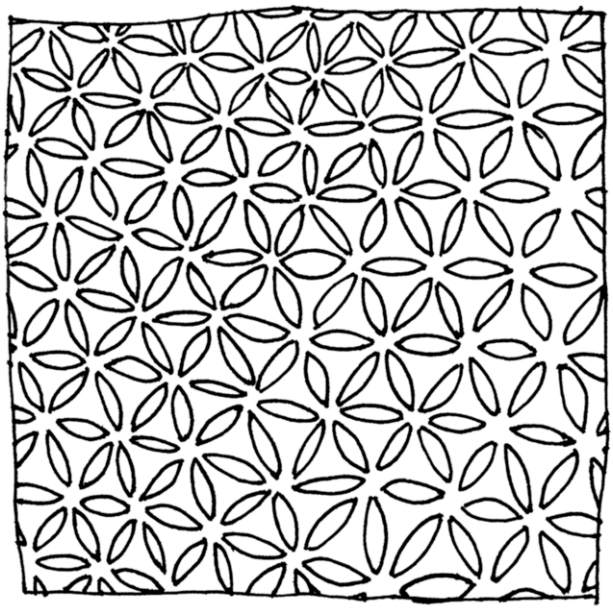
Simulationsverfahren

11



© Der/die Autor(en) 2023

J. Jünger, C. Gärtner, *Computational Methods für die Sozial- und Geisteswissenschaften*, https://doi.org/10.1007/978-3-658-37747-2_11

Zusammenfassung

Dieses Kapitel führt in Computersimulationen ein. Sie lernen die grundlegende Terminologie und verschiedene Arten von Computersimulationen kennen. Anhand eines Beispiels erkunden Sie, wie Agent-Based-Modeling umgesetzt wird.

Im Online-Repositorium unter https://github.com/strohne/cm finden Sie begleitend zum Kapitel weitere Materialien, auf die wir im Text mit ☛ verweisen.

Schlüsselwörter

Agentenbasierte Simulation · Monte-Carlo-Methode · Traceplot · Diffusionsmodelle · Mikro- und Makroebene · R

Das Ganze ist mehr als die Summe seiner Teile[1] – diese auf Aristoteles zurückgehende Feststellung gilt nicht nur für die im vorangegangenen Kapitel vorgestellte Netzwerkanalyse und verdeutlicht, dass sich viele soziale, geistige und kulturelle Phänomene aus dem Zusammenspiel komplexer Prozesse ergeben. Eine Geschichte besteht nicht einfach aus einer Aneinanderreihung von Ereignissen, vielmehr sind die Ereignisse so aufeinander bezogen, dass sich ein Spannungsbogen entwickelt. Auch soziale Systeme sind derart durch das wechselseitige Einwirken von Akteuren geprägt, dass sich daraus etwa die Polarisierung eines politischen Systems oder der öffentlichen Kommunikation ergeben kann. Will man solche Phänomene untersuchen, besteht die Herausforderung darin, Eigenschaften auf der Makroebene (Spannung, Polarisierung) durch Ereignisse auf der Mikroebene (Handlungen, Kommunikation) eines Systems zu erklären. In der wirklichen Welt sind die dahinterliegenden Prozesse in der Regel zu komplex, um sie vollständig zu erfassen – in durch Computersimulationen künstlich erzeugten Welten lassen sich dagegen gezielt einzelne Faktoren isolieren und untersuchen. Computersimulationen erlauben kontrafaktische Experimente. Damit ist gemeint, dass auch Ereignisse untersucht werden können, die in Wirklichkeit noch gar nicht eingetreten sind oder nicht eintreten werden. So ließe sich etwa für eine Videoplatt-

[1] „Das, was in der Weise zusammengesetzt ist, daß das Ganze Eines ist, ist nicht wie ein Haufen, sondern wie eine Silbe. Die Silbe ist aber nicht dasselbe wie ihre Buchstaben, BA ist nicht dasselbe wie B und A, ebenso Fleisch nicht dasselbe wie Feuer und Erde […]“ (Aristoteles 2013, S. 205).

