

AGENT-BASED MODELLING & GEOGRAPHICAL INFORMATION SYSTEMS

A PRACTICAL PRIMER



Andrew Crooks
Nick Malleson
Ed Manley
Alison Heppenstall



AGENT-BASED MODELLING & GEOGRAPHICAL INFORMATION SYSTEMS

In the digital age, social and environmental scientists have more spatial data at their fingertips than ever before. But how do we capture this data, analyse and display it, and most importantly, how can it be used to study the world?

Spatial Analytics and GIS is a series of books that deal with potentially tricky technical content in a way that is accessible, usable and useful. Early titles include *Urban Analytics* by Alex Singleton, Seth Spielman and David Folch, and *An Introduction to R for Spatial Analysis and Mapping* by Chris Brunsdon and Lex Comber.

Series Editor: Richard Harris

About the Series Editor

Richard Harris is Professor of Quantitative Social Geography at the School of Geographical Sciences, University of Bristol. He is the lead author on three textbooks about quantitative methods in geography and related disciplines, including *Quantitative Geography: The Basics* (Sage, 2016).

Richard's interests are in the geographies of education and the education of geographers. He is currently Director of the University of Bristol Q-Step Centre, part of a multimillion pound UK initiative to raise quantitative skills training among social science students, and is working with the Royal Geographical Society (with IBG) to support data skills in schools.

Books in this Series:

An Introduction to Big Data and Spatial Data Analytics in R

Lex Comber & Chris Brunsdon

An Introduction to R for Spatial Analysis and Mapping, 2nd Edition

Chris Brunsdon & Lex Comber

Geocomputation, Chris Brunsdon & Alex Singleton

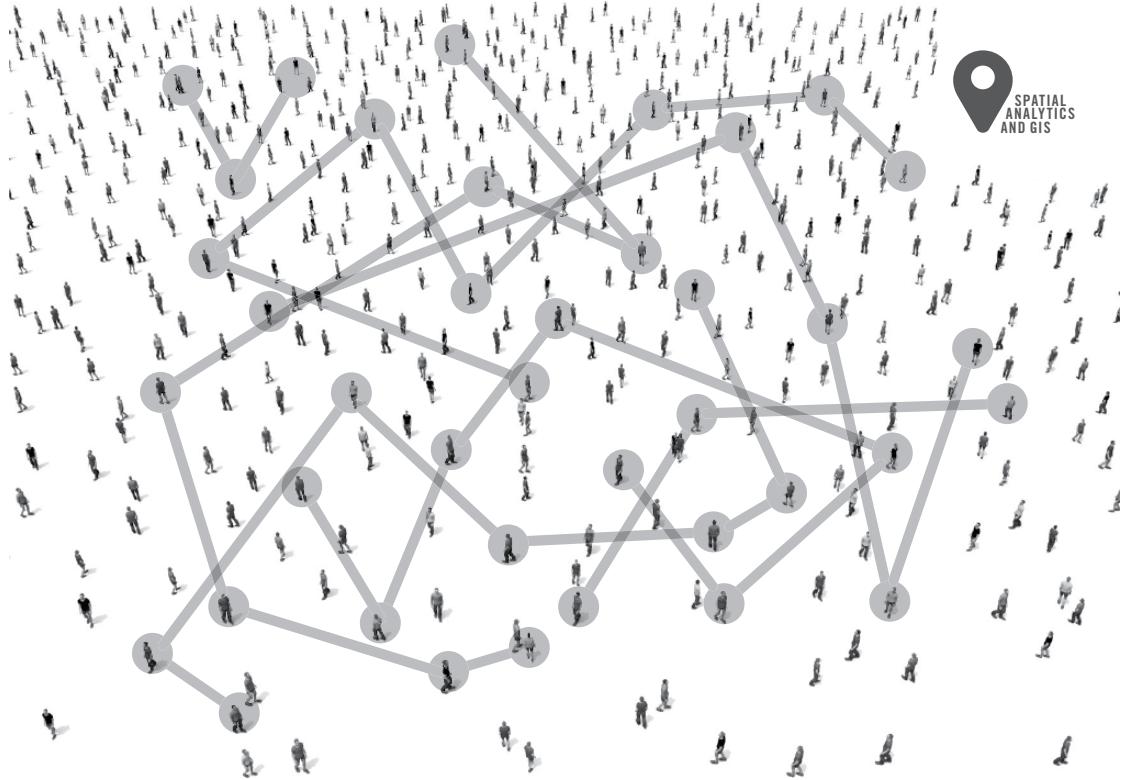
Agent-Based Modelling & Geographical Information Systems

Andrew Crooks, Nicolas Malleson, Ed Manley & Alison Heppenstall

Modelling Environmental Change, Colin Robertson

Published in Association with this Series:

Quantitative Geography, Richard Harris



AGENT-BASED MODELLING & GEOGRAPHICAL INFORMATION SYSTEMS

A PRACTICAL PRIMER

Andrew Crooks
Nick Malleson
Ed Manley
Alison Heppenstall

 SAGE

Los Angeles | London | New Delhi
Singapore | Washington DC | Melbourne



Los Angeles | London | New Delhi
Singapore | Washington DC | Melbourne

SAGE Publications Ltd
1 Oliver's Yard
55 City Road
London EC1Y 1SP

SAGE Publications Inc.
2455 Teller Road
Thousand Oaks, California 91320

SAGE Publications India Pvt Ltd
B 1/I 1 Mohan Cooperative Industrial Area
Mathura Road
New Delhi 110 044

SAGE Publications Asia-Pacific Pte Ltd
3 Church Street
#10-04 Samsung Hub
Singapore 049483

Editor: Robert Rojek
Assistant editor: John Nightingale
Production editor: Katherine Haw
Copyeditor: Richard Leigh
Proofreader: Sunrise Setting Ltd
Indexer: David Rudeforth
Marketing manager: Susheel Gokarakonda
Cover design: Wendy Scott
Typeset by: C&M Digitals (P) Ltd, Chennai, India
Printed in the UK

© Andrew Crooks, Nick Malleson, Ed Manley and
Alison Heppenstall 2019

First published 2019

Apart from any fair dealing for the purposes of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act, 1988, this publication may be reproduced, stored or transmitted in any form, or by any means, only with the prior permission in writing of the publishers, or in the case of reprographic reproduction, in accordance with the terms of licences issued by the Copyright Licensing Agency. Enquiries concerning reproduction outside those terms should be sent to the publishers.

Library of Congress Control Number: 2018945199

British Library Cataloguing in Publication data

A catalogue record for this book is available from the British Library

ISBN 978-1-4739-5864-7
ISBN 978-1-4739-5865-4 (pbk)

At SAGE we take sustainability seriously. Most of our products are printed in the UK using responsibly sourced papers and boards. When we print overseas we ensure sustainable papers are used as measured by the PREPS grading system. We undertake an annual audit to monitor our sustainability.

CONTENTS

<i>List of Figures</i>	xii
<i>List of Tables</i>	xxi
<i>Preface</i>	xxii
<i>List of Acronyms</i>	xxiv
<i>About the Authors</i>	xxv
<i>Foreword</i>	xxvi
1 Agent-Based Modelling and Geographical Information Systems	1
1.1 Introduction	1
1.2 Complexity and Geographical Systems	2
1.3 Models	5
1.4 Data	7
1.5 Individuals	7
1.6 Agent-Based Modelling and Geographical Information Systems	9
1.7 Outline of the Book	10
1.8 Annotated Bibliography	12
2 Introduction to Agent-Based Modelling	14
2.1 Introduction	14
2.1.1 What Is an Agent?	15
2.1.2 Agent Rules	18
2.1.3 An Agent's World	19
2.2 Advantages of Agent-Based Modelling	20
2.3 Limitations of Agent-Based Modelling	22
2.4 A Gallery of Applications	23
2.4.1 Segregation	24
2.4.2 Sugarscape	26
2.4.3 Transportation Modelling	28
2.4.4 Decision-Making	30
2.5 Discussion	31
2.6 Annotated Bibliography	33

CONTENTS

3 Designing and Developing an Agent-Based Model	35
3.1 Introduction	35
3.2 Overview	37
3.2.1 Purpose and Process	38
3.2.2 Data Collection and Evaluation Plan	39
3.2.3 Development and Software	40
3.2.4 Visualisation	40
3.2.5 Biases, Uncertainty and Assumptions	41
3.3 World	42
3.3.1 External Systems	42
3.3.2 Space	43
3.3.3 Time	45
3.3.4 Populations	45
3.3.5 Physical Rules	46
3.4 Interactions	46
3.4.1 Physical Interactions	46
3.4.2 Communication	48
3.4.3 Resource Exchange	48
3.5 Agents	49
3.5.1 Characteristics	49
3.5.2 Decisions	50
3.5.3 Actions	51
3.6 Building a Segregation Model	52
3.6.1 Segregation Model: Overview	52
3.6.2 Segregation Model: World	54
3.6.3 Segregation Model: Interactions	56
3.6.4 Segregation Model: Agents	56
3.7 Other Frameworks and Approaches	58
3.7.1 Pattern-Orientated Modelling	58
3.7.2 Overview, Design Concepts and Details (ODD) Protocol	58
3.8 Discussion	59
3.9 Annotated Bibliography	60
4 Building Agent-Based Models with NetLogo	62
4.1 Introduction	62
4.2 NetLogo Basics	63
4.2.1 The NetLogo Program	65
4.2.2 Contexts: Observer, Turtle, Patch (and Link)	68
4.2.3 The ask Command	70

CONTENTS

4.3	First NetLogo Model	73
4.3.1	Creating the World	74
4.3.2	Buttons and Procedures	75
4.3.3	Sliders and Variables	77
4.3.4	Creating Turtles and Patches	78
4.3.5	Making the Model go	80
4.4	Advanced NetLogo Model	83
4.4.1	Advanced Variables	84
4.4.2	Grass Grows ...	87
4.4.3	Giving Birth	89
4.4.4	Creating a Graph	91
4.5	Annotated Bibliography	93
5	Fundamentals of Geographical Information Systems	95
5.1	Introduction	95
5.2	A (Very Brief) History of GIS	96
5.3	Representing the World	98
5.3.1	Raster Data	99
5.3.2	Vector Data	101
5.4	Time in GIS	102
5.5	GIS Software	103
5.6	Sources of Geographical Data	105
5.6.1	Volunteered and Ambient Geographical Information	106
5.6.2	Social Media Data	106
5.6.3	Remotely Sensed Data	108
5.7	Preparing GIS Data Using QGIS	110
5.7.1	Performing Spatial Operations	111
5.7.2	Manipulating Vector Table Data	113
5.8	Visualising Results	117
5.8.1	Map Types	117
5.8.2	Map Elements	119
5.8.3	Styling Trends	120
5.8.4	Interactivity	121
5.9	Discussion	122
5.10	Annotated Bibliography	124
6	Integrating Agent-Based Models and GIS	125
6.1	Introduction	125
6.2	Coupling and Embedding GIS and Agent-Based Models	127

CONTENTS

6.3 Tools for Constructing and Developing Agent-Based Models	129
6.3.1 Swarm	132
6.3.2 MASON	132
6.3.3 Repast	132
6.3.4 NetLogo	135
6.3.5 GAMA	137
6.4 Integrating GIS Data into Agent-Based Models	138
6.4.1 Using Raster Data in NetLogo	139
6.4.2 Using Vector Data in NetLogo	149
6.5 Exporting Data	165
6.6 Discussion	169
6.7 Annotated Bibliography	170
7 Modelling Human Behaviour	172
7.1 Introduction	172
7.2 The Challenge of Simulating Human Behaviour	174
7.3 Behavioural Frameworks	176
7.3.1 Types of Behavioural Frameworks	177
7.3.2 Challenges	180
7.4 Mathematical Approaches	180
7.4.1 Probabilistic Models	181
7.4.2 Threshold Models	182
7.5 Conceptual Cognitive Models	183
7.5.1 Beliefs-Desires-Intentions Model	183
7.5.2 Fast and Frugal Model	184
7.5.3 Physical Conditions, Emotional State, Cognitive Capability and Social Status Model	186
7.6 Case Study: Simulating Consumer Behaviour Using Probabilistic Rules	187
7.7 Case Study: Simulating Behaviour in Riots Using a Cognitive Model	189
7.8 Discussion	190
7.9 Annotated Bibliography	192
8 Networks	194
8.1 Introduction	195
8.2 Basic Network Properties	196
8.2.1 Defining Graphs Mathematically	196
8.2.2 Building Graphs in NetLogo	197
8.2.3 Adjacency Matrices and Node Degree	198

CONTENTS

8.2.4	Traversing Graphs	199
8.2.5	Graph Density	200
8.2.6	Calculating Node Importance	201
8.3	Social Networks	203
8.3.1	Random Networks	204
8.3.2	Small-World Networks	206
8.3.3	Scale-Free Networks	207
8.4	Transport Networks: Agents Navigating a Road Network	208
8.5	Linking Geographical and Social Networks	217
8.6	Discussion	223
8.7	Annotated Bibliography	224
9	Spatial Statistics	226
9.1	Introduction	227
9.2	Hypothetical Data	228
9.3	Goodness of Fit with Global Statistics	230
9.4	Visual Comparisons	233
9.5	Description of the Properties of Point Data	235
9.6	Local Indicators of Spatial Association (LISA)	237
9.6.1	Dual KDE	237
9.6.2	GI^*	237
9.7	Multi-scale Error Analysis	239
9.8	Discussion	241
9.9	Annotated Bibliography	242
10	Evaluating Our Models: Verification, Calibration, Validation	244
10.1	Evaluating Models: An Overview	244
10.2	Verification	245
10.2.1	Code Testing	246
10.2.2	Simplifying Environments	247
10.2.3	Expected Outcome Alignment	250
10.2.4	Docking	250
10.3	Calibration	251
10.3.1	Qualitative Calibration	253
10.3.2	Quantitative Calibration	254
10.3.3	Quantitative Calibration Example: WalkThisWay	259
10.4	Validation	261
10.5	Discussion	261
10.6	Annotated Bibliography	263

11 Alternative Modelling Approaches	265
11.1 Introduction	265
11.2 An Overview of Different Modelling Approaches	266
11.2.1 Cellular Automata	266
11.2.2 Microsimulation	268
11.2.3 Discrete Event Simulation	270
11.2.4 System Dynamics	271
11.2.5 Spatial Interaction Models	272
11.3 Comparing Different Modelling Approaches	274
11.4 A Practical Comparison: The SIR Model	276
11.4.1 The System Dynamics Approach	276
11.4.2 The Agent-Based Approach	276
11.4.3 The Cellular Automata Approach	279
11.4.4 The Discrete Event Simulation Approach	279
11.4.5 Comparative Analysis	280
11.5 Discussion	283
11.6 Annotated Bibliography	283
12 Summary and Outlook	285
12.1 Introduction	285
12.2 Remaining Challenges	286
12.2.1 Reasons for Modelling	286
12.2.2 Theory and Models	287
12.2.3 Inter-model Comparison	287
12.2.4 Replication and Experiment	288
12.2.5 Verification and Validation	289
12.2.6 Agent Representation, Aggregation and Dynamics	290
12.2.7 Behaviour	290
12.2.8 Sharing and Dissemination of the Model	291
12.2.9 Data Challenges	292
12.3 Looking Ahead	294
12.3.1 Big Data and Agent-Based Modelling	295
12.3.2 Model Integration	298
12.3.3 Uncertainty and Ensembles	303
12.3.4 Data Assimilation	304
12.3.5 Spatially Learning Agents	304
12.4 Discussion	305
12.5 Annotated Bibliography	306

CONTENTS

<i>Appendices</i>	309
<i>A Gallery of Applications</i>	311
A.1 Disease Dynamics in a Refugee Camp	311
A.2 Hiker Movements in the Dolomites UNESCO World Heritage Site	313
A.3 Modelling the Emergence of Riots	313
A.4 The Foothill Yellow-Legged Frog Assessment Model	314
A.5 Understanding Artificial Anasazi	314
A.6 The Spread of Agriculture during the Neolithic Period	315
A.7 WalkThisWay	317
A.8 Natural Disasters and Humanitarian Relief	318
A.9 Using Social Media Content to Inform Agent-Based Models for Humanitarian Crisis Response	319
A.10 Modelling Transportation and Development for Reston,VA	320
A.11 Agent-Based Modelling for Community Resource Management	320
A.12 Agent-Based Modelling of Conflict Diamonds	322
A.13 Exploring the Growth of Slums	323
A.14 RiftLand: Analysing Conflict, Disasters and Humanitarian Crises in East Africa	325
A.15 Modelling Forced Migration	326
A.16 Studying Coupled Human-Artificial-Natural Systems in Boreal and Arctic Regions	327
<i>References</i>	329
<i>Index</i>	368

LIST OF FIGURES

1.1	A simple hierarchical structure of a city composed of multiple neighbourhoods which form a hierarchy at the more macro level but also have interactions (e.g. commuter flows) with each other	5
2.1	(A) A human agent and (B) a supermarket retailer agent and their representation within an object-orientated environment	18
2.2	Schematic illustrating how emergence occurs through individual autonomy and interaction between agents	19
2.3	Conceptualisation of an agent-based model where people are connected to each other and take actions when a specific condition is met	20
2.4	The growth in agent-based modelling: from search results of Web of Science and Google Scholar	24
2.5	Progression of segregation over time: agents want to live in a neighbourhood where 40% are of the same colour	25
2.6	Examples of how changing agents' neighbourhood preference levels leads to different patterns of segregation emerging	26
2.7	SugarScape wealth distribution model	27
2.8	Simple traffic model where each car is an agent. In (A), (B) and (C) from top left clockwise: model parameters, a chart of car speeds and the spatial agent environment	29
2.9	User interface of SimTable: (A) entire study area; (B) an active fire model	31
3.1	Logic of simulation	36
3.2	An overview of the typical modelling process	38
3.3	Different forms of spatial representations that may be used in agent-based models	44
3.4	The visualisation tools used in the Schelling model as implemented in NetLogo	53
3.5	The structure of the ODD+D protocol. Grey boxes indicate new design concepts/categories compared to the ODD protocol. The numbers of added new questions are noted in parentheses. The different aspects of the new design concept of 'individual decision-making' are displayed on the right	59

LIST OF FIGURES

4.1	The NetLogo Segregation model	64
4.2	The main components of the NetLogo interface	66
4.3	NetLogo contexts: the observer, patches and turtles	69
4.4	Examples of some of the variables that the NetLogo contexts (observer, patches and turtles) have access to	69
4.5	Using <code>ask</code> to make all patches go blue: <code>ask patches [set pcolor blue]</code>	71
4.6	A visual representation of the use of <code>ask</code> and <code>with</code>	72
4.7	A picture of the environment configuration	74
4.8	How to create a button	75
4.9	Creating a <code>setup</code> button	76
4.10	A NetLogo button and its associated code	77
4.11	Configuring a NetLogo slider	78
4.12	An example of the model interface once the sheep and grass have been created	81
4.13	The speed slider can be used to slow down or speed up the rate that the model iterates	82
4.14	The final version of the simple model; sheep move around and turn green patches to brown	83
4.15	An illustration of the final model that will be completed in Section 4.4	84
4.16	Creating a slider to set the grass regrowth rate	88
4.17	The graph that shows the number of agents in the model (y -axis) at each iteration (x -axis)	91
4.18	Configuring a plot that will graph the number of alive turtles at each iteration	92
 5.1	 An example of (A) a vector and (B) a raster representation of wards (administrative units) in the Greater London Authority area	 99
5.2	An example of reading in raster data and creating a landscape: (A) the original ASCII file from a GIS; (B) the resulting space created in NetLogo	100
5.3	The basic building blocks of vector data: points, lines and polygons. Different types of objects can be combined to represent the geometric properties of an area	101
5.4	Using GeoDa to create a .gal file: (left) a collection of polygons (objects); (right) the .gal file where each polygon records its neighbours	104
5.5	Examples of volunteered geographical information: (A) Maps.Me; (B) SeeClickFix; (C) Mapillary	107
5.6	Geosocial analysis: linking places, people and events	108

LIST OF FIGURES

5.7	Mining tweets relating to entertainment and contained within and near the Theatre Subdistrict of New York City	110
5.8	An example of using open data, in this case data from Natural Earth and WorldPop, for a 10×10 km area near the city of Makindye in Uganda	111
5.9	An example of using open data for the initialisation of an agent-based model	112
5.10	An example of how to load and ‘clip’ vector and raster data files using QGIS	114
5.11	Instructions for opening the attribute table for a vector layer using QGIS	115
5.12	How to add a new column to the attribute table that underpins a vector layer using QGIS and calculate new values for the column	116
5.13	A dot density map of income distributions in Winnipeg, Canada, constructed from 2006 Census data	118
5.14	Map from Demšar et al. (2018) demonstrating strong map design, through inclusion of a legend, scale bar, supporting labels, north arrow and accessible colour design	120
5.15	A screenshot of the TasP trial dashboard installed at the Africa Health Research Institute in Kwa-Zulu Natal, South Africa, which enables interactive exploration of global and local trends in HIV rates, and combines hexagon-based choropleth mapping with supporting figures, graphs and text-based notes	122
6.1	Abstracting from the real world to a series of layers to be used in the artificial world upon which the agent-based model is based	126
6.2	A selection of MASON spatial models: (A) agents (red) exiting a building based on raster data and the resulting trails (yellow); (B) an urban growth model where red areas represent new developments; (C) a Schelling type of model using census areas in Washington, DC as its spatial environment; (D) agents (red circles) moving along on sidewalks (grey lines)	133
6.3	Examples of vector agent-based models in Simphony: (A) agents (red stars) moving along on sidewalks (grey lines); (B) an agent-based model overlaid on NASA WorldWind	134
6.4	A selection of geographically explicit agent-based models utilising NetLogo: (A) rain (blue) falls and flows to a lower elevation based on a digital elevation model captured in 2 and 3D; (B) agents	

LIST OF FIGURES

(white) moving along on sidewalks (orange); (C) Schelling type model using census areas in Washington, DC as their spatial environment; (D) commuting along a road network	136
6.5 GAMA platform: (A, B) advanced visualisation of simulations; (C) built-in charting and functions; (D) user interface	137
6.6 A SLEUTH-type model based on growth rules and coefficients as outlined in Clarke et al. (1997). In this example, the model is simulating the rates of land-use change in Santa Fe, New Mexico	140
6.7 A basic SLEUTH-type model based on growth rules and coefficients driving the rates of land-use change in Santa Fe, New Mexico implemented in NetLogo: (A) initial conditions (blue cells); (B) urban growth (red cells)	141
6.8 A simple pedestrian evacuation model where agents are exiting a room: (A) initial starting conditions where the red dots represent agents and the blue represent a cost surface; (B) pedestrian paths when the door is one cell wide; (C) pedestrian paths when the door is two cells wide	143
6.9 Simple pedestrian model: CAD floor plans of a building (A) are converted into a raster layer (B) and are used as the environment for the agent-based model; (C) shows the simulation running with agents (red) who are exiting the building and leaving behind walking traces (grey)	144
6.10 A simple hydrological model in which rain falls from the sky and flows downhill: (A) initial starting conditions where the blue dots represent agents (i.e. water) and the underlying digital elevation model; (B) over time rain accumulates in Crater Lake; (C) water depth at time step 500 and (D) the corresponding water; (E) Crater Lake has enough water for it to discharge	146
6.11 Setting the NetLogo patch size so that it corresponds with some vector GIS data	152
6.12 The Schelling segregation model implemented using polygons as agents	152
6.13 Reading in a shapefile and creating agents	162
6.14 (A) Instantiation of a segregation model where each agent is created from attribute information and (B) the resulting end state when all agents are satisfied with their neighbourhood	163
6.15 An example of taking attribute data from a model and then exploring it in a GIS. Here, agents are exported from a RepastJ model of segregation as a point shapefile. Their locations are then used to calculate the population density using ArcGIS	166

LIST OF FIGURES

<p>6.16 An example of taking attribute data from the NetLogo Schelling model and displaying it in a GIS. Agents are exported from the Schelling model as a .txt file at the start and end of the simulation. They are displayed in a GIS. Initially there are 1320 agents distributed over 118 polygons (of which 75 are blue, 75 are red, 38 are unoccupied). Each agent has a preference to be in an area where 60% of their neighbours are of the same colour. Over the course of the simulation this results in agents moving and altering the population and colours of the polygons (114 blue, 58 red, 16 unoccupied)</p> <p>6.17 Exporting raster data from a simple urban growth NetLogo model and visualising it in a GIS for further analysis. In this example the model ran for 100 time steps and the area urbanised went from 21% to 92%</p> <p>7.1 In relation to a range of specific example ABMs; above, models are plotted according to spatial and temporal scales, and below behavioural complexity is plotted against environmental complexity</p> <p>7.2 Resulting patterns of segregation from different threshold levels</p> <p>7.3 Heuristic model of route choice: (A) breakdown of space into a hierarchy, differentiating regional, node-based and road segment-based decision-making; (B) an example route choice process, where different heuristic rule sets are engaged at each level of the hierarchy</p> <p>7.4 Basic spatial environment created within NetLogo that the consumer agents occupy</p> <p>7.5 A high-level representation of the resident agent behaviour incorporated into the PECS models</p> <p>8.1 From left to right, a basic graph, a social network and a simplified transportation network</p> <p>8.2 The seven bridges of Königsberg problem: (A) the physical depiction of the bridges; (B) the graph representation</p> <p>8.3 Directed and undirected links as displayed in NetLogo. These examples are from the Undirected Network Demo and Directed Network Demo in the accompanying resources</p> <p>8.4 Adjacency matrix for undirected and directed graphs</p> <p>8.5 Different network structures based on 25 nodes, with 0.2 probability of a link: (A) random; (B) small-world; (C) scale-free</p> <p>8.6 Growth of a scale-free network via preferential attachment</p> <p>8.7 Network representation within a GIS, with two types of networks: walkways and rivers</p>	<p>167</p> <p>168</p> <p>175</p> <p>182</p> <p>185</p> <p>188</p> <p>189</p> <p>195</p> <p>196</p> <p>197</p> <p>199</p> <p>205</p> <p>208</p> <p>209</p>
---	---

LIST OF FIGURES

<p>8.8 A road network imported into NetLogo, from the GMU-Roads model</p> <p>8.9 Metadata from Twitter tweets (top) and Flickr photos (bottom)</p> <p>8.10 Schematic diagram illustrating how different networks can be linked together. From physical (L_1, L_2), and social (L_3) spaces through to deriving place abstractions (L_4) for different locations (N_1, N_2)</p> <p>8.11 Agents and Networks: (A) the physical environment; (B) the agents' social (cyber) networks</p> <p>8.12 Agents and networks related to going to and from work: (A) initial social-network-based home locations; (B) evolution of the social network over time as agents interact with each other</p> <p>9.1 Overview of the statistics and methods that will be introduced in this chapter</p> <p>9.2 Hypothetical point data used to illustrate the use of statistics in this chapter: two hypothetical model results and an observed ('real') data set. The points are shown in the top row, and in the bottom row are the number of points in each administrative area (dark-red areas have more points)</p> <p>9.3 The results of two hypothetical models (A and B) plotted against observed data. The lines of best fit ($y = x$) illustrate the locations of 'perfect' model results. In this example, model B is a better fit to the observed data. Note that the graphs only illustrate values of $x > 50$ for clarity</p> <p>9.4 Visual comparisons of point pattern similarity: (top) mapping raw points; (middle) aggregating points to an administrative area; (bottom) calculating point density with kernel density estimation</p> <p>9.5 Ripley's K with distance ($K(d)$) for the three example data sets. Model B data has a $K(d)$ value that is more similar to the observed data than model A, suggesting a closer fit</p> <p>9.6 The difference between observed point densities and simulated point densities, calculated using kernel density estimation. Red areas highlight negative values (model under-predictions), blue areas highlight high values (model over-predictions)</p> <p>9.7 Getis-Ord GI^* hotspots</p> <p>9.8 Fuzzy maps produced by the multi-scale error analysis process. Darker blue colours indicate areas where the model results show the greatest divergence from the observed data</p> <p>9.9 The total error between the observed data and model B at different resolutions</p>	<p style="margin-top: 0;">210</p> <p style="margin-top: 0;">218</p> <p style="margin-top: 0;">221</p> <p style="margin-top: 0;">221</p> <p style="margin-top: 0;">222</p> <p style="margin-top: 0;">228</p> <p style="margin-top: 0;">229</p> <p style="margin-top: 0;">230</p> <p style="margin-top: 0;">234</p> <p style="margin-top: 0;">237</p> <p style="margin-top: 0;">238</p> <p style="margin-top: 0;">238</p> <p style="margin-top: 0;">240</p> <p style="margin-top: 0;">241</p>
--	---

LIST OF FIGURES

10.1	Inspecting the current state of an agent, and watching it evolve during the simulation	247
10.2	Simplified types of environment in the Rainfall model that can be used to test the behaviour of the agents (rain droplets)	248
10.3	An overview of the process of calibration	252
10.4	NetLogo's BehaviorSpace interface	255
10.5	A normalised heat map generated from real trajectory data of people's movements	260
10.6	The WalkThisWay model, reimplemented in NetLogo	260
11.1	Diagrammatic representation of 2D cellular automata and the most commonly used neighbourhoods: (1) von Neumann neighbourhood; (2) Moore neighbourhood; and extended (3) von Neumann and (4) Moore neighbourhoods	266
11.2	Example of cells changing state from dead (white) to alive (black) over time depending on the states of their neighbouring cells	267
11.3	Schematic outlining the basic process of creating a synthetic population using microsimulation	268
11.4	A simple example of dynamic microsimulation process where individuals change over time (m = married, d = divorced, w = widowed)	269
11.5	Simple example of a discreet event simulation	270
11.6	Airport security line simulation	271
11.7	Basic elements of a system dynamics model	272
11.8	An example of a wolf-sheep predation model implemented as a system dynamics model	273
11.9	System dynamics process	277
11.10	System dynamics flowchart	277
11.11	Agent-based modelling: agent decision process	278
11.12	Display of the agent-based model: green = susceptible, red = infected, blue = recovered	278
11.13	Cellular automata cell changing process	279
11.14	Display of the CA model: green = susceptible, red = infected, blue = recovered	280
11.15	Discrete event simulation process	280
11.16	Results for the different models. Clockwise from top left: SD model, agent-based model, DES and CA	281
11.17	Results for the different models with infection rate 0.02. Clockwise from top left: SD model, agent-based model, DES and CA	282
12.1	Agent-based models running in a web browser: (A) the Segregation model in NetLogo Web; (B) the Droplet model where agents flow down an elevation service in Agentscript	293

LIST OF FIGURES

<p>12.2 A sample instantiation of the Haiti model, where the roads were initialised from OpenStreetMap data; the blue dots represent aid centres and the red dots represent agents who are seeking out food</p> <p>12.3 Hotspots of activity of Twitter users: tweet locations and associated densities for a selection of prolific users</p> <p>12.4 Tweet sentiments between 16 June and 12 July for the Waldo Canyon wildfire, log-scaled to give a sense of the change of rank</p> <p>12.5 A sample of application domains for agent-based models for geographical systems</p> <p>12.6 (A) System structure; (B) system hierarchy; and (C) related subsystems/processes for urban systems</p> <p>12.7 A ‘spaghetti plot’ illustrating pressure contours as forecast from numerous models in an ensemble. In (A), the results across different models are similar and hence the forecast has relatively low uncertainty. In (B), however, the results vary substantially in some areas so the forecast is less certain</p> <p>A.1 Graphical user interface of the model. From top left clockwise: legend, simulation display, health status of individual camp dwellers, agents’ activities, household size composition, parameter settings and a clock recording the time</p> <p>A.2 Graphical user interface of the model. From left to right: input parameters, spatial environment and output statistics such as the amount of traffic and modal split</p> <p>A.3 Graphical user interface of the riot model based on the Kibera slum of Nairobi. From top left clockwise: spatial environment, input parameters, legend and summary of residents’ activities</p> <p>A.4 Graphical user interface of the FYFAM based on South Fork Trinity River site in California</p> <p>A.5 Graphical user interface of the Understanding Artificial Anasazi model</p> <p>A.6 Exploring the origins of agriculture in the Iberian peninsula during the Neolithic period</p> <p>A.7 Graphical user interface of the WalkThisWay pedestrian model. Clockwise from top left: spatial environment, charts recording average walking speed, crowd density and model parameters</p> <p>A.8 Graphical user interface of the humanitarian relief model. Clockwise from top left: the spatial environment and its various layers; charts recording agent deaths, food levels and densities at various aid centres and agent activities; model information</p>	<p style="margin-top: 0;">296</p> <p style="margin-top: 0;">297</p> <p style="margin-top: 0;">298</p> <p style="margin-top: 0;">299</p> <p style="margin-top: 0;">300</p> <p style="margin-top: 0;">303</p> <p style="margin-top: 0;">312</p> <p style="margin-top: 0;">313</p> <p style="margin-top: 0;">314</p> <p style="margin-top: 0;">315</p> <p style="margin-top: 0;">316</p> <p style="margin-top: 0;">317</p> <p style="margin-top: 0;">318</p> <p style="margin-top: 0;">319</p>
--	---

LIST OF FIGURES

A.9 Graphical user interface of the Hotspots model showing the spatial environment; red dots denote agents	320
A.10 Graphical user interface of the Reston commuter model. From left to right: input parameters, visual display and model outputs during a representative model run	321
A.11 Graphical user interface of the acequia-based agriculture model showing the spatial environment, including land use, and charts recording the number of farmers (parciantes) and urban-versus-agricultural percentages during a representative model run	322
A.12 Graphical user interface of the Geography of Conflict Diamonds model showing the spatial environment centred on Sierra Leone and charts recording agent activities (e.g. working, mining, rebelling) and the model parameters during a representative model run	323
A.13 Graphical user interface of the Slumulation model, including input parameters (left), the spatial environment based on Ahmedabad, India (centre), and various output parameters (right)	324
A.14 The RiftLand model of parts of East Africa: (A) the modelled region, where yellow to green represents vegetation, blue is fresh water, grey is the Indian Ocean and brown are nature reserves; (B) an example of decision-making of the household herding model; (C) a zoomed-in section of the model where individual herders and farmers are shown	325
A.15 An agent-based model of forced migration: top: the spatial environment, where lines represent migration routes, and nodes represent numbers of migrants. Purple nodes represent final destinations, red nodes show migrant deaths and green nodes show migrants en route. Bottom: charts recoding average financial status of the migrants and the number that have died en route	326
A.16 Graphical user interface of the NorthLands model. Clockwise from left: the spatial environment showing the region of Canada with population distribution in the year 1921, parameter settings and an output time series measuring rural (red) and urban (blue) households	327

LIST OF TABLES

1.1 Key terms used in complexity theory	3
1.2 NetLogo models introduced throughout the book, often with accompanying code examples	11
4.1 The main components of the NetLogo interface	66
4.2 An explanation of the code to make the grass regrow. Code Example 4.4.2 illustrates the full code	89
5.1 Sample of GIS type data, representing a layer of data in a GIS	103
5.2 A sample of GIS platforms	105
5.3 Some examples of online geographical data sources	105
6.1 Comparison of coupling approaches	128
6.2 A selection of open-source agent-based modelling toolkits for creating geographically explicit models	131
7.1 The five main dimensions for distinguishing agent architectures	177
7.2 Overview of the key assumptions and application areas of popular theories used in representing behaviour	178
7.3 Summary of customer group characteristics	187
8.1 Pairs through the network	202
9.1 An example of a table that stores the number of points per area. This can be used to generate goodness-of-fit statistics	231
9.2 The values of the non-spatial statistics when applied to hypothetical example data	232
9.3 The average NNI values for the three example data sets. The data for model B is closer to the observed data, suggesting a better fit	236
11.1 A comparison of modelling techniques used to study aspects of geographical systems	275
12.1 Comparison of sentiment classifications of real tweets	299

PREFACE

This book has emerged through years of discussions, comments and questions we have received about building agent-based models and integrating geographical information. When we started developing agent-based models (some time ago) there were very few textbooks, tools or resources available. While this has changed over the last two decades as the popularity of agent-based modelling has grown, we felt there was still a need to provide a resource that helps students and researchers gain an understanding of how to incorporate and consider geography and geographical information in the building of agent-based models.

We approached this book from two standpoints. First, to provide a synthesis of the underpinning ideas, techniques and frameworks for integrating agent-based modelling and geographical information systems. Second, building on our experiences of teaching at various levels, to provide a practical set of information for the development of agent-based models for geographical systems.

From these two standpoints we have developed a book that provides a practical primer in the integration of agent-based modelling and geographical information systems. In outlining the subject we will cover many examples of geographical phenomena, from linking the individual movements of pedestrians to aggregate patterns of urban growth, to the integration of social networks into modelling mobility. Through this text, we hope the reader will understand how the field has developed, how agent-based models are different from other modelling approaches, and the future challenges we see lying ahead. Through use of sample code (all of which can be found on the accompanying website www.abmgis.org) we provide the reader with many of the basic building blocks for constructing agent-based models linked to geographical information systems. Throughout the book we use the software package NetLogo (Wilensky, 1999), providing an easy route into learning about agent-based modelling through its ease of use and relatively shallow learning curve.

We believe there is an exciting future for agent-based modelling within the geographical context – we hope this book convinces you of the same, and inspires many more models of the future!

We would like to thank the following people who have helped us along the way with words of advice, encouragement and support (in alphabetical order): Peggy Agouris, Rob Axtell, Mike Batty, Mark Birkin, Tao Cheng, Claudio Cioffi-Revilla,

PREFACE

Arie Croitoru, Tomas Crols, Tim Gulden, Andy Hudson-Smith, Andrew Jenkins, William Kennedy, Sean Luke, Ron Mahabir, David O’Sullivan, Amit Patel, Dieter Pfoser, Denise Pumain, Jacek Radzikowski, Anthony Stefanidis, Qing Tian, Paul Torrens, Sarah Wise, Andreas Zufle. An extra special thank-you must go to Yang Zhou who helped develop many of the NetLogo models used throughout this book. Alison’s biggest thanks go to her Doctor. We should also thank WhatsApp and Skype for helping us maintain contact at all hours of the day and night (even if we didn’t want to).

A special thank-you also goes to SAGE, and in particular John Nightingale and Robert Rojek, who have been immensely patient with us as we battled competing work and life priorities alongside the writing of this book.

Finally, and most importantly, the authors would like to thank their families who have suffered in various ways during the writing of this book – Sophie, William and Matthew, Megan, Toby and Jacob, Lianna and Greta, Elspeth, Iseabail and Iona.

Andrew Crooks

Department of Computational and Data Sciences and
Department of Geography and GeoInformation
Science, George Mason University, USA

Nick Malleson

Centre for Spatial Analysis and Policy at the School of
Geography, University of Leeds, UK

Ed Manley

Centre for Advanced Spatial Analysis (CASA),
University College London, UK

Alison Heppenstall

Centre for Spatial Analysis and Policy at the School of
Geography, University of Leeds, UK

LIST OF ACRONYMS

AGI	Ambient geographical information
BDI	Beliefs–desires–intentions
CA	Cellular automata
CCTV	Closed-circuit television
DEM	Digital elevation model
DES	Discreet event simulation
Exif	Exchangeable image file
GAMA	GIS Agent-based Modelling Architecture
GIS	Geographical information system(s)
GPS	Global positioning system
GUI	Graphical user interface
JSON	JavaScript Object Notation
KIDS	Keep it descriptive, stupid
KISS	Keep it simple, stupid
MASON	Multi Agent Simulation Of Neighbourhood
MSM	Microsimulation models
ODD	Overview, Design concepts, and Details
ODD+D	Overview, Design concepts, and Details + Decision
PECS	Physical conditions, emotional states, cognitive capabilities and social status
PB	Petabyte
POM	Pattern-oriented modelling
Repast	Recursive Porous Agent Simulation Toolkit
SIR	Susceptible–infected–recovered
VGI	Volunteered geographical information

ABOUT THE AUTHORS

Andrew Crooks is an Associate Professor in Computational Social Science with a joint appointment between the Department of Computational and Data Sciences and the Department of Geography and GeoInformation Science at George Mason University. His areas of expertise specifically relate to integrating agent-based modelling and geographic information systems (GIS) to explore human behaviour. Moreover, his research focuses on exploring and understanding the natural and socio-economic environments specifically of urban areas using GIS, spatial analysis, social network analysis, social media and agent-based modelling methodologies. More information about this work can be seen at www.gisagents.org.

Nick Malleson is an Associate Professor in Geographical Information Science at the School of Geography, University of Leeds, and a member of the Centre for Spatial Analysis and Policy (CSAP). His research is interdisciplinary and centres on the development and application of spatio-temporal computational models in the social sciences, with a particular focus on crime simulation and modelling. More recently, he has been conducting research that explores the nature of massive ‘crowdsourced’ data in the social sciences and related modelling approaches. More information about this work can be seen at <http://nickmalleson.co.uk/>.

Ed Manley is an Associate Professor at the Centre for Advanced Spatial Analysis (CASA), University College London. His research involves measuring and modelling the individual and collective behaviours that shape urban dynamics, drawing on new streams of data and simulation methods. Ed is particularly interested in advancing data-driven agent-based modelling in order to improve the understanding and prediction of complex social systems. More information about this work can be seen at www.urbanmovements.co.uk.

Alison Heppenstall is Professor of Geocomputation in the School of Geography, University of Leeds and an associate of Leeds Institute for Data Analytics. She is an expert in the development of spatial agent-based models with a focus on understanding and simulating behaviour. Her current interests are concerned with linking agent-based models to artificial intelligence and machine learning methodologies, the role of big data in creating more robust agent-based models, and the relationship of agent-based modelling to social theory. More information can be found at <https://alisonheppenstall.co.uk>.

FOREWORD

Half a century ago, the key problem in economics was how to reconcile the micro with the macro. Micro-economics had developed around theories of how resources were allocated by firms to production and by individuals to consumption. The theory of the firm was largely based on how to choose resources so that profits might be maximised while the theory of the individual (or household) was based on how to choose resources that maximised income or utility. Elegant formal frameworks for finding equilibria between these models of supply and demand were developed but it was widely recognised that these individualistic theories were largely ‘thought’ experiments that might help us understand how a hypothetical rather than real economy might function.

In parallel to the micro, macro-economics developed to treat whole aggregates of individuals and firms expressed as population, employment, total income, revenue, GDP and so on. These defined the dynamics of an economy where various econometric models were used to simulate and predict the way such aggregate economies might develop with respect to different economic policies. The link between macro and micro was tentative to say the least with one of the key problems – the ‘aggregation problem’ – widely regarded as being intractable in and of itself. Micro did not scale to macro and vice versa because the focus and language in which the two related approaches were formally developed, was irreconcilable despite an urgent and ever pressing concern for integrated thinking.

At the same time, a new geographic science was in the making, drawing on many of these traditions in economics but attempting to reconcile the culture and economy of human settlements with a science of their dynamics. The so-called quantitative revolution in geography, despite its pretensions, threw up key problems of representation. Initially, problems were articulated from an aggregate point of view in that geographic space was partitioned largely into homogeneous units, often assumed to be aggregations of some finer geographic representation. Individuals specified by locations were aggregated to larger spatial units such as neighbourhoods and districts in cities, to entire towns and regions, and such like groupings. This logic was based on the untested but widely accepted notion that true understanding could only come from applications of the law of large numbers and the notion that we could pursue individual explanation was regarded as flawed. But as these ideas developed, a counter force emerged in which individual

behaviour was identified as being a more important determinant of how geographic phenomena should be represented than had hitherto been assumed, thus echoing the aggregation problem in economics. The same degree of intractability in dealing with individual and aggregate behaviours and structures continues to this very day to characterise geography, particularly geographic information science and human spatial behaviour.

In the years from when these different viewpoints concerning aggregate and disaggregate approaches were most clearly articulated, the emergence of a style of thinking that defines the system of interest, not in aggregates formed from individual populations but with respect to individual, autonomous entities at many different levels of sectoral or spatial aggregation, has come to the fore. These are defined as *agent-based approaches* mainly expressed through *agent-based models* or *modelling* (ABM). These models tend to cut across the old distinctions between aggregate and disaggregate, macro and micro, but also introduce time into their formulation, something that was largely disregarded all those years ago when the aggregation problem came to the forefront in economics and geography. Agent-based modelling first came onto the agenda about 20 years ago in response to critiques concerning aggregation with ABM tending to circumvent these difficulties by proposing much more flexible, less formal frameworks for simulation. These incorporated human spatial behaviours into models that operate in time as well as with respect to several different varieties of entities and objects expressed as agents functioning in time and space. In fact, agent-based modelling is largely characterized by the fact that agents can be defined as any item or object that is sufficiently autonomous to have some kind of behavioural dynamics attached to its function. In this sense, then, ABM is more pragmatic than most of the models developed hitherto in the spatial sciences in that the theory to which they might be applied is often implicit with the user or model-builder defining what they consider the most appropriate model might be.

This book is all about ABM as it is applicable to spatial systems, that is, to cities, regions, national and global entities, as well as fine-scale patterns of movement and location down to the level at which individuals move within buildings and very local spaces. It is thus a book about *spatial ABM* but in developing these ideas, readers will encounter a generic approach which is applicable to systems other than those that are purely spatial. Indeed, although the approach espoused here is practical and empirical, it is even possible to use these ideas to test and develop formal and non-empirical theories of the kind that micro-economics has traditionally worked with. However, in adopting the standpoint of ABM, the authors have to dwell on what an agent is and how its behaviour can be operationalised. In this sense, this book is about how one might articulate and apply basic theories of human behaviour, particularly in space, and this gives the book a very special flavour. There is nothing comparable so far in my view and although

the platform that the authors use to illustrate their models – Netlogo – is very obviously spatially-based in that all the action takes place on a grid or some transformation thereof, the focus here is on models that have been defined in many other ways. This enables the authors to show how models built in other styles, such as spatial interaction and systems dynamics, can be reconciled with ABM.

The book has a very clear structure. Chapters 1 to 4 define the general notion that ABM is associated with the wider field of complexity theory where there is an emphasis on how systems evolve and change from the bottom up. ABM are bottom-up type models and although this kind of modelling may not be exclusively from the bottom up in that there may be top and intermediate level down actions, ABM of course is consistent with systems where novelty and surprise emerge. Thus this defines the general context of complexity theory that the authors develop in the first chapter. If ABMs are consistent with notions about emergence, heterogeneity, path dependence and related features of complex systems, it is imperative that we define what an agent actually is, and this the authors do in Chapter 2. Ideas about autonomy in terms of behavioural purpose, adaptation and learning as well as functional activity and interactions all define the essence of what an agent is and does and although there is no single easy-to-characterise definition, this chapter makes it very clear what agents are and what they are not. Moreover, the idea of rule-based behaviour is spelt out at this point in the book and various useful vignettes are liberally used to illustrate these different points.

The process of model design which is elaborated in the third chapter is a blow by blow account of how a model emerges from data and is then tested in various ways. A particularly important point emphasised here is that when an ABM is designed, it is essential to grasp the notion of how much complexity needs to be modelled. This, to an extent, relates to how parsimonious an ABM might be, for it is easy to keep on adding agents, especially as the systems that are dealt with here tend to be extremely data-hungry with respect to the number of agents and their attributes. Chapter 4 in fact provides a detailed description of how agent-based models are designed using the Netlogo language and software. This is particularly useful for by now, any reader who has absorbed the preliminaries can begin to design their own models. The range of examples and ideas in this chapter are very substantial and it represents a fitting conclusion to the first part of the book that defines what ABM is all about.

GIS (geographic information systems) are discussed in the next chapter and this too represents a useful primer that complements, in many ways, the simple spatial focus of the earlier chapters. That the ABMs in this book are manifestly spatial is clear from the word go, but only when we reach this chapter do we see the kind of spatial detail that is necessary to urban models being imported into the various software that enables professional applications to be developed. One might argue that Netlogo is a platform that is largely educational but, in fact, very

serious applications can be made with this software and what is nice about the treatment here is that any reader reaching this point will feel that this is the case. GIS is defined with respect to the age-old distinction between vector and raster spatial representations and the general ideas of building up pictures of geographic systems through layers of different activity is writ-large with respect to how geographic data might be imported into an ABM. All of this is a prelude to illustrating in Chapter 6 how ABM and GIS might be merged and this is done with respect to illustrating how the main packages related to ABM have taken on the challenge: here the reader is pointed to Repast, MASON, GAMA and Netlogo which are four widely known software systems that embrace GIS data and functionality in different but related ways.

