrightarrow |\rho(X, Y)| = 1 \quad (3.23)$$

2. Sind die Zufallsvariablen X und Y unabhängig, dann sind sie auch unkorreliert.

$$\begin{aligned} & (X \text{ und } Y \text{ sind unabhängig}) \\ \Rightarrow & (X \text{ und } Y \text{ sind unkorreliert}) \end{aligned} \quad (3.24)$$

3. Aber Vorsicht: Sind die Zufallsvariablen X und Y unkorreliert, müssen sie nicht unbedingt unabhängig sein. Unkorreliert bedeutet nur, dass sie nicht linear abhängig sind. Aber es gibt noch weitere Arten der Abhängigkeit, wie Abbildung 3.4 D verdeutlicht.

$$\begin{aligned} & (X \text{ und } Y \text{ sind unabhängig}) \\ \neq & (X \text{ und } Y \text{ sind unkorreliert}) \end{aligned} \quad (3.25)$$

3.3.5 Median

Ein robustes Lagemaß ist der Median x_{med} . Der Einfluß eines Extremwertes oder Ausreißers ist begrenzt, im Gegensatz zum arithmetischen Mittel. Er ist ein Spezialfall der α -Quantile: $x_{med} = x_{0.5}$. Die geordnete Liste $x_{(1)} \leq \dots \leq x_{(n)}$ entsteht durch aufsteigende Sortierung der Werte x_1, \dots, x_n . Der Median ist definiert durch

$$x_{med} = \begin{cases} x_{\frac{n+1}{2}} & \text{für } n \text{ ungerade} \\ \frac{1}{2}(x_{\frac{n}{2}} + x_{\frac{n}{2}+1}) & \text{für } n \text{ gerade} \end{cases} \quad (3.26)$$

In vielen Fällen ist der Median eine gute Alternative zum arithmetischen Mittel, denn der Median ist bei ungeradem n ein Wert, der tatsächlich beobachtet wurde.

3.3.6 Modus

Der *Modus* ist der Wert mit der größten Häufigkeit. Bei stetigen Zufallsvariablen ist er der Wert, für den die Wahrscheinlichkeitsdichtefunktion maximal ist. Im Gegensatz zum arithmetischen Mittel und Median kann es mehrere Modi oder auch keinen Modus geben. Besitzt die Wahrscheinlichkeitsdichtefunktion einer Zufallsvariablen genau einen Modus, nennt man sie unimodal. Besitzt sie zwei bzw. mehrere Modi, nennt man sie bimodal bzw. multimodal.

3.3.7 Quantile

Bei der Auswertung von Zeitreihen ist die Streuung der Messwerte um ihr Zentrum interessant. Die *Quantile* sind geeignete Maßzahlen, um die Streuung der Daten zu charakterisieren. Das p -Quantil x_p trennt die Daten so in zwei Teile, dass $p \cdot 100\%$ der Daten darunter und $(1 - p) \cdot 100\%$ darüber liegen. Daher ist der Median x_{med} , wie oben schon erwähnt, das 0.5-Quantil $x_{0.5}$. *Quartile* (Schreibweise beachten!) sind durch vier Datenbereiche definiert, die durch $x_{0.25}$, $x_{0.5}$ und $x_{0.75}$ getrennt werden. Weitere häufig benutzte Quantile sind die *Dezile* mit $p = 0.1, 0.2, \dots, 0.9$.

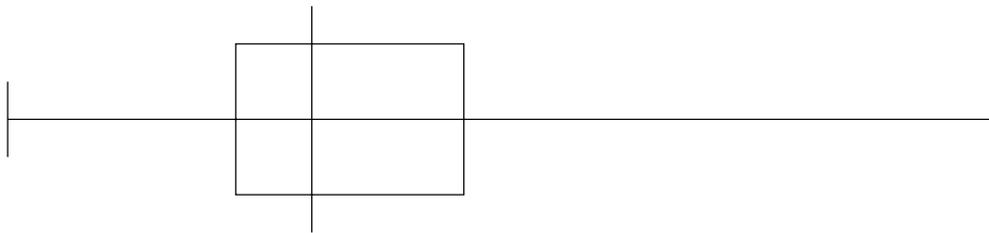


Abbildung 3.6: Der Box-Plot visualisiert x_{min} , x_{max} , $x_{0.25}$ bis $x_{0.75}$ und $x_{0.5}$.

Der *Box-Plot* ist eine einfache und nützliche Möglichkeit der Visualisierung. Man zeichnet eine waagerechte Strecke, die durch den minimalen Messwert x_{min} und den maximalen Messwert x_{max} begrenzt ist. Nun zeichnet man eine Box um die Messwerte im Bereich $x_{0.25}$ bis $x_{0.75}$. Der Mittelwert $x_{0.5}$ wird durch eine horizontale Strecke gekennzeichnet. Diese einfache Zeichnung beschreibt nun qualitativ die Verteilungsdichte der zugehörigen Zufallsvariablen. Es gibt noch einige Variationen des Box-Plots, die in [27] Kapitel 6.4.3 und in [16] Kapitel 2.2.2 beschrieben sind.

3.3.8 Konfidenzintervalle

Die *Punktschätzung* liefert einen Wert, der in der Regel von dem Erwartungswert abweicht. Zum Beispiel ist das arithmetische Mittel ein Punktschätzer für den Erwartungswert. Daher ist es sinnvoll, ein Maß für die Präzision eines Schätzverfahrens anzugeben. Die $(1 - \alpha)$ -*Konfidenzintervalle* sind ein solches Maß. α ist die Irrtumswahrscheinlichkeit. Gesucht ist ein Intervall $[g_u, g_o]$ mit $\Pr(g_u \leq \mu \leq g_o) = 1 - \alpha$. Eine genaue Definition ist in [16] Kapitel 9.4 zu finden. Wir beschränken uns hier auf Konfidenzintervalle für den Erwartungswert.

Es seien X_1, \dots, X_n unabhängige Wiederholungen einer $N(\mu, \sigma^2)$ verteilten Zufallsvariablen. N sei die Normalverteilung und $N(0, 1)$ sei die Standardnormalverteilung. Für den Erwartungswert μ soll ein Konfidenzintervall bestimmt

werden. Der Punktschätzer \bar{X} , das arithmetische Mittel, ist normalverteilt und lässt sich mit dem zentralen Grenzwertsatz standardisieren:

$$\frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}} \sim N(0, 1) \quad (3.27)$$

Ist $z_{1-\frac{\alpha}{2}}$ das $(1 - \frac{\alpha}{2})$ -Quantil der Standardnormalverteilung, lässt sich ein zweiseitig beschränkter Bereich angeben, in dem μ mit der Wahrscheinlichkeit $(1 - \alpha)$ liegt:

$$\begin{aligned} 1 - \alpha &= \Pr(-z_{1-\frac{\alpha}{2}} \leq \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}} \leq z_{1-\frac{\alpha}{2}}) \\ &= \Pr(\bar{X} - z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + z_{1+\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}) \quad . \end{aligned} \quad (3.28)$$

Aus der letzten Gleichung lässt sich das $(1 - \alpha)$ -Konfidenzintervall ablesen:

$$[\bar{X} - z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}, \bar{X} + z_{1+\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}] \quad . \quad (3.29)$$

Damit ist die Breite des Konfidenzintervalls gegeben durch

$$b_{KI} = 2z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}} \quad . \quad (3.30)$$

Die Breite b_{KI} hängt also von der Irrtumswahrscheinlichkeit α , der Standardabweichung σ und dem Stichprobenumfang n ab. Diese Abhängigkeiten hätte man auch intuitiv gewählt.

Es wurde bei den bisherigen Betrachtungen immer vernachlässigt, dass meistens nicht die Standardabweichung σ , sondern nur die empirische Standardabweichung s bekannt ist. Da der Punktschätzer s nicht exakt die Standardabweichung σ beschreibt, muss ein weiterer Fehler berücksichtigt werden. Die Variable $S = \sqrt{\frac{\sum_{i=1}^n (X_i - \bar{X})^2}{(n-1)}}$, die durch den Punktschätzer s realisiert wird, besitzt eine t -Verteilung mit $n - 1$ Freiheitsgraden (vgl. [16], Kapitel 6.3.3). Dies führt zu folgender Gleichung:

$$\begin{aligned} 1 - \alpha &= \Pr(-t_{n-1, 1-\frac{\alpha}{2}} \leq \frac{\bar{X} - \mu}{\frac{S}{\sqrt{n}}} \leq t_{n-1, 1-\frac{\alpha}{2}}) \\ &= \Pr(\bar{X} - t_{n-1, 1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}} \leq \mu \leq \bar{X} + t_{n-1, 1+\frac{\alpha}{2}} \frac{S}{\sqrt{n}}) \quad . \end{aligned} \quad (3.31)$$

Das $(1 - \alpha)$ -Konfidenzintervall ist in diesem Fall:

$$[\bar{X} - t_{n-1, 1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}}, \bar{X} + t_{n-1, 1+\frac{\alpha}{2}} \frac{S}{\sqrt{n}}] \quad . \quad (3.32)$$

Damit ist die Breite des Konfidenzintervalls

$$b_{KI} = 2t_{n-1, 1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}} \quad . \quad (3.33)$$

Die Breite b_{KI} hängt also immer noch von der Irrtumswahrscheinlichkeit α , der empirischen Standardabweichung S und dem Stichprobenumfang n ab. Die Dichtefunktion der t -Verteilung ist etwas breiter als die Dichtefunktion der Standardnormalverteilung, daher ist auch das Konfidenzintervall breiter, das den Fehler durch den Punktschätzer s berücksichtigt.

Einige Beispiele zu Konfidenzintervallen findet man in [27] Kapitel 4.5. Außerdem ist in [27] Anhang T.1 eine Tabelle aufgeführt, in der man Werte für $t_{\nu,\gamma}$ ablesen kann. In [16] Kapitel 9.4.1 wird zwischen einseitigen und zweiseitigen Konfidenzintervallen unterschieden. Ein approximatives Konfidenzintervall für beliebige Verteilungen und ein Konfidenzintervall für die Varianz wird ebenfalls vorgestellt.

3.4 Stationarität

Stationarität ist eine wichtige Eigenschaft von Modellen, die in vielen Fällen eine detaillierte Auswertung erst sinnvoll macht. Ausgehend von stochastischen Prozessen werden in der mathematischen Sichtweise Definitionen für die schwache und die strenge Stationarität gegeben. In der simulativen Sichtweise wird die Zufallsverteilung eines bestimmten Modellzeitpunktes stärker betont. Die Definition der simulativen Sichtweise ist allerdings identisch zur Definition der strengen Stationarität. Die spezielle Situation bei Markov-Modellen wird in der analytischen Sichtweise berücksichtigt.

3.4.1 Mathematische Sichtweise

Ein *stochastischer Prozess* ist ein dynamischer Vorgang mit Zufallscharakter. Er ist definiert durch eine Folge $(X_t)_{t \in T}$ von Zufallsvariablen X_t . Der Index t (oder Zeitparameter) ist Element einer höchstens abzählbar unendlichen Indexmenge T . Ist $T = \mathbb{N}$, dann ist der stochastische Prozess zeitdiskret. Ist $T = \mathbb{R}$, dann ist der stochastische Prozess zeitkontinuierlich.

Die *Mittelwertfunktion* $\mu(t)$ (oder $\mu_t, \mu_X(t)$) des stochastischen Prozesses (X_t) ist gegeben durch

$$\mu(t) := E[X_t] \quad . \quad (3.34)$$

Die *Varianzfunktion* $\sigma^2(t)$ (oder $\sigma_t^2, \sigma_X^2(t)$) ist gegeben durch

$$\sigma^2(t) := Var[X_t] \quad . \quad (3.35)$$

Die *Kovarianzfunktion* $\gamma(s, t)$ ordnet jedem Paar von Zeitpunkten $s, t \in T$ die Kovarianz der Zufallsvariablen X_s und X_t zu:

$$\gamma(s, t) := Cov[X_s, X_t] = E[(X_s - \mu_X(s))(X_t - \mu_X(t))] \quad . \quad (3.36)$$

Entsprechend ist die Korrelationsfunktion $\varrho(s, t)$ definiert:

$$\varrho(s, t) := \rho[X_s, X_t] = \frac{\text{Cov}[X_s, X_t]}{\sqrt{\sigma_X^2(s)\sigma_X^2(t)}}. \quad (3.37)$$

Diese Definitionen sind alle keine wirklich neuen Kenngrößen, sondern nur die Übertragung schon genannter Kenngrößen auf stochastische Prozesse. Genauere Hinweise sind in [31] Kapitel 2.1 und [27] Kapitel 4.3 zu finden.

Jede Zufallsvariable X_t eines stochastischen Prozesses kann durch ihre Kenngrößen, wie Erwartungswert, Varianz, etc. beschrieben werden. In der Regel ist man aber an den gemeinsamen Kenngrößen der Zufallsvariablen interessiert. Dieses Vorgehen ist nur dann sinnvoll, wenn die einzelnen Kenngrößen der Zufallsvariablen nicht zu sehr voneinander abweichen. Man könnte auch sagen, dass der Prozess stationär ist. Die folgenden Definitionen sollen eine formale Basis geben, bevor wieder das Praxisrelevante im Vordergrund steht.

Ein stochastischer Prozeß $(X_t)_{t \in T}$ heißt

- *mittelwertstationär*, wenn μ_t konstant ist:

$$\mu_t = \mu \quad \text{für alle } t \in T \quad (3.38)$$

- *varianzstationär*, wenn σ_t^2 konstant ist:

$$\sigma_t^2 = \sigma^2 \quad \text{für alle } t \in T \quad (3.39)$$

- *kovarianzstationär*, wenn $\gamma(s, t)$ nur von der Entfernung $s - t$ abhängt:

$$\gamma(s, t) = \gamma(s - t) \quad \text{für alle } s, t \in T \quad (3.40)$$

- *schwach stationär*, wenn er mittelwert-, varianz- und kovarianzstationär ist.
- *streng stationär*, wenn die gemeinsame Verteilungsfunktion jedes endlichen Systems von Zufallsvariablen $(X_{t_1}, X_{t_2}, \dots, X_{t_n})$ des Prozesses identisch mit der gemeinsamen Verteilungsfunktion des um s Zeitpunkte verschobenen Systems $(X_{t_1+s}, X_{t_2+s}, \dots, X_{t_n+s})$ ist.

Bei einem schwach stationären Prozess sind also nur die Mittelwertfunktion, die Varianzfunktion und die Kovarianzfunktion invariant gegen Zeitverschiebungen. Bei einem streng stationären Prozess müssen aber alle Kenngrößen invariant gegen Zeitverschiebungen sein. Dies ist in der Praxis kaum überprüfbar, daher wird bei vielen praktischen Problemen nur die schwache Stationarität bestimmt. In der Literatur wird meistens die "schwache Stationarität" durch "Stationarität" abgekürzt. Die Definitionen sind aus [31] Kapitel 2.2 entnommen.

In [31] Kapitel 2.3.3 bis Kapitel 2.3.5 werden einige stochastische Prozesse genauer klassifiziert. Es handelt sich hierbei um Moving-Average-Prozesse (MA), Autoregressiven-Prozesse (AR) und Autoregressiven-Moving-Average-Prozesse (ARMA). Für diese Klassen lassen sich Stationaritätsbedingungen formulieren. Leider ist dies für die allgemeineren stochastischen Prozesse nicht möglich.

3.4.2 Simulative Sichtweise

Sei Y_1, Y_2, \dots ein stochastischer Prozess. Dann sei $F_i(y|I) = \Pr(Y_i \leq y|I)$ die Verteilungsfunktion der Zufallsvariablen Y_i und dem Startzustand I . Im Allgemeinen ist $F_i(y|I)$ verschieden für unterschiedliche Werte i und unterschiedliche Startzustände I . $f_i(y|I)$ sei die zugehörige Dichtefunktion. Die Dichtefunktion zeigt, wie die Werte bei unterschiedlichen Simulationsläufen und gleichen Bedingungen schwanken. Für konstante y und I ist die Folge $F_1(y|I), F_2(y|I), \dots$

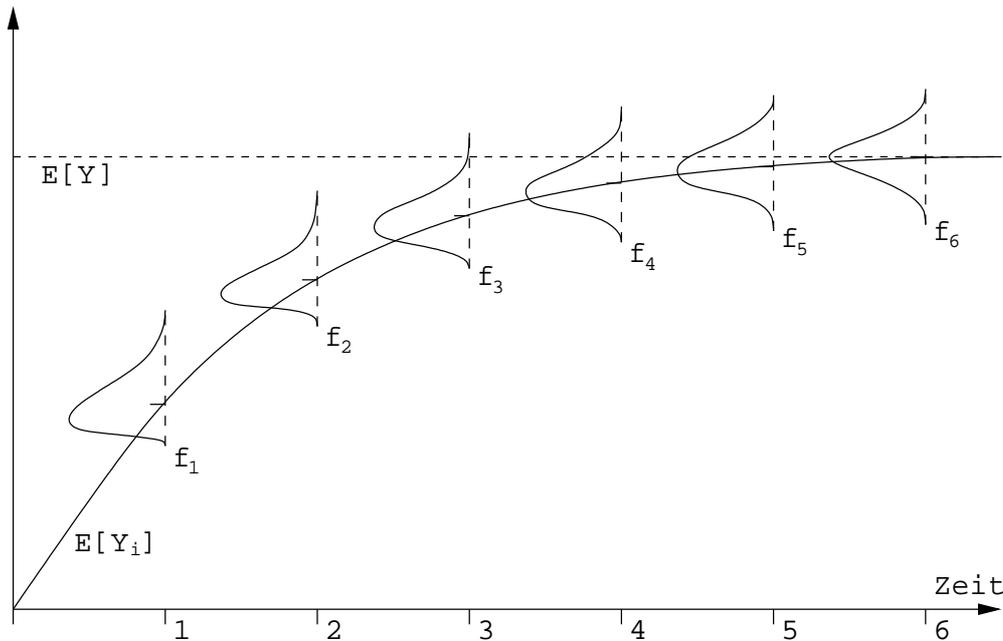


Abbildung 3.7: Die Dichtefunktionen f_i eines stochastischen Prozesses. Die stationäre Zustandsverteilung F muss nicht normal verteilt sein.

nur eine Sequenz von Zahlen. Wenn $F_i(y|I) \rightarrow F(y)$ mit $i \rightarrow \infty$ für alle y und jeden Startzustand I gilt, wird $F(y)$ die *stationäre Zustandsverteilung* genannt. Genau genommen ist die stationäre Zustandsverteilung nur für $i \rightarrow \infty$ definiert, in der Praxis wird aber davon ausgegangen, dass ab einem konkreten Zeitpunkt k die Unterschiede der Verteilungen $F_i(y|I)$ mit $k \leq i$ vernachlässigbar sind (vgl. [27], Kap. 9.2).

Wenn sich das Modellverhalten mit der Zeit einer stationären Zustandsverteilung nähert, nennt man das Modell *stationär* (stable model: [25], Kapitel 24.3).

Ist ein Modell stationär, bedeutet dies nicht, dass nach einer gewissen Zeit sein Zustand konstant ist, sondern dass die Zustandsverteilung konstant ist.

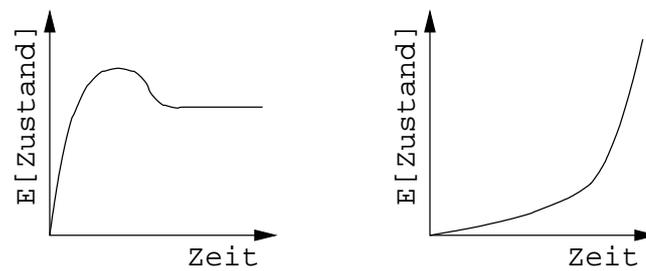


Abbildung 3.8: Ein stabiles und ein instabiles Modell.

In den meisten Fällen sind nur die Messwerte interessant, die bei stationärer Zustandsverteilung beobachtet wurden. Wenn sich das Modellverhalten nur noch geringfügig von der stationären Zustandsverteilung unterscheidet, hat die Simulation die *stationäre Phase* erreicht. Die Anfangsphase, in der sich das Modell bis zur stationären Zustandsverteilung einschwingt, wird *transiente Phase* genannt.

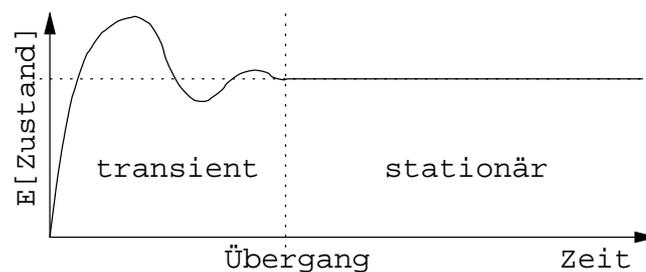


Abbildung 3.9: Die transiente und die stationäre Phase.

Hier lassen sich nun zwei wichtige Probleme erkennen. Zum einen ist die stationäre Zustandsverteilung von größter Bedeutung (steady state performance). Daher ist ihre Berechnung (Schätzung) aus den Daten einer gemessenen Zeitreihe sehr wichtig. Da hier die Messwerte aus der transienten Anfangsphase stören, ist es zum anderen sehr wichtig, den Anfang der stationären Phase aus den Daten einer Zeitreihe zu bestimmen (transient removal).

3.4.3 Analytische Sichtweise

Eine weitere Klasse bilden die schon erwähnten Markov-Prozesse. Für sie lassen sich Stationaritätsbedingungen formulieren ([7], Kapitel 2.1.1, Kapitel 2.2.3 und Kapitel 2.3.1). Im Zusammenhang mit Markov-Prozessen wird oft von der *stationären Zustandsverteilung* gesprochen. Für zeitdiskrete Markov-Ketten (Prozesse)

ist sie zum Beispiel definiert durch:

$$\pi_j = \lim_{m \rightarrow \infty} \pi_j^{(m)} \quad . \quad (3.41)$$

$\pi_j^{(m)}$ ist die Wahrscheinlichkeit, dass die zeitdiskrete Markov-Kette am Zeitpunkt (Schritt) m im Zustand j ist. Die stationäre Zustandsverteilung π_j ist unabhängig von der Zeit. Die Ähnlichkeiten mit der simulativen Sichtweise sind offensichtlich.

3.5 Ergodizität

Zum Schätzen der Parameter $E[X]$, $Var[X]$ und $Cov[X, Y]$ wird im Allgemeinen vorausgesetzt, dass die Zufallsvariablen X, Y unabhängig und identisch verteilt sind. Unter diesen Bedingungen sind die empirischen Schätzwerte für diese Parameter *konsistent* bzw. *erwartungstreu*. Erwartungstreue bedeutet, dass der Erwartungswert des Schätzers gleich dem eigentlichen Parameter ist. Zum Beispiel muss für einen Mittelwertschätzer $\tilde{\mu}$ die Bedingung $E[\tilde{\mu}] = \mu$ gelten, damit er erwartungstreu ist (vgl. [10], Kapitel 3.2). Eine Folge von Schätzfunktionen ist konsistent, wenn Konvergenz im quadratischen Mittel vorliegt. Bei der Konvergenz wird, im Gegensatz zur Erwartungstreue, also noch die Varianz der Schätzfunktion berücksichtigt. Genauere Definitionen und der Zusammenhang zwischen Erwartungstreue und Konsistenz werden in [31] Anhang C.6 beschrieben.

Bei der *Ensemble-Analyse* wird versucht, Aussagen über den Modellzustand zu einem festen Zeitpunkt t^* zu erhalten. Hierbei werden mehrere Simulationsläufe durchgeführt und die verschiedenen Messwerte zum Zeitpunkt t^* bearbeitet. Dieses Vorgehen wird auch *Querschnittsanalyse* genannt. Die *Zeitreihenanalyse* ist hingegen eine *Längsschnittsanalyse*. Die einzelnen Messwerte sind eine Folge von Beobachtungen zu verschiedenen Zeitpunkten. Diese müssen als Realisierung verschiedener, abhängiger Zufallsvariablen betrachtet werden. Nun stellt sich die Frage, wann die Querschnittsanalysen durch eine Längsschnittsanalyse ersetzt werden können. Konkret: Wann können Schätzfunktionen auch auf Realisierungen von abhängigen Zufallsvariablen angewendet werden?

Die *Ergodizität* kann als Konsistenz bestimmter Schätzfunktionen bei abhängigen Zufallsvariablen interpretiert werden. Wenn zeitliche Mittelwerte (Längsschnittsanalyse) gegen Ensemblemittelwerte (Querschnittsanalyse) konvergieren, heißt das zugehörige Modell ergodisch.

Sei (X_t) ein (schwach) stationärer Prozess mit dem Prozessmittel μ und der Kovarianzfunktion (γ_t) .

Definition: (X_t) heißt *mittelwertergodisch*, wenn

$$\lim_{N \rightarrow \infty} E\left[\left(\frac{1}{N} \sum_{t=1}^N X_t - \mu\right)^2\right] = 0 \quad . \quad (3.42)$$

Definition: (X_t) heißt *kovarianzergodisch*, wenn für alle τ gilt

$$\lim_{N \rightarrow \infty} E\left[\left(\frac{1}{N} \sum_{t=1}^N (X_t - \mu)(X_{t+\tau} - \mu) - \gamma_\tau\right)^2\right] = 0 \quad . \quad (3.43)$$

Definition: Die Folge (γ_t) heißt *absolut summierbar*, wenn die Folge der Partialsummen

$$s_t = |\gamma_{-t}| + |\gamma_{-t+1}| + \dots + |\gamma_0| + |\gamma_1| + \dots + |\gamma_t| \quad (3.44)$$

für $t \rightarrow \infty$ gegen einen endlichen Grenzwert konvergiert.

Satz: Ist (γ_t) absolut summierbar, so ist (X_t) mittelwertergodisch.

Diese Definitionen und der Satz können genutzt werden, um festzustellen, ob Schätzfunktionen auf eine Zeitreihe angewendet werden dürfen, ohne ihre Konsistenz zu verlieren. Ist zum Beispiel (X_t) ein stationärer und mittelwertergodischer Prozess, ist \bar{x} eine konsistente Schätzung für μ . Der Satz hilft bei der Frage, ob ein Prozess mittelwertergodisch ist. In [31] Kapitel 5 werden noch mehrere Sätze aus dem Bereich Ergodizität vorgestellt. Die Schätzung des Erwartungswertes (Kapitel 5.2), der Kovarianzfunktion (Kapitel 5.3) und der Korrelationsfunktion (Kapitel 5.4) wird sehr ausführlich beschrieben.

3.6 Stichprobenvergleich

In vielen Fällen interessiert, ob zwei Stichproben eine identische Zufallsverteilung besitzen. Natürlich kann diese Frage nicht mit "Ja" oder "Nein" beantwortet werden. Vielmehr sollte die Wahrscheinlichkeit ermittelt werden, mit der eine identische Zufallsverteilung vorausgesetzt werden kann.

3.6.1 χ^2 -Test

Der χ^2 -Test vergleicht eine Stichprobe mit der Dichtefunktion einer Zufallsvariablen. Aus der Stichprobe wird ein Histogramm erstellt. Die Höhe dieses Histogramms wird mit der Dichtefunktion der Zufallsvariablen in Verbindung gebracht, indem die Differenz zwischen diesen beiden Größen für jedes Intervall

des Histogramms bestimmt wird. Diese Differenzen werden geeignet normiert, sodass ihre Summe die χ^2 -Zufallsverteilung besitzt, wenn die Stichprobe so verteilt ist, wie die Zufallsverteilung. Mit geeigneten Tabellen und der Wahl eines Konfidenzlevels kann nun der kritische Punkt der χ^2 -Verteilung bestimmt werden. Dieser Punkt kann als Grenzwert betrachtet werden, an dem die Annahme der identischen Verteilung angenommen oder verworfen wird. Eine detaillierte Beschreibung des χ^2 -Test ist in [27] Kapitel 6.6.2 nachzulesen.

Der χ^2 -Test kann auf den Vergleich von zwei Stichproben erweitert werden, indem das Histogramm der zweiten Stichprobe anstatt der Dichtefunktion benutzt wird. In diesem Fall ist anzuraten, dass die Intervallgrößen der beiden Histogramme gleich gewählt werden.

3.6.2 Kolmogoroff-Smirnoff-Test

Der Kolmogoroff-Smirnoff-Test basiert auf dem Vergleich einer Stichprobe mit der Verteilungsfunktion einer Zufallsvariablen. Hierzu muss zunächst die Verteilungsfunktion der Stichprobe ermittelt werden. Diese Verteilungsfunktion ist eine Treppenfunktion, die bei jedem Wert der Stichprobe um einen konstanten Wert erhöht wird. Ähnlich wie bei dem χ^2 -Test wird nun die Differenz der beiden Verteilungsfunktionen ermittelt. Für diese Differenz kann nun ebenfalls ein kritischer Punkt ermittelt werden. Eine detaillierte Beschreibung des Kolmogoroff-Smirnoff-Tests kann ebenfalls in [27] Kapitel 6.6.2 nachgelesen werden.

Auch der Kolmogoroff-Smirnoff-Test kann auf den Vergleich von zwei Stichproben erweitert werden. Die Erweiterung erfolgt analog zu dem Vorgehen des χ^2 -Tests.

3.6.3 Mann-Whitney U-Test

Bei dem Mann-Whitney U-Test werden die Stichprobenwerte der zwei gegebenen Stichproben gemischt und sortiert. Die sortierte Folge wird durchlaufen. Für jeden Wert der ersten Stichprobe wird bestimmt, wieviele Werte der zweiten Stichprobe einen niedrigeren Rang in der sortierten Folge besitzen. Die ermittelten Werte werden summiert. Anhand von Tabellen kann festgelegt werden, mit welcher Wahrscheinlichkeit die Stichproben die gleiche Zufallsverteilung besitzen. In [8] Kapitel 8.3 ist eine genauere Beschreibung zu finden.

3.6.4 Wilcoxon Signed-Rank Test

Bei dem Wilcoxon Signed-Rank Test werden zwischen den Werten der zwei Stichproben Paare gebildet. Die Differenz der Paare wird ermittelt und ihr wird ein

Rang zugeordnet. Im nächsten Schritt erhalten alle Ränge ein negatives Vorzeichen, die zu Differenzen gehören, in denen der Wert der zweiten Stichprobe größer ist. Die positiven und die negativen Ränge werden getrennt voneinander aufsummiert. Der Betrag von beiden Ergebnissen sollte annähernd gleich groß sein, wenn beide Stichproben die gleiche Zufallsverteilung besitzen. In [8] Kapitel 8.4 ist eine genauere Beschreibung zu finden.

3.6.5 Differenz-Test

Für den Differenz-Test müssen beide Stichproben die gleiche Anzahl an Werten enthalten. Beide Stichproben werden getrennt voneinander sortiert. Die Werte der beiden Stichproben, die nun denselben Rang besitzen, bilden ein Paar (a_i, b_i) . Die genormte Differenz e_i des Wertepaares wird ermittelt:

$$e_i = \frac{a_i - b_i}{a_i} \quad (3.45)$$

Es ist darauf zu achten, dass keine Division durch Null stattfindet. Sollte $a_i = b_i = 0$ gelten, so sei $e_i = 0$. Der Mittelwert aller genormten Differenzen wird ermittelt. Dieser Mittelwert entspricht einer prozentualen Abweichung. Auch für diesen Test kann nun ein Grenzwert festgelegt werden. Eine Angabe, mit welcher Wahrscheinlichkeit die zwei Stichproben die gleiche Zufallsverteilung besitzen, ist nicht möglich.

3.7 Unabhängigkeitstest

Viele statistische Verfahren setzen die Unabhängigkeit von Stichproben voraus. Aus diesem Grund gibt es Verfahren, die eine Wahrscheinlichkeit bestimmen, mit der die vorliegende Stichprobe unabhängig ist.

3.7.1 Run-Test (steigend/fallend)

In der Stichprobe liege eine feste Reihenfolge vor. Der Run-Test bildet nun die sogenannten "Runs". Bei dieser Version des Run-Test besteht ein Run aus aufeinanderfolgenden Stichprobenwerten, die entweder monoton fallen, oder monoton steigen. Wenn die Stichprobenwerte unabhängig sind, entstehen viele kleine Runs. Bei starker Abhängigkeit entstehen wenige große Runs. Die Anzahl der Runs ist also ein Maß für die Unabhängigkeit. In Tabellen kann nun die Wahrscheinlichkeit abgelesen werden, mit der die Werte unabhängig sind. (Siehe Dokumentation zu *octave*)

3.7.2 Run-Test (größer/kleiner)

Bei dieser Version des Run-Tests wird der Mittelwert der Stichprobe ermittelt. Ein "Run" besteht aus aufeinanderfolgenden Stichprobenwerten, die entweder alle größer oder alle kleiner sind als der Mittelwert. Ansonsten gibt es keine weiteren Unterschiede zwischen den beiden Versionen des Run-Tests. (Siehe [8], Kapitel 6.2.15)

Kapitel 4

Allgemeine Auswertungsstrategie

Das Ziel der Simulation ist, mehr über das betrachtete System zu erfahren. Zu Beginn der Analyse steht die Frage im Vordergrund, ob das System überhaupt stabil ist, oder ob das System überlastet ist. Ist das System überlastet, können weniger Anfragen bearbeitet werden, als dem System gestellt werden. In diesem Fall bilden sich lange *Warteschlangen* an überlasteten Komponenten des Systems. Die Auslastung der einzelnen Komponenten eines Systems ist meistens sehr unterschiedlich. So gibt es bei stabilen und bei instabilen Systemen immer eine oder mehrere Komponenten, die am höchsten ausgelastet sind. Diese Komponenten bilden den sogenannten *Flaschenhals*. Ein weiteres Ziel der Simulation ist, die Auslastung der einzelnen Stationen zu ermitteln und damit den Flaschenhals zu bestimmen.

Anfragen, die an das System gestellt werden, bezeichnet man als "*jobs*". Die Anzahl dieser Anfragen pro Zeiteinheit ist die *Ankunftsrate*. Die *Abgangsrate* ist die Anzahl bearbeiteter Anfragen pro Zeiteinheit. Ein wichtiges Kriterium für ein stabiles System ist, dass die Ankunftsrate nicht größer als die Abgangsrate ist. In diesem Fall wird von dem *Durchsatz* des Systems gesprochen. Warteschlangen bilden sich auch in einem stabilen System. Warteschlangen aus "*jobs*", die auf ihre Bearbeitung warten, entstehen häufiger bei hochausgelasteten Systemen. Die *Population* eines Systems ist die Anzahl aller "*jobs*" in einem System, die bearbeitet werden oder auf ihre Bearbeitung warten. Mathematisch fundiertere Definitionen dieser und anderer *Leistungsmaße* werden in dem Kapitel 7.2.2 vorgestellt. Eine genauere Beschreibung ist in [10] nachzulesen.

Die Simulation liefert im Allgemeinen Messreihen dieser Leistungsmaße. Diese Messreihen müssen nun mit geeigneten statistischen Methoden bearbeitet werden, um Aussagen treffen zu können. In vielen Fällen interessiert der Erwartungswert der Leistungsmaße. Allerdings interessieren auch andere Größen wie zum Beispiel die Varianz und die Standardabweichung der Messwerte von dem Erwartungswert. Ein allgemeiner Überblick über die statistischen Grundlagen ist

in Kapitel 3 festgehalten. Auf die spezielle Situation bei der Auswertung von Simulationsdaten wird in den folgenden Abschnitten eingegangen.

4.1 Simulationsdaten

Die Simulation liefert als Basisgrößen nur Ereignisse, die während der Simulationsdurchführung entstehen. Dieser *Ereignisstrom* kann direkt genutzt werden, um das System zu bewerten. Im Allgemeinen wird allerdings versucht, aus diesen Ereignissen diverse Leistungsmaße zu bestimmen, um so eine geeignete Messreihe zu erhalten. Dies hat den theoretischen Hintergrund, dass die Analyse der Messreihen auf die Analyse von *stochastischen Prozessen* zurückgeführt werden kann (siehe Kapitel 3.4.1).

Die Berechnung der Leistungsmaße aus den Ereignissen ist allerdings schon eine Abstraktion, die gewisse Probleme beinhaltet. Zur Berechnung eines Wertes der Messreihe eines Leistungsmaßes müssen einige Ereignisse ausgewählt werden. Hier gibt es sehr viele Möglichkeiten, die alle Vor- und Nachteile besitzen. Die zwei üblichsten Methoden werden in den beiden folgenden Abschnitten beschrieben.

4.1.1 Disjunkte Intervalle

Die Methode der “disjunkten Intervalle” ist eine Möglichkeit, um aus Ereignissen Leistungsmaße zu bestimmen. Die Modellzeit wird in Intervalle zerteilt, die alle gleich groß sind und sich nicht überlappen. Nun werden alle Ereignisse dieser Intervalle herangezogen, um das stellvertretende Leistungsmaß zu bestimmen.

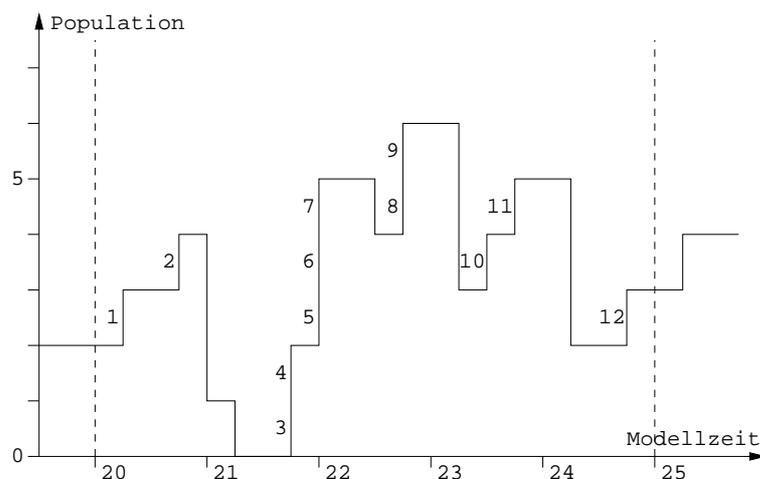


Abbildung 4.1: Bestimmung der Ankünfte in “disjunkten Intervallen”.

Beispiel 1: Es sei ein Intervall gegeben, das bei Modellzeit 20 startet und bei Modellzeit 25 endet. Von Interesse sei die Ankunftsrate in diesem Intervall. Daher sind auch nur die Ereignisse bedeutend, die eine Ankunft repräsentieren. Die Anzahl dieser Ereignisse sei 12. Die Ankunftsrate ist definiert als Ankünfte pro Zeit. In diesem Fall ergibt sich eine Ankunftsrate von $\frac{12}{25-20} = 2.4$.

Diese Methode hat den Vorteil, dass sie einfach zu implementieren ist. Ein schwieriger Nachteil ist jedoch, dass die Größe der Intervalle sinnvoll gewählt werden muss. Sind die Intervalle zu klein gewählt, kommt es zu hohen Schwankungen in der Messreihe. Sind die Intervalle zu groß gewählt, muss die Simulation unnötig lange durchgeführt werden, damit die Messreihe genügend viele Daten enthält.

Ein weiterer Nachteil ist, dass nicht jeder konkreten Modellzeit ein Wert eines Leistungsmaßes zugeordnet werden kann, da die Werte nur für die Intervalle bestimmt werden.

4.1.2 Wandernde Intervalle

Die Methode der “wandernden Intervalle” besteht darin, dass ein Sichtfenster (Intervall) konstanter Größe in der Modellzeit verschoben wird. Die Verschiebungen in der Modellzeit sind stets aufwärtsgerichtet, müssen aber nicht konstant sein.

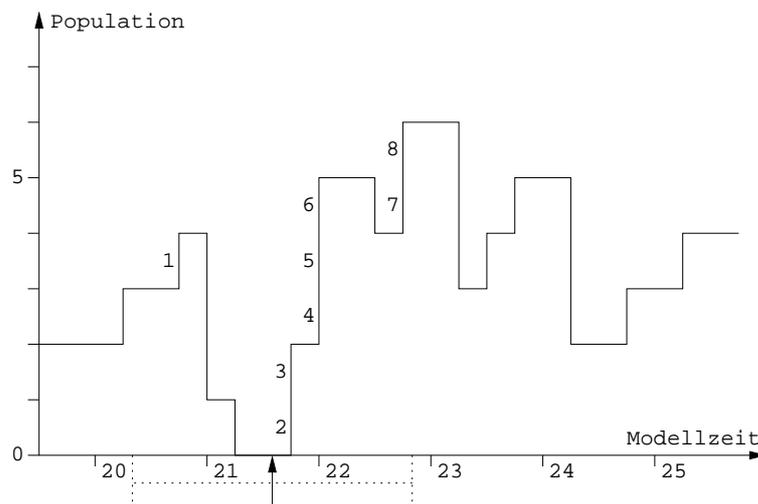


Abbildung 4.2: Bestimmung der Ankünfte in “wandernden Intervallen”.

Der Vorteil dieser Methode ist, dass man einen Repräsentanten für ein Leistungsmaß für jeden Modellzeitpunkt berechnen kann. Schwierig ist wiederum die Wahl der Größe des Sichtfensters.

Die Methode der “disjunkten Intervalle” ist ein Spezialfall dieser Methode, wenn die Verschiebung in der Modellzeit konstant ist und der Sichtfenstergröße entspricht. Ist die Verschiebung in der Modellzeit viel kürzer als die Größe des Sichtfensters, kann es signifikante Abhängigkeiten in der Messreihe des resultierenden Leistungsmaßes geben. Ist die Verschiebung in der Modellzeit jedoch viel weiter als die Größe des Sichtfensters, werden einige Ereignisse nicht berücksichtigt und es entstehen unnötig lange Simulationen. Vor diesem Hintergrund ist die Methode der “disjunkten Intervalle” ein gesundes Mittelmaß.

4.2 Ensemble-Analyse

Die Ensemble-Analyse wird häufig bei der Simulation mit begrenztem Horizont (siehe Kapitel 2.3.1) angewendet. Ziel der Analyse ist, eine Aussage zu treffen, wie das Systemverhalten bei einem festen Initialisierungszustand ist. Hierzu werden mehrere unabhängige Simulationsläufe benötigt.

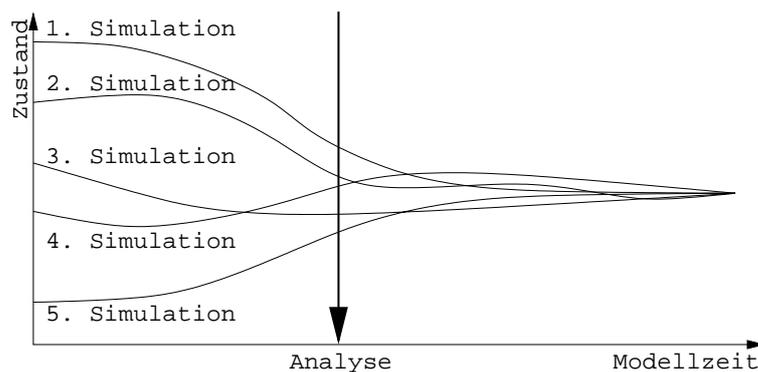


Abbildung 4.3: Die Ensemble-Analyse zu einem bestimmten Zeitpunkt.