form simulieren, welche Konsequenzen sich aus dem Ein- oder Abschalten des Empfehlungssystems für die Zugriffe auf einzelne Videos ergeben oder wie sich eine stärkere oder geringere Mitteilungsbereitschaft von Akteuren in der gesellschaftlichen Meinungsverteilung niederschlägt. Durch die notwendige Formalisierung von Prozessen zwingen Simulationsmodelle zu einer intensiven theoretischen Auseinandersetzung mit den untersuchten Phänomenen, erweitern dadurch das Verständnis und erlauben schließlich die Erklärung, Vorhersage und Erkundung komplexer Wirklichkeiten (Waldherr et al. 2021, S. 245 ff.).

Grundlegend werden zwei Arten von Simulationen unterschieden:

- **Agent-Based Modeling:** Bei agentenbasierten Simulationen (einführend zum Beispiel Gilbert und Troitzsch 2005) wird eine Welt mit Akteuren erzeugt, die sich im Zeitverlauf entwickelt. Die Akteure können etwa Menschen sein, die Nachrichten lesen und verbreiten. Dazu werden Regeln definiert, nach denen sich die Akteure verhalten. In jedem Schritt, das heißt in einer Epoche, führt jeder Akteur die festgelegten Aktionen aus. Diese Aktionen können etwa darin bestehen, sich zum einen in der Welt zu bewegen und zum anderen immer dann, wenn sich zwei Akteure treffen, Nachrichten weiterzugeben. Am Ende lässt sich beispielsweise auswerten, wie lange die Diffusion der Nachrichten bei unterschiedlichen Regelsätzen gebraucht hat. Zur Interpretation werden unter anderem Traceplots verwendet, auf denen Kennwerte wie die prozentuale Verbreitung einer Nachricht im Zeitverlauf dargestellt werden. Agentenbasierte Simulationen sind zwar abstrakt, wenn die Akteure aber in einer zweidimensionalen Welt platziert werden, lassen sie sich anschaulich wie in einem Computerspiel visualisieren.
- **Monte-Carlo-Simulationen:** Interessiert weniger der Zeitverlauf als das Ergebnis, so lassen sich Probleme als Monte-Carlo-Studien (einführend zum Beispiel Dunn und Shultis 2012) anlegen, deren Ergebnisse alternativ nur umständlich durch Formeln berechenbar wären. Ermittelt werden dabei die Wahrscheinlichkeiten, mit denen ein bestimmtes Ereignis eintritt. So lässt sich etwa bei verschiedenen politischen Wahlverfahren (z. B. Mehrheitswahl, Verhältniswahl) berechnen, mit welcher Wahrscheinlichkeit ein:e Kandidat:in in ein Gremium gewählt wird. Auch Verfahren wie das Topic Modeling (siehe Abschn. 8.2) basieren darauf, die wahrscheinlichste Zuordnung eines Themas zu einem Wort oder Text herauszufinden. Zunächst wird ein (hypothetischer) Datensatz mit den Ausgangsbedingungen definiert, aus dem dann viele Zufallsstichproben gezogen werden (engl. *resampling*). In Bezug auf politische Wahlen ergäbe sich die Wahlwahrscheinlichkeit, dass ein Ereignis ein-

tritt, dann aus der mittleren Wahrscheinlichkeit über viele Ziehungen hinweg. In Bezug auf die Themenstruktur beim Topic Modeling werden ebenfalls eine Vielzahl an möglichen Themenzuordnungen gezogen, um dann die insgesamt am häufigsten aufgetretene Zuordnung als den wahrscheinlichsten Fall anzunehmen.

