Ludwig-Maximilians-Universität München

Bachelorseminar
Information und statistische Inferenz
Wintersemester 2014/15

Reproduzierbare Simulationen in der Ökonometrie

Autor:
Caroline Cazim

Betreuer:
Paul Fink M.Sc.

10. Januar 2015

Inhaltsverzeichnis

1 Einleitung

Simulationen verschiedenster Arten, sind ein wichtiger Teil der Veröffentlichung wissenschaftlicher Arbeiten. Simulationen ermöglichen es komplexe Situationen der realen Welt am Computer zu untersuchen und Lösungen dafür zu finden. Sie sind vielseitig anwendbar und können sehr unterschiedlich genutzt werden. Heutzutage finden Computerexperimente, die zufällige Elemente enthalten und dafür meist Simulationen notwendig sind, immer mehr Anwendung.
Ein wichtiges Thema hierbei, ist die Reproduzierbarkeit von Simulationen, d.h. die Möglichkeit die vorhandenen Simulationen exakt nachzustellen.
Auch in der Ökonometrie, hat die Möglichkeit und vor allem die Wichtigkeit der Reproduzierbarkeit, in den letzten Jahren immer mehr Aufmerksamkeit gewonnen. Um Simulationen reproduzierbar, sowie auch verständlich zu machen, benötigt es bestimmte Kriterien und Richtlinien, die von den Autoren eingehalten werden sollten, um eine einheitliche Arbeitsweise zu schaffen. Die Möglichkeit der Nachvollziehbarkeit und Reproduzierbarkeit geht oft verloren, da in den meisten Arbeiten nur Beschreibungen in Textform vorliegen und kein dazugehöriger Code oder Daten, zur Verfügung gestellt werden. Diese sehr unterschiedliche Art und Weise, wie Simulationen verwendet, veröffentlicht und zur Verfügung gestellt werden, werden später näher erläutert.
Der Artikel *Reproducible Econometric Simulations* von Christian Kleiber und Achim Zeileis (Kleiber und Zeileis, 2013) geht detailliert, auf die in der Praxis bestehende Problematik ein und ist die Grundlage dieser Arbeit.
Im nachfolgenden Kapitel wird der grundlegende Aufbau einer Simulation erklärt. Anschließend wird die derzeitige Praxis und die vorhandene Problematik thematisiert. Ein wichtiges Thema hierbei ist die Archivierung der Artikel mit dazugehörigen Codes. Um eine sinnvolle Vorgehensweise in wissenschaftlichen Arbeiten in der Zukunft zu ermöglichen, wurden von Chritian Kleiber und Achim Zeileis (Kleiber und Zeileis, 2013) Richtlinien vorgeschlagen, die bei Einhaltung der Autoren, zu einer wesentlichen Verbesserung der derzeitigen Problematik führen würde. Abschließend wird in Kapitel 5 ein Beispiel einer reproduzierbaren Simulation wiedergegeben, um die vorgeschlagenen Richtlinien zu verdeutlichen und verständlich zu machen.

2 Simulationen

Um das Problem der Reproduzierbarkeit zu verdeutlichen und nachvollziehen zu können, wird im Folgenden der grundlegende Aufbau einer Simulation erklärt.

2.1 Simulationsaufbau

Prinzipiell besteht eine Simulation aus den drei folgenden Bestandteilen:

1. Modell zur Generierung der Daten (Datengenerierender Prozess, DGP)
2. Zu untersuchendes statistisches Verfahren
3. Simulationsdurchführung

Jeder Bestandteil enthält Parameter, die variiert werden können, wie z.B. der Stichprobenumfang n. Eine konkrete Kombination dieser Parameter ergibt ein Simulationsszenario.(Berger und Oberhauser, 2014)

1. Modell zur Generierung der Daten

Statistische Analyseverfahren beruhen in der Regel auf einer Annahme darüber, wie die zu analysierenden Daten entstanden sind. Diese Annahme ist eine Verteilungsannahme, das sogenannte datengenerierende Modell DGP. Man nimmt also an, dass die Daten Realisierungen von Zufallsgrößen mit bestimmten Verteilungen sind.
Die Funktion dgp(), mit der man das datengenerierende Modell erzeugt, ist meist ein knapper und einfacher Code, die nun für jegliche interessierenden Testprozeduren verwendet werden kann.
Datengenerierende Modelle können einfache Wahrscheinlichkeitsverteilungen für einzelne Variablen, aber auch komplexe Zusammenhangsmodelle sein, wie z.B. lineare Modelle (Wiencierz, 2010).

Möchte man die Daten zufällig generieren, benötigt es bestimmte Kriterien, die einzuhalten sind. Für die Erzeugung zufälliger Daten, sollte man einen Zufallszahlengenerator (RNG) verwenden. Entweder man konstruiert diesen selbst oder bedient sich einer bereits programmierten RNG. In beiden Fällen, muss die RNG explizit angegeben werden. In der Anwendung erzeugen Zufallszahlengeneratoren sogenannte pseudozufällige Zahlen, d.h. sie erzeugen eine Zahlenfolge, die zwar zufällig aussieht, es jedoch nicht

ist, da sie durch einen deterministischen Algorithmus berechnet wird. Solche Pseudozufallszahlen sind von Computern sehr einfach zu erzeugen und sind in praktisch allen Programmiersprachen verfügbar. Diese können später reproduziert werden, indem man einen Startwert, den sogenannten *seed* setzt und einen Pseudozufallsgenerator erhält. Für die Verwendung eines RNG benötigt man einen Algorithmus, der gleichverteilte Zahlen zufällig erzeugt. Sehr beliebt hierfür ist das Mersenne Twister Verfahren. Diese werden jedoch nur auf einem Intervall [0,1] erzeugt und müssen anschließend, je nachdem welche Verteilung benötigt wird, tranformiert werden. Dies geschieht meist mit der Inversionsmethode.
Wichtig zu wissen, um später die unten aufgeführte Problematik besser nachvollziehen zu können, ist dass in vielen Software-Paketen, oft vorgefertigte Funktionen zur Verfügung gestellt werden, wodurch die Generierung der Zufallszahlen und die Transformation, im Nachhinein nicht nachvollziehbar sind.