No book on ABM would be complete without a foray into how human behaviour can be defined formally and in Chapter 7 an attempt is made to relate the somewhat pragmatic definitions of behavioural rules which dominate the various examples in the book with more considered views about theories of formal behaviours. For example, prospect theory, ideas about cognition, thresholds and learning, and experimental exploration are implied here and although only hints as to how these ideas might influence ABM are made, a path forward to enriching such models with much more detailed studies of human behaviour is charted. The same is true of networks in the next chapter and although Netlogo deals quite extensively with how networks might be embodied in ABM model frameworks, this chapter provides some very useful ideas about network science which can be made operational fairly quickly within most packages which enable ABMs to be built. Spatial statistics follow in Chapter 9 where the focus is on how various interpolations and segregation statistics can be mapped, such as those based on kernels and local statistics. These are all relevant to network and GIS functionality, the subjects of the previous two chapters.

Chapter 10 is an essential but thorny one that deals with verification, validation and calibration. Many ABMs are not calibrated as such for their data requirements are often immense while it is often hard to get good individual data to fit them to real systems. Moreover, ABMs invariably deal with dynamics, and temporal behavioural data is frequently lacking for such systems. Nevertheless, this is a nice summary of what is needed and it points the reader towards empirical validation while making a very clear distinction between how one might verify that the code of any model is correct. In ABM, this is as important as the notion of maximising the goodness of fit of the model to data. There are pointers here to ideas about data assimilation which represent the cutting edge of such modelling, while in the last chapter, comparisons are made between different modelling approaches. The models chosen can all be transformed into agent-based approaches, thus showing how ABM is much more generic than many alternative modelling styles such as spatial interaction, systems dynamics, microsimulation and cellular automata modelling.

FOREWORD

The book concludes with a gallery of applications which are briefly explained in a series of appendices.

If you have read this far, my general point is that ABM has come of age and this book is the best example of this to date. If you persevere through the chapters that follow, you will get as good a grounding as can be had in this newer style of simulation modelling that has rapidly emerged over the last twenty years and now finds itself at the cutting edge of social simulation, big data and the study of real human behaviour. Furthermore, although the focus in this book is on spatial and geographical applications, readers will find that there is enough here to enable them to think out-of-the-box so-to-speak and to adapt these tools to a much wider cornucopia of applications in the spatial and non-spatial sciences.

Michael Batty
CASA, University College London

1

AGENT-BASED MODELLING AND GEOGRAPHICAL INFORMATION SYSTEMS

Chapter Outline

The overarching aim of this chapter is to give the reader a contextual background and general overview of the major developments in geographical modelling for the simulation of the individual. The reader is introduced to a discussion around the purpose of modelling as well how complexity theory has influenced the way that we view (and simulate) geographical systems. We end the chapter by discussing the benefits of bringing together agent-based modelling and GIS.

1.1 Introduction

Geographers have always been interested in the role and influence of the individual within geographical systems. What are the consequences of individual behaviours and decision-making over space and time? Until relatively recently, answering this question has been beyond the reach of researchers due to a lack of appropriate data and the methods of both analysis and simulation to explore geographical systems. The last twenty years have witnessed an explosion in computer processing power and storage and micro-level data sets, accompanied by the diffusion of ideas from complexity science. This has led to new ways of thinking about and simulating geographical systems. This is clearly evidenced through the development and uptake of agent-based modelling. This approach puts the individual at the centre of the simulation. Through careful interrogation of new data sources, researchers can construct individuals who are endowed not only with basic characteristics drawn from quantitative data sets, such as age and sex, but also with more qualitative aspects such as opinions and preferences. As well as

richer sources of data on individuals, these new forms of data are often spatially referenced, thus giving a greater insight into the interplay between individual and space. This is where technologies such as geographical information systems (GIS) have a role to play. These tools are well established and give the researcher the ability to manipulate and process large quantities of diverse geographical data. However, these systems are largely limited in their inability to handle dynamic processes. Almost all GIS analysis uses snapshots (i.e. static) of data. For the geographical researcher, the holy grail is a system whereby rich representations of individuals can be created and embedded within a rich (realistic) environment. This is where agent-based modelling and GIS come in, and is the motivation behind the writing of this book.

This chapter lays the foundation for the remainder of the book by demonstrating how the infusion of ideas (notably from complexity theory) has impacted upon how we view geographical systems, in particular by allowing us to capture the heterogeneity that makes up the world around us (Batty, 2008). These concepts can be readily embedded within the framework that agent-based modelling offers (Section 1.2). It is the intention of this book to explore not only this change in thinking, but also how, through the growth of computational power, data and tools are opening up new avenues of research in geographical systems and new ways of thinking about geographical processes and problems from the bottom up. In this chapter we begin with an exploration of how complexity theory underpins this form of geographical study (Section 1.2), why there is a need to model (Section 1.3), and how data is shedding new light on geographical systems (Section 1.4). Coinciding with the growth in data and the rise of computational power, there has also been a shift in thinking about these systems, from aggregate populations to individuals that make up such populations (Section 1.5). In summarising, we will explain the rationale for bringing together agent-based modelling and GIS (Section 1.6), before Section 1.7 provides an outline of the book.

1.2 Complexity and Geographical Systems

Complexity arises when a small number of rules or laws, applied at a local level and among many entities, are capable of generating complex global phenomena: collective behaviours, extensive spatial patterns, hierarchies, etc. These are manifested in such a way that the parts do not simply sum to the activity of the whole. This way of thinking maps naturally onto how we can view geographical systems. Interest in how complexity theory can be used within geographical systems has been increasing over the past 20 years, with Manson (2001, 2007) presenting a typology for the different types of complexity that can be found within geographical systems. Of the types that Manson (2001, 2007) presented, O'Sullivan (2004) considers aggregate complexity of most interest and relevance to geography.

Aggregate complexity is the study of how local or micro individual elements or components interact and produce complex patterns at the global or macro level, such as collective behaviours, extensive spatial patterns, self-organisation, emergence, feedback, path dependence, and hierarchies (see Table 1.1 for an explanation of these terms). Characteristics of complex systems provide a new way of understanding geographical phenomena. Rather than pursuing a reductionist (or top-down) approach, complexity examines systems by dissecting them into logically justified components (i.e. subsystems). The complex-systems approach can be characterised by being generative (or bottom-up) (Epstein and Axtell, 1996). Phenomena of interest are studied as the product of multiple interactions among simpler basic units which correspond to identifiable entities – for example, studying

Table 1.1 Key terms used in complexity theory

Term	Explanation
Self-organisation	The system's ability to self-organise – without higher-level direction – leads to emergent phenomena. Self-organisation allows entities to change their internal structure or behaviour in order to interact and adapt with the environment. For example, in economics, national and global markets evolve from locally interacting agents all pursuing what they want. This means that the system cannot be described as structurally stable, as individual agents respond differently to their environment.
Nonlinear	Outputs do not have to be proportional to their inputs, which results in the system being unstable. Where change is more random and therefore less predictable, it involves discontinuities, rapid changes as opposed to smooth ones and persistence; low, for instance, does not always follow high.
Feedback	Self-organisation results from feedback mechanisms (both positive and negative) as a result of interactions between individual entities. Negative feedback tends to dampen activity, in the sense that changes get quickly absorbed and the system gains stability. Classic examples of this are a thermostat that attempts to regulate the temperature in a room, and not going to the bar when it's crowded (see Arthur, 1994). Positive feedback, on the other hand, amplifies changes, leading to instability and sometimes with catastrophic consequences such as population collapses. Good examples of positive feedback include the idea that the more criminals there are, the greater the chance of more criminals and thus rising crime rates, and a run on a bank which can cause it and other banks to fail (Ormerod, 1998). Feedback within the system leads to path dependence. Of course systems and models can have both types of feedback; for example, in Conway's Game of Life (see Section 11.2.1), an intermediate amount of life begets life (a positive feedback), but too much or too little life leads to death (a negative feedback).
Path dependence	The concept of how history dictates how systems evolve and restructure. For example, Batty (2001) showed how positive feedback reinforced the existing rank size distribution of cities within Great Britain between 1901 and 1991; similar patterns are seen elsewhere, such as France (Pumain, 2012). Urban growth also impacts future urban growth (Xie, 1996), and residential decisions impact land markets (Parker et al., 2012)

the emergence of traffic jams through the modelling of individual vehicle movement and interaction (see Section 2.4.3). Individuals interact, learn and adapt from both their environments and from each other through feedback. How this can result in the emergence of patterns at the macro level through self-organisation based on what has occurred in the past is a central focus of complexity science. It highlights the need for dynamic models built from the bottom up, as opposed to models operating from the top down.

The use of complexity theory has numerous advantages with regard to our understanding and interpretation of geographical systems. Cities, for example, have long been recognised as problems of organised complexity, or as ‘people systems’ (Jacobs, 1961), in the sense that they present situations in which half a dozen quantities are all varying simultaneously and in subtly interconnected ways. Change is only noticeable when different patterns become discernible, so before change at the macro level can be observed,¹ change is taking place at multiple micro levels (i.e. subsystems) simultaneously, all of which are interacting separately, contributing to a complex web of interactions.

This notion of micro-level interactions in heterogeneous subsystems is often seen as one of the hallmarks of complex systems (Simon, 1996; An, 2012) and these interrelated subsystems (parts within parts) can form hierarchies. This notion of subsystems echoes those of *near-decomposability* (Simon, 1996): here a system (i.e. the city in this example) has multiple subsystem components (e.g. neighbourhoods) interacting among themselves, and while the interactions can be relatively weak, they are not negligible (Cioffi-Revilla, 2014). A notional idea of neighbourhoods as subsystems and the idea of near-decomposability are presented in Figure 1.1, where we have individual neighbourhoods at the micro level forming a hierarchy at the macro level (i.e. that of the city). Interactions, in this context, can be thought of as the flows of commuters from home to work locations. Simon (1996) argues that hierarchy is a fundamental property of how a complex system holds itself together: ‘hierarchical organization from the bottom up is essential for evolving systems and ... hierarchical structures are the way nature and society develop robust and resilient structures’ (Batty, 2013, p. 23). But these subsystems do not operate in isolation. In the short term they might appear to be independent of the rest of the system, but in the long run they are dependent on the aggregate system behaviour. This notion of near-decomposability allows us to focus on specific subsystems within geographical systems (such as housing markets, crime, residential location) and therefore allows abstraction and focus on the problem at hand. However, we must remain cognisant that in any complex system there exist interactions between multiple subsystems.

¹ See Section 2.1.2 for an example of this.

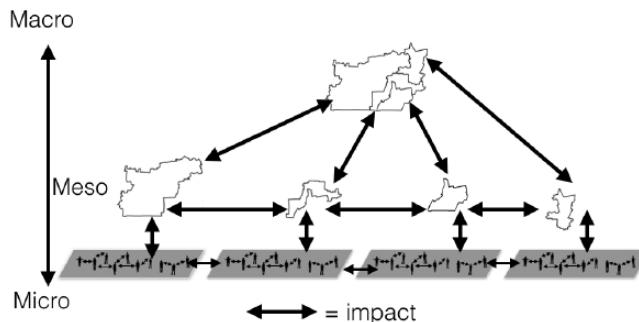


Figure 1.1 A simple hierarchical structure of a city composed of multiple neighbourhoods which form a hierarchy at the more macro level but also have interactions (e.g. commuter flows) with each other

1.3 Models

The goal of science is to make the wonderful and the complex understandable and simple – but not less wonderful. (Simon, 1996, p. x)

In seeking to understand the complexity of the world around us, we require methods that can capture and regenerate the characteristics of complex systems. Our understanding must extend across the microscopic-, mesoscopic- and macroscopic-level drivers of these phenomena, yielding comprehensive encapsulations of these systems. To achieve this, the best tool we have available is modelling. A model can be considered as a theoretical description of the way a system or process of interest works, and is often expressed in mathematical terms (or in code). Simply stated, a model is a simplified representation of a complex reality, to the point that the reality is understandable and analytically manageable (Wilson, 1974), by representing (abstracting) one or more processes that are believed to occur in the ‘real world’ processes that exist, have existed or might exist (Longley and Batty, 2003; Miller and Page, 2007). A model can be constructed as a computer program that uses (to some degree) a simplified digital representation of one or more aspects of the real world, transforming them to create a new representation. Models differ from theories in that a theory provides a series of connected statements used in the process of explanation, while models provide an idealised and structured representation of the theory – or, as often in the case of agent-based models, they are being used to develop theory (Axelrod, 1997a).

There are many reasons and purposes for modelling: from description and explanation to prediction (for a more detailed discussion of such reasons and purposes, see Lowry, 1965; Epstein, 2008). The type of model depends on its general purpose of application. The objective of descriptive models is the explanation of phenomena by reducing the complexity of the real world to a coherent and

rigorous framework to build, confirm or falsify some theory. While descriptive models might be considered abstract or ‘toy models’, such as the Schelling (1971) segregation model (see Section 2.4.1), they are often the first step in building more predictive models (see Parker et al., 2001) as they provide a way to discover new relationships. Take, for example, von Thünen’s descriptive model of 1826 (von Thünen, 1966) operationalised in digital form by Batty et al. (2004a). This simple descriptive model explains land-use patterns around a village and how such patterns can change if transportation costs are changed, and serves as a stepping stone to more predictive models of land markets and land-use change. The purpose of predictive models is to give a forecast of the phenomena under study – for example, pedestrian movement models (see Johansson and Kretz, 2012; Crooks et al., 2015a) or land-use change models (see Landis, 2001; Clarke et al., 2006).

Once the purpose and type of model has been decided (i.e. descriptive or predictive), the model needs to be created or built. A model is developed at various levels of abstraction (levels of generalisation or simplification), starting from real world concepts (high level) towards an implementation in a computer system (low level) through the process of data modelling. Data modelling is the process of structuring data about the real or artificial world. At a high level of abstraction closer to the real world object or system being modelled, it is the process of organising the real world into a well-defined and formalised set of concepts. At a lower level closer to the internal details of a computer system, it is the process of mapping these concepts onto data structures and ultimately to computer memory or disk. This progression from high-level concepts of the real world to low-level digital representation in a computer necessarily involves abstraction and formalisation. The result of the data-modelling process is a data model which can be populated by data, underpinned by a framework which maps the concepts between levels of abstraction (for more discussion on abstraction and models, see Longley et al., 2005; Gilbert and Troitzsch, 2005).

An important distinction to make is between modelling and simulation. While used interchangeably throughout the literature, there is a subtle difference. Batty (1976) writes that ‘all mathematical models which involve the use of large-scale computational facilities are referred to as simulation models’. The distinction between analytic and simulation methods with regard to modelling is that ‘analytic methods of modelling involve the use of mathematical analysis to arrive at explicit equations representing the behaviour of the system. Simulation methods are used to derive the behaviour of the system when the system is too complex to be modelled using the more direct analytic approach’ (Batty, 1976). An example of a simulation model is White and Engelen’s (1993) cellular automata model of urban land-use development over time. Wilson (2000) writes that ‘simulation is a critical concept in the future development of modelling because it provides a way of handling complexity that cannot be handled analytically’.

1.4 Data

While models provide one route towards understanding geographical complexity, we are also rapidly gaining a richer understanding of the world through the vast amount of new data becoming available and enabling new veins of research. Today we are experiencing a ‘data deluge’, and by the year 2020, many predict the global universe of accessible data to be of the order of 44 zettabytes or 44 trillion gigabytes, with no signs of slowing down (EMC2, 2014). However, within the last decade, there has been a shift in the source of this data. In the past, the main generators of digital data were government and commercial organisations (e.g. US Census Bureau, as discussed in Section 5.5) but it is estimated that approximately 75 per cent of all digital data is now contributed by individual users (Mearian, 2011), a trend that is expected to increase (Hollis, 2011) as computing and technological advances are solidifying the role of the general public as both the main contributor and consumer of data.

New data sources will play a significant role in shaping the future of both agent-based modelling and GIS. As technology is becoming more pervasive, this is resulting in more data becoming available. When combined with the growth of the internet of things, open data initiatives (e.g. www.data.gov/), and new international standards for indicators for city services and quality of life, researchers will have a more detailed picture of how diverse groups of people perceive and use space around the world (Crooks et al., 2017). The key question is, as Harris et al. (2017) state, ‘how can we use new forms of data to understand how real people shape and are shaped by geographical processes?’ Clearly, data alone is not enough to understand such systems, while some note how data is now providing us with ‘the capacity to collect and analyse data at a scale that may reveal patterns of individual and group behaviours’ (Lazer et al., 2009). This area is becoming known as computational social science (CSS), but others (e.g. Torrens, 2010b; Cioffi-Revilla, 2014; Parrett et al., 2018) note that data and data analysis techniques do not replace the scientific method of hypothesis, model and test. Data (and data science techniques) can help with answering research questions but often fails to reveal the causal effects (Mazur and Manley, 2016). Data does not give us the ‘why’, only the ‘what’, in the sense that it gives us the *patterns* but not the *processes* that cause them to emerge. For this, we need to model social interaction – this is the broader view of CSS (Torrens, 2010b; Cioffi-Revilla, 2014).

1.5 Individuals

There has been a marked change over the past sixty years in how we simulate the individual within geographical systems. Previous generations of researchers did not have either the computational power or access to data that today’s researchers do.

The lack of data and tools for simulating individuals meant that models examined the systems as a whole, based on aggregate analysis (e.g. Alonso, 1964; Wilson, 1974), offering a limited way to examine the dynamics taking place within such systems (Batty, 1995). Digital computation began to emerge in the 1950s, and this led to the creation of ‘artificial laboratories’ where a variety of different scenarios and policy interventions could be tested. The ability to experiment with alternative ideas and theories led to a new perspective on how geographical systems should be simulated by drawing on ideas and concepts from other disciplines such as physics and biology. While this was impacting on the way that researchers were both thinking about and modelling geographical systems, the lack of data and computing power meant that most of this work was focused at the aggregate level (Torrens, 2000). For reasons of simplicity and parsimony, it was believed that homogeneity in populations was the best way to account for behaviour in space and time (Batty, 2012). An excellent overview of the main strands of geographical modelling over the past fifty years can be found in Batty (2012).

With the rapid increase in both computing power and data availability in the past twenty years, the individual has begun to emerge within geographical simulations. Whilst the individual has been represented within the approaches of cellular automata and microsimulation (these are discussed in Chapter 11), it was not until the appearance of agent-based modelling that a rich representation of individual characteristics and behaviour could be created. The rise of agent-based modelling coincided with the appearance of geocomputation, the application of computers and computational methods to solve geographical problems. Harris et al. (2017) remark that the establishment of geocomputation has ‘been cemented by increases in computer processing power, data storage, developments in computer programming languages and easily accessible frameworks that enable rapid development of models with minimal programming experience’.

The increase in computation, individual-level data sources and the infusion of ideas from complexity have created a move away from our traditional understanding of geographical systems. This has resulted in a greater awareness of the role that self-organisation and emergence (outlined earlier) play within these systems; for example, it is the local-scale interactive behaviour (commuting, moving) of many individual objects (vehicles, people) from which structured and ordered patterns emerge in the aggregate, such as peak-hour traffic congestion (Nagel et al., 1997) and the large-scale spatial clustering of socio-economic groups by residence (Schelling, 1978). In urban economies, large-scale economies of agglomeration and dispersion have long been understood to operate from local-scale interactive dynamics (Krugman, 1996). Understanding how geographical systems grow and evolve from the bottom up allows us to understand how uncoordinated local decision-making gives rise to coordinated global patterns which define the shape of the world around us, and how these global patterns then influence the local decisions of individuals.

1.6 Agent-Based Modelling and Geographical Information Systems

While data can take us some of the way towards understanding the world around us, only models allow us to build a *comprehensive* understanding of the mechanisms and behaviours that shape complex geographical phenomena. In seeking to develop this view, and being increasingly uninhibited by computational capabilities, agent-based modelling represents the most promising methodology for capturing these systems. Inherent in the agent-based modelling approach is an ability to capture – in depth – the heterogeneity in behaviour, effects of interaction and spatial dependence so important in understanding complex geographical phenomena. The approach allows us to combine quantitative measures and qualitative theory, to combine observations and hypotheses within a single, integrative framework. While other approaches can yield compelling and realistic representations of the world, agent-based modelling represents the most comprehensive framework for modelling the world.

Yet geographical modelling is not only about understanding; in a world of increasing uncertainty and interdependence, we require reliable and sophisticated ways in which to predict its future evolution. Agent-based modelling provides us with the strongest capability to capture and predict the future change. A predictive agent-based model is able to capture many of the factors influencing the actions and interactions of individuals, and how changes in conditions influence how aggregate patterns of behaviour emerge. This task remains a key challenge and, as we will discuss, requires careful integration of evaluation methods and uncertainty assessment, but one which agent-based modelling is well placed to deliver.

Geographical information systems represent the foundation of much of what we do in measuring and modelling geographical systems. It would be easy to simply assume that GIS falls into a wider ecosystem of technologies encompassed by ‘data science’, but this would be to ignore the importance of geography, and the analytical methods developed within GIS that allow us to expose and understand spatial phenomena. GIS forms another integrative technology, linking interesting behaviours, activity and events with a discrete location or set of locations. Using GIS, these locations allow us, through measurements of spatial association and dependence (covered in Chapter 9), to build a wider understanding of the geographical process of interest or the behaviours being observed.

More broadly, GIS provides us with a set of data structures and protocols for integrating real world observations with a set of computational analysis and modelling methods. For instance, the crime that takes place at specific locations can be recorded within a GIS, along with a set of informative metadata, and then be easily integrated into analysis processes (e.g. for predicting future crime locations, or informing the deployment of police officers) and modelling (e.g. through an agent-based model of crime occurrence and prediction of future change).

GIS provides the enabling structure to allow this broader analysis, and supports a swathe of analytical activity.

As we have highlighted, understanding, analysing, and modelling geographical systems is a fundamentally important task, and as new forms of data allow us to measure these systems in greater detail than ever before, we require methods that can respond to the opportunities for improvement. In agent-based modelling and GIS we have two complementary technologies that enable scalable and granular analysis and modelling of geographical systems. Agent-based modelling is underpinned by GIS, contributing predictive capability at the individual level, allowing a full expression of heterogeneity, environmental influence and agent interaction. GIS not only provides the spatial representation upon which agent-based models can be built, but also through analysis of observation data can inform and shape the modelling, creating new rules and insights for predicting future states. In analysing and modelling geographical processes, agent-based modelling and GIS form a comprehensive set of methods.

1.7 Outline of the Book

The overarching aim of the book is to facilitate the in-depth learning of both the fundamental knowledge and practical skills required for building and running agent-based models for simulating geographical systems. In order to do so, fundamental theory and practical examples are fused together to deepen learning and understanding through application. While there are many excellent books which sketch out how to build agent-based models (e.g. Railsback and Grimm, 2011; O’Sullivan and Perry, 2013; Wilensky and Rand, 2015) and others that provide an overview of agent-based models applied to geographical systems (e.g. Gimblett, 2002; Benenson and Torrens, 2004; Heppenstall et al., 2012b), and books on GIS and geocomputation more generally (e.g. de Smith et al., 2009; Longley et al., 2010; O’Sullivan and Unwin, 2010; Abrahart and See, 2014; Brunsdon and Singleton, 2015a), there is a notable absence of books that provide step-by-step processes on how to bring agent-based models and GIS together.

This book presents the rudimentary basics, guiding readers from simple model examples through to demonstrating more complex models that can handle both human behaviour and data in a spatial context. All examples (developed using the NetLogo programming environment (Wilensky, 1999)) in the book are supplemented by code, with further examples placed in a code repository on the accompanying website: www.abmgis.org. Companion teaching materials developed as PowerPoint presentations are available for practitioners to download and customise.

Chapter 2 introduces the fundamental theory and applications of agent-based modelling. Key concepts and ideas are presented that are developed throughout the book. Chapter 3 moves on to describing an overview of how to design and develop agent-based models, before Chapter 4 provides a tutorial on the basics

of building agent-based models with NetLogo (the agent-based modelling environment used within this book). Attention is then turned to GIS, the other core component of the book, with the core elements presented in Chapter 5. Chapter 6 discusses the rationale for merging agent-based modelling and GIS and provides examples of where this has been performed. One of the challenges of using agent-based modelling is modelling human behaviour, and how we may do this is the subject of Chapter 7. Attention is then turned to how networks (both social and physical) can be integrated into agent-based models (Chapter 8), highlighting how this allows us to study links and connections between people and places. Chapter 9 presents approaches for measuring spatial distributions and model fit, an issue that is developed in terms of verification, calibration and validation in Chapter 10. In the final core chapter, Chapter 11, we provide a more holistic view of the field by contrasting agent-based modelling with other modelling approaches used to understand geographical systems. In closing, in Chapter 12, the challenges and opportunities that lie ahead with respect to integrating GIS and agent-based modelling are discussed.

Where possible, all examples within the book have been developed in NetLogo and made available via the website. Table 1.2 presents a list of the models used within the book. Furthermore, Appendix A provides a gallery of applications of agent-based models that explore various aspects of real world systems, from pedestrian modelling to natural disasters. These are provided to showcase more sophisticated models as well as those used to demonstrate pedagogic points throughout this book.

Table 1.2 NetLogo models introduced throughout the book, often with accompanying code examples

Section	Model	Description
2.4.1	Segregation (Schelling, 1971)	Schelling's well-known segregation model is introduced and discussed
2.4.2	SugarScape (Epstein and Axtell, 1996)	A seminal example of how to 'grow' a phenomenon from the bottom up
3.6	Segregation (Schelling, 1971)	An example of how to plan and build an agent-based model, using the Segregation model as implemented in NetLogo (Wilensky, 1997a)
4.3	First NetLogo Model	A simple model introduced as part of the tutorial on using NetLogo to build agent-based models
4.4	Advanced NetLogo Model	A more advanced adaptation of the introductory model
5.3.1	Loading ImportRasterSample	A demonstration of how to load raster data using NetLogo
5.7	Uganda Population	A simple population model of Uganda, demonstrating how to manipulate GIS data in preparation for modelling

(Continued)

Table 1.2 (Continued)

Section	Model	Description
6.4.1	Urban Growth	An example of the SLEUTH model for urban growth
6.4.1	Pedestrian Modelling	An example of modelling pedestrian movements, using a cost surface to drive their trajectories
6.4.1	Rainfall (raster)	A worked example of the use of raster data in an agent-based model, focusing on rainfall
6.4.2	Revised Segregation (I)	An update to the classic Schelling model, demonstrating how to read and work with GIS vector data in NetLogo
6.4.2	Revised Segregation (II)	A further update to the Schelling model, incorporating numerous agents per polygon
7.6	Consumer Behaviour	An example of modelling the behaviour of consumers, using a simple probabilistic model of behaviour (Sturley et al., 2018)
7.7	Riot	An example of simulating crowds and riots using the PECS behavioural framework (Pires and Crooks, 2017)
8.3.1	Networks Example	Examples of how to create different types of network in NetLogo
8.4	Modelling Road Networks	A worked example demonstrating how to create a road network in NetLogo from GIS vector data, and how to restrict the movements of agents to roads
8.5	Modelling Social Networks	An example that includes both physical and social networks
10.3.3	WalkThisWay	An example of model calibration using real individual trajectory data
11.4	Susceptible–infected–recovered (SIR)	The SIR model, used to compare alternative modelling approaches

Chapter Summary

This chapter has presented the broad context that the rest of the book will build upon. Three key themes were discussed: what a model is, complexity and the role of the individual within geographical modelling. The chapter ended with a discussion of the benefits of bringing together agent-based modelling and GIS. Subsequent chapters will build upon these ideas and concepts.

1.8 Annotated Bibliography

For an overview of how urban modelling has changed over time, readers are referred to:

- Torrens, P.M. (2000) How land-use-transportation models work. Working Paper 20, Centre for Advanced Spatial Analysis, University College London.

- Batty, M. (2008) Fifty years of urban modelling: Macro-statics to micro-dynamics. In S. Albeverio, D. Andrey, P. Giordano and A. Vancheri (eds), *The Dynamics of Complex Urban Systems: An Interdisciplinary Approach*, pp. 1–20. Heidelberg: Physica-Verlag.

For a detailed philosophy of complexity science and computing, readers are referred to:

- Simon, H.A. (1996) *The Sciences of the Artificial* (3rd edn). Cambridge, MA: MIT Press.

For a gentle introduction to how complex systems, ranging from political parties to stock markets and ant colonies, can be explored through models, readers are referred to:

- Miller, J.H. and Page, S.E. (2007) *Complex Adaptive Systems*. Princeton, NJ: Princeton University Press.

For a short overview of agent-based models and spatial sciences (including both human and physical geography), readers are referred to:

- Torrens, P.M. (2010) Agent-based modelling and the spatial sciences. *Geography Compass*, 4(5), 428–448.

For readers wishing to know more about computational social science and the role of ‘big data’ see:

- Lazer, D., Pentland, A., Adamic, L., Aral, S., Barabási, A., Brewer, D., Christakis, N.A., Contractor, N., Fowler, J., Gutmann, M., Jebara, T., King, G., Macy, M., Roy, D. and Van Alstyne, M. (2009) Computational social science. *Science*, 323(5915), 721–723.
- Cioffi-Revilla, C. (2014) *Introduction to Computational Social Science: Principles and Applications*. New York: Springer.
- Torrens, P.M. (2010b) Geography and computational social science. *GeoJournal*, 75(2), 133–148.

2

INTRODUCTION TO AGENT-BASED MODELLING

Chapter Outline

In this chapter we introduce the key concepts behind agent-based modelling. What is an agent, and what are rules? These are discussed along with a consideration of the main advantages and disadvantages for simulating spatial systems. A range of established applications are presented to give a flavour of how agent-based models can be successfully applied. The overarching aim of this chapter is to give the reader an understanding of what an agent-based model is. This knowledge will be built upon in subsequent chapters.

2.1 Introduction

Agent-based models are beginning to appear in many different aspects of our lives. This has been particularly noticeable in digital media; for example, Massive (2017) has used agent-based modelling in a number of films such as *World War Z*, *Spectre* and *Dawn of the Planet of the Apes* to re-create large-scale crowd or fight scenes. Understanding the attraction of agent-based modelling for creating these scenes gives an insight into the appeal that this approach has for researchers in the social and geographical sciences. Agent-based modelling allows individuals with their own set of unique characteristics and rules of behaviour to be created. Furthermore, these individuals can be placed within a realistic environment and their interactions (both with other individuals and their environment) can be charted. This ability to capture and potentially understand individual behaviour over multiple spatial scales and time-scales offers new opportunities to understand the processes and drivers that shape social and geographical systems. Before agent-based models appeared, researchers were confined to using mathematical models – for example, spatial interaction (Fotheringham and O’Kelly, 1989) and system

dynamics (Forrester, 1969) models, to simulate the behaviour of populations. The main drawback of these approaches was their inability to simulate individuals; instead diverse populations had to be simulate as one homogeneous group with the same behaviour and characteristics (Wilson, 1974). Whilst these approaches were successful at reproducing macro-level patterns, they could not offer any insights into why these patterns appeared.

In the last twenty years, agent-based modelling has established itself as a discipline in its own right and is becoming a standard tool in the social science researcher's toolkit. Richer representations of individuals that incorporate not only characteristics such as age and sex but also beliefs, opinions and behaviour are beginning to appear, driven by new sources of micro-data ('big data') that are continually appearing. Combined with detailed sources of environmental data – for example, in cities we have numerous 'smart' cameras and sensors continually collating data – researchers are able to create 'artificial laboratories' within which a range of interventions and scenarios can be played out. These could include examining the impacts of a change in a transport infrastructure, or developing emergency protocols to be enacted in the case of a disease outbreak or incident within a population centre. The number and diversity of agent-based modelling applications will only continue to increase in the future.

In this chapter, the fundamentals of agent-based models are introduced; this knowledge creates the foundation for the concepts that are introduced in subsequent chapters. The remaining subsections of Section 2.1 outline the rise of agent-based modelling, focusing on the different elements that make up an agent. Understanding the advantages (Section 2.2) and limitations (Section 2.3) of agent-based models is key before making a decision about their suitability for a particular application. Section 2.4 provides a broad overview of notable applications that agent-based modelling has been used to successfully explore. In subsequent chapters, further discussion is given to the considerations that the researcher needs to be aware of when working with agent-based models (Chapter 3), how we can link such models to geographical information systems (Chapter 6) and how behaviour and interactions can be captured in these models (Chapters 7 and 8, respectively).

2.1.1 What Is an Agent?

As noted above, agent-based modelling has experienced a rapid growth in the last twenty years. While applications can now be found across a range of different disciplines, its initial development was confined to one or two fields such as geography and ecology. These fields developed different aspects of agent-based modelling to fulfil specific purposes of their applications; within geography, methods for embedding space into agent-based models have been developed (see Chapter 6), whilst ecologists developed methods to simulate large numbers of

agents. This fractured and rapid development of agent-based modelling has given rise to the absence of any central repository, protocols or body of knowledge outlining what an agent or agent-based model is. While each discipline may have its own definition, there is at least some consensus about the different elements that an agent should be comprised of. The following list of key agent attributes is drawn from Wooldridge and Jennings (1995), extended and explained further by Franklin and Graesser (1996), Epstein (1999), Macal and North (2005) and Crooks and Heppenstall (2012).

- *Autonomy.* Agents are autonomous, meaning that they act outside of the direct influence of an external, centralised control. Agents undertake actions according to an independent decision-making process. These behaviours may be influenced by information, which they may share with other agents or their environment. They may interact with agents in a variety of ways, limited by the environment, context and model design, which does not (necessarily) impact on their autonomy.
- *Heterogeneity.* Agent-based models incorporate heterogeneity in the expression of agent characteristics. A human agent, for example, might be attributed an age, gender, or job, representative of an individual drawn from the population being modelled. Groups, or types, of agents may be incorporated, and an agent-based model may include a number of different agent groups, each representing different entities. For example, a housing agent-based model might involve buyer agents, seller agents, broker agents, and even house agents, each with independent characteristics and behaviours.
- *Active.* Agents undertake independent actions during the course of a simulation. The following agent characteristics are commonly modelled:
 - *Goal-directed.* Agents are given goals that they are expected to attempt to achieve (or maximise) during the simulation.
 - *Reactive or Perceptive.* Agents may be designed to have an awareness of their environment and the presence of other agents. Agents may or may not be provided with a prior knowledge of their environment. Where agents are granted environmental knowledge this may be perfect or bounded, the latter representing an individual ‘mental map’ of the environment. Within this environment, agents interact, make use of and avoid environmental features and obstacles in undertaking actions.
 - *Rationality.* Within social sciences, the dominant form of decision modelling is based upon the rational choice paradigm (Axelrod, 2007). Rational choice models generally assume that agents are rational

optimisers with perfect access to information, foresight and infinite analytical ability (Parker et al., 2003). However, agents may be *bounded* in their rationality (potentially with agent-level heterogeneity), as first introduced by Simon (1955). Bounded rational agents make inductive, personal and adaptive choices.

- *Interactive or Communicative.* Agents have the ability to interact with other agents and their environment in a variety of ways. This could involve interactions based on spatial proximity or via a network-based connectivity (e.g. a social or trade network).
- *Mobility.* Agents can move around their modelled environment, allowing them to pick up information for decision-making, interact with agents and perform actions. The space allowed for interaction may be restricted either spatially (e.g. constrained to a road network) or topologically (e.g. social network).
- *Learning and Adaptation.* Agents can be granted the ability to learn and adapt. They can be designed to alter their current state depending on previous states or memories, permitting them to adjust behaviour with reference to a form of memory or learning.

Whilst extensive, this list is by no means exhaustive. With the increasing number of agent-based models on offer, in particular the increase in behavioural agent-based models, the list will only continue to grow. It should also be noted that, depending on the application, agents will possess a mixture of these characteristics. This point highlights one of the most attractive features of agent-based modelling: its flexibility.

An agent can represent any entity that has a degree of autonomy. Examples include individuals, buildings, cars, land parcels, water droplets and insects. Figure 2.1 shows the similarities between the concepts of a ‘social’ human and a supermarket retailer and between the representations of such agents within an object-orientated program. For further information about frameworks and the construction of agent-based models, see Grimm and Railsback (2012) and Chapter 3.

Figure 2.2 diagrammatically represents some of the core features of an agent presented above. Here, a group of heterogeneous agents are situated in their own space within an *artificial world*. These spaces could be geographical areas exerting environmental influences on the agents (as will be shown in Chapter 6) – for example, the availability of a particular service. Through interactions the agents can exchange information or ideas which can lead to the *emergence* of new knowledge or ideas. This newly emerged knowledge may result in the agent reacting and pursuing a new form of decision-making to reach its goal. A simple worked example along these lines will be presented in Section 2.1.2.

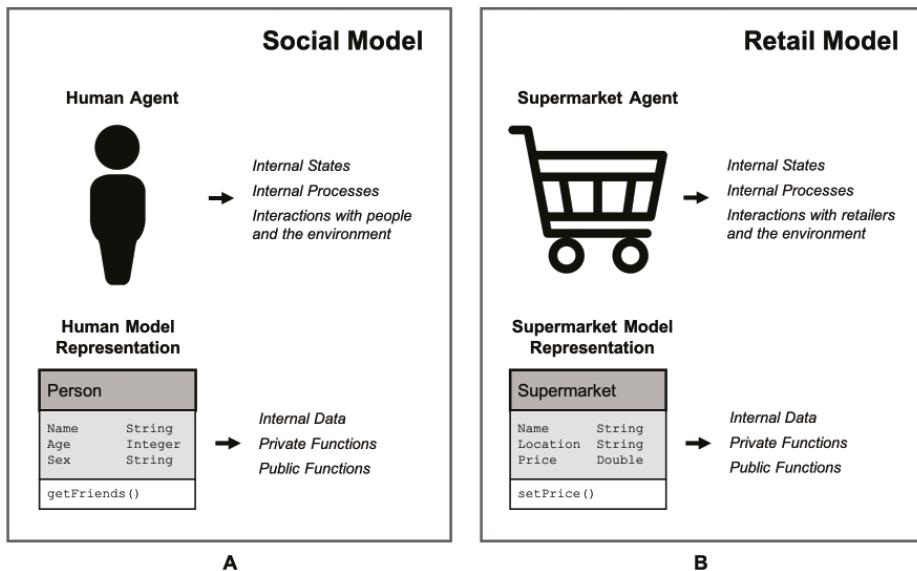


Figure 2.1 (A) A human agent and (B) a supermarket retailer agent and their representation within an object-orientated environment

2.1.2 Agent Rules

Rules are a set of commands that are assigned to each agent to guide their behaviour and decision-making (see Section 3.5 and Chapter 7 for more information on creating rules and behaviour). These rules also influence an agent's relationships with other agents and/or their surrounding environment. Assigning 'accurate' rules to an agent remains a significant challenge, requiring a balance of information from published literature, expert knowledge, data analysis and numerical work (some practical guidelines are given in Chapter 3). Rule sets can be uniformly applied to all agents or can be unique to an agent. Heppenstall et al. (2006) created petrol station agents that each operated the same basic rule set based on a desire to maximise profits. This initial work was further refined with 'realistic' rule sets created to reflect different types of petrol retailer such as supermarkets, international, national and independent stations (Heppenstall et al., 2006).

Rules are typically based around *if–then–else* statements as shown in Figure 2.3, with agents carrying out an action once a specified condition has been satisfied. Agents can be embedded with a notion of learning, often referred to as *intelligence*, through evolutionary computation (see Heppenstall et al., 2007, for details of the use of a genetic algorithm for optimising behaviour). The increase in new forms of individual-level data ('big data') that reveals information about preferences and opinions has led to an increase in agent-based models that are capable of representing

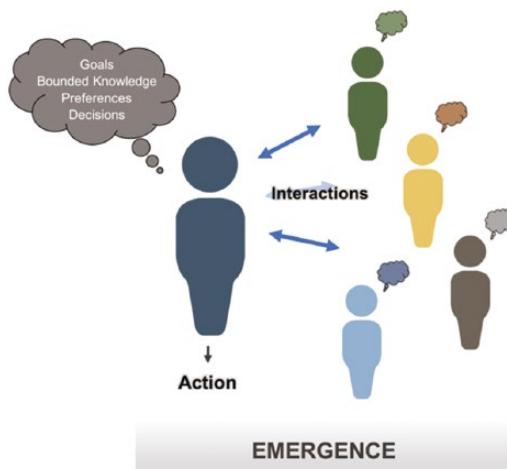


Figure 2.2 Schematic illustrating how emergence occurs through individual autonomy and interaction between agents

human behaviour more realistically. For example, Malleson et al. (2010) used the PECS (physical conditions, emotional states, cognitive capabilities and social status) framework to represent the motivations and desires of criminals. Chapter 7 provides a more detailed treatment of behaviour along with how different frameworks can be used for handling human behaviour in agent-based models.

2.1.3 An Agent's World

A wealth of rich spatial data is becoming increasingly available through mapping agencies (such as the Ordnance Survey in the UK and the National Geospatial-Intelligence Agency in the USA) and open source sites such as OpenStreetMap (which will be discussed in Section 5.6). The availability of detailed data sets about the world around us offers opportunities to create highly ‘realistic’ environments that agents can inhabit and interact with. Through linkage to geographical information systems (see Chapter 6), these ‘artificial worlds’ can be created using a variety of geographical data types (see Sections 5.3 and 6.4). This allows space to be represented in different ways. Taking proximity (a key influence in many agent-to-agent interactions) as an example, this may be defined by spatial distance for continuous space, adjacency for grid cells (as in the example in Section 2.4.1), or by connectivity in social networks (as shown in Figure 2.3).

This richness and abundance in both environmental- and individual-level data has given rise to the idea of an agent-based model as a miniature laboratory where new ideas and theories about the system under consideration can be tested out. The Schelling (1971) model (introduced in Section 2.4.1) is a

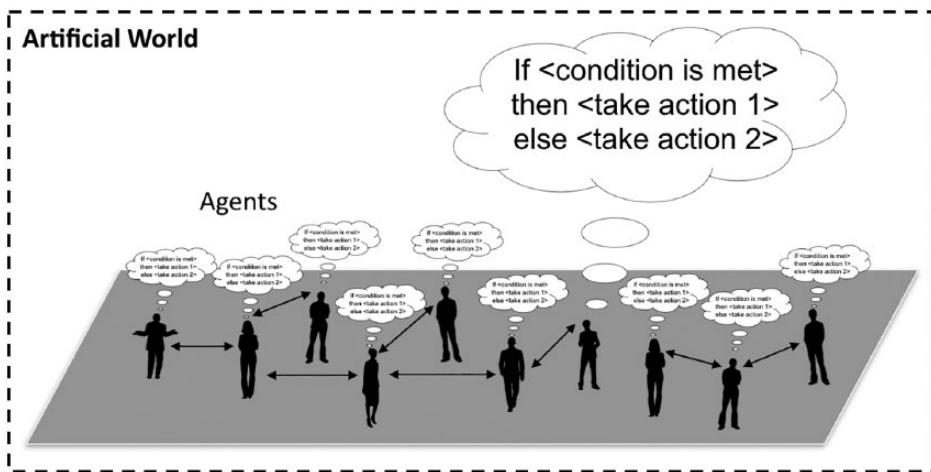


Figure 2.3 Conceptualisation of an agent-based model where people are connected to each other and take actions when a specific condition is met

good example of this. This environment allows a range of scenarios to be examined that would be impossible to test out in the real world – for example, how individuals would react to a fire in a building or an explosion in a city centre. Re-creating these scenarios within a model allows the modeller to identify potential problems such as bottlenecks, and allows for the testing of numerous scenarios such as the way various room configurations can impact on evacuation time (e.g. Castle, 2007a; Wagoum et al., 2012; Torrens, 2012). By building such models we can focus on mitigation and preparedness rather than response and recovery in emergency incident management.

2.2 Advantages of Agent-Based Modelling

Increases in both computational power and data have changed the way researchers think about geographical systems. The previous generation of researchers used abstract geographical theories – for example, Christaller (1933) – and aggregate-level data sets, to inform their understanding about the world around us (see Chapter 1). The influx of individual-level data has opened up new insights into the processes and drivers that shape our systems. Researchers now view geographical systems, such as cities, as being driven by potentially infinitely many individual decisions (for a discussion, see Batty, 2013; O'Sullivan et al., 2012). This recognition of the importance of the individual (and their decisions) makes agent-based modelling an obvious framework within which simulations can be created and ideas tested out (Gilbert and Terna, 2000). A particular advantage of agent-based

models is their ability to represent and test social theory which cannot easily be described using mathematical formulae (Axelrod, 1997b).

Crooks and Heppenstall (2012) highlighted several advantages of agent-based modelling over other techniques (see also Chapter 11). These included the ability to capture emergent phenomena through bottom-up modelling (see Sections 1.2 and 1.6), provide a natural environment for the study of social and geographical systems (agents can move around spatial simulations), the ready inclusion of a stochastic element (i.e. models can contain an element of randomness), agents that are adaptive to changes in circumstances, and the ability to have unlimited numbers of parameters and rules. In addition, Bonabeau (2002) presented a number of situations where agent-based modelling can be advantageous:

1. Interaction between agents is complicated, nonlinear, discontinuous or discrete. This can be particularly useful if describing the discontinuity of individual behaviour – for example, using differential equations.
2. The ability to design a heterogeneous population of agents with an agent-based model is significant. Agents can represent any type of unit. Heterogeneity also allows for the specification of agents with varying degrees of rationality. This offers advantages over approaches that assume perfectly rational individuals, if they consider individuals at all.
3. The topology of agent interactions is heterogeneous and complex. Aggregate flow equations usually assume global homogeneous mixing, but the topology of an interaction network can lead to significant deviations from predicted aggregate behaviour.
4. Agents exhibit complex behaviour, including learning and adaptation.

Another advantage of agent-based models is that they are dynamic; they simulate the emergence of a phenomenon over time. This is a marked contrast to the static nature of earlier styles of urban and regional modelling (see Batty, 1976). While time is represented in discrete steps, time steps can easily equate to seconds, days or years. Multiple time-scales can be represented within agent-based models through the use of automaton clocks (Torrens, 2003). These have great utility for realistically simulating urban development or a particular geographical phenomenon (O'Sullivan, 2001). Related to this is the issue of spatial scale. Agent-based models can be defined within any application area – for example, a building, a city or a road network. This means that agent-based models are scaleless with the phenomena of interest driving the scale to be used – for example, from the micro-movement of pedestrians within a building during an evacuation (e.g. Helbing et al., 2000), to the movement of cars on a street network (e.g. Nagel, 2003) and the study of urban growth (e.g. Brown et al., 2005). It is also possible to combine

these objects to represent phenomena at different scales within the same model. Finally, agent mobility makes agent-based models very flexible in terms of potential variables and parameters that can be specified.

2.3 Limitations of Agent-Based Modelling

Agent-based modelling is continually gaining in popularity, and Section 2.4 presents an array of applications to which this approach is currently being applied. This growth in popularity shows no sign of slowing down, a factor that will be cemented by the continual increase in both computing power and rich sources of individual-level data. However, as with any modelling approach, researchers need to be aware of the main constraints associated with this method.

As alluded to in Section 2.1, the growth of agent-based modelling has been fractured: different disciplines have advanced parts of agent-based modelling in isolation to developments occurring elsewhere. This lack of a central repository means that there is no template or universally accepted way to design and build agent-based models. Much of the published literature contains models that are badly designed and not reproducible (i.e. important details about the model are absent). There is some effort within the community to create accepted protocols, notably the Overview, Design concepts and Details (ODD) protocol of Grimm et al. (2006) (introduced in Section 3.7), but this remains a major drawback of the discipline. Without central guidance, it is hard to know how much detail to input into models or what the acceptable level of abstraction is (Coulcelis, 2002; Crooks and Heppenstall, 2012). This issue is particularly felt when considering how to embed human behaviour within agent-based models. Whilst there are an abundance of behavioural frameworks available (see Kennedy, 2012, and Chapter 7 for an overview of how behavioural frameworks can be implemented in agent-based models), which one to use remains a challenge.

Another criticism levelled at agent-based models is that they are data hungry. Agent-based models can be applied to systems that contain potentially millions of agents, each with their own unique characteristics and individual rule sets. Clearly setting up a model of this scale requires a vast amount of data not only for creating the initial variable and rules, but also for the purpose of calibration and validation. Evaluating these models is perhaps the biggest challenge facing agent-based modellers. Rigorous evaluation of models is important if the models are to bridge the gap between academia and the policy arena. However, this is often neglected in studies involving agent-based modelling.