Wenngleich beide Verfahren andere Perspektiven nahelegen, vereinen Simulationsstudien meist eine regelbasierte und eine wahrscheinlichkeitstheoretische Herangehensweise. Auch agentenbasierte Simulationen laufen nicht deterministisch ab: Welche Richtung ein Akteur auf dem Spielplan einschlägt oder welche Aktion ausgeführt wird, hängt wie bei Würfelspielen häufig von vorgegebenen Wahrscheinlichkeiten ab. Um den typischen Verlauf zu identifizieren, wird eine Vielzahl agentenbasierter Welten mit identischen Startbedingungen simuliert. Effektiv wird dadurch eine Stichprobe möglicher Welten gezogen und die Verteilung der resultierenden Kennwerte wird analysiert, um so etwa die Wahrscheinlichkeit für die Diffusion von Nachrichten zu schätzen. Besonders interessant werden solche Simulationen in Kombination mit Verfahren wie der Netzwerkanalyse, um die Diffusion von Wissen oder auch Viren in einer Gesellschaft zu untersuchen. Letztendlich ist es also eine Frage der Schwerpunktsetzung, für welches Verfahren man sich entscheidet.

In einem allgemeinen Sinn lassen sich alle zufallsbasierten Methoden als Simulationsverfahren begreifen. Auch die Signifikanztests der klassischen Statistik basieren darauf, die bei der Datenerhebung beobachteten Werte mit denjenigen Welten zu vergleichen, die sich bei zufällig aus einer Stichprobe gezogenen Werten ergeben. Dementsprechend finden sich eine Vielzahl an Algorithmen, Packages und Tools, mit denen sich Simulationen durchführen lassen. Zum Erlernen agentenbasierter Simulationen bietet sich beispielsweise Netlogo[2] an, das eine grafische Oberfläche, Beispielmodelle und eine einfach zu erlernende Programmiersprache bereitstellt. Simulationen können durchaus rechenintensiv sein, für die Kombination mit High Performance Computing (Abschn. 6.4) eignen sich Toolkits wie Repast.[3]

Das Grundprinzip einer Computersimulation lässt sich gut mit Skripten in Python (Abschn. 5.2) oder R (Abschn. 5.1) aufbauen und veranschaulichen. Stellen Sie sich dazu eine Welt vor, in der sich 100 Agenten zufällig bewegen. Einer dieser Agenten verbreitet Desinformationen und diese Informationen werden immer dann

[2] Siehe Wilensky (2021; https://ccl.northwestern.edu/netlogo/6.2.1/).

[3] Siehe Collier und North (2013; https://repast.github.io/).