Ein Grund für die vielen verschiedenen Simulationsergebnisse, sind die Unterschiede in den Zufallszahlengeneratoren. Idealerweise sollten die Unterschiede jedoch gering sein. Wichtig ist, wie bereits oben erwähnt, dass die RNG immer genau angegeben wird. Es kann auch sein, dass die selbe RNG benutzt wird und es trotzdem zu geringen Abweichungen kommen kann, je nachdem wie die Software, mit der man arbeitet, den *seed* initialisiert.

2. Zu untersuchendes statistisches Verfahren

Je nachdem was untersucht werden möchte, werden hier verschiedene statistische Verfahren eingesetzt, um die zu intererssierenden Analysen durchzuführen.
Für die Analysen, werden die im ersten Schritt generierten Daten verwendet. Hier finden auch oft die Wiederholungen des Simulationsszenarios statt. Es ist wichtig, mehrere Wiederholungen von dem selben Szenario durchzuführen, wenn Zufallsvariablen verwendet werden, um die Variabilität der Modellparameter schätzen zu können (Eugster, 2012).

3. Simulationsdurchführung

In der Simulationsdurchführung werden Parameter variiert, um die Ergebnisse für verschiedene Szenarien zu erhalten.

2.2 Simulationsaufruf

Der Simulationsaufbau sollte optimalerweise so gestaltet sein, dass durch einen Funktionsaufruf `results` ← `simulation()` alle Berechnungen vollständig durchgeführt werden, d.h. durch die Funktion simulation(), die die Simulationsdurchführung beinhaltet, werden alle weiteren Funktionen aufgerufen. Hier wird auch der *seed* gesetzt, um die Simulation im Nachhinein exakt nachstellen zu können. Dieses ist die Basis der Reproduzierbarkeit.

Der Simulationsaufbau, sowie der Aufruf, werden in Kapitel 5, anhand eines Beispiels verdeutlicht.

2.3 Information durch Simulation

Welche Informationen sich aus einer Simulation bestimmen lassen, hängt stark davon ab, was genau man untersuchen möchte.
Eine Simulationsstudie dient im Allgemeinen dazu, ein statistisches Verfahren anhand von selbst erzeugten Daten, deren Eigenschaften bzw. Struktur man kennt, zu untersuchen.
Ziel vieler Analyseverfahren ist es dann, etwas über die durch das Modell beschriebene Verteilung zu lernen, z.B. Verteilungsparameter zu schätzen.
Die zu simulierenden Daten können also einzelne Variablen, in einem bestimmtem Zusammenhang stehende Variablen oder z.B. auch Schätzer und Teststatistiken sein. (Wiencierz, 2010)

Möchte man beispielsweise die Güte eines Tests untersuchen, muss man die Fehlerwahrscheinlichkeit 1. Art, den sogenannten α-Fehler, wenn die Nullhypothese, also die Annahme, die methodisch untersucht werden soll, zurückgewiesen wird, obwohl sie in Wirklichkeit wahr ist, eines Tests untersuchen. Man simuliert die Daten entsprechend der Nullhyphothese. Es wird ein Signifikanzniveau α vorgegeben und die Testgröße berechnet, sowie auch die Testentscheidung durchgeführt. Dies geschieht für jedes Simulationsszenario und wird jeweils als ein Testergebnis gespeichert. Durch die relative Häufigkeit der Fälle, in denen die Nullhypothese fälschlicherweise verworfen wurde, wird die Fehlerwahrscheinlichkeit 1. Art geschätzt. Man betrachtet dann, ob die aus der Simulation geschätzte Güte, für verschiedene Parameterkombinationen variiert.
Ein weiteres Beispiel für die Erhaltung von Informationen aus Simulationen ist die Untersuchung der Power, welche den Wert 1-β besitzt, wobei β die Wahrscheinlichkeit bezeichnet, einen Fehler 2. Art zu begehen, d.h. die Nullhypothese beizubehalten, obwohl in Wirklichkeit die Alternativhypothese gilt. Dabei geht man analog, wie eben bei der Güte vor. Der Unterschied ist, dass die Daten entsprechend dem Modell der Alternativhypothese simuliert werden müssen. Durch die relative Häufigkeit der Fälle, in denen die Nullhypothese zu Recht verworfen wurde, wird die Power des Tests geschätzt und auch wieder für jedes Simulationsszenario durchgeführt um verschiedene Parameterkombinationen zu betrachten.
Sehr häufig werden auch Untersuchungen der Eigenschaften eines statistischen Verfahrens bzw. ein Vergleich von mehreren statistischen Verfahren durch Simulationen durchgeführt. Hierbei variiert man systematisch bestimmte Charakteristika der simulierten Daten und untersucht entweder, wie gut die interessierende Struktur in den verschiedenen Situationen durch das bzw. die verschiedenen Analyseverfahren wiedergefunden wird. Oder man weicht in der Datensimulation schrittweise von dem für das Analyseverfahren vorausgesetzte Idealmodell ab und untersucht, wie sich dies auf die Ergebnisse des