With the advent of ‘big data’, data availability is becoming less of a problem. However, there are issues of data quality and linkage to be considered when using these data sets. Furthermore, there is the issue of the lack of appropriate

tools to extract significant patterns and processes from these data types. Without understanding what drives the processes within our systems, we are greatly limited in our ability to reproduce them.

2.4 A Gallery of Applications

Agent-based modelling is being used in a diverse set of areas ranging from archaeology to zoology (e.g. Ahearn et al., 2001; Axtell et al., 2002). It is rather impracticable to cover all the applications of agent-based models here as this is an ever increasing list! Figure 2.4, for example, shows the number of search results retrieved for the term ‘agent-based modelling’ from the Web of Science and Google Scholar. Agent-based models have been used to explore theories of political identity and stability (Cioffi-Revilla and Rouleau, 2010), the processes that lead to state formation (Cederman, 2001), biological models of infectious diseases (Eidelson and Lustick, 2004), models of crime (Malleson et al., 2013; Birks et al., 2014), growth of bacteria colonies (Krzysztof et al., 2005), stock trading (Carrella, 2014), labour market dynamics (Guerrero and Axtell, 2011), tax compliance (Bloomquist, 2011), housing markets (Geanakoplos et al., 2012) and voting behaviours in elections (Laver and Sergenti, 2012), to name but a few. From a geographical perspective, agent-based models have been applied to an equally wide range of phenomena, ranging from the micro-movement of pedestrians over seconds and hours (e.g. Torrens, 2014a) to the rise of city systems over centuries (e.g. Pumain, 2012) and almost everything in-between.

In the remainder of this section, established agent-based modelling examples, the Schelling (1971) Segregation model (Section 2.4.1) and SugarScape (Section 2.4.2), are presented. Following these examples, an application area that has received considerable attention from agent-based modellers, traffic simulation (Section 2.4.3), is discussed. Finally, applications of agent-based models which have impacted decision-making are presented in Section 2.4.4. These examples demonstrate not only the main characteristic of an agent (as discussed in Section 2.1), but also how higher-order patterns emerge from the interaction of individual agents. Furthermore, these models also demonstrate that even though the rules are well specified, the results are not. We do this to draw attention to two assertions discussed by Simon (1996): that ‘A simulation is no better than the assumptions built into it’ and ‘A computer can do only what it is programmed to do’. While these statements are true, behind them lies the reasoning that even if assumptions are correct it is difficult to discover what they imply without running the model. This relates to the fact that in agent-based models, feedback between agents and their environment and the stochasticity that is inherent in such models make it difficult to know precisely what the outcomes will be before we run them.

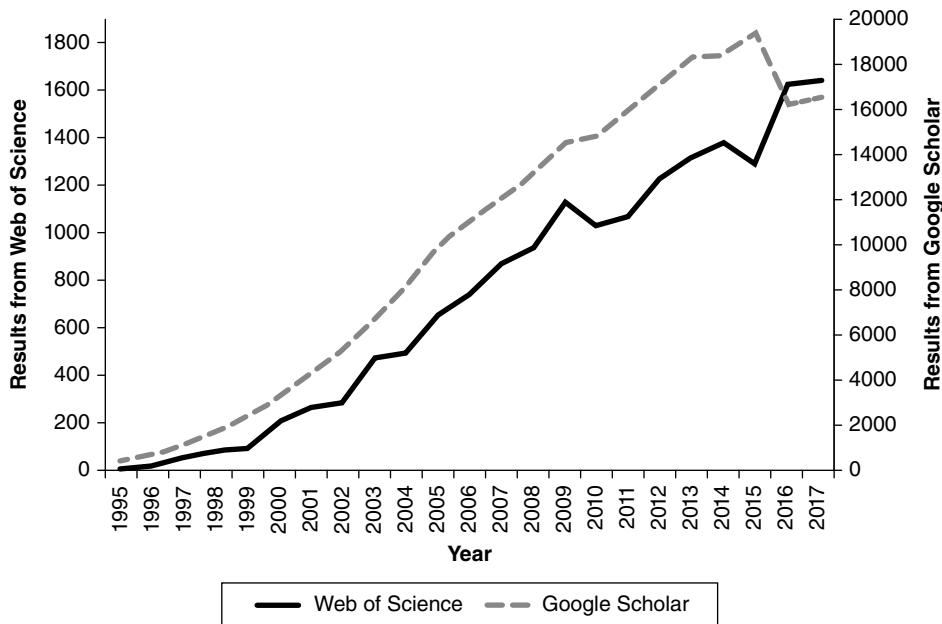


Figure 2.4 The growth in agent-based modelling: from search results of Web of Science and Google Scholar

2.4.1 Segregation

An example of how micro-level agent choices lead to the emergence of macro-patterns comes from the simple Segregation model of Schelling (1971). The model has been implemented in NetLogo by Wilensky (1997a). In this model we have two types of agents (red and green, in this example), as shown in Figure 2.5. Each agent wants to live in a neighbourhood, defined by its eight surrounding cells, where a certain percentage of neighbours are the same colour (here this is set at 40%). Initially the agents are randomly distributed throughout the environment. As the simulation begins, agents evaluate the colour of their neighbours: if their neighbourhood preferences are not met (i.e. the condition to move), they move to another cell. If their preferences are met, they remain where they are. In Figure 2.5, dissatisfied agents are marked with an 'X'. Over time, agents move to empty areas (i.e. the black cells) and segregated neighbourhoods slowly emerge at the aggregate level.

The utility of such a model is that even by changing just one model parameter different results can be observed. For example, by increasing the preference for the percentage of agents similar to themselves, more pronounced patterns of segregation will emerge, as shown in Figure 2.6. While this is a simple example, such mild

tastes and preferences for certain neighbourhood types have been shown to cause real world segregation to emerge (see Benenson et al., 2002; Mahdavi Adrestani et al., 2018). Moreover, such a style of model allows us to explore the underlying causes of segregation. While we are able to quantify the degree of segregation within neighbourhoods (e.g. Reardon and O’Sullivan, 2004), such analysis tells us little about the behaviour that leads to or from particular outcomes. Without this knowledge, trying to prevent such a process or phenomenon becomes challenging. One might think that individuals must have strong preferences for these racially or economically homogeneous neighbourhoods to emerge. However, empirical evidence suggests that individuals do not have strong racial preferences but have rather mild preferences (e.g. Clark, 1991). To find clear examples of this segregation process taking place is difficult, because it only becomes noticeable when it is clearly under way, and by then a detailed chronology becomes impossible to reconstruct (Batty et al., 2004b). To understand this behaviour, we have to examine how the process of individual choice leads to these outcomes, and this is where agent-based modelling comes in.

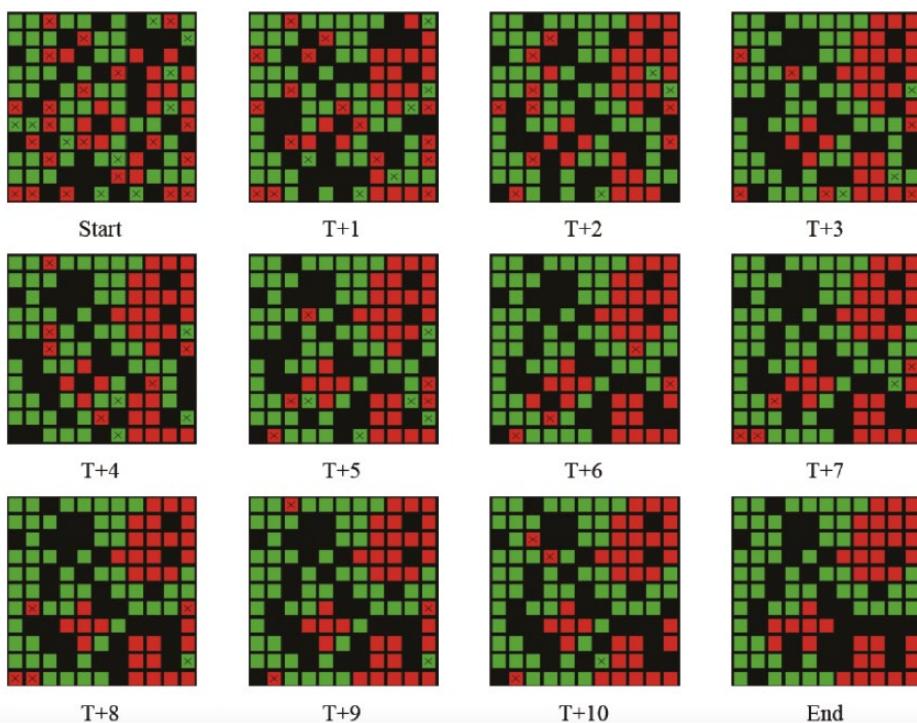


Figure 2.5 Progression of segregation over time: agents want to live in a neighbourhood where 40% are of the same colour

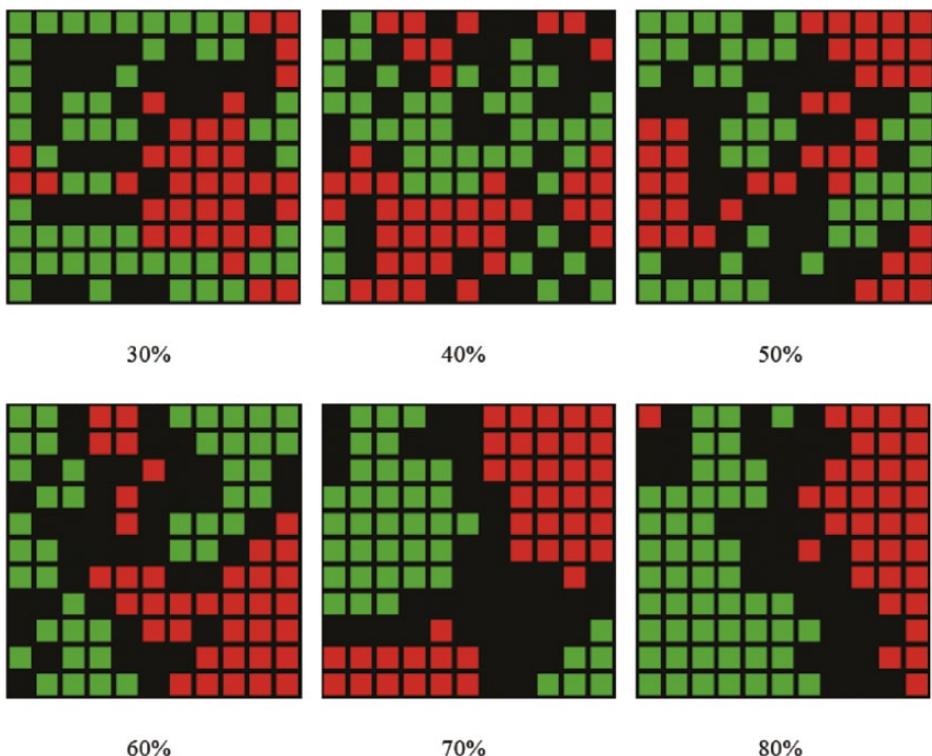


Figure 2.6 Examples of how changing agents' neighbourhood preference levels leads to different patterns of segregation emerging

2.4.2 SugarScape

Agent-based models did not begin to feature prominently in the geographical literature until the mid-1990s, when Epstein and Axtell (1996) extended the notion of modelling people to growing entire artificial cities. The goal was to understand the emergence of patterns, trends and other characteristics observable in society or geography. Epstein and Axtell's (1996) book *Growing Artificial Societies* introduced a series of models under the umbrella of SugarScape that demonstrated that agents could emerge with a variety of characteristics and behaviours suggestive of a rudimentary society (e.g. death, disease, trade, health, culture, conflict, war). From a geographical point of view what is particularly interesting about this model is that it demonstrates not only how terrain could be used in a model to act as its 'artificial world' but also how one can incorporate human and environmental interactions which are impacted by the landscape characteristics.

In Figure 2.7, an example model from Epstein and Axtell's (1996) book is recreated in NetLogo by Li and Wilensky (2009) to simulate how wealth distributions

emerge. Here, a terrain map is imported into the model and the values correspond to the amount of resources the cell has (i.e. the amount of sugar) which ranges from light to dark yellow (the importation of data into models will be discussed in more detail in Chapter 6). Initially agents are randomly distributed over the terrain, as shown in the top of Figure 2.7, and assigned a certain amount of sugar, an age and a metabolic rate. Once the agents have been created the simulation progresses following a few rules applied to the agents and the environment. At each time step, an agent checks its surrounding cells and moves to an unoccupied cell with the highest amount of sugar (in this model, sugar is synonymous with

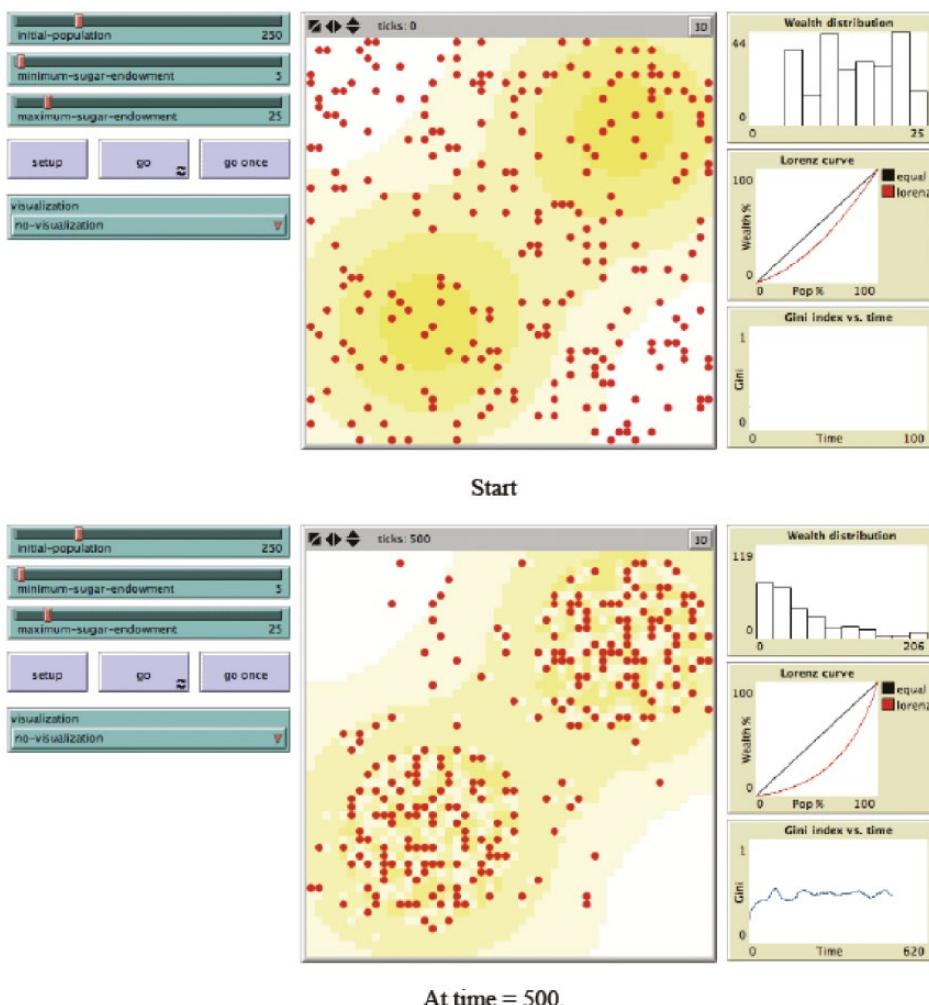


Figure 2.7 SugarScape wealth distribution model (Li and Wilensky, 2009)

wealth), and eats all the sugar (and thus accumulates wealth). To simulate the need for food, each agent loses energy at each time step of the simulation based on its metabolism rate. An agent dies if it runs out of energy; otherwise it dies when it reaches a certain age. When an agent dies, a new agent is created and randomly placed in the environment. Thus the population remains constant throughout the simulation. As sugar is taken away from the environment the amount of sugar declines, but to stop the landscape becoming barren, sugar also grows back at a specified rate. Through the interplay of these simple rules, we see agents clustering in high values of sugar and we can see how wealth is distributed over the population (i.e. the Lorenz curve) and the inequality of wealth distribution (i.e. the Gini index) as shown in the bottom of Figure 2.7. What is remarkable about this model is that from simple initial conditions and rules, unequal distributions of wealth emerge, with only a small percentage of the population having above average wealth.

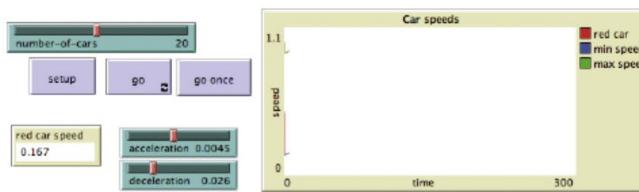
2.4.3 Transportation Modelling

Transportation is part of many people's daily lives, and traffic congestion is the norm for many cities around the world such as Los Angeles, London, Istanbul and Beijing. The cost of congestion is not only the time citizens spend in traffic jams, but also pollution (e.g. NO_x, CO₂) and traffic accidents. As the population of cities increases, an efficient transportation system is essential for urban futures. With individual decisions being at the core of this issue (i.e. do I take the car or bus to work?), it is unsurprising that agent-based modellers have turned their attention to trying to provide solutions. Agents in these models are typically individual travellers who can make independent decisions about their actions (Raney et al., 2003). What emerges from the micro-movements of individuals are macro-patterns, such as traffic congestion (e.g. Beuck et al., 2008).

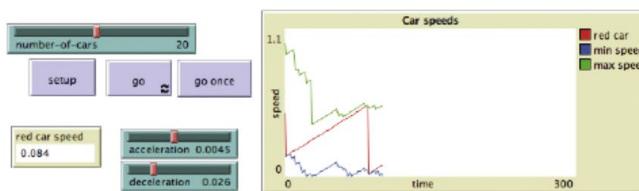
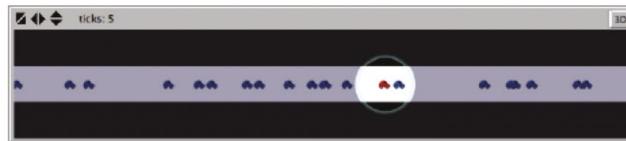
Agent-based models have been developed to explore many traffic-related issues, such as the mechanisms of stop-and-go traffic (Helbing, 2001), the potential for accidents between cars and people (e.g. Banos and Genre-Grandpierre, 2012), the impact of tolls for entering an area (Takama and Preston, 2008), how severe weather can impact the flow of traffic (Zhao and Sadek, 2012), evacuation routing (Wise, 2014), the effects of restricted parking (Benenson et al., 2008), the impact of the closure of a section of a road (Manley et al., 2011) and how one can improve the flow of traffic during the journey to work by combining traffic light controls and speed limits (Nagel, 2003).

To give a simple example of how these models work, consider how a traffic jam forms in the opposite lane to a traffic accident, a consequence of 'rubber-necking'. We can explore such a question within an agent-based model as shown in Figure 2.8. In this example each car is an agent, and is given two very simple rules of behaviour: first, if there is a car in front of it, the car slows down; and second, if there is no car in front, the car speeds up. These two simple rules applied

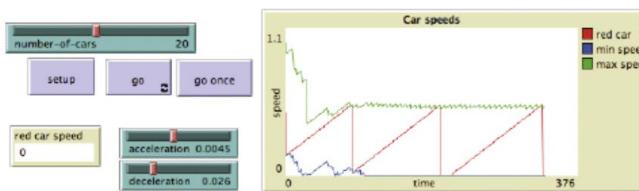
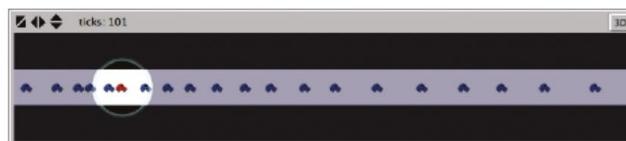
INTRODUCTION TO AGENT-BASED MODELLING



A



B



C

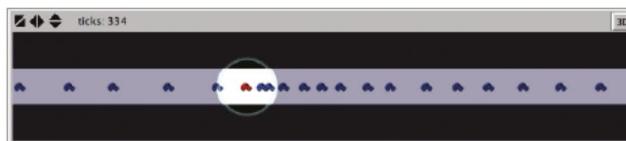


Figure 2.8 Simple traffic model where each car is an agent. In (A), (B) and (C) from top left clockwise: model parameters, a chart of car speeds and the spatial agent environment (Wilensky, 1997c)

to many agents can demonstrate how traffic jams can form without any serious incident. Moreover, through the modelling of individuals, we can see that there is great variation between individual car speeds which would be lost if we only looked at the aggregate (average) speed of traffic (such behaviours are explored in more detail later). However, to answer the question of whether simple rules can explain the emergence of traffic jams, readers are referred to Sugiyama et al. (2008), while for more complex agent-based models using similar rules, see Dawson et al. (2011).

2.4.4 Decision-Making

While the examples above might be considered rather abstract or academic, that is not to say that agent-based models are not being used outside of academia. Agent-based models have been used to support decision-making in a number of contexts. In this section we provide a brief review of some such models. Many of the early examples of the use of agent-based models come from the work of the BiosGroup,¹ an offshoot of the Santa Fe Institute which carried out over 50 projects for Fortune 500 clients (Mackenzie and Tzar, 2002). Such work from the BiosGroup included using an agent-based model to improve how Southwest Airlines handled cargo (Seibel and Thomas, 2000), how the Société Générale Bank in France could test procedures to minimise risk with respect to errors in trading (Bonabeau, 2003b) and how the decimalisation of the NASDAQ (i.e. moving to a decimal system rather than dollars and fractions of dollars) might impact stock market behaviour (Darley and Outkin, 2007).

Other examples of where agent-based models have been used in the private sector include: using agent-based modelling to explore capacity and demand in theme parks (Bonabeau, 2002); how Eli Lilly used agent-based modelling for drug development; Pacific Gas and Electric using agent-based modelling to see how energy flows through the power grid; Macy's (a US department store) use of agent-based modelling for store design; Hewlett-Packard's use of agent-based modelling to understand how hiring strategies affect corporate culture (see Bonabeau, 2003a,b); and Procter and Gamble's use of an agent-based model to understand its consumer markets (see North et al., 2010).

As alluded to in Section 2.4.3, agent-based modelling is widely used to support the management and planning of vehicle and pedestrian traffic (e.g. Atkins, 2009; Helbing and Baitelli, 2011) and can be used to plan for disease outbreaks (e.g. Eubank et al., 2004). In a similar vein, agent-based modelling has also been used for wildfire training, incident command and community outreach (Guerin and Carrera, 2010). For example, SimTable (2017), which

¹ For more information about the BiosGroup, see <https://en.wikipedia.org/wiki/BiosGroup> and Mackenzie and Tzar (2002).

combines digital sandtables, agent-based models and GIS, has been used in a number of active fire areas including the 2016 Sand Fire in California;² a sample of its user interface is show in Figure 2.9. Agent-based models are being used more and more in estimating the spread of diseases such as Ebola (see Waldrop, 2017). Finally, from a river-management perspective, agent-based models (or, to be more precise, individual-based models as they are termed in ecology) have been developed to explore how fish populations react to environmental and biological changes in rivers, such as the InSTREAM model by Railsback and Harvey (2002). While the above examples are not a comprehensive list, it is hoped that these examples are useful to show how agent-based models are being used outside of academia. In the remainder of this book we will explore the steps needed to create geographically explicit agent-based models.

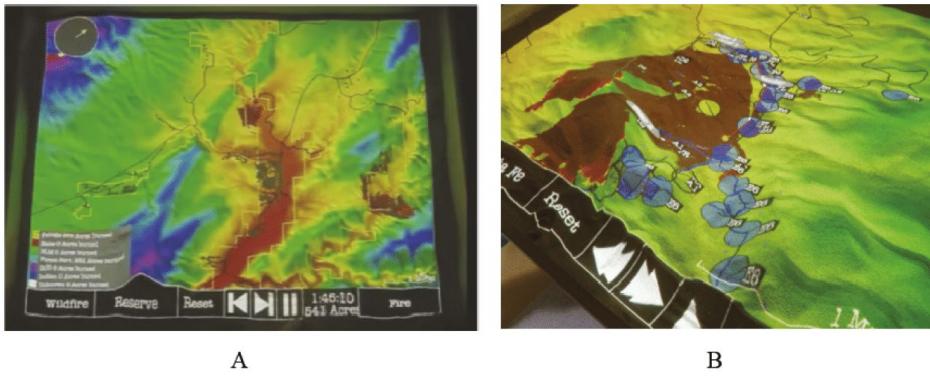


Figure 2.9 User interface of SimTable: (A) entire study area, (B) an active fire model (SimTable, 2017)

2.5 Discussion

Agent-based models have seen an unprecedented rise in popularity over the last couple of decades. This has been aided by increases in computational power, easy-to-use development environments (e.g. NetLogo) and new forms of rich micro-level data. For many applications that researchers within the social and geographical sciences are interested in understanding, the individual is key. Agent-based modelling puts the individual at the centre of the simulation and offers the

² See: www.simtable.com/ and www.cw6sandiego.com/sdsu-tracking-sand-fire-to-help-keep-firefighters

researcher the opportunity to both track and understand the consequences of individual decision-making over space and time.

There is no universal consensus about the definition of an agent, but there is agreement about the core elements that agents should possess. Perhaps the most important of these is the notion of being autonomous: an agent's ability to make decisions based on the knowledge it possesses about the world around it. This knowledge can arise from its own interactions with other agents, or from its interactions with its environment. Through these interactions, new knowledge or ideas can emerge. This 'bottom-up' approach to simulating geographical systems can potentially allow us to track the development of patterns (and thus shed light on processes) over multiple spatial and temporal scales.

One of the key challenges with agent-based models is designing the agents and the rule sets that will determine their behaviour and decision-making. These issues are further dealt with at various points throughout this book. Creating a rule set requires a detailed understanding of the variables and behaviours that produce the observed patterns. Often we lack suitable amounts of data to be able to mine this type of information from the systems under consideration. This is particularly the case when trying to understand the drivers behind dynamic processes – for example, how does segregation or gentrification occur in a neighbourhood? Many of the data sets that we have at our disposal are static, i.e. they reflect one moment in time. Agent-based models, akin to other types of models are data hungry. Depending on the application, detailed data is required on who the individual is (age, sex, education, etc.), patterns of mobility and, if possible, behaviour and opinions. Whilst there is an abundance of data on the former (e.g. national censuses), new forms of data, including mobile phone records and social media output, are increasingly being used to create a complete picture of an individual that can be translated into an agent.

While one of the main advantages of agent-based modelling is its ready ability to build reasonably 'realistic' agents by extracting information from a variety of data sources, there are also important constraints that the researcher needs to be aware of. What is the purpose of the model – is it to explain or to predict? Does the design of the model (the selection of variable and agent rules) lead to the emergence of the correct patterns? Is the model rigorously evaluated, and can we generate a level of confidence to accompany the simulation results? Evaluating agent-based models remains a significant challenge (see Chapter 10). It requires not only vast amounts of data, but also an understanding of what is being evaluated, a metric, pattern or process – and at what spatial or temporal scale? Whilst there are advances being made within this area, any design of agent-based modelling should include a detailed plan of how the model is to be rigorously calibrated and validated.

Agent-based models are firmly established as an important tool for understanding processes and dynamics within social and geographical systems. Through simulations, they allow researchers to uncover new knowledge, test out theories and gain insight into the inner workings of systems. However, these models require careful design and robust evaluation. These issues will be more closely examined in subsequent chapters.

Chapter Summary

Successfully replicating the processes and dynamics that occur within real world geographical systems is highly challenging. There are a potentially infinite number of individual components linked together by often unknown interconnected processes that play out at different spatial and temporal scales. The notion of bottom-up modelling advocated by agent-based modelling allows the results of local phenomena to be understood and measured at an aggregate level. While established methods, such as spatial interaction modelling, treat populations as aggregate homogeneous components, agent-based models potentially allow every individual to be assigned their own characteristics. This is a powerful paradigm that holds great promise for facilitating greater understanding of real world geographical systems.

This chapter has provided a general introduction to agent-based modelling. The main characteristics of agent-based modelling have been presented along with the advantages and limitations of this approach for geographical systems. The chapter concluded by exploring a diverse range of geographical applications of agent-based modelling and setting the scene for the remainder of the book.

2.6 Annotated Bibliography

For a general introduction to agent-based modelling along with other social simulation techniques, readers are referred to:

- Gilbert, N. and Troitzsch, K.G. (2005) *Simulation for the Social Scientist* (2nd edn). Maidenhead: Open University Press.

For agent-based modelling and geographical systems, readers are referred to:

- Batty, M. (2005) *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-Based Models, and Fractals*. Cambridge, MA: MIT Press.
- Benenson, I. and Torrens, P.M. (2004) *Geosimulation: Automata-Based Modelling of Urban Phenomena*. Hoboken, NJ: John Wiley & Sons.

A great resource for agent-based modelling in general and using NetLogo is:

- Wilensky, U. and Rand, W. (2015) *An Introduction to Agent-Based Modelling: Modelling Natural, Social, and Engineered Complex Systems with NetLogo*. Cambridge, MA: MIT Press.

Web resources:

- www.openabm.org/ A collection of agent-based models, accompanying documentation, along with resources to get up to speed quickly with agent-based modelling.
- www.intro-to-abm.com/ The website containing models and other material from Wilensky and Rand (2015).
- <https://ccl.northwestern.edu/netlogo/> Agent-based modelling environment which we use extensively throughout this book.
- www.abmgis.org/Chapter_2 Here you will find all the models from this chapter as well as additional resources.

3

DESIGNING AND DEVELOPING AN AGENT-BASED MODEL

Chapter Outline

What are the questions that social scientists and geographers need to consider when designing and building an agent-based model? What design frameworks and software toolkits are available to use? What are their relative pros and cons? What methods are available for documenting design concepts and why are they useful to modellers? This chapter will introduce the core concepts and frameworks that can be used to plan, implement and disseminate geographical agent-based models.

3.1 Introduction

Although ABM [agent-based modelling] is technically simple, it is also conceptually deep. This unusual combination often leads to improper use. (Bonabeau, 2002)

What to leave out, what to include? It is neither possible, nor useful, to model everything. Therefore modellers always need to abstract from the ‘real world’ and focus on the specific question at hand. A model is a simplification of reality that takes theoretical abstractions and puts them into a form that can be manipulated. Model outcomes generate data that can then be compared to the real world data, and this allows an assessment of how well the model captures the real system that it is attempting to replicate. Figure 3.1 (from Gilbert and Troitzsch, 2005) illustrates this basic process. It begins with a target system (i.e. the real world), from which abstractions (i.e. theories about how the system works) can be constructed. These are used to create a model, from which new data can be generated via simulation. These modelled data can be compared to those collected from the real system (i.e. real world data) to determine how similar the model abstraction is to the real target system.

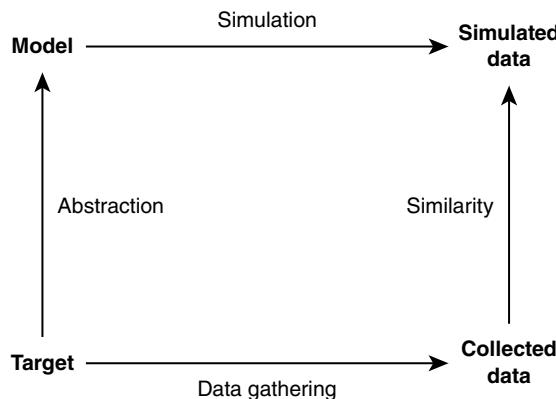


Figure 3.1 Logic of simulation (Gilbert and Troitzsch, 2005, p. 17)

This process is somewhat unusual in comparison to traditional scientific methods. In defining the purpose of agent-based modelling, Axelrod (1997a) writes that traditional modelling within the social sciences was either *deductive* or *inductive*. However, agent-based modelling can be described as a ‘third way’:

Like deduction, it *starts with a set of explicit assumptions*. But unlike deduction, it does not prove theorems. Instead, a simulation *generates data that can be analyzed inductively*. Unlike typical induction, however, the simulated data comes from a rigorously specified set of rules rather than direct measurement of the real world. While induction can be used to find patterns in data, and deduction can be used to find consequences of assumptions, simulation modeling can be used as an aid to intuition. (Axelrod, 1997a, p. 24; emphasis added)

In essence agent-based models can be viewed as:

laboratories, where we attempt to ‘grow’ certain social structures in the computer or *in silico*, the aim being to discover fundamental local or micro mechanisms that are sufficient to generate macroscopic social structures and collective behaviors of interest. (Epstein and Axtell, 1996)

Agent-based modelling is potentially an extremely powerful methodology. However, it is easy to include too much complexity and to over-complicate models unnecessarily (Batty and Torrens, 2005). This has the potential to lead to models that, rather than simplifying a system to aid understanding, become even more difficult to understand than the real world. To mitigate these risks, the agent-based modelling community has developed a suite of approaches that can inform the design, implementation and testing of models. These include innovative means of evaluating models such as pattern-oriented modelling (POM: Grimm et al., 2005)

as well as standard approaches to the design and documentation of models. The most popular of these is the Overview, Design concepts and Details (ODD) protocol (Grimm et al., 2006, 2010), which has recently been extended to incorporate more options for describing human decision-making (ODD+D: Müller et al., 2013). This chapter will discuss some of these methods in order to illustrate best practice in building reliable agent-based models.

Figure 3.2 presents an overview of the typical model design process. It is important to note that this is not the only process that a modeller might use in order to create their model. For further ideas, the interested reader should refer to any of the excellent agent-based modelling textbooks that are available, such as Wilensky and Rand (2015) or Railsback and Grimm (2011), and for steps in modelling within the social sciences more generally, see Gilbert and Troitzsch (2005).

The remainder of this chapter will discuss some of the considerations required during the first stage indicated in Figure 3.2, the design and preparation of an agent-based model. In later chapters we will discuss the implementation and validation of our models. The framework outlined below is hierarchical in structure, moving from high-level questions around the purpose, development process and evaluation plan for the simulation, through to the model environments, and eventually to the behavioural mechanisms through which the simulation will unfold. The framework is accordingly split into several sections: first an overview (Section 3.2), before moving onto the simulation world to be created (Section 3.3), next considerations with respect to interactions between agents (Section 3.4), and finally agent behaviour (Section 3.5). Although this framework is flexible and might not be followed rigorously, it should at least act as a checklist and set of considerations to be borne in mind when considering the development of an agent-based model. Following on from the framework, the application of this structured approach to the development of an agent-based model is outlined with reference to the development of an existing model of residential segregation (Section 3.6). The chapter concludes in Section 3.7 with a summary of alternative design frameworks for agent-based simulation development that have been proposed elsewhere.

3.2 Overview

The *overview* refers to everything that exists outside of the simulation environment. These are the constructs within which the simulation exists, including the theoretical constructs, in addition to the noted role of the modeller, and the environment within which the simulation will be developed. It is important that all aspects of the design are considered prior to continuing with model development, so as to ensure that the modeller is taking the right steps in choosing to develop an agent-based model in lieu of other approaches. In outlining the overview, the following elements should be considered: the purpose and process

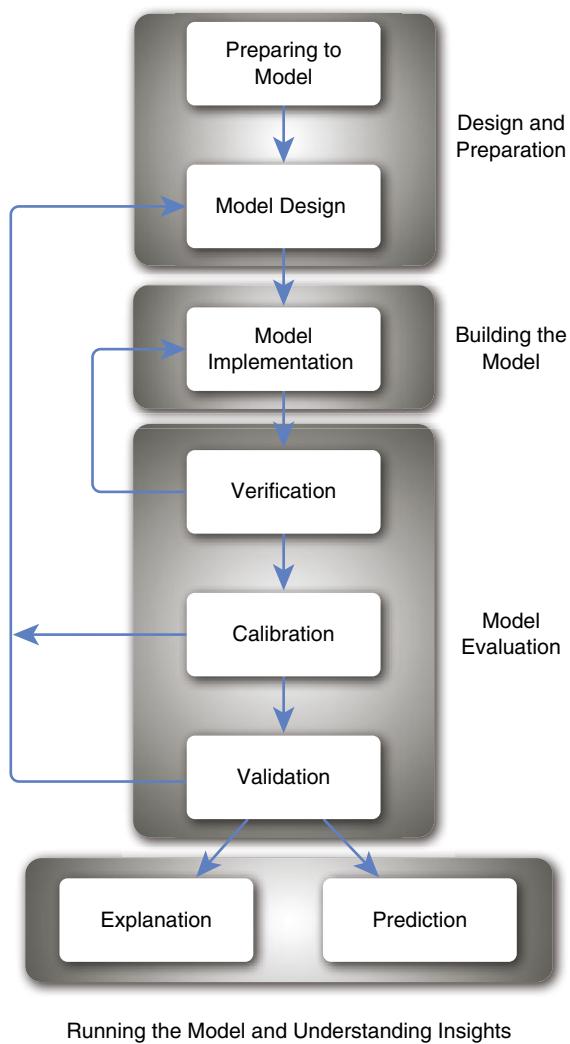


Figure 3.2 An overview of the typical modelling process

to be studied (Section 3.2.1), what data is required to build and evaluate the model (Section 3.2.2), how the model will be developed (Section 3.2.3), what sorts of visualisations are needed (Section 3.2.4) and considerations for bias and uncertainty within the model (Section 3.2.5).

3.2.1 Purpose and Process

The *purpose and process* encompass the main goals that should be achieved through the development of an agent-based model. The matter of the purpose of, or need

for, an agent-based model is intertwined with the social process being simulated. There are many potential reasons to model – perhaps the simulation will expose the behavioural mechanisms and interactions that influence observed social phenomena, test the implementation of a systemic change or policy implementation, or do something else completely. Clarifying *why* the simulation is needed will shape the way the rest of the model is developed (Epstein, 2008, cites 16 possible reasons to consider).

Along with a demonstrable purpose for a model, there must be a process to simulate. The process could feasibly cover any number of social, spatial, interactive or mathematical phenomena. When considering the use of an agent-based model, it is important to consider who the agents might be, how they are involved in the process, how interactions manifest, and what are the features of the environment within which this process takes place. These will all be defined in more detail later, but initial consideration (and research) is important at this stage. One of the benefits of agent-based modelling is its lack of model constraint, and therefore one should not necessarily shy away from modelling where considerable complexity does exist. For example, it is relatively trivial to include a vast array of agent heterogeneity; this is something that is much more difficult with some aggregate approaches.

Once the model purpose and process are clear, it is important to clarify whether agent-based modelling is the most appropriate approach. As Chapter 11 will illustrate, there are various other approaches for modelling social phenomena, each with their own advantages and disadvantages. These need to be understood before embarking on an agent-based model, to ensure that the methodology has been chosen for the correct reasons. Importantly, consider whether or not an agent-based model is going to be able to answer the question of interest, and, even if so, whether it is the most efficient approach to use.

3.2.2 Data Collection and Evaluation Plan

With a model purpose and process in mind, the next consideration could be the data that are required to build, calibrate, and eventually evaluate the model. In most cases, good data will be central to the development of the model. They not only provide insights into the process itself and the agent behaviour, but also represent a benchmark against which the simulation parameters can be configured, and can be used to validate the robustness of the simulation when deployed in other contexts and settings.

Different data sets will be suitable for different tasks. Surveys, interviews, observation studies and individual data capture aspects of behaviour that may be used in the developing a model of agent behaviour (see Chapter 7). However, larger-scale data collection, such as ‘big data’, large-scale surveys and censuses, are better placed as a basis for calibrating and validating a simulation against real world phenomena.

As with any data set, prior to using it, its strengths, limitations, biases and gaps need to be well understood as these will impact the representativeness of the simulation. Also, consideration needs to be given to collecting data at multiple levels. For example, aggregate data might be appropriate for determining whether the global patterns that are produced by the model are reliable, but provide no information as to whether the *individual behaviours* of agents are robust. POM (see Grimm et al., 2005, and Section 3.7.1 for more detail) is relevant here.

Agent-based models are extremely flexible with respect to the data that they generate. The developer can choose what data is extracted from the simulation, how often it is extracted and the format used. This choice often depends on the validation data set that the model will be compared against. For example, in a traffic flow simulation, traffic counts over 5-minute periods at various points in the road network might be collected; with a model of crime events the output might be all individual crime occurrences and their spatio-temporal location, to be compared with corresponding police data. Although the simulation will reflect all interactions within the system of interest, data outputs from the simulation will ideally match the spatial and temporal constraints of the validation data, and therefore require forethought.

3.2.3 Development and Software

The next consideration should be how the model will actually be developed. There are a variety of agent-based modelling toolkits that employ a range of programming languages. In Chapter 6, a full review of the different approaches, and their various pros and cons, will be given. This book predominantly uses NetLogo (Wilensky, 1999), but it is worth carefully considering the requirements at an early stage, especially if the time taken to learn a new programming language offsets the benefits offered by a particular tool.

Aside from the technical skills required to use a particular programming language, the size and scale of the potential simulation also need to be considered. Simulations of thousands or millions of agents with a high degree of behavioural complexity might need to be executed on large-scale computation facilities. If this scale is reached, then identifying a framework that supports that level of expansion is vital.

At this stage it is also important to consider whether there are other simulations that could be extended. Model development can be time-consuming, so adapting existing models can be extremely useful. Provided a simulation has been peer-reviewed and built on reliable, well-documented assumptions, this could reduce the need to implement some aspects of the model.

3.2.4 Visualisation

The *visualisation* of an agent-based model can be important in two respects. From a more functional perspective, it gives the modeller reassurance that the

simulation is working as expected (Mandelbrot, 1983; Grimm, 2002). Obvious problems with the behaviour of the agents can become visible and might highlight errors in the model logic. More pragmatically, a visualisation can really impress an audience. Never underestimate the power of a colourful, dynamic visualisation to impress people and get them excited about the work! It may well be that most of the work has been directed at building, calibrating and validating the simulation, but the visualisation can be the pretty icing on the cake that really engages people.

Nevertheless, prior to starting development, it is important to consider what precisely should be shown, and what the computational cost involved in generating that representation would be. The visualisation should be developed at a scale that reflects the key dynamics in a meaningful way, and allows the modeller to tell a strong ‘story’ about what has been found. It is also important to think about the aggregate, emergent patterns that are being generated through the simulation, and how these can best be reflected. Visualisations can also include charts and figures that summarise the outputs of interaction. NetLogo is particularly good at combining outputs in this way, and updating at custom intervals. A key decision here will be identifying the most meaningful scale and interval at which to collect and simulate the data. But remember that each visualisation requires resources, and can reduce the speed of models considerably.

If the model is being developed purely for aesthetic purposes, it is worth considering developing the model within visualisation software or a game development package, such as Unity or 3ds Max. While more time and computational resources are devoted to visualisation, the results can be visually outstanding. For further information about visualisation, see Section 5.8.

3.2.5 Biases, Uncertainty and Assumptions

A final point to consider when planning the model is the impact of biases and assumptions in the simulation. There are two key sources of bias to consider: the modeller, and the data. Modellers naturally bring their own background, perspectives and training when they look at any problem. This will influence how the model is developed, and thus the final results. Some modellers might be more interested in the mathematical properties of social interaction, some in the spatial evolution, others in the properties of the social network. Each approach is valid, and will result in a different perspective on the problem. It is important to consider the approach, and the validity of other approaches, when beginning to develop a simulation. A broader discussion of the biases associated with the agent-based modelling approach can be found in O’Sullivan and Haklay (2000).

Data has many uses in the development of a model, and will be covered at various points in this book. However, data comes with its own set of biases, uncertainties and limitations. These are introduced through the method by

which the data is collected, who the data is collected about, the time period over which the data is collected and the purpose for which it is collected in the first place. And, importantly, these problems do not disappear with more and more data! It is important to be cognisant of the nature of the data that is being used to calibrate or validate the model (see Chapter 10). All such biases, uncertainties and limitations should be clearly stated.

3.3 World

The *world* refers to the modelled environment within which the simulation will take place. The world encompasses not only the physical constraints of the model, but also the global rules that define the behaviour of all objects within it. This definition should only be made in relation to the specifications made during the *overview* definitions, and not with respect to *agent* design.

Once more, a number of design aspects must be considered at this stage, only within the context of those design decisions made during the *overview* specification. Each design aspect again presents a number of questions that must be answered. These include specifying what is exogenous to the model (Section 3.3.1), how space (Section 3.3.2) and time (Section 3.3.3) are represented in the model, what constitutes the agent population (Section 3.3.4) and what physical rules need to be accounted for (Section 3.3.5).

3.3.1 External Systems

External systems relate to the wider context in which the simulation is taking place. These are systems which, while not directly involved in the simulation, could be considered to have an impact on the evolution of the process being simulated. Within a spatial context, these systems might typically include wider economic systems, social structures and norms, and transport patterns and flows. Designing the model requires a consideration of how to treat and involve these exogenous systems within the model itself.

In some cases, these systems will be important for defining the extent of the simulation. A traffic flow simulation, for example, will capture the interactions between vehicles on a road-by-road basis, resulting in the emergence of traffic flow and congestion. This environment is, however, influenced by a wider set of transport demand-flows defined by the distribution of homes, workplaces, schools, activities and the people wishing to use their cars to travel between these locations. Within any dynamic system, there are temporal evolutions to consider as well. Within a traffic simulation, however, we might not want to have to consider everything, and just take an indicative indication of demand that adequately accounts for these complex external processes.

In these cases, an aggregate modelling approach can be used to capture a demand input (e.g. system dynamics, spatial interaction models), or observation data from a real world example could be obtained. Both approaches will generate a demand profile for agents, defining where and when agents should appear within the simulation environment. The strengths and limitations of these approaches are described in more detail in Chapter 11. In many cases the influence of external systems may be so minimal, or so difficult to define, that they can be ignored for the purposes of simulation. Examples of these might include the influence of air streams on flocking behaviour, or the role of the US political system in UK house building. However, the modeller needs to consider them due to the notion of near-decomposability (introduced in Section 1.2).

3.3.2 Space

Within an agent-based model, *space* can refer to the representation of space over which agent movements and interactions will take place. Within each representation, further restrictions on movement may be imposed (e.g. walking and non-walking spaces), but the definition of spaces sets up the foundations on which all subsequent rules are defined. There are typically four main types of space found in an agent-based model, as shown in Figure 3.3. The usage of each involves different considerations:

- *Continuous space*. This form typically relates to abstract and non-geographical spaces where agents can move and interact without the constraint of a grid or other structure. Examples include pedestrian simulation, flocking and animal interactions, and exploratory social simulation (e.g. Torrens, 2014a).
- *Geographical space*. While similar to continuous space, this form of space differs by making explicit reference to a specific geographical location and associated coordinate system. And while not necessary for simulating all spatial interactions, this representation enables integration of spatial data recorded using the same coordinate system, which may provide location data for the agents themselves or additional context. Examples encompass all simulations of real world processes, such as housing, transport and crime. Geographical space is described in detail when the relationship between agent-based modelling and geographical information is explored in Chapters 5 and 6.
- *Cellular space*. Cellular space is a grid-like representation of continuous or geographic space. The locations of objects in the environment (including the agents) are restricted to the locations of grid patches. Examples include large-scale land-use models, traffic flow simulation and cellular automaton processes (e.g. Conway's (Gardner, 1970) Game of Life; see Section 11.2.1).

- *Topological or network space.* A topological space may also be referred to as a network, whereby objects are connected via edges, across which interactions take place. The network constrains the routes of interaction, and determines which objects are allowed to interact. Topological space may be combined with another form to represent different types of interaction. For example, a pedestrian agent may physically interact with other agents within a geographical space, but maintain a topological link to specific ‘friend’ agents with whom other interactions occur. Examples of topological models include social networks, trade networks and disease transmission (e.g. the SIRS model). Networks are described in more depth in Chapter 8.