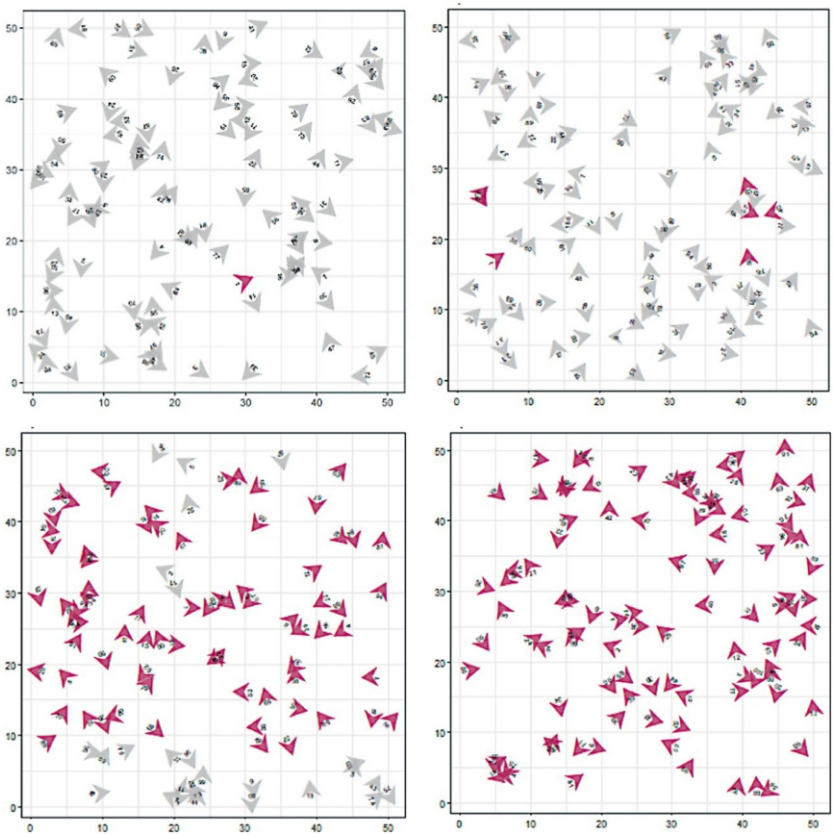


Abb. 11.1 Diffusion in einer künstlichen Welt. Die dunklen Agenten hatten bis zur jeweiligen Epoche mindestens eine Begegnung mit einem anderen dunklen Agenten (☛ *Repositorium*). (Quelle: eigene Darstellung)

weitergegeben, wenn sich zwei Agenten begegnen (Abb. 11.1). Mit diesem sehr einfachen Aufbau lässt sich zunächst modellieren, wie lange und in welcher Art und Weise ein Diffusionsprozess abläuft. Das Modell kann vielfältig interpretiert oder erweitert werden. So kann man dadurch etwa die Verbreitung von Wissen oder die Entstehung von Meinungsverteilungen erkunden. Komplexer werden solche Modelle, wenn verschiedene Themen in Konkurrenz zueinander stehen, Counterspeech eingebaut wird, die Akteure sich nicht zufällig bewegen, sondern untereinander vernetzt sind, oder einzelne Agenten in der Rolle von Journalist:innen eine Multiplikatorfunktion einnehmen.

11.1 Eine Welt entsteht

Eine Simulation lässt sich mit wenigen Schritten in R durchspielen. Legen Sie im ersten Schritt die Rahmenbedingungen der Welt fest, etwa die Breite und Höhe des Feldes, die Anzahl der Agenten auf diesem Feld und die Anzahl der Epochen, über die die Simulation laufen soll:

```
world_width <- 50
world_height <- 50
world_epochs <- 100
world_agents <- 100
```

Auf dieser Grundlage können nun einhundert Agenten zufällig in der Welt verteilt werden. Dazu erstellen Sie einen Dataframe (siehe Abschn. 5.1) mit den Spalten `no` für die von 1 an durchgezählte Nummer des Agenten, `x` sowie `y` für die Position auf dem Feld und `dir` für die Richtung zwischen 0 bis 359 Grad, in die der Agent blickt. In der Spalte `state` wird im Beispiel festgehalten, ob ein Agent die Nachricht wahrgenommen hat oder nicht.

```
agents <- data.frame(
  no   = c(1: world_agents),
  x    = sample(world_width, world_agents,replace=T),
  y    = sample(world_height, world_agents,replace=T),
  dir = sample(360, world_agents, replace=T) - 1,
  state = FALSE
)
```

Um die Agenten zufällig zu initialisieren, wird im Beispiel die Funktion `sample()` verwendet. Der erste Parameter gibt an, aus welchem Wertebereich eine Zahl zufällig gezogen werden soll, der zweite Parameter gibt die Anzahl der benötigten Werte an und der Parameter `replace=T` legt fest, dass auch mehrfach die gleiche Zahl gezogen werden darf, Agenten dürfen also die gleiche Position oder Ausrichtung haben.

Computer erzeugen in der Regel keine echten Zufallszahlen, sondern berechnen ausgehend von einem sogenannten Seed-Wert weitere Zahlen, die sich wie Zufallszahlen verhalten. Daraus ergibt sich der Vorteil, dass Zufallsziehungen und auch Simulationen reproduzierbar sind – setzen Sie vorab mit `set.seed(1852)` Ihre persönliche Lieblingszahl als Startwert!