untersuchten Verfahrens bzw. der verschiedenen Verfahren auswirkt (Wiencierz, 2010). Aus den Simulationen werden meist die durchschnittlichen Kriteriumswerte bestimmt, z.B. der Bias, der später in Kapitel 5 anhand eines Beispiels näher erläutert wird oder der MSE, die mittlere quadratische Abweichung. Die betrachteten Kriteriumswerte werden für verschiedene Parameterkombinationen variiert.
In der Bayes-Statistik können Informationen über die benötigten Posteriori-Verteilungen, durch eine Monte-Carlo-Simulation bestimmt werden. Eine zentrale Rolle spielen dabei die Algorithmen der Markov-Chain-Monte-Carlo-Simulation (MCMC-Simulation). Mit dem Einsatz von MCMC Methoden, die eine Bayes-Analyse für viele komplexe Probleme erlauben, erfuhr der Bayesianische Ansatz zunehmende Bedeutung und verbreitete Anwendung. Numerische Approximationsverfahren erlauben es ebenfalls, die Posteriori-Verteilung aus komplizierten, auch hochdimensionalen Verteilungen, zu berechnen (Wagner, 2010).
Grundlegend ist zu sagen, dass die Informationen, die sich aus Simulationen bestimmen lassen, viele Bereiche und vor allem die Statistik erheblich vereinfachen. Aufwendige Berechnungen von beispielsweise Approximationen komplexer Verteilungen auf die Normalverteilung, können durch Simulationen bestimmt werden um diese Approximationsregeln dann in der Praxis zu verwenden. Oder auch der eben erwähnte Vergleich statistischer Verfahren, wird in der Praxis sehr häufig verwendet, da Simulationen einen erheblichen Zeitaufwand ersparen. Im Bereich der Statistik, sowie auch in der Ökonometrie und vielen anderen Bereichen sind Simulationen nicht mehr wegzudenken.

3 Probleme in der Anwendung

In diesem Kapitel, soll die bereits zu Beginn kurz erwähnte Problematik näher erläutert werden, indem genauer auf die derzeitige, sehr uneinheitliche Vorgehensweise eingegangen wird.

3.1 Verwendung und derzeitige Praxis

Scientific progress depends on the possibility to verify or falsify results (Kleiber und Zeileis, 2013) Eine Überprüfung vieler verwendeter Simulationen in der Ökonometrie ist derzeit so gut wie unmöglich, da viele Simulationen nicht reproduzierbar und somit auch nicht überprüfbar sind. In den meisten Artikeln wird nur in Textform erläutert, wie die Simulation aufgebaut und durchgeführt wurde. Ohne einen verfügbaren Code ist es nicht möglich, die Richtigkeit der Arbeit zu überprüfen oder auch Arbeiten weiterzuführen und die Simulationen für andere statistische Verfahren bzw. andere Daten zu verwenden. Man kann in den meisten Fällen den datengenerierenden Prozess und vor allem die exakte Verteilung der Daten nicht nachvollziehen.
Ein möglicher Grund, für die Vernachlässigung des detaillierten Simulationsaufbaus, könnten die damit verbundenen Kosten sein. Diese werden meist als höher angenommen, als sie dann tatsächlich wären. Da die Autoren selber über ihr Material verfügen, wäre der Aufwand für die Aufbereitung der Daten, damit sie veröffentlicht werden können, sehr gering im Vergleich zu den daraus entstehenden Vorteilen.

3.2 Archivierung

Um die Problematik zu verdeutlichen, wurden zwei bekannte Journale, von Christian Kleiber und Achim Zeileis (Kleiber und Zeileis, 2013, S. 92) auf die Archivierung untersucht. Das Journal *Journal of Applied Econometrics* (JAE) und das *Journal of Econometrics* (JoE) sind beides bekannte Journale, um wissenschaftliche Arbeiten zu veröffentlichen. Im Unterschied zum JoE, besitzt das JAE ein Datenarchiv, in dem jegliches Material der Autoren bereitgestellt werden könnte.
Es wurden Band 23 des JAE und Band 153 des JoE untersucht.

Tabelle 3.1: Archivierung (Kleiber und Zeileis, 2013, S. 92)

		JAE	JoE
Anzahl Artikel	Gesamt	40	15
	Simulation enthalten	33	14
Anzahl verfügbare Daten	Archiv	31	0
	Proprietäre Software	6	0
	Keine Daten verfügbar	0	12
	Kein Code genutzt	3	3
Anteile aller Artikel	Simulation enthalten	82.5 % (33)	93.3 % (14)
	Software angegeben	65 % (26)	26.7 % (4)
	Bereitgestellter Code	62.5 % (25)	6.7 % (1)
Anteil Simulationen	*replication file* enthalten	30.3 % (10)	7.1 % (1)
	random seed enthalten	15.2 % (5)	7.1 % (1)

Es ist erkennbar, dass die meisten Arbeiten Simulationen verwenden, jedoch sind davon beim JAE maximal 15 der 33 Simulationen reproduzierbar und beim JoE höchstens zwei von insgesamt 14 Simulationen. Nur insgesamt sechs der 55 Artikel nutzen einen *random seed*. Ein Artikel hat explizit angegeben, dass kein *random seed* verwendet wurde. Bei den anderen Simulationen ist es nicht ersichtlich, ob ein *seed* gesetzt wurde oder nicht. In insgesamt 17 Artikeln ist nicht einmal ersichtlich, mit welcher Software gearbeitet wurde.
In der Kategorie *replication files*, die replizierbaren Dateien, sind die Simualtionen zwar reproduzierbar, jedoch kann es sein, dass z.B. der datengenerierende Prozess oder die exakte Verteilung nicht ersichtlich sind, was nicht optimal ist. Zusammenfassend stellten die beiden Autoren fest, dass die verfügbaren Informationen sehr unteschiedlich sind:

- Artikel mit wenigen Informationen zum dazugehörigen Code im Artikel selbst und auch keine Ergänzungen im Anhang
- Artikel mit detaillierten Informationen zum Algorithmus, jedoch ohne vorhandenen Code
- Artikel, die nur kurz auf den Code eingehen, aber nicht erwähnen, dass ein Code vorhanden ist
- Artikel mit einer Referenz auf eine andere Arbeit, deren Codes benutzt wurden

Beim JAE konnten die beiden Autoren den Code manchmal im Archiv finden, teilweise jedoch in gezippten Dateien mit dem Namen *data*, so dass sie den gesamten Inhalt des Archivs inspizieren mussten.
Eine weitere Problematik ist das Verwenden proprietärer Software. Die Lizenzen hierfür

sind meist sehr teuer, weshalb man die Codes solcher Software nur überpüfen kann, wenn man eine solche Lizenz besitzt. Es wäre von Vorteil, wenn Autoren mit öffentlich zugänglicher Software, wie z.B R (R Core Team, 2014) arbeiten, so dass für jeden die Möglichkeit besteht, den Code überprüfen zu können. Zudem kommt hinzu, dass teilweise keine ausreichenden Kenntnisse in statistischer Software bei manchen Autoren vorhanden waren.
Ein auffallender Unterschied zwischen dem JAE und dem JoE ist, obwohl keines der beiden Journale von den Autoren verlangt, Codes aus ihren Artikeln bereitzustellen, die höhere Bereitschaft, wenn ein Archiv vorhanden ist, einen Code zu den Daten mit zu veröffentlichen. Im JoE ist nur für einen Artikel der Code, auf der Webseite eines Autors bereitgestellt. Das Bereitstellen von replizierbaren Details würde ein besseres Verständnis der verwendeten Vorgehensweise ermöglichen. Die einzelnen Schritte können besser nachvollzogen werden und ermöglichen auch eine eventuelle Nutzung der Simulation in einem anderen Kontext. Zusätzlich zum Code, ist es wünschenswert, die Ergebnisse der Simulationen selbst, in Form von Grafiken oder Tabellen immer mit anzugeben. Aus der Untersuchung zogen sie folgende vorläufige Schlussfolgerungen (Kleiber und Zeileis, 2013, 92):

1. Ohne Archivierungsmöglichkeiten besteht nur eine geringe Chance darauf, dass relevantes Material zur Verfügung gestellt wird

2. Archivierungsmöglichkeiten direkt bei den Journalen, würden zu einer Verbesserung führen

3. standardisierte Regeln sollten aufgrund der unterschiedlichen Vorgehensweise, für die Veröffentlichung von dem Code, dem Anhang usw. festgelegt werden

Die Autoren kamen, durch die Untersuchung zu dem Entschluss, dass es zwingend erforderlich ist, dass jedes Journal ein Archiv anbietet, in dem Code archiviert werden kann. Optimalerweise sollte die Archivierung im Prozess der Herausgabe eingebunden sein, um den Inhalt prüfen zu lassen. Die fehlende Archivierung, ist eines der Hauptgründe, weshalb reproduzierbare Simulationen eher die Ausnahme sind. Viele Journale führten in letzter Zeit Archive ein, manche von ihnen sogar extra Archivierungmöglichkeiten für Codes, die man in den Artikeln verwendet hatte. Trotzdem ist die Reproduzierbarkeit oft nicht gewährleistet, da es keine allgemeingültigen Regeln gibt, an die sich publizierende Autoren halten müssen. Darauf soll im nächsten Kapitel eingegangen werden.

4 Richtlinien

Um die derzeit vorhandene und eben beschriebene Problematik in Zukunft zu vermeiden bzw. zu verbessern, wird im Folgenden erläutert, wie ein guter wissenschaftlicher Artikel der Ökonometrie auszusehen hat bzw. wie ein Artikel, der Simulationen beinhaltet, sinnvoll aufgebaut sein sollte.
Die folgenden Richtlinien wurden von Christian Kleiber und Achim Zeileis (Kleiber und Zeileis, 2013) vorgeschlagen und ausgearbeitet.

4.1 Aufbau eines repräsentativen wissenschaftlichen Artikels

Im Artikel *Reproducible Econometric Simulations* werden Richtlinien erläutert, die als Vorschlag dienen sollen:

1. ausführliche Beschreibung des verwendeten statistischen bzw. ökonometrischen Modells
2. technische Informationen
3. Code
4. replizierbare Dateien
5. Ergebnisse inklusive eventueller Zwischenergebnisse

Die Einhaltung dieser Punkte, ermöglicht einen modularen Aufbau, der übersichtlich ist und alles wichtige für eine Veröffentlichung mit reproduzierbaren Simulationen, beinhaltet.

1. ausführliche Beschreibung des verwendeten statistischen bzw. ökonometrischen Modells

Eine ausführliche Beschreibung des verwendeten Modells bildet die Grundlage einer reproduzierbaren Simulation. In dieser sollte die Methodik genaustens erklärt werden. Für komplizierte oder außergewöhnliche bzw. spezielle Techniken sollte der Anhang verwendet werden. Wichtig ist auch, dass immer Angaben zur Genauigkeit gemacht werden, z.B. in Form von Standardfehlern.