In der Praxis wird versucht, die Unabhängigkeit über eine geeignete Wahl der Startseeds für jede Simulation zu erreichen. Es wird davon ausgegangen, dass die Simulationen unabhängig sind, wenn sie nicht dieselbe Folge von Zufallszahlen benutzen. Wenn die Simulationen nacheinander durchgeführt werden, kann jede neue Simulation mit dem letzten “Zufallswert” der vorherigen Simulation gestartet werden. Wenn die Simulationen jedoch parallel durchgeführt werden, sollte die Periode des Zufallszahlengenerators in entsprechend viele Teilbereiche zerlegt und jede Simulation mit dem Anfangswert eines Teilbereiches initialisiert werden. Beide Verfahren können nicht garantieren, dass eine Zufallszahl nicht für mehrere Simulationen genutzt wird. Sie garantieren jedoch, dass das maximale Maß an Unabhängigkeit erreicht wird. (siehe Kapitel 2.3.4)

Da alle Simulationen mit dem gleichen Initialisierungszustand gestartet werden, sind die Daten vergleichbar, die zu einem bestimmten Modellzeitpunkt in

allen Simulationen gesammelt wurden. Mit der Annahme der Unabhängigkeit lassen sich nun die Standard-Methoden anwenden, um den Erwartungswert, die Varianz und das Konfidenzintervall zu schätzen (siehe Kapitel 3.3). Weitere Hinweise sind in [8] Kapitel 6.1 vermerkt.

4.3 Zeitreihen-Analyse

Die Zeitreihen-Analyse wird bei der Simulation mit unbegrenztem Horizont angewandt. Ziel bei dieser Analyse ist, Aussagen über die stationäre Zustandsverteilung zu treffen. Voraussetzung ist, dass das System eine stationäre Zustandsverteilung besitzt. Die mathematische Definition zur Stationarität wird in Kapitel 3.4 gegeben. In [8] Kapitel 6.2 wird ein detaillierterer Überblick der Zeitreihen-Analyse gegeben. Auf die von der Simulation unabhängige allgemeine Zeitreihen-Analyse geht die Literatur [31] ein.

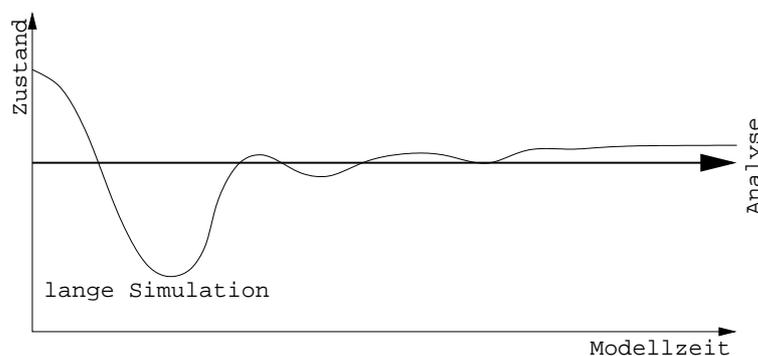


Abbildung 4.4: Die Zeitreihenanalyse an einem Simulationslauf.

Die Wahl des Startzustandes beeinflusst das “typische Systemverhalten” nicht. Allerdings hat der Startzustand immer Einfluss auf die ersten Daten der Messreihe einer Simulation. Dieser Einfluss wirkt störend auf die Analyse, da Aussagen über die “typische Zustandsverteilung” gemacht werden sollen. Der Startzustand ist aber unter Umständen “untypisch”. Die Standard-Lösung für dieses Problem ist, so viele Daten aus einem Simulationslauf zu sammeln, bis der Fehler durch den Startzustand vernachlässigt werden kann. In Kapitel 6 wird ein alternativer Lösungsansatz detailliert diskutiert, zunächst folgt jedoch die Beschreibung der Standard-Lösung.

Die Annahme, dass das Modell eine stationäre Zustandsverteilung besitzt, macht die Daten unterschiedlicher Modellzeitpunkte, die nicht mehr durch den Startzustand beeinflusst werden, miteinander vergleichbar. Der Zeitcharakter der Messreihe ist vernachlässigbar. Eine Schwierigkeit ist jedoch, dass die Werte einer Zeitreihe im Allgemeinen voneinander abhängig sind. Die meisten Moment-

schätzer (siehe Kapitel 3.3) sind allerdings nur erwartungstreu für unabhängige Stichproben. Daher führt man eine Abstraktion von der eigentlichen Messreihe durch, um die Unabhängigkeit zu gewährleisten. Dies wird näher in Kapitel 6 beschrieben.

Werden nun die neu ermittelten unabhängigen Daten behandelt, können die gängigen Momentschätzer angewendet werden. Unter anderem können auch die Konfidenzintervalle geschätzt werden. In der Regel werden Simulationen so lange durchgeführt, bis die Grenzen eines entsprechenden Konfidenzintervalls weniger als 5% von dem Mittelwert abweichen.

4.4 Auswertungsbeispiel

In diesem Abschnitt soll eine Zeitreihenanalyse durchgeführt werden, die mit den “üblichen” Methoden arbeitet. Als Beispiel dient hier ein M/M/1-Server. Der Name M/M/1 ist der Kendall-Notation entnommen (siehe [25], Kapitel 30.1). Die beiden “M” stehen für eine negativ-exponentielle Ankunfts- und Bedienrate. Die “1” bedeutet, dass gleichzeitig nur 1 Auftrag bearbeitet werden kann. Das Verhalten des M/M/1-Servers ist durch theoretische Analysen bestens bekannt. Daher wird der M/M/1-Server in der Literatur häufig benutzt, um die Qualität von statistischen Methoden zu demonstrieren. Das soll auch hier der Fall sein.

Die Abbildung 4.5 zeigt die schematische Darstellung des M/M/1-Servers in einer Prozessdarstellung. Die hier verwendete Prozessdarstellung ist der “B-Modellformalismus” (siehe [11]). Er wurde im Rahmen des Sonderforschungsbereichs 559 “Modellierung großer Netze in der Logistik” vom Lehrstuhl IV des Fachbereichs Informatik an der Universität Dortmund entwickelt. Zu Beginn wird das System mit 100 wartenden Kunden initialisiert. Danach wird im Durchschnitt 1 Auftrag pro Zeiteinheit an das System gestellt und der Bedienungswunsch beträgt im Durchschnitt 0.5 Modellzeiteinheiten.

4.4.1 Theoretische Betrachtung

Sei λ die Ankunftsrate in Aufträgen pro Zeit und μ sei die Servicerate in Aufträgen pro Zeit. Die Verkehrsintensität ρ wird durch $\rho = \frac{\lambda}{\mu}$ definiert. Wenn mehr Aufträge gestellt werden als bearbeitet werden können, gilt $\lambda \geq \mu$ und die Warteschlange des Systems wird immer mehr wachsen. Die Bedingung für ein stabiles System ist daher $\rho < 1$. Die Wahrscheinlichkeit, dass sich n -Kunden im System befinden, ist $p_n = (1 - \rho)\rho^n$. Der Erwartungswert $E[n]$ der Anzahl der Kunden im System ist definiert durch

$$E[n] = \frac{\rho}{1 - \rho} \quad . \quad (4.1)$$

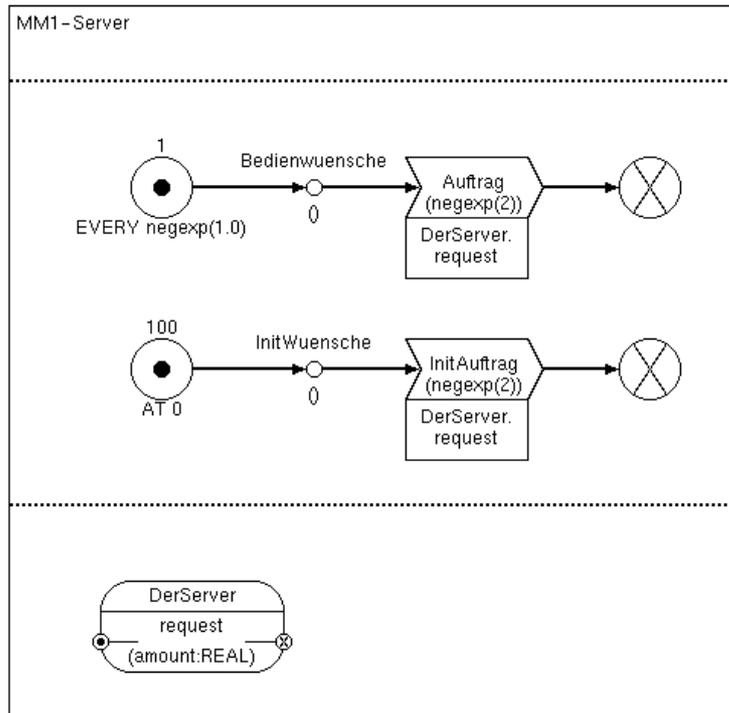


Abbildung 4.5: Das Prozessmodell eines M/M/1-Servers, der mit 100 Aufträgen initialisiert wird.

Die Varianz dieses Erwartungswertes lässt sich bestimmen durch

$$Var[n] = \frac{\rho}{(1 - \rho)^2} \quad . \quad (4.2)$$

Dies sind wohl die beiden Größen, die am häufigsten gefragt sind. In [25] Kapitel 31.2 wird die Herleitung dieser und weiterer Größen beschrieben. Die Gleichungen 4.1 und 4.2 werden im Folgenden die Grundlage bilden, um die Ergebnisse zu bewerten, welche die simulative Auswertung liefert.

In dem konkreten Beispiel ist $\lambda = 1$ und $\mu = 2$. Damit ist $\rho = 0.5$ und die Bedingung für ein stabiles Verhalten ist erfüllt. Da $E[n] = \frac{0.5}{1-0.5} = 1$ ist, ist zu erwarten, dass sich im Durchschnitt 1 Kunde im System befindet. Die Varianz des Erwartungswertes ist $Var[n] = \frac{0.5}{(1-0.5)^2} = 2$. Der Startzustand muss nicht berücksichtigt werden, da er auf das “typische” Verhalten keinen Einfluss hat.

4.4.2 Standard-Methoden

Das hierarchische-evaluierungs “Tool” *HIT* (“Hierarchical Evaluation Tool”) wurde hier zur Simulation und Analyse benutzt. Es wurde am Lehrstuhl für “Quantitative Methoden” des Fachbereichs Informatik an der Universität Dortmund entwickelt. *HISLANG* ist die zugehörige Simulationssprache, mit der parallel agierende Prozesse beschrieben werden können. Eine detaillierte Beschreibung der

verwendeten Simulationssprache HISLANG und dem Software-Tool HIT ist in [9] nachzulesen.

M/M/1-Server in HISLANG-Syntax

```
%COMMON
%%PARAM = WARN
%%END
%COPY "counter"
%COPY "semaphor"
%COPY "observer"
TYPE voidRecordName RECORD; END TYPE voidRecordName;
TYPE initServer_Typ MODEL;
  COMPONENT DerServer : server
    (LET ACCEPT := ALWAYS,
     LET SCHEDULE := FCFS (LET CAPACITY := 1),
     LET DISPATCH := EQUAL (LET SPEED := 1.0),
     LET OFFER := ALL);
  TYPE Aufruf SERVICE;
  USE
    SERVICE Anfrage(B2H___amount : REAL DEFAULT 1.0);
  END USE;
  BEGIN
    Anfrage(negexp(2));
  END {SERVICE} TYPE Aufruf;
  TYPE Quellen SERVICE;
  BEGIN
    LOOP
      hold(negexp(1));
      CREATE 1 PROCESS Aufruf;
    END LOOP;
  END {SERVICE} TYPE Quellen;
  TYPE InitQuellen SERVICE;
  BEGIN
    CREATE 100 PROCESS Aufruf;
  END {SERVICE} TYPE InitQuellen;
  REFER
    Aufruf
  TO
    DerServer
  EQUATING
    Aufruf.Anfrage WITH DerServer.request;
  END REFER;
BEGIN
  CREATE 1 PROCESS InitQuellen;
  CREATE 1 PROCESS Quellen;
END {MODEL} TYPE initServer_Typ;
{-----}
EXPERIMENT versuch METHOD SIMULATIVE;
BEGIN
  EVALUATE MODEL initServer : initServer_Typ(LET SEED := 13);
  EVALUATIONOBJECT
    messgeraet VIA initServer.DerServer;
  BEGIN
    MEASURE POPULATION AT messgeraet;
    MEASURE UTILIZATION AT messgeraet;
    MEASURE THROUGHPUT AT messgeraet;
    MEASURE TURNAROUNDTIME AT messgeraet;
    CONTROL AT messgeraet STOP
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE POPULATION AND
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE UTILIZATION AND
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE THROUGHPUT AND
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE TURNAROUNDTIME;
  END EVALUATE;
END EXPERIMENT versuch;
```

Die Zeitreihenanalyse dieses Modells wurde mehrfach durchgeführt, um die Qualität der Ergebnisse zu überprüfen. Die Simulationen erfolgten mit jeweils unterschiedlichem Startseed des Zufallszahlengenerators. Die Startseeds wurden so gewählt, dass sie eine Periode des Zufallszahlengenerators in gleich große Teilintervalle unterteilt. So kann gewährleistet werden, dass alle Simulationen

mit der maximalen Anzahl an unterschiedlichen Zufallsvariablen arbeiten. Dieses Vorgehen wird in Kapitel 2.3.4 näher erläutert. Das Software-Tool HIT benutzt den in Kapitel 2.3.3 beschriebenen Zufallszahlengenerator. Die Periode muss durch die Anzahl der Simulationen geteilt werden, dadurch wird die Größe eines Teilintervalls bestimmt. Mit der geschlossenen Form des Zufallszahlengenerators können die Startseeds ermittelt werden.

Bei den Simulationen wurde die *Population* (siehe [10]) des Systems bestimmt. In diesem Fall ist die Population die Anzahl der Kunden im System. Die Ergebnisse der Simulation lassen sich also direkt mit den Ergebnissen der theoretischen Analyse vergleichen. Die Simulation wurde so lange durchgeführt, bis das 95%-Konfidenzintervall nur noch 5% vom Mittelwert abwich. Die Ergebnisse der Simulationen sind in der folgenden Tabelle aufgeführt.

| Startseed | Modellzeit | Population | Abweichung |
|-------------|------------|------------|------------|
| 13 | 203522 | 1.034563 | 3.46% |
| -536870899 | 294804 | 1.027489 | 2.75% |
| -1073741811 | 239675 | 1.012724 | 1.27% |
| -1610612723 | 263975 | 1.011144 | 1.11% |
| -2147483635 | 157909 | 1.032464 | 3.25% |
| 1610612749 | 221171 | 1.019919 | 1.99% |
| 1073741837 | 194202 | 1.036445 | 3.64% |
| 536870925 | 214661 | 1.040207 | 4.02% |
| Mittelwert | 223740 | 1.026869 | 2.67% |

Abbildung 4.6: Ergebnisse der HIT-Simulationen

Die Analyse des Software-Tools HIT arbeitet, von kleinen Ausnahmen abgesehen, wie es in Kapitel 4.3 beschrieben ist. Der Tabelle aus Abbildung 4.6 ist zu entnehmen, dass alle Populationswerte nah an dem theoretischen Wert liegen. Kein Wert weicht mehr als 5% von dem theoretischen Wert ab.

Erstaunlich ist allerdings, dass alle Populationswerte größer als der theoretische Wert sind. Bei acht unabhängigen Ergebnissen, die alle größer als der theoretische Wert sind, ist es sehr unwahrscheinlich, dass der theoretische Wert der Erwartungswert der Simulationsergebnisse ist. Mit hoher Wahrscheinlichkeit entspricht der Erwartungswert der Simulationsergebnisse nicht dem theoretischen Wert. Woran mag dies liegen? Der recht "untypische" Initialisierungszustand wird das Simulationsergebnis verfälschen. Alle Simulationsergebnisse tendieren in "Richtung" des Initialisierungszustandes. Es wurde nicht nur das "typische" Modellverhalten beobachtet, sondern auch die Einschwingphase, in welcher der anfängliche Stau abgebaut wird. Da zur Berechnung des Mittelwertes keine Daten verworfen wurden, kann der Mittelwert kein erwartungstreuer Schätzer für

den theoretischen Wert sein. Der Schätzer ist in diesem Fall nur asymptotisch erwartungstreu, da der Einfluss des Startzustandes im Unendlichen vollständig vernachlässigt werden kann. Da aber keiner Simulation unendlich viel Modellzeit zur Verfügung steht, hat dies keinen praktischen Nutzen.

Die benötigte Modellzeit sollte ebenfalls zum Nachdenken anregen. Ein Durchschnitt von 223740 Modellzeiteinheiten ist für dieses einfache Modell sehr viel. Ein Modell des M/M/1-Server, der nur mit einem Auftrag initialisiert wird, benötigt im Durchschnitt nur 16808 Modellzeiteinheiten. Dies ergab eine zweite Experimentserie mit denselben Startseeds und dem veränderten Startzustand. Das mit 100 Aufträgen initialisierte Modell benötigt also mehr als 13 mal so viel Modellzeit, nur um den Einfluss des Initialisierungszustandes zu verringern.

Es werden also intelligentere Methoden benötigt, die den Einfluss des Initialisierungszustandes berücksichtigen. Diese Diplomarbeit versucht, an diesem Problem anzusetzen. Ausgehend von den Standard-Methoden soll ein Weg gefunden werden, die Genauigkeit der Simulationsergebnisse zu verbessern und die benötigte Modellzeit zu reduzieren. Wünschenswert ist ein Verfahren, das den theoretischen Wert erwartungstreu schätzt.

Kapitel 5

Chaos - eine andere Sichtweise

Auch die Physik beschäftigt sich mit realen Systemen. Dies sind Systeme aus den verschiedensten Bereichen, wie zum Beispiel der Mechanik, der Thermodynamik oder der Kern- und Elementarteilchenphysik. Meist steht hier jedoch nicht die Analyse eines konkreten Systems im Vordergrund, sondern die Herleitung physikalischer (mathematischer) Modelle und deren Eigenschaften. Das Bohrsche Atommodell ist zum Beispiel ein Regelwerk für die atomaren Abläufe. Es hat sich gezeigt, dass mit dem Bohrschen Atommodell einige Eigenschaften vorausgesagt werden können, allerdings beschreibt es nicht die volle Komplexität der realen Systeme (Atome).

In letzter Zeit sind die sogenannten *chaotischen Systeme* in den Mittelpunkt der Aufmerksamkeit gerückt. In einem chaotischen System haben minimale Parameteränderungen fundamentale Auswirkungen auf das Systemverhalten. Ein Beispiel für ein chaotisches System ist der Bender-Oszillator. In [37] Kapitel 12 wird der Bender-Oszillator in dem Essay "Chaos - eine ordentliche Unordnung" von S. Walker beschrieben. Einige Parameterwerte (genauer: Zwangsfrequenzen) dieses Systems führen zu einem Systemverhalten, das sich nach einer Anfangsphase einpendelt und dann konstant verhält. Andere Parameterwerte führen hingegen zu einem sich ständig ändernden Systemzustand, der daher nicht vorhersehbar ist. Dies ist das typische Verhalten eines chaotischen Systems.

5.1 Ein Modell des Bender-Oszillators

Walker weist darauf hin, dass die Iterationsgleichung $x_{k+1} = f(x_k) = wx_k(1 - x_k)$ ein gutes mathematisches Modell des Bender-Oszillators ist. Der Parameter w übernimmt die Aufgabe der Zwangsfrequenz. Es wird ein beliebiger Startwert x_0 gewählt und damit x_1 bis x_n ermittelt. Es können beliebig viele Iterationsschritte durchgeführt werden. Das Modell liefert zwar nicht dieselben Messwerte wie der Bender-Oszillator, aber das Verhalten ist qualitativ gleich. Für manche Para-

meterwerte w ist das Modell konvergent, für andere nicht. Der Wertebereich für den Startwert x_0 ist $]0; 1[$ und der Wertebereich für den Parameter w ist $[0; 4]$. Alle weiteren Ergebniswerte x_k mit $k > 0$ liegen im Bereich $[0; 1]$.

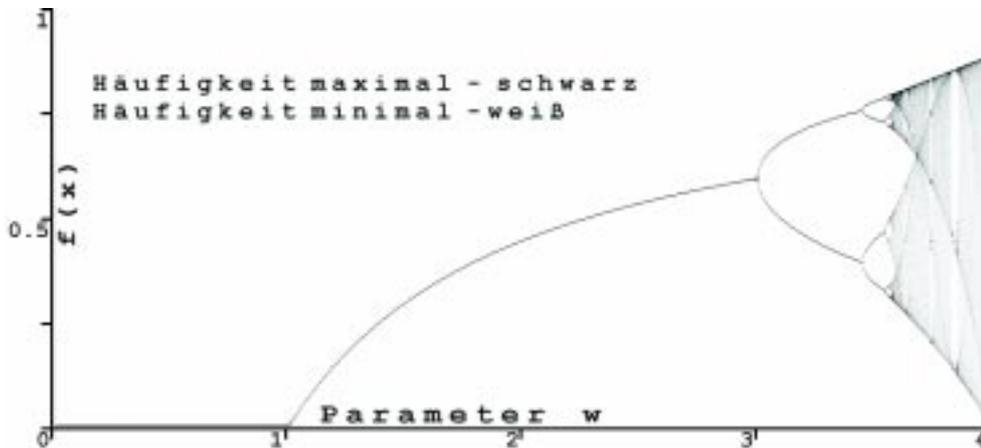


Abbildung 5.1: Darstellung der Iterationsgleichung für $0 \leq w \leq 4$ mit $x_0 = 0.5$

Die Abbildung 5.1 ist eine Darstellung mehrerer Iterationen. Auf der X-Achse sind die w -Werte im Bereich $0 \leq w \leq 4$ aufgetragen, auf der y-Achse ist der Bildbereich $[0; 1]$ der Iterationsgleichung aufgetragen. Der Darstellung liegt eine BMP-Datei der Größe 1000×400 Pixel zugrunde. Die Schrittweite im Wertebereich $[0; 4]$ des Parameters w ist also 0.004. Der Wertebereich $[0; 1]$ der Iterationswerte x_k ist in 400 gleiche Intervalle unterteilt. Während der Iteration mit einem festen Parameter w werden die Iterationswerte x_k pro Intervall gezählt. Nach der Iteration werden die Zählwerte der Intervalle linear auf den Maximalwert 255 und den Minimalwert 0 normiert. Diese normierten Zählwerte werden nun an die entsprechende Stelle in die BMP-Datei geschrieben und ergeben die Grauwerte der Darstellung. Die Grauintensität der einzelnen Punkte ist daher stellvertretend für die Häufigkeit des entsprechenden Ergebniswertes, während einer Iteration mit festem w . Die Darstellung ist ein 3-dimensionales Histogramm und die Grauintensität ersetzt in diesem Fall die Balkenhöhe normaler Histogramme. Wichtig ist daher nicht nur, wo sich ein Punkt in der Darstellung befindet, sondern auch, welche Grauintensität er hat.

Es ist zu erkennen, dass die Iterationsgleichung für Parameterwerte $w < 3$ (ungefähr) gegen einen Grenzwert konvergiert. Ab dieser Grenze beginnen nun Bifurkationen. Die Abstände der Bifurkationen werden immer kleiner, so dass für (ungefähr) $w > 3.5$ keine Struktur mehr erkennbar ist.

Dieser erste Eindruck ist nicht richtig. Die Abbildung 5.2 beweist das Gegenteil. Dargestellt wird eine höhere Auflösung in dem Bereich von $3.4 \leq w \leq 3.9$. Es ist erkennbar, dass in diesem Bereich viele Abschnitte sind, in denen konvergen-

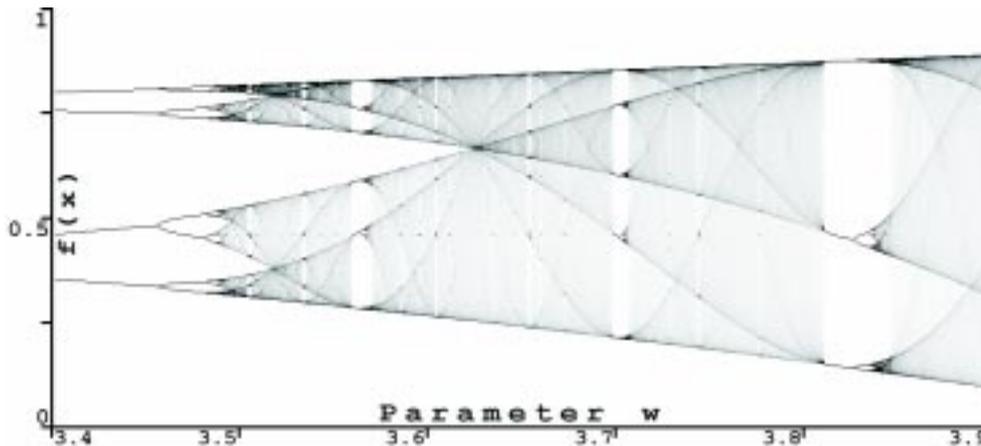


Abbildung 5.2: Darstellung der Iterationsgleichung für $3.4 \leq w \leq 3.9$ mit $x_0 = 0.5$.

tes Verhalten beobachtet werden kann. Diese Abschnitte haben unterschiedliche Abstände und Ausdehnungen. Außerdem ist insgesamt eine Struktur zu erkennen, die sich wie ein Netz aus Schwingungen über den gesamten Bereich legt. Noch höhere Auflösungen würden zeigen, dass sich diese Charakteristika auch in noch kleineren Bereichen wiederfinden lassen. Sie lassen sich speziell dort beobachten, wo die Bifurkationen mit immer kleiner werdendem Abstand auf einen konvergenten Bereich folgen, zum Beispiel im Bereich $3.80 \leq w \leq 3.85$.

5.2 Parallelen zu stochastischen Prozessen

Die Iterationsgleichung kann nicht nur als mathematisches Modell des Bänder-Oszillators aufgefasst werden. Eine weitere Interpretation ist:

- Der Startwert x_0 sei der Startzustand eines Modells. Alle weiteren Ergebniswerte seien Folgezustände.
- Die einzelnen Iterationsschritte entsprechen einem diskreten Schritt in der Modellzeit.
- Der Parameter w sei ein Modellparameter.

In diesem Sinne kann ein Iterationsdurchlauf mit festem Parameter w als eine Zeitreihe aufgefasst werden. Interessant ist, welche Aussagen über diese Zeitreihen unter Berücksichtigung des Hintergrundwissens über chaotische Systeme getroffen werden kann.

Die Abbildung 5.3 zeigt die Werte eines Iterationsdurchlaufes mit dem Startwert $x_0 = 0.001$ und dem Parameter $w = 2$. Wenn diese Werte als Zeitreihe in-

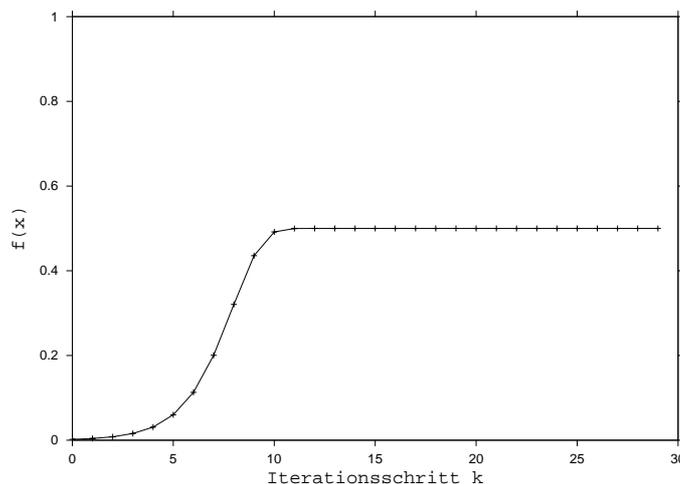


Abbildung 5.3: Eine Iteration mit $w = 2$ und $x_0 = 0.001$.

terpretiert werden, ist eine transiente Phase bis zum Iterationsschritt 11 zu erkennen. Danach nimmt das Modell nicht nur eine stationäre Zustandsverteilung an, sondern einen tatsächlichen stationären Zustand. Der Startzustand kann den stationären Zustand nicht beeinflussen. Er bestimmt aber die Länge der transienten Phase. Wenn der Startwert näher bei 0.5 liegen würde, wäre die transiente Phase wesentlich kürzer. Dies lässt sich leicht mit einem Taschenrechner überprüfen.

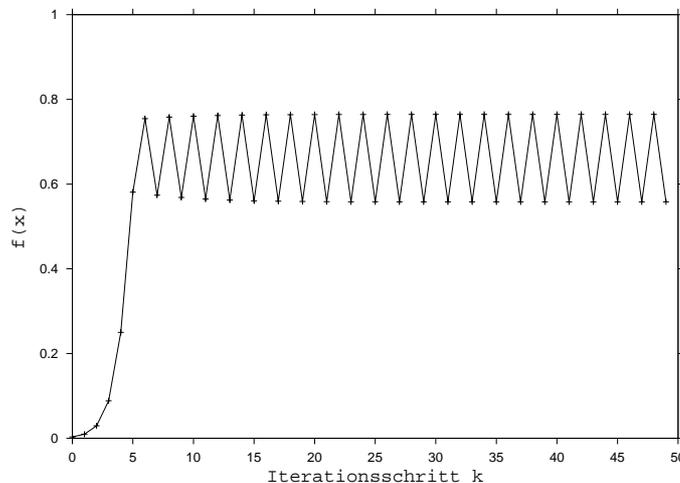


Abbildung 5.4: Eine Iteration mit $w = 3.1$ und $x_0 = 0.001$.

Ein einfacher periodischer Prozess, der zwischen zwei Werten wechselt, wird durch die Iteration mit dem Startwert $x_0 = 0.001$ und dem Parameter $w = 3.1$ erzeugt (siehe Abbildung 5.4). Auch hier gilt, dass der Startwert nur die Länge der transienten Phase beeinflusst.

Die Abbildung 5.5 zeigt die Iteration mit dem Startwert $x_0 = 0.5$ und dem Parameter $w = 3.9$. Es ist klar, dass der Prozess keinen ausgewählten Zustand besitzt, gegen den die Zeitreihe konvergiert. Leider sind die einzelnen Iterations-

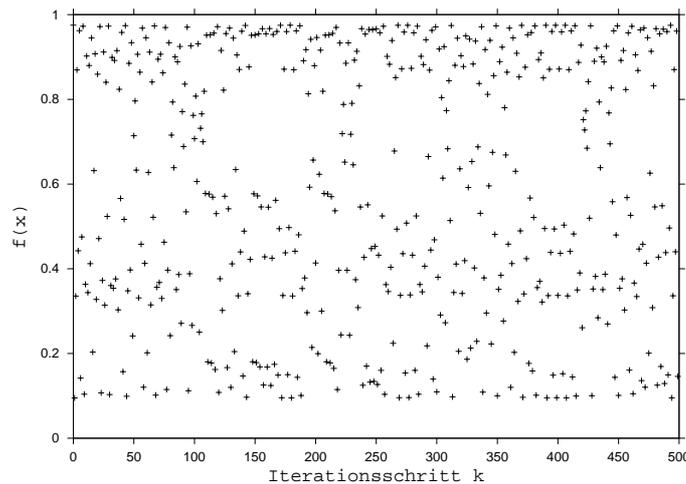


Abbildung 5.5: Eine Iteration mit $w = 3.9$ und $x_0 = 0.5$.

werte x_k rein deterministisch. Daher ist es nicht möglich, eine Zustandsverteilung zum Zeitpunkt k anzugeben, und die Definition der Stationarität für stochastische Prozesse lässt sich nicht anwenden. Es existiert aber eine Zustandsverteilung, wenn die gesamte Iteration mit einem festen Parameter w betrachtet wird. Dies wird deutlich aus Abbildung 5.2. Es wurde bereits erwähnt, dass die Darstellung ein 3-dimensionales Histogramm ist. Bei einem konkreten Parameter w auf der horizontalen Achse, ergeben die Grauintensitäten der vertikalen Achse eine Zustandsverteilung. Es liegt die Vermutung nahe, dass sich diese Zustandsverteilung einer bestimmten Grenzverteilung nähert. Trotzdem ist der Prozess bzw. Iterationslauf nicht stationär, denn auch periodische Prozesse haben diese Grenzverteilung, sind aber nicht stationär. Zu erwähnen ist noch, dass die Verteilung der Messwerte nicht gleichmässig ist, gewisse Häufungspunkte lassen sich hier erkennen. Dies liegt an der schon erwähnten Netzstruktur. Man könnte die Iteration mit $w = 3.9$ als interessanten Sonderfall auffassen, vielleicht als eine Art “Random-Walk”.

Betrachtung der Systemlast

In diesem Abschnitt soll nur ein Ausblick gegeben werden, wie kompliziert das Modellverhalten werden kann, wenn der Parameter w der Iterationsgleichung bei einem Iterationsdurchlauf nicht konstant ist.

Wenn der Parameter w nicht als Systemparameter, sondern als Systemlast interpretiert wird, ergeben sich weitere Möglichkeiten. Das Ändern des Wertes w würde einer Änderung der Last an einem konstanten System entsprechen. In [29] werden interessante Experimente in diesem Sinne mit der Iterationsgleichung beschrieben. Es wird mit dem Wert w periodisch zwischen zwei Werten A und B gewechselt und die Steigung der Iterationsgleichung wird erst loga-

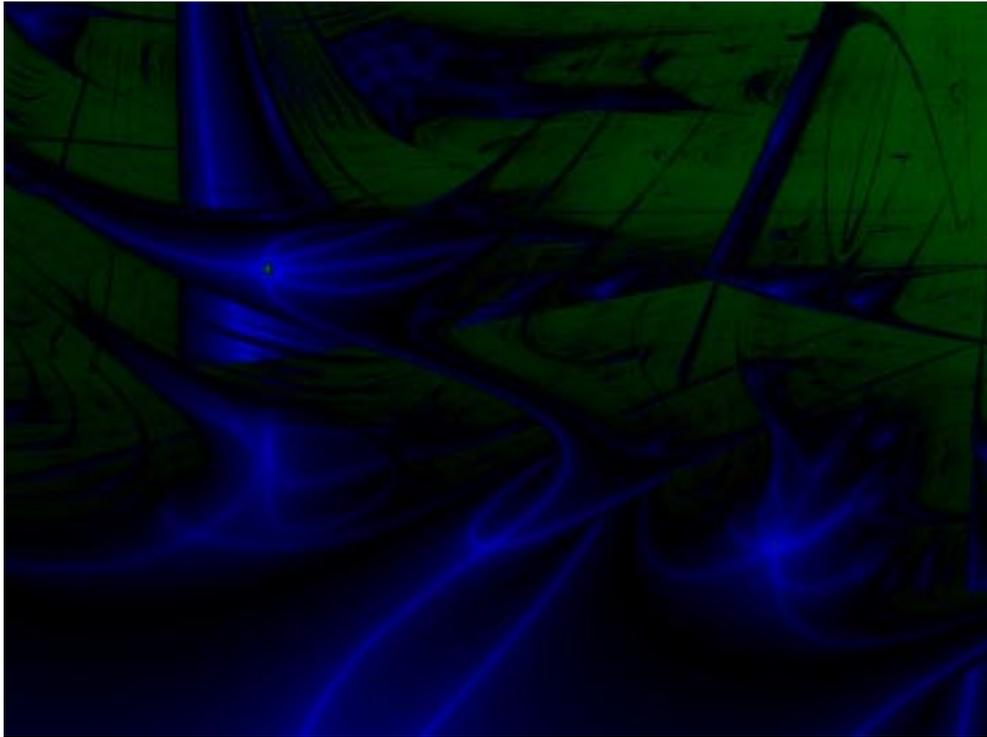


Abbildung 5.6: Ljapunow-Diagramm. Chaotische und geordnete Bereiche ergeben abstrakte Strukturen.

rithmiert und dann summiert. Das Ergebnis ist, dass sich chaotische Bereiche und gleichförmige Bereiche nicht konkret vorhersagen lassen. Werden diese Bereiche farblich gekennzeichnet und dann in einer 2-dimensionalen Grafik (über A und B) aufgetragen, ergeben sich abstrakte Strukturen. Diese Art der Darstellung heißt Ljapunow-Diagramm, siehe Abbildung 5.6.

5.3 Schlussfolgerungen

Die Ermittlung des Systemverhaltens ist eine zentrale Aufgabe der Simulation. Die Darstellungsform, die in Abbildung 5.1 und 5.2 genutzt wird, kann einen sehr guten intuitiven Überblick über das Verhalten eines Systems geben. An dieser Darstellung lassen sich die Zustandsverteilungen in Abhängigkeit von einem Parameter sehr gut ablesen. Allerdings müssen sehr viele Messwerte bestimmt werden, damit die Darstellung aussagekräftig wird.

Da die Simulation von großen Modellen lange dauert, könnten die Ergebnisse der Iterationsschritte als Zeitreihen verwendet werden, um Auswertungsmethoden zu testen. Hierbei stehen dann einfache stationäre und einfache periodische Zeitreihen zur Verfügung. Der stationäre Zustand bzw. die Periode ist leicht zu bestimmen, so dass die Auswertungsmethoden direkt überprüft werden können. Die ungeordneten Bereiche der Iterationsgleichung können einen Sonderfall für

die Auswertungsmethoden ergeben.

Offensichtlich kann die Wahl der Parameter eines Modells darüber entscheiden, ob das Modell stationär ist oder nicht. Überraschend ist allerdings, dass schon minimalste Parameteränderungen fundamentale Auswirkungen haben können. Dies wird durch die Abbildung 5.2 deutlich. Diese Modelle werden chaotisch genannt.

Ebenfalls ist Überraschend, dass schon die einfache Iterationsgleichung $x_{k+1} = f(x_k) = wx_k(1 - x_k)$ chaotisch ist. Modelle, die praktische Relevanz haben, sind viel komplizierter. Diese sind daher nicht automatisch chaotisch, aber die Wahl der Modellparameter muss sehr gründlich mit dem realen System abgestimmt werden. Ist eine exakte Bestimmung der Modellparameter in einem chaotischen Modell nicht möglich, ist eine Auswertung in Bezug auf Stationarität oder andere Modelleigenschaften unsinnig.

Kapitel 6

Verbesserte Auswertungsstrategie

Die Standard-Analyse des Güterverkehrszentrums in Kapitel 1.1 und die Standard-Analyse des schlecht initialisierten M/M/1-Servers in Kapitel 4.4 haben gezeigt, dass intelligentere Analysemethoden benötigt werden. Zum einen sollen diese Analysemethoden den Einfluss des Initialisierungszustandes berücksichtigen, und zum anderen überprüfen, ob ein stabiles System vorliegt. In diesem Kapitel werden einige Verfahren vorgestellt, die in Kombination miteinander eine Verbesserung in Aussicht stellen.

Modelle werden hier als eine nicht einsehbare Einheit betrachtet. Es sei zwar bekannt, aus welchen einzelnen Basiseinheiten das Modell zusammengesetzt ist, aber es sei nicht möglich, einzelne Abläufe oder Zusammenhänge zu erkennen. Dieses "Blackbox"-Vorgehen verhindert, dass instabile Modelle schon an ihrer Struktur erkannt werden können. Dies wäre zwar sinnvoll, ist aber nicht der Schwerpunkt dieser Diplomarbeit.

In den nächsten Abschnitten werden einige Methoden und Verfahren vorgestellt, die alle eine Bedingung an die zu analysierende Messreihe stellen. Hiermit soll sichergestellt werden, dass eine Messreihe eine bestimmte Eigenschaft besitzt. Diese Eigenschaften sind leider meistens keine hinreichenden Bedingungen, sondern nur notwendige Bedingungen für Stationarität. Das bedeutet, dass es fast immer Testfälle gibt, bei denen die Verfahren versagen. Daher ist es sinnvoll, dass der Benutzer die Ergebnisse der Verfahren überprüfen kann. Aus diesem Grund wurde eine graphische Darstellungsform entwickelt, die dem Benutzer Hilfestellung gibt (siehe Kapitel 6.3.2).

6.1 Theoretische Überlegungen

Bei der Analyse eines Systems stellt sich als erstes die Frage, ob ein stabiles System vorliegt. Ist eine Strukturanalyse möglich, kann bei einigen eingeschränkten Systemklassen die Stabilität direkt überprüft werden. In Kapitel 2.2 wurde die

Markov-Bedingung bereits diskutiert. In [7] und [12] werden Ansätze dieser Art weiter verfolgt.

Ist eine strukturelle Analyse nicht möglich, können nur die Simulationsergebnisse betrachtet werden. Wenn die Population (Warteschlangenlänge) eines Systems mit wachsender Modellzeit immer mehr anwächst und es keinen Grenzwert gibt, der die Population einschränkt, kann kein stabiles System vorliegen. Ebenfalls kann kein stabiles System vorliegen, wenn die Ankunftsrate an dem System größer ist als die Abgangsrate. Bei einem M/M/1-Server ist in diesem Fall die Verkehrsintensität $\rho > 1$ und die Stabilitätsbedingung ist verletzt.

Da der Initialisierungszustand Einfluss auf die beobachtete Messreihe hat, ist eine wichtige Aufgabe, zu bestimmen, ab wann dieser Einfluss vernachlässigbar ist. Diese Aufgabe ist jedoch schwierig, da der Einfluss auf die Messreihe ganz unterschiedliche Formen haben kann. Die Phase, in welcher der Einfluss nicht vernachlässigbar ist, wird die *transiente Phase* genannt. Manchmal endet der Einfluss zu einem konkreten Modellzeitpunkt. In diesem Fall kann das Ziel sein, den Modellzeitpunkt möglichst genau zu bestimmen. Manchmal sinkt der Einfluss negativ-exponentiell und ist daher, zumindest prinzipiell, immer vorhanden. In diesem Fall macht es nur Sinn, nach einem Modellzeitpunkt zu suchen, an dem der Einfluss vernachlässigbar klein ist. Es ist aber auch denkbar, dass der Einfluss des Initialisierungszustandes in gewissen Intervallen stärker oder schwächer auftritt. In dieser Situation ist es nicht sinnvoll, nach einem konkreten Modellzeitpunkt zu suchen, da es diesen Zeitpunkt nicht gibt. Daher wird die dritte Variante nicht weiter betrachtet.

Law und Kelton ([27], Kapitel 9.5) schlagen vor, die Daten der transienten Phase bei der Bestimmung der Verteilungsmomente außer Acht zu lassen. Dieses Vorgehen erscheint sinnvoll, da die Verteilungsmomente das "typische" (stationäre) Verhalten beschreiben sollen, die transiente Phase aber von einem "untypischen Zustand" beeinflusst wird. Auf den ersten Blick würde man erwarten, dass durch dieses Vorgehen die Simulationsdauer verlängert wird, weil Daten gelöscht werden. Auf den zweiten Blick erkennt man jedoch, dass das Gegenteil zu erwarten ist. Es müssen nicht mehr so viele Daten gesammelt werden, um den Einfluss des Startzustandes zu verringern.

Law und Kelton ([27], Kapitel 9.5) erwähnen ebenfalls, dass eine Analyse auf mehreren kurzen Simulationen oder auf einer langen Simulation basieren kann. Die einzelne lange Simulation hat den Vorteil, dass die transiente Phase nur ein einziges Mal durchlaufen werden muss. Der Nachteil ist jedoch, dass die während der Simulation gesammelten Daten stark voneinander abhängen. Die meisten Momentschätzer benötigen jedoch unabhängige Daten (siehe Kapitel 3.3). Wird mit mehreren Simulationsläufen gearbeitet, muss die transiente Phase für jeden Simulationslauf erneut durchlaufen werden. Wenn die Simulationsläufe

hintereinander durchgeführt werden, erhöht dies die Simulationsdauer. Es bietet sich daher an, die Simulationen parallel durchzuführen. Ein Vorteil bei mehreren Simulationsläufen ist, dass die Ergebnisse der einzelnen Simulationen nicht voneinander abhängig sind. Die Standard-Momentschätzer lassen sich also direkt anwenden. Hier muss allerdings die Annahme getroffen werden, dass der Zufallsgenerator einen "echten" Zufall erzeugt. Ist dies nicht der Fall, ist das gesamte Ergebnis der simulativen Auswertung zweifelhaft.