Nachdem die Welt auf diese Weise erzeugt wurde, wird im Beispiel für den ersten Agenten der `state` auf `TRUE` gesetzt – dieser einzelne Agent beginnt mit der Verbreitung der Information.

```
agents$state[1] <-  TRUE
```

11.2 Die Welt entwickelt sich

Nun kann sich die Welt entwickeln. Um den Verlauf der Entwicklung festzuhalten, legen Sie einen leeren Dataframe `history` an. In einer Schleife, die die Epochen beginnend mit 1 hochzählt, wechseln sich anschließend zwei Vorgänge ab: Nachdem die Agenten ihre Aktionen ausgeführt haben, wird der aktuelle Zustand protokolliert. Im Beispiel wird im Dataframe `agents` die Nummer der Epoche abgelegt, um dann den kompletten Zustand mit `rbind()` an den bisherigen Verlauf anzuhängen. Dadurch wächst der Dataframe `history` mit jedem Durchgang stark an und kann so an die Grenzen des Arbeitsspeichers gelangen – bei umfangreicheren Simulationen würde man sich effizientere Verfahren ausdenken, etwa die Daten in einer Datei ablegen oder nur die wichtigsten Kennwerte der aktuellen Epoche protokollieren.

```
history <- data.frame()

for (epoch in c(1:world_epochs)) {

  print(epoch)

  # Aktionen ausführen
  agents <- agentsMove(agents)
  agents <- agentsChat(agents)

  # Protokollieren
  agents$epoch <- epoch
  history <- rbind(history, agents)
}
```

Die eigentlichen Aktionen sind hier in den Funktionen `agentsMove()` und `agentsChat()` gekapselt. Die erste Funktion nimmt den aktuellen Zustand der Welt entgegen, dreht jeden Agenten zufällig um maximal 25 Grad und bewegt ihn in die entsprechende Richtung. Wenn Sie die Mathematik dahinter nachvollziehen wollen, dann führen Sie die Zeilen schrittweise aus und verfolgen Sie die Ergebnisse:[4]

```
agentsMove <- function(agents) {

  # Zufällig zwischen -25 und +25 Grad rotieren
  rotate <- sample(c(-25:25),world_agents,replace=T)

  agents$dir <- agents$dir + rotate
  agents$dir <- agents$dir %% 360

  # Horizontal bewegen
  dir_x <- round(sin((agents$dir * pi) / 180))
  agents$x <- agents$x - dir_x
  agents$x <- (agents$x - 1) %% world_width + 1

  # Vertikal bewegen
  dir_y <- round(cos((agents$dir * pi) / 180))
  agents$y <- agents$y + dir_y
  agents$y <- (agents$y - 1) %% world_height + 1

  return (agents)
}
```

Die Funktion gibt den neuen Zustand zurück, damit im nächsten Schritt geprüft werden kann, ob ein Informationsaustausch stattfindet. Zunächst werden alle Informationsträger herausgefiltert. Anschließend erhalten alle Agenten den Zu-

[4] Um aus der Gradzahl eine Änderung in der Horizontalen oder Vertikalen zu berechnen, wird diese mit der Funktion $(dir \times \pi)/180)$ in Radiant (rad) umgerechnet. Mittels Sinus und Cosinus wird daraus die Richtungsänderung berechnet. Die mehrfach eingesetzte Modulofunktion `%%` sorgt dafür, dass Agenten beim Erreichen des Spielfeldrandes oder bei einer Ausrichtung von höher als 359 Grad auf die 0 zurückspringen und so auf der anderen Seite wiederkehren, ganz als ob die Welt eine Kugel wäre.

stand TRUE, deren x- und y-Position mit mindestens einem Informanten übereinstimmen:[5]

```
agentsChat <- function(agents) {

  informants <- agents[agents$state == TRUE,]

  agents$state <-
    (agents$x %in% informants$x) &
    (agents$y %in% informants$y)

  return (agents)
}
```

Der Aufbau des Skripts wäre illustrativer möglich, dann aber weniger effizient, indem das Bewegen und Kommunizieren für jeden Agenten einzeln in einer Schleife (siehe Abschn. 5.1) durchgeführt würde – probieren Sie ruhig aus, die Funktionen entsprechend umzuschreiben oder das Beispiel in Python mit Loops nachzubauen!