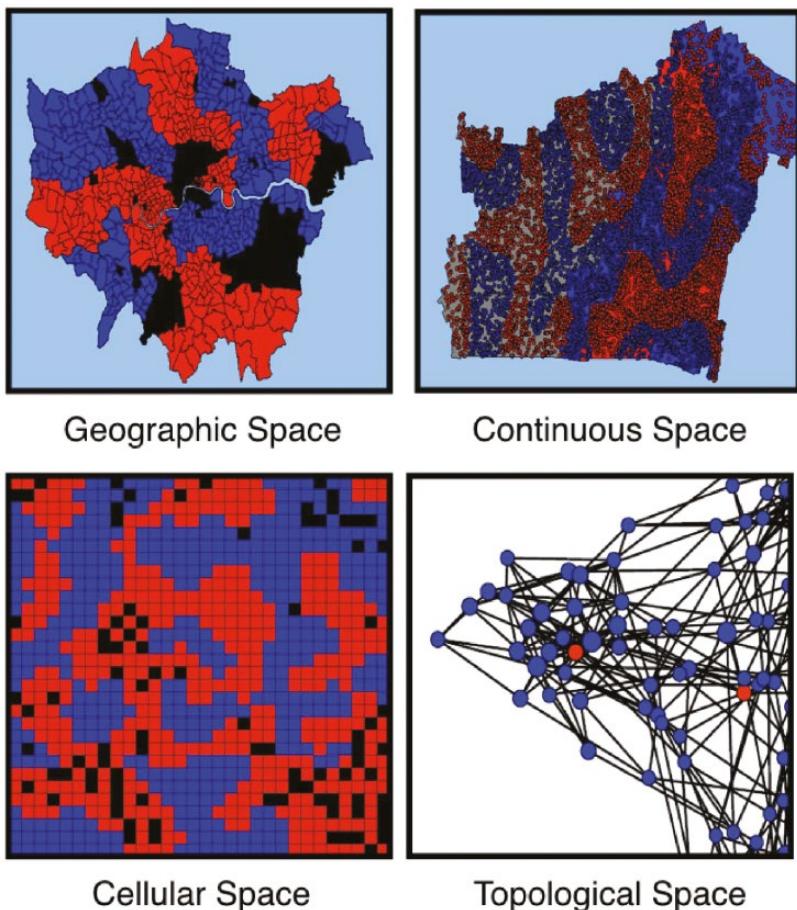


Figure 3.3 Different forms of spatial representations that may be used in agent-based models

In addition to defining the type of space, at this stage there should be a definition of the extent of the area to be simulated. The environment might be limited to a street, a city or a country – the choice must depend on the *process* being simulated, and should reasonably align with the typical scale at which it occurs. This is often common sense. For example, there is little to be gained from simulating the migration of wildebeest over a few hundred metres, while that may be a wholly appropriate area for simulating pedestrian movement. That said, some applications might require a more careful assessment of the areal extent. When modelling a city, for example, a careful assessment of the modelling boundary might need to be made so that the model is not overly disrupted by boundary conditions but equally does not model an inappropriately large area. Models that cover a large area need greater computational resources than those that cover smaller areas (assuming the resolution of the simulation is the same).

3.3.3 Time

Agent-based models are dynamic, and process agent behaviours at time-step intervals. This means that at each interval, agents will move, make decisions and interact with other agents. Therefore the time step must be short enough to capture all potential interactions, but not too short that one iteration is almost identical to the next. At each time step, these processes will need to be executed for all agents, requiring additional computational resources for simulations that have a high temporal resolution.

When setting a time-step interval, it may be useful to think about how much geographical movement an agent might be expected to conduct at each step of the simulation, as this is directly related to the behaviours and interactions that might be expected from them. To illustrate, consider a pedestrian simulation that models the effects of crowds interacting. If we set a time interval to equate to 30 seconds of simulation time, within one time step agents may have walked past each other, crossed over, bumped into each other, but those interactions would not have been captured. More realistically, a time step of 0.25 seconds would allow time for pedestrians to observe, decide and interact with other agents, creating realistic crowd dynamics where all interactions are captured.

The time step will also influence the visualisation of the simulation. Again, striking a balance is important – too short a time step and a viewer will lose interest, too long a time step and the simulation may be difficult to comprehend. Testing out alternatives, and thinking about how the results will be presented, is important at this stage too.

3.3.4 Populations

Prior to defining details about the agents, the *population* simply refers to the number of agents that will be included in the simulation. This value may be

predetermined either through a clear prior understanding of the number of agents required for the context, or due to being specified within an imported data set (e.g. census data, mobility flows). Alternatively, the exact number of agents may not be explicitly stated, and rather become a product of the environment. This could be the case where the number of agents is determined by a density factor for the given simulation space. A final alternative approach may be to specify a possible range in numbers of agents, with this parameter becoming a feature to explore during simulation calibration and validation. Population values will be set for each type of agent.

3.3.5 Physical Rules

Finally, in terms of setting up the simulation environment, it is worth briefly considering the *physical rules* that will influence the simulation space. The simplest example might be that a pedestrian agent cannot walk across water or that a developer cannot build on water. Physical rules become particularly relevant if agents are able to move in three spatial dimensions (e.g. up and down hills), where the effect of physical forces, such as gravity, may be more relevant (see, for example, Crooks et al., 2011; Torrens, 2014b). In many of the cases we are addressing here these will not be concerns we need to address.

3.4 Interactions

The specification of the *interactions* does not refer to defining the agents themselves, but rather how agent-to-agent interaction occurs. These definitions refer to the physical and collective rule sets that organise interactions. It is these interactions within the simulation that result in the emergence of the process that is under investigation. All of the definitions here are made within the context of the broader *overview* and *world* definitions made above.

The main elements of interaction design are physical (Section 3.4.1) agent-to-agent interactions, agent-to-agent communication (Section 3.4.2) and the sharing of resources (Section 3.4.3). In what follows, the relevant design questions will be outlined along with references to higher-level definitions where necessary consideration is required.

3.4.1 Physical Interactions

Physical interactions refer to the physical process by which agents react to their environment and to the presence of other agents. Many of the spatial and social systems described in this book are the product of a physical competition for space or resources, and driven by the movement of autonomous agents within that space.

While movement behaviour will be defined at the *agent* level, there should be some notion here of the outcomes emerging from interaction.

The modelling of agent-to-agent physical interaction can range from quite simplistic – where an agent will not move to a location if there is an agent already present – to very sophisticated – integrating rules of attractive and repulsive forces. More sophisticated rules of physical interaction are generally used where movement is highly influenced by the proximity of other agents – for example, in the cases of pedestrian or traffic simulation. Within Boyd's bird flocking simulation (Reynolds, 1987)¹, multiple forces act upon the bird agent: the intention to move forward, the repulsion of becoming too close to other bird agents and an attractive force to remain within the proximity to others (the flock). Within pedestrian simulations, the force of attraction might extend to family units or groups of friends, who will seek to maintain proximity, while avoiding strangers. Physical interactions might result in beneficial or detrimental effects for either side. For example, the physical interaction between a wolf and sheep is significantly detrimental to the sheep (i.e. it dies, as discussed in Section 4.2.1). Where the emergent properties arising from a simulation relate to the physical interaction of agents, the nature of those interactions becomes more important.

A second form of interaction occurs between the agent and its physical environment. These interactions might again take simple or more sophisticated forms, but are likely to differ from agent-to-agent interactions. Typically these forms of interaction will merely act to constrain an agent to a particular area. Examples might include limiting agents to the types of land they can buy, or restricting movement to along road spaces. Agent–environment interactions can also lead to *indirect* agent–agent interactions. For example, a sheep agent might eat some grass, which will have an influence on another sheep agent later in the simulation when it arrives to find all the grass already eaten.

Given the prior definitions we have made about the simulation, with respect to the *overview* and *world*, it is also worth considering the wider restrictions on interaction. The definition of space may completely rule out physical interaction, but it also might constrain the degree to which discrete interactions are modelled. Physical interactions might also vary temporally; for example, commuters seem happier to squeeze into a train carriage together in the morning than at other times of day. All of the design choices at this stage should be made with a mind to the wider set of definitions.

¹ The flocking simulation is available in the NetLogo models library under Sample models → Biology → Flocking.

3.4.2 Communication

A second form of interaction that might occur is agent-to-agent *communication*. In defining this form of behaviour, the main consideration lies around how communication might take place, a process that might be affected by a range of factors.

The first considerations are the role of proximity and connectivity. Communication may only take place when agents are within a certain distance of each other, or may be limited to a required topological connectivity implied by a ‘social network’ structure (described in Hamill and Gilbert, 2009). Communication may also vary with proximity, whereby information is lost at longer distances, or more prone to disruption. Communication may not be direct. Messages might be passed via a broker or, within a network structure, may pass via numerous other agents before being transmitted to the final recipient. Within a network configuration, the structure and connectivity of the network are vital for the speed and extent of communication, as will be discussed at greater length in Chapter 8.

The role of agent characteristics may also play a role in communication. Some agents may be restricted in their ability to communicate with others, based either on agent type differences or incompatible characteristics (e.g. one agent does not wish to communicate with another). Social rules might also be implemented that shape the nature of communication between different agents.

3.4.3 Resource Exchange

Resource exchange interactions between agents are an extension of communication. There is a vast diversity of items that might be exchanged within a simulation, including goods, money, diseases and information. Once again, this interaction process may underpin the emergence of the social phenomenon of interest.

The important design decisions involved in capturing resource exchange follow those for communication. Namely, it must be clarified how these exchanges occur, where and when the exchanges take place, and the limitations on exchange that may be imposed by agent-level definitions. There are important implications related to the transfer of resources. While communication does not necessarily lead to a change in agent state, the removal of goods from one agent to the benefit of another may change the behaviour of one or both agents. This transfer may be an important element in shaping the emergent properties of the simulated process.

In addition to the practical constraints of interaction, resource exchange is additionally defined by the willingness or ability to interact on the part of the agent. This might be defined by limitations set on the rate or volume of resource exchange (e.g. constraints on agents exchanging all they hold). Resource exchange does not necessarily need to be a mutually beneficial process. The negative outputs of one agent might adversely affect another. For example, within a pollution

simulation, the production of pollution by one agent may severely reduce the health of another.

3.5 Agents

The design of the *agent* refers to capturing the characteristics, actions and decisions that influence its behaviours, movements and interactions with other agents and the wider environment. It is ultimately these behaviours that shape how the overall process is modelled, and how it evolves over space and time. These decisions should be made with reference to the prior design stages. Careful attention should be paid here to the amount of detail added to the modelling, balancing simplicity and explainability (the ‘keep it simple, stupid’ (KISS) argument seen in Axelrod, 1997a) with the inclusion of evidence and knowledge (the ‘keep it descriptive, stupid’ (KIDS) approach described in Edmonds and Moss, 2004).

Once this environment has been defined, the definition of the agents involved in the simulation should proceed naturally. During this final process, design considerations such as choice of agent characteristics (Section 3.5.1) and actions (Section 3.5.3) should be decided. These will be examined in detail below.

3.5.1 Characteristics

The *characteristics* of an agent refer to the properties used to differentiate one agent from another. The characteristics that are built into an agent would usually directly relate to the models of *decisions* and *actions*, as well as the types of interaction the agent is allowed to engage in. These characteristics both distinguish one type of agent from another type (e.g. a bus agent and a pedestrian agent), and refine descriptions of agents at a finer scale (e.g. a diesel-powered bus and an electric-powered bus).

Once the characteristics that agents hold have been decided on, the specification of characteristics to agents is typically made through probabilistic assignment. This process assigns a value to an agent based on a measured or theorised distribution of this characteristic. As an example, consider creating a population of 100 pedestrian agents and assigning each pedestrian a ‘desired walking speed’. There is a good understanding of the distribution of this characteristic within the population of interest, achieved through observations. As the 100 agents are created, values are drawn from this probability distribution and assigned to specific agents. Once all of the agents have been built, the population should be broadly representative of the observed population, providing a reasonable basis from which to start the simulation.

When drawing from a distribution in modelling agents, it is important to be aware of and design for cross-correlated attributes. In these instances, one agent

characteristic is dependent on or affected by another. Taking the example above, the ‘desired walking speed’ might be strongly associated with age. In these cases, if both attributes are relevant to the simulation, then the association should be captured during the creation of the agent.

It is often the case that data representing the characteristics of the model are not available. This could be due to difficulty in measuring the attribute, or the development of the simulation within an unmeasured or uncertain context. In these cases, estimates of the distributions of values that might be represented within the population should be made. Typical approaches will include assignment of categorical values to agents with predefined or equal probabilities, and creation of a distribution of values from a known mean and standard deviation (e.g. according to a normal distribution). Note that for these parameters, in particular, sensitivity testing should be used to quantify the impact that variations in their values will have on the model outcomes. If a model is very sensitive to small changes in a parameter value, then it is especially important that the parameter value has been reliably estimated. Chapter 10 discusses model evaluation (including verification and validation), and Section 10.3.2 discusses sensitivity testing in particular.

A final, special characteristic that deserves mention in the context of spatial simulation is that of an agent’s ‘start location’. Start locations can be assigned randomly, but will usually be constrained or weighted towards particular locations, and be dependent on the time at the start of the simulation. Examples include assignment to household locations, perhaps weighted more towards areas of greater population density, and road space, weighted towards areas of higher traffic flow. Where a simulation is designed to include a ‘warm-up period’, the initial start location of agents is somewhat irrelevant, as results will only be taken once agents are satisfactorily distributed across the simulation space.

The stochasticity inherent in the definition of agent characteristics is an important aspect of agent-based modelling. It captures the heterogeneity intrinsic to agent behaviour, and with it the uncertainty associated with predicting social and spatial phenomena. It also requires us to carefully calibrate and validate the simulations, as will be discussed in Chapter 10.

3.5.2 Decisions

Decision models are fundamental to an agent-based model, forming the link between modelled agent behaviour (in the form of decisions and subsequent actions) and the simulation of collective patterns. Decisions relate to any choice process that drives an action, and can be influenced by both the agent characteristics and their local environment. Like characteristics, there are a variety of decisions that could be incorporated within an agent-based model. The nature of

these decisions can be simple or complex, as can the modelling of those decisions, which might involve simple rules or optimisation, or be underpinned by a more sophisticated cognitive framework.

Decisions are made through access to information. Within an agent-based model, the information is typically obtained dynamically, so as agents traverse the simulation environment, they observe their environment, interact with other agents and develop a changing understanding of their world. With reference to this information, an agent may either make a decision immediately, or defer the decision until adequate information becomes available (e.g. a threshold is met). When designing the decision-making of an agent, it is necessary to estimate how often decisions should be made, and how information to make decisions is accessed.

With the information in hand, a second design choice is made around how the agent processes that information. The design of the decision-making model can vary considerably in terms of the input data and choice mechanics. At the simpler end of this spectrum are decisions based on the local area. In these cases, an agent assesses its surrounding environment, in terms of the nature of agents present (differentiated by type or characteristic), and makes a decision related to their presence. In a foraging model, the agents search for a pheromone or food, and when they find it make a decision to return to their nest; in a segregation model, the agents analyse their surrounding neighbours, and make a choice accordingly. Decisions might also consist of simple optimisation processes, such as the selection of the shortest path between an origin and a destination.

More sophisticated decision models might incorporate hierarchical rule structures, potentially making reference to more agent characteristics, or calculate the utility of different choices based on weights associated with each option. A description of the variety of behavioural models that might feature in an agent-based model is given in Chapter 7. Despite the variation in model complexity, this does not always equate to a complexity in simulation dynamics. Some of the most complex spatial processes can be observed through very simple movement behaviours, made with reference to only neighbouring cells.

3.5.3 Actions

The final choice that needs to be made in the course of designing an agent-based model is around the actions that the agents perform. Actions are initiated in two ways: through planned repetitive performance; or through initiation following a decision of some kind.

Repeated actions are those which an agent undertakes on a regular interval, usually at each simulation time step. A regular (scheduled) action could be a simple check of an agent's environment to gather information (e.g. how many similar

agents are nearby), a movement of a predetermined distance (e.g. 5 metres, 5 grid squares), gain or loss of resources (e.g. money, energy), or adjustment of direction of travel (e.g. relative to other agents). These actions might apply to all agents, or be determined by the agent type or characteristics. These types of actions relate to the dynamic nature of the simulation, and in moving and gathering information the agent will interact with and act upon the simulation environment.

The second type of actions are those prompted by decisions. Once an agent, through its interaction with the environment and other agents, has the basis for making a decision, an action completes the behavioural process. These types of actions will usually result in a stronger impact on the process, and occur on an *ad hoc* basis. Examples might include the decision to buy a house, to seek food or to change lanes in traffic. An important aspect of the design process will be modelling how often and under what circumstances these decisions should take place.

3.6 Building a Segregation Model

The design framework laid out above provides the basis for developing an agent-based model of a system of interest. There is clearly a lot to consider, therefore this section outlines how one might approach the development of a simple agent-based model. The example used is that of Schelling's (1971) segregation model, which was introduced in Section 2.4.1. The model, as implemented by Wilensky (1997a), is available in the NetLogo Models Library, under Sample Models -> Social Science -> Segregation. Relevant NetLogo code examples have been included throughout this section where relevant. Note that Chapter 4 will provide a detailed NetLogo tutorial, so it is not necessary to fully understand the code examples yet.

3.6.1 Segregation Model: Overview

Purpose and Process

The segregation model makes it possible to simulate the process by which spatial clusters of similar agents form over time, through the actions of individual decision-makers. In this case, the purpose of the investigation will be to explore the factors that may be involved in the agent's decision to remain (in which case they are satisfied) or move (when they are unhappy) and what impact this has on aggregate patterns of segregation.

Data Collection and Evaluation Plan

The segregation model is typically deployed within a hypothetical context to explore the fundamental dynamics of behaviour choice and segregation, rather

than in an empirical setting. The hypothetical approach will be maintained here. As a result, data collection is not required in this instance, and analysis will focus on the impact of the variation in parameter settings.

Development and Software

The model will be developed using NetLogo. The software provides a control and visualisation interface, as well as tools for performing actions such as parameter sweeps² (these are discussed in detail in Chapter 10) for testing parameters and analysing the resulting dynamics.

Visualisation

As the simulation will focus on the emergence of spatial clusters and the effect of agent contentment, visualisations will incorporate both of these aspects. A map visualisation will show how the distribution of the different types of agents evolves over time, while charts can be used to show levels of contentment (charts in NetLogo are discussed in detail in Section 4.4.4). These visualisation components are illustrated in Figure 3.4. The code to create the charts is straightforward. For example, to create the graph of the number of ‘happy’ (i.e. content) turtles, the following code will make the calculation:

```
plot count turtles with [ not happy? ]
```

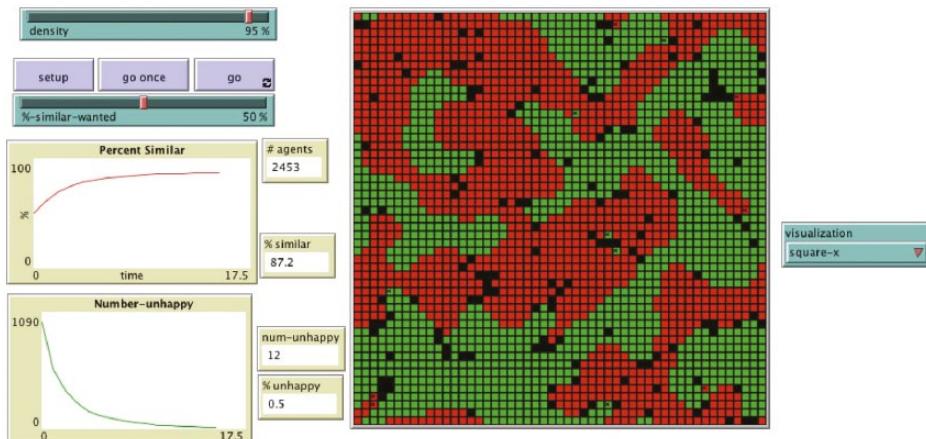


Figure 3.4 The visualisation tools used in the Schelling model as implemented in NetLogo (Wilensky, 1997a)

²The NetLogo Behaviour Space tool provides the mechanism to perform parameter sweeps.

Biases, Uncertainty and Assumptions

There are a variety of assumptions and simplifications within the model. It is assumed that there exists a social and spatial environment where segregation emerges from simple choices relating to nearby agents. There are, of course, a wide variety of additional factors and constraints that affect where a person chooses to live and when they might move. In this simulation, these wider complexities are ignored in favour of a simplification of the decision process. This reduced complexity allows a clarity of understanding of the possible drivers of segregation that would be more difficult to isolate in a more complex model. Noting these limitations, at this stage the aim is not to predict real world phenomena, but rather to focus on the potential impacts of different behaviours on *possible* segregation patterns.

3.6.2 Segregation Model: World

External Systems

This simulation is not set within a real world context, so the degree of interaction with external systems can be controlled. In the case of segregation, one could consider including housing markets, birth and death rates, wider social trends and sentiment, and so on. However, through a simplified approach, the focus will be on the dynamics occurring within the simulation space that result from relatively straightforward agent behaviours.

Space

The representation of space is an important part of a segregation model, as agents use the proximity to different types of agent to assess the suitability of their location. Given the focus on spatial clusters, without application to a real world context, the essential choice here relates to either continuous or cellular space. Continuous space would be suitable if the aim was to allow agents to move unequally across the environment. This would make it possible to represent varying densities of spatial configuration, as are observed in real world environments, and enable variations in how neighbourhoods are assessed (e.g. through the use of variable neighbourhood buffer radii). Cellular space limits movement to within discrete cells. This creates a less complex environment, but one that is not as easy to equate to a real world context. Since the focus here is purely on behavioural dynamics, the simpler cellular space will be used, noting again that the assumptions and limitations are made clear from the beginning.

Time

Within the modelled environment, the simulation need not proceed in line with any specific unit of time. The number of simulated iterations ('ticks') will be the sole measure of time, and there are no limitations on the length of the simulation – except when all agents are satisfied, in which case the simulation effectively stops as there are no more decisions to make.

Populations

The agent population is determined through a density parameter, expressed as a percentage of the simulation cells to be inhabited by agents at the start of the simulation. The parameter is provided for the user to vary (via a slider in the simulation interface) with the aim of exploring the impact of varying density on cluster formation. The density ranges from 0 (no agents) to 100 (every patch houses an agent) and can be used to spawn agents by testing whether or not a randomly chosen number between 1 and 100 is less than the current density value:

```
ask patches [
  if random 100 < density [
    sprout 1 [
      set color one-of [red green] ;; Equal chance of being red or green
    ]
  ]
]
```

Note that Chapter 4 will introduce the NetLogo language in detail, so don't worry if the code presented here seems confusing.

Physical Rules

There is an assumption that no two agents can inhabit the same cell at the same time. Other than that, there are no constraints on movement; every unoccupied cell is an equally valid and likely choice for an agent's destination. The agents choose a new location by walking randomly over the world until they find a patch that is unoccupied:

```
to find-new-spot
  rt random-float 360 ; Rotate to face a random direction
  fd random-float 10 ; Move forward up to 10 patches in that direction
  ; Repeat until the agent finds an unoccupied cell:
  if any? other turtles-here [ find-new-spot ]
  ; If we get here then the agent has found a free patch
  ; Move to its centre
  move-to patch-here
end
```

3.6.3 Segregation Model: Interactions

Physical Interactions

In this model, the ‘neighbourhood’ is defined as the group of cells that are adjacent to an agent’s current location. There are two neighbourhood definitions that could be used: the Moore neighbourhood, which consists of the surrounding eight cells, or the von Neumann neighbourhood, which consists of the four cells that share a border with the agent’s cell (examples of these are shown in Section 11.2.1). Convention suggests using all eight cells, so that is the approach taken here, but a comparison of both approaches and their impact on the simulation dynamics could be undertaken. In NetLogo, the `neighbors` and `neighbors4` commands return the patches within the Moore and von Neumann neighbourhoods, respectively. Code Example 3.6.1 (discussed opposite) illustrates this.

Communication

With proximity acting as the main point of interaction, there is no direct inter-agent communication within this simulation. This could later be included as a factor affecting the decision to move location.

Resource Exchange

The focus here is on physical interaction, hence resource exchanges are not included.

3.6.4 Segregation Model: Agents

Characteristics

A key characteristic of agents within the segregation model is a ‘type’ indicator, against which each agent can assess similarity. Two different types are used here, denoted by different colours (‘red’ and ‘green’). Assignment of types to agents is done randomly, with agents receiving an equal chance of being allocated to either type. The following code (first presented under ‘Populations’ in Section 3.6.2) sets a new agent’s colour to either red or green.

```
set color one-of [red green]
```

A mechanism is also required to determine whether an agent is *content* with its location. This is done by setting the proportion of similar agents (e.g. those of the same colour) within the agent’s neighbourhood that are required for an agent to be content. This parameter is called `%-similar-wanted`. There are a number of different approaches for determining the value of this parameter: assigning a random value to each agent; drawing from a distribution; deriving the rules from data; or assigning a uniform value to the whole population of agents. Given

the focus on simplicity, a global value is used for this parameter (all agents have the same threshold). Adjustments can be made to the value of the parameter to explore the impact of varying sentiment. Code Example 3.6.1 (discussed below) demonstrates how to determine whether or not an agent is happy. If the agent is not happy, then it will move to an unoccupied patch in the next model iteration. Also, Section 10.3.2 demonstrates how to experiment with different values for this parameter and to explore the impacts of changes on the model outcomes.

Decisions

It has already been established that agents' decisions to move are based on the presence of other agents in neighbouring cells. Therefore the agents need information about the proportion of similar and dissimilar agents nearby. NetLogo can use the number of similar and dissimilar agents to calculate a proportion, and then make an assessment of contentment by observing whether or not the proportion is less than the %-similar-wanted threshold. If the proportion of similar neighbouring agents is greater than or equal to the %-similar-wanted value, then the agent is judged to be content. If the proportion falls below this threshold, a decision to move is made (the agent is classified as 'not happy') and it will move at the start of the next model iteration. This binary decision process is clearly very simplistic, but aligns with the broader aims of the model to explore phenomena in a hypothetical environment.

Code Example 3.6.1 illustrates how the proportions of similar and dissimilar agents can be calculated.

Code Example 3.6.1 The update-turtles procedure that determines whether or not an agent is happy with its their current location (Wilensky, 1997a).

```
to update-turtles
  ask turtles [
    ;; in next two lines, we use "neighbors" to test the eight patches
    ;; surrounding the current patch
    set similar-nearby count (turtles-on neighbors) with [ color = [ color ] of myself ]
    set other-nearby count (turtles-on neighbors) with [ color != [ color ] of myself ]
    set total-nearby similar-nearby + other-nearby
    set happy? similar-nearby >= (%-similar-wanted * total-nearby / 100)
  ]
end
```

Actions

The final aspect of behaviour to define is what occurs where a decision to move is made. Following a decision to move, the agent is randomly assigned to another

cell. This action is executed on the next time step following the decision, adding a small measure of latency between decision and action. The code that moves the agents around the environment in search of a new, unoccupied patch was outlined under ‘Physical Rules’ above.

3.7 Other Frameworks and Approaches

3.7.1 Pattern-Orientated Modelling

One of the major challenges associated with building agent-based models is deciding how much complexity should be present, and therefore what the structure of the modelling framework should be. The patterns that emerge from agent-based models can be highly complex, changing over different spatial and temporal scales. These emerging patterns can be seen as indicators of the processes that drive them. Having a model framework that contains the correct variables and processes is imperative if the model is to be able to reproduce the observed processes.

Frameworks are emerging to support identifying model behaviour at numerous hierarchical scales. Pattern-oriented modelling (Grimm et al., 2005) is arguably the best-known and most widely cited example. POM advocates the comparison of numerous patterns produced by models and their counterparts in real (non-simulated) data. Instead of attempting to understand the emergence of a process based on a single point at a single scale, POM advocates examining multiple points at different scales. This should allow an optimised model that is capable of producing observed patterns to be created. Grimm et al. (2005) argue that POM can help to reduce model uncertainty in model parameters in two ways. First, it creates models that are structurally realistic and therefore less sensitive to parameter uncertainty. Second, because of the realism of the structure and mechanisms within POM, the parameters interact in ways that are similar to interactions of real mechanisms.

The number of applications of POM are beginning to increase. For example, Magliocca and Ellis (2013) applied POM in an attempt to understand the patterns in local land use. Through alternating different model parameterisations, the ability of the model to reproduce spatial patterns was greatly improved. Railsback and Johnson (2014) used POM to identify different types of bird behaviours and test out different bird foraging theories.

3.7.2 Overview, Design Concepts and Details (ODD) Protocol

As noted in Section 2.4, agent-based models have developed rapidly over the past 20 years. This development has happened in an almost haphazard manner, with little structure or guidance to shape its evolution. There is still no central repository for code (although efforts are being made; see www.comses.net/) nor universal

guidance for the design and building of agent-based models. Recognising this issue, efforts have been made to formalise the approach to documenting agent-based models to allow these models to be more transparent and easily reproduced. The most prominent of these efforts is the Overview, Design concepts and Details protocol (Grimm et al., 2006, 2010). The increase in agent-based models describing human decision-making has recently resulted in an extension to the ODD: the ODD+D (Decision) (Müller et al., 2013). Figure 3.5 presents the ODD+D structure. The protocol is still shaped around the three main components that need to be documented: overview, design concepts and details. However, adaptations (highlighted in grey) were created to allow the protocol to be extended to human decision-making. Examples of the implementation of this protocol include Pires and Crooks (2017) and Orsi and Geneletti (2016) – details of these models can be found in Appendix A. While proponents of the ODD+D favour its clear structure and the relative ease with which even fairly complex agent-based models can be documented, it is not yet universally used. This is because, for some, the protocol is time-consuming to create and too restrictive to document complex and unique agent-based modelling implementations.

3.8 Discussion

This chapter has introduced some of the factors that need to be considered before embarking on the development of an agent-based model. One of the most important considerations is *what* should be modelled. A model can never replicate

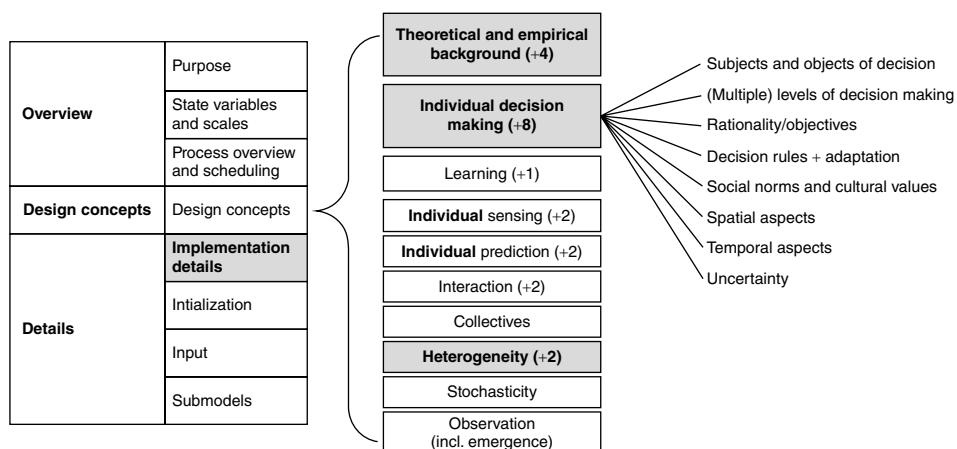


Figure 3.5 The structure of the ODD+D protocol. Grey boxes indicate new design concepts/categories compared to the ODD protocol. The numbers of added new questions are noted in parentheses. The different aspects of the new design concept of ‘individual decision-making’ are displayed on the right (Müller et al., 2013)

the real world in all its detail, indeed if it did then it would cease to be a *model*, so decisions need to be made as to what the most important factors to include are. These are not easy decisions to make, and it can take some considerable time to gather evidence about the system under study in order to isolate the most important processes and behaviours that ultimately drive the system.

Once these key drivers have been identified, it is equally important to decide whether an agent-based model is in fact the most appropriate method to answer the research questions posed. In some instances, alternative (and potentially much less complicated) methods could be leveraged. For example, if the important relationships express themselves adequately at an aggregate level, perhaps a simple (spatial) regression model will suffice.

Having decided that an agent-based model is appropriate, the chapter then moved on to present a framework that can be used to structure some of the decisions that need to be made with regard to the design and implementation of the model. It is important to note that rather than being a prescriptive list of everything that needs to be decided, the framework can function more as a checklist of decisions that need to be given at least some thought.

The chapter then concluded with a worked example of the framework that was applied to a model of segregation before pointing to two other important structures (ODD and POM) that it is useful for agent-based modellers to be aware of. However, once a model has been developed one also needs to consider issues with respect to verification and validation, which are the focus of Chapter 10.

Chapter Summary

This chapter has introduced and discussed the factors that need to be considered before embarking on building an agent-based model. It discussed the importance of deciding whether or not an agent-based model was appropriate in the first place, and then moved on to introduce a framework that can be used to structure the main design decisions.

3.9 Annotated Bibliography

For a good overview on considerations when designing an agent-based model, readers are referred to:

- Kornhauser, D., Wilensky, U. and Rand, D. (2009) Design guidelines for agent based model visualization. *Journal of Artificial Societies and Social Simulation*, 12(2). Available at <http://jasss.soc.surrey.ac.uk/12/2/1.html>.
- Abdou, M., Hamill, L. and Gilbert, N. (2012) Designing and building an agent-based model. In A. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 141–166. New York: Springer.

POM and the ODD protocol are important developments in the agent-based modelling methodology and were only briefly mentioned here. The interested reader could begin with the main papers:

- Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W.M., Railsback, S.F., Thulke, H., Weiner, J., Wiegand, T. and DeAngelis, D.L. (2005) Pattern-oriented modelling of agent-based complex systems: Lessons from ecology. *Science*, 310(5750), 987–991.
- Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S., Huse, G., Huth, A., Jepsen, J., Jørgensen, C., Mooij, W., Müller, B., Pe'er, G., Piou, C., Railsback, S., Robbins, A., Robbins, M., Rossmanith, E., Ruger, N., Strand, E., Souissi, S., Stillman, R., Vabo, R., Visser, U. and DeAngelis, D.L. (2006) A standard protocol for describing individual-based and agent-based models. *Ecological Modelling*, 198(1–2), 115–126.
- Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J. and Railsback, S.F. (2010) The ODD protocol: A review and first update. *Ecological Modelling*, 221(23), 2760–2768.
- Müller, B., Bohn, F., Dreßler, G., Groeneveld, J., Klassert, C., Martin, R., Schläuter, M., Schulze, J., Weise, H. and Schwarz, N. (2013) Describing human decisions in agent-based models – ODD + D, an extension of the ODD protocol. *Environmental Modelling & Software*, 48, 37–48.

4

BUILDING AGENT-BASED MODELS WITH NETLOGO

Chapter Outline

This chapter provides an overview of the programming language and concepts that are used within NetLogo (Wilensky, 1999). NetLogo basics, such as how to create a simple environment, commands and procedures, are presented with step-by-step instructions for creating a simple model. Following this basic model, more advanced features are introduced. The overall aim of this chapter is to provide an understanding of the main components that make a NetLogo program. Subsequent chapters build upon the basics presented here.

4.1 Introduction

NetLogo (Wilensky, 1999) was developed at the Center for Connected Learning and Computer-Based Modeling at Northwestern University and uses a dialect of the Logo language to create agent-based models (while NetLogo itself is written in Scala and Java). Due to the relative simplicity of the programming language and the development environment (see Section 6.3 for a detailed overview of NetLogo and other commonly used tools for developing agent-based models), it is one of the most popular tools for developing and testing agent-based models, with applications found in disciplines varying from biology and physics to the social sciences (see Wilensky and Rand, 2015).

However, as with all programming languages, NetLogo has its own syntax and structures that the programmer needs to be aware of. This chapter provides a practical guide to creating a simple NetLogo model. Section 4.2 presents the basics of NetLogo using the Segregation and Virus models from the Models Library as exemplars. Following this, an overview of NetLogo's basic commands is given in Section 4.2.1, before a first NetLogo model is constructed in Section 4.3.

Once this basic model has been constructed, more advanced features are presented in Section 4.4 using the Wolf Sheep Predation model as an example.

This chapter can be used as an introductory tutorial. To do this, download a copy of NetLogo v.6.0.2 (or a later version) from the Center for Connected Learning and Computer-Based Modeling (Northwestern University) website: <https://ccl.northwestern.edu/netlogo/>. For all of the commands that will be introduced within this chapter, further information can be found in the NetLogo Dictionary.¹

4.2 NetLogo Basics

NetLogo basics and functionality will be introduced through the use of two models. The first is Schelling's (1971) Segregation model (which was first introduced in Section 2.4.1) and the Virus model (Wilensky, 2005a). The purpose of using these two different models is to allow a range of features within NetLogo to be presented.

Schelling's Model of Residential Segregation

NetLogo ships with a variety of ready-made models in the Models Library. To see a list of the models that are available go to File → Models Library. The Segregation model is listed under Social Science → Segregation. Once opened (by double-clicking), Figure 4.1 illustrates how the model will look. Section 2.4.1 discussed the importance of this model in detail, so it is worth reviewing that discussion briefly if needed.

At each iteration of the model, every agent (or *turtle* as they are often referred to in NetLogo) examines its eight neighbours to calculate how many of them are of the same colour (red or green). If this is fewer than a specified percentage, then the individual or agent is deemed unhappy with its location and will move to any unoccupied cell. Note that in NetLogo, the environment is treated as a grid of cells; these cells are often referred to as *patches*.

The model has the following graphical elements:

- Depending on the version that you have, there will be two or three buttons – setup, go once and go. The setup button initialises the model by creating the individual agents and configuring the environment. The go button starts the model running until it is stopped (by pressing go again). The go once button runs the simulation for one iteration.

¹ Go to <https://ccl.northwestern.edu/netlogo/docs/dictionary.html> and search for the relevant command.

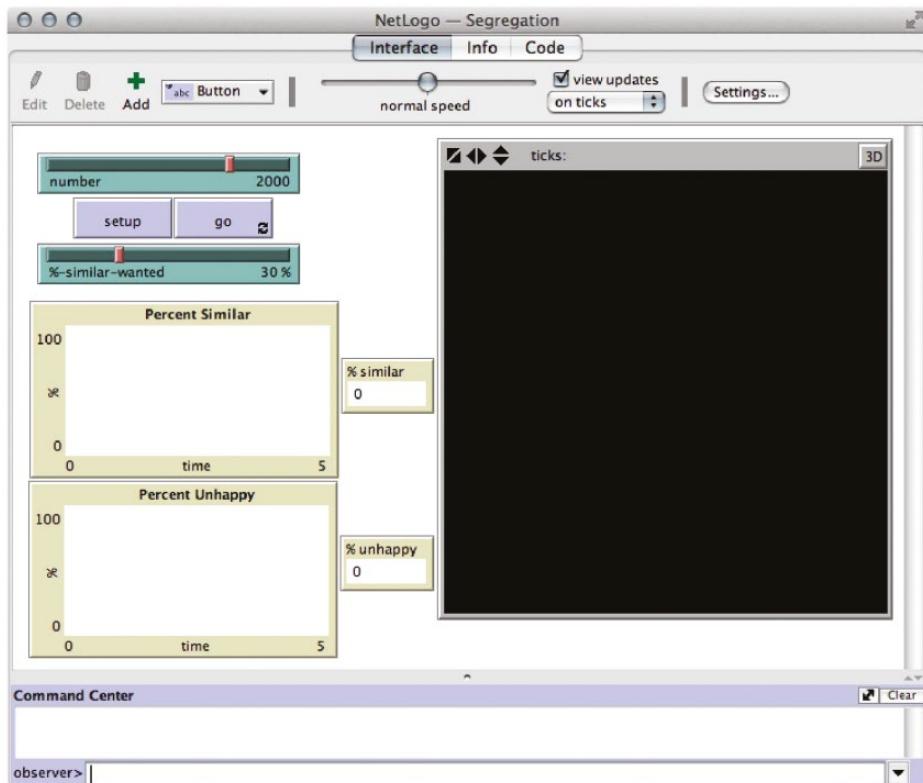


Figure 4.1 The NetLogo Segregation model (Wilensky, 2005a)

- Two sliders set the number of individuals in the model, and their preference for living next to someone of the same type (in this case ‘green’ or ‘red’).
- Two graphs show the average percentage of same-colour neighbours for each individual and the percentage of individuals that have fewer same-colour neighbours than they want (and thus want to move).

For more information about the model, or the function of the sliders and graphs, click on the **Info** tab.

Experiment with the model and consider the following questions:

1. Slowly increase the people’s preference for living next to the same type (%-similar-wanted). What happens to the spatial structure of the population as it increases?
2. What would you expect to see if %-similar-wanted was set to 100%? What actually happens?

3. How does the amount of time taken to reach equilibrium change as you increase and decrease the number of people in the model (e.g. the residential density)?

Finally, examine the code that underpins the model (click on the `Code` tab). NetLogo code has been designed to be easy to use; complex models can be created relatively easily – for example, the Schelling model has been written using only 79 lines.

Virus Model

Here we look at another model called `Virus`, which can be found under `Biology`. This model simulates the spread of a virus through a population as individuals come into contact with each other. Start by clicking on the `Info` tab and reading about how the model works. Experiment with the settings and answer the questions below:

1. What happens as the chance of recovering decreases? What happens when it gets to 0 (i.e. everyone who becomes infected dies)?
2. Does the same thing happen every time the model is run? If not, what might account for these differences?

4.2.1 The NetLogo Program

This section introduces the basic elements of the NetLogo program. It is important to understand how the program itself works. Figure 4.2 illustrates the most common components in the interface and Table 4.1 explains their purpose.

Now try experimenting with the different components:

1. Load the `Wolf Sheep Predation` model (Wilensky, 1997b) from the `models` library (under `Biology`).
2. Press the `setup` button to initialise the model. Sheep, wolves and green grass will appear.
3. Change the `model-version` to `sheep-wolves-grass`. This tells the model to simulate the sheep eating grass – this feature will be needed later.

The `show` command

The first command to try out is `show`. Type the following into the Command Center:

```
show "Hello World"
```

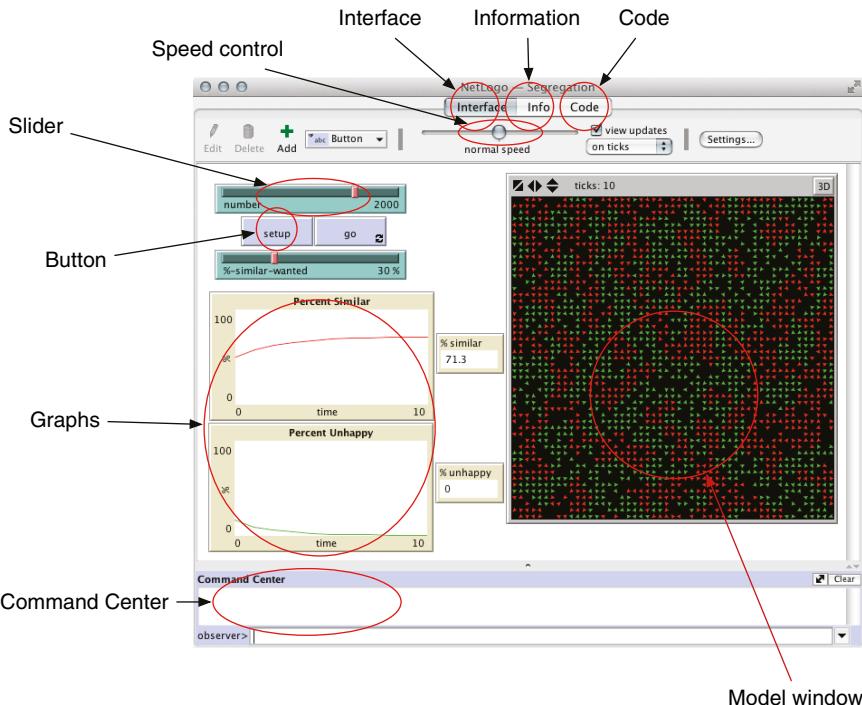


Figure 4.2 The main components of the NetLogo interface

Table 4.1 The main components of the NetLogo interface

Component	Description
Interface	This tab selects the main interface. This is where you can add elements (buttons, sliders, switches, etc.) that control the model and allow you to view the results.
Info	This tab presents information about the model you currently have loaded.
Code	This tab is where the computer code that controls the model is written.
Model window	This window shows the locations of the agents and condition of the patches (basically, what is happening as the model runs).
Command Center	The Command Center displays output messages. It can also be used to interact with the model via commands (commands are introduced shortly).
Buttons, graphs, etc.	These controls are used to interact with the model and understand what it is doing. They are added depending on which ones are needed for the particular application (not all models will need a graph of results, for example).

NetLogo will repeat "Hello World" and print it to screen:

```
observer> show "Hello World"
observer: "Hello World"
```

Try this again with some other text (don't forget to put quotes around the text). This might not seem particularly useful, but the `show` command is extremely useful when combined with other commands.

The count command

Now combine two commands, `show` and `count`, as follows:

```
show count turtles
```

What does this show? Try the following:

```
show count patches
```

What does that tell you?

Two things are happening when these commands are run. The `count` command is counting something, in this case the number of turtles or patches. Remember that agents/individuals are commonly referred to as *turtles*, and the cells that make up the environment as *patches*. The `show` command is behaving the same way as it was before, except now it is showing the number provided by the `count` command, rather than the "Hello World" text.

The with command

The `with` command can be used to influence a group of all the turtles or patches. Try the following:

```
show count patches with [ pcolor = green ]
show count patches with [ pcolor = brown ]
```

Run the model for a few seconds (press `go` to start it, then press it again to stop). Repeat the two commands above. What is the result? (Note that if none of the grass turned brown you need to change the model-version to sheep-wolves-grass.)

The `with` command actually needs two pieces of information. It needs a group of agents or patches on the left, and a test on the right (in this case the colour of the patches is tested).

Variables

In NetLogo, it is possible to save values in ‘variables’ (for more information, see the documentation on variables). Different parts of the model can check the values of the variables to decide what they should do. For example, in the `Wolf Sheep Predation` model, each of the sliders changes the value of a variable. If the `wolf-gain-from-food` slider is moved from 20 to 40, it will change the value of a variable called `wolf-gain-from-food`. Each time a wolf eats a sheep it checks the value of that variable to decide how much new ‘energy’ it will gain. In this case, it will now get 40 units rather than 20.

Individual turtles and patches can also have their own variables; this is one of the ways that models can account for heterogeneity. For example, in the `Wolf Sheep Predation` model, the wolves and sheep have a variable called `energy`. This records how much energy each individual wolf or sheep will have at any given time. If this reaches zero, then they will die. These variables will be examined in more detail in the next section.

The `set` command

The last command to be examined is `set`. This command can be used to manually change the values of variables without having to use a slider. To implement this, simply input the command stating which variable to change, and what to change it to.

Experiment with the following `set` commands. Set up the model, run the command, let the model run for 10–20 seconds, then try the commands below:

```
set sheep-gain-from-food 50
set initial-number-wolves 100
set wolf-reproduce 10
```

4.2.2 Contexts: Observer, Turtle, Patch (and Link)

In NetLogo there are four different *contexts* (although the *link* context is not covered explicitly in this introduction, we come back to it in Section 8.4) as shown in Figure 4.3. The observer oversees the NetLogo world and can give commands to either patches or turtles using commands such as `ask` (see Section 4.2.3). Essentially, the observer is the ‘god’ context that oversees the model. Turtles and patches each have different *variables* that store pieces of information about them. Figure 4.4 gives examples of some of the different variables that patches, turtles and the observer have access to (this example is from the `Wolf Sheep Predation` model, but the general idea is applicable to all models). Note that turtles have variables that store their (*x*, *y*) coordinates and colour (called `xcor`, `ycor`, `color`). To prevent programmers from becoming confused, the equivalent patch variables have a ‘*p*’ in front of them (`pxcor`, `pycor`, `pcolor`).