Die Daten, die bei der Simulation gesammelt werden, haben nur eine Aussagekraft in Bezug auf das "typische" Systemverhalten, wenn sie während eines "typischen" Systemverhaltens erzeugt werden. Also sind sowohl die Daten der transienten Phase unbrauchbar, als auch die Daten eines überlasteten Systems, da ein überlastetes System kein "typisches" Systemverhalten besitzt. Generell sind die Simulationsdaten nur dann interessant, wenn sich das "Systemverhalten" nicht mehr ändert und stabil ist. Wenn die Instabilität von überlasteten Systemen als eine unendlich lange transiente Phase aufgefasst wird, muss das Problem des Initialisierungszustandes und das Problem des instabilen Systems nicht mehr getrennt betrachtet werden. In den Kapiteln 6.2.1, 6.3.1 und 6.3.2 wird daher nicht mehr zwischen diesen zwei Problemen unterschieden, sondern nur noch die transiente Phase im Allgemeinen betrachtet.

6.2 Eine lange Simulation

Die Analyse auf Basis einer langen Simulation lässt sich in 3 Arbeitsschritte aufteilen, die nicht unbedingt in sequenzieller Reihenfolge stattfinden müssen. Ein

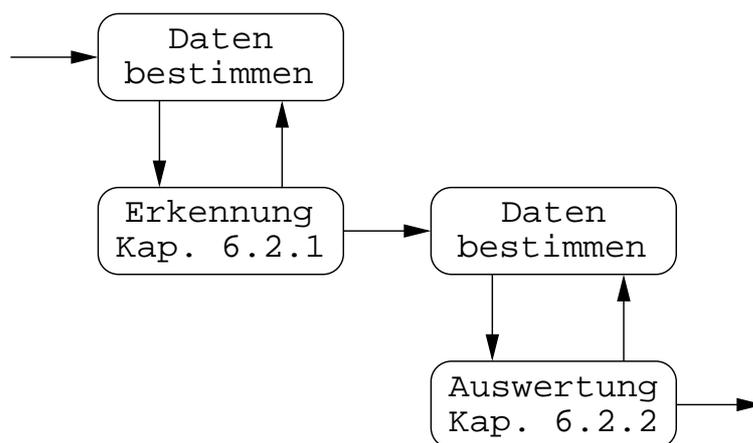


Abbildung 6.1: Ablaufschema der Erkennung der stationären Phase mit anschließender Datenauswertung für eine lange Simulation.

Arbeitsschritt ist die Erfassung der Daten, welche die Simulation zur Verfügung

stellt. Hierbei ist darauf zu achten, dass die Daten in einer geeigneten Datenstruktur abgelegt werden. Langes Suchen oder Sortieren der Daten verhindert eine zeit-effiziente Berechnung benötigter Werte. Ein weiterer Arbeitsschritt ist die Bestimmung des Endes der transienten Phase. Dieser Arbeitsschritt wechselt so lange mit der Erfassung der Daten, bis das Ende gefunden ist. Nun kann zur Bestimmung der Momente übergegangen werden. Dies ist ein weiterer Arbeitsschritt. Wenn auch die Momente bestimmt wurden und die Validität der Werte zufriedenstellend ist, stehen die Ergebnisse fest. Sollte die Validität nicht ausreichen, müssen weitere Daten gesammelt werden.

6.2.1 Erkennung der stationären Phase

Bei der Simulation werden Startbedingungen benötigt, um das vorliegende Modell zu initialisieren. Einen "typischen Systemzustand" zu wählen, wäre die naheliegenste Möglichkeit. Leider ist diese Initialisierung nicht möglich, da das Ziel der Simulation die Bestimmung des typischen Systemzustandes ist. Es würden Daten benötigt werden, die noch nicht bekannt sind. Aus diesem Grund ist der Initialisierungszustand in den meisten Fällen untypisch. Zum Beispiel könnten die Startbedingungen eines Modells einer Fertigungsanlage so gewählt sein, dass sich alle Maschinen und Arbeiter im Ruhezustand befinden. Diese Art der Initialisierung wird "*empty & idle*" genannt.

Wie bereits erwähnt, ist ein häufiges Ziel der Simulation die Bestimmung des typischen Modellverhaltens. Da das Modell allerdings in einem untypischen Zustand gestartet wird, verfälschen die ersten Daten die Auswertung (initialization bias). Wird dies bei der Auswertung nicht berücksichtigt, neigt die Schätzung der stationären Zustandsverteilung zu einem Fehler in Richtung des Initialisierungszustandes. Der Schätzwert ist zu klein bzw. zu groß.

Eine gängige Methode diesen Fehler zu vermeiden ist, dass dem Modell während der Simulation Zeit gegeben wird, sich von den Startbedingungen zu erholen und auf einen typischen Zustand einzupendeln (*warm-up*). Während des "warm-up" befindet sich das Modell in einer *transienten Phase*. Praktisch gesehen werden die Daten, die während der transienten Phase gesammelt werden, bei der Auswertung nicht berücksichtigt und können gelöscht werden.

Nun stellt sich die Frage: "Wann endet die transiente Phase?" Gesucht ist der Modellzeitpunkt, ab dem das Modell Stationarität aufweist (*truncation point*). Bei der Wahl dieses Zeitpunktes ist folgendes zu beachten:

- Wird der Zeitpunkt zu groß gewählt, gehen nützliche Daten verloren. Die Simulation muss mehr Daten liefern, damit die Schätzungen von Modellgrößen in bestimmten Konfidenzintervallen liegen.

- Wird der Zeitpunkt zu klein gewählt, besteht noch immer ein Fehler bezüglich der Startbedingungen.

Die nun folgenden Abschnitte beschreiben Regeln, die benutzt werden können, um das Ende der transienten Phase zu bestimmen. In [30] und [35] befinden sich Zusammenstellungen der Regeln. Interessante Veröffentlichungen, die sich mit der Güte einiger Regeln beschäftigen, sind [20] und [28].

Conway-Regel

Die transiente Phase endet vor dem Messwert, der als Erster weder das Maximum, noch das Minimum von den *nachfolgenden* Messwerten ist. (Siehe [14])

Beispiel: Man betrachte die Messwerte $\{0; 2; 4; 6; 5; 7; 5; 6; 6; 5; 6; 7\}$. Nach der Conway-Regel wird die Sequenz $\{0; 2; 4\}$ bei der Auswertung nicht berücksichtigt. Die Conway-Regel macht die Annahme, dass die stationäre Phase mit der vierten Messung beginnt.

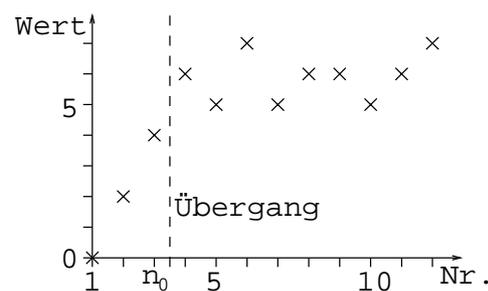


Abbildung 6.2: Die Conway-Regel angewendet auf eine einzelne Messreihe.

Die Conway-Regel liefert nur dann akzeptable Ergebnisse, wenn der Werteverlauf während der transienten Phase monoton ist. Sollte dies nicht der Fall sein, kann die Schätzung des Endes der transienten Phase beliebig schlecht werden.

Modifizierte Conway-Regel

Die transiente Phase endet vor dem Messwert, der als Erster weder das Maximum, noch das Minimum von den *vorherigen* Messwerten ist. (Siehe [20])

Beispiel: Man betrachte die Messwerte $\{0; 2; 4; 6; 5; 7; 5; 6; 6; 5; 6; 7\}$. Nach der modifizierten Conway-Regel wird die Sequenz $\{0; 2; 4; 6\}$ bei der Auswertung nicht berücksichtigt. Die modifizierte Conway-Regel macht die Annahme, dass die stationäre Phase mit der fünften Messung beginnt.

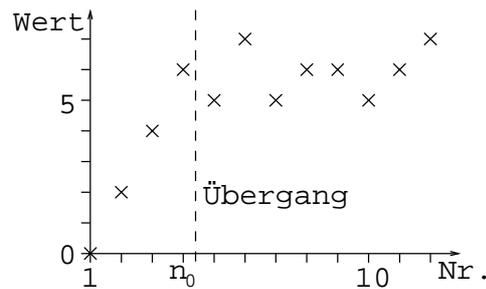


Abbildung 6.3: Die modifizierte Conway-Regel angewendet auf eine einzelne Messreihe.

Die Probleme bei der modifizierten Conway-Regel sind dieselben wie die der Conway-Regel. Wenn keine Monotonie während der transienten Phase vorliegt, können die Ergebnisse beliebig schlecht sein.

Tochter-Regel

Der längste Zyklus des Modells sollte 3 bis 4 mal durchlaufen werden, bevor man annehmen kann, dass der Einfluss der Startbedingung vernachlässigbar ist (Siehe [38]). Mit Zyklus ist hier kein Zyklus der Messreihe der Simulationsdaten gemeint, sondern ein Modellzeitzyklus, der durch die Struktur des Modells vorgegeben wird.

Beispiel: Wenn in einem Modell der längste Zyklus 25 Modellzeiteinheiten dauert, macht die Tochter-Regel die Annahme, dass die stationäre Phase bei der Modellzeit 100 beginnt.

Bei der Tochter-Regel wird zusätzliches Wissen über das Modell benötigt. Ist dieses Wissen nicht vorhanden, kann die Regel nicht angewendet werden. Ist dieses Wissen jedoch vorhanden, können auch bei dieser Regel die Ergebnisse beliebig schlecht sein, da die "Daumenregel" von 3 bis 4 Durchläufen nicht in jedem Fall zutreffen muss.

Relaxationsregel

In vielen physikalischen Gesetzen wird davon ausgegangen, dass der Einfluss eines Anfangszustandes mit der Funktion $e^{-\frac{t}{\tau}}$ abklingt. Wenn dies auf die Simulation übertragen wird, ist τ eine modellabhängige Konstante ("relaxation time"). Um die Frage zu beantworten, wann die transiente Phase endet, wird die Umkehrfunktion gebildet:

$$t_{\beta} = -\tau \ln \beta \quad \text{mit} \quad 0 < \beta < 1 \quad . \quad (6.1)$$

β ist hierbei ein relatives Maß für das Abklingen des Anfangszustandes.

Beispiel: Wird der Zeitpunkt gesucht, an dem der Einfluss des Anfangszustandes nur noch 5% ist, wird $\beta := 0.05$ gesetzt. Die Formel ergibt dann $t_{0,05} = 3\tau$.

In [30] wird näher auf die Modellkonstante τ eingegangen. Es wird erwähnt, dass aus Markov-Modellen eine Formel abgeleitet werden kann, die τ für einige Modellklassen gut beschreibt. Im Allgemeinen steht fest, dass τ für hoch ausgelastete Systeme größer ist als für weniger ausgelastete Systeme. Bei der Relaxationsregel

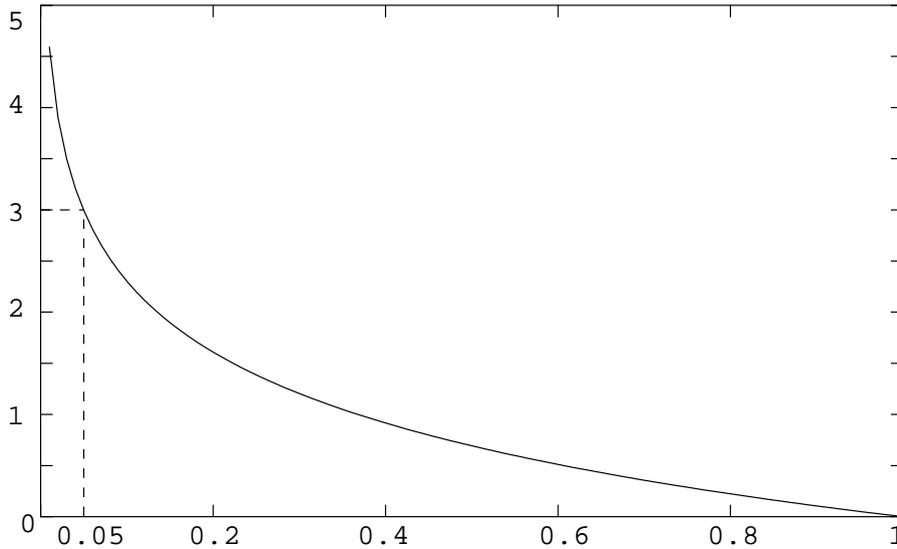


Abbildung 6.4: t_β in Abhängigkeit von β , mit $\tau = 1$.

wird ebenfalls zusätzliches Wissen über das Modell vorausgesetzt. Der Benutzer muss die Konstante τ vorgeben. Die Annahme, dass der Einfluss des Startzustandes durch die Funktion 6.1 beschrieben werden kann, muss nicht immer zutreffen. Diese Annahme stimmt nur für eine eingeschränkte Klasse von Modellen und ist daher auch nur auf diese eingeschränkte Klasse anwendbar.

Konstanter-Mittelwert-Regel

Der Mittelwert $\bar{X}(i)$ der ersten i Messwerte einer Sequenz sei der *laufende Mittelwert* (“running mean”). Wenn sich der laufende Mittelwert ab dem Messwert n_0 nicht mehr signifikant ändert, sind alle nachfolgenden Messwerte Teil der stationären Phase (siehe [15]). Konkret kann die signifikante Änderung definiert werden, als eine Prozentangabe $100\delta\%$. Alle laufenden Mittelwerte, die auf n_0 folgen, sollten nicht mehr als δ von dem letzten laufenden Mittelwert abweichen (siehe [30]):

$$\forall i : n_0 < i \leq n_0 + k \quad \text{gilt} \quad \frac{|\bar{X}(n_0 + k) - \bar{X}(i)|}{|\bar{X}(n_0 + k)|} < \delta \quad (6.2)$$

Beispiel: Die Messwerte $\{0; 2; 4; 6; 5; 7; 5; 6; 6; 5; 6; 7\}$ habe die laufenden Mittelwerte $\{0; 1; 2; 3; 3.4; 4; 4.14; 4.38; 4.56; 4.6; 4.73; 4.92\}$. Ist ein Unterschied von 10% zulässig ($\delta = 0.1$), gilt die Gleichung ab Messwert 9 und die transiente Phase endet (Annahme!) bei dem achten Messwert.

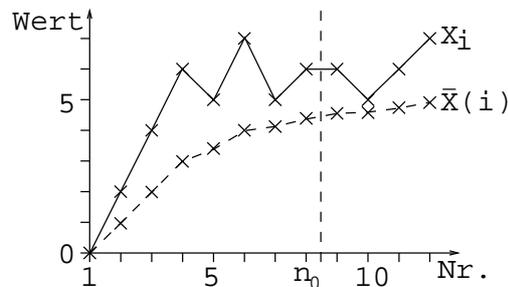


Abbildung 6.5: Der laufende Mittelwert im Vergleich mit den eigentlichen Messwerten.

Die Konstanter-Mittelwert-Regel basiert auf der Berechnung des Mittelwertes. Der Mittelwert ist im Allgemeinen das offensichtlichste Moment. Daher ist diese Regel auch sehr anschaulich. Mathematisch gesehen wird hier die Konvergenz des Mittelwertes (bzw. des laufenden Mittelwertes) ausgenutzt. Es ist denkbar, dass der Mittelwert konvergiert, obwohl die eigentliche Verteilung diese Eigenschaft nicht besitzt. In diesem Fall würde die Konstante-Mittelwert-Regel versagen.

Gruppen-Mittelwert-Regel

Die vorliegende Sequenz von Messwerten wird in Teilbereiche (Gruppen) aufgeteilt, die jeweils ca. $m_0 = 30$ Messwerte enthalten sollten. Von diesen Gruppen wird der Mittelwert $\bar{X}_g(m_0)$ berechnet. Sind diese Mittelwerte relativ konstant, liegt wahrscheinlich die stationäre Phase vor (siehe [15]). Wenn nach g_0 Gruppen (bzw. nach $n_0 = g_0 m_0$ Messwerten) die Differenzen aller nachfolgenden Gruppen kleiner als δ sind, seien die ersten n_0 Messwerte die transiente Phase. Diese Bedingung lässt sich mathematisch so formulieren (siehe [30]):

$$\forall i, j : 0 < i < k - 1, 0 < j < k - 1 \quad \text{gilt} \quad |\bar{X}_{g_0+i}(m_0) - \bar{X}_{g_0+j}(m_0)| < \delta \quad (6.3)$$

Die Gruppen-Mittelwert-Regel hat gegenüber der Konstante-Mittelwert-Regel den Vorteil, dass sie hohe Schwankungen während der stationären Phase ausgleicht. Außerdem ist der Rechenaufwand für viele Messwerte überschaubarer. Dafür

kann sie nur ganze Gruppen der transienten bzw. der stationären Phase zuordnen, wodurch das Verfahren ungenauer ist. Sie basiert ebenfalls auf der Konvergenz des Mittelwertes. Dies führt zu Problemen, wenn der Mittelwert konvergiert, aber die eigentliche Verteilung nicht.

Balance-Regel

Man betrachte eine Sequenz von Messwerten. Sei x_g die Anzahl der Messwerte, die größer sind als der Durchschnitt und sei x_k die Anzahl der Messwerte, die kleiner sind als der Durchschnitt. Wenn $x_g = x_k$ gilt, scheint die Sequenz aus Messwerten der stationären Phase zu bestehen. (Siehe [15])

Beispiel: 4.92 ist ungefähr der Durchschnitt der Messwerte $\{0; 2; 4; 6; 5; 7; 5; 6; 6; 5; 6; 7\}$. Daher ist in diesem Fall $x_g = 9$ und $x_k = 3$. Offensichtlich hat die transiente Phase noch Einfluss auf die Sequenz. Die Sequenz $\{7; 5; 6; 6; 5; 6; 7\}$ besitzt den Mittelwert 6 und es gilt $x_g = x_k = 2$. In diesem Fall macht die Balance-Regel die Annahme, dass die Sequenz aus Messwerten der stationären Phase besteht.

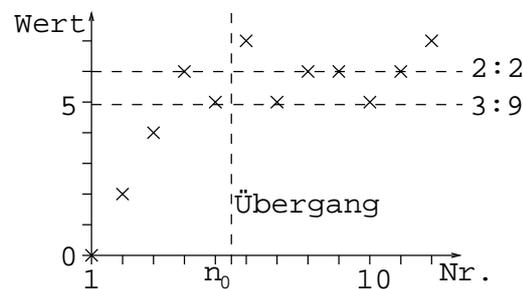


Abbildung 6.6: Die Balance-Werte für die unveränderte Sequenz (Mittelwert bei 4.92) und für die um 5 Werte gekürzte Sequenz (Mittelwert bei 6).

Aus dem Beispiel wird auch das algorithmische Vorgehen deutlich, um die Messwerte der transienten Phase auszuschließen. Es liegt nahe, dass so lange Daten vom Anfang einer gegebenen Sequenz abgeschnitten werden, bis die Balance-Regel (ungefähr) zutrifft. Auch diese Regel basiert auf der Konvergenz des Mittelwertes. Dies kann zu den bereits erwähnten Problemen führen.

Balance- χ^2 -Regel

Eine Erweiterung der Balance-Regel kann durch den χ^2 -Test (siehe [27]) erreicht werden. Die Sequenz der Messwerte, mit dem Mittelwert $\bar{X}(n)$, wird in Teilbereiche (Gruppen) unterteilt, die ca. 30 Messwerte enthalten. Für jede dieser Gruppen

wird nun x_g und x_k bestimmt, mit dem Mittelwert $\bar{X}(n)$. Alle x_g zusammen ergeben ein Histogramm, genauso wie alle x_k . Diese beiden Histogramme werden mit dem χ^2 -Test verglichen. In Abhängigkeit davon wird entweder Stationarität angenommen, oder die erste Gruppe wird der transienten Phase zugeordnet und die verbleibenden Gruppen werden erneut verglichen (siehe [30]). Der Mittelwert muss ebenfalls neu bestimmt werden. Das grundlegende Problem, dass der Test

| | | | | | | |
|---------|---|----|----|----|----|----|
| x_g : | 0 | 7 | 17 | 15 | 17 | 16 |
| | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ |
| x_i : | \bar{X}_1 bis X_{30} X_{31} bis X_{60} X_{61} bis X_{90} X_{91} bis X_{120} X_{121} bis X_{150} X_{151} bis X_{180} | | | | | |
| | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |
| x_k : | 30 | 23 | 13 | 15 | 13 | 14 |

Abbildung 6.7: Die Einteilung in Gruppen und die Berechnung der x_g und x_k . In diesem Beispiel hat der Initialisierungszustand einen negativen Einfluss auf den Mittelwert, da alle Messwerte der ersten Gruppe kleiner als der Mittelwert sind.

auf der Konvergenz des Mittelwertes basiert, bleibt auch bei der Verwendung des χ^2 -Tests bestehen.

Kreuzen des Mittelwertes

Die transiente Phase endet beim Messwert n_0 , wenn die Sequenz der Messwerte den Mittelwert $\bar{X}(n_0)$ k -mal kreuzt (siehe [19]).

Beispiel: Sei $k=2$. Die beobachtete Sequenz sei $\{0; 2; 3; 1; 2; 1; 3; 2; 2; 2; 3; 1\}$ und deren laufende Mittelwerte sind $\{0; 1; 1.33; 1.5; 1.6; 1.5; 1.71; 1.75; 1.78; 1.8; 1.91; 1.83\}$. Bei Anwendung dieser Regel wird angenommen, dass die transiente Phase an dem Messwert $n_0 = 5$ endet, da sich beide Sequenzen zweimal gekreuzt haben.

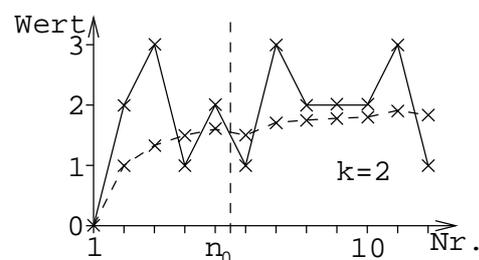


Abbildung 6.8: Der Vergleich des laufenden Mittelwertes mit den eigentlichen Messwerten.

Diese Regel ist sehr anschaulich und für einen Benutzer leicht nachzuvollziehen. Allerdings gibt es gleich zwei grundsätzliche Probleme. Das erste Problem ist, dass die Regel wieder auf der Konvergenz des Mittelwertes basiert. Dies führt zu den bereits erwähnten Schwierigkeiten. Das zweite Problem ist, dass das Kreuzen des Mittelwertes nur indirekt mit dem Einfluss des Startzustandes in Verbindung steht. Somit ist die Bedingung nur eine "Daumenregel", die beliebig schlechte Ergebnisse liefern kann.

Schrubens-Universeller-Test

Schruben [32] beschreibt die Zufallsvariablen eines Prozesses durch

$$Y_i = \mu_i + X_i \quad \text{für } i = 1, 2, \dots, n \quad . \quad (6.4)$$

μ_i repräsentiere die mögliche Verschiebung des Mittelwertes, welche durch die Initialisierung entsteht und es sei $\mu_i = E[Y_i]$. Wenn keine transiente Phase vorhanden ist, ist die Sequenz aller μ_i ($i = 1, 2, \dots, n$) konstant. Außerdem besitzt μ_i keinen Zufallscharakter und ist daher für verschiedene Replikationen ebenfalls konstant. X_i beschreibe die zufälligen Abweichungen der Ziehungen aus Y_i von μ_i . Da $\mu_i = E[Y_i]$ vorausgesetzt wird, muss $E[X_i] = 0$ gelten.

Die Sequenz $S_n(k) = \bar{Y}_n - \bar{Y}_k$ mit $k = 1, 2, \dots, n$ beschreibt den Unterschied zwischen dem Mittelwert aller Beobachtungen und dem Mittelwert der ersten k Beobachtungen. Diese Sequenz lässt sich mit der obigen Beschreibung der Zufallsvariablen auch so formulieren:

$$S_n(k) = M_n(k) + X_n(k) \quad \text{mit} \quad (6.5)$$

$$M_n(k) = \bar{\mu}_n - \bar{\mu}_k \quad \text{und} \quad (6.6)$$

$$X_n(k) = \frac{\sum_{i=1}^n X_i}{n} - \frac{\sum_{i=1}^k X_i}{k} \quad . \quad (6.7)$$

$M_n(k)$ ist hierbei die eigentliche Information (Schruben: the signal) und $X_n(k)$ ist ein "Rauschen", das die Information verschleiert (Schruben: the noise process).

Ein "Browning Bridge"-Prozess (kurz: BBP) hat die Eigenschaft, dass

- $BBP_0 = BBP_1 = 0$, und
- $E[BBP_t] = 0$ für $0 \leq t \leq 1$, und
- $Cov(BBP_{t_1}, BBP_{t_2}) = \min(t_1, t_2)(1 - \max(t_1, t_2))$, und
- mehrere Ziehungen aus BBP_t sind gemeinsam normal-verteilt.

Schruben standardisiert den beobachteten Prozess so, dass er bei langer Laufzeit gegen den BBP konvergiert. So ergibt sich für $X_n(k)$ der standardisierte Prozess

$$B_t = \frac{tnX_n(tn)}{\sqrt{n}\sigma} \quad \text{mit} \quad t = \frac{1}{n}, \frac{2}{n}, \dots, 1 \quad . \quad (6.8)$$

Für $M_n(k)$ ergibt sich der standardisierte Prozess

$$D_t = \frac{tnM_n(tn)}{\sqrt{n}\sigma} \quad \text{mit} \quad t = \frac{1}{n}, \frac{2}{n}, \dots, 1 \quad . \quad (6.9)$$

Mit der Kenntnis der standardisierten Prozesse von $X_n(k)$ und $M_n(k)$ kann nun auch ein standardisierter Prozess für $S_n(k)$ angegeben werden:

$$T_t = \frac{tnS_n(tn)}{\sqrt{n}\sigma} \quad \text{mit} \quad t = \frac{1}{n}, \frac{2}{n}, \dots, 1 \quad . \quad (6.10)$$

Die Graphen der standardisierten Prozesse schneiden die horizontale Achse bei 0 und 1. Existiert eine transiente Anfangsphase, hat der Graph von D_t ein frühes (kleines t) Maximum. Dieses Maximum lässt sich auch in T_t wiederfinden. Dieser Zusammenhang ist die Basis des Tests. (Siehe [32], Abbildung 1 und Abbildung 2)

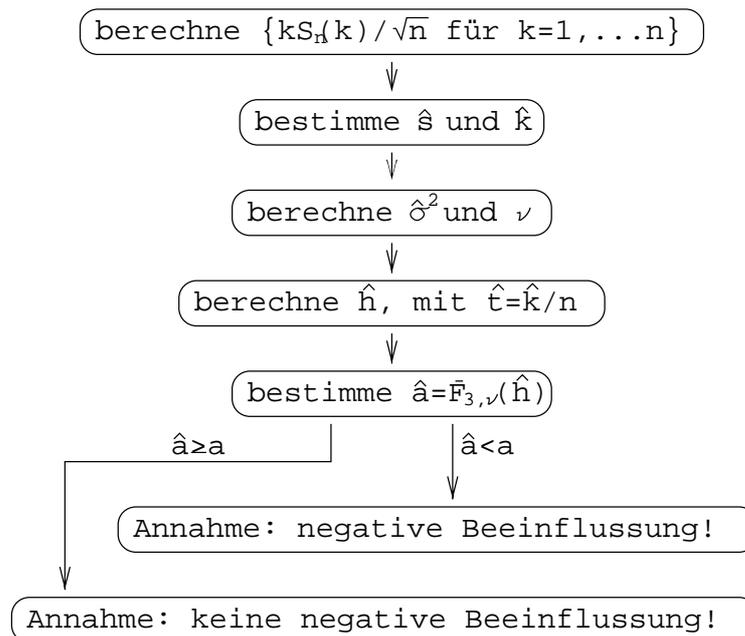


Abbildung 6.9: Verlaufsdiagramm von Schrubens Universellem-Test. Ein Test auf positive Beeinflussung oder ein zweiseitiger Test kann, wie in [30] beschrieben, ergänzt werden. (a sei α)

Für einen BBP lässt sich bestimmen, wie wahrscheinlich ein Maximum mit dem Wert s^* zum Zeitpunkt t^* ist. Es muss folglich der Wert des Maximums \hat{s} und die Position des Maximums \hat{k} des Graphen von T_t bestimmt werden. Nun kann

überprüft werden, wie wahrscheinlich diese Werte für den BBP wären. Wenn die Wahrscheinlichkeit zu gering ist, muss angenommen werden, dass eine transiente Phase vorhanden ist, welche die weitere Analyse beeinflusst. Unter gewissen Voraussetzungen und Bedingungen (siehe [32], Kapitel 3.1 bis 3.3) hat

$$\hat{h} = \frac{\hat{s}^2}{3\hat{\sigma}^2\hat{t}(1-\hat{t})} \quad (6.11)$$

eine F-Verteilung mit 3 und ν Freiheitsgraden. Die Wahrscheinlichkeit $\hat{\alpha} = \bar{F}_{3,\nu}(\hat{h})$ ist die Wahrscheinlichkeit, mit der \hat{s} und \hat{t} aus einem BBP stammen. Ist $\hat{\alpha}$ kleiner als ein bestimmter Grenzwert α , ist anzunehmen, dass eine transiente Phase vorhanden ist.

Es wurde noch nicht erwähnt, auf welche Weise die Varianz $\sigma^2 = Var(S_n)$ und die Anzahl der Freiheitsgrade ν bestimmt werden können. Schruben beschreibt die Bestimmung mittels autoregressiver Modelle und verweist dabei auf Fishman ([18]). Pawlikowski bevorzugt in [30] jedoch die Methode von Heidelberger und Welch ([22]).

Schrubens-Spezieller-Test

Dieser Test von Schruben ([34]) setzt voraus, dass dem Benutzer bekannt ist, wie sich die transiente Phase verhält. Hierbei ist nicht gemeint, dass die Länge der transienten Phase bekannt ist, sondern es sei bekannt, wie sich der von Schruben definierte Störfaktor verhält. In den folgenden Beschreibungen wird deutlich, dass der Benutzer die Größe c_i (wird später definiert) vorgeben muss. Die optimale Wahl von c_i hängt hierbei von der beobachteten Sequenz ab.

Die Sequenz der Beobachtungen sei Y_1, \dots, Y_n und es sei

$$E[Y_i] = \mu_i = \mu(1 - a_i) \quad . \quad (6.12)$$

Die Unbekannte μ sei der Erwartungswert der stationären Zustandsverteilung und a_i sei der Störfaktor. Wenn es keine transiente Phase gibt, ist $a_i = 0$ für alle i .

Die Sequenz $S_n(k) = \bar{Y}_n - \bar{Y}_k$ mit $k = 1, 2, \dots, n$ beschreibt den Unterschied zwischen dem Mittelwert aller Beobachtungen und dem Mittelwert der ersten k Beobachtungen. Unter Berücksichtigung einiger Einschränkungen (Y_i ist nicht unkorreliert, usw.) kann mit Hilfe des Neyman-Pearson Lemma die gemeinsame Dichteverteilung $f(S_1, \dots, S_{n-1}; a_1, \dots, a_n)$ von $\{S_n(k), k = 1, \dots, n-1\}$ angegeben werden. Da diese Dichteverteilung auch von a_1 bis a_n abhängt und die Annahme $\forall i : a_i = 0$ überprüft werden soll, ist der Quotient

$$\frac{f(S_1, \dots, S_{n-1}; a_1, \dots, a_n)}{f(S_1, \dots, S_{n-1}; 0, \dots, 0)} \quad (6.13)$$

von besonderer Bedeutung. Es stellt sich heraus, dass für $\mu > 0$ der Quotient eine wachsende Funktion von

$$T = \sum_{i=1}^n i(a_i - a_{i+1})S_n(i) = \sum_{i=1}^n ic_i S_n(i) \quad \text{mit} \quad c_i = a_i - a_{i+1} \quad (6.14)$$

ist. Die Werte c_i können als eine Gewichtung aufgefasst werden. Schruben schlägt hier die Gewichtung

$$c_i = 1 - \frac{i}{n} \quad (6.15)$$

vor. Denkbar sind auch andere Gewichtungen. Die optimale Gewichtung hängt unter anderem davon ab, ob eine positive oder eine negative Beeinflussung des Mittelwertes durch eine transiente Phase vorliegt. Der Betrag der Gewichtung sollte (monoton) fallend sein, da der Prozess sonst nicht stationär wäre.

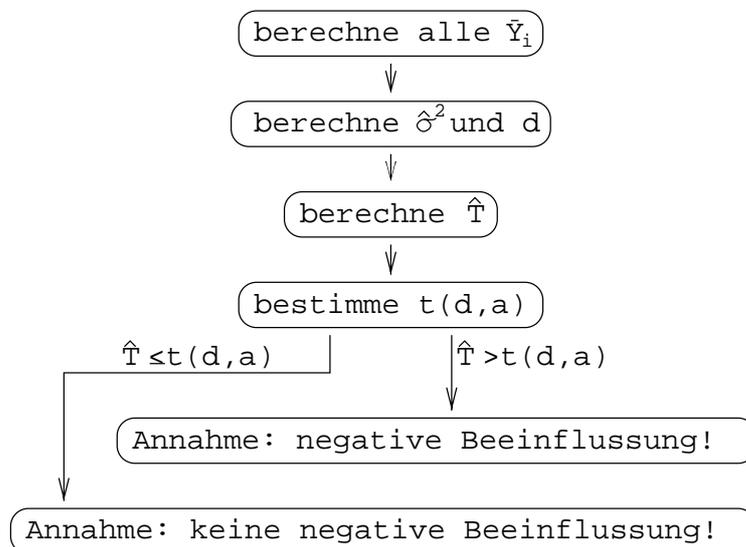


Abbildung 6.10: Verlaufsdiagramm von Schrubens Speziellem-Test. Ein Test auf positive Beeinflussung oder ein zweiseitiger Test kann ergänzt werden. (a sei α)

Es sei $\sigma^2 = \lim_{n \rightarrow \infty} n \text{Var}(\bar{Y}_n)$. Mit Schrubens Wahl der Gewichtung ist T normalverteilt mit

- $E(T) = 0$ und
- $\text{Var}(T) = \frac{n^3 \sigma^2}{45}$.

σ wird geschätzt (siehe [19] und [22]) durch $\hat{\sigma}$. Der Wert $\frac{d\hat{\sigma}}{\sigma}$ hat eine χ^2 -Verteilung mit d Freiheitsgraden. Und die normierte Verteilung

$$\hat{T} = \frac{T}{\text{Var}(T)} = \frac{\sqrt{45}}{n^{\frac{3}{2}} \hat{\sigma}} \sum_{i=1}^n \left(1 - \frac{i}{n}\right) i (\bar{Y}_n - \bar{Y}_k) \quad (6.16)$$

ist t-verteilt mit d Freiheitsgraden. Wenn $\hat{T} > t(d, \alpha)$ ist, muss angenommen werden, dass eine negative Beeinflussung des Mittelwertes durch eine transiente Phase stattfindet. Durch einen Vorzeichenwechsel von \hat{T} kann ein Test auf positive Beeinflussung des Mittelwertes durchgeführt werden. $t(d, \alpha)$ ist das obere $100\alpha\%$ -Quantil der t-Verteilung mit d Freiheitsgraden.

Ist ein Test erwünscht, der sowohl auf negative als auch auf positive Beeinflussung prüft, muss $|T| > t(d, \frac{\alpha}{2})$ getestet werden. Die Signifikanz α muss hierbei halbiert werden ([30]).

Gordon-Regel

Die Varianz eines stationären Prozesses ist umgekehrt proportional zur Anzahl der Beobachtungen. Dies legt folgende Gleichung nahe ([20]):

$$\hat{\sigma}^2[\bar{X}(n)] = \frac{C}{n} + o\left(\frac{1}{n}\right) \quad (6.17)$$

Hierbei sei $\hat{\sigma}^2[\bar{X}(n)]$ ein Schätzer für die Varianz von dem Mittelwert $\bar{X}(n)$. C sei eine modellabhängige Konstante und $o(\cdot)$ sei die übliche o-Notation ([39]).

Die Gordon-Regel besagt, dass die transiente Phase nach n_0 Beobachtungen beendet ist, wenn der Graph $(\log(n), \log(\hat{\sigma}[\bar{X}(n)]))$ ab der Beobachtung n_0 einer Geraden mit der Steigung -0.5 ähnelt (siehe [21], Figure 15-4).

Bisher basierten die meisten Tests auf den Eigenschaften des Mittelwertes. Die Gordon-Regel nutzt hingegen eine Eigenschaft der Varianz. Die Varianz ist allerdings nur eine Abstraktion von der eigentlichen Verteilung. Daher kann es zu Problemen kommen, wenn die Eigenschaft der Varianz auf Stationarität deutet, die Verteilung aber nicht konvergiert.

Autokorrelations-Regel

Auf Grund des Ansatzes von Fishman ([17]) formuliert Shannon ([35]) folgende Idee:

Es wird die längste Modellzeitperiode bestimmt, bei der die Messwerte signifikant korreliert sind. Die Länge dieser Periode sei die Länge der transienten Phase.

Zur Bestimmung der Modellzeitperiode kann das Korrelogramm betrachtet werden. Es sei $r_{\tau_{max}}$ das Maximum des Betrages der Autokorrelationsfunktion $|(r_\tau)|$ und τ_{max} sei die gesuchte Modellzeitperiode.

Es besteht zwar kein direkter Zusammenhang zwischen τ_{max} und der Länge der transienten Phase, allerdings ist die Annahme durchaus sinnvoll, dass τ_{max} ein Modellzyklus ist, der zur Initialisierung erst durchlaufen werden muss.

Diese Regel ist das Gegenstück zur Tocher-Regel. Bei der Tocher-Regel lag die Sichtweise eher auf dem Modell, während bei der Autokorrelations-Regel die Messwerte im Vordergrund stehen.

Erweiterte-Autokorrelations-Regel

Eine ähnliche Regel, wie die Autokorrelations-Regel, formulierte Pawlikowski ([30]) aus dem Ansatz von Fishman ([17]):

Es wird ein korrelierter Prozess mit einem hypothetischen unkorrelierten Prozess verglichen. Der korrelierte Prozess habe den Erwartungswert μ , die Varianz (bzw. die Kovarianz mit Zeitabstand 0) c_0 und bestehe aus n Beobachtungen. Die Ereignisse des unkorrelierten Prozesses haben ebenfalls alle den Erwartungswert μ und die Varianz c_0 . Dann haben l Ereignisse des hypothetischen Prozesses die Varianz $\frac{c_0}{l}$. c_k sei die Autokovarianzfunktion und r_k sei die Autokorrelationsfunktion. Es gilt (nach [17], Gleichung 2.5):

$$\text{Var}(\bar{X}(n)) = \frac{\sum_{k=1}^{n-1} (1 - \frac{|k|}{n}) c_k}{n} \quad , \quad (6.18)$$

und daraus wird (nach [30], R10) und der Annahme, dass $c_k = c_{-k}$ gilt

$$\text{Var}(\bar{X}(n)) = \frac{2 \sum_{k=1}^{n-1} (1 - \frac{k}{n}) c_k}{n} \quad . \quad (6.19)$$

Wird die hypothetische Varianz gleich der realen Varianz gesetzt

$$\frac{c_0}{l} = \text{Var}(\bar{X}(n)) \quad , \quad (6.20)$$

lässt sich folgende Formel ableiten:

$$n_o = \frac{n}{l} = 2 \sum_{k=1}^{n-1} (1 - \frac{k}{n}) \frac{c_k}{c_0} = 2 \sum_{k=1}^{n-1} (1 - \frac{k}{n}) r_k \quad . \quad (6.21)$$

Die eigentliche Interpretation von n_o ist diese: Wenn vom korrelierten Prozess n_o Beobachtungen gesammelt werden, ist das äquivalent zum Sammeln einer Beobachtung im unkorrelierten Prozess. Daher ist zu erwarten, dass bei dem Zeitabstand n_o für die Korrelation $r_{n_o} \approx 0$ gilt. Die transiente Phase ist beendet, wenn der Einfluss des Anfangszustandes abgeklungen ist. Anders gesagt: Wenn die Beobachtungen von der ersten Beobachtung unkorreliert sind. Und das tritt nach n_o Beobachtungen ein.

Bewertung

Es ist ein sehr schwieriges Problem, für die aufgeführten Regeln eine quantitative Bewertung durchzuführen. Bei der Beschreibung der einzelnen Regeln wurde schon festgestellt, dass sie für unterschiedliche Modellklassen Ergebnisse mit

unterschiedlicher Qualität liefern. Um eine “gerechte” quantitative Bewertung durchführen zu können, müssten zunächst geeignete Modellklassen gefunden werden. Die Einteilung sollte aber nicht anhand der Modellstruktur erfolgen, sondern nach der Form von dem Einfluss des Initialisierungszustands auf die Messreihe. Ein Ansatz dieser Klassifizierung wurde schon in Kapitel 6.1 vorgestellt. Ein weiteres Klassifizierungsmerkmal könnte die Monotonie der Messreihe sein. Diese Einteilungsvorschläge sind allerdings nur ein erster Ansatz. Genauere Betrachtungen würden über den Rahmen dieser Diplomarbeit hinausgehen. Die qualitative Bewertung, die zu den einzelnen Regeln gegeben wurde, soll im Folgenden ausreichend sein.

Gafarian, Ancker und Morisaku haben in [20] einen Versuch zur Bewertung ausgewählter Regeln unternommen. Sie erarbeiteten die Bewertungskriterien Allgemeingültigkeit (“Generality”), Richtigkeit (“Accuracy”), Präzision (“Precision”), Effizienz (“Cost”) und Einfachheit (“Simplicity”). Mit einem M/M/1-Server führten sie mehrere Versuchsreihen mit verschiedenen Parametern durch. Untersucht wurde die Conway-Regel, die modifizierte Conway-Regel, die “Kreuzen des Mittelwerts”-Regel, die Geraden-Regel (wird in 6.3.1 erläutert) und die Gordon-Regel. Sie kamen zu dem Ergebnis, dass keine der Regeln den Anforderungen entspricht: “The principal conclusion is that none of the five rules is satisfactory and that they should not be recommended for use by practitioners.” Die empirischen Ergebnisse der Versuche legen diese Schlussfolgerung nahe, aber nur für die Simulation mit einem M/M/1-Server. Diese Schlussfolgerung muss nicht für alle denkbaren Modelle gelten. Die empirischen Versuche von Gafarian, Ancker und Morisaku zeigen jedoch, dass bei einem “ungünstigen” Modell alle Regeln versagen können.

Lee, McNickle und Pawlikowski beschreiben in [28] einige empirische Versuche mit den Testverfahren von Schruben (siehe [32] und [34]). Auch hier wird ausschließlich das Modell eines M/M/1-Servers benutzt. In diesem Fall lassen sich daher leider ebenfalls keine allgemeingültigen Schlussfolgerungen ziehen.

Im Laufe der Zeit wurden immer mehr Regeln aufgestellt, die bei gewissen Modellen bessere Ergebnisse liefern. Es ist jedoch noch nicht gelungen, eine Regel aufzustellen, die für alle denkbaren Formen des Einflusses des Anfangszustandes gleichbleibend gute Ergebnisse liefert. Vielleicht ist das Simulationsverfahren mit nur einem Simulationslauf eine Sackgasse, bei dem Ziel, ein Verfahren zu entwickeln, das nicht nur bei speziellen Modellen verlässliche Ergebnisse liefert.