11.3 Analyse der Ergebnisse

Da der Zustand der Welt in jedem Schritt protokolliert wurde, kann nicht nur das Resultat, sondern der gesamte Verlauf der Simulation nachvollzogen und visualisiert werden. Geht es um die Frage, wie schnell sich Informationen verbreiten, so lässt sich zu jedem Zeitpunkt der Anteil der Informationsträger an allen Agenten berechnen und im Zeitverlauf visualisieren (Abb. 11.2).

Zum Erstellen eines solchen Traceplots wird die Diffusionsrate für jede Epoche berechnet. Der Mittelwert der Variablen state entspricht dem Anteil der

[5] Der Ausdruck mit dem %in%-Operator gibt einen Boolean-Vektor zurück, der so lang ist wie die Anzahl der Agenten. Die Kombination der beiden Vektoren über & sorgt dafür, dass im Ergebnisvektor nur diejenigen Werte TRUE werden, auf die beide Bedingungen zutreffen. Das betrifft auch die Informanten selbst, die damit den Zustand TRUE behalten.

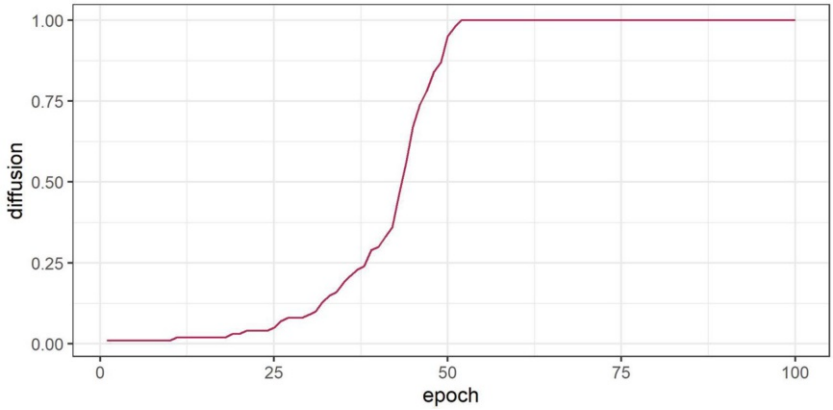


Abb. 11.2 Veränderung der Diffusion im Zeitverlauf (traceplot). (Quelle: eigene Darstellung)

`TRUE`-Werte, da Boolean-Werte bei der Berechnung automatisch zu 0/1 umkodiert werden:

```
trace <- history %>%
    group_by(epoch) %>%
    summarise(diffusion = mean(state)) %>%
    ungroup()
```

Das Ergebnis ist ein Dataframe mit den zwei Spalten `epoch` und `diffusion`. Mit *ggplot* kann daraus ein Liniendiagramm erstellt werden:

```
trace %>%
    ggplot(aes(epoch, diffusion)) +
    geom_line(color = "maroon")
```

Grundsätzlich bietet es sich an, alle Teilschritte der Simulation in eigene Funktionen auszulagern. So können leicht verschiedene Varianten der Diffusion ausprobiert werden. Da alle Epochen mitgeloggt wurden, können Sie dem Weltwissen beim Wachsen zuschauen und den Prozess sogar animieren (☛ *Repositorium*).

Man erkennt an den Skripten schnell, dass es sich um abstrakte Welten handelt. Das Handeln von Menschen wird über die Bearbeitung eines Datensatzes simuliert. Diese Art von Modellen findet sich häufiger in der Biologie, etwa um genetische Evolution zu analysieren. Evolutionsideen haben darüber hinaus auch Eingang in die Sozial- und Geisteswissenschaften gehalten, beispielsweise