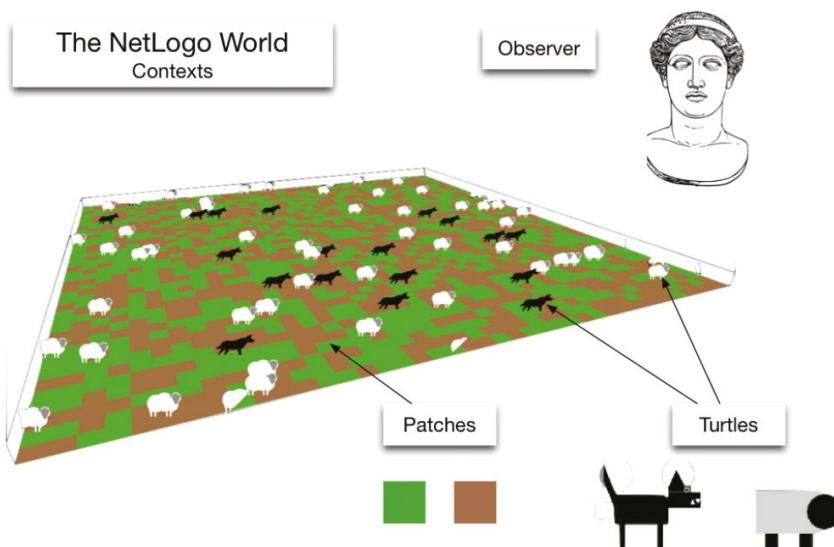


Figure 4.3 NetLogo contexts: the observer, patches and turtles

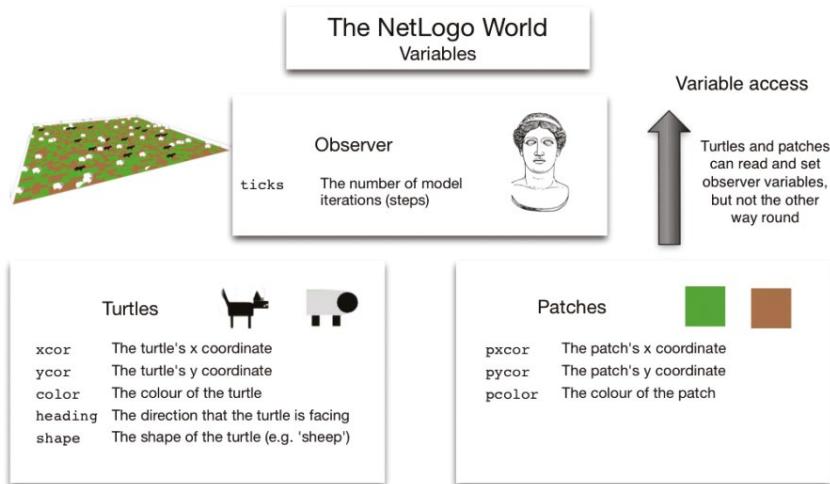


Figure 4.4 Examples of some of the variables that the NetLogo contexts (observer, patches and turtles) have access to

All the commands that have been used so far have been sent to the *observer* context. In order to give commands to turtles or patches, and to change the values of their variables, commands need to be sent to the *turtle* and *patch* contexts, respectively. This is done using the `ask` command.

4.2.3 The ask Command

In this section, an extremely useful command will be introduced: `ask`.

Reopen the Virus model from the Models Library (under Biology) and initialise it (either by using the `setup` button or by issuing the `setup` command in the Command Center). Now try entering the following command into the Command Center to change the value of the patches `pcolor` variable to blue:

```
set pcolor blue
```

What happens?

If everything had worked ‘correctly’ (or at least as it is supposed to) you should have seen the following error in the Command Center:

```
ERROR: You can't use PCOLOR in an observer context,
because PCOLOR is turtle/patch-only.
```

The message is saying that you cannot change the value of the `pcolor` variable, because that variable belongs to turtles and patches only. The observer has no variable called `pcolor`!

To get round this problem, the `ask` command can be used. Issue the same command but use `ask` to send it to the patches, rather than the observer:

```
ask patches [ set pcolor blue ]
```

Figure 4.5 illustrates what should happen.

Did the patches turn blue? There are a few things happening here:

- The `ask` command expects two inputs, both on the right-hand side of the command.
- The first input is the turtles or patches that we want to do something to. In this case, we ask to do something to all patches.
- The second input is a command, or a number of commands, that will be sent to those turtles or patches. So in the example above, the command `set pcolor blue` tells every patch to set their `pcolor` variable to the value `blue`. The square brackets allow us to send more than one command at a time to the turtles. For example, the following will make the turtles go blue, and also set a variable called `deprivation` to 11.

```
ask patches [ set pcolor blue set deprivation 11 ]
```

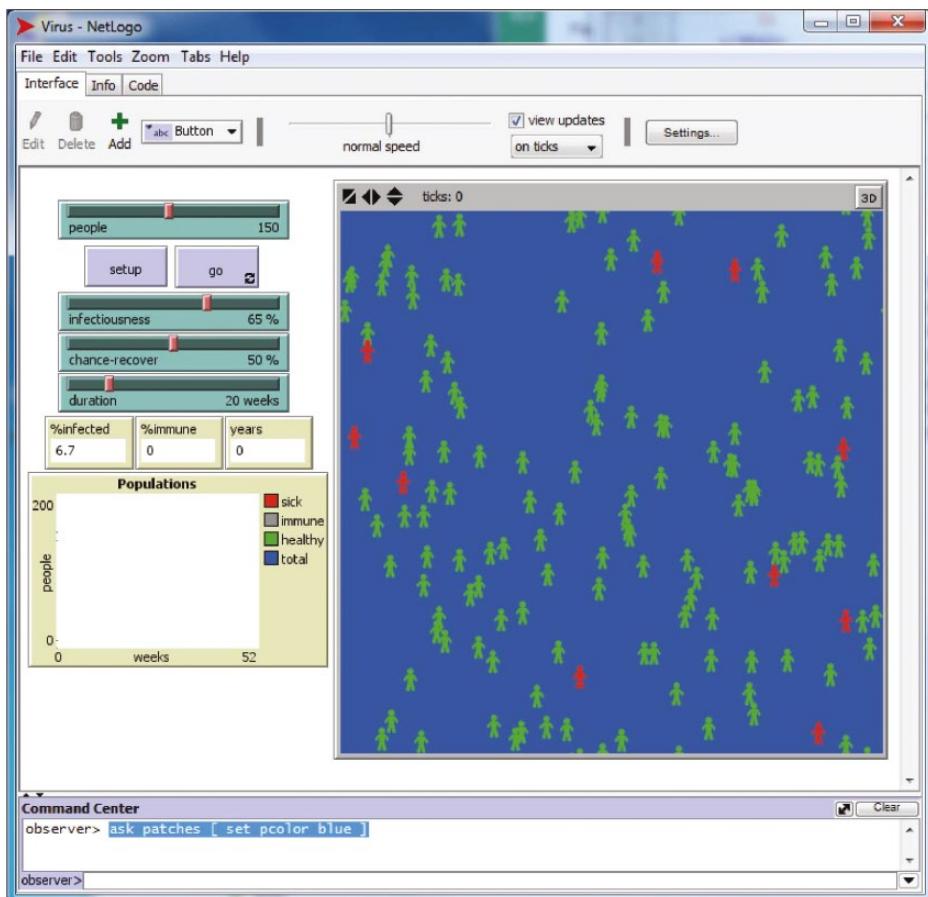


Figure 4.5 Using ask to make all patches go blue: `ask patches [set pcolor blue]`

Now try to do the same thing, but this time for the turtles. Enter the following command:

```
ask turtles [ set color orange ]
```

What happens? All the turtles should turn orange. This is because the value of their `color` variable which, like `pcolor` for patches, controls their colour has been changed.

Now try two commands to set the x and y coordinates of the turtles:

```
ask turtles [ set xcor 1 set ycor 5 ]
```

Where have all the turtles moved to? Try this as well:

```
ask turtles [ set xcor -10 set ycor -5 ]
```

The turtles will move again. Observe what is happening.

Finally, issue the following command to the turtles (note that this time `pcolor` is used):

```
ask turtles [ set pcolor black ]
```

What happens this time? Did some of the patches turn black? Which ones?

What is happening in this last example is extremely useful in models: turtles are given direct access to the variables of the patch that they are currently standing on.

Using ask and with

The previous examples of `ask` have been applied to *all* the turtles or patches in the model. However, most of the time it is more useful to execute a command on a smaller group. To do this, we can use the `with` command.

First reset the model, either by clicking on `setup` or by issuing the `setup` command. Then execute the following command, which will only affect individuals that have been infected by a virus:

```
ask turtles with [ sick? = true ] [ set color brown ]
```

Here, instead of sending the `ask` command to *all* of the turtles, it is sent to the group of turtles who have a value of `true` stored in their variable called `sick` (this is a special variable created specifically for the Virus model). Figure 4.6 illustrates visually how these two commands are constructed.

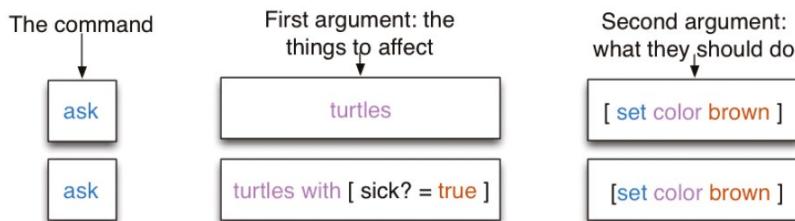


Figure 4.6 A visual representation of the use of `ask` and `with`

Now try it again, but change the colour of those who *are not* ill:

```
ask turtles with [ sick? = false ] [ set color blue ]
```

Each person also has a variable that remembers their age in weeks (this information is within the code). The `ask` command can also be used to run commands on people of different ages:

```
ask turtles with [ age > 1000 ] [ set color gray ]
```

Finally, a combination of `ask` and other commands will be used to do something more interesting than changing colours. Move the `people` slider from 150 down to 10. This will reduce the number of people in the model and make it easier to see the impact that the following commands will have. Set up the model. Only a few people are now visible in the world. Issue the following command:

```
ask turtles [ forward 1 ]
```

What is happening? What happens if a negative number is sent to the `forward` command?

Now try these commands and see what happens (they have to be entered into the Command Center one by one):

```
ask turtles [ facexy 0 0 ]
ask turtles [ forward 1 ]
ask turtles [ forward 1 ]
ask turtles [ forward 1 ]
```

The first command (`ask turtles [facexy 0 0]`) tells each turtle to spin round and face the coordinate (0, 0) (which happens to be in the middle of the world in this model). The commands that follow (`ask turtles [forward 1]`) tell the turtles to move forward one step in the direction that they are facing. This might seem trivial, but you have now covered the main commands that you need to create an agent-based model!

Information about all the different commands that are available can be found in the NetLogo documentation. In particular, the NetLogo Dictionary lists every available command.

4.3 First NetLogo Model

This section demonstrates how to build a new model from scratch. The content within this section has been directly adapted from the official NetLogo tutorial.² The model that will be created, entitled `first_model.nlogo`, is available in the online resources accompanying this book. It is very simple; it will contain some turtles (sheep) who will move around aimlessly eating grass. Figure 4.14

² <http://ccl.northwestern.edu/netlogo/4.1/docs/tutorial3.html>

visualises the complete model. Section 4.4 will then extend the model to add in more sophisticated functionality.

4.3.1 Creating the World

Before creating a model, the *environment* needs to be configured. This is achieved by specifying how large (how many patches) the ‘world’ needs to be and by specifying how the coordinate system works. The instructions below outline how to do this (refer also to Figure 4.7).

1. Open NetLogo and create a new model.
2. Right-click on the display (the part that would usually show the agents) and choose Edit.
3. Choose the location of the origin (0,0) to be Corner. Then also choose Bottom Left. This sets the origin to be at the bottom of the world in the left corner.

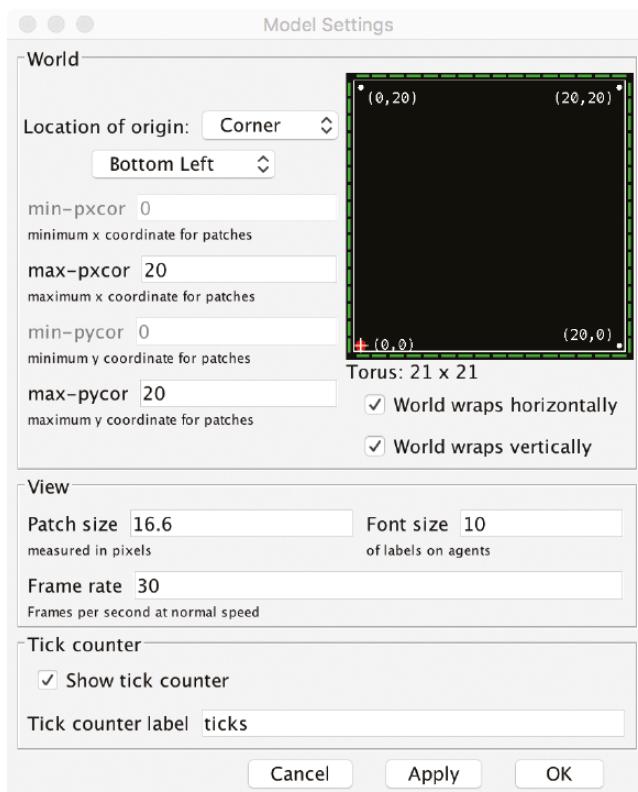


Figure 4.7 A picture of the environment configuration

4. Set the maximum x and y coordinates (`max-pxcor` and `max-pycor`) to 20. This will give a 20×20 cell grid.
5. Tick the boxes so that the world wraps both horizontally and vertically. This means that if a turtle leaves one side of the world it comes back in at the other side.
6. Leave the other variables as they are. Refer to Figure 4.7 to confirm that the environment has been configured correctly.
7. Click OK. Now the world is ready.

4.3.2 Buttons and Procedures

Creating a Button

In this section, we will demonstrate how to create a button that will be used to set up the model.

1. Click on the drop-down list next to Add and select Button (see Figure 4.8).
2. Move the mouse to where the button is going to be and left-click to place it.
3. In the new menu that appears (see Figure 4.9) you can specify what the button should do. Enter `setup` in the commands box (without apostrophes) – this will mean that a procedure called `setup` is activated when the button is clicked. This will be clarified shortly.

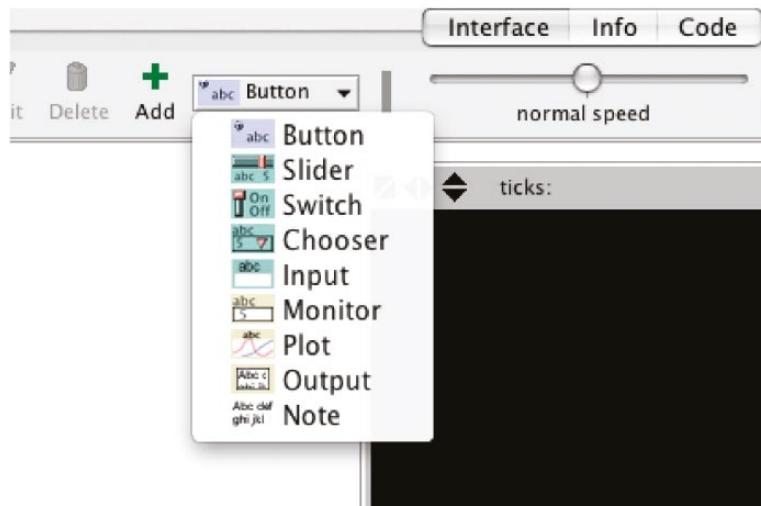


Figure 4.8 How to create a button

4. The ‘Display name’ is a label for the button. This will be displayed on the button and any suitable name can be used. In this case ‘Set up the model’ is used.
5. Click on OK to finish the button.

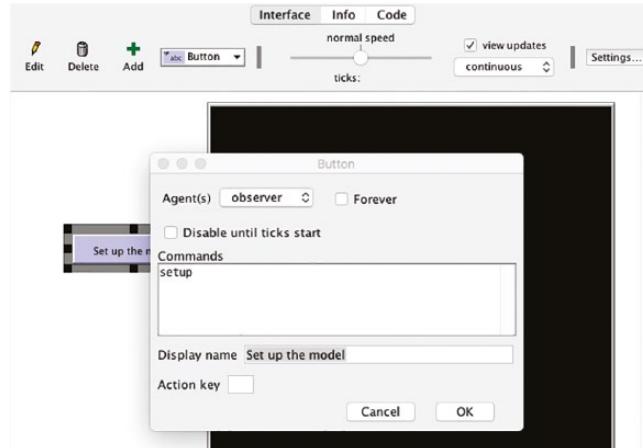


Figure 4.9 Creating a `setup` button

The text on the button is red, which means it refers to a procedure that has not been created yet. The following section will demonstrate how to create the NetLogo code that determines what the button will do.

Procedures

Procedures are a means of grouping several commands together to perform a common task. One example is setting up the model: within this there will be commands to create the turtles, patches and other parts of the model. These can be grouped together in a single procedure.

1. Click on the Code tab. This is where all the commands for the model are stored. A new procedure called `setup` will be created.
2. Enter the following text into the Code tab:

```
to setup
  print "Setting Up Model"
end
```

The code above does three things:

- (a) The first line (`to setup`) says whatever comes next will be part of a procedure called `setup`.
 - (b) The second line (`print "Setting Up Model"`) should be familiar – it will print a message to the Command Center.
 - (c) The third line (`end`) says that we have come to the end of the `setup` procedure. Any more commands that come afterwards are not part of `setup`.
3. Check that this works by clicking on the `Interface` tab. Has the button text gone from red to black?
 4. Click on the button. A message should be displayed in the Command Center.

In summary, the previous steps created a button which when clicked runs a procedure called `setup`. The code for the `setup` procedure instructs NetLogo simply to print the message “Setting Up Model”. Figure 4.10 below illustrates this graphically.

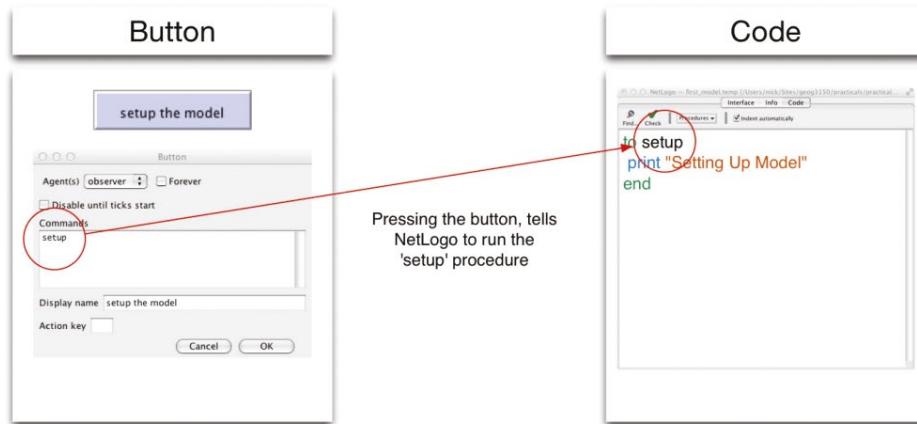


Figure 4.10 A NetLogo button and its associated code

4.3.3 Sliders and Variables

The previous sections have illustrated how buttons and procedures work together. This section will demonstrate how *sliders* connect up to *variables*.

1. Create a slider in the same way that a button was created. This slider is going to determine the number of turtles that will be initially created. Set the following values (as illustrated in Figure 4.11).
 - Global variable: initial-number-of-turtles. This is the name of the variable to be used in the Code tab.
 - Minimum: 0. This is the smallest number that the slider will show.
 - Increment: 1. Moving the slider very slightly will increase or decrease the number by 1.
 - Maximum: 100. This is the largest number that the slider can show.
 - Value: 10. The default value (the number that the slider starts with).
2. Press OK to create the slider. Note: to change its position, right-click on it and choose Select. It is then possible to move it.

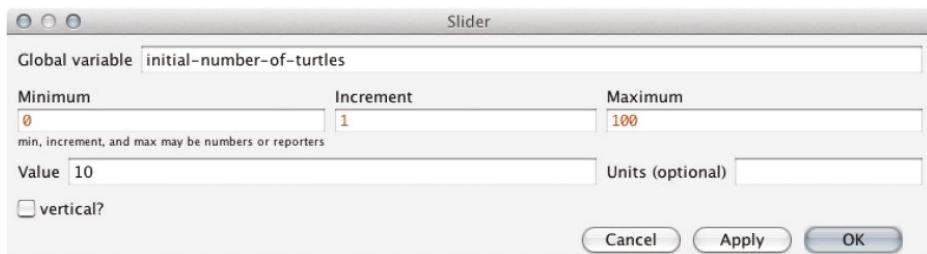


Figure 4.11 Configuring a NetLogo slider

The slider is now ready. However, as with the button, at the moment it does not have any effect on the model.

4.3.4 Creating Turtles and Patches

In this section, the fundamental components of a model will be created: the turtles (agents) and patches (environment).

1. Click on the Code tab. The setup procedure will now be extended so that it creates some turtles.

2. Change the `setup` function so that it reads:

```
to setup
  print "Setting Up Model"
  clear-all
  setup-patches
  setup-turtles
end
```

The `clear-all` command tells NetLogo to reset everything ready for a new model instance. The other two lines (`setup-patches` and `setup-turtles`) call two new procedures that will contain commands to set up the turtles and patches. These commands could be entered directly into the `setup` procedure, but by putting them into their own separate procedures it makes the model code easier to understand.

3. Here is the code for the two new procedures. Copy it directly after the `end` of the `setup` function.

```
to setup-patches
  ask patches [ set pcolor green ]
end

to setup-turtles
  create-turtles initial-number-of-turtles
  ask turtles [
    set shape "sheep"
    setxy (random 20) (random 20)
    set color blue
  ]
end
```

The code above should be fairly self-explanatory. The first procedure (`setup-patches`) asks all of the patches to change their colour to green. This means that each patch will represent a square of grass, ready to be eaten by the turtles. The second procedure (`setup-turtles`) creates the turtles.

There are, however, some new commands that require explanation:

- `create-turtles` makes a number of new turtles. The value from the slider (`initial-number-of-turtles`) is used to determine the number that will be created. If this number was the same for every model instance, then a number could be used instead of a variable (e.g. `create-turtles 10` would always create 10 turtles).

- Next, `ask turtles` is used to run some commands on the turtles that have just been created. Notice that the commands are enclosed in square brackets ([and]).
 - `set shape "sheep"` uses the `set` command to change the shape of the turtle. This is the same as changing other turtles' variables such as their `color`. Refer to the NetLogo Shapes Editor Guide³ to see a list of all the shapes that are available.
 - `setxy (random 20) (random 20)` is a quick way of setting the turtles' *x* and *y* coordinates. `random 20` means choose a random number between 0 and 20. This is exactly the same as using two commands, one after the other: `set xcor (random 20)` `set ycor (random 20)`.
 - Finally, `set color blue` makes the sheep blue.
4. Return to the Interface tab. If there are any mistakes with the model code, NetLogo will present an error.
 5. Press the `setup` button. The turtles should appear randomly in the grid.
 6. Press the `setup` button again. What happens?

Figure 4.12 illustrates how the model will look after it has been initialised with sheep and grass.

4.3.5 Making the Model go

To finish the model, a `go` button is needed. This will start the model simulation.

1. Create a new button.
2. In the Commands section, write `go`. This will make the button start a procedure called `go`. The button can be given any name, but 'go' is typically used in NetLogo models.
3. Important: tick the `Forever` box. This will mean that the procedure is called over and over again until the button is pressed again. If this box were not ticked, then the button would make the model run for one only iteration (time step) and then stop.
4. Press `OK` to create the button.

³ <http://ccl.northwestern.edu/netlogo/docs/shapes.html>

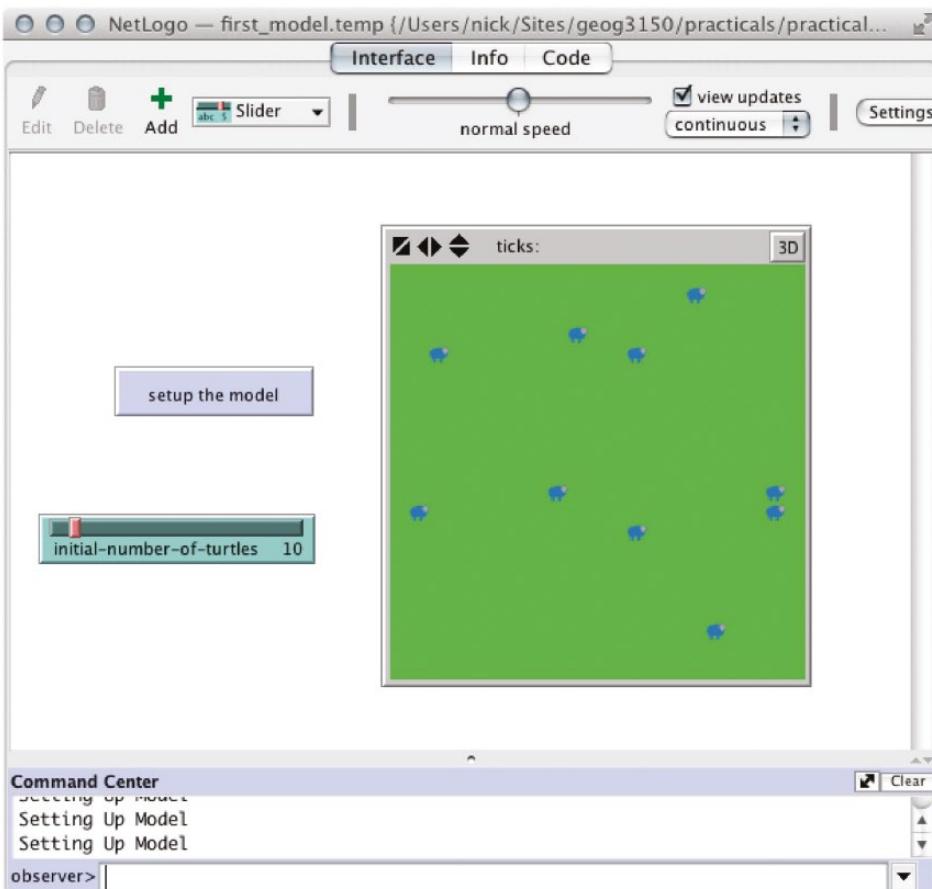


Figure 4.12 An example of the model interface once the sheep and grass have been created

5. Go back to the Code tab and add the following procedure to the code. Add it after the procedures that set up the model.

```
to go
  ask turtles [
    rt (random 360)
    fd 1
    if pcolor = green [
      set pcolor brown
    ]
  ]
end
```

There are some new commands in here, but they are easy to explain:

- The first thing the `go` procedure does is run `ask turtles`. This means that all the commands in between the square brackets will be sent to the turtles.
- `rt (random 360)` tells the turtles to rotate a random amount from 0 to 360 degrees.
- `fd 1` means ‘move forward one step’ in whatever direction they are facing.
- The next part is slightly more complicated. Remember that turtles have access to the variables of the patch that they are standing on. So, the commands

```
if pcolor = green [
  set pcolor brown
]
```

mean ‘if the turtle is standing on a green patch, then make the colour of the patch brown’. Notice the use of square brackets to make it clear which commands should happen if the colour of the patch is green. This basically makes it look like the turtle (or sheep, if you prefer) has just eaten all the grass on that square.

Finally, return to the main `Interface` tab and click on the `go` button. The model probably runs too quickly to see what is happening and the whole world suddenly turns brown. To stop this, move the slider on the `Interface` tab (see Figure 4.13) to the left. Then reset the model and run it again. Now the sheep should move at a more reasonable pace.



Figure 4.13 The speed slider can be used to slow down or speed up the rate that the model iterates

That’s it! This is now a very simple simulation with some turtles that will move around and eat grass (see Figure 4.14). To see the code in full, refer to the `first model.nlogo` model in the accompanying resources. With a few small additions to the code it is possible to have grass that grows and can be eaten, turtles that die

if they do not have enough to eat, and a graph to show how many turtles are alive. These are the subjects of Section 4.4.

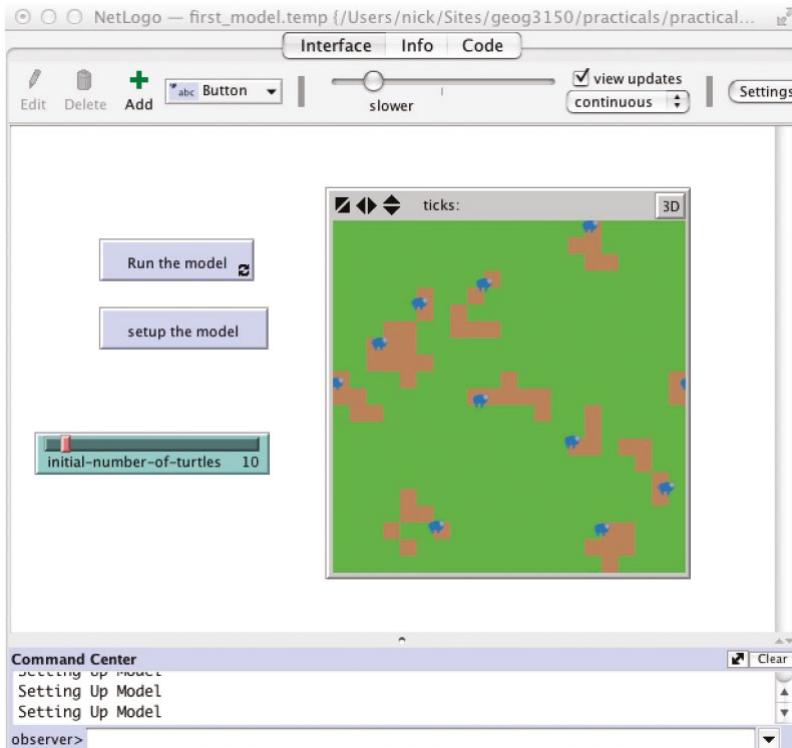


Figure 4.14 The final version of the simple model; sheep move around and turn green patches to brown

4.4 Advanced NetLogo Model

The previous section outlined how to create a full (albeit simple) NetLogo model of some sheep eating grass. In this section, the realism of the model will be improved with some new functionality:

- Tracking the time since a sheep's last meal and killing sheep who starve.
- Healthy sheep can give birth to new sheep.
- Having grass take time to regrow after being eaten.
- A graph of the number of living sheep.

Figure 4.15 illustrates what the final model will look like. Before getting started, make a copy of the NetLogo model that was created in Section 4.3. One way to do this is to open the model, choose *File* → *Save As*, and save it under a new name. This section will involve making modifications to that model, so a copy makes it possible to refer back to the more simple model.

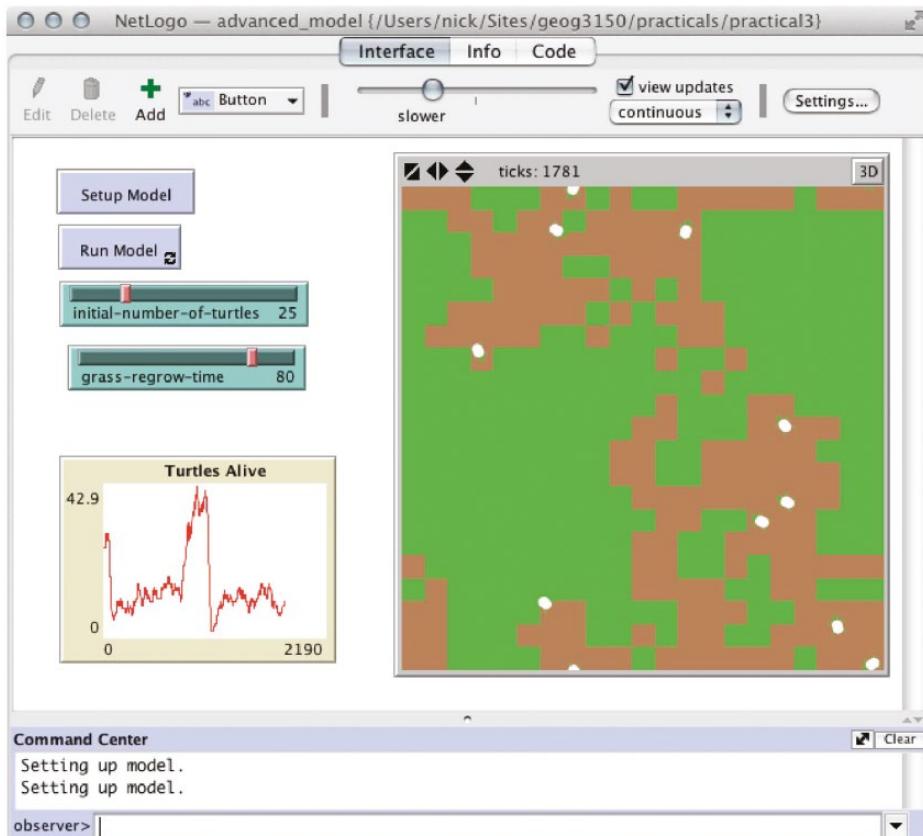


Figure 4.15 An illustration of the final model that will be completed in Section 4.4

4.4.1 Advanced Variables

Recall from Section 4.2.1 that *variables* can be used to hold information about different objects in NetLogo. For example, turtles might have a variable called *age* that saves the age of each turtle. Similarly, patches might have a variable called *grain* that stores (for example) the amount of grain growing on each

patch. Figure 4.4 illustrated some of the variables that are built in for the observer, turtles and patches. Some of these have already been used in previous examples, such as `color`, `xcor` and `ycor` (for turtles), and `pcolor` (for patches). There is a full list of built-in NetLogo variables available in the NetLogo Dictionary.

However, to build more complex models new variables need to be created to store different pieces of information that might be required for the system under investigation. This section will look at how to do this.

1. Open the basic model produced in Section 4.3.

First, two new variables need to be created. One, called `energy`, will store the amount of life energy each turtle has. If this reaches 0 then they die. The second variable, `time-since-eaten`, will store the amount of time that has passed since a patch of grass has been eaten. Grass takes time to regrow, and this variable will tell us when a patch has completely regrown. The following instructions will outline the code for the `energy` variable. Code Example 4.4.1 shows what the final code block should look like.

2. To create these new variables, add the following code at the top of the model (before the `to setup` line).

```
turtles-own [
  energy
]

patches-own [
  time-since-eaten
]
```

In the code above, `turtles-own` tells NetLogo that some new variables are going to be defined that will belong to the turtles. Variables can then be created in between the square brackets (one per line). A single variable, called `energy` is created. Similarly, the `patches-own` code means that variables are being defined for the patches.

3. Add the following line to the `setup-turtles` procedure, just after the `ask turtles [` line:

```
set energy 5
```

This will give each turtle 5 units of energy at the beginning of the model.

4. Also add the following lines to the go procedure, just after the ask turtles [line:

```
set energy (energy - 1)
if energy <= 0 [
  die
]
```

(Code Example 4.4.1 illustrates all of the code required, this can be referred to if the previous instructions are unclear.)

The first line reduces the value of the `energy` variable by one unit. Each time the model iterates, every turtle will lose one unit of energy. The next line (`if energy <= 0 []`) checks to see if the turtle has lost all its energy. If it has, then the `die` command removes the turtle from the model.

Now the turtles have energy when they are created, however, each time the model iterates they lose some. To conclude this part of the model development, code will be added to make the turtles eat grass and gain energy.

5. In the go procedure, find the code that makes the turtles eat the grass. At the moment it reads:

```
if pcolor = green [
  set pcolor brown
]
```

But all that code does is change the colour of the patch that the turtle is standing on. Therefore, add two new lines so that the code now reads:

```
if pcolor = green [
  set pcolor brown
  set energy (energy + 5 )
  set time-since-eaten 0
]
```

The new code still changes the colour of the patch from green to brown to indicate that it has been eaten. However, now it also gives the turtle five units of energy and sets the `time-since-eaten` variable to 0 to indicate that no time has elapsed since the patch was eaten. (This will be important later when the grass grows back slowly over time.)

The `time-since-eaten` variable belongs to the patches, so it should not be possible to change it from within the turtle context (i.e. in an `ask turtles [...]` block). NetLogo makes it possible to access patch variables from

the turtle context for the patch that a turtle is currently standing on. This is an extremely useful and time-saving feature.

- Finally, check that everything works by clicking on the `Interface` tab and running the model. If it runs too quickly to see what is happening use the control at the top to slow it down. The sheep should die eventually. This is because they have the ability to eat grass, but the grass isn't able to grow back! This will be the subject of the next section.

Code Example 4.4.1 illustrates what the final `go` procedure should look like. Note that in NetLogo, anything that appears after a `;;` symbol is called a *comment*. These are not read by NetLogo – they can be used by the developer to explain what it is that their code is doing. Always include comments in code! These not only make it easier for others to understand the code, but also help the original programmer to remember what they have done (and why).

4.4.2 Grass Grows ...

This section is very short – all that is required is to add a few lines so that the grass in the model regrows after a certain amount of time. Importantly, a slider will be created so that the person running the model can experiment with different regrowth times.

Code Example 4.4.1 Allowing sheep to eat grass and get some energy

```

to go
  ask turtles[
    set energy (energy - 1) ; Take away some energy
    ;; If the turtle has no energy, then die
    if energy <= 0 [
      die
    ]
    ;; Move one step in a random direction
    rt random 360
    fd 1
    ;; See if the current patch is good to eat
    if pcolor = green [
      set pcolor brown
      set energy (energy + 5)
      set time-since-eaten 0
    ]
  ]
end

```

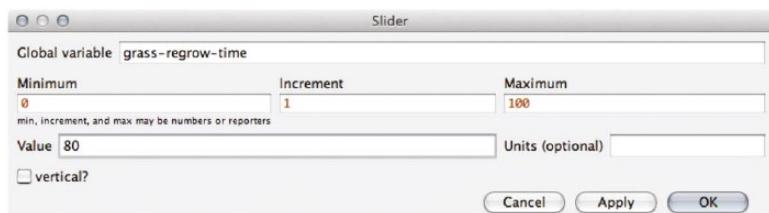


Figure 4.16 Creating a slider to set the grass regrowth rate

1. To begin with, create a slider called `grass-regrow-time` as shown in Figure 4.16. Give the slider a range of 0 to 100 with any default value (here 80 is chosen).

In NetLogo, sliders represent *variables*. Therefore, creating a slider called `grass-regrow-time` will make NetLogo create a variable with the same name that can be referred to in the model code. The variable will be part of the observer context, all turtles and patches can see what value it holds. This variable will be used to determine how long it takes for grass to regrow once it has been eaten.

2. Add the following code to the `go` procedure, once the `ask turtles` block has finished (after the closing ']'):

```
ask patches [
  if pcolor = brown [
    set time-since-eaten time-since-eaten + 1
    if time-since-eaten > grass-regrow-time [
      set pcolor green
    ]
  ]
]
```

The overall job of the code above is to see if enough time has passed for brown patches to become green again. Table 4.2 explains what each line does.

Understanding the logical flow of the program as it compares variables and decides what to do is a common task in computer programming and a very useful skill in general.

3. Finally, go back to the `Interface` tab and try to run the model. The grass should turn brown and later regrow as the turtles move around the environment.

Table 4.2 An explanation of the code to make the grass regrow. Code Example 4.4.2 illustrates the full code

Code	Explanation
ask patches [This tells NetLogo to run through the patches one by one to see if they are ready to regrow.
if pc当地 = brown [This checks the colour of an individual patch. If the patch is brown then it might be time to regrow. If it is green, then it is ignored.
set time-since-eaten time-since-eaten + 1	This increases the value of the time-since-eaten variable by one unit. This is how the amount of time that has passed since the patch became brown (i.e. the grass was eaten) is tracked.
if time-since-eaten > grass-regrow-time [Here, use of the grass-regrow-time slider. If enough time has passed since the patch became brown, then something can be done to it.
set pc当地 green	This simply changes the colour of the patch from brown to green. Importantly, the line above makes sure that this will only happen if enough time has passed. If enough time <i>has not</i> passed, then the patch will stay brown (for now).
]	This finishes the if time-since-eaten > grass-regrow-time [block.
]	This finishes the if pc当地 = brown [block.
]	This finishes the ask patches [block.

4.4.3 Giving Birth

In this final section a little more code will be added to allow the turtles to give birth if they have enough energy.

- Add the following code to the go procedure, after the turtle has decided whether or not a patch of grass is good to eat. Code Example 4.4.2 illustrates the full code.

```
if energy > 50 [
  set energy 10
  hatch 1
]
```

This checks to see if the turtle has more than 50 units of energy stored up. If it does, the amount of energy reduces to 10 (to simulate the stress of giving birth). The hatch command then creates a copy of the turtle.

- Return to the Interface tab and run the model. If the model is running sufficiently slowly, it might be possible to see new turtles being born. Try experimenting with the grass-regrow-time slider. If the amount of time it takes for grass to regrow changes, does this have an impact on the number of turtles in the model?

Code Example 4.4.2 The `go` procedure in full, showing how the turtles eat grass and give birth, and how the grass regrows over time:

```

to go
  ask turtles[
    ;; Take away some energy
    set energy (energy - 1)

    ;; If the turtle has no energy, then die
    if energy <= 0 [die]

    ;; Move one step in a random direction
    rt random 360
    fd 1

    ;; See if the current patch is good to eat
    if pcolor = green [
      set pcolor brown
      set energy (energy + 5)
      set time-since-eaten 0
    ]

    ;; If this turtle is healthy enough, it can give birth
    if energy > 50 [
      set energy 10 ;; Giving birth takes a lot of energy
      hatch 1
    ]
  ]

  ;; See if any of the patches have re-grown
  ask patches[
    if pcolor = brown [
      ;; Increment the time since the patch was eaten
      set time-since-eaten time-since-eaten + 1
      ;; If enough time has passed, make the grass green again
      if time-since-eaten > grass-regrow-time [
        set pcolor green
      ]
    ]
  ]
  tick
end

```

4.4.4 Creating a Graph

It is quite hard to work out how many turtles are actually in the model at any given time. It would be better if there was a way to see *exactly* how many turtles are alive at any given time. This is something that will be addressed in this section. Figure 4.17 illustrates the type of graph that will be created. Creating graphs is explained in detail in the official NetLogo tutorial and this section draws heavily on that tutorial.

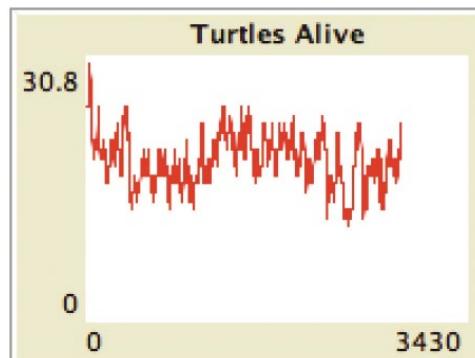


Figure 4.17 The graph that shows the number of agents in the model (y-axis) at each iteration (x-axis)

1. Open the model, go to the Interface tab, and create a new Plot in the same way a button or slider would be created.
2. Enter the following values to configure how the axes are displayed:

Name:	Total Number of Alive Turtles
X axis label	time
Y axis label	count
Auto scale?	tick (this makes the axes increase over time)

Next, NetLogo needs to know how to draw the graph. To do this, it uses something called ‘plot pens’. It is possible to have multiple pens per graph to draw a number of different things simultaneously. Within this example, a single ‘pen’ that draws the number of turtles who are alive at a given time will be created.

3. You should see a default pen in the Plot pens box already. If not, you can just click on Add pen.

4. Give the pen the name `turtles-alive` and choose a colour for the pen by clicking on the cell in the `Color` column (this will determine the colour of the line on the chart).
5. The `Plot update commands` section determines what the chart will plot. In there, type `plot count turtles`. This will plot the number of turtles currently in the model.

Figure 4.18 illustrates the configuration for the plot.

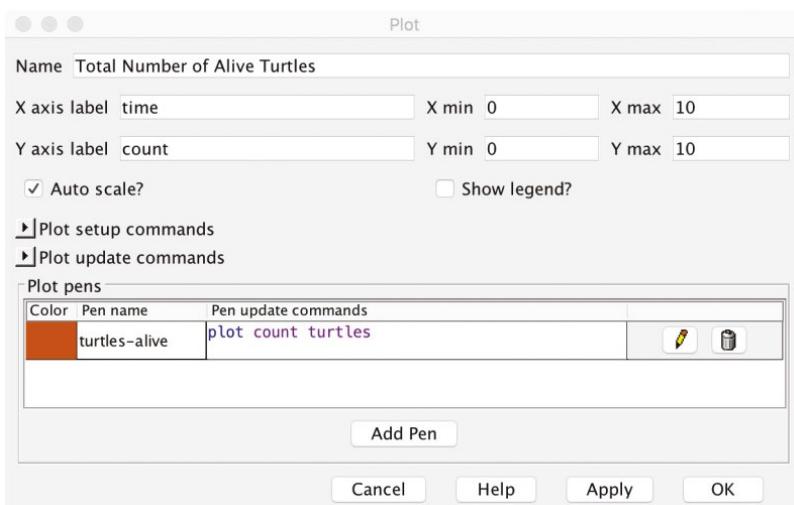


Figure 4.18 Configuring a plot that will graph the number of alive turtles at each iteration

6. Now the graph is ready. Click on `OK` and an empty graph should appear. There are two small changes that need to be made to the code for the graph to work.
7. Go into the `Code` tab.
8. In the `setup` procedure, add the command `reset-ticks` at the end:

```
to setup
  clear-all
  print "Setting up model."
  setup-patches
  setup-turtles
  reset-ticks
end
```

These commands, which are necessary for the plots to work, remove everything from the model and reset the ‘tick counter’ to zero. This will be run each time the `setup` button is clicked. If this step is not taken, the graph never goes back to zero, even after the model has been reset.

9. Find the `go` procedure. At the very end, just before the `end` command (which tells NetLogo that the `go` procedure has finished) add the single command `tick`. This makes NetLogo increment its internal clock (the number of iterations), something the plot needs to know.

That’s it! Now go back to the `Interface` tab and try to run the model. Does the graph work? Experiment with the `grass-regrow-time` slider again. Does the graph make it easier to see what effect the slider has on the size of the turtle population?

Chapter Summary

NetLogo is becoming one of the most popular tools for building and testing agent-based models. This chapter has presented an overview of NetLogo, introducing the environment and key concepts as well as demonstrating its main commands and functionality. A series of examples have given a practical guide to how to build a simple agent-based model that can be further developed for other applications. The strength of NetLogo is the ease with which relatively complicated agents can be created and released into an ‘environment’. The following chapters will explore this link with space, specifically looking at how agent-based models and geographical information systems can be brought together for their mutual benefit.

4.5 Annotated Bibliography

There are a number of excellent textbooks that focusses solely on agent-based modelling with NetLogo:

- Wilensky, U. and Rand, W. (2015) *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. Cambridge, MA: MIT Press.
- Railsback, S.F. and Grimm, V. (2011) *Agent-Based and Individual-Based Modeling: A Practical Introduction*. Princeton, NJ: Princeton University Press.
- O’Sullivan, D. and Perry, G.L. (2013). *Spatial Simulation: Exploring Pattern and Process*. Chichester: John Wiley & Sons.

Web resources:

- <https://ccl.northwestern.edu/netlogo/docs/> The NetLogo documentation website, which is extremely comprehensive.
- www.intro-to-abm.com/ The website containing models and other material from Wilensky and Rand (2015).
- www.railsback-grimmm-abm-book.com/ The website containing models and other material from Railsback and Grimm (2011).
- <http://patternandprocess.org/> The website containing models and other material from O'Sullivan and Perry (2013).
- www.abmgis.org/Chapter4 Here you will find all the models from this chapter as well as additional resources.

5

FUNDAMENTALS OF GEOGRAPHICAL INFORMATION SYSTEMS

Chapter Outline

This chapter presents the main concepts and terminology that students require to understand geographical information systems. The main data types are presented, along with a discussion of relevant issues such as accuracy and precision. A brief overview of the development of GIS is given along with a flavour of the main software available. Using QGIS, we demonstrate how to prepare and manipulate some example GIS data. Where appropriate, we highlight the main issues that need to be considered when using a GIS and agent-based modelling.

5.1 Introduction

Whatever occurs, occurs in space and time. Therefore our perception of the world is inherently spatial and temporal: objects have a location, and events are embedded in a stream of time. (*Wegener, 2000*)

Longley et al. (2010) note that a geographical information system ‘is concerned with the description, explanation and prediction of patterns and processes at geographic scales’ and further remark that a geographical information system is a ‘computer-based system for storing and processing geographic information’. GIS are used in all aspects of our daily lives, including providing inventories on land use, identifying flood plains, deciding to where to place retail stores as well as in-vehicle navigation. For example, after Hurricane Sandy in 2012, the US Federal Emergency Management Agency (FEMA) used GIS data to identify 44,000 households that were impacted by the hurricane. This allowed FEMA to

provide faster assistance to affected communities (Government Accountability Office, 2015). GIS have been proven to play an important role in decision-making; for example, Birkin et al. (1996) discuss how GIS, spatial analysis and modelling provide planning and insights for public and private sectors including retail, health-care and the automobile industries. Take, for example, store location; if we were to site a new store we would gather information about the target audience of the store (i.e. the existing customer base and the customer base in the new area, say, via geodemographic profiling), how accessible the store is (i.e. via road networks) and the potential competition from other stores. All this information can be collected and analysed in a GIS.¹

In simple terms, GIS allow us to store, organise, access and retrieve geographical information. How we use this information depends on the requirements of the research being undertaken. In essence, data is just text and numbers, but ‘spatial is special’. Almost all human activities and decisions involve a geographic component (e.g. where should we live?, How will I get to work?, What are the current traffic conditions?, etc.). In this chapter we attempt to provide an overview of GIS, and specifically their fundamentals, to provide readers with an understanding of how such data can be used in agent-based models and, moreover, which analysis techniques can be used to explore the outputs of such models.