6.2.2 Auswertung der stationären Phase

Ist eine Modellzeit bestimmt worden, ab der sich das Modellverhalten nicht mehr ändert, kann davon ausgegangen werden, dass die stationäre Phase erreicht ist. Ab diesem Zeitpunkt können alle Daten benutzt werden, um das “typische” Sys-

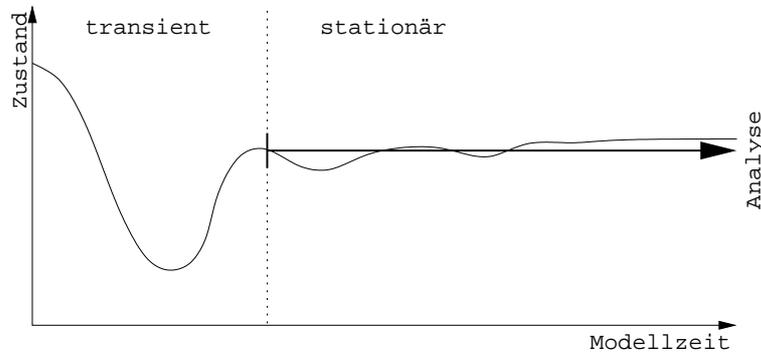


Abbildung 6.11: Die Auswertung der stationären Phase einer langen Simulation.

temverhalten zu analysieren. Leider sind die Daten der Simulation untereinander abhängig. Es ist zum Beispiel sehr wahrscheinlich, dass eine lange Warteschlange an einem M/M/1-Server nach einem Schritt in der Modellzeit immer noch eine lange Warteschlange ist. Aus diesem Grund können die Standard-Schätzer der Statistik (siehe Kapitel 3.3) nicht angewendet werden, da sie auf der Unabhängigkeit der beobachteten Größe basieren. Die folgenden Abschnitte beschreiben Verfahren, die trotz der Abhängigkeit der Daten verlässliche Schätzwerte liefern.

Batch Means

Batch Means ist wohl das bekannteste Verfahren, um abhängige Messreihen zu analysieren. Batch Means versucht die Messreihe so zu transformieren, dass sich Daten ergeben, die unabhängig sind. Auf diese unabhängigen Daten lassen sich dann die Standard-Schätzer anwenden.

Sei Y_1, Y_2, \dots, Y_n ein stochastischer Prozess, der die Messreihe einer Simulation darstellt. Dieser stochastische Prozess wird in m Gruppen der Größe k unterteilt, so dass $n = m \cdot k$ gilt. Y_1 bis Y_k gehören zur ersten Gruppe, Y_{k+1} bis Y_{2k} gehören zur zweiten Gruppe, und so weiter.

$$\bar{Y}_j(k) = \frac{\sum_{i=1}^k Y_{j+i}}{k} \quad \text{mit } j = 1, 2, \dots, m \quad (6.22)$$

sei der Mittelwert der einzelnen Gruppen. Wenn k groß genug gewählt wurde, kann davon ausgegangen werden, dass die Folge der Mittelwerte $\bar{Y}_j(k)$ unabhängig ist. Nun können die Standard-Schätzer auf diese Folge angewendet werden, um zum Beispiel den Gesamtmittelwert und sein Konfidenzintervall zu bestimmen.

Problematisch ist jedoch noch die Wahl der Gruppengröße k . In der Literatur gibt es viele "Faustregeln", die jedoch von Modell zu Modell unterschiedlich sein können. Daher ist es sicherer, aber leider auch zeitaufwendiger, die Intervallgröße k iterativ zu bestimmen.

In der Statistik existieren Testverfahren, die prüfen, ob eine gegebene Stichprobe unabhängig ist. Diese Tests werden unter dem Namen *Run-Test* zusammengefasst (siehe Kapitel 3.7). Einige Run-Tests basieren darauf, wieviele Messwerte hintereinander größer (bzw. kleiner) als der Mittelwert sind. Andere Run-Tests basieren darauf, wieviele Messwerte hintereinander monoton fallend (bzw. steigend) sind. Davon abgesehen können alle Run-Tests eine Aussage darüber treffen, ob die gegebene Messreihe unabhängig ist.

Bei der iterativen Bestimmung einer sinnvollen Gruppengröße k wird mit dem trivialen Fall $k = 1$ gestartet. Die Gruppenmittelwerte werden bestimmt. Auf die Folge der Gruppenmittelwerte wird ein Run-Test angewendet. Wenn der Run-Test bezeugt, dass diese Folge unabhängig ist, wurde k korrekt bestimmt. Wenn der Run-Test jedoch das Gegenteil bezeugt, wird k erhöht (meistens um eins), und die Gruppenmittelwerte werden neu bestimmt.

Overlapping Batch Means

Durch die Verwendung der Gruppenmittelwerte bei der Batch Means Methode entsteht eine erhebliche Datenreduzierung. Je größer die Gruppen, umso weniger Daten bleiben zur Bestimmung des Gesamt-Mittelwertes und des Konfidenzintervalls.

Overlapping Batch Means ist eine Variation von Batch Means, welche diese Schwierigkeit berücksichtigt. Es werden weiterhin Gruppen der Größe k gebildet, allerdings sind die Gruppen nicht disjunkt. Dies bedeutet, dass sich ein Messwert in mehreren Gruppen befinden kann. Anschaulich gesehen, überlappen sich die Gruppen um einige Messwerte. Dadurch erhöht sich die Anzahl der Gruppen sowie die Anzahl der Gruppenmittelwerte. Der Nachteil ist allerdings offensichtlich: Dadurch, dass ein Messwert für die Berechnung mehrerer Gruppenmittelwerte benutzt wird, erhöht sich die Abhängigkeit zwischen den Gruppenmittelwerten. Dies steht im Gegensatz zum ursprünglichen Ziel, der Erzeugung von unabhängigen Daten.

Die Vorteile und Nachteile dieses Verfahrens müssen sorgfältig gegeneinander abgewogen werden, dann ist dieses Verfahren in einigen Fällen sicherlich sinnvoll.

Spaced Batch Means

Spaced Batch Means ist das Gegenteil zu *Overlapping Batch Means*. Bei diesem Ansatz werden Lücken zwischen den einzelnen Gruppen gelassen. Mit Lücken ist hier eine konstante Anzahl von Daten gemeint, die keiner Gruppe zugeordnet werden, und damit auch nicht zur Berechnung der Gruppenmittelwerte beitragen.

Angenommen, die Beobachtungsgrößen seien untereinander stark abhängig. Die Gruppenmittelwerte von zwei direkt benachbarten Gruppen ohne Lücke besitzen dann ebenfalls eine Abhängigkeit, da die Werte am rechten Rand der einen Gruppe von den Werten am linken Rand der anderen Gruppe abhängen. Aus diesem Grund ist es sinnvoll, zwischen den Gruppen Lücken zu lassen. Dies ist allerdings ein verschwenderisches Vorgehen, da die Daten der Lücken unberücksichtigt bleiben.

Ein Extremfall der *Spaced Batch Means* Methode ist, wenn die Gruppengröße $k = 1$ gewählt wird. In diesem Fall wird direkt auf die Originaldaten zurückgegriffen. Es werden aber so viele Daten verworfen, bis keine Abhängigkeit mehr besteht.

Weighted Batch Means

Bei der Erklärung zu *Spaced Batch Means* wurde schon auf das Problem aufmerksam gemacht, dass durch die Randbereiche der Gruppen Abhängigkeiten zwischen den Gruppenmittelwerten entstehen können. Um diesen Abhängigkeiten aus dem Weg zu gehen, können an die einzelnen Messwerte einer Gruppe Gewichtungen vergeben werden. Die Randbereiche der Gruppe werden gegenüber dem Zentrum der Gruppe benachteiligt.

Hierdurch wird der Einfluss der Randbereiche auf den Gruppenmittelwert gemindert, und damit ebenfalls die Abhängigkeit. Es müssen keine Daten unbenutzt verworfen werden. Es besteht aber die Gefahr, dass die Gruppenmittelwerte gegenüber den originalen Daten verfälscht werden. So entsteht eine neue Unsicherheit, die unter gewissen Umständen durch größere Unabhängigkeit gerechtfertigt werden kann.

Regenerative

Die Methode "*Regenerative*" geht davon aus, dass es Systemzustände gibt, bei denen das zukünftige Verhalten des Systems nicht von vorherigen Zuständen abhängt. Das System hat sich also regeneriert. Diese Zustände werden *Regenerationszustände* genannt. Der Zeitpunkt dieser *Regenerationszustände* hat die Bezeichnung *Regenerationspunkt*. Der Zeitraum von einem *Regenerationspunkt*

bis zum nächsten Regenerationspunkt ist der Regenerationszyklus. In den Warteschlangennetzen wird häufig der Systemzustand als Regenerationszustand gewählt, an dem sich kein Kunde mehr im Warteschlangennetz befindet.

Die Anzahl der Regenerationszyklen sei m . Die Länge des j ten Regenerationszyklus sei n_j und z_j sei die Summe aller Messwerte innerhalb dieses Regenerationszyklus. $\bar{z}(m)$ und $\bar{n}(m)$ sind deren Mittelwerte. Alle n_j und z_j seien Realisierungen der Zufallsvariablen Z und N . Der Mittelwert des "typischen" Systemverhaltens kann in diesem Fall durch

$$\mu = \frac{E[Z]}{E[N]} \quad (6.23)$$

beschrieben werden. Leider ist

$$\hat{\mu}(m) = \frac{\bar{z}(m)}{\bar{n}(m)} \quad (6.24)$$

kein unproblematischer Schätzer für μ . Daher kann das Konfidenzintervall nicht durch die Standard-Schätzer bestimmt werden. Die Abweichung des j ten Regenerationszyklus sei $a_j = z_j - \mu n_j$. Alle a_j sind unabhängig und identisch verteilt mit dem Mittelwert 0 und der Varianz σ_a^2 . Mit Hilfe der Varianz σ_a^2 ergibt sich das $(1 - \alpha)$ -Konfidenzintervall

$$\hat{\mu}(m) \pm \frac{z_{1-\frac{\alpha}{2}} \sqrt{\frac{\hat{\sigma}_a^2(m)}{m}}}{\bar{n}(m)} \quad (6.25)$$

$z_{1-\frac{\alpha}{2}}$ ist das $(1 - \frac{\alpha}{2})$ -Quantil der Standardnormalverteilung. Die Anzahl m der Regenerationszyklen muss genügend groß sein, damit das Konfidenzintervall gültig ist. Die genaue Herleitung dieses Konfidenzintervalls und die Berechnung der Varianz σ_a^2 ist in [27] Kapitel 9.5.3 beschrieben.

Die Anforderung der Regenerativ-Methode nach dem Regenerationszustand kann in der Praxis nicht immer erfüllt werden. Daher kann die Regenerativ-Methode nicht auf alle Systeme angewandt werden. Selbst bei Systemen, die einen Regenerationszustand besitzen, kann der Regenerationszyklus unter Umständen so lange dauern, dass nicht genügend Messwerte gesammelt werden können.

Autoregressive

Bei der Methode "Autoregressive" wird davon ausgegangen, dass die beobachtete abhängige Messreihe x_1, x_2, \dots, x_n durch ein autoregressives Modell in eine unabhängige Sequenz von Werten $y_{p+1}, y_{p+2}, \dots, y_{p+n}$ umgewandelt werden kann. Die Messwerte x_i seien Realisierungen der Zufallsvariable X . Der durch x_1, x_2, \dots, x_n gegebene stochastische Prozess muss schwach stationär (siehe Kapitel 3.4) sein, damit eine Analyse mit dieser Methode durchgeführt werden kann. Da zu

diesem Zeitpunkt das Ende der transienten Phase bereits bestimmt wurde, kann diese Annahme vorausgesetzt werden.

Das autoregressive Modell hat die Form:

$$y_i = \sum_{k=0}^p c_k (x_{i-k} - E[X]) \quad \text{für } i = p+1, p+2, \dots, n \quad . \quad (6.26)$$

Zunächst muss nun die Ordnung p und die zugehörigen Gewichtungen c_1, c_2, \dots, c_p bestimmt werden. Durch die Ordnung und die Gewichtungen ergibt sich die unabhängige Sequenz der y_i Werte. Mit deren Hilfe kann die Varianz $Var[\bar{X}(n)]$ errechnet werden. Das $(1-\alpha)$ -Konfidenzintervall für den Erwartungswert besitzt dann die Form

$$\bar{X}(n) \pm t_{\kappa, 1-\frac{\alpha}{2}} \sqrt{Var[\bar{X}(n)]} \quad . \quad (6.27)$$

$t_{\kappa, \gamma}$ sei der kritische Punkt der t -Verteilung mit κ Freiheitsgraden.

Eine detailliertere Beschreibung und weiterführende Literatur wird in [30] vorgestellt. Nicht jede Messreihe lässt sich durch das autoregressive Modell in eine unabhängige Sequenz umwandeln, daher kann mit dieser Methode nicht jedes Modell analysiert werden.

Spektral Analyse

Die Voraussetzung der *Spektral Analyse* ist, dass ein schwach stationäres Modell vorliegt. Durch die Anwendung der Methoden und Regeln aus Kapitel 6.2.1, kann die Voraussetzung angenommen werden.

Dieses Verfahren ist an die physikalische Spektral Analyse angelehnt und trägt aus diesem Grund den gleichen Namen. Die beobachtete Messreihe sei in dieser Sichtweise eine Schwingung, die sich aus der Addition vieler Schwingungen mit unterschiedlicher Frequenz ergibt. Das Spektrum ergibt sich aus

$$g(\tau) = (2\pi)^{-1} \sum_{j=-\infty}^{\infty} c_\tau e^{-ij\tau} \quad \text{für } |\tau| \leq \pi \quad . \quad (6.28)$$

Die Frequenz im physikalischen Modell entspricht hier dem Abstand in der Modellzeit. Diese Größe erhält das Symbol τ . c_τ ist die Autokovarianzfunktion (siehe Kapitel 3.3.3) und i ist die imaginäre Einheit. Der Einfluss der Abhängigkeit innerhalb der Messreihe tritt besonders deutlich bei kurzen Modellzeitabständen auf. Da niedrige Frequenzen einem kurzen Abstand in der Modellzeit entsprechen, ist dieser Frequenzbereich von besonderem Interesse. Für große Stichproben lässt sich zeigen, dass für die Varianz des Mittelwertes der Messreihe

$$Var[\bar{X}(m)] \approx \frac{2\pi g(0)}{m} \quad \text{gilt.} \quad (6.29)$$

Könnte die Varianz $Var[\bar{X}(m)]$ approximiert werden, so kann das $(1 - \alpha)$ -Konfidenzintervall durch

$$\bar{X}(m) \pm t_{f, 1-\frac{\alpha}{2}} \sqrt{Var[\bar{X}(m)]} \quad \text{bestimmt werden.} \quad (6.30)$$

$t_{f, \gamma}$ sei der kritische Punkt der t -Verteilung mit f Freiheitsgraden.

Diese Methode wird in der Praxis selten angewendet, da sie relativ komplex ist. Es gibt viele Schätzer für die Varianz $Var[\bar{X}(m)]$, von denen sich noch keiner als besonders geeignet erwiesen hat.

Standardized Time Series

Die Analyse durch *Standardized Time Series* versucht das gegebene Problem auf besser bekannte Standardfälle zu reduzieren. Schruben (siehe [33]) ist ein Vorreiter dieses Verfahrens, daher ähnelt der hier beschriebene Ansatz den Ansätzen von Schruben aus Kapitel 6.2.1. Die strenge Stationarität der zu analysierenden Modelle ist Voraussetzung.

Die beobachtete Messreihe wird in m Gruppen der Größe k unterteilt. Sei $\bar{X}_j(k)$ der Mittelwert der k ten Gruppe und $\bar{X}(n)$ sei der Gesamtmittelwert. Dann ist

$$A = \left(\frac{12}{k^3 - k} \right) \sum_{j=1}^m \left(\sum_{s=1}^k \sum_{i=1}^s (\bar{X}_j(k) - X_{i+(j-1)k}) \right)^2 \quad (6.31)$$

ein Maß für die Varianz des Gesamtmittelwertes $Var[\bar{X}(n)]$. Es lassen sich Aussagen über die Zufallsverteilung von A treffen. Diese Aussagen führen letztlich zu dem $(1 - \alpha)$ -Konfidenzintervall

$$\bar{X}(n) \pm t_{m, 1-\frac{\alpha}{2}} \sqrt{\frac{A}{mn}} \quad (6.32)$$

$t_{m, \gamma}$ sei der kritische Punkt der t -Verteilung mit m Freiheitsgraden.

Bei diesem Verfahren bleibt es dem Benutzer überlassen, die Größe k der Gruppen zu wählen. Ähnlich wie bei dem Batch Mean Verfahren könnte die Wahl der Gruppengröße k durch einen Run-Test bestätigt werden.

Zusammenfassung

Die Analyseverfahren Regenerative und Autoregressive können nicht auf jedes Modell angewendet werden, da sie gewisse Anforderungen an die Modelle stellen. Die Spektral Analyse und die Methode der Standardized Time Series erfordern erhebliche mathematische Voraussetzungen, um die Güte der Verfahren zu garantieren. Sehr intuitiv erscheinen dagegen das Batch Mean Verfahren und seine abgeleiteten Versionen. Auch die potentiellen Schwierigkeiten bei der Wahl

der Größe der Gruppen scheinen lösbar. Aus diesen Gründen werden die Analysen durch Batch Mean und abgeleitete Varianten im Vordergrund dieser Diplomarbeit stehen.

Alle Verfahren setzen voraus, dass die zu analysierenden Modelle schwach stationär oder streng stationär sein müssen. Da das Ziel die Ermittlung des stationären Systemverhaltens ist, erscheint diese Bedingung selbstverständlich. Stationarität kann bei einer Simulation zwar nie garantiert werden, da nur endlich viele Messwerte zur Verfügung stehen, aber in Kapitel 6.2.1 wird eine große Anzahl an Regeln vorgestellt, welche die Eigenschaft Stationarität "greifbar" machen. Die Qualität der Analyse des stationären Systemverhaltens steht daher in direktem Zusammenhang mit der Qualität der Regeln zur Bestimmung des Endes der transienten Phase.

6.3 Viele kurze Simulationen

Ein alternativer Weg der Simulationsdurchführung ist, viele Simulationsläufe zu starten, die untereinander vergleichbar sind. Vergleichbarkeit kann dadurch sichergestellt werden, dass bei jedem Simulationslauf das gleiche Modell mit denselben Parametern benutzt wird. Nun hat man prinzipiell noch die Möglichkeit den Initialisierungszustand zu variieren. Von dieser Möglichkeit wird hier abgesehen, da die transiente Phase bei einem konkreten Initialisierungszustand von Interesse ist. Damit in einer Computer-Simulation unterschiedliche Verläufe der Modelle erzeugt werden können, muss für jeden Simulationslauf ein anderer Startseed des Zufallszahlengenerators gewählt werden. Hierbei ist zu beachten, dass die Simulationsläufe möglichst viele verschiedene Zufallszahlen benutzen, um eine möglichst große Unabhängigkeit zu erreichen. Da die Anzahl der Zufallswerte eines Zufallszahlengenerators begrenzt ist, sollte das Vorgehen verwendet werden, das in 2.3.4 beschrieben wurde: Die Periode der Zufallszahlen wird in gleichgroße Teilbereiche zerlegt. Jedem Simulationsbereich wird ein Teilintervall zugewiesen.

Auch bei mehreren Simulationsläufen lassen sich Regeln und Verfahren formulieren, die auf das Ende der transienten Phase hinweisen.

6.3.1 Erkennung der stationären Phase mit Reduktion

Alle Messwerte x_1, x_2, \dots, x_n eines Simulationslaufes sind Ziehungen aus den Zufallsvariablen X_1, X_2, \dots, X_n . Um den Charakter dieser Zufallsvariablen mehr zu berücksichtigen, können mehrere Simulationsläufe, sogenannte Replikationen, durchgeführt werden. Dadurch ergeben sich mehrere vergleichbare Ziehungen der Zufallsvariablen, welche die Zufallsvariablen besser beschreiben.

Kumulativer-Mittelwert-Regel

Es werden mehrere Simulationsläufe gestartet, die alle gleich initialisiert werden und zum selben Modellzeitpunkt Beobachtungen liefern. In der Praxis bedeutet dies, dass nur der Seed des Zufallszahlengenerators unterschiedlich gewählt wird. Diese Vorbedingung ist wichtig, damit die gesammelten Beobachtungen untereinander vergleichbar sind. Aus allen Beobachtungen, die zum selben Modellzeitpunkt gemacht wurden, wird der Mittelwert berechnet. Die Folge von

$$\begin{array}{ccccccc}
 1 & : & x_{11} & , & x_{12} & , & \dots & , & x_{1n} \\
 2 & : & x_{21} & , & x_{22} & , & \dots & , & x_{2n} \\
 & & \vdots & & \vdots & & & & \vdots \\
 m & : & x_{m1} & , & x_{m2} & , & \dots & , & x_{mn} \\
 & & \downarrow & & \downarrow & & & & \downarrow \\
 & & \bar{x}_1 & & \bar{x}_2 & & & & \bar{x}_n
 \end{array}$$

Abbildung 6.12: Die Berechnung der Mittelwerte einzelner Beobachtungszeitpunkte, bei m Replikationen und n Beobachtungen pro Replikation.

Mittelwerten, die hierdurch entsteht, beschreibt das Verhalten aller Simulationen und nicht nur eines Simulationslaufes. Prinzipiell können auf diese Folge weitere Regeln angewendet werden, die Folge kann aber auch visuell ausgewertet werden (siehe [21]).

Dieses Verfahren ist im eigentlichen Sinne keine Regel zur Bestimmung des Endes der transienten Phase. Diese Regel sollte mehr als eine Reduktion aufgefasst werden, denn die vielen Simulationsläufe werden auf ihren “durchschnittlichen” Simulationslauf reduziert. Es ergibt sich aus den vielen Messreihen eine einzige Messreihe, die als Stellvertreter fungiert. Die Bestimmung des Endes der transienten Phase einer Messreihe ist ein bereits diskutiertes Problem (siehe Kapitel 6.2.1). Es bestehen Parallelen zur Ensemble-Analyse.

Geraden-Regel

Wie in der Kumulativer-Mittelwert-Regel, werden die Mittelwerte aller Beobachtungszeitpunkte gebildet. Nun werden die Beobachtungszeitpunkte in Gruppen der Größe m_0 eingeteilt und deren Mittelwert bestimmt. Dieses Vorgehen wird ebenfalls bei der Methode *Batch Means* (siehe Kapitel 6.2.2) angewendet, jedoch mit einem völlig anderen Ziel. Die transiente Phase endet mit der Gruppe g_* , bzw. nach der Beobachtung $n_0 = g_* m_0$, wenn die nachfolgenden Gruppenmittelwerte durch eine Gerade ohne Steigung approximiert werden können. (Siehe [26])

Auch diese Regel benutzt die Reduktion auf den Fall mit nur einer Messreihe. Der Vorteil dieser Regel basiert lediglich darauf, dass nicht nur ein Simulations-

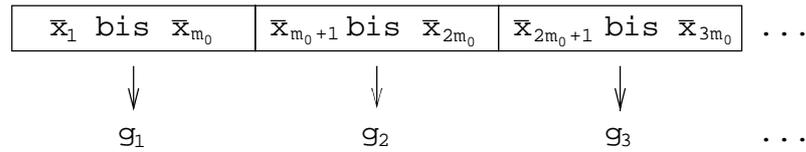


Abbildung 6.13: Die Berechnung der Mittelwerte der einzelnen Gruppen.

lauf analysiert wird, sondern ein durchschnittlicher Simulationslauf. Diese Regel nutzt die Konvergenz des Mittelwertes, was zu einem Problem führen kann, wenn der Mittelwert konvergiert, nicht aber die eigentliche Verteilung.

Welch-Methode

Die Methode von Welch ist in [27] sehr detailliert beschrieben. Sie ist in dieser Auflistung ein Sonderfall, da es sich hierbei um eine Vorgehensweise handelt und nicht um eine Regel. Der Benutzer hat maßgeblichen Einfluss auf das Ergebnis, denn er muss auf Grund von visuellen Eindrücken zum Erkennen der transienten Phase beitragen.

Im Folgenden haben die Variablen und Konstanten diese Bedeutung:

- n sei die Anzahl der Beobachtungen pro Replikation.
- m sei die Anzahl der Replikationen.
- a sei die Größe des Ausschnitts mit $1 \leq a \leq \lfloor \frac{n}{4} \rfloor$.
- n_0 sei das Ende der transienten Phase.
- x_{ji} sei die i te Beobachtung der j ten Replikation.
- $\bar{x}_i = \frac{1}{m} \sum_{j=1}^m x_{ji}$ sei der Durchschnitt der i ten Beobachtungen.
- Der Mittelwert eines Ausschnittes sei:

$$\bar{x}_i(a) = \begin{cases} \frac{\sum_{s=-a}^a \bar{x}_{i+s}}{2a+1} & \text{falls } i = a+1, \dots, n-a \\ \frac{\sum_{s=-(i-1)}^a \bar{x}_{i+s}}{2i-1} & \text{falls } i = 1, \dots, a \end{cases} \quad (6.33)$$

Die Methode basiert darauf, dass im Bedarfsfall weitere Replikationen durchgeführt werden können. So wird mit fünf bis zehn Replikationen gestartet. Die Simulationslänge n sollte viel größer sein als die erwartete Länge der transienten Phase n_0 . Außerdem sollten die längsten Zyklen des Modells mehrfach durchlaufen werden können.

Eine Sequenz $\bar{x}_i(a)$ wird bestimmt unter den Kriterien, dass a möglichst klein ist, die Sequenz aber möglichst glatt (visueller Eindruck) wirkt. Sollte kein Wert

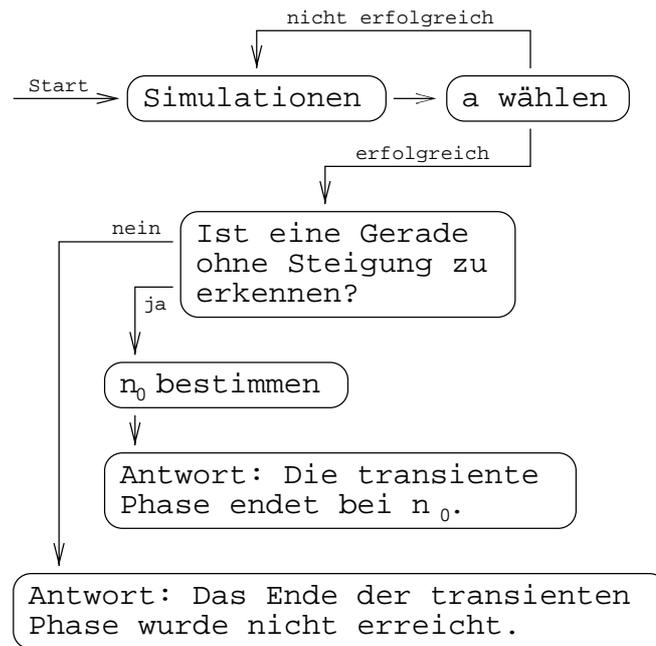


Abbildung 6.14: Verlaufsdiagramm der Welch-Methode.

für a eine zufriedenstellende Sequenz liefern, müssen fünf bis zehn weitere Replikationen durchgeführt werden. Sollte jedoch eine glatte Sequenz $\bar{x}_i(a)$ gefunden werden, wird überprüft, ob die Sequenz ab einem Zeitpunkt annähernd eine Gerade ohne Steigung ist. Wenn nicht, ist entweder das Ende der transienten Phase nicht erreicht worden, oder es gibt keine stationäre Phase. Wenn doch, kann das Ende n_0 der transienten Phase bestimmt werden.

Auch die Methode von Welch benutzt eine Reduktion von vielen Messreihen auf eine typische Messreihe. Ein weitere Schwierigkeit ist, dass die Länge der Simulationen vor dem Simulationsstart festgelegt werden muss. Hat der Benutzer kein Vorwissen über das zu analysierende Modell, ist dieses Verfahren nur schlecht anwendbar.

Zusammenfassung

Da die drei beschriebenen Regeln und Verfahren immer eine Reduktion durchführen, ergeben sich für sie ähnliche Probleme wie für die Regeln, die auf einem Simulationslauf basieren. Diese Regeln sind somit keine große Erweiterung. Interessant wären Verfahren, die direkt auf mehreren Simulationsläufen arbeiten. Sie könnten berücksichtigen, dass alle Simulationsdaten, die zu einem bestimmten Modellzeitpunkt beobachtet werden, die Realisierung derselben Zufallsvariablen sind.

6.3.2 Erkennung der stationären Phase ohne Reduktion

Bei der Durchführung von mehreren Simulationsläufen ergeben sich für einen Modellzeitpunkt mehrere Beobachtungsdaten. Wenn bei jeder Simulation nur der Startseed des Zufallszahlengenerators geändert wird, nicht aber das Modell selber, ist die Annahme gerechtfertigt, dass diese Beobachtungsdaten Stichproben aus der gleichen Zufallsvariablen sind. Das Ziel ist, zu erkennen, ab welcher Mo-

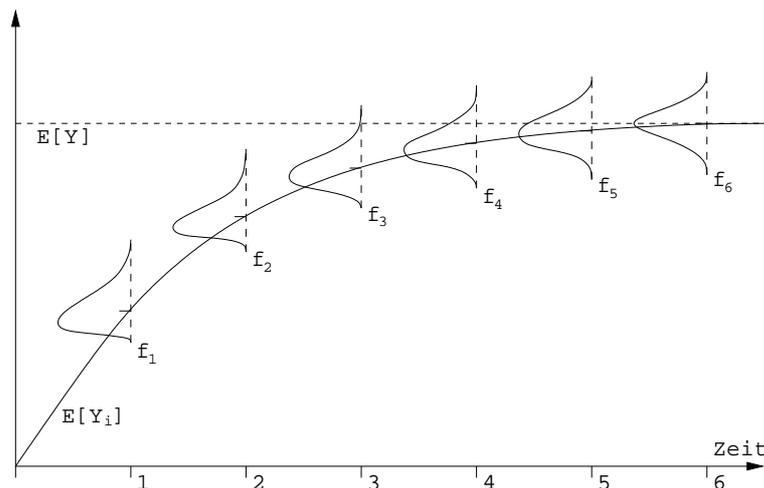


Abbildung 6.15: Die Stichproben einer bestimmten Modellzeit haben eine Zufallsverteilung.

dellzeit die Zufallsvariablen der Beobachtungsdaten identisch verteilt sind. Ab diesem Modellzeitpunkt liegt Stationarität vor (siehe Kapitel 3.4) und die Daten sind für eine Analyse des stationären Systemverhaltens geeignet. Da aber nicht die Zufallsvariablen bekannt sind, sondern nur eine Stichprobe jeder Zufallsvariablen, sollte das Ziel lauten: Gesucht ist die Modellzeit, ab der die Stichproben der Modellzeitpunkte keine signifikanten Unterschiede mehr besitzen.

Um diesem Ziel näher zu kommen, wurden im Rahmen dieser Diplomarbeit zwei Verfahren entwickelt. Das eine Verfahren basiert auf einer visuellen Darstellung, an der ein Benutzer das stabile Systemverhalten erkennen kann. Das andere Verfahren versucht das Problem mathematisch zu lösen.

Zufallsverteilungsdiagramme

Es gibt mehrere Möglichkeiten, ausgehend von einer Stichprobe, die Zufallsverteilung einer Zufallsvariablen darzustellen. Um eine eventuelle stationäre Phase zu erkennen, müssen allerdings sehr viele Zufallsverteilungen visualisiert werden. Die Visualisierung muss in einer Form erfolgen, die einen direkten Vergleich der Zufallsverteilungen untereinander ermöglicht.

Zufallsverteilungsdiagramme aus Box-Plots: Eine Möglichkeit der Darstellung einer Zufallsverteilung ist der Box-Plot (siehe Kapitel 3.3.7). Er basiert auf der Berechnung einiger Quantilen. Bildet der Box-Plot die grundlegende Darstellung,

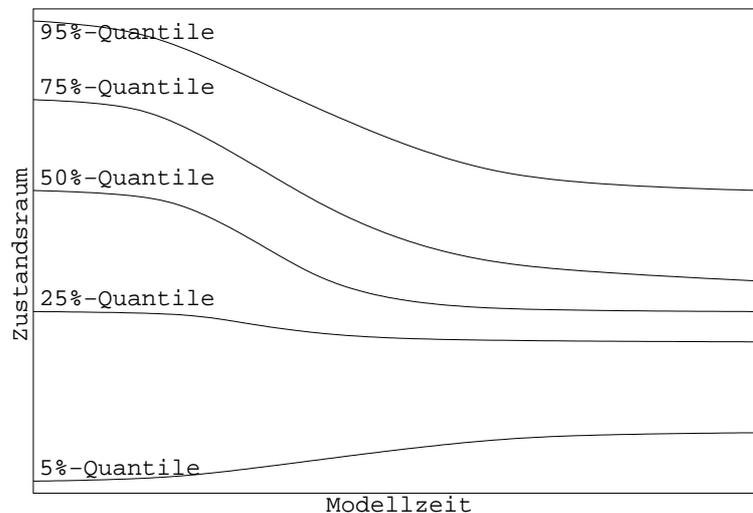


Abbildung 6.16: Ein Zufallsverteilungsdiagramm basierend auf vielen vertikalen Box-Plots.

so könnte eine horizontale Aneinanderreihung von vielen vertikalen Box-Plots eine stationäre Phase erkennbar machen. Auf der horizontalen Achse sei dann die Modellzeit aufgetragen. Natürlich sollten diese Box-Plots nicht streng nach der Beschreibung in Kapitel 3.3.7 gezeichnet werden. Eine farbliche Darstellung der Quantilen wäre angebrachter. So entstehen horizontale Linien für die einzelnen Quantilen. Wenn alle diese Linien ab einer bestimmten Modellzeit durch eine Gerade ohne Steigung approximierbar sind, dann sind die Zufallsverteilungen konstant und die stationäre Phase ist erkannt. Dieses Verfahren bietet sich an, wenn die Stichproben relativ klein sind.

Zufallsverteilungsdiagramme aus Histogrammen: Die zweite Möglichkeit der Darstellung einer Zufallsverteilung ist das Histogramm. Um ein Histogramm zu berechnen, wird der Wertebereich der beobachteten Größe in gleichgroße Intervalle aufgeteilt. Nun wird für jedes Intervall bestimmt, wieviele der beobachteten Werte es enthält. Ein Säulendiagramm fasst die Ergebnisse zusammen. Jedem Intervall ist eine Säule zugeordnet. Die Höhe der Säulen ist durch die Beobachtungen pro Intervall definiert.

Liegen größere Stichproben vor, verspricht die Histogrammdarstellung eindrucksvollere Abbildungen. Wenn in einer 3-dimensionalen Darstellung mehrere Säulendiagramme hintereinander gesetzt werden, entsteht eine Art "Geländeprofil". Sollte dieses Geländeprofil mit fortschreitender Modellzeit konstant blei-

ben, sind die Zufallsverteilungen konstant und die stationäre Phase ist erkannt. Eine 3-dimensionale Darstellung ist aus vielen Gründen jedoch unhandlich. Daher ist es wünschenswert, die 3-dimensionale Darstellung in eine 2-dimensionale Darstellung umzuwandeln. Dies kann so erreicht werden:

- Auf der horizontalen Achse ist die Modellzeit aufgetragen. Da nur diskrete Beobachtungsschritte in der Modellzeit gemacht werden, ist auch die Einteilung der horizontalen Achse diskret. Voraussetzung ist, dass alle Simulationen dieselbe Beobachtungsschrittweite haben. Dies kann durch das Verfahren der disjunkten Intervalle gewährleistet werden (siehe Kapitel 4.1.1).
- Die vertikale Achse der Darstellung beschreibt den Wertebereich der beobachteten Größe. Durch die Einteilung in Intervalle bei der Histogrammberechnung ist auch die Einteilung der vertikalen Achse diskret.
- Da beide Achsen diskrete Bereiche haben, entsteht eine Fläche aus Quadraten (bzw. Rechtecken). Jedes Quadrat entspricht einer Säule. Die Höhe dieser Säule wird durch eine graue Farbabstufung dargestellt. Je höher die ursprüngliche Säule, umso dunkler sei der verwendete Grauton.



Abbildung 6.17: Ein 2-dimensionales Zufallsverteilungsdiagramm basierend auf vielen vertikalen Histogrammen.

Um Darstellungen zu erhalten, die untereinander vergleichbar sind, ist es sinnvoll die Grauwertskala so zu wählen, dass sie in verschiedenen Diagrammen gleich ist. Aus diesem Grund sollten die Grauwerte nicht aus der konkreten Anzahl von Beobachtungen pro Zeitintervall abgeleitet werden, sondern von ihrer relativen Häufigkeit. Dies stellt sicher, dass auch Darstellungen mit unterschiedlicher Stichprobengröße vergleichbar sind. Die allgemeine Idee zu dieser Darstellung ist angelehnt an die übliche Darstellung für chaotische Iterationsgleichungen (siehe Kapitel 5).

Die Abbildung 6.17 zeigt ein 2-dimensionales Zufallsverteilungsdiagramm basierend auf vielen vertikalen Histogrammen. Auf der vertikalen Achse ist der Zustandsbereich aufgetragen. Auf der horizontalen Achse ist die Modellzeit aufgetragen. Die Grauwerte repräsentieren die relative Häufigkeit. Je dunkler der Grauwert, umso größer ist die relative Häufigkeit. Die hellblauen und die dunkelblauen Bereiche liegen außerhalb des Darstellungsbereichs der relativen Häu-

figkeit. Das Begrenzen des Darstellungsbereichs der relativen Häufigkeit ist sinnvoll, damit spezielle Bereiche genauer dargestellt werden können, da nur begrenzt viele Grauwerte zur Verfügung stehen. In diesem Fall besteht die Darstellung aus 256 verschiedenen Farben und Grauwerten.

Eine stationäre Phase liegt dann vor, wenn die Darstellung ab einer bestimmten Modellzeit keine signifikanten Änderungen mehr aufzeigt. Hierbei ist es wichtig, dass die Stichproben der einzelnen Modellzeiten genügend groß sind. Sonst könnte die natürliche Varianz der Zufallsvariablen für relativ große Schwankungen verantwortlich sein, sodass ein eventueller Einfluss des Initialisierungszustandes gar nicht erkennbar sein kann. Ein guter Richtwert sind fünf bis zehn Stichprobenwerte pro Intervall, da dieser Richtwert auch für die Intervallbildung für den χ^2 -Test in [27] vorgeschlagen wird.

Zufallsverteilungsmethode

Die Zufallsverteilungsmethode basiert auf derselben Eigenschaft der Zufallsvariablen einer Messreihe, die bei der visuellen Bestimmung der stationären Phase mit Hilfe der Zufallsverteilungsdiagramme ausgenutzt wird. Es wird nach der Modellzeit gesucht, ab der die Stichproben der einzelnen Modellzeiten keine signifikante Änderung mehr aufweisen. An den Zufallsverteilungsdiagrammen konnte diese Eigenschaft intuitiv erkannt werden. Bei der Zufallsverteilungsmethode muss diese Eigenschaft jedoch mathematisch erkannt werden.

Das Problem der Erkennung der stationären Phase lässt sich mathematisch ausdrücken: Sind alle Zufallsvariablen Y_i einer Messreihe ab einer bestimmten Zufallsvariablen Y_{n_0} identisch verteilt, liegt die stationäre Phase vor. Daher muss überprüft werden, ob die Stichproben zweier Modellzeitpunkte aus derselben Zufallsverteilung stammen. Dieses Problem ist ein Standardproblem der Statistik. Eine ganze Reihe von Testverfahren können hier verwendet werden:

- χ^2 -Test (siehe Kapitel 3.6 und [27])
- Kolmogoroff-Smirnoff-Test (siehe Kapitel 3.6 und [27])
- Mann-Whitney U-Test (siehe Kapitel 3.6 und [8])
- Wilcoxon Signed-Rank-Test (siehe Kapitel 3.6 und [8])
- usw.

Die geeigneten mathematischen Verfahren sind somit vorhanden. Nun muss ein Algorithmus entwickelt werden, in dem diese mathematischen Verfahren effizient angewendet werden können.

Der Proband sei die Stichprobe einer Zufallsvariablen Y_i . Im Laufe des Algorithmus wird der Proband durch immer andere Stichproben definiert werden.

Schrittweise wird er die Zufallsvariablen Y_{i+1} annehmen und dadurch in der Modellzeit immer weiter nach "hinten" verschoben. Am Ende des Algorithmus soll der Proband die Stichprobe der ersten Modellzeit sein, die sich innerhalb der stationären Phase befindet.

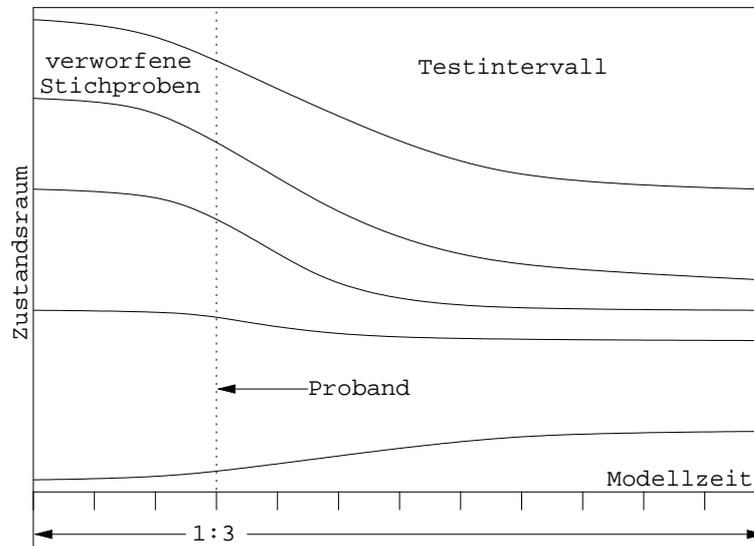


Abbildung 6.18: Schematische Darstellung der Zufallsverteilungsmethode.

Es wird ein Zeitintervall benötigt, das sich direkt an die Modellzeit des Probanden anschließt. Alle Stichproben dieses Testintervalls werden mit dem Probanden verglichen. Wenn weniger als zum Beispiel 5% der Stichproben dieses Intervalls nicht dieselbe Zufallsverteilung wie der Proband besitzen, kann angenommen werden, dass die stationäre Phase erreicht ist. Wird diese Prozent-schranke nicht erreicht, so kann angenommen werden, dass der Proband nicht zur stationären Phase gehört. Die Stichprobe kann verworfen werden und der Proband sei nun die nachfolgende Stichprobe. In diesem Fall muss auch das betrachtete Zeitintervall verschoben werden.

Bei der Wahl der Größe des Testintervalls können zwei Ansätze verfolgt werden. Entweder die Größe des Intervalls ist konstant, oder die Größe des Intervalls steht im Verhältnis zu den bisher verworfenen Probanden. Der zweite Ansatz hat den Vorteil, dass die Anzahl der betrachteten Daten immer in einem bestimmten Verhältnis zu der transienten Phase steht. Je länger die transiente Phase, umso länger das Zeitintervall, das auf Stationarität geprüft wird. Im Folgenden wird nur der zweite Ansatz verfolgt.

Der Algorithmus der Zufallsverteilungsmethode hat folgende Form:

1. Der Benutzer legt das Verhältnis zwischen der Länge der transienten Phase und der Länge der (betrachteten) stationären Phase auf "1 zu $y - 1$ " fest.
2. Der Proband sei die Stichprobe der Modellzeit Null.

3. Es werden y Stichproben gesammelt.
4. Test auf identische Zufallsverteilung des Probanden mit allen folgenden Stichproben. Fall 1: Wenn mehr als $x\%$ der Stichproben nicht die Zufallsverteilung des Probanden besitzen, sei der Proband die nächstfolgende Stichprobe und y neue Daten müssen gesammelt werden: Weiter mit Schritt 3. Fall 2: Wenn weniger als $x\%$ der Stichproben nicht die Zufallsverteilung des Probanden besitzen, sei die Modellzeit des Probanden der Beginn der stationären Phase. Alle Stichproben, die als Probanden verworfen wurden, bilden die transiente Phase. Die stationäre Phase wird durch die nachfolgenden Stichproben des Probanden gebildet.