2. technische Informationen

Die technischen Informationen sollten in einem eigenen Kapitel näher erläutert werden. Dieses beinhaltet die benutzte Software und die verwendete Software-Version. Die Angabe der Versionsnummer ist sinnvoll und wichtig, da sich z.B. Defaultwerte ändern können.

3. Code

Der dritte Punkt meint jegliche Art von Code. Auch Codes, die aus anderen Arbeiten verwendet werden, sollten referenziert und auch angegeben werden. Der Artikel muss explizit darauf hinweisen, ob und wo Code vorhanden ist. Am besten sollte man jeglichen Anhang, also Codes, Ergebnisse, weitere Dokumentationen etc. im Archiv des jeweiligen Journals verfügbar machen. Sollte kein Archiv vorhanden sein, sollte nicht auf die Dokumentation verzichtet werden, sondern Alternativen wie z.B. eine eigene Webpage in Betracht gezogen werden.

4. replizierbare Dateien

Replizierbare Dateien bzw. *replication files* sollten die exakten Funktionen beinhalten, mit der die Tabellen und Grafiken aus der Arbeit repliziert werden können. Wichtig ist auch, dass der *random seed* enthalten ist, ohne diesen ist keine Reproduzierbarkeit möglich. Wenn das verwendete Softwarepaket verschiedene, bereits programmierte, RNGs anbietet, muss die RNG explizit angegeben werden.

5. Ergebnisse inklusive eventueller Zwischenergebnisse

Die Simulationsergebnisse an sich sollten in einer eigenen Datei gespeichert werden. Auch diese gehören in den Anhang oder sollten archiviert werden.

4.2 Vorteile und Wichtigkeit

Eine genaue und detaillierte Arbeitsweise ist hilfreich um Fragen über technische Details zu klären und auch die Arbeit einfacher und verständlicher zu machen. Der Anreiz für die Autoren selbst sollte sein, dass die Arbeit somit anerkannt werden kann. Die Arbeiten können überprüft und Kritiker von der Richtigkeit überzeugt werden, was grundlegend sein sollte, für eine gute wissenschaftliche Arbeit. Auch eventuelle Fehler, können meist nur durch die Veröffentlichung der dazugehörigen Funktionen, erkannt und somit verbessert werden. Wenn die Ergebnisse als Anhang, bestenfalls in einem Archiv bereitgestellt werden, würde es ausreichen hauptsächlich Grafiken der Ergebnisse im Artikel zu verwenden, welche übersichtlicher sind und meist besser verständlich. Sollten Grafiken verwendet werden, sollte der dazugehörige Code, auch im Anhang zu finden sein.
Auch die Vorteile, die generell durch Simulationen entstehen, sollen kurz erwähnt werden. Die Leistungsfähigkeit der Rechner heutzutage, erlaubt es auch komplexe Simula-

tionen relativ schnell durchzuführen. Im Zusammenhang damit ist die Datengewinnung durch Simulation oft viel kostengünstiger und mit weniger Zeitaufwand verbunden als die herkömmliche Gewinnung von experimentellen Daten im Rahmen von realen Labor- bzw. Feldversuchen. Außerdem lassen sich Computerexperimente unter gleichbleibenden Versuchsbedingungen beliebig oft wiederholen, wohingegen beispielsweise bei naturwissenschaftlichen Experimenten das untersuchte Objekt während der Versuche oft beschädigt bzw. zerstört wird. Ein weiterer Grund für die Nützlichkeit von Simulationen besteht darin, dass der Umfang und die Struktur der zu analysierenden Datensätze oft sehr komplex ist und die Verarbeitung sowie Bewertung der Daten nicht oder nur teilweise mit analytischen Formeln beschrieben werden kann (Pantle, 2003).

5 Beispiel einer reproduzierbaren Simulation

Um die Richtlininen im vorangegangenen Teil an einem Beispiel anzuwenden, wird im Folgenden eine sehr vereinfachte Simulation durchgeführt. Diese soll den modularen Aufbau, sowie die Übersichtlichkeit und vor allem die Wichtigkeit der oben beschriebenen Richtlinien verdeutlichen, um Simulationen reproduzierbar zu machen.

5.1 Schätzverfahren für β-Koeffizienten

Im Folgenden werden normalverteilte Daten simuliert, die anschließend durch ein lineares Modell geschätzt werden sollen, d.h. es gilt $y_i \sim N(x_i'\beta, \sigma_\varepsilon^2)$
Der geschätzte β-Koeffizient $\hat{\beta}_1$ wird anschließend mit dem wahren β-Koeffizienten β_1 durch den Bias, der die Abweichung der beiden Koeffizienten wiedergibt, verglichen.
Die Funktionen wurden mit der Software R (R Core Team, 2014) (Version 0.98.501) geschrieben und werden in den R-Codes 1, 2, 3 bereitgestellt. In R-Code 4 erfolgt der Simulationsaufruf, das Setzen des *seeds* und auch die Angabe des verwendeten RNG, also dem Zufallszahlengenerator. Die letzten beiden R-Codes 5 und 6 ermöglichen es die Simulationsergebnisse in Form einer Tabelle, sowie einer Grafik zu replizieren.
Für die Erstellung der Grafik wurde das Paket *graphics* (version 3.0.3) verwendet. Für umfangreicheren Codes bietet es sich an, die technischen Details, wie z.B. die genutzte Software und die dazugehörige Versionsnummer, wie bereits oben bei den Richtlinien erwähnt, in einem eigenen Kapitel unterzubringen.