To this end, in Section 5.2 we provide a brief history of GIS, before discussing how one can represent the world (Section 5.3) and time (Section 5.4) in a GIS. Next the chapter moves onto discussing GIS software (Section 5.5) and sources of geographical data (Section 5.6). Then the chapter demonstrates through the use of QGIS how one can manipulate geographical information (Section 5.7), along with considerations for the visualisation of results involving geographical data (Section 5.8). The chapter concludes with a discussion (Section 5.9), and an annotated bibliography provides recommendations for further reading.

5.2 A (Very Brief) History of GIS

Early attempts at digital mapping can be traced back to Harvard’s Computer Graphics Lab SYMAP (symbol or synagraphic mapping) project, a computer

¹ It should be noted that the term ‘GIS’ means different things to different people, ranging from digital maps to tools for scientific discovery and spatial decision support. Traditionally the S in GIS referred to systems (sometimes abbreviated as GISystems), including hardware, software and data. However, the S can also relate to science (GIScience; see Goodchild, 1992), referring to the study of geographic information (e.g. spatial analysis). As the focus in this book is more on systems than on science, we refer interested readers to Longley et al. (2010) for a more in-depth discussion.

program for displaying (plotting) geographical information on a line printer in the late 1960s.² However, the first GIS was the Canada Geographic Information System established in the mid-1960s, which provided a computerised mapping system for land-use management. This was followed in the late 1960s by the DIME (Dual Independent Map Encoding) program developed by the US Bureau of the Census. This program digitised all US streets in order to facilitate the geo-referencing and aggregation of US census records.

These early initiatives spurred the development of GIS software, with the Harvard Laboratory Computer Graphics vector ODYSSEY GIS developed in the mid-1970s, followed by ESRI's ARC/INFO in 1982 (ArcGIS). In the 1980s and 1990s computers became more widespread, cheaper and more powerful; this resulted in a rapid proliferation of software and data availability. These conditions were perfect for the uptake and establishment of GIS as a tool for research and commercialisation applications (Longley et al., 2010).³ It is probably no coincidence that this growth also coincided with a large growth of data as remote sensing applications, such as the Landsat program, and the proliferation of global positioning systems (GPS), provided more and more data about the world around us.

From its earliest days GIS have been used to facilitate decision-making, such as allocating resources based on census information, finding the most optimal routes for delivery trucks, for customer segmentation and profiling (e.g. geodemographics) and as a spatial planning support tool (see Brail and Klosterman, 2001). Their potential for modelling has also not gone unnoticed: GIS lie at the heart of much geographical modelling, especially with the growth in spatial data, and the rise in computing power and graphical user interfaces (we will come back to the integration of GIS and models in Chapter 6).

However, while there is an abundance of commercial applications for GIS, we have also witnessed an explosion of open source GIS such as GRASS (Geographic Resource Analysis Support System) developed by the US Army Corps of Engineers (Westervelt, 2004) and QGIS (Quantum GIS; see Section 5.5), along with the growth of crowdsourced platforms such as OpenStreetMap (see Section 5.6.1).

Geographical information systems have grown into a billion-dollar industry, but as 'GIS' means different things to different people (e.g. data, software, services, etc.) it is difficult to quantify the exact spend each year. It has been reported that

² Readers wishing to find out more about the early days of digital mapping are referred to Chrisman (2006).

³ Readers wishing to know more about the history of GIS are referred to the short article entitled 'The Remarkable History of GIS' available at <http://gisgeography.com/history-of-gis/>, or the work of Longley et al. (2010).

the US federal government spends billions of dollars annually on geospatial data and services (Government Accountability Office, 2015), with an estimated spend on geospatial software of around \$2.64 billion in 2012⁴ and \$10.6 billion in 2015 (Dempsey, 2012). The commercial value of GIS as a business can be evidenced through the \$1.1 billion global sales in 2014 of ESRI, one the largest GIS software developers (Helft, 2016). With the importance of geography, these trends are expected to grow in the future and become much more integrated into our daily lives. As of 2018 it was estimated that the economic impact of the geospatial industry is \$2210 billion (Narain, 2018).

5.3 Representing the World

One of the central challenges with using a GIS centres around the question of how we represent the complexities of the world in digital form – specifically, the geographic data that links place, time and attributes. At its most basic level, a GIS is simply a model, therefore what we input into it will be an abstraction (i.e. a simplified version) of reality. The most basic way of doing this is to deconstruct the world into two views: *fields* and *objects*.

Field-based views of space represent geographic phenomena that have a value (distribution) everywhere within geographic space. This can be further broken down into continuous (e.g. temperature, elevation, slope) or discrete (e.g. each cell as a specific class such as a specific land-cover class or soil type) representations of the world. An *object-based* view of geographic space is one where space is filled with discrete identifiable units (e.g. houses, lampposts) with well-defined boundaries (e.g. land ownerships). However, not everything fits neatly into these two representations of space; for example, soil types or mountain ranges are neither fields nor objects due to the fuzzy nature of their boundaries – where does a mountain range begin or end?

While fields and objects provide us with a conceptual view of geographic phenomena, they do not help with digitally representing it. For this we need to use data structures. Simply stated, a data structure/model refers to how geographic reality is abstracted or simplified (i.e. modelled) in a GIS. There are several types of geographic data models based on computer-aided design (CAD), graphical models, image models, raster/grid models, vector/georelational topological models, network models, triangulated irregular network models and objects (see Longley et al., 2010). In order to reduce the complexities of geographic phenomena for use by a computer we need to represent these structures in digital space. The two most common ways of doing this are by using the *raster*

⁴ <https://www.directionsmag.com/article/1436>

and *vector* data structures. The vector data structure (Section 5.3.2) is a computer implementation of an object-based (geometric) view of the world, while a raster data structure (Section 5.3.1) is based on the field-based (cell) view of the world in two dimensions (see Couclelis, 1992, for discussion of raster and vector data structures). In Figure 5.1 we show how vector data can capture geometric detail, while the rasterised space only provides information if the cell is inside or outside the area. Of course, raster data can be converted into vector data (vectorisation) or vice versa (rasterisation) if needed.

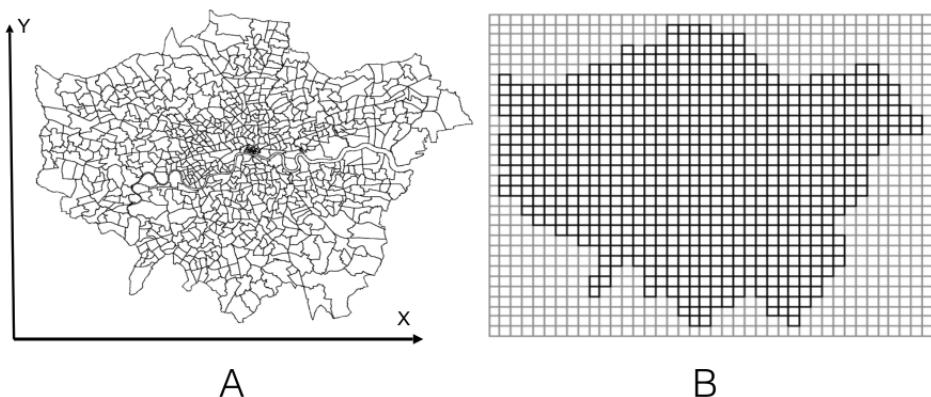


Figure 5.1 An example of (A) a vector and (B) a raster representation of wards (administrative units) in the Greater London Authority area

5.3.1 Raster Data

The raster data model uses an array of cells (typically square) to represent real world objects. All geographic variation is then expressed by assigning properties or attributes to these cells. The cells can hold any attribute values based on one of several encoding schemes, including categories, integers and floating-point numbers. In the simplest case, a binary representation is used (e.g. presence and absence of vegetation) but in other cases floating-point values are preferred (e.g. height above sea level in metres). Usually raster data is stored as an array of grid values, with metadata about the array held in the file header. A simple example of this is shown in Figure 5.2; the full model is available in the accompanying website (see the ImportRasterSample model; additional examples of the use of raster data for models will be demonstrated in Section 6.4.1). Typical metadata includes the geographical coordinates of the upper left-hand corner of the grid, the cell size, the number of row and column elements, and the projection.

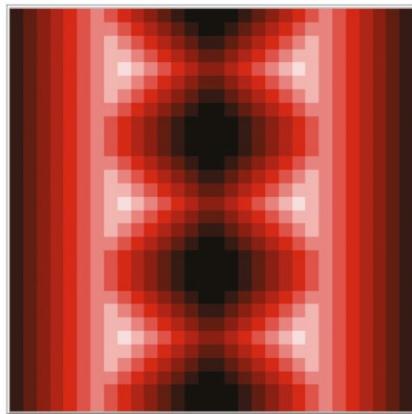


Figure 5.2 An example of reading in raster data and creating a landscape: (A) the original ASCII file from a GIS; (B) the resulting space created in NetLogo

One of the most abundant sources of raster data is remote sensing satellites. For example, Landsat (2018) data has a spatial resolution of 30 metres and is commonly used for applications ranging from land-use mapping to environmental monitoring. Several models have also used Landsat data either for initialisation of land-use/cover conditions or for validating land-use change patterns (e.g. Mena et al. 2011; Xie et al. 2007). This abundance of fine-resolution raster data has resulted in the widespread development of techniques (e.g. map algebra; Tomlin, 1994) and tools (e.g. Raster Calculator in QGIS) for the analysis of this data source.

One of the disadvantages of raster data is that because a cell is given a single value, any other variation that occurs within that cell is lost.⁵ Complications arise if two features (e.g. a lake and a forest) occur in the same grid cell but have different attributes. In these cases, a rule will be required to decide whether the cell is to be characterised as a lake or a forest.

⁵ However, a caveat is needed here. Some raster data models allow for multiple attributes to be stored for each cell in a type of value attribute table where each column is an attribute and each row either a pixel or a pixel class. However, this is not the norm for many GIS and is not supported in NetLogo. See Longley et al. (2010) for an example.

5.3.2 Vector Data

The vector data model is used to represent discrete objects with well-defined boundaries. The vector representation allows for variable resolution of objects. This is in contrast to the raster representation which has a fixed resolution depending on the specified cell size. The vector representation can be advantageous; by defining individual elements as square cells, one could argue that their boundaries bear no relationship to the natural features they are representing. An example of this can be seen in Figure 5.1 which compares cells (raster) and polygons (vector) depicting the same area. To represent this geographical area using the raster data model, a series of cells are needed, whilst the vector model requires only the construction of a polygon.

Within the vector data model, each object is first classed as a geometric type: in the two-dimensional case, these are points, lines or areas (polygons) as shown in Figure 5.3. Points (e.g. people, cars, points of interest) are recorded as a string of coordinate pairs, lines (e.g. roads and rivers) as a series of ordered coordinate pairs (also known as polylines) and polygons (e.g. census tracts, houses) as one or more line segments that close to form an enclosed area. Information pertaining to these discrete entities such as a population can be expressed in a table (i.e. .dbf in the shapefile – one of the standard GIS vector files) with each row corresponding to a different discrete object (e.g. a person) and each column to an attribute of the object (e.g. sex) with its location as a string of coordinate pairs. The most common vector format is the ESRI shapefile.

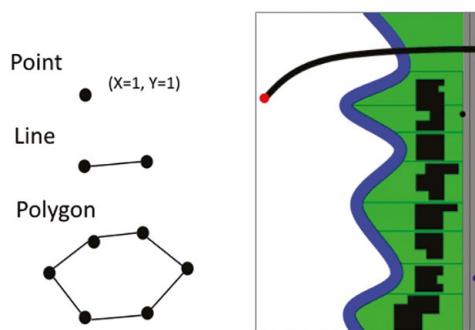


Figure 5.3 The basic building blocks of vector data: points, lines and polygons. Different types of objects can be combined to represent the geometric properties of an area

Much social, environmental and administrative data comes in vector formats such as census tracts, road networks, administrative boundaries and property records. However, until recently vector data was little used in agent-based models. One could argue that this is because many agent-based modelling toolkits did not

support this and also because of the high computational burden of carrying out geometrical operations.

A typical GIS contains multiple layers of data. A layer is made up of a number of elements. For example, a layer might contain a number of houses that represent a part of an area (as shown in Table 5.1), while other layers might include data on environmental factors such as parks. Each house in the layer would be a GIS feature (with an associated feature ID). Each feature in the layer has two aspects to it: its geographical coordinates and the data associated with it (its attributes). A common format for storing this information is the ESRI vector shapefile. Although the shapefile is a proprietary format, it has become the *de facto* standard GIS data format for most GIS tools. A number of files are associated with the shapefile format: (1) the shapefile (.shp), which stores the geographical information needed to display the feature (x , y , z coordinates of vertices and edges of the geometric shapes); (2) the database file (.dbf), which stores the data records for the feature; and (3) the index file (.shx) which links the database file to the shapefile (via the feature ID).

5.4 Time in GIS

GIS is not only about representing the world in a digital form but also quantifying and analysing change over time. Objects (such as people) and features (e.g. land cover, rural or urban) have one or more properties in time and space, and these are subject to change. It is no surprise that the subject of representing time within GIS has received considerable attention (see Langran, 1992; Peuquet, 2005). As noted by Heywood et al. (2006), a GIS should be able to represent temporal change using methods that explicitly represent spatial change, as well as different states through time. Furthermore, methods should allow for the direct manipulation and comparison of simulated or observational data in temporal and spatial dimensions. Unfortunately this is not the case, and methods to record, store and visualise information over time are still in their infancy.⁶

There are three approaches to capturing space-time information within a GIS: location-based, time-based and entity-based. However, the only method of viewing a data model within an existing GIS, as a space-time representation, is as a temporal series of spatially registered ‘snapshots’ (Peuquet, 2005). In this case, information for the entire layer is stored for each time step, regardless of whether any change has occurred since the previous step – for example, taking a

⁶ One possible solution is the geo-atom theory (Goodchild et al., 2007). This has yet to be implemented in a GIS but is starting to be used in modelling (see Jjumba and Dragičević, 2016).

Table 5.1 Sample of GIS type data, representing a layer of data in a GIS

House Layer			
Feature ID	Geographical coordinate	Type	Age
1	22,22	Detached	43
2	23,12	Flat	51
3	23,32	Semi-detached	100

new image via the *Landsat* satellite and then detecting changes using map algebra. With respect to vector data, time can be incorporated by providing a simple attribute field (for objects (features) that do not change shape such as countries, where only the population changes over time). For features that do change shape (e.g. the extent of a wildfire or road changes), each time step can be represented as a different feature (i.e. an entity which can be detected using geoprocessing tools in QGIS).

Why address the issue of time here? Time is an implicit feature within agent-based models. Objects (agents) move or alter the landscape through space and time – for example, agents' movement traces in the case of a pedestrian model (Torrens, 2012) or land-cover change in the case of a farming model (Deadman et al., 2004). Agent-based simulations often generate huge amounts of data, and it is extremely important to consider how to best capture, analyse and visualise these outputs.

The snapshot approach generates a huge amount of redundant data (i.e. objects that have not changed are stored in consecutive snapshots) and change can only be determined between snapshots. It is often challenging to detect the precise moment that a change occurs between snapshots. The lack of tools to capture continuous data over time has resulted in few applications having adequate temporal data available to analyse. However, this is beginning to change, with human movement data from GPS traces becoming more abundant and the availability of repeated coverage from remote sensing satellites. These improvements in the availability of temporal data have led to the development of GIS tools for visualising temporal change – for example, the TimeManager plug-in for QGIS or Time Slider window in ArcGIS.

5.5 GIS Software

There has been a huge growth in GIS software; some of the most widely available packages are listed in Table 5.2. Current commercial and open source GIS software systems all contain tools for acquiring, pre-processing, and transforming data. ESRI is the largest of the commercial companies developing GIS software

with a large user base, both commercially and within academic institutions. This is in part due to its sophisticated suite of tools (e.g. Spatial Analyst, Network Analyst, Geostatistics, etc.) which allows for nearly all GIS tasks to be carried out with relative ease and numerous help forums. However, ESRI GIS products carry a large license fee and require Windows to run. For Mac users, one needs to use a virtualisation program, such as Parallels or Bootcamp, to run the software. There are several open source GIS packages now available, the most popular being Quantum GIS (QGIS). QGIS offers much of the same basic functionality as ArcGIS – the ability to create, edit, analyse and visualise geographical information (via its plug-ins) – but without the cost or restriction to a specific operating system. Another open source product is GRASS. This provides similar functionality to QGIS but also offers more functionality for raster data. There are also other tools available for carrying out spatial analysis. While not a fully fledged GIS, GeoDa (Anselin et al., 2005) provides a GUI to carry out exploratory spatial data analysis, including global and local spatial autocorrelation methods, along with tools to create .gal files. These are text files which record the object and the number of neighbours that each object has. For example, in Figure 5.4 we show how, using GeoDa, we query each polygon (object) for its neighbours and then store this information in a file. This feature is incredibly useful for knowing who the surrounding neighbours are, avoiding complex geometry calculations (we use such an approach in Section 6.4.2).

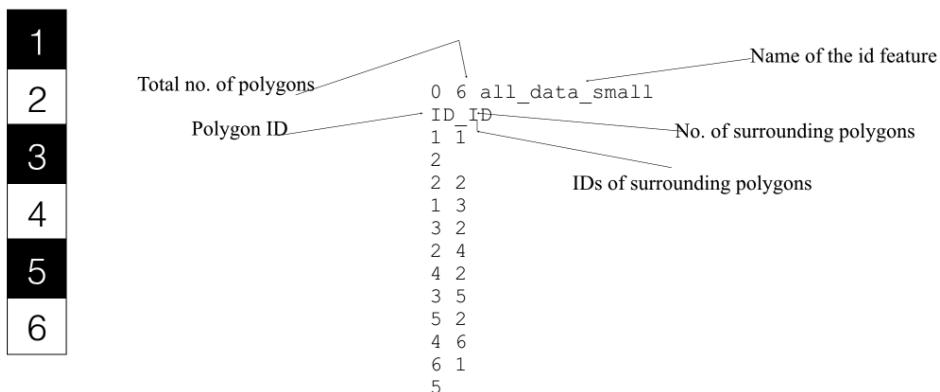


Figure 5.4 Using GeoDa to create a .gal file: (left) a collection of polygons (objects); (right) the .gal file where each polygon records its neighbours

There is also a growing body of tools and packages in R (see Brunsdon and Comber, 2015) and Python (e.g. PySAL (Python Spatial Analysis Library); Rey and Anselin, 2010) which allow for spatial data handling, analysis and the

visualisation of geographical information. NetLogo also has an R extension which allows one to use R within a NetLogo model (Thiele, 2014).⁷

Table 5.2 A sample of GIS platforms

GIS	Brief description	More information
ArcGIS	Leading commercial GIS software	https://www.esri.com/
QGIS	Open source GIS software with many plug-ins	https://qgis.org/
GRASS	Open source GIS software with many add-ons	https://grass.osgeo.org/
GeoDa	Open source tool for exploratory spatial data analysis	http://geodacenter.github.io/

5.6 Sources of Geographical Data

There has been a tremendous growth in geographical data, and there are several online sources from which data can be downloaded, as outlined in Table 5.3. These sources of data not only provide GIS researchers with a valuable means of tracking change or carrying out spatial analysis, but also provide agent-based modellers with new avenues for building spatially explicit models. Over the past few years, various sources of data have become available, offering new opportunities for GIS and agent-based modelling researchers alike.

Table 5.3 Some examples of online geographical data sources

Name	Brief description	More information
Natural Earth Data	Public domain vector and raster data sets of counties and points of interest	http://www.naturalearthdata.com/
USGS Earth Explorer	Remote sensing data (e.g. Landsat and digital elevation)	https://earthexplorer.usgs.gov/
National Land Cover Database	30 m land-cover database for the USA	https://www.mrlc.gov/
OpenStreetMap	Crowdsourced map of the world	https://www.openstreetmap.org
WorldPop	100 × 100 m gridded population for low- and middle-income countries	http://www.worldpop.org.uk/

⁷ To find out more about the NetLogo R extension, see <https://ccl.northwestern.edu/netlogo/docs/r.html>

5.6.1 Volunteered and Ambient Geographical Information

The growth of crowd-contributed geographical data (often referred to as ‘crowdsourced’) can be largely attributed to either volunteered geographical information (VGI; Goodchild, 2007) where a crowd actively contributes geographical knowledge (e.g. OpenStreetMap, Wikimapia), or ambient geographical information (AGI, Stefanidis et al., 2013b), where the data needs to be mined to derive geographical content (e.g. Twitter or Flickr). In both instances one can consider individuals as sensors, capturing, commenting and editing the world around them in a digital form. Figure 5.5 shows three representative examples of VGI. Maps.Me (Figure 5.5A) is a mobile application for use on smartphones and tablets that provides offline maps using OpenStreetMap data. Users can also edit the map and upload their changes to OpenStreetMap. SeeClickFix (Figure 5.5B) enables users to communicate issues such as graffiti or, in this case, potholes to local governments. Mapillary (Figure 5.5C) allows users to upload geotagged photos including panoramas and photo spheres and could be considered an open-source version of Google Street View. Unlike VGI, where users knowingly contribute information, AGI has emerged from instances where users contribute geographic information unknowingly – for example, posting a tweet from a smartphone where the GPS has been enabled. Such information needs to be processed and mined using geographical analysis tools or machine learning techniques such as topic modelling and sentiment analysis to extract geographical information. Examples of AGI include delineating the extent of earthquakes using Twitter (Crooks et al., 2013) and wildfires using Flickr (Panteras et al., 2015, 2016), access to healthy food (Tenkanen et al., 2016), and studying the spread of invasive species (Daume, 2016), visitors to lakes (Keeler et al., 2015), other recreational sites (Wood et al., 2013) and protected areas (Tenkanen et al., 2017).

5.6.2 Social Media Data

Ten to twenty years ago, the only data option that researchers had when studying geographical systems was to draw on authoritative sources – for example, census data pertaining to people, or remotely sensed data for land use (as discussed in Section 5.5). Such data, while extremely valuable, only provided snapshots of the people and environment in which they lived. The arrival of new and various forms of data in the ‘big data’ context, especially in the form of VGI and AGI, means that there is now an unprecedented amount of geographical data, giving us the ability to study the connections between people and places at fine spatial and temporal scales. This provides us a new lens to study how people interact with each other and use the space around them (Crooks et al., 2015b). For example, crowdsourced data allows us to study not only the form

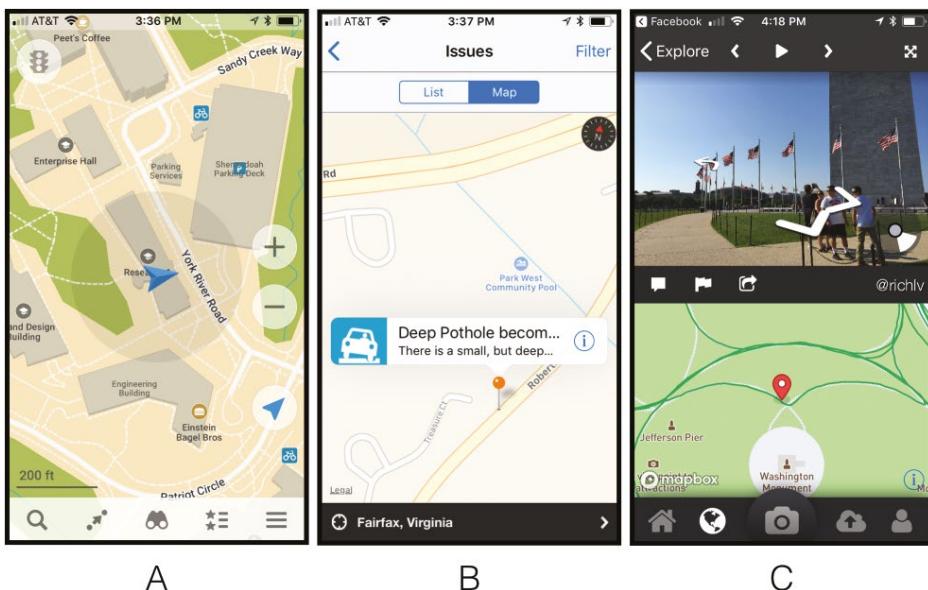


Figure 5.5 Examples of volunteered geographical information: (A) Maps.Me; (B) SeeClickFix; (C) Mapillary

(e.g. the physical aspects of the city) but also its function (e.g. how people use the space of the city). Together with more traditional sources of data, it provides us with new opportunities to study urban morphology.

There are also implicit data sources whose original content was not meant to directly convey form or function information, but this information can be extracted from it (e.g. extracting a road network from geolocated traffic data or address points from Foursquare). In this sense, explicit information is directly contributed, whereas implicit is extracted from other data sources (using techniques such as topic modelling; see Jenkins et al., 2016, for such an example).

Such data offers us new, detailed information about the behaviour (e.g. movement, actions and decisions) of individuals, how they react to events or their opinions on a diverse range of topics such as natural disasters, political events, protest movements and international relations (e.g. Vieweg et al., 2010; Stefanidis et al., 2013a; Croitoru et al., 2015).

Much of this data has a substantial geographical component (e.g. coordinates from which the contributions originate, whether from tweets or from references to specific locations), and as it is contributed by individuals, the term *geosocial* has emerged (Croitoru et al., 2013). At the same time, information on the underlying social structure of the user community can be derived by studying the interactions between users (e.g. formed as they respond to each other via retweets

or direct messages, or follow other users), and this information can provide additional context to the data analysis. With the rise of this data we can now also explore not only the geographical environment (e.g. through OpenStreetMap or other physical terrain data sets) and the people (e.g. through census data and social media) but also how such people react and respond to events (e.g. via news reports or eye witnesses), as shown in Figure 5.6.

Moreover, by taking a geosocial view of the world we can uncover networks across both cyber and physical spaces – for example, how a protest in New York city or an explosion in Boston reverberates around the world (Croitoru et al., 2015) or how the physical spread of the zika virus or a measles outbreak is captured by the public in areas not in close proximity to it, along with who are the key actors in a specific debate (e.g. Stefanidis et al., 2017; Radzikowski et al., 2016).



Figure 5.6 Geosocial analysis: linking places, people and events

5.6.3 Remotely Sensed Data

While so far we have focused on socially constructed data, there are also other, non-social aspects of ‘big data’. Earlier we highlighted that the continuous collection of data in GIS was often a challenge for traditional data collection due to a lack of tools to capture data and the sparsity of sensors (e.g. satellites). However, with increased computational power and advances in technology, remote sensing from satellites has also grown tremendously over the last ten years. Today we have hundreds of Earth observation satellites, under civilian and/or commercial control, that provide measurements for many earth science processes (see Belward and Skøien, 2015). These are expected to increase significantly with the rapid proliferation of constellations of CubeSats (see Selva and Krejci, 2012, for a review) such as those from Planet. These will provide an opportunity for high temporal and spatial resolution multispectral imagery, thus adding to the ‘big data’ deluge. Companies such as Digital Globe have a 100-petabyte (PB) digital library of high-resolution satellite imagery, and the

NASA Earth Observing System Data and Information System is estimated to house over 7.5 PB of archived imagery, with NASA generating approximately 5 TB of data per day (Vatsavai and Bhaduri, 2013). Such data is now providing us with the opportunity to observe and monitor the Earth like never before. Sites such as Google Earth require Google to archive over 20 PB of imagery, from satellite to aerial and ground-level street view images (McKenna, 2013).

Moving from space to the surface of the Earth, there has also been the proliferation of new data sources that collect information remotely. Perhaps the most invasive of these are video surveillance systems, deployed indoors and outdoors for a wide variety of facilities, ranging from hospitals, schools and shopping malls to airports, government facilities and military installations. To give a sense of the scale of this, it is estimated that in the city of Chicago there are approximately 10,000 video cameras deployed for this purpose (ABC7 News, 2010). In the UK, it is estimated that between 2 and 4 million closed-circuit TV (CCTV) cameras are currently active, with over 500,000 operating in London alone (Norris et al., 2004; Gerrard and Thompson, 2011). With Wi-Fi technology (see Torrens, 2008), radio-frequency identification (see Hahnel et al., 2004), GPS (e.g. Orellana and Wachowicz, 2011), mobile phones (see Manley and Dennett, 2018) and travel smart cards (see Zhong et al., 2016), we have many new opportunities to study human mobility and interactions. With respect to models, it has already been noted that many models make use of remotely sensed data for initialisation, calibration and validation (Sections 5.3.1 and 6.4.1), and this new data should be of value to such applications. Similarly, for models focusing on human mobility, data from CCTV, smart card transactions, mobile phone traces and GPS can also be used.

How can we use ‘big data’ both to create new understanding about the world around us and to improve our agent-based models (or modelling more generally)? A great deal of work has been carried out analysing ‘big data’ to explore issues impacting society from the perspectives of who is contributing information, where they are located, what they are doing or reacting to and, perhaps most importantly, why. For example, ‘big data’ has been used to detect and delineate events such as earthquakes (e.g. Crooks et al., 2013), wildfires (e.g. Panteras et al., 2015) and protest movements (e.g. Croitoru et al., 2015). It is also being used to examine how people perceive neighbourhoods (e.g. Cranshaw et al., 2012), travel with regularity and irregularity (Manley et al., 2018), to explore populations at risk (Malleson and Andresen, 2015, 2016), navigation strategies (e.g. Manley et al., 2015a), and to investigate travel patterns (Chen and Schintler, 2015). For example, Figure 5.7 is taken from the work of Jenkins et al. (2016) who mined Twitter to find spatially significant entertainment clusters within New York City (red dots) and examined how these aligned with the actual Theatre Subdistrict utilising advances in topic modelling (e.g. Blei et al., 2003).

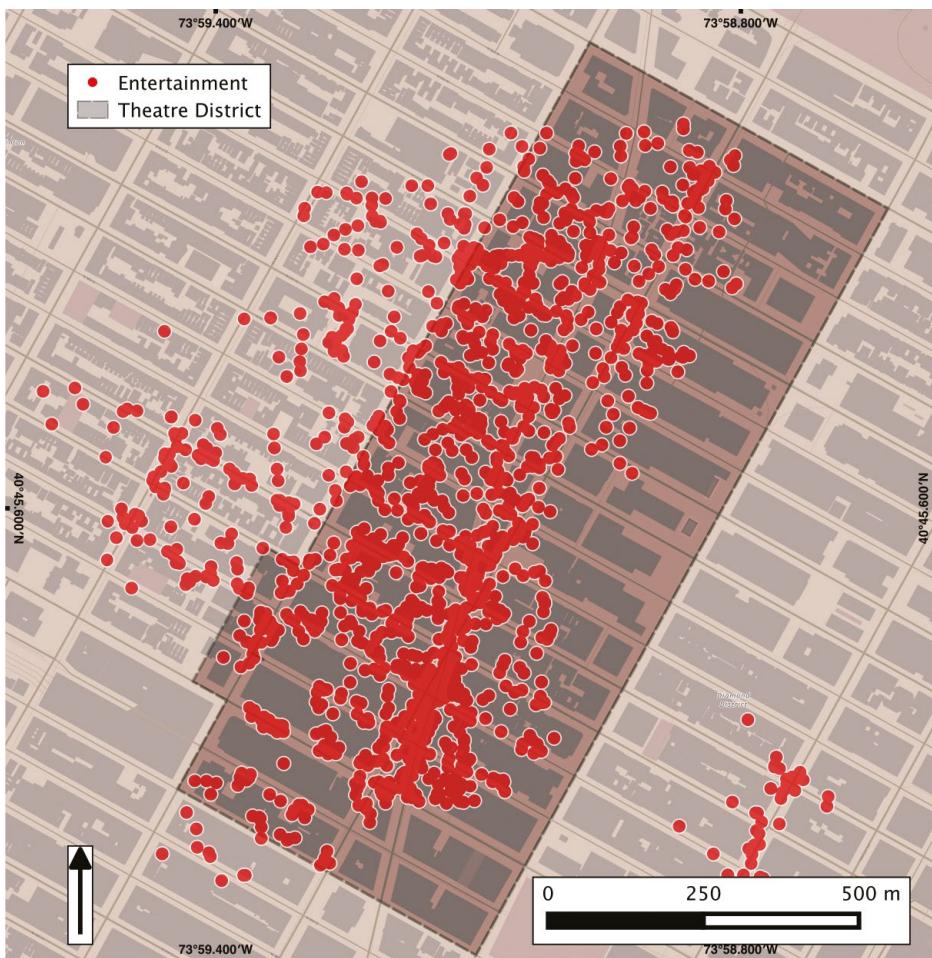


Figure 5.7 Mining tweets relating to entertainment and contained within and near the Theatre Subdistrict of New York City (Jenkins et al., 2016)

5.7 Preparing GIS Data using QGIS

QGIS has become a popular GIS platform. Part of its success stems from the fact that it is free, open source and runs on many different operating systems. QGIS is being updated constantly, so rather than providing a full tutorial here – there are some excellent tutorials available already (some of which are included in the bibliography for this chapter) – this section will briefly outline some of the common GIS operations that are likely to be useful for developing spatially explicit agent-based models.

5.7.1 Performing Spatial Operations

It is common to use a GIS to manipulate spatial data before they are used as input into a GIS. This section will make use of WorldPop⁸ and Natural Earth⁹ data (see Table 5.3) as shown in Figure 5.8. It will demonstrate how to load the data sets into QGIS, ‘clip’ them (a common spatial operation) and then load them into NetLogo in such a way that the WorldPop data can be used to spawn new agents, and the Natural Earth data can be used to mask out areas of water that cannot hold

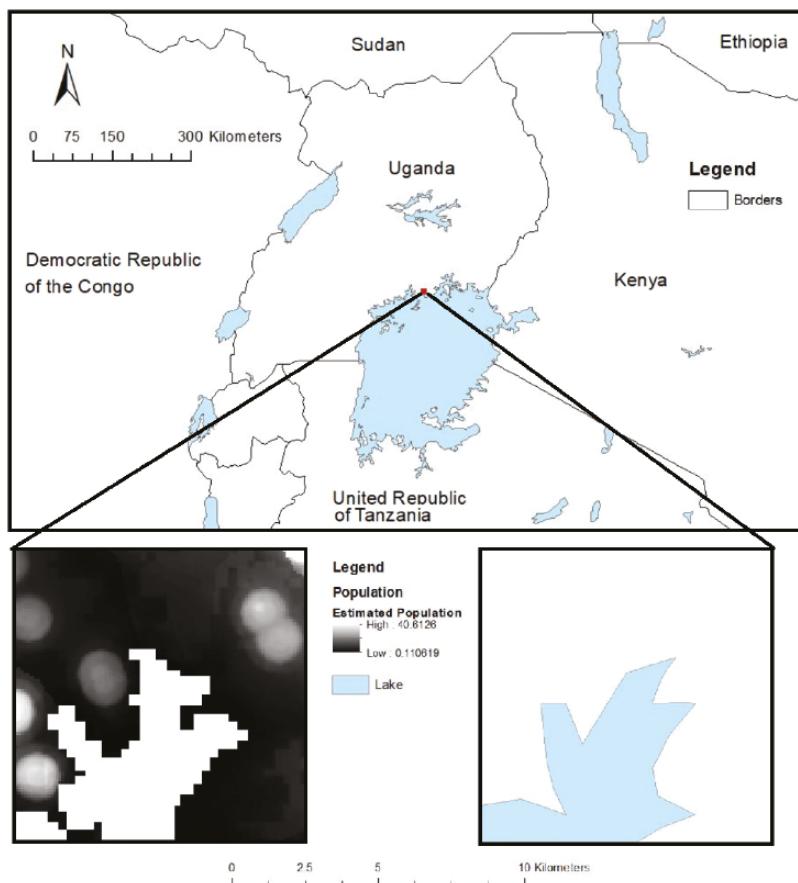


Figure 5.8 An example of using open data, in this case data from Natural Earth and WorldPop, for a 10 × 10 km area near the city of Makindye in Uganda

⁸ www.worldpop.org.uk/

⁹ www.naturalearthdata.com/

populations (this prevents the agents from trying to move onto water). This model is presented in Figure 5.9. The required data files and model code are available in the `UgandaExample` directory on the accompanying website. Figure 5.10 outlines the overall process.

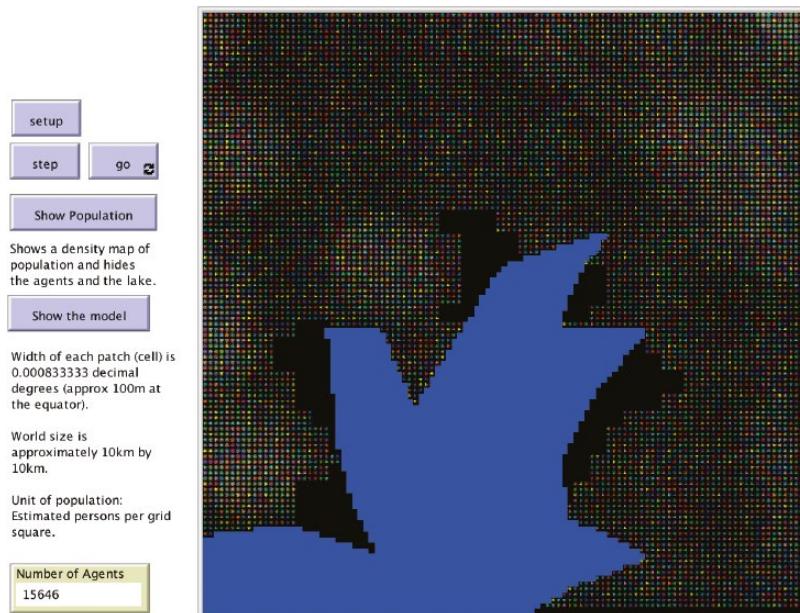


Figure 5.9 An example of using open data for the initialisation of an agent-based model

To begin with, obtain one of the raster population files.¹⁰ In this example we will model an area in Uganda, but the following steps could be applied to any available area. After downloading a population file it can be opened in QGIS by using the following menu selection: Layer → Add Layer → Add Raster Layer. This will load the whole population file for all of Uganda. In this example we are only interested in a 10 km² area near the city of Makindye on the shore of Lake Victoria, so we need to *clip* (a.k.a. ‘cookie-cut’) the population data. We have already prepared a 10 km² vector file called `clip`, which can be loaded into QGIS with a command that is similar to the one used to load raster data: Layer → Add Layer → Add Vector Layer.

After loading the `clip` file a small square appears over the population data. It is easier to see this in more detail by zooming into the area that we are going

¹⁰ www.worldpop.org.uk/data/

to clip. To do this, right-click on the `clip` file (in the left-hand window) and choose `Zoom to Layer`.

The next step is to cut ('clip') the area from the larger population file. This can be accomplished in QGIS by selecting `Raster -> Extraction -> Clipper` and choosing the `Mask layer` option.

We have now the clipped population data for the area that we are interested in. However, if this data were included in our model right away, there would be nothing to stop the agents moving into the lake! Therefore we have to prepare one more GIS file that we can use to prevent the agents entering the water. Download the 10 m Lakes GIS vector data file from Natural Earth¹¹ and load it into QGIS. As before, the lake will cover a much larger area than we are interested in, so again we need to clip the lake to our area of interest. This menu for clipping vector data is slightly different from that for raster data: `Vector -> Geoprocessing Tools -> Clip`.

That's it! We now have a clipped population file that specifies the initial positions of our agents and a clipped lake file that we can use to restrict their movements. Figure 5.10 illustrates the whole process. For an example of how to read these files and use them in NetLogo, refer to the `UgandaExample` model.

5.7.2 Manipulating Vector Table Data

The previous subsection has demonstrated how to perform some simple spatial operations (namely 'clip') using QGIS. This section will look at *vector* data in more detail. Whereas raster data usually only represents a single value in each area – such as the height of land, or the number of people per grid cell – vector data can be underpinned by large tables, with each row representing an area. For example, some vector data of census tracts could include a variety of demographic information about the residents such as age, income and gender. This subsection will briefly illustrate how to manipulate the tables that underpin vector data. Schelling's (1971) well-known model of segregation will be used as the example (discussed in detail in Section 2.4.1). Rather than using a regular grid as Schelling did, we will load some spatial data for the area of Washington, DC, and simulate the interactions of the agents in an environment that is a closer representation of a real area. Later, in Chapter 6 (Section 6.4.2) the NetLogo model itself will be discussed; the focus here is on data preparation. Specifically, we will demonstrate how to create a new column in the Washington table to tell NetLogo whether the agent who lives in an area should be 'blue' or 'red'.

The following steps are illustrated by Figure 5.11. To begin with, load the Washington DC file into QGIS. The required shapefile, called `DC.shp`, is available

¹¹ www.naturalearthdata.com/downloads/10m-physical-vectors/

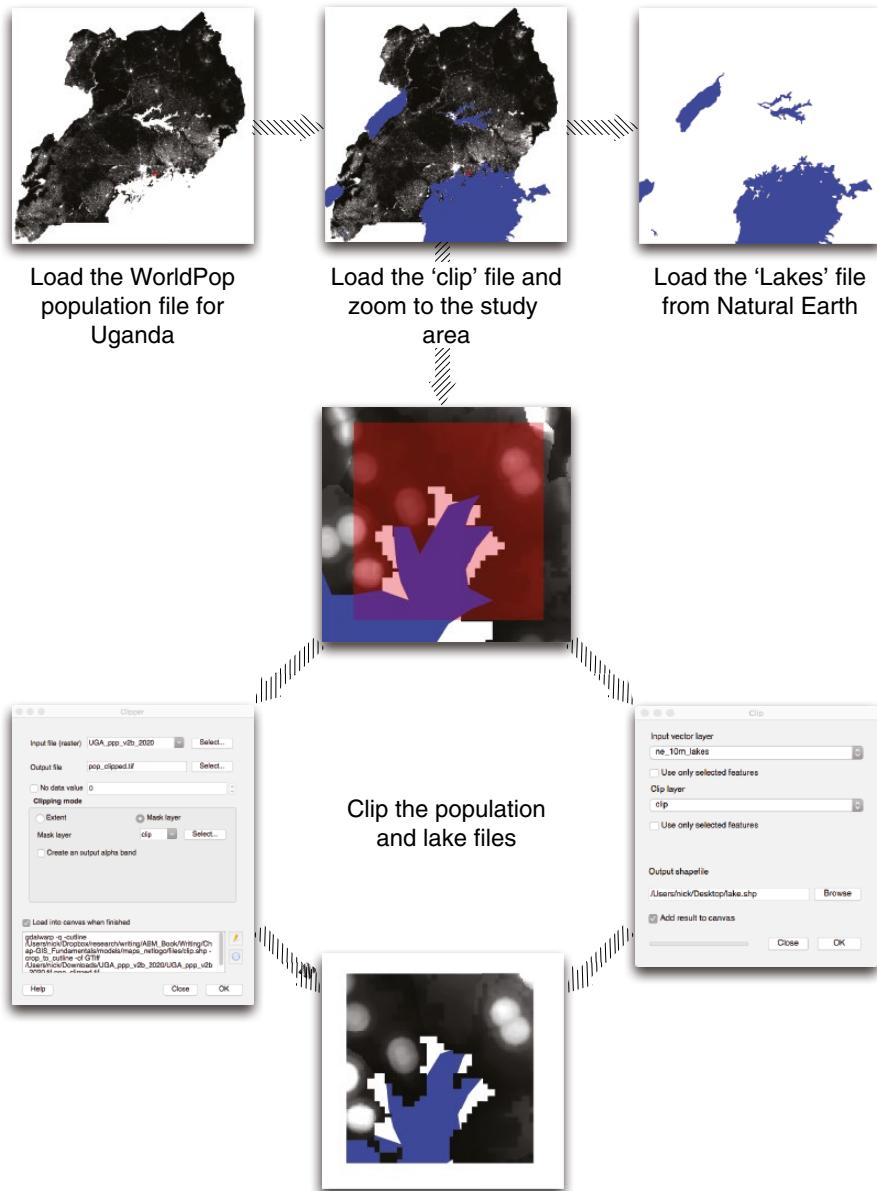


Figure 5.10 An example of how to load and ‘clip’ vector and raster data files using QGIS

in the DC Data directory of the accompanying online material for this chapter. As in the previous section, the data can be loaded by selecting Layer → Add Layer → Add Vector Layer.

Open the ‘attribute table’ to see the table of data that underpins the shapefile. To do this, right-click on the DC layer (in the Layers Panel on the left) and choose Open Attribute Table. You should now see the attribute table for the Washington, DC data. Each row represents an area and the columns hold the different values for each area. There are five columns in the data. The next task is to create a new column and calculate whether the agents in the area should start off by being ‘red’ or ‘blue’.

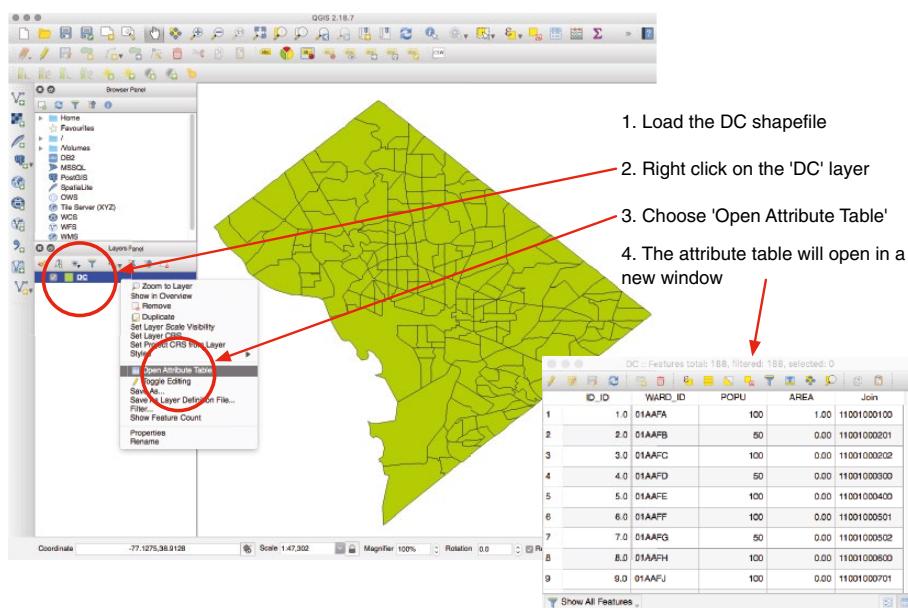


Figure 5.11 Instructions for opening the attribute table for a vector layer using QGIS

To create a new column, we first need to tell QGIS that we want to edit the layer. Do this by clicking on the pencil icon (the Toggle editing mode option) at the top of the attribute table; see Figure 5.12). Once the layer is editable, click on the New Field option. QGIS now needs some information about the new field:

- *Name*. This is the name of the field. Call it “SOC”.
- *Comment*. This can be used to save additional information (metadata) about the new field. You can leave it blank, or write some notes in there.
- *Type*. This tells QGIS what data type we want to store (e.g. integers, decimals or text). We want to store text, so choose Text (string).

- *Length.* The length determines how many characters we can store. We need to be able to store one of the following words: BLUE, RED or UNOCCUPIED. Ten characters will therefore be sufficient, so choose length 10.

Finally, we need to populate the "SOC" column. Usually this value would be calculated from some other fields. For example, we might have socio-demographic data for our study area (e.g. use the mean income in a neighbourhood, or the level of deprivation) and would use this to set the colour of the agents accordingly (e.g. by allocating all neighbourhoods above a certain level of deprivation to BLUE). In this example, however, we will set the "SOC" of each area randomly. Figure 5.12 illustrates how to do this using the Field calculator. We use a function called `rand` that chooses a random number between 1 and 3 (inclusive) and then choose 'RED', 'BLUE' or 'UNOCCUPIED' depending on whether the random value is 1, 2 or 3. The Field calculator is very powerful and also very well documented. Code Example 5.7.1 outlines the code that was used to calculate the "SOC" randomly.