wenn es um die Verbreitung von Memes (Dawkins 1976) oder Informationen geht. Das Ergebnis demonstriert einen Grundmechanismus der Informationsverbreitung und vieler anderer sozialer, geistiger, biologischer und physikalischer Prozesse: exponentielles Wachstum. Trotz der extrem geringen Ausgangswahrscheinlichkeit, dass zwei Agenten aufeinandertreffen und einer dieser beiden Agenten zudem noch die Information trägt, verbreitet sich die Information innerhalb von kurzer Zeit in der gesamten Welt. Schaffen Sie es, das exponentielle Wachstum mit veränderten Regeln (Vergessen), zusätzlichen Dingen (konkurrierende Mitteilungen, Netzwerkverbindungen) oder weiteren Eigenschaften der Agenten (Agilität, Wahrnehmungskapazität) und Mitteilungen (Neuigkeitswert) zu stoppen?

Übungsfragen

1. Wozu werden Simulationsmodelle eingesetzt?
2. Was unterscheidet agentenbasierte Simulationen von Monte-Carlo-Simulationen?
3. Was ist mit den Begriffen „Epoche“ und „Traceplot“ gemeint?
4. Installieren Sie Netlogo und implementieren Sie damit das im Kapitel vorgestellte Modell (Hinweis: in der Model Library finden Sie ein ähnliches Modell in der Kategorie *Biology/Evoplution/Genetic Drift/* unter dem Namen *GenDrift T interact*, das Sie entsprechend abwandeln können)!

Weiterführende Literatur

Dunn, W. L. & Shultis, J. K. (2012). *Exploring Monte Carlo methods*. Amsterdam: Academic Press.

Gilbert, G. N. & Troitzsch, K. G. (2005). *Simulation for the social scientist* (2. Aufl.). Maidenhead: Open University Press.

Snijders, T. A. (1996). Stochastic actor-oriented models for network change. *The Journal of Mathematical Sociology, 21*(1–2), 149–172. https://doi.org/10.1080/0022250X.1996.9990178.

Literatur

Aristoteles. (2013). *Metaphysik. Schriften zur ersten Philosophie*. Stuttgart: Reclam.

Collier, N. & North, M. (2013). Parallel agent-based simulation with Repast for High Performance Computing. *Simulation, 89*(10), 1215–1235. https://doi.org/10.1177/0037549712462620

Dawkins, R. (1976). *The selfish gene*. New York: Oxford University Press.

Dunn, W. L. & Shultis, J. K. (2012). *Exploring Monte Carlo methods*. Amsterdam: Academic Press.

Gilbert, G. N. & Troitzsch, K. G. (2005). *Simulation for the social scientist* (2. Aufl.). Maidenhead: Open University Press.

Waldherr, A., Hilbert, M. & González-Bailón, S. (2021). Worlds of Agents: Prospects of Agent-Based Modeling for Communication Research. *Communication Methods and Measures*, *15*(4), 243–254. https://doi.org/10.1080/19312458.2021.1986478

Wilensky, U. (2021). NetLogo (Version 6.2.1) [Computer software]: Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. https://ccl.northwestern.edu/netlogo/6.2.1/

Open Access Dieses Kapitel wird unter der Creative Commons Namensnennung 4.0 International Lizenz (http://creativecommons.org/licenses/by/4.0/deed.de) veröffentlicht, welche die Nutzung, Vervielfältigung, Bearbeitung, Verbreitung und Wiedergabe in jeglichem Medium und Format erlaubt, sofern Sie den/die ursprünglichen Autor(en) und die Quelle ordnungsgemäß nennen, einen Link zur Creative Commons Lizenz beifügen und angeben, ob Änderungen vorgenommen wurden.

Die in diesem Kapitel enthaltenen Bilder und sonstiges Drittmaterial unterliegen ebenfalls der genannten Creative Commons Lizenz, sofern sich aus der Abbildungslegende nichts anderes ergibt. Sofern das betreffende Material nicht unter der genannten Creative Commons Lizenz steht und die betreffende Handlung nicht nach gesetzlichen Vorschriften erlaubt ist, ist für die oben aufgeführten Weiterverwendungen des Materials die Einwilligung des jeweiligen Rechteinhabers einzuholen.