Während der Proband immer um einen Schritt in der Modellzeit verschoben wird, werden dem zu vergleichenden Testintervall immer y Daten hinzugefügt. Daher wird die Laufzeit der Iterationen des Algorithmus immer länger werden. Der Vorteil ist jedoch, dass "vernünftige" Längenverhältnisse bestehen bleiben. Je länger die transiente Phase ist, umso mehr Daten der stationären Phase werden betrachtet. Dies stellt sicher, dass die stationäre Phase korrekt erkannt wird.

6.3.3 Auswertung der stationären Phase

Bei der Durchführung von mehreren Simulationen ergibt sich eine völlig neue Ausgangssituation. Während bei einer einzelnen Simulation für einen Modell-

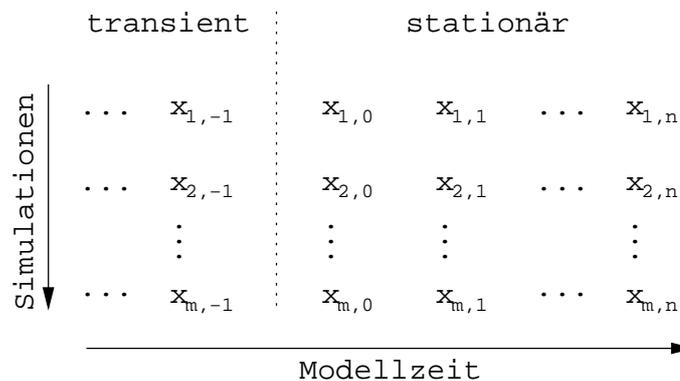


Abbildung 6.19: Die Stichprobenmenge bei mehreren Simulationen.

zeitpunkt nur ein Messwert vorliegt, ergeben sich bei mehreren Simulationen viele Messwerte pro Modellzeitpunkt. $x_{i,j}$ sei der beobachtete Wert der i ten Simulation (Replikation) zum j ten Beobachtungsschritt nach Beginn der stationären Phase. n sei die Anzahl der Beobachtungen der stationären Phase einer Simulation und m sei die Anzahl der Simulationen. $x_{i,0}$ ist also die erste Beobachtung innerhalb der stationären Phase der i ten Simulation.

Die in Kapitel 6.2.2 vorgestellten Analyseverfahren lassen sich nicht direkt anwenden, da sie für die Analyse einer einzigen Messreihe entwickelt wurden. Nun können zwei verschiedene Strategien verfolgt werden. Bei der einen Strategie wird versucht, viele Messreihen auf eine zu reduzieren, bei der anderen Strategie wird versucht, andere Analyseverfahren für die neue Ausgangssituation zu benutzen.

Reduktion auf eine Messreihe

Die Reduktion von mehreren Messreihen auf eine Messreihe ist denkbar einfach. Die neue Messreihe bestehe aus den Mittelwerten der Stichproben der einzelnen Modellzeitpunkte.

$$\bar{x}_j = \sum_{i=1}^m \frac{x_{i,j}}{m} \quad \text{für } 0 < j \leq n \quad (6.34)$$

Da bei diesem Vorgehen die einzelnen Simulationen miteinander vermischt werden und ein durchschnittliche Messreihe entsteht, besteht die Hoffnung, dass die Daten der neuen Messreihe untereinander nicht mehr so stark abhängig sind. Dieser Effekt wird umso stärker, je mehr Simulationen betrachtet werden. Ge-

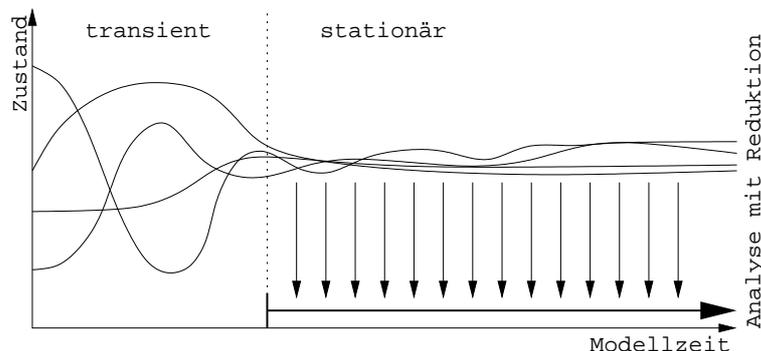


Abbildung 6.20: Die Auswertung der stationären Phase vieler Simulationen mit Reduktion.

nerell muss allerdings noch davon ausgegangen werden, dass eine Abhängigkeit bestehen bleibt. Aus diesem Grund bringt die Auswertung mit mehreren Simulationsläufen bei einer Reduktion auf eine durchschnittliche Messreihe keinen grundsätzlichen Vorteil. Die neue Messreihe ist weiterhin abhängig und muss in eine unabhängige Messreihe transformiert werden. Hierzu können die in Kapitel 6.2.2 beschriebenen Methoden verwendet werden.

Direkte Analyse

Durch die Regeln und Verfahren aus Kapitel 6.3.1 und 6.3.2 sei der Beginn n_0 der stationären Phase bekannt. n_0 sei der Index der ersten Beobachtung, die sich

innerhalb der stationären Phase befindet. Ab dem zugehörigen Modellzeitpunkt beschreiben alle Daten das stationäre Systemverhalten und sind daher miteinander vergleichbar. Sei \bar{S}_i der Mittelwert aller Daten der i ten Simulation, die sich innerhalb der stationären Phase befinden.

$$\bar{S}_i = \sum_{j=n_0}^n \frac{x_{i,j}}{n - n_0} \quad \text{für } 0 < i \leq m \quad (6.35)$$

Da die Daten einer Simulation unter Umständen stark abhängig sind, ist es bei dieser Methode nicht sinnvoll, die Standardabweichung oder ein Konfidenzintervall für \bar{S}_i zu berechnen. Dieser Mittelwert wird lediglich als ein Punktschätzer verwendet. Da m Simulationen durchgeführt wurden, ergeben sich auch m Mit-

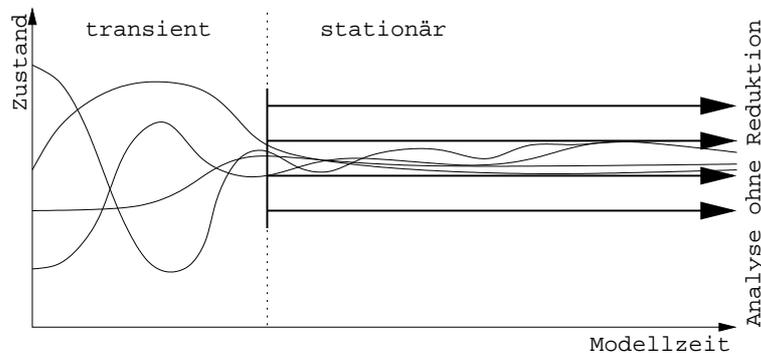


Abbildung 6.21: Die Auswertung der stationären Phase vieler Simulationen ohne Reduktion.

telwerte. Wird davon ausgegangen, dass der verwendete Zufallszahlengenerator für jede Simulation einen “unterschiedlichen” und unabhängigen Zufall generiert, sind alle \bar{S}_i unabhängig und normalverteilt um den Erwartungswert der stationären Zufallsverteilung (siehe [27], Kapitel 9.5.2). Diese Eigenschaften sind die Basis dafür, dass die Standard-Methoden zur Berechnung des Konfidenzintervalles angewendet werden können.

Was ist aber, wenn das Konfidenzintervall im Vergleich mit dem Mittelwert zu groß ist? In diesem Fall könnten zum einen nachträgliche unabhängige Simulationsläufe durchgeführt werden, um durch eine größere Anzahl an Mittelwerten \bar{S}_i den Gesamtmittelwert weiter einzuschränken. In vielen Fällen soll die bestehende Anzahl an Simulationen jedoch ausreichen. In diesem Fall ist es zweckmäßig, die bestehenden Simulationen fortzusetzen, um die Mittelwerte \bar{S}_i näher an den Erwartungswert zu führen. Nach dem Gesetz der großen Zahlen (“strong law of large numbers”: [27], Kapitel 4.6) wird dieses Vorgehen Erfolg haben.

Bewertung

Die direkte Analyse wirkt “natürlicher” als die Reduktion auf die stellvertretende Messreihe und ist einfacher zu realisieren, da die Transformation von einer abhängigen Messreihe auf unabhängige Daten entfällt. Da die Transformation auf unabhängige Daten ein weiterer Unsicherheitsfaktor ist, sind die Ergebnisse der direkten Analyse wohl verlässlicher. Aus diesen und weiteren Gründen wird die direkte Analyse auch in der Literatur bevorzugt (siehe [27], Kapitel 9.5.2).

Kapitel 7

Implementierte Software-Tools

Alle Software-Tools wurde in der Programmiersprache C++ entwickelt und für das Betriebssystem UNIX optimiert. Als Referenzliteratur diene hier [36] und [24]. In den folgenden Abschnitten wird die Funktionsweise der implementierten Software-Tools näher erläutert. Es wird speziell darauf eingegangen, welche Voraussetzungen für eine effiziente Anwendung geschaffen werden müssen. Auf die konkrete Umsetzung wird nur kurz eingegangen. Im Vordergrund steht hier die Erläuterung der Modulstruktur der einzelnen Programme.

Die in diesem Kapitel beschriebenen Software-Tools können alle separat benutzt werden. Für den Benutzer ist es allerdings einfacher das Programm *ssa* ("steady-state" Analyse) zu verwenden. *ssa* ruft alle benötigten Dienstprogramme automatisch in geeigneter Reihenfolge auf und nimmt dem Benutzer hierdurch viel Arbeit ab. Um *ssa* aufzurufen, muss das zu analysierende HISLANG-Modell vorliegen und die in Kapitel 7.1.1 genannten Voraussetzungen müssen erfüllt sein.

7.1 *ssa*: "steady-state" Analyse

Das Programm *ssa* ("steady-state" Analyse) ist ein Software-Tool zur Analyse von Simulationsdaten. Es wurde unter besonderer Berücksichtigung der Probleme entwickelt, die durch einen "untypischen" Initialisierungszustand entstehen. Die Analysealgorithmen sollen die Auswertungsschwierigkeiten umgehen, die in Kapitel 1.1 am Beispiel des Güterverkehrszentrums und in Kapitel 4.4 am Beispiel eines M/M/1-Servers dargestellt werden. Um dieses Ziel zu erreichen, werden einige der in Kapitel 6 vorgestellten Verfahren implementiert und geeignet miteinander kombiniert. Die Ideen zur Betrachtung der Zufallsverteilungen, die in Kapitel 6.3.2 geschildert wurden, werden weiter verfolgt.

Bei der Auswahl der implementierten Analysemethoden wurde Wert darauf gelegt, dass sowohl ein langer Simulationslauf, als auch mehrere kurze Simula-

tionsläufe analysiert werden können. Aus jeder Strukturgruppe der Analysemethoden (siehe Struktur des Kapitels 6) wird mindestens ein Vertreter implementiert.

7.1.1 Voraussetzungen

Das Programm *ssa* ist ein komplexes Software-Tool, da es parallele Simulationen durchführen kann und auf anderen Programmen basiert. Aus diesem Grund müssen einige Voraussetzungen geschaffen werden, damit ein effizienter Programmablauf gewährleistet ist.

Um *ssa* mitzuteilen, welche Rechner für die parallelen Simulationen zur Verfügung stehen, muss eine Datei mit diesen Informationen angelegt werden. Eine Zeile dieser Konstellationsdatei besteht aus drei Einträgen, die durch ein Leerzeichen voneinander getrennt sind. Der erste Eintrag ist der Rechnername. Dieser Name muss so gewählt sein, dass mit dem Standard-Tool "rsh" auf ihn zugegriffen werden kann. Der zweite Eintrag steht für die Geschwindigkeit des Rechners. Hier sollte die Taktfrequenz in MHz angegeben werden. Prinzipiell können hier auch beliebige andere Werte angegeben werden, entscheidend ist nur, dass das Verhältnis dieser Werte die Geschwindigkeitsunterschiede deutlich macht. Der dritte Eintrag steht für die Menge an Hauptspeicher in MB, die für die Simulationen auf dem Rechner zur Verfügung stehen soll. Als gutes Maß hat sich die Hälfte des gesamten Hauptspeichers erwiesen, denn es muss eingeplant werden, dass andere Benutzer den Rechner ebenfalls benutzen. Hier das Beispiel einer Konstellationsdatei, welche die Rechnerkonstellation beschreibt:

```
bruno 800 1024
gerd 400 64
christel 400 64
roland 333 64
astrid 300 64
bernd 143 64
elke 143 64
```

Der Zugriff auf die Dienstprogramme *steady*, *hit2steady* und *steady_hit* muss *ssa* ermöglicht werden. Auch dies geschieht über eine Datei. Diese Steuerdatei hat folgendes Format: In der ersten Zeile wird der Pfad und die Datei angegeben, welche die Rechnerkonstellation beinhaltet. In der zweiten, dritten und vierten Zeile wird die Position der Dienstprogramme *steady*, *hit2steady* und *steady_hit* angegeben. Das Beispiel einer Steuerdatei:

```

/app/unido-i04/SFB559_M1/SteadyStateAnalyse/rechner.txt
/app/unido-i04/SFB559_M1/SteadyStateAnalyse/steady
/app/unido-i04/SFB559_M1/SteadyStateAnalyse/hit2steady
/app/unido-i04/SFB559_M1/SteadyStateAnalyse/steady_hit

```

Diese Steuerdatei ermöglicht einen flexiblen Umgang mit den Dienstprogrammen. Das Verschieben der Dienstprogramme oder das Anbinden anderer Dienstprogramme mit ähnlicher Funktion wird hierdurch möglich. Eine einmal erstellte Steuerdatei muss nicht mehr geändert werden, solange keine Systemveränderungen vorliegen. Die Angabe der Rechnerkonstellation in einer gesonderten Datei ermöglicht es, dass jeder Benutzer seine eigene Systemkonfiguration benutzt.

Die externen Tools *xv* (Version 3.10a), *gnuplot* (Unix Version 3.7) und *octave* (Version 2.0.16) werden benötigt. Der Benutzer von *ssa* muss einen automatisierten Zugriff ermöglichen. Bei der Systemkonfiguration am Lehrstuhl IV des Fachbereichs Informatik der Universität Dortmund kann der Zugriff über die Module “extra/xv/3.10a”, “extra/gnuplot/3.7” und “extra/octave/2.0.16” hergestellt werden.

Die parallelen Simulationen werden über die “Remote Shell” *rsh* realisiert. Es muss sichergestellt sein, dass der Rechner, auf dem *ssa* gestartet wurde, über *rsh* mit den Rechnern der Konstellationsdatei verbunden werden kann (siehe “man rsh”). Bei jedem internen Aufruf von *rsh* wird die “Loginshell” ausgeführt. Die Befehle der “Loginshell” können sehr zeitaufwendig sein, obwohl ihre Durchführung keine Auswirkungen hat. Daher sollte an einer geeigneten Stelle in der “Loginshell” folgende Zeile eingefügt werden:

```
if ($?prompt == 0) exit
```

Diese Anweisung verhindert die weitere Ausführung der “Loginshell”, wenn durch *rsh* ein Befehl auf einem externen Rechner durchgeführt wird. In der Praxis hat sich gezeigt, dass dieses Vorgehen erhebliche Geschwindigkeitsvorteile mit sich bringt.

Da jede Verbindung über *rsh* zu einem anderen Rechner einen Teil des Hauptspeichers belegt, ist die Anzahl der parallelen Simulationen begrenzt. Erfahrungswerte haben gezeigt, dass 100 parallele Simulationen von einem Rechner mit 256MB Hauptspeicher (zum Beispiel “siegmar”) durchgeführt werden können. Da die graphischen Benutzeroberflächen (zum Beispiel “Solaris”) ebenfalls Hauptspeicher belegen, ist es sinnvoll, aber nicht notwendig, dass der Rechner, von dem die Simulationen gestartet werden, per “Remote” benutzt wird.

Wenn der Rechner, auf dem *ssa* gestartet wird, per “Remote” benutzt wird, ist zu beachten, dass die Bildschirmausgabe umgeleitet werden muss. Die Umleitung kann durch die üblichen Befehle erfolgen. Wenn der Arbeitsplatzrechner

"astrid" heißt, und der "Remote"-Rechner "siegmar", muss auf "astrid" der Befehl

```
xhost +siegmar
```

und danach auf "siegmar" der Befehl

```
setenv DISPLAY astrid:0
```

eingegeben werden.

Eine weitere wichtige Voraussetzung ist, dass das Programm *ssa* in dem Verzeichnis gestartet werden muss, in dem sich die zu analysierende HISLANG-Datei befindet. Da im Laufe der Analyse ein Ergebnisverzeichnis und mehrere temporäre Dateien angelegt werden, ist es sinnvoll, ein gesondertes Arbeitsverzeichnis zu erzeugen. In dieses Arbeitsverzeichnis muss die zu analysierende HISLANG-Datei kopiert werden. Ein Aufruf von *ssa* kann nun innerhalb dieses Arbeitsverzeichnisses erfolgen.

7.1.2 Aufruf und Ablauf

Wenn alle Voraussetzungen beachtet wurden, ist der Aufruf von *ssa* einfach:

```
PFAD/ssa STEUERDATEI
```

Zu Beginn des Programms *ssa* muss der Benutzer einige Experimentparameter festlegen. Als Erstes kann er den Startseed festlegen. Werden parallele Simulationen durchgeführt, ist der Startseed der Beginn der Zufallsperiode. Als Nächstes muss das zu analysierende HISLANG-Modell festgelegt werden. Hierbei ist zu beachten, dass nur die HISLANG-Modelle gewählt werden können, die sich im aktuellen Arbeitsverzeichnis befinden. Ist das HISLANG-Modell gewählt, macht *ssa* automatisch Vorschläge, welche Modellkomponenten betrachtet werden können. Durch "(j)a" und "(n)ein" kann hier eine Auswahl getroffen werden. Als Nächstes legt der Benutzer den Modellzeitabstand fest, an dem Simulationsdaten aus den Simulationsereignissen berechnet werden sollen. Dieser Modellzeitabstand entspricht der Intervallgröße des Filterprogramms *steady*.

Nun müssen die Auswertungsalgorithmen festgelegt werden. Zunächst wird ein Verfahren festgelegt, mit dem der Anfang der stationären Phase bestimmt werden soll. Je nach gewähltem Verfahren, müssen unterschiedliche Auswertungsparameter angegeben werden. Als Nächstes wird festgelegt, mit welchem statistischen Verfahren die stationäre Phase ausgewertet werden soll. Auch hierbei sind für unterschiedliche Verfahren unterschiedliche Auswertungsparameter anzugeben.

Während des Programmablaufes wird im aktuellen Arbeitsverzeichnis ein Ergebnisverzeichnis angelegt. In diesem Verzeichnis werden die Abbildungen gespeichert, die während der Analyse erzeugt werden. Des Weiteren wird der Benutzer über jedes Ergebnis bei jedem durchgeführten Test über eine Bildschirmausgabe informiert. Wird eine stationäre Phase erkannt, wird ihr Beginn (Modellzeit) auf dem Bildschirm ausgegeben und es werden die statistischen Auswertungsmethoden der stationären Phase angewendet. Auch hier wird der Benutzer jedesmal informiert, wenn sich durch weitere Messwerte ein neuer Mittelwert und ein neues Konfidenzintervall ergibt. So steht am Ende der Programmausführung das Analyseergebnis in der Bildschirmausgabe.

Bei der Analyse der gewählten HISLANG-Datei wird die Strategie aus Kapitel 6 verfolgt. Ausgewählte Verfahren wurden implementiert. Bei der Auswahl der Verfahren wurde darauf geachtet, dass mindestens ein Vertreter jeder Gruppe (siehe Struktur Kapitel 6) realisiert wurde.

Die Abbildung 7.1 zeigt das Ablaufdiagramm von *ssa*. Das Diagramm zeigt, an welcher Stelle des Programmablaufs die einzelnen Dienstprogramme automatisch ausgeführt werden. Der Benutzer muss nur das Programm *ssa* aufrufen, alle weiteren Aufrufe werden automatisch durch *ssa* durchgeführt. Des Weiteren ist zu erkennen, in welcher Reihenfolge die Analysemethoden angewendet werden. Die aufgeführten Kapitelnummern verweisen auf die entsprechenden Kapitel in dieser Diplomarbeit. Es ist noch anzumerken, dass zusätzlich zur abschließenden Ergebnisdarstellung auch während der Erkennung und Auswertung der stationären Phase die bisherigen Ergebnisse durch geeignete Diagramme dargestellt werden.

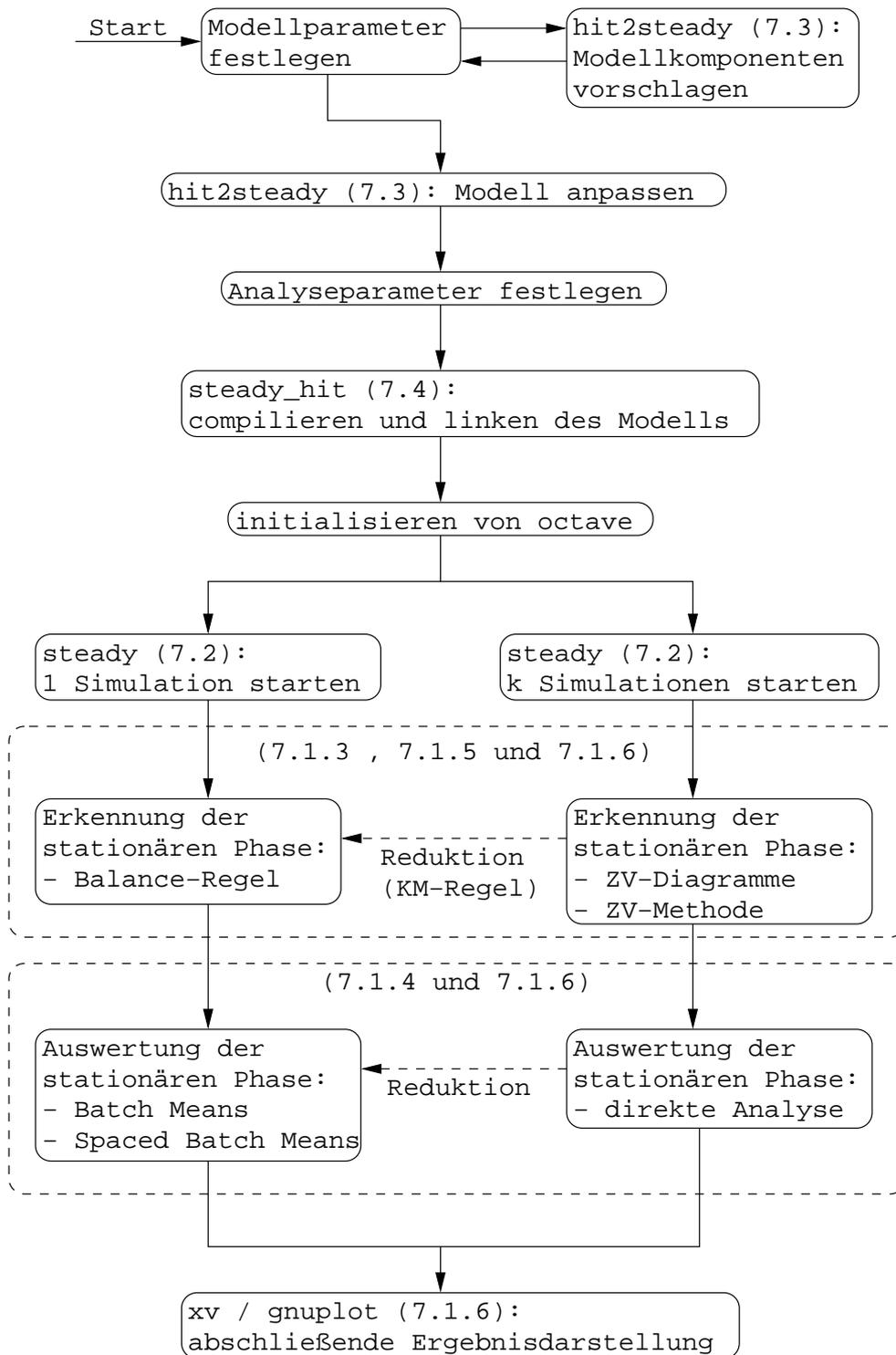


Abbildung 7.1: Das Ablaufdiagramm des Analyse-Tools *ssa* mit den automatisierten Aufrufen der Dienstprogramme *hit2steady*, *octave*, *steady_hit*, *steady*, *xv* und *gnuplot*.

7.1.3 Erkennung der stationären Phase

Zur Erkennung der stationären Phase wurde die “Balance-Regel” (6.2.1), die Zufallsverteilungsdiagramme (6.3.2), die Zufallsverteilungsmethode (6.3.2) und die “kumulativer-Mittelwert-Regel” (6.3.1) realisiert. Die folgenden Abschnitte beschreiben die genaue Umsetzung und welche Auswertungsparameter der Benutzer wählen kann.

Zufallsverteilungsdiagramme

Der Benutzer kann die horizontale Auflösung der Zufallsverteilungsdiagramme wählen. Hierbei ist zu beachten, dass pro Bildzeile ungefähr fünf Simulationen benötigt werden, um ein aussagekräftiges Diagramm zu erhalten. Da hundert parallele Simulationen eine realistische Größenordnung sind, entstehen recht kleine Diagramme mit einer horizontalen Auflösung von 20 Pixeln. Das verwendete Darstellungsprogramm “xv” ermöglicht jedoch eine vergrößerte Darstellung. Die Zufallsverteilungsdiagramme, die durch *ssa* erzeugt werden, wirken hierdurch im Vergleich zu den Zufallsverteilungsdiagrammen, die durch *visual* erzeugt werden, relativ grob. Sie ermöglichen allerdings trotz dieser Schwierigkeit die Erkennung der stationären Phase.

Nach dem Start der parallelen Simulationen werden die übermittelten Daten sequenziell durch die Zufallsverteilungsmethode analysiert. Hiervon unabhängig kann der Benutzer jederzeit die Analyse durch einfachen Tastendruck unterbrechen. Nach dem Tastendruck werden die Verteilungsdiagramme der vier Leistungsmaße Ankunftsrate, Abgangsrate, Population und Verweilzeit erzeugt und mittels “xv” dargestellt. Die Ausgabe erfolgt sortiert nach Modellkomponenten und Diensten. Eine direkte Bearbeitung der Diagramme ist möglich. Der Benutzer kann die Darstellungsgrenzen der relativen Häufigkeit festlegen und dadurch bestimmte Darstellungsbereiche hervorheben. Die veränderte Darstellung wird automatisch im Ergebnisverzeichnis gespeichert. Zu beachten ist noch, dass zu jedem Leistungsmaß eines Dienstes nur jeweils das letzte Diagramm gespeichert wird, da sonst die große Datenmenge schnell an die Systemgrenze stoßen könnte.

Zufallsverteilungsmethode

Zunächst legt der Benutzer die Anzahl der parallelen Simulationen fest. Das Verhältnis zwischen verworfenen Probanden und Beobachtungsintervall (siehe Kapitel 6.3.2) kann der Benutzer ebenfalls festlegen. Der dritte Parameter, den der Benutzer festlegt, bezieht sich auf den Vergleich zweier Stichproben. Hier kann der Benutzer durch einen Grenzwert festlegen, bei wieviel “Unterschied” zwei Stichproben noch als “gleichverteilt” gelten.

Nach dem Start der parallelen Simulationen werden schrittweise neue Simulationsdaten ermittelt. Der Proband wird mit dem Beobachtungsintervall verglichen, wie es in Kapitel 6.3.2 beschrieben ist. Zum Vergleich zweier Stichproben können mehrere statistische Verfahren benutzt werden. Diese Verfahren sind in Kapitel 3.6 beschrieben. Bei der Implementierung von *ssa* wurden der χ^2 -Test, der Kolmogoroff-Smirnoff-Test, der Mann-Whitney U-Test und der Wilcoxon Signed-Rank-Test getestet. Für einfache Modelle lieferten sie alle befriedigende Ergebnisse. Bei größeren Modellen war die Laufzeit des Algorithmus jedoch nicht mehr akzeptabel. Dies liegt nicht an einer ineffizienten Realisierung der Tests in dem Statistik-Tool *octave*, sondern an der Kommunikation zwischen *octave* und *ssa*. Sehr viele Daten müssen zwischen den eigenständigen Programmen hin und her transportiert werden. Auf Grund dieser Schwierigkeiten wurde in *ssa* der einfache aber schnelle Differenz-Test (siehe Kapitel 3.6.5) implementiert.

Balance-Regel

Die “Balance-Regel” basiert auf einem einzelnen Simulationslauf, daher sind mehrere parallele Simulationen nicht nötig. Im Gegensatz zu dem Vorgehen aus Kapitel 6.2.1 hat bei dieser Implementierung die Datenmenge zur Bestimmung des Mittelwertes nicht dieselbe Länge wie die Datenmenge, die auf “Balance” getestet wird. Dieses Vorgehen verspricht einen “genaueren” Mittelwert und ist ein Kompromiss aus der “Balance-Regel” und der “Balance- χ^2 -Regel”. Der Benutzer kann also beide Parameter getrennt wählen. Die Anzahl der zu verwerfenden Stichproben ist von dem Benutzer ebenfalls frei wählbar. Hierdurch kann eine beliebige Genauigkeit des Verfahrens erreicht werden. Es ist jedoch zu beachten, dass eine hohe Genauigkeit mit einer langen Laufzeit verbunden ist. Durch einen weiteren Parameter kann der Benutzer bestimmen, ab wann eine Stichprobenmenge als “balanciert” gilt. Durch einen Grenzwert wird der erlaubte Bereich der Abweichung der beiden Balance-Werte festgelegt.

Kumulativer-Mittelwert-Regel

Die Reduktion von mehreren Simulationsläufen auf einen stellvertretenden Simulationslauf wird ebenfalls berücksichtigt. Bei dieser Reduktionsmethode muss die Anzahl der parallelen Simulationen durch den Benutzer festgelegt werden. Um den Programmieraufwand gering zu halten, wird auf den stellvertretenden Simulationslauf die “Balance-Regel” angewendet. Der Benutzer wählt die entsprechenden Auswertungsparameter für die “Balance-Regel”, wie zuvor beschrieben.

7.1.4 Auswertung der stationären Phase

Zur statistischen Auswertung der stationären Phase wurden in dem Programm *ssa* drei Verfahren implementiert. Zum einen wurde die “direkte Analyse” aus Kapitel 6.3.3 umgesetzt. Dieses Verfahren entspricht dem “Replication/Deletion”-Vorgehen, das von Law und Kelton in [27] vorgeschlagen wird. Zum anderen wurde das Verfahren “Batch Means” und die Variante “Spaced Batch Means” (siehe Kapitel 6.2.2) implementiert. Die Reduktion von mehreren Simulationen auf einen stellvertretenden Simulationslauf wurde ebenfalls berücksichtigt.

Direkte Analyse

Bei der “direkten Analyse” der parallelen Simulationen kann der Benutzer einen Grenzwert für den Quotienten aus dem Mittelwert und dem 95%-Konfidenzintervall wählen, bei dem die Simulation abgebrochen werden soll. Als Konfidenzintervall wurde das Standardkonfidenzintervall aus Kapitel 3.3.8 gewählt. Sollte der Quotient aus dem Mittelwert und dem 95%-Konfidenzintervall den Grenzwert überschreiten, so wird die Simulation fortgesetzt.

Batch Means

Bei dieser Implementierung des Verfahrens “Batch Means” wird mit der Gruppengröße eins gestartet. Diese Gruppengröße wird solange um eins erhöht, bis der “Run-Test” des Statistik-Tools “octave” die Unabhängigkeit der Daten erkennt. Mit diesen Gruppen wird dann so verfahren, wie es in Kapitel 6.2.2 beschrieben ist. Der Benutzer hat auch hier die Möglichkeit, einen Grenzwert für den Quotienten aus dem Mittelwert und dem 95%-Konfidenzintervall zu wählen. Außerdem kann der Benutzer den Erfolg des “Run-Test” durch einen weiteren Grenzwert festlegen.

Spaced Batch Means

Die Implementierung von “Spaced Batch Means” ist an “Batch Means” angelehnt. Eine Gruppe besteht immer nur aus einem Wert. Es wird lediglich die Anzahl an nicht betrachteten Werten zwischen den Gruppen variiert. Auch hierbei testet der “Run-Test” des Statistik-Tools “octave” die Daten auf Unabhängigkeit. Daher legt der Benutzer dieselben Parameter fest, wie bei der Auswertung durch “Batch Means”.

Reduktion

Bei der Reduktion auf eine stellvertretende Datenmenge werden keine gesonderten Parameter benötigt. Auf die stellvertretende Datenmenge kann entweder das

"Batch Means"-Verfahren oder das "Spaced Batch Means"-Verfahren angewendet werden.

7.1.5 Verkehrsflussgleichgewicht

Da das Dienstprogramm *steady* die Leistungsmaße Ankunftsrate und Abgangsrate liefert, kann überprüft werden, ob Verkehrsflussgleichgewicht (siehe [10], Kapitel 4) besteht. Das Prinzip des Verkehrsflussgleichgewichts ist eine notwendige Bedingung für Stationarität. Wenn sich die Ankunftsrate und die Abgangsrate mit steigender Simulationszeit nicht annähern, kann das zu analysierende Modell keine stationäre Phase haben. Leider lässt sich hierdurch noch keine Abbruchbedingung für die Simulation ableiten. Wenn eine bestimmte Simulationszeit betrachtet wurde, in der sich kein Verkehrsflussgleichgewicht eingestellt hat, kann noch nicht angenommen werden, dass das Modell keine stationäre Phase besitzt. Die stationäre Phase könnte in der weiteren Simulation noch eintreten. Das Prinzip des Verkehrsflussgleichgewichts kann also nur, wie schon erwähnt, als eine notwendige Bedingung für Stationarität betrachtet werden.

7.1.6 Ergebnisdarstellung

Damit der Benutzer von *ssa* über den Verlauf der Tests und Analysen informiert ist, werden die gewonnenen Ergebnisse graphisch dargestellt. Anhand dieser Darstellungen kann der Auswertungsverlauf verfolgt werden. Der Benutzer erkennt, ob die Auswertungsparameter geeignet gesetzt wurden.

Ankunftsrate versus Abgangsrate

Die Abgangsrate und die Ankunftsrate werden von *ssa* miteinander verglichen. Die Abbildung 7.2 zeigt den zeitlichen Verlauf der Vergleiche. Aufgetragen ist sowohl der Vergleich der jeweiligen Intervalle, als auch der Mittelwert aller betrachteten Vergleiche.

Ein Vergleich zwischen der Ankunftsrate a und der Abgangsrate c findet durch Differenzbildung statt:

$$e = \frac{a - c}{c} \quad (7.1)$$

e sei die genormte Differenz. Sollte für die Abgangsrate $c = 0$ gelten, wird durch die Ankunftsrate a dividiert. Sollte auch die Ankunftsrate $a = 0$ sein, so ist $e = 0$. Durch diese etwas umständliche Berechnung wird eine Division durch Null vermieden.

Eine stationäre Phase kann nur erreicht werden, wenn die genormte Differenz e klein ist. Nur in diesem Fall kann das Verkehrsflussgleichgewicht eintreten. Ist

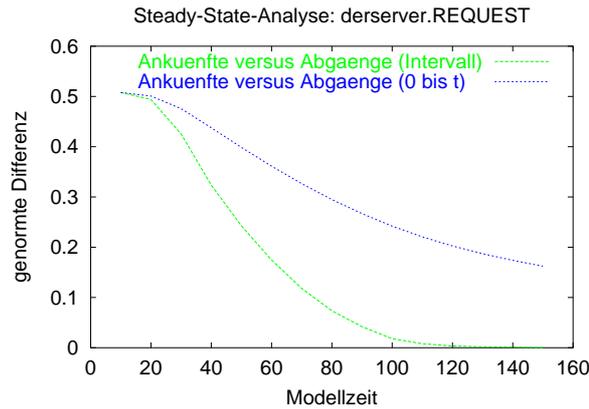


Abbildung 7.2: Genormte Differenz der Ankunftsrate und Abgangsrate.

die genormte Differenz e konstant auf einem hohen Niveau, ist das betrachtete Modell nicht stabil.

Vergleichsergebnisse der Zufallsverteilungsmethode

Bei der Zufallsverteilungsmethode wird eine ausgewählte Stichprobe mit ihren Nachfolgern verglichen. Der zeitliche Verlauf dieser Vergleiche ist in Abbildung

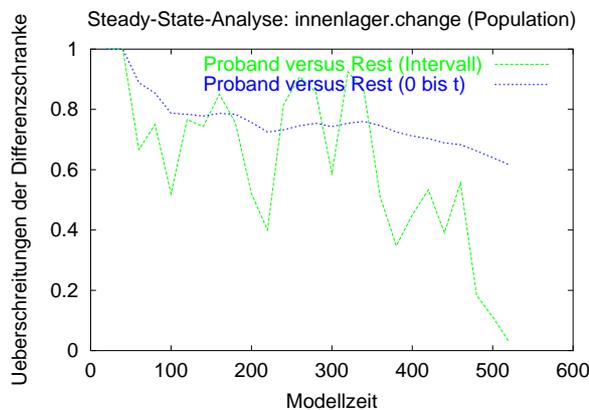


Abbildung 7.3: Der zeitliche Verlauf der Vergleiche der aktuellen Stichprobe mit den nachfolgenden Stichproben.

7.3 aufgetragen. Die eine Kurve repräsentiert die Vergleiche der aktuellen Stichprobe (Proband) mit ihren nachfolgenden Stichproben. Die andere Kurve zeigt den Mittelwert aller Vergleiche der bisherigen Probanden. Fällt die erste Kurve auf ein niedriges Niveau, kann angenommen werden, dass eine stationäre Phase vorliegt. Bleibt die Kurve konstant auf einem hohen Niveau, sind die aufeinander folgenden Zufallsverteilungen unterschiedlich und es liegt keine stationäre Phase vor.

Stationärer Mittelwert verglichen mit Gesamt-Mittelwert

Die Abbildung 7.4 besteht aus drei Kurven. Die erste Kurve ist das eigentliche

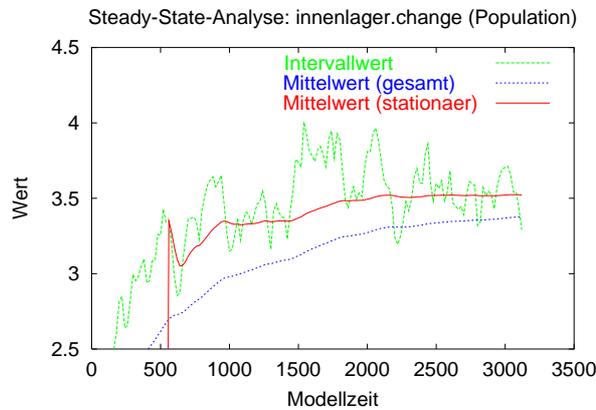


Abbildung 7.4: Stationärer Mittelwert versus gesamter Mittelwert.

Leistungsmaß der einzelnen Intervalle. Die beiden anderen Kurven sind Mittelwerte dieser Leistungsmaße. Die eine Kurve beschreibt den Mittelwert von der Modellzeit Null bis zur aktuellen Modellzeit (Gesamt-Mittelwert). Die andere Kurve zeigt den Mittelwert beginnend mit dem Anfang der stationären Phase bis zum aktuellen Modellzeitpunkt (stationärer Mittelwert). Da diese drei Kurven in einem Diagramm aufgetragen werden, lässt sich der Vorteil, bedingt durch das Verwerfen der Daten der transienten Phase, direkt ablesen.

Stationäre Phase

Die übliche Darstellung von Leistungsmaßen, die durch Simulation bestimmt werden, wird auch durch *ssa* unterstützt. Hier wird jedoch die transiente Pha-

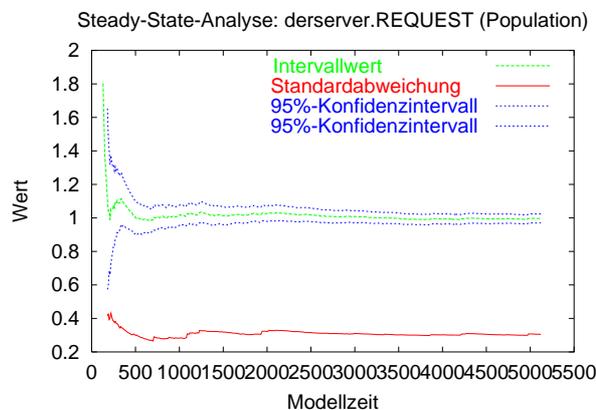


Abbildung 7.5: Mittelwert und Konfidenzintervall.

se komplett außer Acht gelassen, so dass die Darstellung mit dem Beginn der stationären Phase beginnt. Aufgetragen ist der Mittelwert mit seinem Konfidenzintervall und der Standardabweichung. Der horizontale Darstellungsbereich lässt sich beliebig wählen. Die Abbildung 7.5 ist ein Beispiel dieser Darstellungsform.

7.1.7 Parallelisierung

Die parallelen Simulationsläufe werden entsprechend der Rechnerkonstellation und der Rechengeschwindigkeiten auf die Rechner verteilt. So entsteht eine sternförmige Struktur der Simulationen mit dem Programm *ssa* im Mittelpunkt.

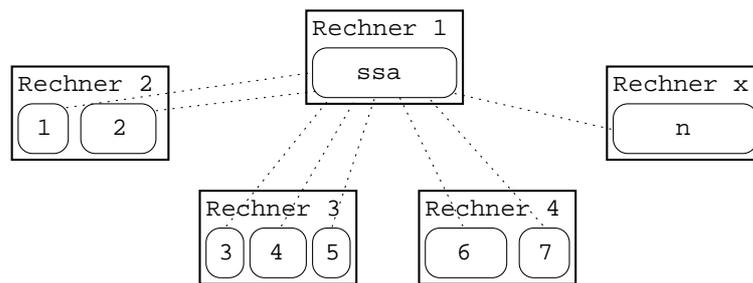


Abbildung 7.6: Die n Simulationen werden auf x Rechner verteilt.

Die technische Umsetzung erfolgte mit Hilfe der Bibliothek “unistd”. Der Aufruf der parallelen Simulationen erfolgt durch Abspalten eines Prozesses durch die Funktion “vfork” und anschließendem Systemaufruf durch “execlp”. Die Einbindung von “Remote”-Rechnern erfolgt durch die “Remote Shell” *rsh*.

Für die Kommunikation zwischen den Simulationen und dem Programm *ssa* wurden sogenannte “Pipes” angelegt. Die Simulationen benutzen das eine Ende der “Pipe” zum Schreiben ihrer Simulationsdaten. Dadurch kann das Programm *ssa* die Daten am anderen Ende der “Pipe” empfangen. Da das Filterprogramm *steady* die Simulationsdaten normalerweise an den Standardausgabekanal leitet, muss ein kleiner Trick angewendet werden. Mit der Funktion “dub2” kann die Ausgabe an den Standardausgabekanal in die vorgesehene “Pipe” umgeleitet werden.

An jede “Pipe” ist ein Datenpuffer angeschlossen. Dieser Datenpuffer sorgt dafür, dass die geschriebenen Daten nicht direkt gelesen werden müssen. Wenn dieser Datenpuffer jedoch an seine Kapazitätsgrenze stößt, wird der Prozess blockiert, der die Daten in die “Pipe” schreiben will. Ist der Datenpuffer hingegen leer, wird der Prozess blockiert, der aus der “Pipe” lesen will. Daher hat die Verwendung der “Pipes” den Vorteil, dass die Synchronisation der Simulationen mit dem Programm *ssa* automatisch geschieht. Der schnellere Prozess wartet auf den langsameren Prozess.