Code Example 5.7.1 Code for the QGIS Field calculator

```
CASE
WHEN rand( 1,3 ) = 1 THEN 'BLUE'
WHEN rand( 1,3 ) = 2 THEN 'RED'
ELSE 'UNOCCUPIED'
END
```

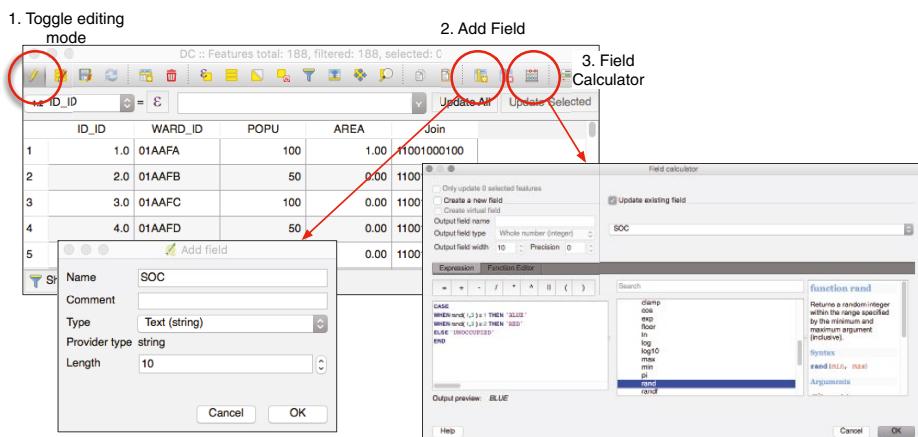


Figure 5.12 How to add a new column to the attribute table that underpins a vector layer using QGIS and calculate new values for the column

Finally, to complete the field calculation, click on the Toggle editing button and save the changes to the layer. We now have a new column that can be used to create agents of a specific type. Section 6.4.2 will discuss the model that can make use of these new data.

5.8 Visualising Results

Building effective visualisations of spatial analysis and modelling outcomes, be they derived from GIS or agent-based modelling, is a vital final component of the analysis process. Good-quality maps and visualisations not only explain the outcomes of the analysis, but also aid interpretation by allowing observers to easily draw out insights. The process of visualisation development combines science and art, and while not all of us may count ourselves as artistic, there are numerous ‘rules of thumb’ that can be followed to ensure an effective data visualisation is achieved.

Any strong geographic visualisation emphasises and contextualises the key trends found during the analysis. Of particular interest are areas of spatial clustering and dispersion in our system of interest, such as the residential clustering of demographics, variations in traffic or pedestrian flow, and inequality in the health outcomes. The occurrence of these visual patterns (which can also be statistically measured, as discussed in Chapter 9) is indicative of the heterogeneity and irregularity that notably characterise spatial and social systems. These same patterns are what we are seeking to achieve through agent-based modelling.

5.8.1 Map Types

There are a variety of approaches for visualising spatial data, and choice much depends on the level of aggregation the data is available in (e.g. point, line, polygon), and the scale at which you are visualising the data. In making a choice, it is important to remember that humans are only able to visually comprehend a limited amount of information, and therefore control of this information is vital. Some of the main options available for visualising spatial data are summarised below:

Choropleth. A choropleth map is a visualisation of a spatial trend at the polygon level. The colour of each polygon reflects the attribute of interest, often with the visually more intense colours indicative of higher values. Typical examples include population volumes and densities, crime rates, and political allegiances. Given that polygons are naturally aggregations of a trend, they are suitable for visualisation of large-scale systems, at country level and beyond, but this will naturally depend on the size of the polygons. Care should be taken to avoid misrepresenting the spatial nature of trends through the placement of the aggregating unit; this issue is known in geography circles as the modifiable areal unit problem.

Point. A simple point map visualises the locations of point features over space. Points represent features that can be linked to a specific coordinate, and that point may be associated with a value or set of values. The associated value, be it categorical (e.g. city status, station operator) or continuous (e.g. population, passenger count), can be visualised either through colour intensity (like a choropleth) or variation in point size, or both. Many agent-based model visualisations consist simply of points moving around space!

Dot density. A combination of choropleth and point maps is found in the form of dot density maps. In these visualisations, a value captured through polygon-level



Figure 5.13 A dot density map of income distributions in Winnipeg, Canada, constructed from 2006 Census data

aggregation is represented as a distribution of randomly located points within the bounds of the polygon, where each point feature represents a single count from the polygon data. An example of a dot density map is shown in Figure 5.13. Clearly, the randomised locations of the points can potentially introduce confusion, given that no point can be sure to represent the location of any feature, but if presented at a broad enough scale the visualisation can be highly visually appealing.

Lines. Where spatial data does not suitably align with a polygon or point representation, line maps are a necessary alternative. Lines can be visually manipulated for thickness and colour to emphasise locations with high values. Typical applications relate to transportation and network flows.

Densities. Density maps refer to the range of approaches for mapping distributions of values over space using a continuous surface. These are achieved through mathematical calculation of density or accumulation over space, yielding higher values in areas of higher density. The continuous surface can thereafter be visualised to emphasise areas of high value. Alternatively, simplistic aggregation may be applied in the form of hexagonal or triangular polygons, enabling more specific interrogation of specific regions.

3D. Visualisation in three dimensions enable exploration of additional dimensions of data within a single visualisation. For example, a choropleth map of life expectancy may be supported by a third dimension indicating government spending on health-care. Visualisation in 3D is, however, difficult to properly achieve from 2D perspectives, and is best used within a video or interactive presentation.

5.8.2 Map Elements

In building a data visualisation of spatial phenomena, supportive information is vital for helping the viewer understand and locate the trends and scale of variation. Things to consider for inclusion are:

Legend. A legend or key associates value ranges with the colours shown in the map. While the gradient of the colour may be indicative of the trends, a legend allows the viewer to more accurately associate a location with a value.

Scale bar. A scale bar indicates the relationship between the observed distances within a map and their actual distance in the real world. Scale bars are more important where a viewer may be unfamiliar with the mapped location, but nonetheless represent a subtle addition to a map that can aid interpretation.

Labels. Labels can be extremely useful in guiding the viewer's understanding of a visualisation. The label may indicate a location, such as a town or region name, or signify exact values, drawn from the map for additional explanation (e.g. a maximum or minimum value).

North arrow. Convention indicates that the top of a map indicates north, but in cases where a map might be presented otherwise (e.g. within an interactive

environment, or absent from other text) inclusion of an arrow pointing towards north is a useful addition.

Text. Supporting text is another vital component for aiding understanding of a visualisation. This could be as simple as a title, or some narrative explanation. But take care not to overcomplicate a map with large amounts of text.

Graphs. Graphs and other plots can add further insight into the spatial trends shown in a map, and are particularly useful where certain distributions cannot be properly interpreted spatially.

An example of the effective use of these features is shown in Figure 5.14.

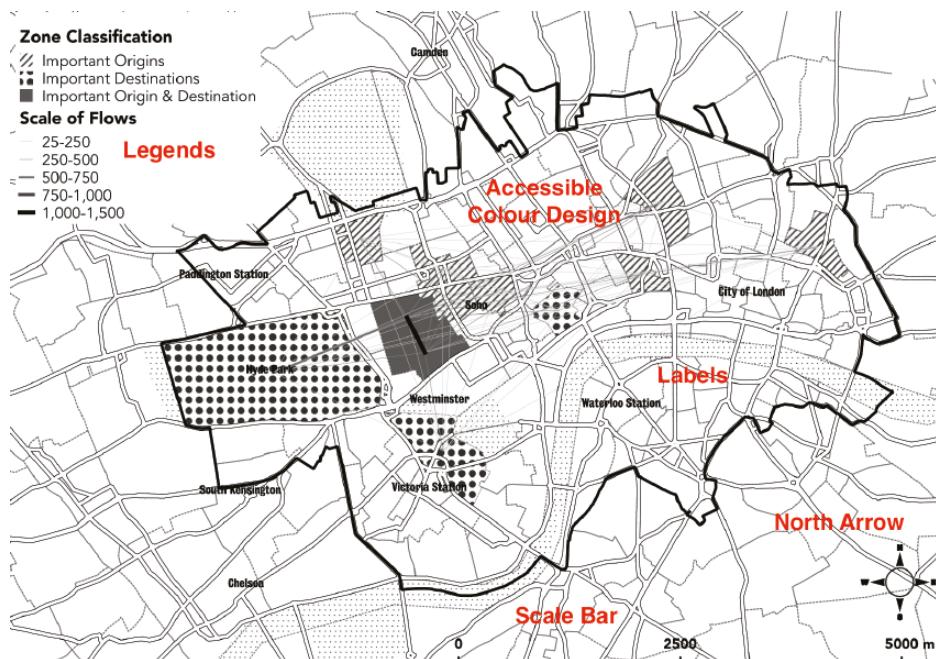


Figure 5.14 Map from Demšar et al. (2018) demonstrating strong map design, through inclusion of a legend, scale bar, supporting labels, north arrow and accessible colour design

5.8.3 Styling Trends

Colour can be highly instructive in describing the nature of spatial trends. Different shades and intensities can provide an indication of continuous progression or categorical change, but also, most importantly, allow the viewer to easily interpret spatial areas of similarity and difference. In selecting a colour scheme to adopt for your visualisations, there are a few considerations to bear in mind.

First, the number of *classes* into which to split the data must be defined. Typically a spatial visualisation will not use fewer than four or more than seven classes to visualise the data. These classes generally ensure there is enough differentiation across the map, but without overwhelming the viewer. Next, one must decide how the data will be split into these classes, and where the boundaries will be drawn. There are numerous methods for calculating these ‘breaks’, each with their advantages and disadvantages (a more informative discussion can be found in Longley et al., 2010, p. 313). Generally the objective is to balance an equal number of spatial units within each class with maintaining similar values within the classes, thus providing as indicative an overview of the spread of data as possible. Jenks breaks (also known as natural breaks) constitute a method that works particularly well in achieving this, through optimising to both minimise intra-class and maximise inter-class variance. Alternatively, quantile breaks, also widely implemented, simply create groups with equal features counts in each.

Once the classes are defined, a second task is to define the colour range. Colour speaks its own language – some colours indicate heat, some cool, some are startling, while others are calming – so colour selection is a vital component of map design. The colour scheme should also reflect the nature of the data being mapped – be it sequential (e.g. populations, volumes), divergent (e.g. positively or negatively deviating from a mean or zero) or categorical. Fortunately, there is an excellent tool available to take the guesswork out of colour selection. The colorbrewer2.org website provides a range of colour schemes that can be adapted for the type of data being mapped and the number of classes desired (Harrower and Brewer, 2003). Colorbrewer2 was developed by Cynthia Brewer, a specialist in cartographic design, to help best visualise and emphasise similarity and variation in spatial data. The tool also offers a colourblind safe test, ensuring that resulting visualisations are fully accessible to all potential viewers.

5.8.4 Interactivity

A final consideration in visualising data is whether and how to incorporate interactivity. Interactivity can be highly instructive when used in the right context, as it allows users to interrogate spatial trends and gain more insight into how local variation contributes to global patterns. This means more information can be built into a visualisation than may be apparent from an initial view of the map. The example in Figure 5.15 shows how a hexagon-based choropleth map may be interrogated using a mouse or touchscreen, and allows the user the capability to drill down into further local area-specific data.

Achieving interaction is not so straightforward, however. Interactive visualisations are typically built using web technologies, and therefore command a learning curve prior to the development of any visualisation. Many modern interactive visualisations are built using the JavaScript-based package D3.js (readers are directed

to the wealth of impressive examples at d3js.org), but this requires advanced coding skills. Fortunately, there are an increasing number of easier-to-use tools, building on the advances introduced through D3.js, that integrate well with data handling and analysis workflows. In particular, readers are directed to investigate Bokeh,¹² which builds on Python, and Shiny,¹³ based on the R scripting language.

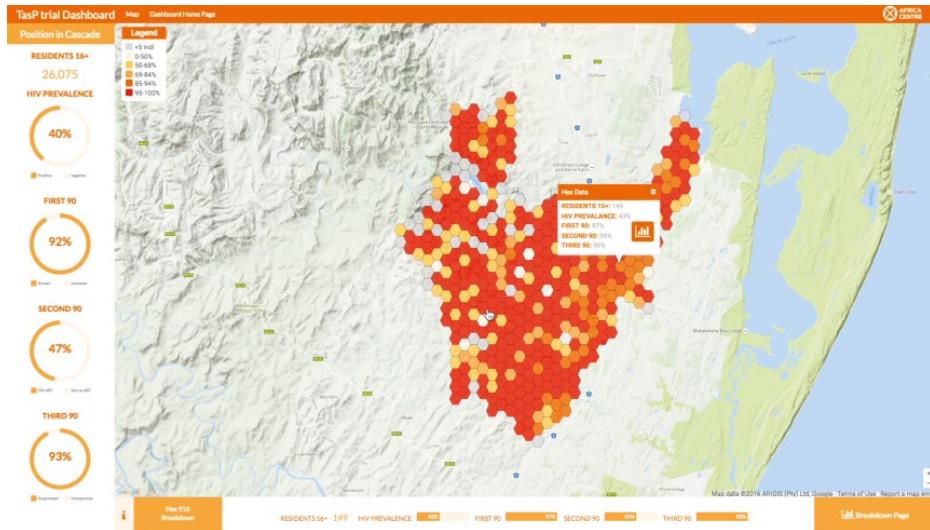


Figure 5.15 A screenshot of the TasP trial dashboard installed at the Africa Health Research Institute in Kwa-Zulu Natal, South Africa, which enables interactive exploration of global and local trends in HIV rates, and combines hexagon-based choropleth mapping with supporting figures, graphs and text-based notes

5.9 Discussion

Since its first appearance in the late 1960s, GIS have become a standard tool for researchers interested in manipulating and visualising spatial data. With the mass proliferation in georeferenced data, GIS is playing a significant role in unlocking this data for use both in academia and by policy-makers. When GIS became fully established in the 1980s and 1990s, one company, ESRI, dominated the scene. However, with the advent of the internet and increases in data storage and processing power, several alternative GIS products have appeared on the market.

¹² <http://bokeh.pydata.org>

¹³ <http://shiny.rstudio.com/gallery>

The most notable of these is QGIS, which can perform many of the same processes as ESRI's ArcGIS, but is free and open source. This factor, along with an increase in the availability of different forms of open source data, is beginning to transform the GIS industry.

As with any analysis or modelling approach, using a GIS requires careful planning. The researcher should decide which of the data structures (raster or vector) is most appropriate for their application, what resolution the data needs to be at, and whether the data is current enough (or over a long enough time period) to represent the feature of interest. Whilst much of the power of a GIS lies in its ability to readily allow management and processing of spatial data, how the data is visualised is also key. Section 5.8 presented some of the different forms of visualisation available, but the key message is to think about who the results are being communicated to. What information is most important to be represented on the map?

While GIS can easily enable the manipulation, analysis and visualisation of spatial data, they are limited by their inability to handle dynamic processes. For example, a GIS could be used to create a map showing hotspots of a particular type of crime within an area. What they cannot easily do is reveal the evolution of those hot spots over time. Section 5.4 discussed some of the issues with analysing data over time, but for many research applications, understanding the dynamic nature of the system is key: how it evolved to that state, and how it will change in the future. While there have been attempts to include dynamic modelling elements within GIS (e.g. ESRI's Agent Analyst), many researchers use GIS solely for the preparation and analysis of data, with the modelling component undertaken in a different environment such as NetLogo for agent-based models. How and why we bring agent-based modelling and GIS together is the subject of the following chapter.

Chapter Summary

Geographical information systems are the mainstay of many researchers' toolbox. The ability to easily manipulate, analyse and visualise data makes them a powerful tool. However, researchers need to be aware of the underlying principles behind GIS, such as the data structures that are used and the limitations of GIS – in particular, their limited ability to handle and simulate dynamic processes.

This chapter has laid out the fundamental concepts that researchers need to be aware of, and presented an overview of the most common proprietary packages. A simple tutorial demonstrated how to bring data into a GIS and perform a simple spatial query. The next chapter will build upon this foundation and introduce the concept of coupling GIS with agent-based models.

5.10 Annotated Bibliography

For a detailed list of analytical techniques and methods used within GIS, readers are referred to:

- de Smith, M.J., Goodchild, M.F. and Longley, P.A. (2009) *Geospatial Analysis: A Comprehensive Guide to Principles, Techniques and Software Tools* (3rd edn). Winchelsea: Winchelsea Press.
- and the accompanying website: www.spatialanalysisonline.com/

To find out more about GIS, spatial analysis and modelling, along with the tools and methods one can use to explore geographical processes, readers are referred to:

- Maguire, D.J., Batty, M. and Goodchild M.F. (eds) (2005) *GIS, Spatial Analysis and Modelling*. Redlands, CA: ESRI Press.

Readers wishing to know more about using open source GIS software (including QGIS), tutorials and data are referred to:

- Graser, A. (2016) *Learning QGIS*. Birmingham: Packt Publishing.
- Abernathy, D. (2016) *Using Geodata and Geolocation in the Social Sciences: Mapping Our Connected World*. London: Sage.
- Online tutorials and documentation: www.qgis.org/en/docs/index.html and www.qgistutorials.com/en/

Readers wishing to know more about how field- and object-based views of space can be integrated into agent-based models (before waiting for Chapter 6) are referred to:

- Brown, D.G., Riolo, R., Robinson, D.T., North, M.J. and Rand, W. (2005) Spatial process and data models: Toward integration of agent-based models and GIS. *Journal of Geographical Systems*, 7(1), 25–47.

6

INTEGRATING AGENT-BASED MODELS AND GIS

Chapter Outline

Previous chapters outlined the fundamentals of GIS and agent-based modelling. In this chapter we will discuss the benefits to linking these approaches and explore the ways that this can be undertaken. We will explain loose and tight coupling, critiquing the relative advantages and disadvantages of both. We then present an overview of open source toolkits that can be used for the creation of geographically explicit agent-based models, before providing a critical look at where and how GIS and ABM should be combined, offering practical advice on best practice.

6.1 Introduction

Consideration of space is often integral to the success of agent-based models, especially those that are applied to geographical systems. For example, in the Schelling model presented in Chapter 2 (Section 2.4.1), an agent's decision whether to move or not is directly influenced by its neighbours. If the agent is dissatisfied with its current location (based on the mix of neighbours) it can move to an empty cell. This ability of agents to react to each other and to changes in their environment makes agent-based modelling highly relevant to many geographical problems (as discussed in Chapter 1). It is no surprise that there has been growing interest in the integration of GIS and agent-based modelling (e.g. Benenson and Torrens, 2004; Gimblett, 2002; Heppenstall et al., 2012b). Bringing these approaches together allows modellers to think about how objects or agents and their aggregations interact and change in space and time (Batty, 2005). For GIS users, it provides the ability to model the emergence of phenomena through individual interactions of features within a GIS over time and space. This last point is a movement away from the traditional focus of GIS on spatial representation, often ignoring temporal

representations (Peuquet, 2002). Moreover, from a perspective of understanding geographical systems, this linkage is highly appealing. GIS provides the ability to monitor the world, pulling together detailed information on areas, but it is unable to give insight into decision-making frameworks – for example, why people choose to live in such areas (Robinson et al., 2007). Through the integration of GIS and agent-based modelling we can capture both of these elements.

The simplest way to visualise the integration of geographical data and agent-based modelling is by taking a GIS view of the world, as shown in Figure 6.1. The complexity of the world is abstracted away and represented as a series of layers (such as the physical environment, the built environment). These layers form the environment for our *artificial* world (see Figure 2.3 in Chapter 2) which the agents inhabit; they can act as boundaries for our simulations, or fixed layers such as roads and houses that provide a means for agents to move from A to B and a place to live. Aggregate spatial data also allow for model validation (as discussed in Chapter 10): for example, are the land-use patterns we see emerging from a model of urban growth matching that of reality? If they do, it provides us with an independent test of the micro-level processes encoded within the model.

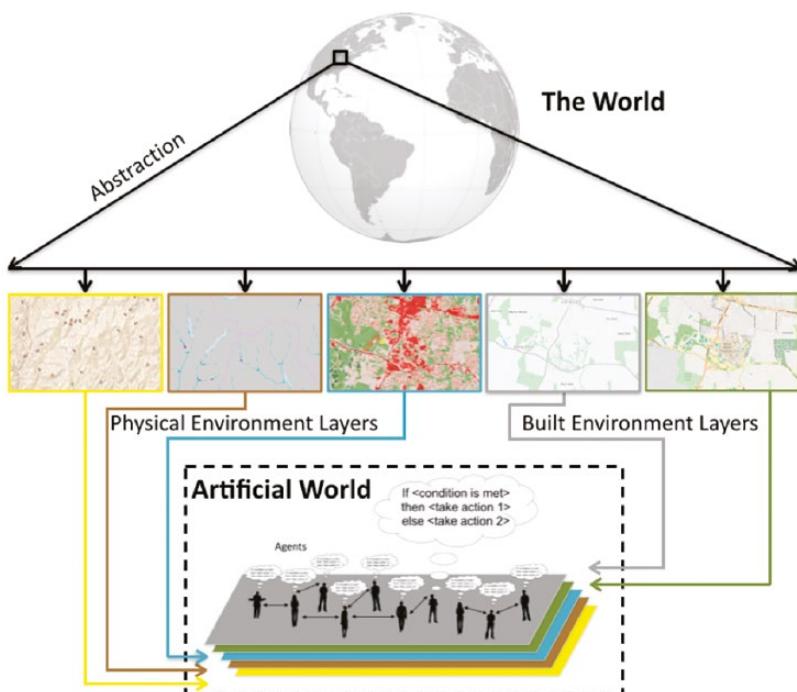


Figure 6.1 Abstracting from the real world to a series of layers to be used in the artificial world upon which the agent-based model is based

In this chapter we sketch out the considerations needed to be taken when choosing to develop a spatially explicit agent-based model (Section 6.2), before discussing the various toolkits which allow for the integration of GIS data (Section 6.3). How to integrate GIS and agent-based models is presented in Section 6.4; Sections 6.4.1 and 6.4.2 respectively discuss and demonstrate how to integrate both raster and vector data into agent-based models. A brief discussion is presented in Section 6.6.

Box 6.1 Space in agent-based models

Space within agent-based models serves two purposes: firstly, it contains the agents; and secondly, it defines the spatial relationships between the agents and controls their movement. Spatial data allows us to:

- Document the macro-phenomena.
- Inform micro-level process modelling – drivers of change (e.g. calculating accessibility indices' impact on house prices or analysis of land-use histories).
- Derive maps of the agents' environments (e.g. of physical networks such as roads for the agents to inhabit).
- Derive demographic variables for agent populations.
- Use macro-data for model validation, thus providing an independent test of the micro-level processes encoded in the model.

These topics will be addressed within this chapter.

6.2 Coupling and Embedding GIS and Agent-Based Models

The question faced by many modellers is how to integrate geographical data into models. A drawback of many traditional GIS platforms is that they are not capable of representing continuous data over time and space. This has led to modellers either linking (coupling) GIS and agent-based modelling, or embedding GIS into the agent-based model (see Crooks and Castle, 2012, for a detailed discussion). Coupling can be broadly defined as the linkage of two stand-alone systems by data transfer. Westervelt (2002) identifies three coupling approaches (loose, moderate and tight) that can be used with GIS and agent-based modelling. Loose coupling involves the asynchronous operation of functions within each system, with data exchanged between systems in the form of files. For example, the GIS might be used to prepare inputs which are then passed to the modelling system. After execution the results of the model are returned to the GIS for display and analysis (e.g. Crooks, 2010). This approach requires the GIS and modelling system

to understand the same data format (e.g. ESRI shapefiles). At the other extreme is tight coupling, which can be characterised by the simultaneous operation of systems allowing direct inter-system communication during the program execution (e.g. Leavesley et al., 1996; Benenson et al., 2005). For example, standards such as Microsoft's Component Object Model and .NET allow a single script to invoke commands from both systems (Ungerer and Goodchild, 2002). In the middle is moderate coupling. Here, remote procedures call and share database access between the GIS and the modelling system, allowing indirect communication between the systems (e.g. Harper et al., 2002). A summary of the pros and cons of different coupling approaches is presented in Table 6.1; readers wishing to find out more should see Westervelt (2002) for a detailed review.

Traditionally, coupling has often been the preferred approach for linking GIS and modelling systems (with loose coupling being the predominant approach).¹ However, this has tended to result in very specialised and isolated solutions, which have prevented the standardisation of general and generic linkage. An alternative to coupling is to embed or to integrate the required functionality of either the GIS or modelling system within the dominant system, using its underlying programming language (Maguire, 2005). The final system is either referred to as *GIS-centric* or *modelling-centric*, depending on which system is dominant. In both instances, the GIS tools or modelling capabilities can be executed by calling functions from the dominant system, usually through a graphical user interface (GUI). Compared to coupling, an embedded or integrated system will appear seamless to a user (Maguire, 1995).

Table 6.1 Comparison of coupling approaches

Objective and explanation	Loose	Moderate	Close/tight
<i>Integration speed:</i> The programmer time involved in linking the programmes	Fast	Medium	Slow
<i>Programmer expertise:</i> Required level of software development expertise	Low	High	Medium
<i>Execution speed:</i> How rapidly does the integrated software execute?	Slow	Medium	Fast
<i>Simultaneous execution:</i> Can components of the system run simultaneously and communicate with one another? Can the components operate on separate platforms?	Low	Low	High
<i>Debugging:</i> How difficult is it to locate execution errors in the linked system?	Easy	Moderate	Hard

Source: adapted from Westervelt (2002) and Crooks and Castle (2012)

¹ Loose coupling is the approach we take in this book and is also the norm in many agent-based modelling toolkits.

Interest in modelling-centric systems has increased considerably over recent years, predominantly due to the development of modelling toolkits with scripting capabilities that do not require advanced computer programming skills (as discussed in Section 6.3; for more details, see Castle and Crooks, 2006; Gilbert and Banks, 2002). Often the modelling toolkit can access GIS functions, such as data management and visualisation capabilities, from a GIS software library. For example, the MASON toolkit (see Section 6.3.2) exploits functions from GeoTools (a Java GIS software library) for importing and exporting data, Java Topology Suite for data manipulation, and its own GUI for visualisation. The toolkit itself maintains the agents and environment (i.e. their attributes), using identity relationships for communication between the different systems. Functions available from GIS software libraries reduce the development time of a model, and are likely to be more efficient because they have been developed over many years with attention to efficiency. Additionally, the use of standard GIS tools for spatial analysis improves the functional transparency of a model, as it makes use of well-known and well-understood algorithms (Castle and Crooks, 2006).

Conversely, the GIS-centric approach is an attractive alternative; not least because the large user base of GIS expands the potential user base for the final model. Analogous to the modelling-centric approach, GIS-centric integration can be carried out using software libraries of modelling functions accessed through the GIS interface. While there are many examples of modelling systems integrated within commercial GIS – including the Consequences Assessment Tool Set (Kaul et al., 2004) system which was designed for emergency response planning; the Hazard Prediction and Assessment Capability (Defense Threat Reduction Agency, 2001) system, for predicting the effect of hazardous material releases into the atmosphere; and the NatureServe Vista (2016) system, for land-use and conservation planners – there are few GIS-centric implementations from an ABM perspective. One such example is Agent Analyst for ArcGIS (see Johnston, 2013).

6.3 Tools for Constructing and Developing Agent-Based Models

Before discussing ways of integrating GIS into agent-based models, we discuss the range of tools available for building geographically explicit agent-based models. Traditionally the building of agent-based models required development from scratch using conventional programming languages (Gilbert and Banks, 2002) such as C++, Java and Python.² While programming from the ground up allows complete control over every aspect of the agent-based model, this can be

² For more on creating agent-based models in Python, see <https://github.com/projectmesa/mesa>

a time-consuming option unless the researcher is an experienced programmer. Moreover, common elements such as graphics libraries, algorithms and analysis tools must be repeatedly reimplemented by developers working on different models. Furthermore, this approach creates a boundary for researchers who want to build agent-based models but do not possess the necessary programming skills (Railsback et al., 2006). However, over the years several agent-based modelling toolkits have been developed which allow for more ease with respect to the development of agent-based models (especially in the case of NetLogo). These toolkits reduce the burden that modellers face in programming parts of a simulation that are not content-specific (e.g. GUIs, data import and export, visualisation/display of the model). Toolkits also increase the reliability and efficiency of the model, because complex parts have often been created and optimised by professional developers as standardised modelling functions (Castle and Crooks, 2006).

However, there are limitations to using toolkits to develop agent-based models. For example, a substantial amount of effort is required to understand how to design and implement a model in some toolkits; the programming code of demonstration models or models produced by other researchers can be difficult to understand, poorly documented or be intended for another purpose; a modeller will have to learn or already have an understanding of the programming language required to use the toolkit (e.g. Java, C); and the desired/required functionality may not be present, although additional tools might be available from the user community or from other software libraries. Benenson et al. (2005) also note that toolkit users are plagued by the fear of discovering that a particular function cannot be used, will conflict or is incompatible with another part of the model late in the development process.

In this section we present an overview of a selection of open source ABM toolkits that have the capability to process spatial (GIS) data (see also Table 6.2). Projects' websites are listed in order for readers to explore the current state of development of each toolkit. The rationale for choosing open source toolkits is that source code is published and made available to the public, enabling anyone to copy, modify and redistribute the system without paying royalties or fees. This is particularly useful for verifying and validating a model (see Crooks et al., 2008). Readers interested in other platforms or guidelines for choosing an agent-based modelling toolkit are referred to reviews by Abar et al. (2017), Castle and Crooks (2006), Kravari and Bassiliades (2015), Nikolai and Madey (2009), Railsback et al. (2006) and Robertson (2005). A comprehensive list of agent-based modelling toolkits can be found on Wikipedia.³

³ See https://en.wikipedia.org/wiki/Comparison_of_agent-based_modeling_software

Table 6.2 A selection of open-source agent-based modelling toolkits for creating geographically explicit models

	Swarm	MASON	Repast	NetLogo	GAMA
Developers	Santa Fe Institute/ SWARM Development Group, USA	Evolutionary Computation Laboratory and Center for Social Complexity, George Mason University, USA	University of Chicago, Department of Social Science Research Computing and Argonne National Laboratory, USA	Center for Connected Learning and Computer-Based Modeling, Northwestern University, USA	UMMISCO, France
Date of inception Website	1996 http://www.swarm.org/	2003 http://cs.gmu.edu/~ecilab/projects/mason	2000 https://repast.github.io/	1999 http://ccl.northwestern.edu/netlogo	2007 http://gamma-platform.org
Implementation language(s)	Objective-C/Java	Java	Java, Microsoft .Net Python, Groovy, ReLogo	NetLogo scripting (a derivative of Logo)	Proprietary scripting; GAMA Modelling Language
Required programming experience	Strong	Strong	Medium to strong	Basic	Basic to medium
Integrated GIS functionality	Yes (e.g. Kenge GIS library for Raster data; see Box, 2001)	Yes	Yes	Yes	Yes
Integrated charting/ graphing/statistics	Yes (e.g. R and S-plus statistical Packages)	Yes (e.g. wrappers for JFreeChart)	Yes	Yes	Yes
Availability of demonstration models	Yes	Yes	Yes	Yes	Yes
Tutorials/how-to documentation	http://www.swarm.org/wiki/	http://cs.gmu.edu/~ecilab/projects/mason/docs/	https://repast.github.io/docs.html	https://ccl.northwestern.edu/netlogo/resources.shtml	http://gamma-platform.org/edu/netlogo/resources.shtml
Additional Information	Mirar et al. (1996)	D-MASON ^a GeoMASON ^b	Useful weblog, ^c Agent Analyst ^d	Wilensky and Rand (2015) NetLogo-R extension ^e and a selection of GIS examples ^f	GAMA GitHub page https://github.com/gama-platform

Adapted and extended from Parker et al. (2001) and Castle and Crooks (2006).

^awww.dimason.org

^b<http://cs.gmu.edu/~ecilab/projects/mason/extensions/geomason/>

^c<http://crimesim.blogspot.com/>

^d<http://resources.arcgis.com/en/help/agent-analyst/>

^e<http://r-ext.sourceforge.net/>

^f<http://www.gisagents.org/search/label/NetLogo>

6.3.1 Swarm

Swarm could be classed as the original agent-based modelling toolkit, designed specifically for the development of simulations of complex adaptive systems (Swarm, 2016). Inspired by artificial life, Swarm was designed to study biological systems and infer mechanisms observable in biological phenomena (Minar et al., 1996). In addition to modelling biological systems such as fish (e.g. Railsback and Harvey, 2002), Swarm has been used to develop models for anthropology, computer science and ecological, economic, geographical and political science purposes (e.g. Deadman and Schlager, 2002; Johnson, 2002; Lim et al., 2002; Polhill et al., 2007). Useful examples of spatially explicit models include the simulation of pedestrians in the urban centres (Haklay et al., 2001) and the examination of crowd congestion at London's Notting Hill carnival (Batty et al., 2003).

6.3.2 MASON

MASON (Multi Agent Simulation Of Neighbourhood) is developed by the Evolutionary Computation Laboratory (ECLab) and the Center for Social Complexity at George Mason University (see Luke et al., 2005). Core functionality includes dynamic charting (e.g. histograms, line graphs) and model output during a simulation. It is supported by GeoMASON (Sullivan et al., 2010) which allows GIS vector and raster data to be imported and exported. MASON has a comprehensive set of technical documents and well-commented Javadocs. MASON's how-to documentation, demonstration models and several publications detailing the implementation and/or application of MASON are available for a prospective modeller to evaluate the system further (see MASON, 2016). Examples of spatially explicit models utilising MASON's GIS functionally are shown in Figure 6.2.⁴ Spatial applications of MASON include exploring disease spread (Crooks and Hailegiorgis, 2014), evacuations during humanitarian crises (Wise, 2014), conflict between herds-men and farmers in East Africa (Kennedy et al., 2010), movement across national borders (Łatek et al., 2012), to name but a few. More recently, a distributed version of Mason (D-MASON; Cordasco et al., 2013) has been created which allows MASON models to be run over cluster and cloud-computing architectures.

6.3.3 Repast

Repast (Recursive Porous Agent Simulation Toolkit) was originally developed at the University of Chicago, and is currently maintained by Argonne National Laboratory. Earlier incarnations of Repast catered for the implementation of models in three programming languages: Python (RepastPy); Java (RepastJ); and

⁴ These models and many more are available at <https://github.com/eclab/mason>

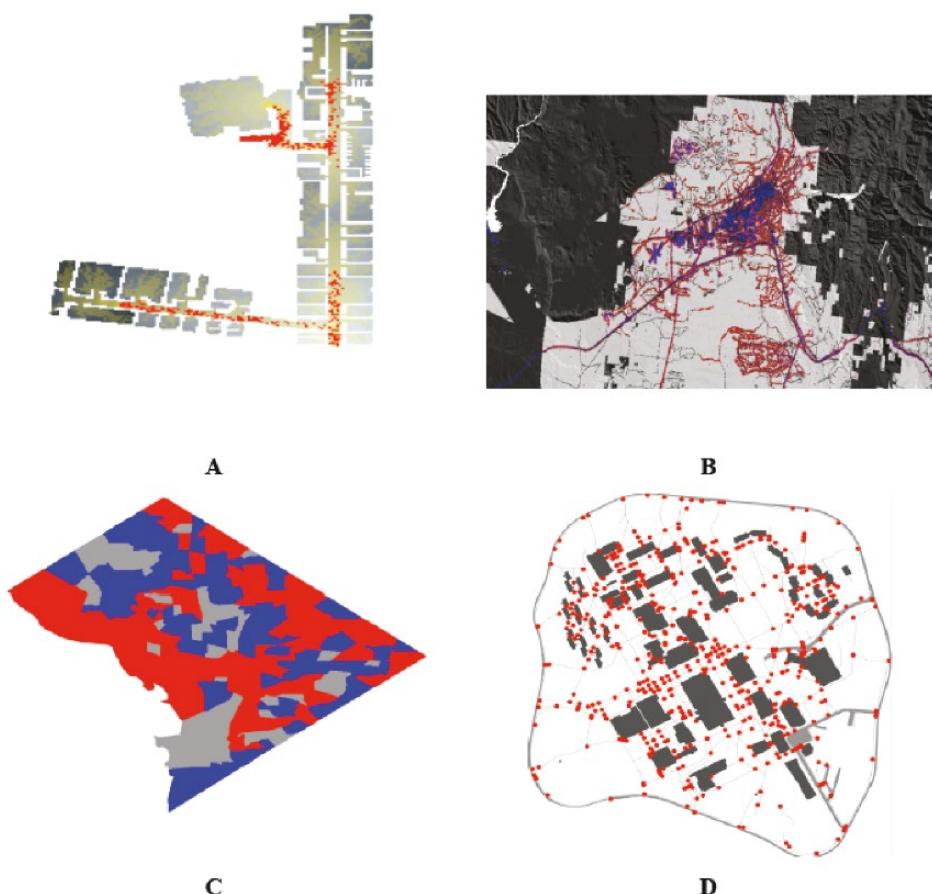


Figure 6.2 A selection of MASON spatial models: (A) agents (red) exiting a building based on raster data and the resulting trails (yellow); (B) an urban growth model where red areas represent new developments; (C) a Schelling type of model using census areas in Washington, DC as its spatial environment; (D) agents (red circles) moving along on sidewalks (grey lines)

Microsoft.Net (Repast.Net) (see Collier and North, 2004; Vos and North, 2004; North et al., 2006, for more details, and for a review of their GIS functionality, see Crooks, 2007b). These earlier versions have been superseded by Repast Simphony (North et al., 2013), which provides all the core functionality of previous versions but allows models to be developed in several ways including the ReLogo (a dialect of Logo; Ozik et al., 2013), point-and-click statecharts (Ozik et al., 2015), Groovy or Java.

The Repast development team have provided a series of articles on Repast Simphony. The architecture and core functionality are introduced by North

et al. (2005a), and the development environment is discussed by Howe et al. (2006). The storage, display and behaviour/interaction of agents, as well as features for data analysis (i.e. via the integration of the R statistics package) and presentation of models within Repast Simphony are outlined by North et al. (2005b). Within Repast Simphony it is possible to embed spatially explicit agent-based models directly into a three-dimensional GIS display. For this Repast Simphony provides methods to directly visualise agent-based models on NASA's virtual globe – WorldWind. This interactive 3D GIS display allows one to visualise agents with satellite imagery, elevated terrain and other scientific data sets as shown in Figure 6.3. Repast Simphony also supports the importation of NetLogo models into the Repast framework (Ozik et al., 2013; North et al., 2013). Such functionality aims to allow for rapid prototyping of agent-based models by first building simple agent-based models in NetLogo and, once satisfied with their basic functionality, migrating and extending them in Repast Simphony (for a comparison of NetLogo and ReLogo, see Lytinen and Railsback, 2012). Repast Simphony also supports high-performance distributed computing, via Repast for High Performance Computing (Repast HPC; see Collier and North, 2013) and has an extension called Agent Analyst that allows users to create, edit and run Repast models from within ArcGIS (see Johnston, 2013). Useful examples of spatially explicit models created using Repast include the study of segregation, and residential and firm location (Crooks, 2006), residential dynamics (Jackson et al., 2008), urban regeneration (Jordan et al., 2014), crime (Malleson et al., 2010), land-use change (Deadman et al., 2004), pedestrian evacuation (Castle, 2007a) and disaster management (Mysore et al., 2006).

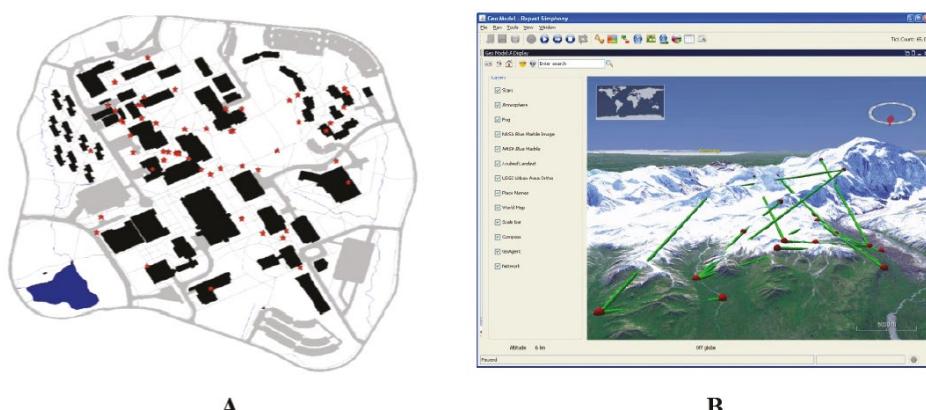


Figure 6.3 Examples of vector agent-based models in Simphony: (A) agents (red stars) moving along on sidewalks (grey lines); (B) an agent-based model overlaid on NASA WorldWind (Repast, 2016)

6.3.4 NetLogo

NetLogo (Wilensky, 1999)⁵ was developed at the Center for Connected Learning and Computer-Based Modeling at Northwestern University and uses a dialect of the Logo language to create agent-based models (NetLogo itself is written in Scala and Java). NetLogo has been used to develop applications in disciplines varying from biology and physics to the social sciences (see Wilensky and Rand, 2015). It has extensive how-to documentation/tutorials and demonstration models which are available from its website, and functionality can be extended through application programming interfaces. For example, NetLogo also has an R extension allowing for statistical analysis of model structure and dynamics (Thiele and Grimm, 2010). Another interesting feature of NetLogo is that it allows for multi-level modelling (via LevelSpace) by being able to connect several models together. For example, if one had a model of population growth, another on food projection and a third on weather, it would be possible to explore how these three systems impact each other (i.e. how bad weather impacts food production and how this impacts on population growth). Moreover, models developed in NetLogo can be run on the Web via NetLogoWeb. NetLogo is simple to use and it is possible to import both raster (in the form of .asc files) and vector data (shapefiles). This ability opens up a range of possibilities for the easy creation of spatial agent-based models, as shown in Figure 6.4.

For example, for the studying of basic concepts of water flow and surface erosion as shown in Figure 6.4A a raster file of surface elevation is loaded into a NetLogo model where the agents follow the surface to lower elevations. Such functionality potentially lowers the barrier to linking agent-based models and GIS for non-expert programmers. NetLogo models can also be viewed in a 3D environment and 3D surfaces can also be incorporated in such models as shown at the bottom of Figure 6.4A. Vector data can also be imported and used as a foundation for models. For example, lines can act as sidewalks or roads for agents to navigate urban environments (Figure 6.4B, D), and polygons can act as cells for a segregation type of model (Figure 6.4C). Useful examples of spatially explicit models created using NetLogo include the study of gentrification (Torrens and Nara, 2007), residential housing demand (Fontaine and Rounsevell, 2009), tourism (Orsi and Geneletti, 2016) and the reimplementation of Axtell et al.'s (2002) Artificial Anasazi model by Janssen (2009).

⁵ Throughout this book we use NetLogo (Wilensky, 1999) to highlight agent-based modelling principles, along with example models and the use of geographical information in such models. We have selected NetLogo for its ease of use and rapid prototyping capability. We would like to make readers aware that while we do not prefer it over other toolkits (we actually use several ourselves), NetLogo is an excellent tool for teaching and outlining the basics of this book's subject matter.

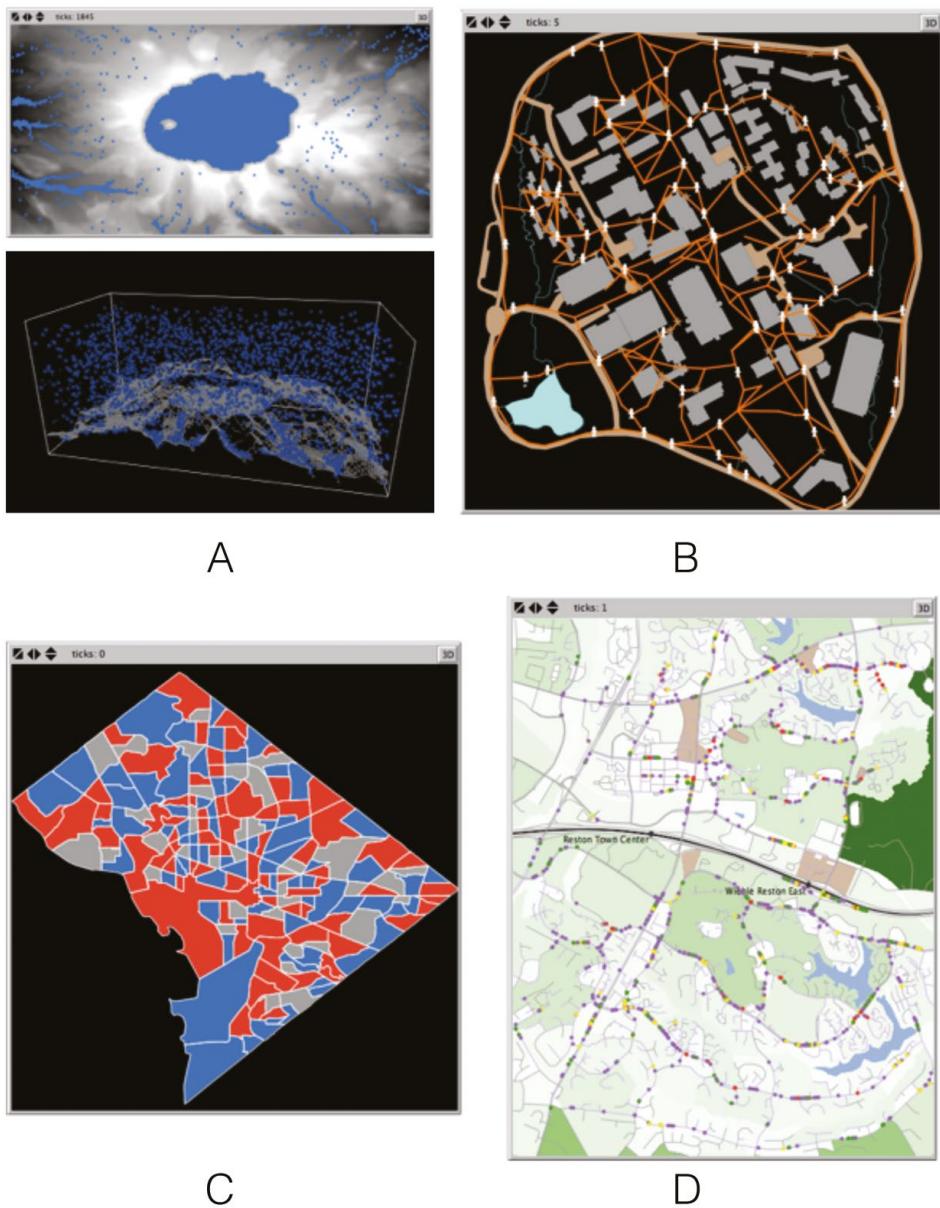


Figure 6.4 A selection of geographically explicit agent-based models utilising NetLogo: (A) rain (blue) falls and flows to a lower elevation based on a digital elevation model captured in 2 and 3D; (B) agents (white) moving along on sidewalks (orange); (C) Schelling type model using census areas in Washington, DC as their spatial environment; (D) commuting along a road network

6.3.5 GAMA

GAMA (GIS Agent-Based Modelling Architecture) is the only platform we review here that was specifically designed for the development of spatially explicit models ranging from hundreds to millions of agents (Taillandier et al., 2016). It was developed by several teams under the direction of the Unit for Mathematical and Computer Modelling of Complex Systems (UMMISCO), France (Amouroux et al., 2007). Its intention is to allow for the simple creation of models (akin to NetLogo) but with the ability to carry out experiments and simulations as in MASON and Repast (Grignard et al., 2013). At the time of its inception, it was noted that existing toolkits had several weaknesses with respect to creating geographically explicit models – complex programming for new modellers when using Swarm, the lack of GIS support in MASON and the inability of NetLogo to have complex models – which GAMA aimed to overcome (Amouroux et al., 2007).