Im ersten Schritt werden die Daten zufällig generiert. Der dafür benutzen Funktion `dgp()` werden die Parameter n, der Stichprobenumfang, die Koeffizienten β_0, β_1 und die Varianz σ^2 übergeben. Anschließend werden x und ε als normalverteilte Zufallsvariablen erzeugt, wodurch im nächten Schritt das wahre y erzeugt werden kann. Die Funktion gibt einen data.frame zurück, der die x- und die y-Variablen enthält.

R-Code 1: Datengenerierender Prozess

```
dgp <- function(n, beta0, beta1, sigma2) {

x <- rnorm(n, mean=2, sd=1)

epsilon <- rnorm(n, mean=0, sd=sqrt(sigma2))

y <- beta0 + beta1*x + epsilon

return(data.frame(y=y, x=x))

}
```

Aufbauend auf der Funktion `dgp()`, schätzt im zweiten Schritt die Funktion `estimation()`, das lineare Modell. Durch den übergebenen Paramter `nrep` wird das Verfahren mehrmals wiederholt, in diesem Fall 100 mal, was in der nächsten Funktion festgelegt wird und durch das Dreipunkt-Argument an `estimation()` weitergegeben wird. Die Wiederholungen durch `nrep`, führen dazu, dass die Varianz im Mittel kleiner wird. Es ist wichtig, mehrere Wiederholungen vom selben Szenario durchzuführen, wenn Zufallsvariablen verwedet werden, um im Mittel einen guten Schätzer zu erhalten. Anschließend wird der Bias für jede Wiederholung berechnet und als Mittelwert aller Wiederholungen zurückgegeben.

R-Code 2: statistisches Verfahren

```
estimation <- function(nrep, beta1,...) {

ergebnis_bias<-matrix(nrow=nrep, ncol=2)
bias<-matrix(nrow=nrep, ncol=1)

for (i in 1:nrep) {

mod <- lm(y~x, data=dgp(beta1,...))
ergebnis_bias[i,] <- coefficients(mod)

}

bias[,1] <- ergebnis_bias[,2] - beta1

return(colMeans(bias))

}
```

Die nächste Funktion `simulation()` führt die Simulation durch. Der Funktion, werden die zu variierenden Parameter übergeben. Hier soll die Simulation für drei verschiedene Stichprobenumfänge n und für unterschiedliche Werte des wahren β-Koeffizienten β_1, durchgeführt werden. In der Parameterübergabe wird auch festgelegt, dass β_0=0, die Varianz σ^2=2 und nrep=100 sein sollen. In dieser Funktion wird die Funktion `estimation()` für jede Parameterkombination aufgerufen, die wiederum die datengenerierende Funktion aufruft. Die Parameter werden an die weiteren Funktionen wei-

tergereicht, so dass z.B. der datengenerierende Prozess für alle vorgegebenen Stichprobenumfänge erzeugt wird. Deswegen reicht es, den *seed* im Simulationsaufruf, wie in R-Code 4, zu setzen, da die Funktionen rückwärts nacheinander aufgerufen werden.

R-Code 3: Simulationsdurchführung

```
simulation <- function(n=c(10,100,1000), beta1=c(1,2,3),
                       beta0=0, sigma2=2, nrep=100,...){

parameter <- expand.grid(n=n, beta1=beta1)
nparameter <- nrow(parameter)

ergebnis <- matrix(rep(NA, nparameter), ncol=1)
colnames(ergebnis) <- "bias"

for(i in 1:nparameter) {

ergebnis[i,] <- estimation(n=parameter$n[i], beta1=parameter$beta1[i],
                           beta0=0, sigma2=2, nrep=100,...)

}

parameter_final<-cbind(parameter, ergebnis)
return(parameter_final)

}
```

Der Simulationsaufruf sollte optimalerweise so gestaltet sein, dass durch das Speichern der Simulation in `sc_sim`, die Simulation persistent aufrufbar ist.
`RNGkind()` definiert den Zufallszahlengenerator. In diesem Fall wurden die Defaultwerte verwendet. Für `kind` wird per Default der Mersenne Twister verwendet und für `normal.kind` die Inversionsmethode.
Mersenne Twister ist ein sehr beliebter und häufig genutzter RNG, um gleichvereteilte Zufallszahlen zu erzeugen.
Die Inversionsmethode ist, wie oben bereits kurz erwähnt, ein Simulationsverfahren, um aus gleichverteilten Zufallszahlen andere Wahrscheinlichkeitsverteilungen zu erzeugen. Sie basiert auf der Tatsache, dass man mit der Inversen einer gegebenen Verteilungsfunktion einen Zusammenhang zwischen der Gleichverteilung und der vorgegebenen Verteilung herstellen kann.

R-Code 4: Simulationsaufruf

```
RNGkind(kind="default", normal.kind="default")
set.seed(1234)
sc_sim <- simulation()
```

Nach Durchführung der Simulation, erhält man für jede Parameterkombination den Bias, der aus nachstehender Tabelle (Tabelle 5.1) entnommen werden kann. Da es ein sehr einfaches und knappes Simulationsbeispiel ist, ist der Code sehr übersichtlich. In den meisten Fällen ist es jedoch so, dass viel mehr Daten simuliert werden und die statisti-

sche Auswertung viel aufwendiger ist, weshalb es sich immer empfiehlt, die Ergebnisse zusätzlich als Grafik, wie in R-Code 6, mit anzugeben. Hier sollte auch der modulare Aufbau der Simulation aufgezeigt werden, oft bietet es sich allerdings an, den verwendeten, meist umfangreichen, Code in den Anhang zu stellen und nur die Ergebnisse als Grafik, in dem Artikel zu veröffentlichen.