Neben dieser automatischen Prozesssteuerung kann *ssa* die Simulationen über Signale steuern. Dazu muss *ssa* lediglich die Prozess-ID des Filterprogramms *steady* und der zugehörigen Simulation bekannt sein. Die Prozess-IDs können durch die Funktion “*getid*” erfragt werden. Bei dem Senden der Signale hilft die Funktion “*kill*” aus der Bibliothek “*signal*”. Die nützlichsten Signale sind

SIGKILL: zum Beenden eines Prozesses,

SIGSTOP: zum Anhalten eines Prozesses,

SIGCONT: zum Fortsetzen eines Prozesses.

Sollte das Programm *ssa* aus irgendeinem Grund vorzeitig abgebrochen werden, werden zuerst alle gestarteten Simulationen beendet. Dies verhindert, dass nach Beendigung von *ssa* Simulationsprozesse weiterarbeiten, obwohl sie nicht benötigt werden. “Signal-Handler” stellen diese Funktion sicher.

Eine genaue Beschreibung der verwendeten Bibliotheken und Funktionen sind in [24] nachzulesen.

7.1.8 Implementierung

Das Programm *ssa* ist durch “Prototyping” entstanden. Das bedeutet, dass zuerst eine Version von *ssa* entwickelt wurde, die nur wenige Funktionen besaß. In weiteren Versionen wurde die Funktionalität erweitert. So ergaben sich Methoden, die semantisch zusammen gehören. Diese Methoden wurden nach und nach ausgelagert. So entstand die Modulstruktur von *ssa*. Die Module haben folgende Bedeutung:

main: Steuerung

transient: Erkennung der stationären Phase

analyse: Statistischen Auswertung der stationären Phase

daten: Datenstruktur für Simulationsdaten

rechner: Datenstruktur für die Rechnerkonstellation

job: Datenstruktur für Simulationsläufe

darstellung: Erzeugung von Abbildungen

bmp: Erzeugung von Dateien im BMP-Format

octave: Start und Kommunikation mit “octave”

tools: Sonstiges

Ein Schwerpunkt der Implementierung ist die Datenstruktur für die Simulationsdaten. Sie ist sortiert nach Komponenten, Diensten und Leistungsgrößen. Eine Leistungsgröße besteht wiederum aus vielen Modellzeitpunkten, die eine Stichprobe repräsentieren. Um eine hohe Effizienz zu garantieren, wurden doppelt verkettete Listen (Bibliothek “list”) und binäre Suchbäume (Bibliothek “map”) verwendet. Bei langen Simulationen wird die Datenstruktur sehr groß, sodass Effizienz für eine kurze Laufzeit sehr wichtig ist.

Eine große Anzahl an weiteren Standardbibliotheken wurde verwendet, auf eine Aufzählung und eine Beschreibung wird hier jedoch verzichtet.

7.2 steady

Das Programm *steady* kann die Ereignisse eines Simulationslaufes interpretieren und die Leistungsgrößen Ankunftsrate, Abgangsrate, Verweilzeit und Population berechnen. Die Ausgabe der Ereignisse der Simulation muss in dem Format erfolgen, das in [9] Anhang G.5.1.2 beschrieben ist. Das Programm *steady* ist ein Filter, der aus den Ereignissen einer Simulation geeignete Leistungsmaße bestimmt. Der Programmablauf von *steady* findet parallel zur Simulation statt und die Leistungsmaße werden daher ebenfalls parallel zur Simulationszeit berechnet und ausgegeben.

7.2.1 Aufruf und Ergebnisse

Der direkte Aufruf von *steady* ist kompliziert, da das Programm *steady* nur als ein Dienstprogramm für *ssa* gedacht ist. Der Benutzer muss zum einen sicherstellen, dass die zu analysierende HISLANG-Datei mit dem Programm *hit2steady* bearbeitet wurde. Des Weiteren muss die Messobjektdatei (siehe Kapitel 7.3) vorhanden sein und den Namen der zu analysierenden HISLANG-Datei besitzen, ergänzt um den Anhang “.steady.obj”. Die durch *hit2steady* erzeugte HISLANG-Datei muss mit *steady_hit* in eine ausführbare Datei umgewandelt werden. Alle Dateien, die erzeugt wurden, müssen sich in einem Verzeichnis befinden. Sind diese Voraussetzungen geschaffen, kann *steady* manuell mit dieser Aufrufsyntax gestartet werden:

```
PFAD/steady HISLANGDatei Intervall Ausgaben Speicherbereich
```

Mit “HISLANGDatei” ist der Dateiname der ursprünglichen HISLANG-Datei gemeint. “Intervall” ist die Schrittweite in der Modellzeit, nach der eine Ausgabe der Leistungsmaße erfolgt. Durch “Ausgaben” kann die Anzahl der Ausgaben der Leistungsmaße festgelegt werden. Wird dieser Übergabeparameter auf “-1” gesetzt, erfolgt eine unbegrenzte Ausgabe. Der Parameter “Speicherbereich” legt

die Größe des Hauptspeichers fest, die der Simulation zur Verfügung gestellt wird (siehe Kapitel 7.4).

Zu Beginn der Ausführung werden die Prozess-IDs des “steady”-Prozesses und des “steady.hitcode”-Prozesses an den Standardausgabekanal geliefert. Der “steady.hitcode”-Prozess entsteht durch den Simulationsstart (siehe Kapitel 7.4). Mit Hilfe dieser Prozess-IDs kann das Programm *ssa* die Ausführung anhalten, fortsetzen oder beenden. Alle berechneten Leistungsmaße werden ebenfalls an den Standardausgabekanal geliefert. Die Ausgabe erfolgt in einer festen Form:

- Eine Ausgabezeile besteht aus der aktuellen Modellzeit, dem Komponentennamen, dem Dienstenamen und den vier Leistungsmaßen Ankunftsrate, Abgangsrate, Verweilzeit und Population. Alle Werte sind durch ein Leerzeichen voneinander getrennt.
- Für jeden Dienst einer Komponente wird eine dieser Zeilen ausgegeben.
- Durch den Übergabeparameter “Intervall” wird festgelegt, zu welchen Modellzeiten eine Ausgabe erfolgt.
- Die Ausgaben der unterschiedlichen Modellzeiten werden durch eine Leerzeile getrennt.

Das nachfolgende Beispiel ist die Ausgabe einer Simulation, bei der nur eine Komponente betrachtet wurde:

```
5 derserver REQUEST 21 2.2 93.5785 98.6922
10 derserver REQUEST 1.2 1.2 91.4399 94.7961
15 derserver REQUEST 1 1.8 88.7344 92.6599
20 derserver REQUEST 1.2 2 82.4731 88.6372
25 derserver REQUEST 0.8 2 78.1499 81.923
30 derserver REQUEST 1.4 1.2 77.8613 80.5881
35 derserver REQUEST 1.2 2.2 73.2195 77.3992
40 derserver REQUEST 1.4 1.4 72.6554 75.0551
45 derserver REQUEST 0.2 2.4 65.5841 69.4173
50 derserver REQUEST 1.2 1 63.8301 65.6739
55 derserver REQUEST 0.6 1.6 60.142 63.7609
60 derserver REQUEST 1.2 2.6 52.0813 57.5621
65 derserver REQUEST 1.2 1.6 49.4571 51.6168
70 derserver REQUEST 0.6 2.2 42.8613 47.0197
75 derserver REQUEST 1.2 1.2 41.0098 43.8736
80 derserver REQUEST 0.4 2 35.6201 39.5753
85 derserver REQUEST 1.2 2.4 27.4399 32.0767
90 derserver REQUEST 1 2.6 20.6134 25.7083
95 derserver REQUEST 0.8 3.8 5.98514 13.7242
100 derserver REQUEST 0.8 2 2.02523 4.18755
```

Das Format ist sehr einfach und es müssen keine großen Änderungen vorgenommen werden, wenn eine weitere Leistungsgröße ergänzt werden soll.

7.2.2 Leistungsgrößen

Auf die Bestimmung von Leistungsmaßen wurde schon in Kapitel 4.1 eingegangen. In diesem Abschnitt soll die konkrete Realisierung in dem Programm *steady* beschrieben werden.

In dem Programm *steady* basiert die Berechnung der Leistungsmaße auf disjunkten Intervallen. Diese Intervalle werden herangezogen, um die Ankunftsrate, die Abgangsrate, die Verweilzeit und die Population zu berechnen. Diese vier Leistungsmaße sind klar definierte Begriffe (siehe [10]). Ihre Berechnung aus den Ereignissen der jeweiligen Intervalle wird in den folgenden Abschnitten geschildert.

Ein HISLANG-Modell besteht aus hierarchisch strukturierten Komponenten. Jede dieser Komponenten besteht aus einer "Entry-Area", einer "Service-Area" und einer "Exit-Area" (siehe [9]). Diese Bereiche haben eine unterschiedliche Intention. In dieser Diplomarbeit werden diese Bereiche jedoch außer Acht gelassen. Es werden nur die Ereignisse "Entry" und "Exit" berücksichtigt.

Ankunftsrate

Die Ankunftsrate a beschreibt, wieviele "jobs" (Definition siehe [10]) pro Zeiteinheit eine Modellkomponente betreten. Das Betreten einer Modellkomponente geschieht durch das Ereignis "Entry". Zur Bestimmung der Ankunftsrate a wird ein Zähler A benötigt, der zu Beginn eines Intervalls auf Null gesetzt wird

$$A = 0 \quad (\text{zum Intervallbeginn}) \quad (7.2)$$

und jedes "Entry"-Ereignis des Intervalls zählt:

$$A := A + 1 \quad \text{für Ereignis "Entry"}. \quad (7.3)$$

Ist das Ende des Intervalls erreicht, ergibt die Division dieses Zählers durch die Modellzeitlänge T des Intervalls die Ankunftsrate:

$$a = \frac{A}{T} \quad (7.4)$$

Da die Länge der Intervalle immer größer als Null ist, ist eine Division durch Null unmöglich.

Abgangsrate

Die Abgangsrate c wird analog zur Ankunftsrate a berechnet. Das grundlegende Ereignis ist in diesem Fall ein "Exit".

Verweilzeit

Mit der Verweilzeit ist hier die kumulative Verweilzeit J gemeint. Eine genaue Definition ist in [10] nachzulesen. Zur Berechnung der kumulativen Verweilzeit

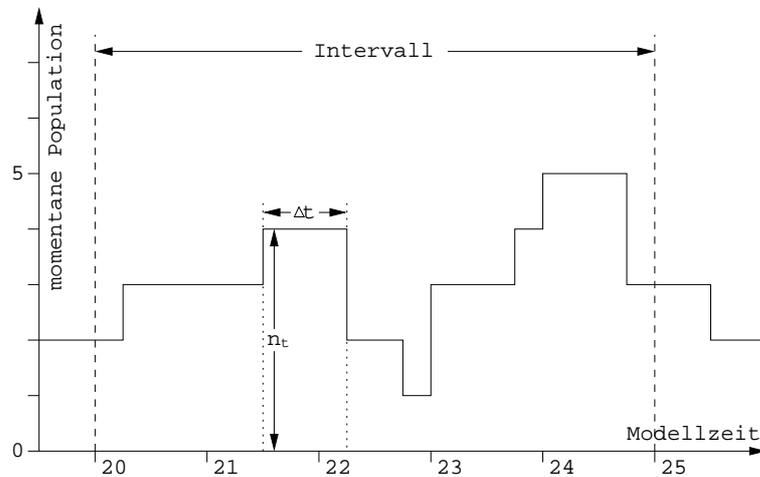


Abbildung 7.7: Berechnung der kumulativen Verweilzeit eines Intervalls aus den konstanten Teilintervallen.

J eines Intervalles muss seine Zustandstrajektorie (siehe [10]) betrachtet werden. Aus der Zustandstrajektorie ist ersichtlich, wie lange wie viele “jobs” in der Modellkomponente waren. Die kumulative Verweilzeit J lässt sich durch die momentane Population n_t bestimmen. Die momentane Population n_t ist die Anzahl der “jobs” zum Modellzeitpunkt t in der Modellkomponente. Durch eine Ankunft oder einen Abgang ändert sich die momentane Population. Die Zeit, in der n_t konstant ist, sei δt . Die Multiplikation $n_t \cdot \delta t$ ergibt die kumulative Verweilzeit $J_{\delta t}$ des Teilintervalls δt . Die Summe aller $J_{\delta t}$ ergibt die kumulative Verweilzeit J des gesamten Intervalls.

Population

Die durchschnittliche Population n eines Intervalls wird von der kumulativen Verweilzeit abgeleitet:

$$n = \frac{J}{T} \quad . \quad (7.5)$$

T sei die Länge des betrachteten Intervalls.

7.2.3 Implementierung

Das Programm *steady* ist in die Module “main”, “datenstruktur”, “kenngrossen” und “tools” aufgeteilt. Das Modul “main” steuert den Programmablauf. In dem

Modul “datenstruktur” befinden sich alle Funktionen, die benötigt werden, um eine Datenstruktur aufzubauen und zu verwalten. Diese Datenstruktur beinhaltet, welche Modellkomponenten und Dienste es gibt. Ebenso verwaltet sie den Weg eines Prozesses (oder “jobs”) durch das Modell. Das Modul “kenngrößen” ist für die Interpretation der Ereignisse zuständig. Die Leistungsmaße werden in Abhängigkeit von den Ereignissen berechnet und in der Datenstruktur aktualisiert. Das Modul “tools” übernimmt sonstige Aufgaben.

Zusätzlich werden einige Standardbibliotheken benutzt. Neben den üblichen Bibliotheken `unistd.h`, `iostream`, `string`, `fstream`, `strstream`, `stdio.h` und `stdlib.h` werden die Bibliotheken `signal.h` und `sys/wait.h` verwendet. Die beiden letztgenannten Bibliotheken ermöglichen eine effiziente Kommunikation zwischen *steady* und der Simulation. Des Weiteren ermöglichen sie eine Steuerung der Simulation durch das Filterprogramm *steady*. Die Datenstruktur setzt sich zusammen aus einigen Standard-“Dictionaries”. Da bei einer Simulation sehr viele Ereignisse entstehen, ist die Wahl dieser “Dictionaries” von großer Bedeutung. Je nach Anforderung wurden entweder eine doppelt verkettete Liste (Bibliothek: `list`), ein Hashing-Verfahren (Bibliothek: `hash_map`) oder ein binärer Suchbaum (Bibliothek: `map`) verwendet.

Anzumerken ist noch, dass die Ausgabereihenfolge der Ereignisse durch die Simulation willkürlich ist, wenn Ereignisse zur selben Modellzeit stattfinden. Dies bedeutet, dass nicht immer ein “Exit” auf ein “Entry” folgen muss. Diese Besonderheit tritt häufig dann auf, wenn ein HISLANG-Modell aus Komponenten besteht, die keine Modellzeit verbrauchen. Bei der Berechnung der Leistungsmaße muss dies berücksichtigt werden, denn eine Auswertung, die in der Trace-Ausgabe der Simulation Zeile für Zeile vorgeht, kann zu Problemen führen.

7.3 hit2steady

Das Programm *hit2steady* erweitert ein gültiges HISLANG-Modell um einige Eigenschaften. Eine Eigenschaft ist, dass der Startseed der Simulation über eine externe Datei festgelegt werden kann. Eine weitere Eigenschaft ist, dass das Modell alle Simulationsereignisse an den Standardausgabekanal liefert. Das Format dieser Ausgabe und die Definition der Ereignisse lassen sich in [9] Anhang G.5.1.2 nachlesen. Der ursprüngliche Experimententeil wird komplett ersetzt.

7.3.1 Aufruf und Ergebnisse

hit2steady wird mit folgender Syntax aufgerufen:

```
PFAD/hit2steady [OPTION] [OPTION] ... HITDATEI
```

Beispiele:

```
hit2steady -info HitDatei.hit
hit2steady -interact HitDatei.hit
hit2steady -interact -messobjekte objekte.txt HitDatei.hit
```

In den drei Beispielen sind schon alle wichtigen Optionen aufgeführt. Das Programm hat noch mehr Optionen, diese werden an dieser Stelle jedoch nicht erwähnt, da ihre Funktionsweise für diese Diplomarbeit nicht von Interesse ist.

-info

Diese Option unterdrückt die Erzeugung einer überarbeiteten HISLANG-Datei. Es wird lediglich die Datei HITDATEI.cmp und die Datei HITDATEI.typ erzeugt.

-interact

Diese Option erzeugt eine überarbeitete HISLANG-Datei, die keine Abbruchbedingung besitzt.

-messobjekte DATEINAME

Wenn diese Option nicht gewählt ist, werden die Ereignisse aller Modellkomponenten protokolliert. Wenn über diese Option eine Datei angegeben wird, dann werden nur die Modellkomponenten protokolliert, die in der Datei vermerkt sind. Diese Datei muss das selbe Format wie die Datei HITDATEI.cmp besitzen.

Da die Erstellung der Messobjektdatei, welche die zu analysierenden Messobjekte benennt, für den Benutzer recht aufwendig sein kann, erzeugt die Option “-info” die Datei “HITDATEI.cmp”. Diese Datei beinhaltet alle automatisch erkannten Modellkomponenten. Durch Löschen der überflüssigen Modellkomponenten aus dieser Datei kann sich der Benutzer seine eigene Messobjektdatei erzeugen. Das Beispiel einer “HITDATEI.cmp”-Datei:

```
SSM
SSM.DerServer
```

“SSM” ist hierbei die oberste Modellkomponente und “SSM.DerServer” ist eine untergeordnete Komponente. Formal besteht die Datei aus den Namen der Modellkomponenten, die durch einen Zeilenumbruch voneinander getrennt sind. Die Hierarchiestufen werden durch einen “.” deutlich gemacht.

Da der Experimentteil komplett ersetzt wird, muss auch ein neuer Modellname festgelegt werden. Es wird der Name “SSM” (Steady-State-Modell) benutzt. Der ursprüngliche Typname des Modells wird in der Datei “HITDATEI.typ” festgehalten.

Der Startseed der Simulation wird über eine externe Datei mit dem Namen "seed.txt" festgelegt. Diese Datei muss sich in demselben Dateiverzeichnis wie die überarbeitete HISLANG-Datei befinden. Die Seeddatei besteht aus einem Seedwert, der durch einen Zeilenumbruch abgeschlossen wird. Der Seedwert muss eine Integerzahl aus dem Bereich $\{-2^{31}; 2^{31} - 1\}$ sein.

Das neue HISLANG-Modell hat den Dateinamen des ursprünglichen HISLANG-Modells mit angefügtem ".steady". Die Ergänzungen durch *hit2steady* an den HISLANG-Modellen werden nun an einem kleinen Beispiel demonstriert. Das ursprüngliche HISLANG-Modell hat folgende Form:

```
%COMMON
%%PARM = WARN
%%END
%COPY "counter"
%COPY "semaphor"
%COPY "observer"
TYPE voidRecordName RECORD; END TYPE voidRecordName;
TYPE initServer_Typ MODEL;
  COMPONENT DerServer : server
    (LET ACCEPT := ALWAYS,
     LET SCHEDULE := FCFS (LET CAPACITY := 1),
     LET DISPATCH := EQUAL (LET SPEED := 1.0),
     LET OFFER := ALL);
  TYPE Aufruf SERVICE;
  USE
    SERVICE Anfrage(B2H___amount : REAL DEFAULT 1.0);
  END USE;
  BEGIN
    Anfrage(negexp(2));
  END {SERVICE} TYPE Aufruf;
  TYPE Quellen SERVICE;
  BEGIN
    LOOP
      hold(negexp(1));
      CREATE 1 PROCESS Aufruf;
    END LOOP;
  END {SERVICE} TYPE Quellen;
  TYPE InitQuellen SERVICE;
  BEGIN
    CREATE 100 PROCESS Aufruf;
  END {SERVICE} TYPE InitQuellen;
  REFER
    Aufruf
  TO
    DerServer
  EQUATING
    Aufruf.Anfrage WITH DerServer.request;
  END REFER;
  BEGIN
    CREATE 1 PROCESS InitQuellen;
    CREATE 1 PROCESS Quellen;
  END {MODEL} TYPE initServer_Typ;
  {-----}
  EXPERIMENT versuch METHOD SIMULATIVE;
  BEGIN
    EVALUATE MODEL initServer : initServer_Typ(LET SEED := 13);
    EVALUATIONOBJECT
      messgeraet VIA initServer.DerServer;

  BEGIN
    MEASURE POPULATION AT messgeraet;
    MEASURE UTILIZATION AT messgeraet;
    MEASURE THROUGHPUT AT messgeraet;
    MEASURE TURNAROUNDTIME AT messgeraet;

    CONTROL AT messgeraet STOP
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE POPULATION AND
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE UTILIZATION AND
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE THROUGHPUT AND
    CONFIDENCE LEVEL 95 WIDTH 5.0 MEASURE TURNAROUNDTIME;
  END EVALUATE;
```

```
END EXPERIMENT versuch;
```

Das überarbeitete Modell hat die Form:

```
%COMMON
%BIND "seedDateiName" TO seed.txt {*1*}
%BIND "TRACE" TO SYSOUT {*2*}
%PARM = TRACEFORMAT = 2 {*2*}
%PARM = WARN
%END
%COPY "counter"
%COPY "semaphor"
%COPY "observer"
TYPE voidRecordName RECORD; END TYPE voidRecordName;
TYPE initServer_Typ MODEL;
  COMPONENT DerServer : server
    (LET ACCEPT := ALWAYS,
     LET SCHEDULE := FCFS (LET CAPACITY := 1),
     LET DISPATCH := EQUAL (LET SPEED := 1.0),
     LET OFFER := ALL);
  TYPE Aufruf SERVICE;
  USE
    SERVICE Anfrage(B2H___amount : REAL DEFAULT 1.0);
  END USE;
  BEGIN
    Anfrage(negexp(2));
  END TYPE Aufruf;
  TYPE Quellen SERVICE;
  BEGIN
    LOOP
      hold(negexp(1));
      CREATE 1 PROCESS Aufruf;
    END LOOP;
  END TYPE Quellen;
  TYPE InitQuellen SERVICE;
  BEGIN
    CREATE 100 PROCESS Aufruf;
  END TYPE InitQuellen;
  REFER
    Aufruf
  TO
    DerServer
  EQUATING
    Aufruf.Anfrage WITH DerServer.request;
  END REFER;
  BEGIN
    CREATE 1 PROCESS InitQuellen;
    CREATE 1 PROCESS Quellen;
  END TYPE initServer_Typ;

EXPERIMENT SteadyState METHOD SIMULATIVE;
VARIABLE meinSeed : INTEGER;
      seedDatei : INFILE;
BEGIN
  OPEN seedDatei, "seedDateiName" LENGTH 80; {*1*}
  READ FILE seedDatei, meinSeed; {*1*}
  CLOSE seedDatei; {*1*}
  EVALUATE MODEL SSM : initServer_Typ(LET SEED := meinSeed);
  EVALUATIONOBJECT messobjekt VIA SSM;
  BEGIN
    MEASURE POPULATION AT messobjekt;
    CONTROL TRACEALL {*2*}
      AT messobjekt; {*2*}
  END EVALUATE;
END EXPERIMENT SteadyState;
```

Die mit “{*1*}” markierten Zeilen sorgen dafür, dass der Startseed aus der externen Datei “seed.txt” gelesen wird. Die mit “{*2*}” markierten Zeilen erzeugen eine Ausgabe der Ereignisse. In diesem Fall werden die Ereignisse aller Modellkomponenten ausgegeben. Sollen jedoch nicht alle Modellkomponenten betrachtet werden, so hat der Experimentteil eine etwas unterschiedliche Form:

```

EXPERIMENT SteadyState METHOD SIMULATIVE;
VARIABLE meinSeed : INTEGER;
        seedDatei : INFILE;
BEGIN
  OPEN seedDatei, "seedDateiName" LENGTH 80; {*1*}
  READ FILE seedDatei, meinSeed; {*1*}
  CLOSE seedDatei; {*1*}
  EVALUATE MODEL SSM : initServer_Typ(LET SEED := meinSeed);
    EVALUATIONOBJECT
      messobjekt VIA SSM;
      messobjekt0 VIA SSM.DerServer;
  BEGIN
    MEASURE POPULATION AT messobjekt;
    CONTROL {*2*}
      AT messobjekt0 TRACE; {*2*}
  END EVALUATE;
END EXPERIMENT SteadyState;

```

Mit diesem Experimentteil wird nur die Komponente “SSM.DerServer” betrachtet. Da die Ausgabe der Ereignisse eine sehr große Datenmenge zur Folge hat, ist es sehr wichtig, dass nur ausgewählte Komponenten betrachtet werden können.

| | | | | |
|---------------|-----------------|--------------|---------|-----|
| 0.000000E+000 | 3 BIRTH | initserver_t | aufruf | |
| 0.000000E+000 | 3 ANNOUNCE | | | |
| 0.000000E+000 | 3 >ENTRY | initserver_t | aufruf | ... |
| 0.000000E+000 | 3 ENTRY>SERVICE | initserver_t | aufruf | |
| 0.000000E+000 | 3 ANNOUNCE | initserver_t | aufruf | ... |
| 0.000000E+000 | 3 >ENTRY | derserver | REQUEST | |
| 0.000000E+000 | 3 ENTRY>SERVICE | derserver | REQUEST | ... |
| 3.083724E-001 | 3 SERVICE>EXIT | derserver | REQUEST | |
| 3.083724E-001 | 3 EXIT> | derserver | REQUEST | |
| 3.083724E-001 | 3 SERVICE>EXIT | initserver_t | aufruf | |
| 3.083724E-001 | 3 EXIT> | initserver_t | aufruf | |
| 3.083724E-001 | 3 DEATH | initserver_t | aufruf | |

Diese Ereignisse sind nur ein willkürlicher Auszug aus der Ausgabe einer Simulation. Das genaue Ausgabeformat ist in [9] Anhang G.5.1.2 beschrieben. Dort wird ebenfalls die Interpretation der Ereignisse angegeben.

7.3.2 Implementierung

Einen Hauptteil der Implementierung macht das Erkennen der Struktur des HISLANG-Modells aus. Die hierarchische Struktur muss in einer geeigneten Datenstruktur erfasst werden können, um dem Benutzer eine Hilfestellung in Bezug auf die Wahl der zu analysierenden Modellkomponenten geben zu können. Hierbei muss zwischen dem Typ und der Instanz einer Komponente unterschieden werden.

Die ursprüngliche HISLANG-Datei wird zeilenweise eingelesen. Ergänzungen, die nicht von der hierarchischen Struktur abhängen, können direkt eingefügt werden. Der ursprüngliche Experimentteil wird gelöscht und ersetzt. Der neue Experimentteil kann erst erzeugt werden, wenn die zu analysierenden Modellkomponenten bekannt sind.

Die Programmstruktur von *hit2steady* ist modular aufgebaut. Die Module sind so gewählt, dass semantisch ähnliche Funktionen zusammengefasst wurden:

einlesen:

Einlesen der ursprünglichen HISLANG-Datei und der Messobjekte.

erzeugen:

Schreiben der veränderten HISLANG-Datei.

interact:

Erzeugt den Experimentteil für eine Simulation ohne Abbruchbedingung.

main:

Erkennen der Aufrufparameter und Ablaufsteuerung.

struktur:

Datenstruktur für ein hierarchisches HISLANG-Modell.

utils:

Sonstige Funktionen

Neben den schon erwähnten Modulen wird auf einige Standardbibliotheken zurückgegriffen: `iostream`, `string`, `fstream`, `strstream`, `list`. Diese Standardbibliotheken stellen eine Vielzahl von Standardoperationen zur Verfügung.

7.4 steady_hit

`steady_hit` ist eine Befehlsabfolge (“Script”), die eine HISLANG-Datei in ein ausführbares Programm verwandelt. Zunächst wird die Syntax des HISLANG-Modells überprüft, bei einem Fehler wird die Ausführung abgebrochen. Im nächsten Schritt erfolgt eine “Compilation” mit anschließendem “Linken”. Bei diesen Schritten werden temporäre Links und Dateien erzeugt. Ebenfalls wird eine ausführbare Datei mit dem Namen `steady.hitcode` erzeugt. Bei der Ausführung dieser Datei wird eine Simulation des ursprünglichen HISLANG-Modells durchgeführt.

Das “Script” `steady_hit` ist von dem bekannten “Script” `hit` abgeleitet. Während `hit` jedoch auf die “Compilation” und das “Linken” eine Simulation folgen lässt, wird bei der Ausführung von `steady_hit` nur eine ausführbare Datei erzeugt, ohne die Simulation zu starten.

Zu beachten ist noch, dass das Programm `steady.hitcode` einige der temporären Dateien zur Ausführung benötigt. Daher werden nicht alle temporären Dateien durch das Script `steady_hit` gelöscht. Diese temporären Dateien können von dem Benutzer nach der Simulationsausführung gelöscht werden.

Die Syntax zum Aufruf von `steady_hit` besteht aus dem Befehlsnamen und der HISLANG-Datei:

```
PFAD/steady_hit Dateiname
```

Der Aufruf von *steady.hitcode* hat folgende Syntax:

```
PFAD/steady.hitcode -p -g -k=Speicherbereich
```

Die Optionen “-p” und “-k” sind Standardoptionen, die immer angegeben werden sollten. Durch “-k=Speicherbereich” kann die Größe des Speicherbereiches festgelegt werden. Ist dieser Wert zu groß gewählt, werden die überschüssigen Daten aus dem Hauptspeicher des Computers auf die Festplatte ausgelagert und die Simulationsdauer wird massiv verzögert. Wird dieser Wert allerdings zu klein gewählt, muss die Simulation ihren Speicherbereich sehr oft reorganisieren, um nicht mehr benötigte Daten zu löschen. Dies führt ebenfalls zu einer sehr langsamen Simulation. Die Wahl der Größe des Speicherbereichs ist also entscheidend. Die Angabe erfolgt in “Kilobyte”. Beispiel:

```
steady.hitcode -p -g -k=65536
```

In diesem Beispiel wird eine Simulation gestartet, die 64MB (oder 65536KB) Speicher benutzt.

steady_hit wird im weiteren Verlauf dieser Diplomarbeit benutzt, um eine ausführbare Simulationsdatei zu erzeugen. Diese Datei wird mehrfach kopiert und mit unterschiedlich großem Speicherbereich auf unterschiedlichen Computern gestartet. So können mehrere Simulationen parallel gestartet werden.

7.5 visual

visual ist ein Programm, das Darstellungen erzeugt, wie sie in Kapitel 6.3.2 vorgestellt werden. Die horizontale Achse der Darstellungen beschreibt die Zeit. Die vertikale Achse beschreibt den Wertebereich der beobachteten Größe. Durch eine Skala von Grauwerten wird die relative Häufigkeit deutlich gemacht. Ein dunkler Grauton bedeutet hohe relative Häufigkeit, ein heller Grauton bedeutet niedrige relative Häufigkeit. Die Bereiche mit einer relativen Häufigkeit von Null werden durch einen hellen Blauton gekennzeichnet, um diese Bereiche hervorzuheben. Der Benutzer kann den dargestellten Bereich der relativen Häufigkeit wählen. Alle Werte, die sich unterhalb dieses Bereiches befinden, werden in einem mittleren Blauton dargestellt. Alle Werte, die sich oberhalb dieses Bereiches befinden, werden in einem dunklen Blauton dargestellt.

visual entnimmt die darzustellenden Werte einer Datei. Diese Datei sollte während der Ausführung nicht verändert werden, daher kann *visual* nur auf einem abgeschlossenen Datensatz arbeiten. Dieses Programm ist ein Prototyp für die Darstellungsalgorithmen, die bei dem Programm *ssa* benutzt werden. Liegt jedoch ein abgeschlossener Datensatz vor, so ist die Verwendung von *visual* sinnvoller.

Die Daten der Eingabedatei werden in eine Datenstruktur übertragen, die im Laufe des Programms einen effizienten Zugriff ermöglicht. Durch geeignete Algorithmen wird eine weitere Datenstruktur erzeugt, welche die berechneten relativen Häufigkeiten verwaltet. Im letzten Schritt wird eine Grafikdatei im bekannten BMP-Format erzeugt. Das BMP-Format hat den Vorteil, dass es von allen Grafikprogrammen gelesen werden kann und dass die BMP-Dateien in alle anderen Formate umgewandelt werden können.

7.5.1 Aufruf

Den Programmablauf von *visual* kann der Benutzer mittels Übergabeargumenten steuern. Die Syntax lautet folgendermaßen:

```
PFAD/visual [OPTION] [OPTION] ... INPUTDATEI OUTPUTDATEI
```

Beispiele:

```
visual meineDaten.daten meinBild
visual -b -m kleineDaten.txt kleinesBild
visual -o 0.5 -u 0.4 -q -a 200 meineDaten.erg meinBild
visual --help
```

Bis auf die Angabe des Eingabedateinamens und des Ausgabedateinamens sind alle Übergabeargumente optional. Die Übergabeargumente haben folgende Bedeutung:

- a INTEGER oder -aufloesung INTEGER
Die Aufloesungen der Intervалldarstellungen der Zustandsverteilungen. INTEGER legt die Anzahl der Intervalle fest.
Wertebereich: positive ganze Zahlen
Default fuer - DOUBLE Werte: 100
- INTEGER Werte: MaxWert-MinWert+1
- b oder -bildschirm
Mit dieser Option kann festgelegt werden, ob eine Bildschirmausgabe stattfinden soll. Fuer grosse Datenmengen ist dies nicht zu empfehlen.
- m oder -modellzeit
Wenn die Daten im Eingabefile nach Modellzeitpunkten sortiert sind, muss diese Option gewaehlt sein.

-o DOUBLE oder -obergrenze DOUBLE

Durch diese Option kann die Obergrenze der darzustellenden relativen Haeufigkeiten angegeben werden.

Wertebereich:]0,1] und groesser als die Untergrenze

Default: 1

-p oder -pur

Diese Option unterdrueckt die textuelle Beschriftung des Bildes. Die BMP-Datei enthaelt also nur die eigentlichen Daten.

-r oder -replikation

Wenn die Daten im Eingabefile nach Replikationen sortiert sind, kann diese Option gewaehlt sein.

Sie ist DEFAULT-Einstellung.

-u DOUBLE oder -untergrenze DOUBLE

Durch diese Option kann die Untergrenze der darzustellenden relativen Haeufigkeiten angegeben werden.

Wertebereich: [0,1[und kleiner als die Obergrenze

Default: 0

-q oder -quartil

Dies Option erzwingt die Darstellung der 25%-, 50%- und 75%-Quantilen (bzw. Quartil) durch rote Linien.

--help

Anzeigen der Benutzungshinweise.

Das Format der Eingabedatei ist an das gängige *gnuplot*-Format angelehnt. Die einzelnen Daten werden durch einen Zeilenumbruch getrennt. Datenblöcke (z.B. Zeitreihen) werden durch zwei weitere Zeilenumbrüche getrennt. Das BMP-Ausgabeformat benutzt die 8Bit-Farbdarstellung. Die Ausgabedatei ist unkomprimiert.

Die durch *oszillator* erzeugten Dateien lassen sich direkt mit *visual* darstellen. Auch durch HIT lassen sich Dateien erzeugen, deren Darstellung durch *visual* möglich ist. Hierbei werden jedoch einige Ergänzungen nötig, besonders im Experimentteil des HISLANG-Modells.

7.5.2 HISLANG-Modelle als Datenquelle

Zunächst muss sichergestellt werden, dass eine Ausgabedatei existiert, in welche die Daten der Simulation geschrieben werden können. Diese Ergänzungen werden im Beispielmmodell durch `{*1*}` gekennzeichnet. Die Ausgabe in die Datei erfolgt an den Stellen, die mit `{*2*}` gekennzeichnet sind. Der komplette Experimentteil ist so angelegt, dass mehrere Simulationen gestartet werden, die sich nur durch ihren Startseed unterscheiden. Damit die Simulationen keine Zufallszahl mehrfach benutzen, wird immer mit der zuletzt benutzten Zufallszahl weitergearbeitet. Sollten mehr Zufallszahlen benötigt werden, als in einer Periode des Zufallszahlengenerators vorhanden sind, versagt dieses Vorgehen. Der hinzugefügte HISLANG-Service *P97Ausgabe* sorgt dafür, dass zu dem richtigen Modellzeitpunkt eine Ausgabe in die Ergebnisdatei erfolgt.

```
%COMMON
%PARM = WARN
%BIND "OUTDatei" TO EXTEND erg_server.txt {*1*}
%END
%COPY "counter"
%COPY "semaphor"
%COPY "observer"

TYPE voidRecordName RECORD; END TYPE voidRecordName;
TYPE Flserver01_Typ MODEL(ausgabefile:OUTFILE); {*1*}
  COMPONENT F23Bediener : server
    (LET ACCEPT := ALWAYS,
     LET SCHEDULE := FCFS (LET CAPACITY := 1),
     LET DISPATCH := EQUAL (LET SPEED := 1.0),
     LET OFFER := ALL);

  TYPE REC5_ProzessAufruf RECORD;
  END {RECORD} TYPE REC5_ProzessAufruf;

  TYPE P5ProzessAufruf SERVICE(data : POINTER FOR REC5_ProzessAufruf);
  USE
    SERVICE L8Aufruf(B2H___amount : REAL DEFAULT 1.0);
  END USE;

  BEGIN
    NEW REC5_ProzessAufruf POINTER data;
    L8Aufruf(negexp(1/0.5));
  END {SERVICE} TYPE P5ProzessAufruf;

  TYPE REC97_Ausgabe RECORD;
  VARIABLE
    serverpop : INTEGER DEFAULT 0;
  END {RECORD} TYPE REC97_Ausgabe;

  TYPE P97Ausgabe SERVICE(data : POINTER FOR REC97_Ausgabe);
  USE
    PROCEDURE L100Anfrage RESULT INTEGER;
  END USE;

  BEGIN
    NEW REC97_Ausgabe POINTER data;
    data.serverpop := L100Anfrage;
    WRITELN FILE ausgabefile, data.serverpop; {*2*}
  END {SERVICE} TYPE P97Ausgabe;

  TYPE EveryUnbedQuellen SERVICE;
  BEGIN
    CONCURRENT
      LOOP
        hold(negexp(1.0));
        CREATE 1 PROCESS P5ProzessAufruf(NONE);
      END LOOP;
    TO
      LOOP
        hold(1);
        CREATE 1 PROCESS P97Ausgabe(NONE);
      END LOOP;
    END CONCURRENT;
  END {SERVICE} TYPE EveryUnbedQuellen;
```

```

REFER
  P5ProzessAufruf, P97Ausgabe TO F23Bediener
EQUATING
  P5ProzessAufruf.L8Aufruf WITH F23Bediener.request;
  P97Ausgabe.L100Anfrage WITH F23Bediener.popul;
END REFER;
BEGIN
  CREATE 1 PROCESS EveryUnbedQuellen;
END {MODEL} TYPE Flserver01_Typ;

EXPERIMENT versuch METHOD SIMULATIVE;
  VARIABLE wiederholung : INTEGER;
           stopen       : INTEGER;
           zeilenbruch  : OUTFILE;
BEGIN
  OPEN zeilenbruch, "OUTDatei" LENGTH 80; { *1* }
  stopen := 500;
  FOR wiederholung := 1 STEP 1 UNTIL 1000
  LOOP
    WRITELN "Wiederholung: ", wiederholung;
    EVALUATE
      MODEL Flserver01:Flserver01_Typ(zeilenbruch,LET SEED:=last_seed);
      EVALUATIONOBJECT messgeraet VIA Flserver01;
    BEGIN
      MEASURE THROUGHPUT AT messgeraet;
      CONTROL AT messgeraet STOP MODELTIME stopen;
    END EVALUATE;
    WRITELN FILE zeilenbruch; { *2* }
    WRITELN FILE zeilenbruch; { *2* }
  END LOOP;
  CLOSE zeilenbruch; { *1* }
END EXPERIMENT versuch;

```

Dieses HISLANG-Modell kann als Vorlage dienen, um beliebige HISLANG-Modelle manuell anzupassen. Durch eine Auswertung mit HIT wird gleichzeitig ein Datenfile erzeugt, das mit *visual* bearbeitet werden kann. Die Abbildung auf dem Titelblatt dieser Diplomarbeit wurde auf diesem Weg erzeugt.

7.6 seed

Das Programm *seed* berechnet die *i*te Zufallszahl einer speziellen Klasse von Zufallszahlengeneratoren. Diese Zufallszahlengeneratoren wurden bereits in Kapitel 2.3.3 vorgestellt. Bei der Analyse einiger Modelle war es notwendig, mehrere Simulationsläufe mit dem Simulationstool HIT durchzuführen. Da die Simulationsläufe möglichst unabhängig voneinander durchgeführt werden sollten, wurde schon in Kapitel 2.3.4 ein Verfahren vorgestellt, dass die maximale mögliche Unabhängigkeit gewährleistet. Mit Hilfe von *seed* können die Startseeds der Simulationen bestimmt werden.

Das Programm *seed* erwartet drei Übergabeargumente:

1. *i* - die *i*te Position der Zufallszahlenperiode
2. Z_0 - die erste Zahl der Zufallszahlenperiode
3. *a* - der Parameter des Zufallszahlengenerators

Das Ergebnis ist der Wert von Z_i . Er wird über den Standard-Ausgabekanal übergeben.

7.7 oszillator

Das Programm *oszillator* berechnet die Iterationsgleichung

$$x_{i+1} = p \cdot x_i \cdot (1 - x_i) \quad . \quad (7.6)$$

Der Parameter p durchläuft ein vom Benutzer vorgegebenes Intervall mit einer bestimmten Schrittweite. Für jeden Wert des Parameters p wird eine gesonderte Berechnung der Iterationswerte x_1, x_2, \dots, x_n durchgeführt. Alle Iterationen benutzen denselben Startwert x_0 und dieselbe Iterationstiefe n . Die Ergebnisse werden in einer Datei gespeichert.

Das Programm *oszillator* lieferte die Datenbasis für die Abbildungen 5.1, 5.2, 5.3, 5.4 und 5.5 aus Kapitel 5. Des Weiteren diente es dazu, große Datenmengen zu erzeugen, um andere Programme zu testen. In diesen Fällen wurden die Iterationswerte als die Messwerte einer Simulation interpretiert. Im Allgemeinen ist die Laufzeit einer Simulation recht lange, daher hat die Verwendung des Programmes *oszillator* praktische Vorteile. Im Kapitel 5 wird dargestellt, dass auch die Werte des Oszillators eine transiente und eine stationäre Phase besitzen können.

Das Programm *oszillator* wurde so implementiert, dass es möglichst flexibel und bedienungsfreundlich ist. So kann jeder Parameter direkt vom Benutzer festgelegt werden. Jeder Parameter hat einen geeigneten Vorgabewert, damit der Benutzer nicht jeden Parameter festlegen muss. Der Aufruf des Programms *oszillator* hat folgende Syntax:

```
PFAD/oszillator [OPTION] [OPTION] ... -d DATEINAME
```

Beispiel:

```
oszillator -a 1000 -s 3.4 -e 3.9 -t 10000 -d meineDaten
```

Über die im Folgenden beschriebenen Optionen können die Parameter festgelegt werden. Bis auf die Angabe des Dateinamens sind alle Angaben optional. Die angegebene Datei enthält nach der Programmausführung alle Ergebniswerte.

-a INTEGER

Mit INTEGER wird die (A)nzahl der Berechnungsschritte zwischen dem Startpunkt und dem Endpunkt festgelegt.
Wertebereich: positive ganze Zahlen und Null
Default: 500

-b INTEGER

Mit INTEGER wird festgelegt, ob und welche (B)ildschirmAusgabe stattfinden soll.
Wertebereich: 0-keine BildschirmAusgabe

```
1-Berechnungsschritt
2-Jeder Wert und Berechnungsschritt

Default: 0
-d STRING
  Mit STRING wird der (D)ateiname definiert. Die
  Dateierweiterung wird automatisch angefügt.
-e DOUBLE
  Der (E)ndpunkt wird durch DOUBLE festgelegt.
  Wertebereich: [0;4] und >Startpunkt
  Default: 4.0
-o DOUBLE
  DOUBLE ist der (O)ffset, mit dem jede Iteration
  gestartet wird.
  Wertebereich: [0;1]
  Default: 0.5
-s DOUBLE
  Der (S)tartpunkt wird durch DOUBLE festgelegt.
  Wertebereich: [0;4] und <Endpunkt
  Default: 0.0
-t INTEGER
  Die (I)terationstiefe wird durch INTEGER festgelegt.
  Wertebereich: positive ganze Zahlen
  Default: 1000
--help
  Anzeigen der Benutzungshinweise.
```

Das Format der Ergebnisdatei ist so angelegt, dass es für eine automatische Weiterverarbeitung geeignet ist. Alle Iterationswerte sind durch einen Zeilenumbruch voneinander getrennt. Die Iterationsläufe werden durch drei Zeilenumbrüche voneinander getrennt. Die Reihenfolge der Ergebnisse entspricht der Berechnungsreihenfolge. Dieses Format hat den Vorteil, dass gängige Darstellungstools (zum Beispiel *gnuplot*) die Ergebnisdatei als Eingabedatei ohne weitere Bearbeitungsschritte benutzen können.

Kapitel 8

Testmodelle

In diesem Kapitel wird die erweiterte Analyse an ausgewählten Beispielen demonstriert und getestet. Die transiente Phase wird ermittelt und bei der weiteren Analyse nicht berücksichtigt. Der Unterschied zu dem Standard-Analyseverfahren wird in einigen Diagrammen deutlich gemacht. Das Analysebeispiel des Güterverkehrszentrums (siehe Kapitel 1.1) und des schlecht initialisierten M/M/1-Servers (siehe Kapitel 4.4) werden erneut aufgegriffen. An ihnen sollen die Vorteile des erweiterten Analyseverfahrens veranschaulicht werden.

In den folgenden Ergebnisauswertungen wird oft auf die von der Simulation benötigte Modellzeit Bezug genommen. Die Modellzeit steht hier stellvertretend für die benötigte Menge an Beobachtungsdaten. Zwischen der Modellzeit und der Menge an Beobachtungsdaten besteht ein linearer Zusammenhang.

Die CPU-Zeit sei die tatsächliche Zeit, in der die Auswertungsmethoden und die Simulationen den Prozessor belegen. Ist ein Modell stabil, sind in grober Näherung in jedem gleichgroßen Modellzeitintervall gleichviele Kundenankünfte (bzw. "jobs") zu bearbeiten. Dies bedeutet, dass gleichgroße Modellzeitintervalle ungefähr gleichviel CPU-Zeit verbrauchen. Aus diesem Grund kann angenommen werden, dass bei stabilen Modellen der Unterschied zwischen der Modellzeit und der CPU-Zeit ein konstanter Faktor ist. Diese Annahme gilt auch zwischen unterschiedlichen Simulationsläufen mit demselben Modell. Weniger benötigte Modellzeit führt daher zu einer kürzeren benötigten CPU-Zeit und damit zu einer schnelleren Analyse in der realen Zeit.

8.1 Güterverkehrszentrum

In der Einleitung wurde schon auf die Auswertungsschwierigkeiten bei der Analyse des Güterverkehrszentrums hingewiesen (siehe Kapitel 1.1). Das große Problem ist, dass der konstante Bereich der Population zu Beginn der Simulation mit der stationären Phase verwechselt werden kann. Das Modell ist nicht stabil,

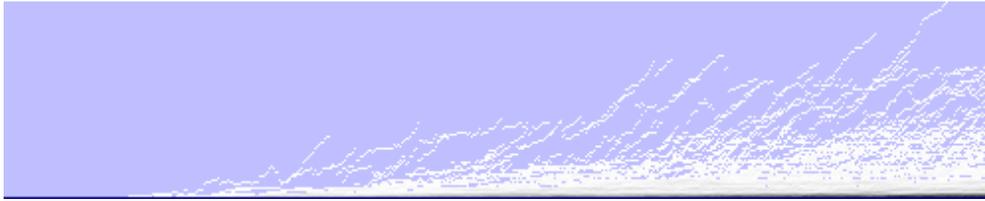


Abbildung 8.1: Das Zufallsverteilungsdiagramm der Kunden im Lager

wie schon in [3] und in [6] bewiesen wurde. Daher führt eine Verwechslung zu gänzlich falschen Annahmen.

8.1.1 Test auf Stationarität

Das Programm *ssa* benutzt die Stationaritätsbedingung, dass die Ankunftsrate und die Abgangsrate annähernd gleich sein müssen. Ist dies nicht der Fall, wächst die Anzahl der Kunden im System an und das Modell kann nicht stabil sein. Die Abbildung 8.2 zeigt den zeitlichen Verlauf der genormten Differenz der Ankunftsrate und der Abgangsrate. Die Differenz steigt immer weiter an. Dies ist ein erster Hinweis darauf, dass das Güterverkehrszentrum kein stabiles Modell ist. Jedoch ist der Betrag der Differenz bei der Modellzeit 100 noch so gering, dass eine Abweichungsgrenze von 5% noch unterschritten wird.

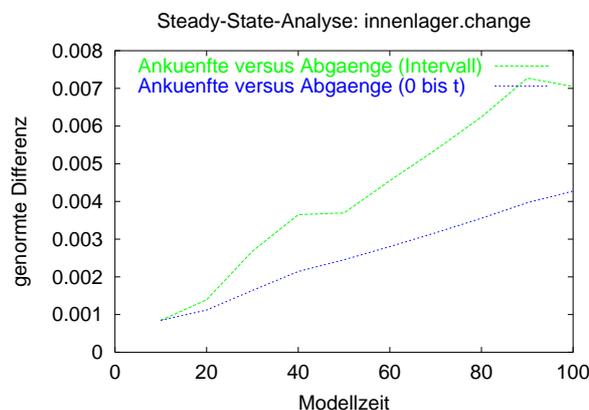


Abbildung 8.2: Genormte Differenz der Ankunftsrate und Abgangsrate.

Die Zufallsverteilungsmethode basiert auf dem Vergleich einer ausgewählten Stichprobe mit den nachfolgenden Stichproben. Die Anzahl der nachfolgenden Stichproben kann konstant sein oder im Verhältnis zu den bereits verworfenen Stichproben stehen (siehe Kapitel 6.3.2). Egal welche Strategie in diesem Fall bevorzugt wird, entscheidend ist, dass die Anzahl der nachfolgenden Stichproben sehr groß gewählt wird. Ist diese Voraussetzung erfüllt, erkennt die Zufallsver-

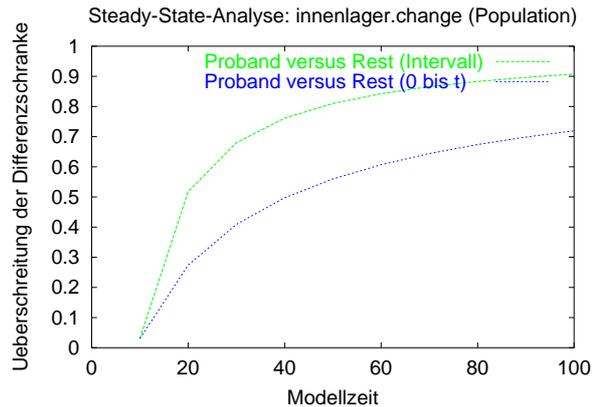


Abbildung 8.3: Vergleich der ausgewählten Stichprobe mit dem Rest in dem Verhältnis 1:199. (Population)

teilungsmethode, dass kein stationäres Modell vorliegt. Abbildung 8.3 zeigt den zeitlichen Verlauf des Testergebnisses. Dieser Verlauf steigt steil an und daher werden die Unterschiede der Zufallsverteilungen immer größer. Dieser Testkurve liegt eine Simulation zugrunde, bei der das Verhältnis der Datenmenge der transienten und der stationären Phase auf 1:199 festgelegt wird. Die zuletzt betrachtete Stichprobe (der Proband) liegt bei der Modellzeit 100. Auf Grund der Wahl des Verhältnisses wird insgesamt allerdings bis zur Modellzeit 20000 simuliert.

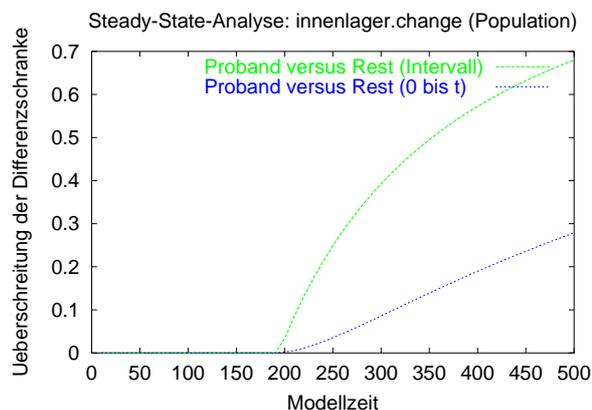


Abbildung 8.4: Vergleich der ausgewählten Stichprobe mit dem Rest in dem Verhältnis 1:49. (Population)

Die Abbildung 8.4 zeigt den Testverlauf der Zufallsverteilungsmethode mit einer Simulation im transient-stationären Verhältnis von 1:49. Es ist ebenfalls ein steiler Anstieg der Testkurve zu erkennen. Bis zur Modellzeit 190 zeigen die Tests jedoch keinen Unterschied. Ein Verhältnis von 1:49 ist daher noch zu klein, da die

Zufallsverteilungsmethode in diesem Beispiel bis zur Modellzeit 190 annimmt, dass ein stabiles Modell vorliegt.

8.1.2 Bewertung

Der Verlauf der Differenz der Ankunftsrate und der Abgangsrate steigt deutlich an. Dies ist ein starkes Indiz für ein nicht stabiles Modell. *ssa* betrachtet nicht den Verlauf der Differenz, sondern nur den tatsächlichen Betrag. In diesem Fall ist der Betrag jedoch so niedrig, dass *ssa* dieses Indiz für ein nicht stabiles Modell übersieht.

Die Zufallsverteilungsmethode erkennt, dass die Zufallsverteilungen nicht konstant sind und somit ein instabiles Modell vorliegt. Dies ist ein großer Fortschritt gegenüber den Standard-Analyseverfahren. Problematisch ist jedoch, dass der Erfolg der Zufallsverteilungsmethode so extrem von den gewählten Parametern abhängt. Vom Benutzer wird ein Modellverständnis vorausgesetzt, das er eigentlich erst durch die Analyse erhalten kann. Bei geeigneten Parametern der Zufallsverteilungsmethode liefert die erweiterte Analyse korrekte Ergebnisse und ist damit eine echte Verbesserung der Standard-Analyse.

8.1.3 Lagerbestand

Ein weiteres wichtiges Leistungsmaß in diesem Modell ist der Lagerbestand. Da das Modell zu Beginn der Simulation genau zur Hälfte gefüllt ist, hat die Zufallsverteilung des Lagers zu Beginn einen sehr scharfen Peak in der Mitte der Verteilung. Die einzelnen Simulationen verhalten sich wie ein "Random-Walk" (siehe [31], Kapitel 2.1). Da sie bei einem Simulationsschritt immer zufällig von dem aktuellen Zustand (Bestand) sowohl positiv als auch negativ abweichen, verwischt dieser Peak im Laufe der Simulationen. Nach einer gewissen Zeit ist die Wahrscheinlichkeit groß, dass die Simulationen den Rand des Lagerbestandes erreichen. Sie können nicht über den Rand hinaus gelangen. Die Wahrscheinlichkeit, von dem aktuellen Zustand (Bestand) positiv oder negativ abzuweichen, ist gleich groß. Daher werden nun einige Prozesse blockiert. Nach einer sehr langen Simulation werden fast alle Simulationsläufe am Rand des Lagerbestandes liegen. Die Zufallsverteilung des Lagerbestandes wird daher an den beiden Rändern einen hohen Peak haben, während die Aufenthaltswahrscheinlichkeit in der Mitte sehr gering wird.

Die Anzahl der Kunden im Lager ist in der Anfangsphase konstant, da alle Simulationen eine bestimmte Zeit brauchen, bis sie in den Randbereich des Lagerbestandes gelangen. Erst nach dieser Phase werden einige Kunden blockiert und die Population wächst. Bei der Betrachtung des Lagerbestandes wird deutlich,



Abbildung 8.5: Das Zufallsverteilungsdiagramm des Lagerbestandes

warum das Güterverkehrszentrum nicht stabil ist und warum die Population in der Anfangsphase konstant ist.

Eine weitere Besonderheit ist, dass der Mittelwert des Lagerbestandes im Laufe der Simulationen immer gleich ist. Aus diesem Grund versagen die statistischen Methoden, die auf Basis des Mittelwertes eines Leistungsmaßes die stationäre Phase erkennen. Die Zufallsverteilungsmethode hat hierdurch keine weiteren Probleme.

Da der Lagerbestand beschränkt ist und der Mittelwert im Laufe der Zeit immer konstant bleibt, wird ein festes Konfidenzintervall im Laufe der Simulation immer enger um den Mittelwert liegen. Eine Analyse, die nur den Mittelwert betrachtet und abbricht, wenn das 95%-Konfidenzintervall nur noch 5% vom Mittelwert abweicht, würde bei der Betrachtung des Lagerbestandes zu einem Ergebnis führen. Dieses Ergebnis wird aber zu einer Fehlinterpretation führen, wenn der Charakter der Zufallsverteilungen des Lagerbestandes nicht bekannt ist.

8.2 M/M/1-Server: schlecht initialisiert

In Kapitel 4.4 wurde schon das Modell eines M/M/1-Servers vorgestellt, das zu Beginn mit 100 Aufträgen belastet wird. Dies ist ein "untypischer" Startzustand. Die theoretische Betrachtung ergibt, dass im Durchschnitt genau ein Kunde im System ist. Es wurde bereits deutlich, welche Auswirkungen dieser Startzustand hat und welche Schwierigkeiten bei der Auswertung durch das Standard-Analyseverfahren auftreten.

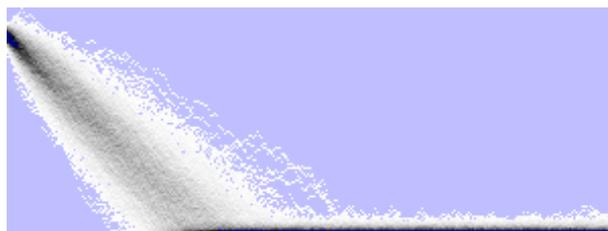


Abbildung 8.6: Das Zufallsverteilungsdiagramm der Population.

Um deutlich zu machen, wie sich das Leistungsmaß Population im Laufe der Simulation verhält, wird mit dem Software-Tool *visual* das Zufallsverteilungsdiagramm der Population entworfen. Die Abbildung 8.6 wird aus 1000 Simulationenläufen erstellt. Die transiente Anfangsphase ist deutlich zu erkennen. Nachdem der anfängliche Stau abgearbeitet wurde, verhält sich das Modell sehr stabil. Die Zufallsverteilungen in der rechten Hälfte des Diagramms wirken konstant.

In den folgenden Abschnitten sollen Analysen durchgeführt werden, die berücksichtigen, dass die Daten der transienten Anfangsphase nicht für eine Analyse des stationären Verhaltens geeignet sind.

8.2.1 Zufallsverteilungsmethode und direkte Analyse

Die Zufallsverteilungsmethode basiert darauf, dass während der stationären Phase die Zufallsverteilungen der betrachteten Leistungsmaße konstant sind. In dem

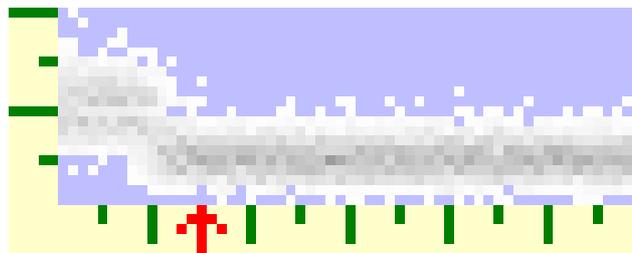


Abbildung 8.7: Das Zufallsverteilungsdiagramm der Abgangsrate.

Modell des schlecht initialisierten M/M/1-Servers ist die Auswertung der Ankunftsrate nicht sehr wichtig, da ihre Zufallsverteilung nur in der ersten Modellzeiteinheit nicht konstant ist. Dies beruht auf den 100 Initialisierungsankünften zu Beginn der Simulation. Für die Leistungsmaße Abgangsrate und Population ist dies nicht der Fall. An diesen Leistungsmaßen lässt sich die transiente Phase

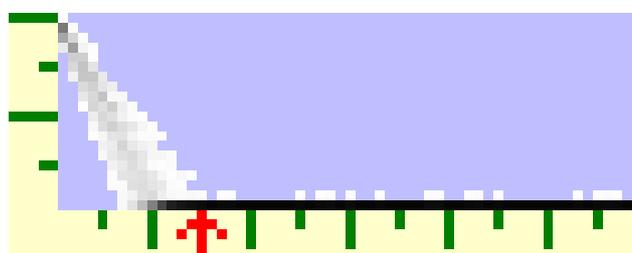


Abbildung 8.8: Das Zufallsverteilungsdiagramm der Population.

eindeutig erkennen. Dies zeigen die Abbildungen 8.7 und 8.8. Sie werden durch das Analysetool *ssa* entworfen und basieren auf 100 parallelen Simulationen.

Mit der Zufallsverteilungsmethode erkennt ssa den Beginn der stationären Phase zu dem Modellzeitpunkt 160. Die Abbildungen 8.9 und 8.10 bestätigen diese Annahme. Aufgetragen ist jeweils der Mittelwert der einzelnen Intervalle.

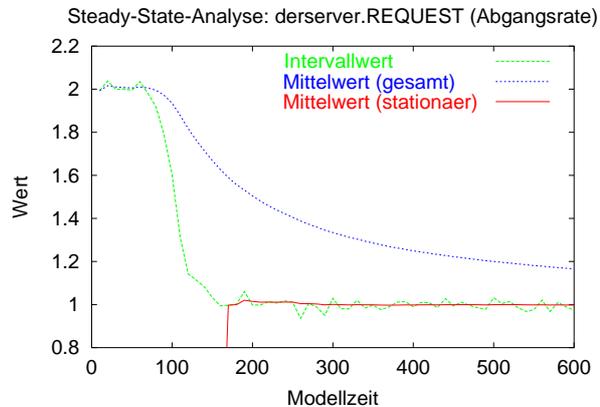


Abbildung 8.9: Stationärer Mittelwert versus gesamter Mittelwert (Abgangsrate).

Die beiden anderen Kurven zeigen den Mittelwert von den Intervallmittelwerten aller Daten und den Mittelwert von den Intervallmittelwerten der stationären Phase. Ab dem Modellzeitpunkt 160 wirken die Intervallmittelwerte konstant, sodass auch der Mittelwert der stationären Phase konstant wirkt. Er bildet in beiden Diagrammen nahezu eine Gerade ohne Steigung und bestätigt dadurch die richtige Wahl des Beginns der stationäre Phase.

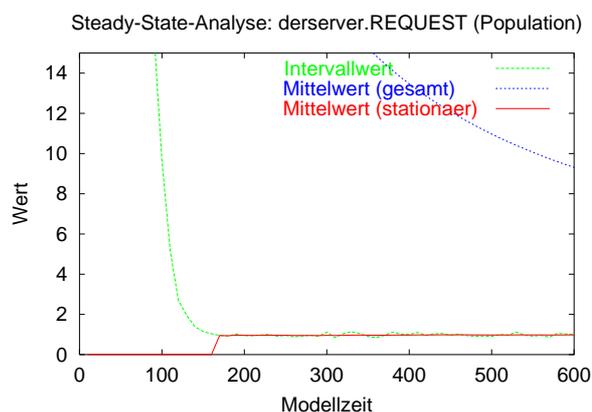


Abbildung 8.10: Stationärer Mittelwert versus gesamter Mittelwert (Population).

Bei der Zufallsverteilungsmethode wird eine ausgewählte Stichprobe mit den nachfolgenden Stichproben verglichen. Die zeitlichen Verläufe dieser Vergleiche sind in Abbildung 8.11 und 8.12 beschrieben. Es ist deutlich zu erkennen, dass beide Kurven bis zum Modellzeitpunkt 160 steil abfallen. Dieses Abfallen ist bedingt durch die stationäre Phase. Die Kurve der Abgangsrate hat eine weitere Be-

sonderheit: Bis zur Modellzeit 80 liegt die Abgangsrate relativ konstant bei zwei (siehe Abbildung 8.9). Dieses konstante Verhalten bewirkt, dass die Kurve in Abbildung 8.11 auf einem niedrigen Niveau startet, da zu Beginn nur Vergleiche zwischen Stichproben der transienten Phase stattfinden. Sobald zum Vergleich einige Stichproben der stationären Phase genutzt werden, steigt die Kurve an.

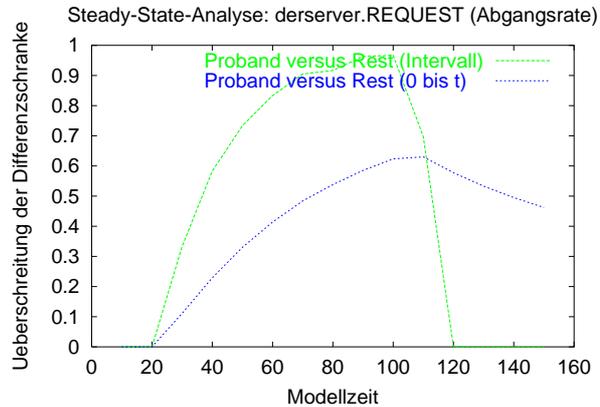


Abbildung 8.11: Vergleich der ausgewählten Stichprobe mit dem Rest. (Abgangsrate)

Der Vergleich der Ankunftsrate mit der Abgangsrate verhält sich wie erwartet (siehe Abbildung 8.13). Zu Beginn ist die genormte Differenz auf einem hohen Niveau. Im Laufe der Simulation sinkt die Differenz dann bis auf ein Niveau nahe bei Null. Auch dieses Diagramm bestätigt, dass die stationäre Phase bei der Modellzeit 160 beginnt.

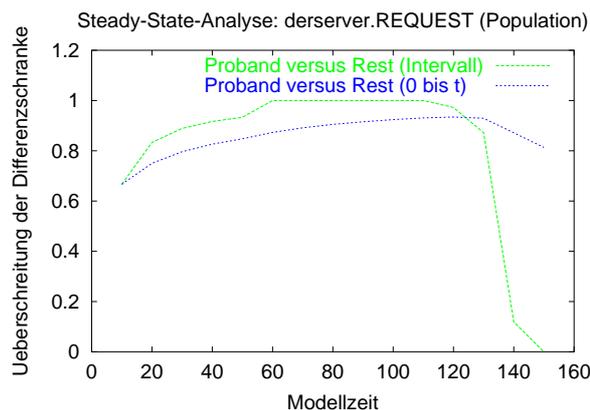


Abbildung 8.12: Vergleich der ausgewählten Stichprobe mit dem Rest (Population).

An den Abbildungen 8.9 und 8.10 wird der Unterschied zwischen dem Mittelwert aller Daten und dem Mittelwert der Daten der stationären Phase deutlich.

Dieser große Unterschied bewirkt die lange Simulationsdauer bei einer Analyse ohne Verwerfen der Daten der transienten Phase. Da in *ssa* nur die Daten der stationären Phase berücksichtigt werden, ist eine wesentlich kürzere Simulationszeit zu erwarten.

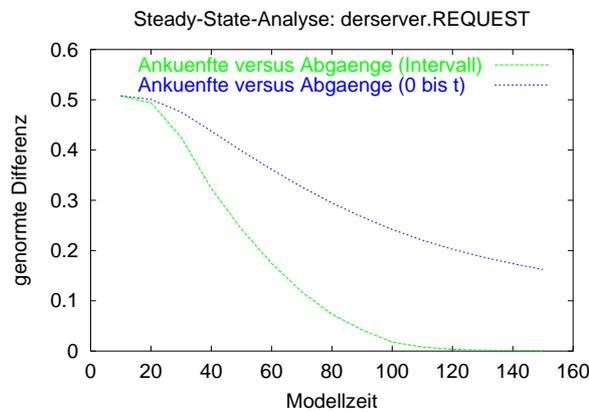


Abbildung 8.13: Genormte Differenz der Ankunftsrate und Abgangsrate.

Die Auswertung der Daten der stationären Phase erfolgt durch die direkte Analyse ohne Reduktion auf eine stellvertretende Messreihe. Die berechneten Mittelwerte der einzelnen Simulationen können als unabhängig betrachtet werden (siehe Kapitel 6.3.3). Nach 600 Modellzeiteinheiten hat das 95%-Konfidenzintervall für die Population nur noch 3.2% der Größe des Mittelwertes. Der Mittelwert der Population kann auf 1.00086 festgelegt werden. Der Betrag des 95%-Konfidenzintervalls ist 0.0322425. Der theoretische Erwartungswert liegt eindeutig im Bereich des 95%-Konfidenzintervalls. Die Differenz zwischen dem theoretischen Erwartungswert und dem berechneten Mittelwert beträgt ungefähr 0.09%. Im Vergleich zu der Standard-Analyse ist die Qualität des Ergebnisses viel besser. Zu berücksichtigen ist noch, dass nur 600 Modellzeiteinheiten benötigt werden. Unter Berücksichtigung der 100 parallelen Simulationsläufe ist ersichtlich, dass weniger Simulationsdaten erforderlich sind, als bei dem Standard-Analyseverfahren aus Kapitel 4.4. Da die Simulationsdaten durch parallele Simulationsläufe berechnet werden, ist auch die Simulationsdauer viel geringer.

8.2.2 Balance-Regel und Batch Means für viele Simulationen

Bei diesem Experiment werden acht parallele Simulationen durchgeführt, die im Verlauf der Analyse durch Mittelwertberechnung auf eine Messreihe reduziert werden. Die Balance-Regel basiert auf dem Vergleich zwischen der Anzahl der Werte, die über bzw. unter dem Mittelwert liegen. Mit Hilfe dieser Regel kann *ssa* den Beginn der stationären Phase auf die Modellzeit 120 festlegen.

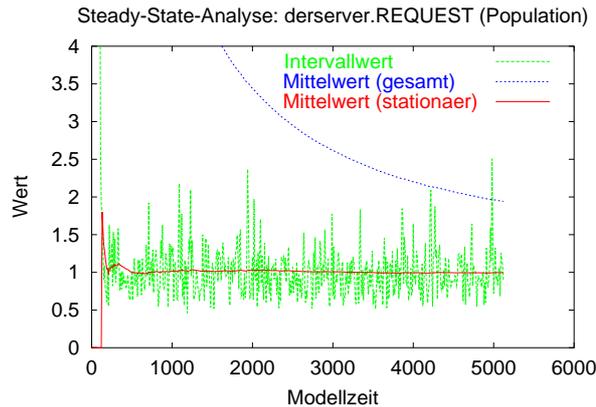


Abbildung 8.14: Stationärer Mittelwert versus gesamter Mittelwert (Population).

Die Abbildungen 8.14 und 8.15 zeigen, dass auch der Balance-Test die stationäre Phase richtig erkennt. Der Beginn wird nur etwas zu früh (Modellzeit) festgestellt. Hierdurch wird der Einfluss des Startzustandes nicht ganz beseitigt und eine längere Simulation ist zu erwarten. Da bei dieser Analyse nur acht parallele Simulationen durchgeführt werden, schwankt die Kurve der Intervallmittelwerte stärker als bei der Analyse mit 100 parallelen Simulationen.

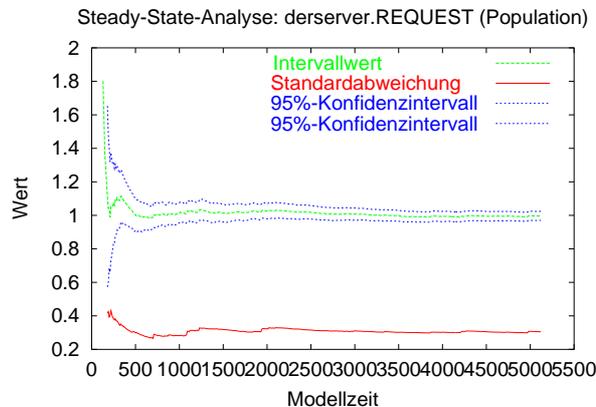


Abbildung 8.15: Die stationäre Phase der Population.

Mit der Methode “Batch Means” werden die Daten nun ausgewertet. Bei einer Gruppengröße von 14 Daten ergibt der Unabhängigkeitstest ein akzeptables Ergebnis. Nach 5120 Modellzeiteinheiten hat das 95%-Konfidenzintervall für die Population weniger als 5% der Größe des Mittelwertes und beträgt 0.0324191. “Batch Means” berechnet den Mittelwert 0.997219 für die Population. Die Differenz zum theoretischen Wert liegt ungefähr bei 0.28%.

Die Balance-Methode kombiniert mit “Batch Means” liefert Ergebnisse, die qualitativ hochwertiger sind, als die Ergebnisse der Standard-Analyseverfahren.

Die Simulationszeit kann durch die Balance-Methode wesentlich verkürzt werden.

8.2.3 Spaced Batch Means für viele Simulationen

Bei diesem Analysebeispiel wird der Beginn der stationären Phase auf dieselbe Weise wie in Kapitel 8.2.2 bestimmt. Da derselbe Startseed verwendet wird, ergibt sich auch hier der Beginn der stationären Phase bei Modellzeit 120. Auch hier bilden acht parallele Simulationen die Auswertungsgrundlage.

Das in *ssa* implementierte “Spaced Batch Means” benutzt aus jeder gebildeten Gruppe nur einen Messwert. Daher ist die Berechnung eines Mittelwertes überflüssig und die Folge der unabhängigen Daten besteht aus einer Auswahl der Originaldaten. Bei der Gruppengröße von zwölf Daten ergibt der Unabhängigkeitstest ein akzeptables Ergebnis. Für die Population kann erst nach 18320 Modellzeiteinheiten ein 95%-Konfidenzintervall bestimmt werden, dass weniger als 5% vom Mittelwert abweicht. Es beträgt 0.0496998. Der Mittelwert wird durch “Spaced Batch Means” auf 1.01587 festgelegt. Die Differenz zum theoretischen Mittelwert ist in diesem Fall 1.59%.

Die Auswertung durch “Spaced Batch Means” liefert in diesem Auswertungsbeispiel Ergebnisse, die qualitativ schlechter sind als die Ergebnisse von “Batch Means”. “Spaced Batch Means” benötigt mehr als drei mal so lange Simulationen wie “Batch Means”. Ein Vorteil von “Spaced Batch Means” ist jedoch, dass die Gruppengröße mit zwölf Daten kleiner ist als die Gruppengröße bei “Batch Means”. In diesem Beispiel ist “Batch Means” der bessere Auswertungsalgorithmus. Für eine Verallgemeinerung dieser Aussage reichen diese Ergebnisse nicht aus.

8.2.4 Balance-Regel und Batch Means für eine Simulation

Bei diesem Auswertungsbeispiel wird erneut die “Balance-Regel” zur Bestimmung des Beginns der stationären Phase verwendet. Der Beginn der stationären Phase kann auf die Modellzeit 200 festgelegt werden. Anschließend erfolgt eine Analyse durch “Batch Means”. Im Unterschied zu Kapitel 8.2.2 liefert bei dieser Auswertung nur eine Simulation die Datengrundlage.

Die Abbildung 8.16 zeigt, dass die Varianz der Intervallmittelwerte der Population bei nur einer Simulation sehr groß ist. Der Unterschied zwischen dem Mittelwert der Daten der stationären Phase und dem Mittelwert aller Daten wird dennoch deutlich. Abgesehen von der Anfangsphase bildet der stationäre Mittelwert schnell eine Gerade ohne Steigung, während sich der Gesamt-Mittelwert nur sehr langsam dem theoretischen Mittelwert nähert. Abbildung 8.17 verdeut-

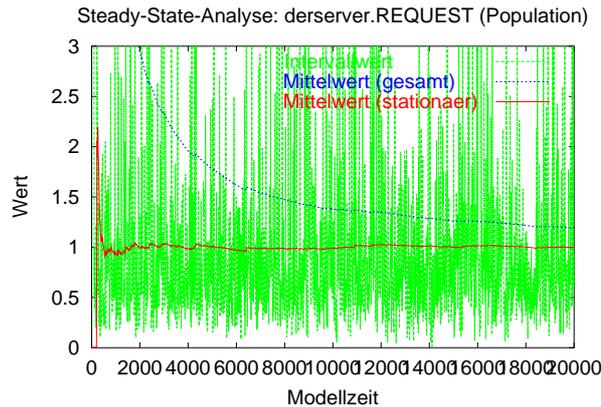


Abbildung 8.16: Stationärer Mittelwert versus gesamter Mittelwert (Population).

licht, dass sich der stationäre Mittelwert schnell auf den theoretischen Mittelwert einpendelt. Nach 20400 Modellzeiteinheiten hat das 95%-Konfidenzintervall den Wert 0.0500609. Der Mittelwert kann auf 1.00208 festgelegt werden. Damit liegt der theoretische Mittelwert im Bereich des Konfidenzintervalls. Obwohl bei dieser Analyse nur ein Simulationslauf durchgeführt wird, kann die Simulationsdauer im Vergleich zur Standard-Analyse sehr stark verkürzt werden.

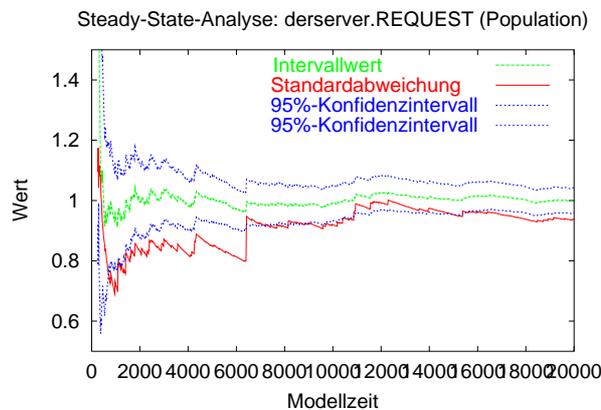


Abbildung 8.17: Die stationäre Phase der Population.

8.2.5 Ergebnisüberblick

Die Tabelle 8.18 fasst die Auswertungsergebnisse aus Kapitel 4.4 und Kapitel 8.2 zusammen. Alle Ergebnisse wurden mit demselben Modell ermittelt. Die Simulationsläufe wurden abgebrochen, als der Betrag des 95%-Konfidenzintervalls nicht mehr größer als 5% des Mittelwertes war. Die erste Zeile enthält die Mittelwerte der acht Standard-Analysen (siehe Tabelle 4.6). Die zweite Zeile zeigt die Ergebnisse der Standard-Analyse mit der kürzesten Modellzeit und die dritte Zeile

| Analysemethode | Modellzeit | k | n_0 | Ergebnis | Abweichung |
|------------------------|------------|-----|-------|----------|------------|
| Standard (Mittelwert) | 223740 | 1 | | 1.02687 | 2.69% |
| Standard (beste MZ) | 157909 | 1 | | 1.03246 | 3.25% |
| Standard (bestes Erg.) | 263975 | 1 | | 1.01114 | 1.11% |
| Balance / Batch Means | 20400 | 1 | 200 | 1.00208 | 0.21% |
| Balance / Batch Means | 5120 | 8 | 120 | 0.99722 | 0.28% |
| Balance / Spaced BM | 18320 | 8 | 120 | 1.01587 | 1.59% |
| ZVM / direkte Analyse | 600 | 100 | 160 | 1.00086 | 0.09% |

Abbildung 8.18: Überblick über die Auswertungsergebnisse des schlecht initialisierten M/M/1-Servers.

enthält die Ergebnisse der Standard-Analyse mit dem qualitativ besten Ergebnis. Die vier folgenden Zeilen beinhalten die Ergebnisse der erweiterten Analysen aus Kapitel 8.2.1 bis Kapitel 8.2.4. In der Spalte k ist die Anzahl der parallelen Simulationen aufgeführt. Die Spalte n_0 zeigt den Beginn der stationären Phase in Modellzeiteinheiten.

Es ist eindeutig zu erkennen, dass die erweiterten Analyseverfahren weniger Modellzeiteinheiten benötigen als die Standard-Analyseverfahren. Auch unter Berücksichtigung der Anzahl der parallelen Simulationen trifft diese Aussage noch zu. Abgesehen von der Auswertung durch Spaced Batch Mean liefern die erweiterten Analyseverfahren wesentlich bessere Ergebnisse. Dies zeigt die Spalte "Abweichung". Aufgeführt ist hier die prozentuale Abweichung vom theoretischen Wert.

Die letzte Zeile der Tabelle zeigt die Ergebnisse der Analyse aus Kapitel 8.2.1. Mit dieser Auswertungsstrategie konnte das qualitativ beste Ergebnis erreicht werden. Auch die benötigte Modellzeit ist um mehrere Größenordnungen kleiner als bei der Standard-Analyse. Wird die benötigte Modellzeit mit der Anzahl der parallelen Simulationen multipliziert, ist zu erkennen, dass nur $\frac{1}{3}$ bis $\frac{1}{4}$ der Datenmenge der Standard-Analysen benötigt wurde.

8.3 M/M/1-Server: normal ausgelastet

Ein M/M/1-Server wird mit der Verkehrsintensität $\rho = 0.8$ betrachtet. Die Simulationen beginnen in dem Systemzustand "empty und idle". Die theoretische Analyse ergibt, dass im Durchschnitt vier Kunden im System sein werden:

$$E[n] = \frac{\rho}{1 - \rho} = \frac{0.8}{1 - 0.8} = 4 \quad . \quad (8.1)$$

Der Einfluss der Startzustandes "empty und idle" reicht bei einem nicht überlasteten M/M/1-Server theoretisch bis ins Unendliche. Je höher die Auslastung,

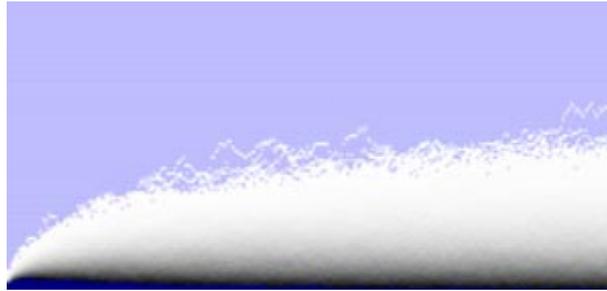


Abbildung 8.19: Das Zufallsverteilungsdiagramm der Population.

desto deutlicher ist der lange Einfluss zu erkennen. Das Zufallsverteilungsdiagramm aus Abbildung 8.19 zeigt die negativ-exponentielle Form des Einflusses. Die Abbildung 8.19 wurde unabhängig von der Analyse aus Kapitel 8.3.2 entworfen.

8.3.1 Analyse ohne Ausschluss der transienten Phase

Für die Analyse ohne Ausschluss der transienten Phase wird das Simulationstool *HIT* verwendet. Die Abbruchbedingung der Simulation ist, dass das 95%-Konfidenzintervall nicht größer als 5% des berechneten Mittelwertes ist.

Die Simulation benötigt 122861 Modellzeiteinheiten, um die Abbruchbedingung zu erfüllen. In dem Simulationsmodell liegt die durchschnittliche Anzahl der Kunden im System bei 4.150498. Das 95%-Konfidenzintervall beträgt 4.87% des Mittelwertes. Diese Ergebnisse stimmen mit der theoretischen Analyse überein.

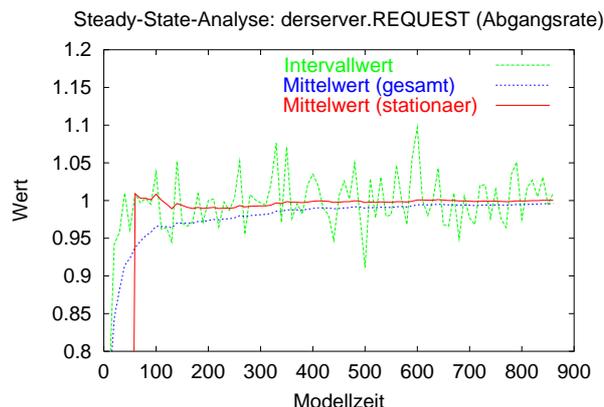


Abbildung 8.20: Stationärer Mittelwert versus gesamter Mittelwert (Abgangsrate).

8.3.2 Analyse mit Ausschluss der transienten Phase

Bei der Analyse mit Ausschluss der transienten Phase werden 100 parallele Simulationen gestartet. Das Ende der transienten Phase wird mit der Zufallsverteilungsmethode bestimmt. Die Analyse der stationären Phase erfolgt durch die direkte Analyse ohne Reduktion. Hierbei wird dasselbe Abbruchkriterium verwendet, wie bei der Analyse ohne Ausschluss der transienten Phase.

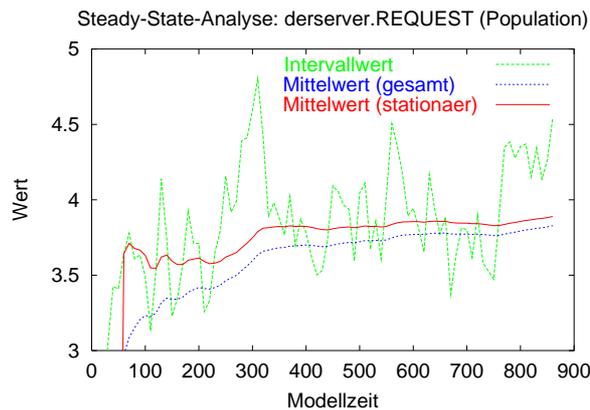


Abbildung 8.21: Stationärer Mittelwert versus gesamter Mittelwert (Population).

Das Ende der transienten Phase kann nach 50 Modellzeiteinheiten ermittelt und die Abbruchbedingung kann nach 860 Modellzeiteinheiten erfüllt werden. Im Durchschnitt sind während der transienten Phase 3.93825 Kunden im System. Das 95%-Konfidenzintervall beträgt 0.189646. Dieses Ergebnis stimmt mit der theoretischen Analyse überein.

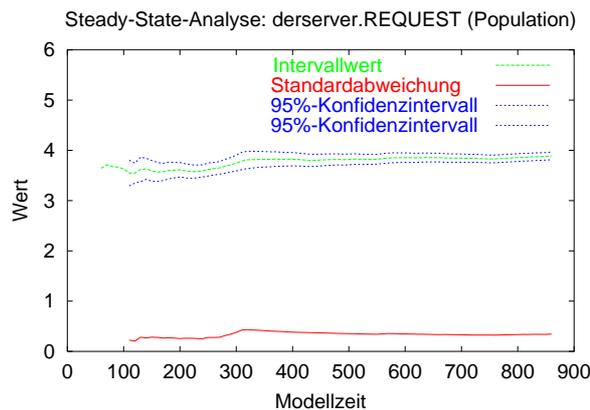


Abbildung 8.22: Die stationäre Phase der Population.

Die Abbildungen 8.20 und 8.21 zeigen den Unterschied zwischen dem gesamten Mittelwert und dem Mittelwert der Beobachtungen aus der stationären

Phase. Erneut werden hier nur die Abgangsrate und die Population betrachtet. Es wird deutlich, dass der Mittelwert der stationären Phase von Beginn an viel näher an dem theoretischen Wert liegt als der Mittelwert der gesamten Daten.

Die übliche Darstellung der Simulationsergebnisse durch den Mittelwert, die Standardabweichung und das 95%-Konfidenzintervall für das Leistungsmaß Population wird in Abbildung 8.22 gezeigt. Zu beachten ist allerdings, dass in diesem Fall nur die stationäre Phase aufgetragen ist.