R-Code 5: Tabelle

```
1 tab <- xtabs(bias~n+beta1, data=sc_sim)
```

	β_1		
n	1	2	3
10	0.109769512	-0.052106790	0.041154562
100	0.003539250	0.030180485	0.015511602
1000	0.002915847	-0.001357494	-0.005232557

Tabelle 5.1: Bias

R-Code 6 ermöglicht es untenstehende Grafik (Abbildung 5.1) zu erzeugen. Man sieht, dass mit wachsendem Stichprobenumfang, der Bias kleiner wird, das heißt je größer n, desto näher liegt der wahre β_1-Koeffizient am geschätzten $\hat{\beta}_1$.

R-Code 6: Grafik

```
1 farbe <- sc_sim$bias
2 farbe[sc_sim$n==10] <- 3
3 farbe[sc_sim$n==100] <- 4
4 farbe[sc_sim$n==1000] <- 5
5 plot(x=sc_sim$beta1, y=sc_sim$bias, col=farbe, pch=19, xlab=expression(beta[1]),
      ylab="Bias")
6 abline(h=0, lty=3)
7 library(graphics)
8 legend(x=2.6, y=0.11, c("n=10", "n=100", "n=1000"), fill=c(3,4,5))
```

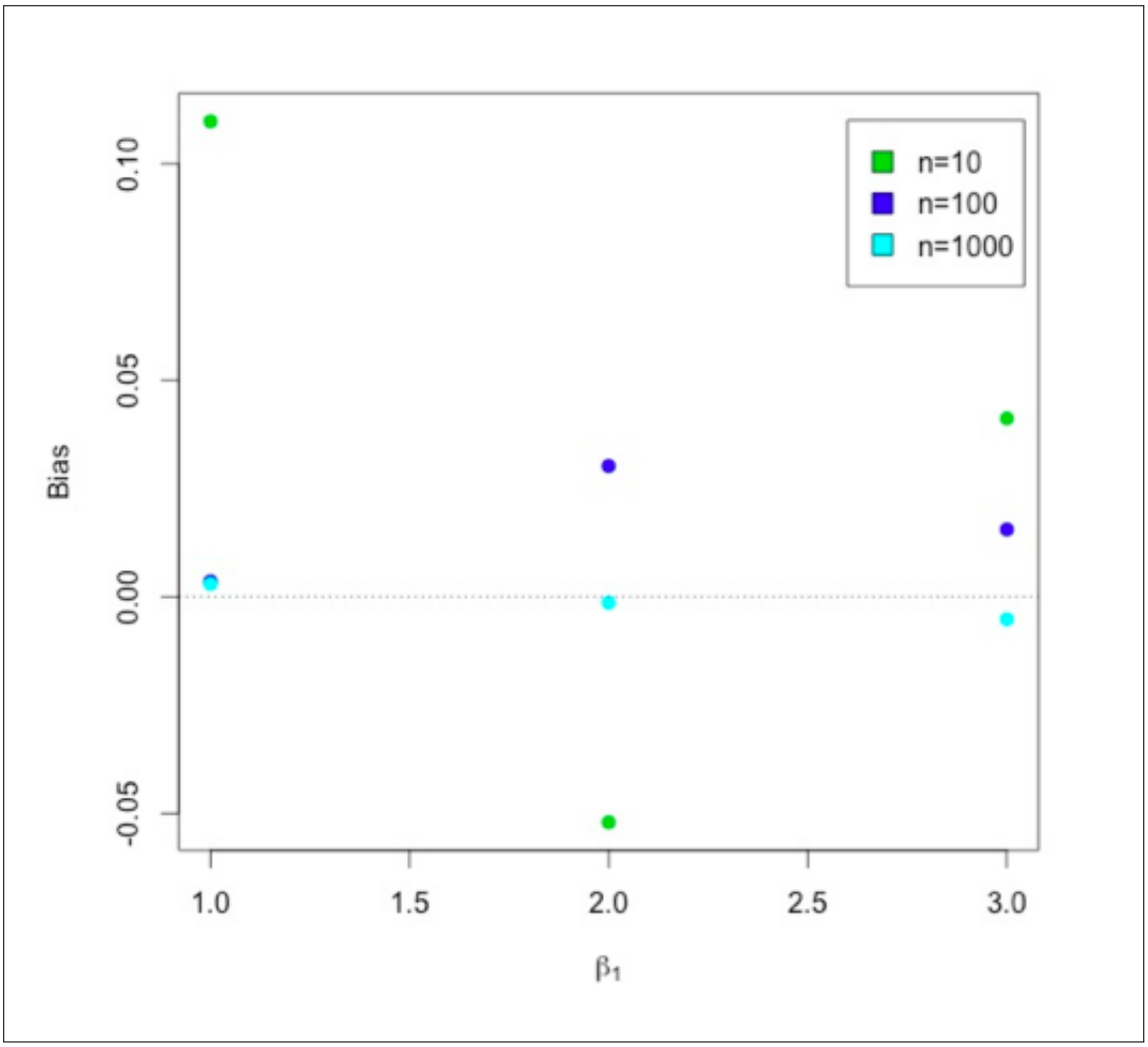

Abbildung 5.1: Schätzung des β_1-Koeffizienten

6 Zusammenfassung und Ausblick

Um die Simultionsergebnisse überprüfen zu können, müssen diese nachvollziehbar und reproduzierbar sein. Im Falle von Computerexperimenten in der Ökonometrie ist dies unter den derzeitigen Umständen nicht möglich, da die dafür benötigten Informationen in den meisten Fällen nicht ausreichend sind. Um diesen, in heutiger Sicht, nicht tragbaren Zustand zu ändern, sind folgende Maßnahmen unumgänglich.
Als Erstes sollten sich Autoren ausreichend über die Voraussetzungen für Reproduzierbarkeit informieren und auch über die nötigen Kenntnisse in der Softwarebenutzung verfügen, um jeden Schritt der Simulation zu erarbeiten und zu verstehen und nicht vorgefertigte Funktionen einer Software benutzen, die man im Nachhinein nicht grundlegend analysieren kann. Man findet nützliche Hinweise zur Benutzung von R in Bezug auf Simulationen im Buch von *Applied Econometrics with R* von Christian Kleiber und Achim Zeileis (Kleiber und Zeileis, 2008), insbesondere in Kapitel 7, Programming Your Own Analysis.
Man sollte als Autor die Qualität nicht vernachlässigen, gewissenhaft arbeiten und immer versuchen, die verwendeten Codes aufzuwerten und zu veröffentlichen, dass sie reproduzierbar sind. Auch wenn keine Richtlinien derzeit vorhanden sind, sollte man an sich selbst hohe Ansprüche haben und alles genauestens aufschreiben und archivieren, auch wenn es nicht verlangt wird.
Ein zweiter wichtiger Punkt ist, dass Journale immer ein Codearchiv zur Verfügung stellen sollten. Entweder zusätzlich zum Datenarchiv oder im Datenarchiv integriert. Idealerweise sind diese Archive im Prozess der Herausgabe eingebunden, so dass der Inhalt kontrolliert werden kann.
Wie oben bereits erwähnt, wird unterschiedliche Software benutzt um Simulationen zu erstellen. Da hier nur R betrachtet wurde, soll abschließend eine Möglichkeit erwähnt werden, die in R zur Verfügung steht und von Vorteil für reproduzierbare Simulationenen ist.