8.3.3 Bewertung

Die Abbildungen 8.20 und 8.21 machen deutlich, dass es vorteilhaft ist, die Daten der transienten Phase bei der Analyse nicht zu berücksichtigen. Die Kurve des Mittelwertes der stationären Phase liegt von Beginn an näher am theoretischen Wert. Werden die Daten der transienten Phase doch berücksichtigt, muss eine sehr lange Simulation durchgeführt werden, um den Initialisierungseinfluss vernachlässigbar zu machen. So lässt sich die lange Simulationsdauer (122861 MZE) bei der Analyse durch *HIT* gegenüber der kurzen Simulationsdauer (860 MZE) bei der Analyse durch *ssa* erklären.

Bei diesem Modell kann der Einfluss der transienten Phase nicht vollkommen unterdrückt werden. In diesem Fall hat der Einfluss der transienten Phase einen "negativ-exponentiellen Charakter". Er ist nach einer sehr langen Simulationszeit noch vorhanden, allerdings mit nur sehr geringem Ausmaß. Der Vorteil der Zufallsverteilungsmethode ist hierbei, dass die Haupteinflussphase des Anfangszustandes eliminiert wird. Im strengen Sinne darf hier jedoch nicht von *dem Ende* der transienten Phase gesprochen werden.

Eine Simulation kann nur einen endlichen Horizont haben. Der Einfluss des Anfangszustandes verschwindet in diesem Fall aber erst nach einer unendlichen Simulationsdauer. Der Erwartungswert jeder Simulation kann daher nicht mit dem theoretischen Erwartungswert übereinstimmen. Jedes Simulationsergebnis wird durch den unendlich langen Einfluss des Anfangszustandes verfälscht. Die Simulationsergebnisse der Analyse mit und ohne Ausschluss der transienten Phase sind in diesem Fall nicht erwartungstreu.

8.4 M/M/1-Server: überlastet

Bei diesem Experiment wird ein M/M/1-Server mit der Verkehrsintensität $\rho = 2$ betrachtet. Da die Stationaritätsbedingung $\rho < 1$ verletzt ist, liegt kein stationäres Modell vor. Es existiert demnach keine stationäre Phase und kein Ende der transienten Phase.

8.4.1 Test auf Stationarität

Für den Test auf Stationarität wird die Methode der Zufallsverteilungsdiagramme und ein Vergleich zwischen Ankunfts- und Abgangsrate gewählt. Die Ab-

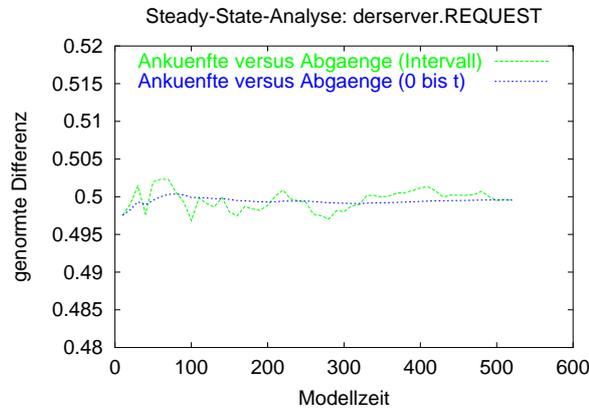


Abbildung 8.23: Genormte Differenz der Ankunftsrate und Abgangsrate.

Abbildung 8.23 zeigt, dass die genormte Differenz zwischen der Ankunftsrate und der Abgangsrate konstant auf einem zu hohen Niveau liegt, welches nicht durch natürliche Schwankungen erklärt werden kann.

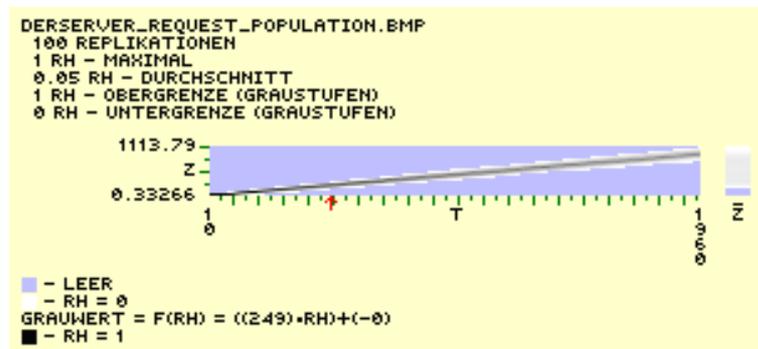


Abbildung 8.24: Zufallsverteilungsdiagramm der Population.

Die Abbildung 8.24 zeigt das Zufallsverteilungsdiagramm der Population. Die Zufallsverteilungen ändern sich mit wachsender Modellzeit. Dies ist ein weiterer Hinweis, dass das vorliegende Modell nicht stationär sein kann. Die Analyse wurde nach der Modellzeit 1960 abgebrochen.

8.4.2 Bewertung

Das Analysetool *ssa* hat richtig erkannt, dass nach der Modellzeit 2080 noch keine stationäre Phase vorliegt. Leider kann *ssa* keine Aussage darüber treffen, ob in

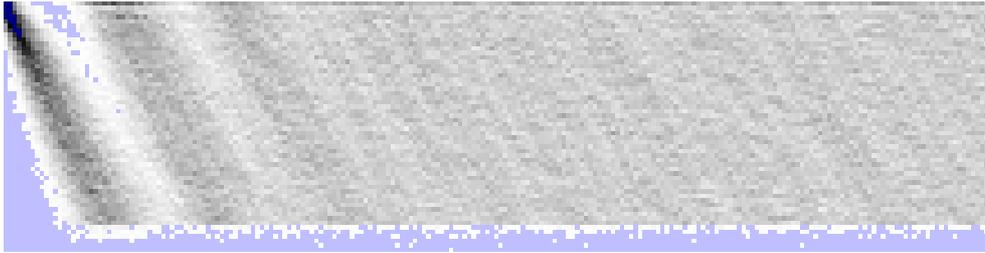


Abbildung 8.26: Zufallsverteilungsdiagramm des Lagerbestandes.

lungsmethode wird eine ausgewählte Stichprobe (der Proband) mit den nachfolgenden Stichproben verglichen. Das Ergebnis dieser Vergleiche bei dem Leistungsmaß Population ist in Abbildung 8.27 aufgetragen. Die Vergleiche ergeben im Laufe der Simulation immer kleinere Unterschiede, bis schließlich eine stationäre Phase vorausgesetzt werden kann.

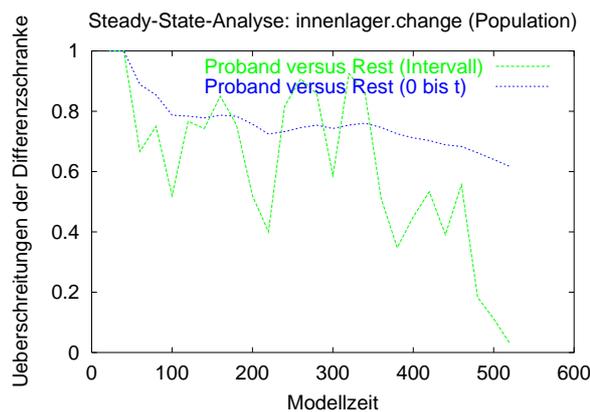


Abbildung 8.27: Vergleich der ausgewählten Stichprobe mit dem Rest. (Population)

Der Vergleich des Mittelwertes der Daten der stationären Phase mit dem Mittelwert der gesamten Daten macht deutlich, dass es sinnvoll ist, bei diesem Modell die Daten der transienten Phase zu verwerfen.

8.5.2 Bewertung

Bei vielen praktischen Problemen ist oft nicht klar, ob eine nicht zu vernachlässigende transiente Phase vorliegt. In diesem Fall ist es wichtig, die Zufallsverteilungsdiagramme zu betrachten. Die Abbildung 8.28 zeigt, dass hierdurch unnötig lange Simulationszeiten vermieden werden können.

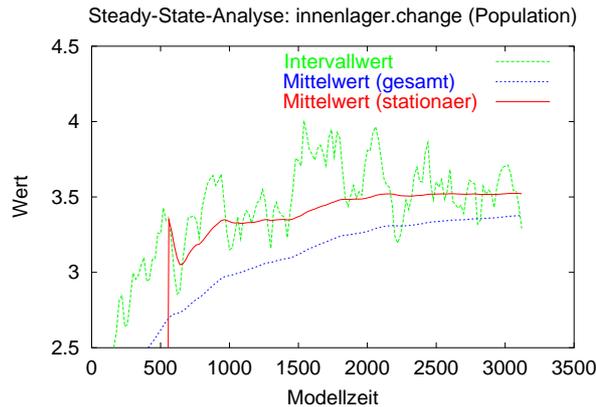


Abbildung 8.28: Stationärer Mittelwert versus gesamter Mittelwert (Population)

8.6 Warteschlangennetz mit mehreren Kundenklassen.

Dieses Experiment bezieht sich auf ein Modell von Banks und Dai (siehe [5]). Es handelt sich hierbei um ein Warteschlangennetz mit verschiedenen Kundenklassen, bestehend aus zwei $M/M/1$ -Servern. Die Bedienreihenfolge richtet sich nach der Kundenklasse. Die Verkehrsflussintensität ρ ist an beiden Servern kleiner als Null. Das Modell ist jedoch nicht stabil. Hier wird deutlich, dass bei einem System aus mehreren Servern die Stationaritätsbedingung der Verkehrsflussintensität nur eine notwendige Bedingung für Stationarität ist.

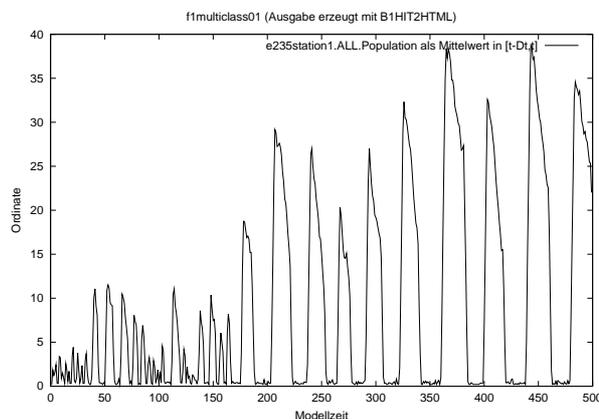


Abbildung 8.29: Die Population des Servers “Station1”.

Eine genaue Modellbeschreibung lässt sich in [5] Kapitel 1 nachlesen. Dieses Modell lässt sich in *HISLANG* mit Hilfe von *prioservern* modellieren. Das *HISLANG*-Modell ergibt ebenfalls die typischen Diagramme für die Population an den Servern “Station1” und “Station2” (siehe Abbildung 8.29 und 8.30). Interessant ist, ob das Modell mit Hilfe der Zufallsverteilungsmethode als “nicht-stationär” entlarvt werden kann.

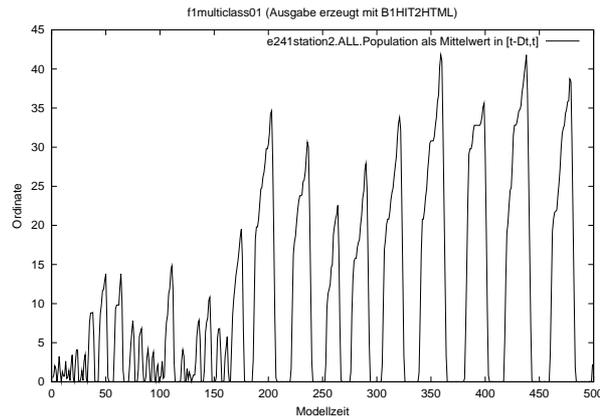


Abbildung 8.30: Die Population des Servers "Station2".

8.6.1 Test auf Stationarität

Bei der Zufallsverteilungsmethode wird eine ausgewählte Stichprobe mit den nachfolgenden Stichproben verglichen. Dieser Test wurde für die beiden Server "Station1" und "Station2" durchgeführt. Der zeitliche Verlauf dieser Tests ist in Abbildung 8.31 und 8.32 dargestellt. Beide Testverläufe lassen ein hohes Maß an Differenz erkennen. Dies deutet darauf hin, dass keine stationäre Phase vorliegt.

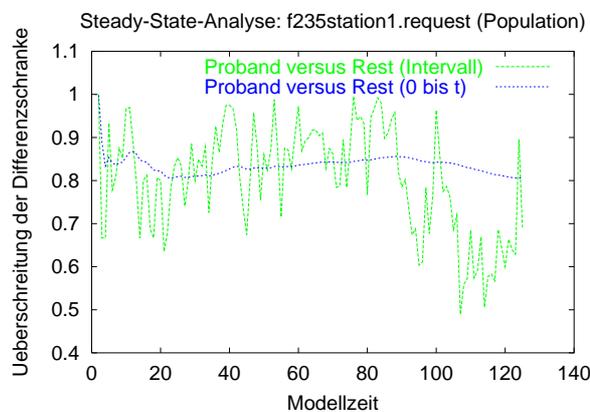


Abbildung 8.31: Vergleich der ausgewählten Stichprobe mit dem Rest. (Population/Station1)

Die Abbildungen 8.33 und 8.34 sind die Zufallsverteilungsdiagramme der Population an "Station1" und an "Station2". Die grobe Struktur dieser Abbildungen weist auf den Zusammenhang mit den Abbildungen 8.29 und 8.30 hin. Die Populationen wachsen im Laufe der Zeit an und ihre Zufallsverteilungen sind daher nicht konstant. Dies ist die Bestätigung für ein nicht stabiles Modell ohne stationäre Phase.

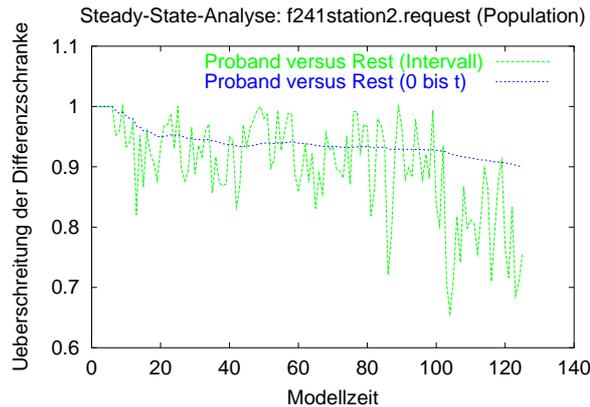


Abbildung 8.32: Vergleich der ausgewählten Stichprobe mit dem Rest. (Population/Station2)

8.6.2 Bewertung

In den Diagrammen 8.29 und 8.30 lassen sich Phasen erkennen, in denen die Population nahezu Null ist. Bei einem Test auf Stationarität besteht das potentielle Problem, dass eine dieser *Null-Phasen* mit der stationäre Phase verwechselt wird. Die Lage der Null-Phasen ist abhängig vom Startseed der Simulation. Bei mehreren Simulationen vermischen sich diese Null-Phasen mit Bereichen, in denen eine Warteschlange vorliegt. Aus diesem Grund erzielt die Zufallsverteilungsmethode korrekte Ergebnisse und die Null-Phasen werden nicht mit der stationären Phase verwechselt.

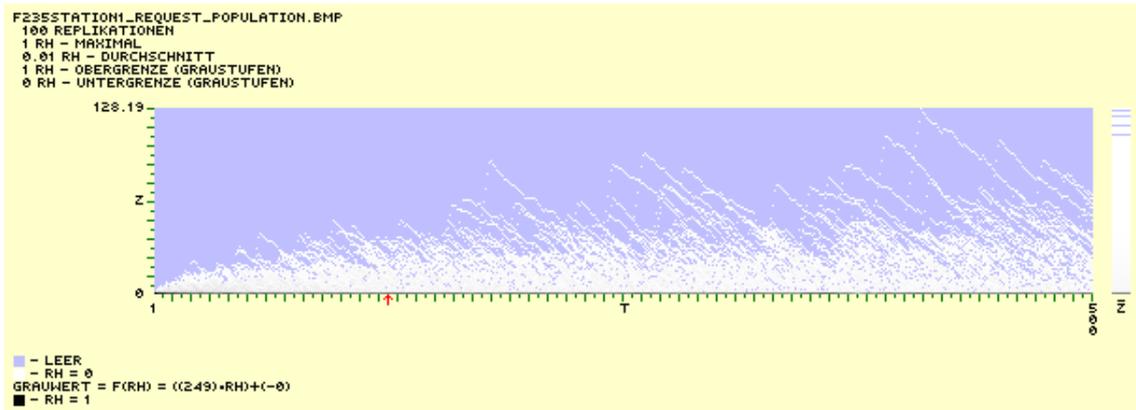


Abbildung 8.33: Das Zufallsverteilungsdiagramm der Population am Server “Station1”.

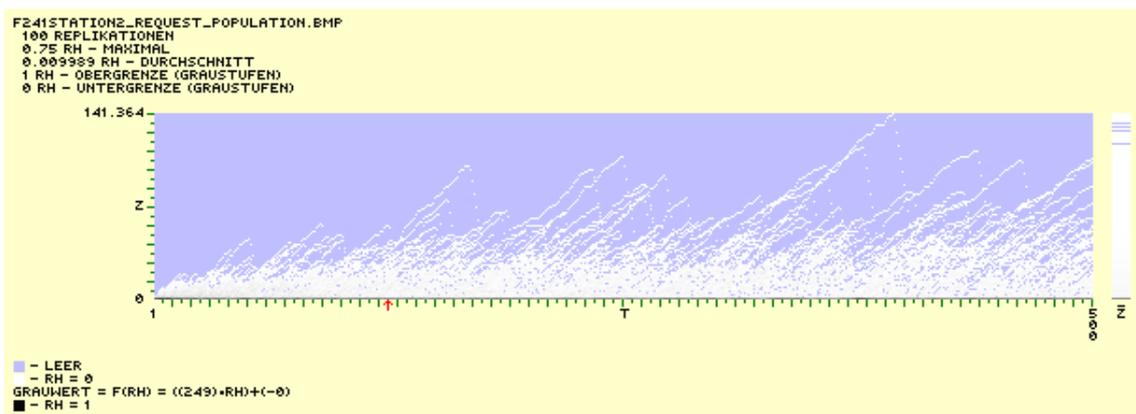


Abbildung 8.34: Das Zufallsverteilungsdiagramm der Population am Server “Station2”.

Kapitel 9

Schlussfolgerung und Ausblick

In diesem abschließenden Kapitel werden die Ergebnisse dieser Diplomarbeit kritisch hinterfragt. Die Vor- und Nachteile der erweiterten Auswertungsmethoden werden geschildert. In manchen Fällen ist es möglich, weitere Verbesserungsvorschläge zu geben.

9.1 Erkennung der Stationarität

Das Erkennen eines nicht stabilen Modells durch das erweiterte Analyseverfahren ist mit zusätzlichem Aufwand verbunden. Des Weiteren sind die Verfahren teilweise kompliziert in der Anwendung, da der Benutzer durch die Wahl der Analyseparameter auf die Auswertung Einfluss nehmen muss. Diese und weitere Punkte werden in diesem Kapitel diskutiert.

Höherer Aufwand:

Das Auswertungsbeispiel des Güterverkehrszentrums zeigt, dass eine Überprüfung der Stabilität des zu analysierenden Modells vorteilhaft ist. Nicht ausreichend ist das Abbruchkriterium, dass ein gewähltes Konfidenzintervall weniger als eine bestimmte Prozentzahl vom Mittelwert abweicht. Das Analysetool *ssa* besitzt hier einige alternative Möglichkeiten, deren Erfolg an dem Modell des Güterverkehrszentrums demonstriert wird (siehe Kapitel 8.1). Im Gegensatz zu den Standard-Analyseverfahren kann die Zufallsverteilungsmethode die Instabilität des Güterverkehrszentrums erkennen. Der höhere Aufwand der erweiterten Analyseverfahren ist daher gerechtfertigt.

Analyseparameter:

Bedingt durch die lange konstante Anfangsphase in der Simulation des Güterverkehrszentrums muss der Benutzer die Auswertung durch eine geschickte Wahl

der Analyseparameter maßgeblich beeinflussen. Um diese Parameter geeignet zu wählen, wird von dem Benutzer ein Modellverständnis verlangt, das er unter Umständen nicht hat. Es ist daher sinnvoll, diese Parameter durch *ssa* automatisch bestimmen zu lassen. Zunächst sollte hier jedoch die Machbarkeit überprüft werden. Dieses Vorgehen wird in dieser Diplomarbeit nicht weiter verfolgt.

Abbruchbedingung bei Instabilität:

Wünschenswert ist die Erweiterung des Analysetools *ssa* um eine Funktion, die selbstständig über die Annahme eines instabilen Modells entscheidet. Bisher sucht *ssa* bei einem instabilen Modell solange nach einer stationären Phase, bis die Suche durch den Benutzer abgebrochen wird.

In den sequentiellen Algorithmen aus [30] wird ein ähnliches Vorgehen empfohlen: Der Benutzer gibt eine konkrete Modellzeit vor, bei der die Suche nach dem Ende der transienten Phase beendet wird. Auch bei diesem Verfahren kann der Benutzer keine Aussage darüber treffen, ob bei einer längeren Suche vielleicht doch noch eine stationäre Phase eingetreten wäre.

Für den Benutzer ist es natürlich von Vorteil, wenn er definitiv weiß, ob ein Modell wirklich nicht stabil ist. Dieses Problem kann nicht durch die Betrachtung der Simulationsdaten gelöst werden. Eine Analyse der Modellstruktur ist hier erforderlich. Darauf wird in dieser Diplomarbeit verzichtet.

9.2 Erkennung der stationären Phase

Die Länge einer Simulation und die Qualität der Ergebnisse sind die wichtigsten Faktoren bei der simulativen Analyse. Ob die erweiterten Analyseverfahren positiven Einfluss auf diese Faktoren haben, wird nun Anhand der Testmodelle überprüft.

Verkürzte Simulationszeit:

Die Vorteile des Verwerfens von den Daten der transienten Phase wird bei den Analysebeispielen mit dem schlecht initialisierten M/M/1-Server deutlich (siehe Kapitel 8.2). Je nach Modell können die Simulationsläufe auf Grund der Parallelisierung und der intelligenteren Auswertung massiv verkürzt werden. Bei dem schlecht initialisierten M/M/1-Server kann die benötigte Modellzeit um mehr als das 350-fache (223740ME zu 600ME) verkürzt werden.

Die Einsparung in der Modellzeit bedeutet weniger benötigte Realzeit. Abzuwägen ist jedoch, dass die Zeiteinsparung durch 100 parallele Simulationsläufe ermöglicht wird. Oft stehen ausreichend Rechner zur Verfügung, sodass die benötigte Parallelisierung keine Einschränkung darstellt. Des Weiteren können auf

einem Rechner mehrere Simulationsläufe gestartet werden, deren Durchführung dann allerdings länger dauert.

Steht genügend Rechnerkapazität zur Verfügung, dann sind die erweiterten Analyseverfahren, die auf der Parallelisierung basieren, eine gute Alternative zu den Analyseverfahren, die nur einen einzelnen Simulationslauf benötigen. Der höhere technische Aufwand der Parallelisierung, lässt sich durch eine bessere Ergebnisqualität und kürzere Simulationen rechtfertigen.

Erwartungstreue:

Häufig kann der Einfluss des Startzustandes durch einen multiplikativen Faktor beschrieben werden, der von der Modellzeit abhängig ist. Dieser Faktor nimmt den Wert eins an, wenn die stationäre Phase erreicht ist (siehe Abbildung 9.1). Hat der Einfluss noch Auswirkung auf die Daten der Simulation, ist der Faktor größer oder kleiner als eins.

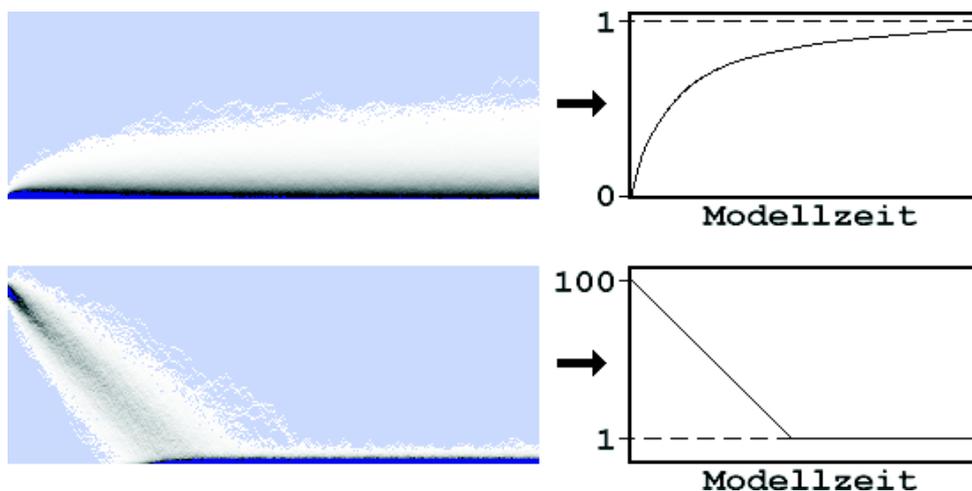


Abbildung 9.1: Der Einfluss des Startzustandes dargestellt für einen normal ausgelasteten und einen schlecht initialisierten M/M/1-Server. Der Einfluss wird durch einen modellzeitabhängigen Faktor beschrieben. Im ersten Beispiel nimmt der Einfluss nur sehr langsam ab. Erst im Unendlichen ist der Faktor gleich eins. Im zweiten Beispiel gibt es einen konkreten Modellzeitpunkt, an dem der Einfluss nicht mehr vorhanden ist.

Der Ergebnisvergleich der Auswertungsbeispiele “normal ausgelasteter Server” und “schlecht initialisierter Server” ergibt: Je ausgeprägter die transiente Phase, desto leichter ist die Erkennung der stationären Phase. Die Methoden zur Erkennung des Beginns der stationären Phase, die im Auswertungstool *ssa* implementiert wurden, sind darauf spezialisiert, einen konkreten Modellzeitpunkt zu bestimmen, an dem der Einfluss des Startzustandes vernachlässigt werden kann.

Nicht alle Modelle haben einen konkreten Modellzeitpunkt, an dem dieser Einfluss komplett verschwindet. In dem Beispiel des normal ausgelasteten Servers nimmt der Einfluss des Startzustandes nur sehr langsam ab (siehe Abbildung 9.1). Erst im Unendlichen ist er ganz verschwunden und daher in praktischen Auswertungen immer vorhanden. Egal wieviele Daten verworfen werden, der Mittelwert kann kein erwartungstreuer Schätzer für das durchschnittliche Verhalten im Unendlichen sein.

Bei der Auswertung des schlecht initialisierten Servers wird deutlich, dass der Einfluss des Startzustandes nicht immer bis ins Unendliche reicht. In diesem Beispiel ist er annähernd linear (siehe Abbildung 9.1). Es existiert daher ein konkreter Modellzeitpunkt, an dem der Einfluss des Startzustandes ganz verschwunden ist. Der durch *ssa* bestimmte Mittelwert schätzt den theoretischen Wert daher erwartungstreu.

Im Gegensatz dazu steht die Auswertung durch die Standard-Analyseverfahren. Die Schätzung des theoretischen Wertes erfolgt hierbei nicht erwartungstreu, da die Daten der transienten Phase den Mittelwert in Richtung des Initialisierungszustandes verfälschen – egal wie lang die Simulation ist. Dies bestätigt die Ergebnistabelle aus Kapitel 4.4. In diesem Fall ist der höhere Aufwand der erweiterten Analyse nur ein unbedeutender Nachteil, gegenüber dem Vorteil der deutlich besseren Ergebnisqualität.

Das Modell entscheidet also darüber, ob die Analyse durch *ssa* asymptotisch erwartungstreu oder wirklich erwartungstreu ist. Der Benutzer muss in der Lage sein, die Form des Einflusses des Startzustandes richtig zu erkennen, um damit die Qualität der Ergebnisse einschätzen zu können. Behilflich sind ihm hierbei die erzeugten Zufallsverteilungsdiagramme (siehe Abbildung 9.1). Eine gewisse Grundkenntnis wird jedoch benötigt, um die Zufallsverteilungsdiagramme zu interpretieren. Wünschenswert ist eine Erweiterung von *ssa*, die dem Benutzer Auskunft darüber gibt, ob die Ergebnisse erwartungstreu sind.

9.3 Statistische Auswertung der stationären Phase

Die Auswertung durch einen einzelnen Simulationslauf kann als Spezialfall der Auswertung durch mehrere Simulationsläufe betrachtet werden. Trotz dieses Zusammenhangs sind die Probleme beider Ansätze grundsätzlich verschieden und werden an dieser Stelle getrennt voneinander betrachtet.

Eine Simulation:

Die Sequenz der Daten, die durch “Batch Means” entsteht, hat generell nicht so hohe Schwankungen wie die Datensequenz aus “Spaced Batch Means”, da die

Gruppen-Mittelwerte aus mehreren Daten berechnet werden. Bei “Spaced Batch Means” werden Daten völlig außer Acht gelassen, sodass durch die entstandenen Lücken die Unabhängigkeit gefördert wird. “Spaced Batch Means” tendiert daher wohl eher zu kleineren Gruppen, die mit derselben Wahrscheinlichkeit unabhängig voneinander sind. Beide Verfahren besitzen Vor- und Nachteile und bilden in dem Analysetool *ssa* zwei alternative Auswertungsmöglichkeiten, die sich gegenseitig ergänzen.

Viele Simulationen:

Werden mehrere Simulationsläufe durchgeführt, entscheidet die Wahl der Startseeds darüber, ob die Ergebnisse der einzelnen Simulationen als unabhängig aufgefasst werden können. Bei einer Wahl der Startseeds, wie sie in Kapitel 2.3.4 beschrieben ist, kann die Unabhängigkeit vorausgesetzt werden. Diese unabhängigen Ergebnisse können direkt durch die Berechnung von Konfidenzintervallen ausgewertet werden.

Alternativ können die einzelnen Messreihen der Simulationen durch Mittelwertbildung auf eine stellvertretende Messreihe reduziert werden. Die Daten dieser stellvertretenden Messreihe sind im Allgemeinen aber nicht unabhängig. Aus diesem Grund wird ein Verfahren wie “Batch Means” benötigt. Hierbei ist ein Unabhängigkeitstest erforderlich. Dieser Unabhängigkeitstest ist jedoch für eine weitere Unsicherheit verantwortlich, da die Unabhängigkeit nur für eine bestimmte Wahrscheinlichkeit gilt.

Die Anzahl der Daten zur Berechnung des Konfidenzintervalls ist bei der direkten Analyse durch die Anzahl der Simulationsläufe beschränkt. Dies sind im Allgemeinen viel weniger Daten als bei einer Analyse durch Reduktion. Kleine Datenmengen tendieren zu größeren Konfidenzintervallen. Dafür haben diese wenigen Daten eine viel geringere Varianz. Je kleiner die Varianz, umso kleiner ist das Konfidenzintervall.

Die direkte Analyse und die Analyse mit Reduktion besitzen Vor- und Nachteile. Daher ist keine Analysemethode eindeutig der anderen vorzuziehen. In der Durchführung ist die direkte Analyse problemloser, da bei ihr kein Unabhängigkeitstest benötigt wird.

9.4 Implementierung

Bei der Implementierung wird auf schon bestehende Programme zurückgegriffen. Da diese Programme einen fortgeschrittenen Entwicklungsstand besitzen, bieten sie viele Vorteile. Problematisch ist jedoch, dass die Schnittstellen dieser Programme nicht auf das hier betrachtete Problem abgestimmt sind.

Parallelisierung:

Die Parallelisierung ist ein wichtiger Vorzug des Analysetools *ssa*. Die Anwendung der Zufallsverteilungsmethode wird hierdurch erst möglich. Leider ist die Anzahl der parallelen Simulationen auf Grund des begrenzten Hauptspeichers beschränkt. Mit den zur Verfügung stehenden Rechnern konnte eine Höchstzahl von 100 parallelen Simulationen erreicht werden. Die Zufallsverteilungsdiagramme in Abbildung 8.1, Abbildung 8.5 und Abbildung 8.6 wurden mit weitaus mehr Simulationsläufen entworfen, die nacheinander durchgeführt wurden.

Eine größere Anzahl an parallelen Simulationen ist wünschenswert. Es ist zu überlegen, ob bei wenigen parallelen Simulationen die Zufallsverteilungsdiagramme, basierend auf den Box-Plots (siehe Kapitel 6.3.2), mehr Sinn machen, auch wenn ihr Layout weniger beeindruckend ist.

Leistungsmaße:

Das Modell des Güterverkehrszentrums hat gezeigt, dass eine Betrachtung des Lagerbestandes wünschenswert ist. Anhand des Lagerbestandes lässt sich ohne Probleme die Instabilität des Modells erkennen. Unter Betrachtung der Leistungsmaße Ankunftsrate, Abgangsrate, Population und Verweilzeit ist dies nur schwer möglich.

Eine Auswertung, basierend auf einer flexibleren Auswahl an Leistungsmaßen ist daher vorteilhaft. Das Filterwerkzeug *steady* berechnet die Leistungsmaße aus den Daten, die bei der Simulation mit *HIT* durch das "Trace-Format" ausgegeben werden. Aus diesen Daten lässt sich der Lagerbestand leider nicht bestimmen. Aus diesem Grund beschränkt sich diese Diplomarbeit auf die Leistungsmaße Ankunftsrate, Abgangsrate, Population und Verweilzeit.

Externe Statistik-Tools:

Bei dieser Diplomarbeit ist die Verwendung eines externen Statistik-Tools von Vorteil. Das ausgewählte Statistik-Tool *octave* erfüllt alle Anforderungen in Bezug auf die benötigte Funktionalität. Leider ist die Kommunikation mit *octave* auf Grund der großen Datenmengen ineffizient. Bei der Implementierung müssen also Kompromisse gefunden werden, um zu hohe Laufzeiten zu vermeiden.

Dieses Problem tritt prinzipiell bei jedem externen Statistik-Tool auf, mit dem über "Files" oder "Pipes" kommuniziert wird. Wenn die Algorithmen des Statistik-Tools die in *ssa* aufgebaute Datenstruktur direkt nutzen könnten, wäre dies eine Lösung des Problems. Wird innerhalb des externen Statistik-Tools eine unabhängige Datenstruktur erstellt, ist dies ebenfalls eine Lösung, hätte jedoch einen höheren Speicherbedarf zur Folge.

9.5 Fazit

Die erweiterte Auswertungsstrategie ist gegenüber der allgemeinen Auswertungsstrategie bei den betrachteten Testmodellen eine echte Verbesserung. Das Überprüfen der Stationarität ist bei einigen Modellen notwendig für korrekte Ergebnisse. Das Verwerfen der Daten der transienten Phase führt nicht nur zu einer effizienteren Simulation, sondern in gewissen Fällen auch zu einer höheren Qualität (Erwartungstreue) der Ergebnisse. Diese Vorzüge rechtfertigen den höheren Aufwand der erweiterten Auswertungsstrategie.

Insbesondere sind das Zufallsverteilungsdiagramm und die Zufallsverteilungsmethode hervorzuheben, die im Rahmen dieser Diplomarbeit konkretisiert werden. Im Gegensatz zu anderen Methoden überprüfen diese beiden Verfahren nicht nur die Mittelwertstationarität. Da diese Verfahren auf Stichproben eines konkreten Modellzeitpunktes basieren, wird annähernd die strenge Stationarität geprüft. Die eigentliche strenge Stationarität kann nicht überprüft werden, da sie für die Zufallsverteilungen definiert ist (siehe Kapitel 3.4) und hier nur endliche Stichproben zur Verfügung stehen.

Die Zufallsverteilungsmethode besitzt gegenüber anderen Methoden viele Vorteile, da sie den Zufallscharakter der Beobachtungen besser berücksichtigt, in dem sie mit einer Abstraktion der eigentlichen Zufallsverteilung arbeitet. Die Zufallsverteilungsdiagramme ermöglichen dem Benutzer ein tiefes Modellverständnis, da sie sich nicht nur auf den Verlauf des Mittelwertes beschränken. Die Parallelisierung hat sich als ein wichtiges Werkzeug erwiesen, um die vorgestellten Methoden zu realisieren.

Für die Testmodelle dieser Diplomarbeit konnte gezeigt werden, dass die Verwerfung von den Daten der transienten Phase von Vorteil ist. Dies gilt auch für den normal ausgelasteten M/M/1-Server ($\rho = 0.8$). Die Auswirkungen auf die Qualität der Ergebnisse sind in diesem Fall jedoch nicht so gravierend wie in anderen Fällen. Dies liegt daran, dass es kein konkretes Ende der transienten Phase gibt. Bei einem hoch ausgelasteten M/M/1-Server ($\rho > 0.9$) verringert sich der Einfluss des Startzustandes noch wesentlich langsamer, sodass in Frage gestellt werden kann, ob die Unterteilung in eine transiente und eine stationäre Phase sinnvoll ist. In diesem Fall hat das Verwerfen anfänglicher Daten nur noch geringste Auswirkungen. Der höhere Aufwand durch die erweiterten Auswertungsstrategien kann in diesem Beispiel nicht gerechtfertigt werden. Es ist jedoch durchaus sinnvoll die Zufallsverteilungsdiagramme zu betrachten, um die hohe Auslastung zu Erkennen.

Bei den Analysen der Testmodelle stellt sich heraus, dass eine detaillierte Ergebnisdarstellung wichtig ist. Der Benutzer sollte nicht nur das Endergebnis erfahren, sondern auch über den Simulationsverlauf informiert werden, um Fehl-

interpretationen der Ergebnisse zu vermeiden. In den implementierten Analyseprogrammen wird angestrebt, den Verlauf der Simulation möglichst genau nachvollziehbar zu gestalten. Der Benutzer von *ssa* kann schon während der Simulation in vielen Abbildungen die Darstellungsgrenzen optimal anpassen. Durch die Bildschirmausgabe und die vier unterschiedlichen Ergebnisdarstellungen (siehe Kapitel 7.1.6) erhält der Benutzer einen tiefen Einblick in die internen Abläufe des zu analysierenden Modells.

Literaturverzeichnis

- [1] S. Albers. Effiziente Algorithmen. Vorlesungsscript, Universität Dortmund - Fachbereich Informatik - Lehrstuhl II, 2000.
- [2] J. Arndt and C. Haenel. *Pi - Algorithmen, Computer, Arithmetik*. Springer Verlag, 1998.
- [3] M. Arns, F. Bause, M. Fischer, P. Kemper, and H. Tatlitürk. Modellierung und Analyse des KV-Terminals im Güterverkehrszentrum. SFB-Bericht, Universität Dortmund - Fachbereich Informatik - Lehrstuhl IV, 2000.
- [4] J. Banks, editor. *Handbook of Simulation*. John Wiley & Sons, Inc., 1998.
- [5] J. Banks and J. G. Dai. Simulation studies of multiclass queueing networks. *IIE Transactions*, 29:213–219, 1997.
- [6] F. Bause and H. Beilner. Intrinsic problems in simulation of logistic networks. *11th European Simulation Symposium and Exhibition (ESS99)*, Oktober 1999.
- [7] F. Bause and P. S. Kritzinger. *Stochastic Petri Nets*. Vieweg, 1996.
- [8] H. Beilner. Simulation. Vorlesungsscript, Universität Dortmund - Fachbereich Informatik - Lehrstuhl IV, 1998.
- [9] H. Beilner. HI-SLANG Reference Manual. Manual, Universität Dortmund - Fachbereich Informatik - Lehrstuhl IV, 1999.
- [10] H. Beilner. Leistungsbewertung von Rechen- und Kommunikations-Systemen. Vorlesungsscript, Universität Dortmund - Fachbereich Informatik - Lehrstuhl IV, 1999.
- [11] H. Beilner, F. Bause, H. Tatlitürk, A. van Almsick, and M. Völker. Zum B-Modellformalismus — Version B1 – zur Vorbereitung automatisierter Analysen von Modellen logistischer Systeme hinsichtlich technischer, ökonomischer und ökologischer Ziele. SFB-Bericht, Universität Dortmund - Fachbereich Informatik - Lehrstuhl IV, 1999.

- [12] E. Best, J. Desel, and J. Esparza. Traps characterize home states in free choice systems. *Theoretical Computer Science*, 101:161–176, 1992.
- [13] P. Bratley, B. L. Fox, and L. E. Schrage. *A Guide to Simulation*. Springer, 1987.
- [14] R. W. Conway. Some tactical problems in digital simulation. *Management Science*, 10(1):47–61, Oktober 1963.
- [15] J. R. Emshoff and R. L. Sisson. *Design and use of Computer Simulation Models*. The MacMillan Company, 1970.
- [16] L. Fahrmeir, R. Künstler, I. Pigeot, and G. Tutz. *Statistik*. Springer, 1990.
- [17] G. S. Fishman. Estimating sample size in computer simulation experiments. *Management Science*, 18(1):21–38, September 1971.
- [18] G. S. Fishman. *Concepts and Methods in Discrete Event Digital Simulation*. John Wiley & Sons, Inc., 1973.
- [19] G. S. Fishman. Statistical analysis for queueing simulation. *Management Science*, 20(3):363–369, November 1973.
- [20] A. V. Gafarian, C. J. Ancker, and T. Morisaku. Evaluation of commonly used rules for detecting steady state in computer simulation. *Naval Research Logistics Quarterly*, pages 511–529, 1978.
- [21] G. Gordon. *System Simulation*. Prentice-Hall, 1969.
- [22] P. Heidelberger and P. D. Welch. A spectral method for confidence interval generation and run length control in simulations. *Communications of the ACM*, 24(4):233–245, April 1981.
- [23] P. Heidelberger and P. D. Welch. Simulation run length control in the presence of an initial transient. *Operations Research*, 31(6):1109–1144, November-December 1983.
- [24] H. Herold. *UNIX - Systemprogrammierung*. Addison-Wesley, 1996.
- [25] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc., 1991.
- [26] W. D. Kelton and A. M. Law. A new approach for dealing with the startup problem in discrete event simulation. *Naval Research Logistics Quarterly*, 30:641–658, 1983.
- [27] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 2000.

- [28] J-S. R. Lee, D. McNickle, and K. Pawlikowski. Initial transient period detection for steady-state quantile estimation. *Proceedings of the Summer Computer Simulation Conference*, pages 169–174, 2000.
- [29] M. Markus. Ljapunow - Diagramme. *Spektrum der Wissenschaft*, pages 66–73, April 1995.
- [30] K. Pawlikowski. Steady-state simulation of queueing processes: a survey of problems and solutions. *ACM Computing Surveys*, 22:123–170, June 1990.
- [31] R. Schlittgen and B. H. J. Streitberg. *Zeitreihenanalyse*. Oldenbourg, 1997.
- [32] L. W. Schruben. Detecting initialization bias in simulation output. *Operations Research*, 30(3):569–590, May-June 1982.
- [33] L. W. Schruben. Confidence interval estimation using standardized time series. *Operations Research*, 31:1090–1108, 1983.
- [34] L. W. Schruben, H. Singh, and L. Tierney. Optimal tests for initialization bias in simulation output. *Operations Research*, 31(6):1167–1178, November-December 1983.
- [35] R. E. Shannon. *Systems simulation: the art and science*. Prentice-Hall, Inc., 1975.
- [36] Bjarne Stroustrup. *Die C++ Programmiersprache*. Addison-Wesley, 1998.
- [37] P. A. Tipler. *Physik*. Spektrum Akademischer Verlag, 1994.
- [38] K. D. Tocher. *The Art of Simulation*. D. Van Nostrand Co., Inc., 1963.
- [39] I. Wegener. *Effiziente Algorithmen für grundlegende Funktionen*. B. G. Teubner Stuttgart, 1989.