Ein Grund, weshalb Simulationen oft im Nachhinein nicht reproduzierbar sind, ist dass oft neuere Versionen der genutzten R Pakete, nach der Veröffentlichung zur Verfügung stehen. Dabei kann es sein, dass sich z.B. Defaultwerte ändern und man die Simulation nur nachstellen kann, wenn man die Codes entsprechend anpasst, dass sie mit der älteren Softwareversion übereinstimmen. Um dieses Problem zu umgehen, könnte man vom Reproducible R Toolkit (RRT) in R, Gebrauch machen. RRT ist eine Sammlung von R-Paketen, die es ermöglicht, Ergebnisse eines R-Codes im Nachhinein ohne großen Aufwand reproduzierbar zu machen. Die meisten R-Skripte benötigen bestimmte Pakete, die jedoch immer wieder auf eine neuere Softwareversionen aktualisiert werden. Um sicherzustellen, dass Ergebnisse in R reproduzierbar sind, ist es wichtig, R-Codes mit genau dem gleichen Paket und deren verwendeter Version auszuführen, wie zu dem

Zeitpunkt, als der Code geschrieben wurde. R Toolkit bietet im Paket *checkpoint* (Analytics, 2014) eine gleichnamige Funktion namens `checkpoint()` an, die sicherstellt, dass alle notwendigen R-Pakete mit der richtigen Version installiert werden. Hierfür ist es notwendig der Funktion, das Datum zu übergeben, an dem die Simulation geschrieben wurde. Das macht es einfach, die Ergebnisse zu einem späteren Zeitpunkt oder auf einem anderen System zu reproduzieren. Der Checkpoint Server macht täglich eine Kopie aller CRAN Pakete, um sicherzustellen, dass jede Paketversion auch später noch verfügbar ist. (*About Reproducible R Toolkit*, 2010)
Diese Alternative soll eine Möglichkeit aufzeigen, einem repräsentativen Ansatz etwas näher zu kommen.

Literaturverzeichnis

About Reproducible R Toolkit (2010). http://projects.revolutionanalytics.com/documents/rrt/rrtpkgs/.

Analytics, R. (2014). *checkpoint: Install Packages from Snapshots on the Checkpoint Server for Reproducibility*. R package version 0.3.3.
URL: *http://CRAN.R-project.org/package=checkpoint*

Berger, M. und Oberhauser, C. (2014). Programmieren mit statistischer Software, http://www.stablab.stat.uni-muenchen.de/sites/files/ProgStat14_Simulation1.pdf.

Eugster, M. J. A. (2012). Programmieren mit statistischer Software, http://www.stat.uni-muenchen.de/institut/ag/leisch/teaching/progstat12/progstat-folien-09.pdf.

Kleiber, C. und Zeileis, A. (2008). *Applied Econometrics with R*, Springer.

Kleiber, C. und Zeileis, A. (2013). Reproducible econometric simulations, *Journal of Econometric Methods 2013* **2**(1): 89–99.

Pantle, U. (2003). Monte-Caro-Simulation, http://www.mathematik.uni-ulm.de/stochastik/lehre/ss03/markov/skript/node23.html.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *http://www.R-project.org/*

Wagner, P. D. H. (2010). Numerische Methoden der Bayes-Inferenz: Simulationsbasierte Methoden, http://www.statistik.lmu.de/~thomas/Lehre/wise1011/Bayes_1011/material/Bayes7b.pdf.

Wiencierz, A. (2010). Simulationsstudien, http://www.statistik.lmu.de/~thomas/Lehre/abschluss/2010_WS/Tut2_Simulationsstudien.pdf.