GAMA has its own domain-specific language, Gama Modelling Language (GAML), which is coded in Java and its application is based on the rich client platform architecture provided by Eclipse. It allows for the importation of shapefiles and OpenStreetMap data along with grid and 3D files (e.g. .3ds file format)

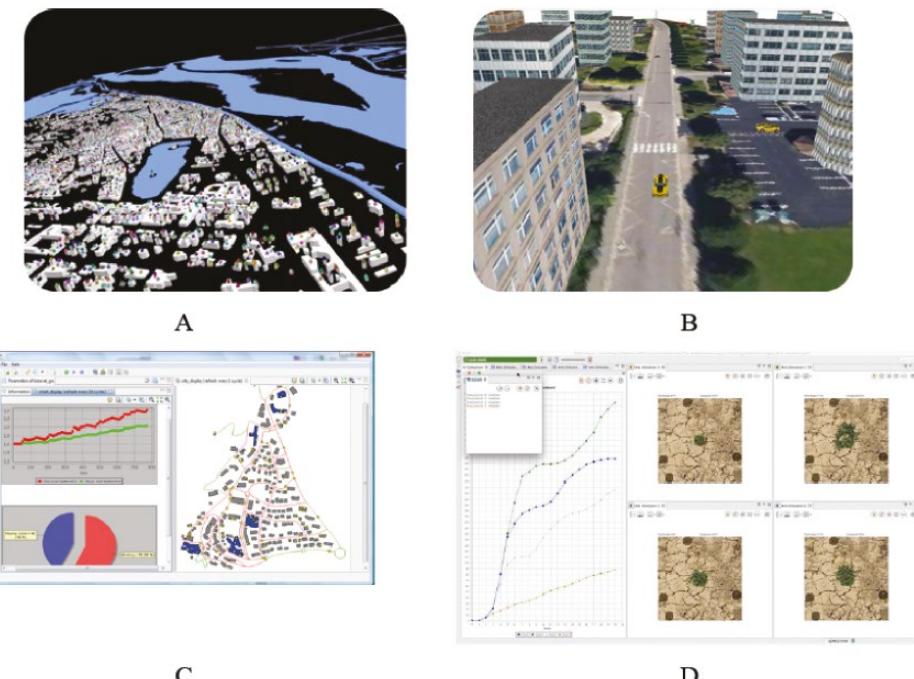


Figure 6.5 GAMA platform: (A, B) advanced visualisation of simulations; (C) built-in charting and functions; (D) user interface (GAMA, 2016)

for the creation of multi-level, highly visual and geographically explicit models in 2D and 3D as shown in Figure 6.5. Similar to other platforms, it has built-in charting and statistical functions including the Fuzzy-Kappa Simulation (van Vliet et al., 2013) and an integrated beliefs–desires–intentions cognitive architecture that others can use if they wish (Caillou et al., 2017). Example spatially explicit applications utilizing GAMA include farming (Taillandier et al., 2012), 2D and 3D pedestrian evacuation (Nguyen et al., 2011; Macatulad, 2014), traffic simulation (Taillandier, 2014), urban growth (Taillandier et al., 2016), land-use planning (Caillou et al., 2017), fire spread (Shaham and Benenson, 2018) and river sedimentation (Grignard, 2016).

6.4 Integrating GIS Data into Agent-Based Models

Space in agent-based models can range from abstract, as in the Schelling model first introduced in Chapter 2, to geographically explicit either through the use of vector or raster data structures. Traditionally many researchers focused on cell-based models, while more recently vector-based models have emerged, as well as combinations of the two. For example, in the cholera model of Crooks and Hailegiorgis (2014), raster data was used to represent the physical environment such as the terrain, and vector data (i.e. roads) was used for moving from one location to another (this model will be discussed in more detail in Section 8.4).

The way in which space is represented within a model will depend on the data model used (e.g. raster or vector). Typically, models whose space is generated from raster data are termed *discrete*; the movements of agents are restricted to a discrete number of cells. Geometry can be represented only coarsely, and neighbourhoods are often based on cell neighbours (e.g. Moore or von Neumann neighbourhoods; see Section 11.2.1). Those that are created using vector data are usually termed *continuous* because the agents' locations, and those of other objects, can take any position (within the bounds of precision offered by floating-point numbers). This allows for a more precise representation of geometry. Agents' neighbourhoods in continuous space are often based on a buffer of a specified distance around them.

Box 6.2 NetLogo's GIS extension

NetLogo's GIS extension provides support for reading both vector data (in the form of ESRI shapefiles) and raster data (in the form of ESRI ASCII grid files). The [NetLogo documentation](#) states: 'This extension adds GIS (Geographic Information Systems) support to NetLogo. It provides the ability to load vector GIS data (points, lines, and polygons), and raster GIS data (grids) into your model.'

As with all NetLogo extensions, the model needs to ‘activate’ it with the following line at the beginning of the model code:

```
extensions [gis]
```

For more information about the functionality provided by the GIS extension, refer to the documentation at <https://ccl.northwestern.edu/netlogo/docs/gis.html>. For detailed code examples, see Section 6.4.1 (raster) and Section 6.4.2 (vector).

6.4.1 Using Raster Data in NetLogo

In cellular space models, the world is divided up into a series of cells. This is analogous to the *raster* data model, which was discussed in Section 5.3.1. In a cellular space, agents’ movements are restricted to one of these discrete cells. The Schelling Segregation model (introduced in Chapter 2) is a well-known example of this type of model. Generally all cells (or ‘patches’ in NetLogo speak) are the same shape. Such a representation of space has similarities to that of cellular automata (CA) models (as discussed in Section 11.2.1). Indeed, hybrid models that have an environment of cells that are able to influence each other, as well as mobile agents who traverse the environment, are popular in fields such as land-use and land-cover change (Parker et al., 2003). In cell-based models, an agent’s neighbourhood is often calculated using Moore or von Neumann neighbourhoods, or variations of these which can be weighted to account for distance from the agent (e.g. White and Engelen, 1993). Code Example 6.4.1 illustrates the NetLogo commands that can be used to query the neighbourhood of patches or turtles. Note that the command `neighbors` returns the eight von Neumann neighbours and the command `neighbors4` returns the surrounding four Moore neighbours. For more complicated neighbourhoods that use a radius to search beyond the immediate surrounding patches, refer to the *Moore & Von Neumann Example* in the NetLogo Models Library.

Code Example 6.4.1 Finding turtles or patches within von Neumann (8) or Moore (4) neighbourhoods in NetLogo

```
; Examples of finding neighbours in NetLogo (these are either patch
; or turtle context)

;; Ask the eight von Neumann neighbouring patches to turn blue:
ask neighbors [ set pcolor blue ]

;; Ask all turtles on neighbouring patches to turn brown
ask turtles-on neighbors [ set pcolor brown ]
```

```
;; Repeat, but this time using the Moore neighbourhood
ask neighbors4 [ set pcolor blue ]
ask turtles-on neighbors4 [ set pcolor brown ]
```

This section will now briefly present an overview of two example raster-based models, before focusing on a particular example of a rainfall model in Section 6.4.1.

Raster Model Examples: Urban Growth and Pedestrian Modelling

Urban Growth. The SLEUTH model – Slope, Land-use (e.g. urban and non urban), Exclusion (where one can and cannot build), Urban Extent, Transportation (road network) and Hillshade – is a popular tool that has been used to model land-use change and urban growth. It is a CA model that has been used in over 30 applications both in the USA and around the world (see Clarke et al., 2006). The model takes multiple input data sets that capture five different coefficients (parameters) – dispersion, breed, spread, slope and road gravity – applied to four different growth rules – spontaneous growth, new spreading centres, edge growth and road-influenced growth. In this manner, the SLEUTH model can then be used to simulate urban growth.⁶

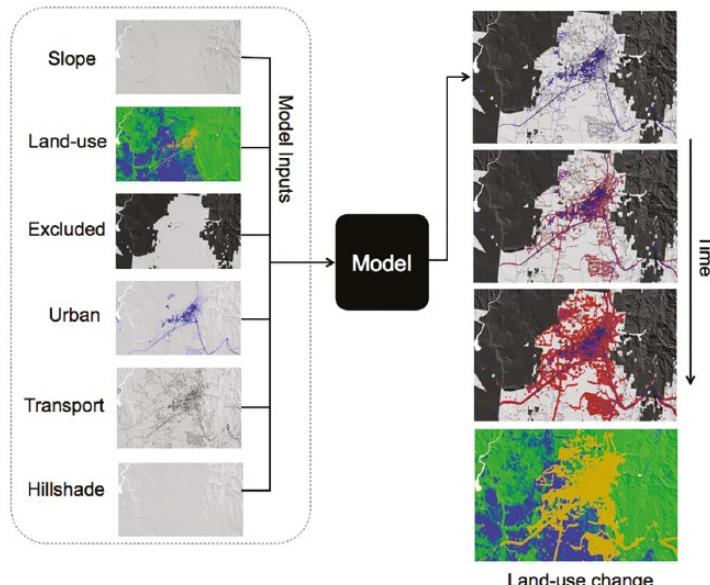


Figure 6.6 A SLEUTH-type model based on growth rules and coefficients as outlined in Clarke et al. (1997). In this example, the model is simulating the rates of land-use change in Santa Fe, New Mexico

⁶To find out more about the SLEUTH model or download it, see the Project Gigalopolis website: <http://ncgia.ucsb.edu/projects/gig/>

Figure 6.6 depicts the layers that can be used as inputs into a SLEUTH-type model and the outputs that the model can generate (Clarke et al., 1997). In this example, it explores the extent of future urban growth, shown from blue (the current urban extent) to red for the possible future urban extent (see Clarke et al., 1997, for more details).

The model presented in Figure 6.6 has been implemented using MASON (see Section 6.3.2). However, the interested reader can also refer to a NetLogo version of the same model (Figure 6.7). The code and data are available, in full, in the accompanying online resources.

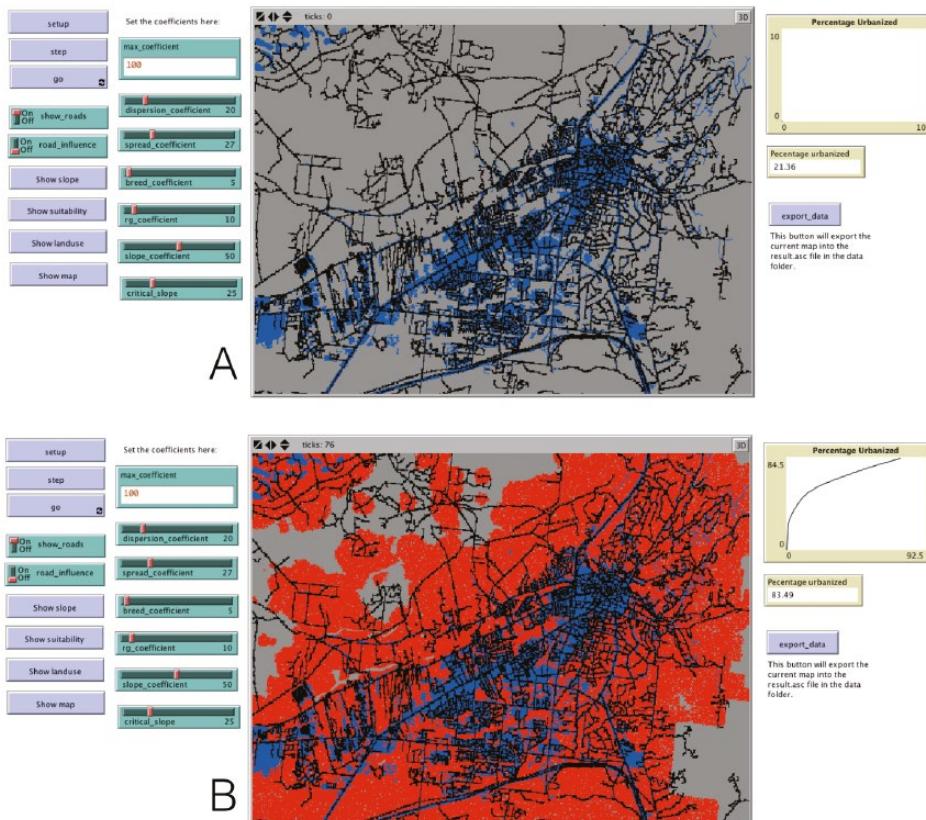


Figure 6.7 A basic SLEUTH-type model based on growth rules and coefficients driving the rates of land-use change in Santa Fe, New Mexico implemented in NetLogo: (A) initial conditions (blue cells); (B) urban growth (red cells).

Pedestrian Modelling. We now turn our attention to agent-based models of pedestrians that use a geographical space to represent direct agent interactions more closely. Models that use raster data form the basis for a wide range of phenomena where the movement of people plays a critical role (see Torrens, 2012). These include

crowd modelling (Henein and White, 2005), simulating disasters (van der Wal et al., 2017), exploring how people find out and search for food in times of crisis (Crooks and Wise, 2013), understanding how people move around town centres (Haklay et al., 2001), exploring the spread of diseases within refugee camps (Crooks and Hailegiorgis, 2014), understanding dynamics in enclosed spaces (Crooks et al., 2015a) and evaluating event safety (Batty et al., 2003; Johansson et al., 2012). It is possible to create simple agent-based models without the use of geographical data in order to experiment with the fundamental drivers of crowd dynamics. For example, Figure 6.8 shows a simple pedestrian model of agents trying to exit a room. This model is especially simple in that, rather than implementing complicated routing and navigation algorithms, or even using ‘social forces’ (Helbing and Molnár, 1995), this model employs a *cost surface* to help the agents to navigate to the exit. The values of the cost surface are lowest near the door (depicted as white in Figure 6.8A) and increase with distance. Thus the agents can simply move from cells with higher costs to those with lower costs (Castle, 2007a). Code Example 6.4.2 provides the code that moves the agents towards the door. It is then possible to develop ‘what if’ scenarios, such as: how would an increase, or decrease, in the size of the exit affect evacuation times? Figures 6.8B and C show this; a relatively small increase in the size of the door can have a substantial increase in the rate at which agents are able to leave. The code required to run this model is available, in full, in the accompanying online resources.

Code Example 6.4.2 Code to move agents down a cost surface, as illustrated by the model in Figure 6.8

```
ask turtles [
  ;; Decide which grid cell to move to
  ;; - 'elevation' is the value of the cost surface
  ;; - 'count turtles-here * 9999999' stops the agent from
  ;;   moving onto an occupied cell
  ;; The turtle wants to choose the neighbouring cell with the
  ;; lowest elevation
  let target min-one-of neighbors [
    elevation + (count turtles-here * 9999999)
  ]

  ;; If the free cell has a lower elevation than the current
  ;; cell, move to it. Otherwise don't do anything (stay still)
  if [elevation + (count turtles-here * 9999999)] of target <
    [elevation] of patch-here [
      face target
      move-to target
    ]
  ]
]
```

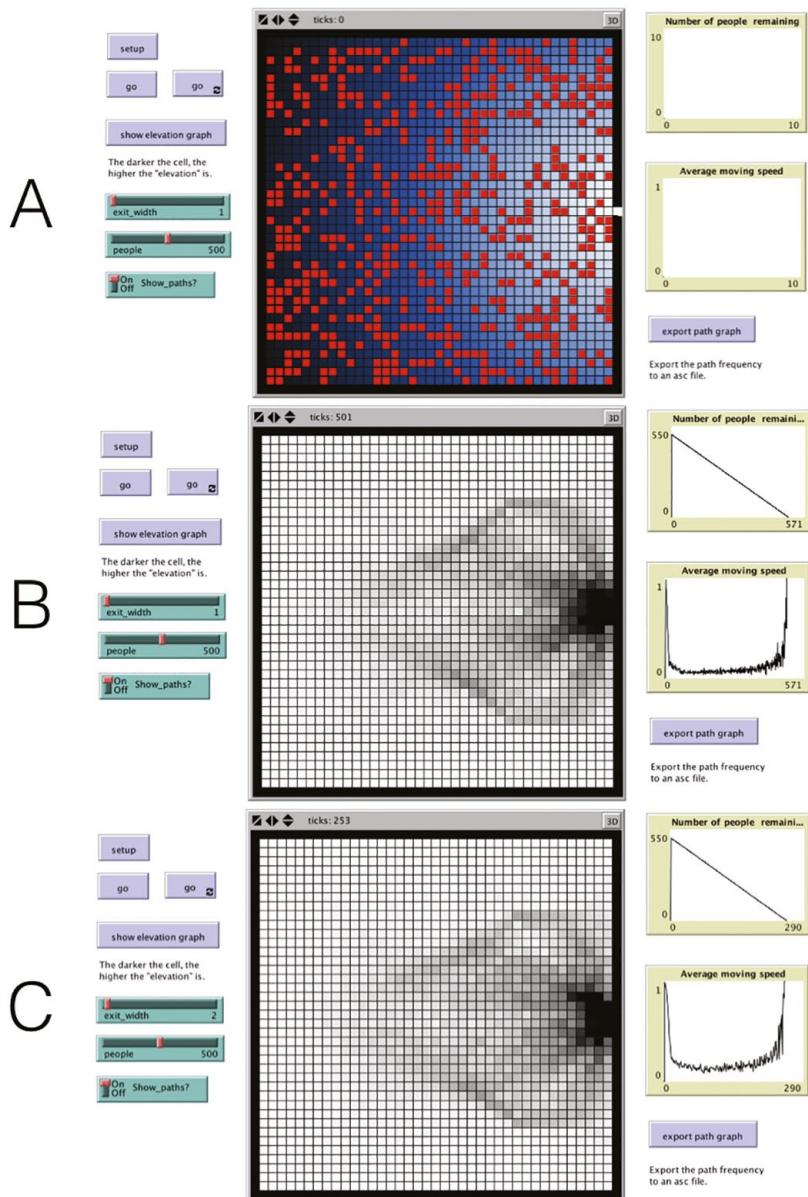


Figure 6.8 A simple pedestrian evacuation model where agents are exiting a room: (A) initial starting conditions where the red dots represent agents and the blue represent a cost surface; (B) pedestrian paths when the door is one cell wide; (C) pedestrian paths when the door is two cells wide

On occasion, however, real world data is required. Fortunately it is relatively straightforward to use NetLogo to include GIS data in a pedestrian model, as illustrated in Figure 6.9. In that example, an architect's CAD file of a building on the George Mason University Fairfax campus is used.

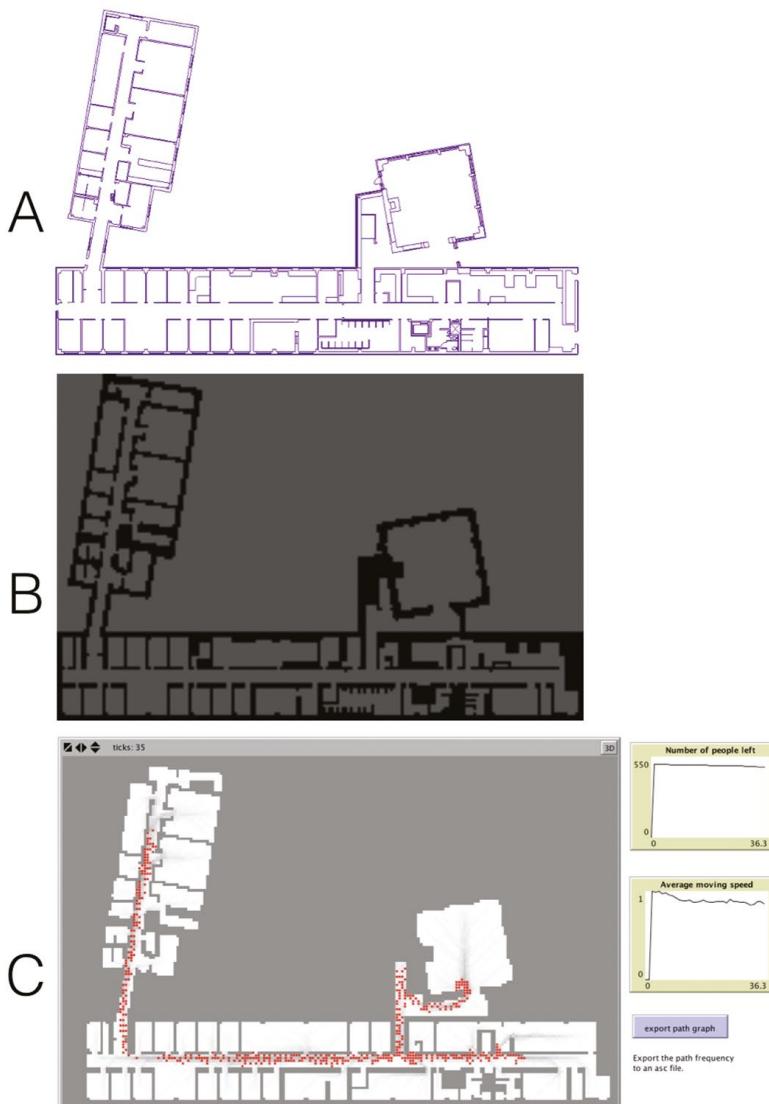


Figure 6.9 Simple pedestrian model: CAD floor plans of a building (A) are converted into a raster layer (B) and are used as the environment for the agent-based model; (C) shows the simulation running with agents (red) who are exiting the building and leaving behind walking traces (grey)

It is possible to georeference the building to its real world location and then rasterise the space so that each cell represents approximately 50 cm². This approximates the anthropomorphic dimensions of an individual (Pheasant and Haslegrave, 2006). Subsequently, the regular lattice structure can be used as the artificial world in a similar way to that of other pedestrian models (e.g. Batty et al. 2003). It is worth noting that a continuous (i.e. vector) space representation could also be used if so desired (see Castle, 2007b, for a discussion of the role of space within pedestrian models). Section 6.4.2 will demonstrate how vector data can be used as an alternative.

Once the spatial layout of the building has been incorporated, the building can be populated with agents. In this example the rules are relatively simple: once an alarm is activated, agents follow emergency signage or move to the nearest exit. Again, a cost surface is used. The simulation demonstrates how bottlenecks form at exits and how this causes agents to cluster around such obstacles.

Detailed Raster Example: Rainfall Model

The previous examples have shown what is possible with relatively simple pedestrian models. This section will work through an example in more detail: the Rainfall model as depicted in Figure 6.4. Although the ‘agents’ in this model represent drops of rain, rather than people, the techniques for loading and working with GIS data are applicable to human-focused applications as well. The model also demonstrates how agent-based models can be used to explore physical processes as well as social processes.

The study area is Crater Lake, a caldera lake in the Crater Lake National Park, Oregon. The data that are used to represent the area consist of a digital elevation model (DEM), that is, a raster map where each pixel represents a height value. The model demonstrates how rain can fall from the sky and flow to lower elevations. If the water reaches a point from which it can flow no further (e.g. a depression) it pools until the depression is full, at which point the water will flow again. For example, Figure 6.10 shows how Crater Lake fills up with water over time until a point when the rim is breached and the water can flow out of the lake and down the hillside. Furthermore, the model also has features to ensure the flow of water is working as intended – a form of *verification*, as discussed in detail in Section 10.2. These features allow the researcher to change the landscape to flat, cone and hill shapes, and check that the water behaves as expected in these simple environments.

Box 6.3 Exercise

Experiment with the Rainfall model, which is available in the accompanying material. If you change the landscape (MapType) to Flat, Cone and Hill, does the water behave as expected?

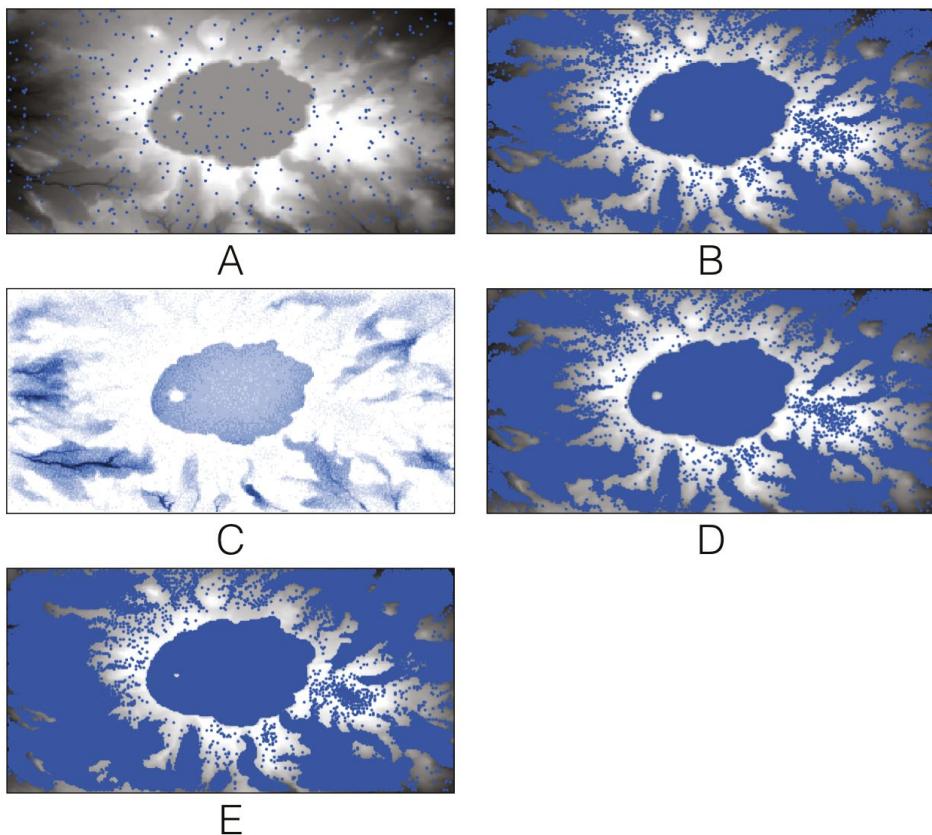


Figure 6.10 A simple hydrological model in which rain falls from the sky and flows downhill: (A) initial starting conditions where the blue dots represent agents (i.e. water) and the underlying digital elevation model; (B) over time rain accumulates in Crater Lake; (C) water depth at time step 500 and (D) the corresponding water; (E) Crater Lake has enough water for it to discharge

In the code examples below we highlight the parts that are relevant to using GIS specifically. For the full model code, refer to the accompanying resources. To begin with, the GIS extension NetLogo needs to be activated, using the following line at the beginning of the model code:

```
extensions [gis]
```

It is also necessary to define some global variables and some turtle and patch variables. Code Example 6.4.3 illustrates the required code. In particular, note the `elevation-dataset` global variable (this is used to store the raster GIS data) and the `elevation` patch variables (these are used to keep track of the initial elevation and current elevation once water starts to flow onto each patch).

Next it is necessary to read the DEM data. They are stored as an .asc file. This is a common raster format used by most GIS applications and was first introduced in Section 5.3.1. To load the DEM data into NetLogo it is first necessary to resize the model environment so that there is precisely one NetLogo patch per raster pixel. It is possible to find the number of rows and columns in a raster data set using a GIS, but the process is even simpler with .asc files. Recall from Section 5.3.1 (Figure 5.2) that it is possible to read .asc files using a text editor. The first few lines of the file provide some information about the rest of the data. For example, the first six lines of the `elevation.asc` file, which we will read here, contain the following:

ncols	285
nrows	143
xllcorner	-122.26638888878
yllcorner	42.855833333
cellsize	0.001111111111859
NODATA_value	-9999

The first two lines declare the number of columns and rows of pixels. Therefore the NetLogo environment must be resized so that it has exactly that many rows and pixels. In this example we would like the origin to remain in the centre of the world, hence it can be resized as follows:

```
resize-world -142 142 -71 71
```

Following this, the `envelope` needs to be configured so that it will map exactly on to the raster GIS data extent (refer to the [gis:set-world-envelope documentation](#)⁷ for more information).

```
gis:set-world-envelope gis:envelope-of elevation-dataset
```

Then the two `apply-raster` commands read the value of each raster cell and store this value in the patches' `elevation` and `initial-elevation` variables (to begin with the elevation and initial elevation are the same because the model has not simulated any water). In effect this creates the mapping between raster cells and NetLogo patches.

```
gis:apply-raster elevation-dataset elevation
gis:apply-raster elevation-dataset initial_elevation
```

⁷ <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:set-world-envelope>

Code Example 6.4.3 Global, patch, and turtle variables required by the Rainfall model

```
globals [
    elevation-dataset
    border          ; Keep the patches around the edge in a global variable
                    ; useful to keep a record of these
                    ; so we don't have to recalculate at each iteration:
    min-e           ; Minimum elevation
    max-e           ; Maximum elevation
    the-row         ; Used in export-data. It is the row being written
    list_of_rain_amount
]

patches-own [
    elevation
    initial_elevation
    elevation_change
    amount_rain     ; How many drops of rain here
]
]

turtles-own [
    soil            ; How much soil a raindrop is carrying
]
```

At this point, the raster data has been read in and the NetLogo environment has been configured to match the raster data. From this point on, working with the raster data is the same as working with any other NetLogo model. Turtles are created to represent water drops and they use the height of the surrounding patches to move downhill where appropriate. Code Example 6.4.4 illustrates how this has been accomplished in the Rainfall model; it has been adapted from the NetLogo Grand Canyon example under Earth Science in the Models Library. For further details, see the model code in full in the accompanying resources.

Code Example 6.4.4 An example of some turtle code that moves the turtles down a slope, taking account of whether water is pooling and raising the elevation of the land.

```
; NOTE: this procedure uses codes from the Grand Canyon library model
; with some modifications
to flow
    ; Choose the neighbouring patch with the lowest elevation
    let target min-one-of neighbors
        [ elevation + (count turtles-here * water-height) ]

    ; Move if the height is lower than my current position
    ; If no move is possible, then become a 'waters' type of
    ; agent to show that I am part of pooling water
    ifelse [elevation + (count turtles-here * water-height)] of target
```

```

< (elevation + ((count turtles-here - 0) * water-height))
[ face target
  move-to target ]
[ set breed waters ]
end

```

Box 6.4 Summary: Raster data in agent-based models

This section has highlighted how raster data can be used as a foundation for geographically explicit models. One of the advantages with raster data is that it fits well with the cell-based nature of many agent-based models. This is especially true of the direct relationship between raster *cells* and NetLogo *patches*. Raster data is also often widely available, especially remotely sensed data, and multiple raster layers can be combined to build rich models (as in the case of the SLEUTH example). In the case of pedestrian movement, cells can approximate the space occupied by an average person.

There are, of course, disadvantages with cell-based representations of space. The sizes of cells are often the same throughout the environment, which makes it difficult to represent land parcels of different sizes. Furthermore, the movement of agents on cell-based environments is limited to discrete parcels. This makes it difficult to represent very fine-scale movements. To overcome these issues it is possible to use a continuous space and vector data, which is the subject of the following section.

6.4.2 Using Vector Data in NetLogo

While many artificial worlds in agent-based models are based on raster data, they lack *geometric* detail (Batty, 2005). In order to capture geometric detail, vector data are required. There is an abundance of applications that make use of vector data to create artificial worlds. These include: the movement of pedestrians over streetscapes (Torrens, 2012); routing vehicles over a street network (Manley et al., 2014); responses to chemical attacks (Mysore et al., 2006) and disease outbreaks (Crooks and Hailegiorgis, 2014).

Although many of the toolkits discussed above do allow for the use of vector data, each will have its own particularities. In essence, most toolkits allow for the importation of ESRI shapefiles (i.e. points, lines and polygons). These were first introduced in Section 5.3.2. Recall that an ‘ESRI shapefile’ is actually a collection of different files that represent the geometries, attributes, projection, etc. The accompanying .dbf (database) file contains the attribute data associated with each

⁸ Note that when using a GIS, or an agent-based modelling toolkit such as NetLogo, the .dbf will be read automatically so the user may not even be aware that a ‘shapefile’ is actually made up of separate files.

area.⁸ These attributes can be used to populate the environment (e.g. by setting the land-cover type of a land parcel) or provide attributes to the agents (e.g. by assigning socio-demographic characteristics to agents based on where they live).

To explain this in more detail, consider the example of attempting to model community cohesion. The aim is to create a model that contains a number of agents in a small town, each of whom will have their own unique socio-demographic characteristics. The model can then allow the agents to move around the environment, interact and attempt to observe the processes that lead to (or detract from) community cohesion. In this example, the population of individuals might have been created using microsimulation and then stored as a shapefile that contains each person's age, income, ethnicity and education level. An agent-based modelling tool, such as NetLogo, can then read the shapefile and create agents who are located in the correct position in the town, and have the same features as those found in the shapefile (i.e. age, income, ethnicity and education level). The model can then begin to simulate the behaviour of the agents, whose behaviour can be determined in part by the values of their attributes. The following section will work through an example of a model that uses vector data in more detail.

Detailed Vector Example: Schelling and Polygons

Chapter 2 introduced Schelling's (1971) model of residential segregation and demonstrated its basic properties. It was also noted that the model has been used to study real world examples of segregation (Benenson et al., 2002; Jordan et al., 2014). In this vein, two variations of the model will be demonstrated here, with the main purpose of illustrating how vector GIS data can become part of a NetLogo model. The first example uses polygons to replace the regular cell structure that is typically associated with NetLogo models⁹ and the second (in Section 6.4.2) demonstrates how attributes in the GIS data can be used to instantiate agents.

Figure 6.12 shows how vector data can be used as a basis for a model, using census tracts for Washington, DC as the environment rather than the hypothetical one that Schelling originally used. The model logic is similar, though; agents have a preference for being surrounded by neighbours who are similar to themselves. Here, the neighbourhood is defined by adjacent polygons rather than the Moore neighbourhood (i.e. the eight cells that surround the agent). As in the original

⁹ The model here is not the first model to demonstrate the use of vector polygons instead of patches (see Flache and Hegselmann, 2001; Crooks, 2007a), but it is used here to highlight a model that can be instantiated using real world data (in this case census tracts for Washington, DC).

model, agents move to an unoccupied polygon if their preference is not being met. Another important difference is that the agents are not randomly placed in the artificial world. Rather their original locations are based on information from the shapefile. In this simple example, one agent is created per polygon.

In the remainder of this section the important parts of the model code will be highlighted. The model, entitled `Segregation DC 1.nlogo`, is available, in full, in the accompanying online resources. Many of the steps are similar to those of Section 6.4.1 where raster data were used to initialise the model.

Aside: Setting the Patch Size. In Section 6.4.1 it was demonstrated that the number of patches in the NetLogo environment needed to be the same as the number of cells in the underlying raster data. When using vector GIS data, there is not necessarily a direct link between the dimensions of the objects in the vector data and the sizes of the NetLogo patches. However, in some cases the patches might need to be equal to a particular geographic dimension, say 100 m^2 , relative to the size of the vector features. To ensure that this is the case, it is possible to load a vector file of known dimensions and then adjust the number of patches in the environment until they coincide with the dimensions of the vector data.

The `PatchSize` model, available in the accompanying online resources, illustrates how this can be accomplished. It reads a shapefile that has been created previously. The shapefile simply contains some overlapping squares with known side lengths (100 m, 500 m and 1 km). The number of patches, and the size of the patches, can then be configured manually to find the patch size that corresponds to the same dimensions as the squares in the GIS data. Figure 6.11 illustrates the input data and the model with patches that are the equivalent of 500 m^2 . The code configures the patch size to correspond to exactly 500 m^2 , which was established through trial and error:

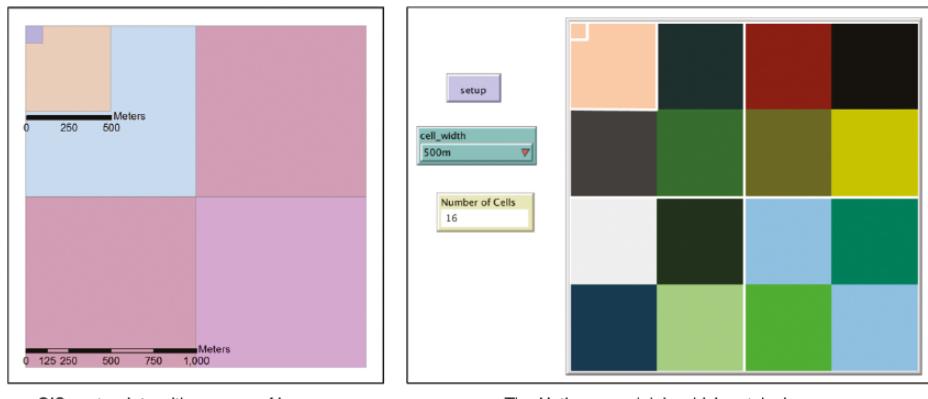
```
resize-world 0 3 0 3
set-patch-size 100
```

For 100 m^2 patches, the code would be:

```
resize-world 0 19 0 19
set-patch-size 20
```

Initialising the Model. The discussion now returns to the GIS-enabled segregation model (`Segregation DC 1.nlogo`). As with all GIS-enabled models in NetLogo, the first step is to activate the GIS extension:

```
extensions [gis]
```



GIS vector data with squares of known dimensions (100m, 500m and 1km)

The NetLogo model, in which patch sizes can be configured to match the GIS data

Figure 6.11 Setting the NetLogo patch size so that it corresponds with some vector GIS data

Then a global variable that stores the Washington, DC data needs to be defined (called dc-dataset) as well as some patch and turtle variables. Code Example 6.4.5 defines the required code.

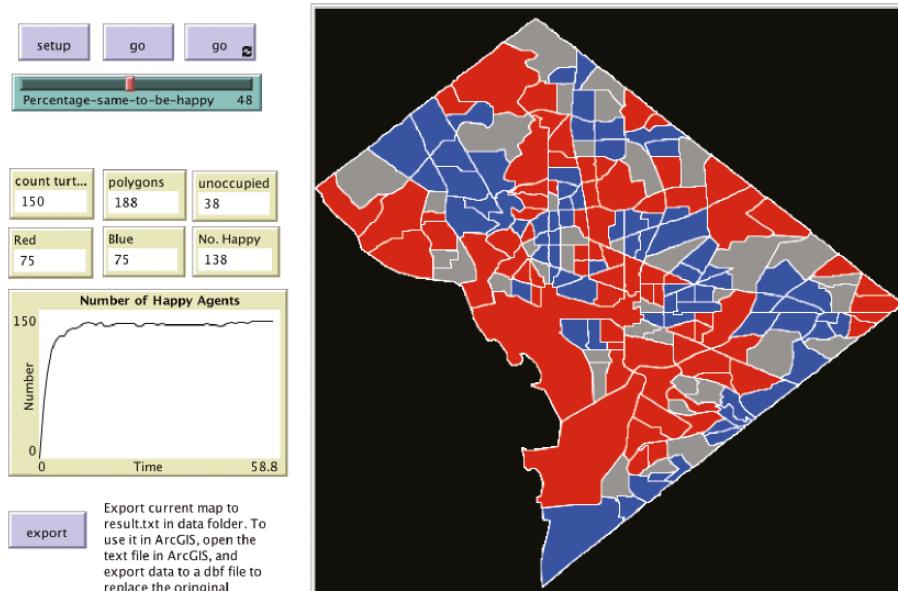


Figure 6.12 The Schelling segregation model implemented using polygons as agents

Reading the Data. As with the Rainfall model (Section 6.4.1), the first thing to do to initialise the model is to read the GIS data. Here, rather than reading an .asc file (that format was used to store the raster data), an ESRI shapefile is read:

```
set dc-dataset gis:load-dataset "data/DC.shp"
```

It is also necessary to set the extent of the world using the `gis:set-world-envelope` command, which maps the envelope of the NetLogo world to the GIS envelope, maintaining the same *x* and *y* scales (refer to the [gis:set-world-envelope documentation⁷](#) for more information).

```
gis:set-world-envelope gis:envelope-of dc-dataset
```

Drawing the Spatial Features. Once the data have been read, it is possible to draw a map. The code for this looks somewhat confusing, but is actually quite simple. Code Example 6.4.7 shows the code in full, but it will be explained step by step below. To begin with the `gis:feature-list-of` command provides a list that contains every feature (polygon) in the data set and the `foreach` loop is used to process each feature.

```
foreach gis:feature-list-of dc-dataset [
    feature ->
```

Code Example 6.4.5 The NetLogo code that defines the global, patch and turtle variables required by the GIS-enabled Schelling Segregation model

```
extensions [gis]

globals [ dc-dataset ] ;; This stores the GIS data

patches-own [
    ID          ;; Patch ID, identical to polygon ID
    popu        ;; Population
    mycolor     ;; Patch colour
    myneighbors ;; Neighbouring polygons' centroid patches
]

turtles-own [
    tcolor      ;; Turtle colour (red or blue)
    percentage-same ;; Percentage of same colour turtles on neighbours
    happy?      ;; Happy if neighbouring enough same colour agents
    tneighbors   ;; An agentset of its neighbour turtles
    rneighbors   ;; Number of red neighbours
    bneighbors   ;; Number of blue neighbours
    tneighborpolygons;; Neighbouring polygons' centroid patches
]
```

Once inside the loop, the first thing to do is tell NetLogo how we would like to refer to the objects that are being iterated over. Any word or character will do. Here the word `feature` is used, as it is polygon features that are being iterated over after all.

Box 6.5 foreach loops in NetLogo

In NetLogo version 6, the developers introduced a new syntax for `foreach` loops. In the new version, the programmer defines the name of the variable that they would like to use to refer to the object that they are iterating over. For example, in the following code, the word `feature` is used, but this could be anything (e.g. the character `x` is common).

```
foreach gis:feature-list-of dc-dataset [
  feature ->
  ;; .. do something with the feature here
]
```

Iterating over lists using commands such as `foreach` is something that many people will not have come across. The [NetLogo dictionary](#) has full details of the `foreach` loop.¹⁰ We also recommend reading the sections in the NetLogo Programming Guide on *Iterating over Lists* and *Task Primitives*. The documentation for `gis:feature-list-of`¹¹ and similar functions is also helpful.

Note that in some cases, you might see the strange characters `?1` in a loop. These occur when models that were created in NetLogo version 5 are converted to version 6. The converter does not know what variable name the programmer would like to use, so chooses `?1`.

After entering the loop, the following code will read the value of the "SOC" attribute for each feature and set its colour accordingly. It uses the `gis:property-value` command to retrieve the value of the feature. That command requires two pieces of information: the object from which to retrieve the data from (`feature`) and the name of the attribute to get ("SOC" in this case).

¹⁰ <https://ccl.northwestern.edu/netlogo/docs/dict/foreach.html>

¹¹ <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:feature-list-of>

Note that "SOC" is a feature in the shapefile that has been used to store the initial colour of each polygon.

```
if gis:property-value feature "SOC" = "RED" [
  gis:set-drawing-color red
  gis:fill feature 2.0
]
```

The `gis:set-drawing-color` command determines the colour that will be used to draw the feature, and `gis:fill` then fills the layer with the colour previously specified by `gis:set-drawing-color`. Note that `gis:fill` requires two pieces of information: the object to fill (feature tells NetLogo to colour the current feature in the list) and the line thickness around the edges (2.0 in this case). Again, refer to the documentation for `gis:set-drawing-color`¹² and `gis:fill`¹³ for more information about how the features are actually drawn.

After iterating over all features, the drawing colour is returned to white and the whole data set can be drawn:

```
gis:set-drawing-color white
gis:draw dc-dataset 1
```

Box 6.6 Calculating polygon neighbours

As an aside, it is also necessary to tell each polygon who its neighbours are. In this example, the neighbours are calculated separately in a GIS (using the Polygon Neighbors function in ArcGIS in this case) and stored in a plain text (.txt) file. Code Example 6.4.6 demonstrates the code to read the file and store each polygon's neighbours.

Code Example 6.4.6 Code to read the neighbours of each polygon from an accompanying .txt file

```
; Find neighbours of each polygon using a txt file produced by
;; ArcGIS Polygon Neighbours function
file-close
file-open "data/neighbors.txt"
```

¹² <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:set-drawing-color>

¹³ <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:fill>

```

while [not file-at-end?] [
  let x file-read let y file-read
  ask patches with [ID = x] [
    set myneighbors ( patch-set myneighbors patches with [ID = y] )
  ]
]
file-close

```

At this point, NetLogo has everything that it needs in order to display the polygons stored in the shapefile. However, the actual information associated with each feature has not yet been stored by the model, it has only been used to draw the polygons. We next demonstrate how to store the required information from each polygon for use in the model.

Code Example 6.4.7 NetLogo code to read a shapefile and draw a map of the features, colouring them depending on the value of an attribute

```

;; Go over each feature (polygon) in the dataset using a 'foreach' loop
foreach gis:feature-list-of dc-dataset [
  ;; When inside the 'foreach' loop, tell NetLogo how we would
  ;; like to refer to each feature. Here we will use the word 'feature',
  ;; but any word or character can be used
  feature ->

  ;; The following 3 lines set the colour of the polygon to red, blue
  ;; or grey, depending on the value of the 'SOC' attribute in the shapefile
  if gis:property-value feature "SOC" = "RED" [
    gis:set-drawing-color red
    gis:fill feature 2.0
  ]
  if gis:property-value feature "SOC" = "BLUE" [
    gis:set-drawing-color blue
    gis:fill feature 2.0
  ]
  if gis:property-value feature "SOC" = "UNOCCUPIED" [
    gis:set-drawing-color grey
    gis:fill feature 2.0
  ]
]

;; Set the border of all polygons to white
gis:set-drawing-color white

;; Finally draw the dataset
gis:draw dc-dataset 1

```

Storing the Values of the Polygons. NetLogo environments are based on a grid of square cells ('patches'). The question arises therefore: how is it possible to store

information associated with land parcels that do not align precisely with square cells? One solution, which is used here, is to find the patch that intersects the centre of each polygon, and store the polygon information in that patch. Code Example 6.4.8 outlines the code to do this in full, and the following will explain it in detail.

So that it is possible to distinguish between patches that store information about polygons (hitherto called ‘centroid patches’ because they intersect the centroid of a polygon) from all other patches, begin by creating a variable that can count up from 1. This is used to assign a unique ID number to each centroid patch.

```
let n 1
```

Then start iterating over all of the polygons in the data set:

```
foreach gis:feature-list-of dc-dataset [
```

As before, the word `feature` is used to refer to the current polygon in the `foreach` loop (although any word or character would be acceptable). This can be used, in combination with the `gis:location-of` and `gis:centerid-of` commands, to find the centroid of the polygon:

```
let center-point gis:location-of gis:centerid-of feature
```

The commands above are easier to understand by working from right to left. The `gis:centerid-of` command wants a feature, from which it will return its centroid. Therefore `gis:centerid-of feature` calculates the centroid of the current polygon and returns it. This feature is given to the `gis:location-of` command, which in turn returns the (x, y) coordinates of the feature. These coordinates, which are stored in the variable called `center-point`, are actually a list, so the following code will separate out the x and y coordinates from the list:

```
let x-coordinate item 0 center-point
let y-coordinate item 1 center-point
```

Having found the coordinates of the centre of the polygon, the last thing to do is to find the patch at those coordinates, set its ID, and also assign its initial colour (this will change as agents move around). Note that, as with drawing the polygons, the `gis:property-value` command is used to find the value of the “SOC” attribute that is stored in the shapefile for the current polygon.

```
ask patch x-coordinate y-coordinate [
  set ID n
  set mycolor gis:property-value feature "SOC"
]
```

Finally, before moving on to the next polygon in the data, add 1 to the counter so that the next polygon will have a unique ID.

```
set n n + 1
```

Moving the Agents. After having initialised the environment, it is possible to begin to design the rules that control the agents' behaviour. These are similar to those of the original Schelling model and provided, in full, in Code Example 6.4.9. The commands are largely self-explanatory.

After the agents have finished moving, the very last thing that the model needs to do is update the colours of the polygons depending on the new number of agents in each area. The code to do this is very similar to previous examples outlined here and outlined in full in Code Example 6.4.10.

That's it! This section has demonstrate how to read vector GIS data into a NetLogo model, display the spatial features, read the attributes of the features (the initial polygon colour in this case), move agents around the environment and update the polygon colours accordingly. Although some of the new commands that are provided by the GIS extension might seem complicated at first, they are usually fairly straightforward and are well documented in the [NetLogo GIS Extension](#). Figure 6.12 illustrates how the final model will look.

Detailed Vector Example: Multiple Agents per Polygon

This section will build on the previous example of the Schelling model applied to Washington, DC. Here, however, rather than having only a single agent in each polygon, there will be multiple agents per polygon. The model is called `Segregation_DC_2` and is available in full in the accompanying online resources.

Code Example 6.4.8 Code to store the attributes from a shapefile within the patch that intersects the centre of the polygon feature

```
;; Each polygon identifies a patch at its centroid, which is used to
;; record the colour and population here

let n 1 ;; This is used so that the centre patches have an ID

foreach gis:feature-list-of dc-dataset [
    feature ->

        ;; Find the centroid of the polygon
        let center-point gis:location-of gis:centerid-of feature

        ;; Get the separate x and y coordinates from the (x,y) pair
        let x-coordinate item 0 center-point
        let y-coordinate item 1 center-point

        ;; Get the patch at those coordinates
        ask patch x-coordinate y-coordinate [
```

```

;; Set its ID
set ID n

;; Set its colour, using the value stored in the polygon
set mycolor gis:property-value feature "SOC"
]

set n n + 1 ;; Increment the ID counter
]

```

This takes an important step towards a more realistic model of segregation, applied to a real geographical area. Much of the code, particularly for reading and manipulating the spatial data in NetLogo, is similar to that outlined in Section 6.4.2, so this section focuses on the new aspects. The most important new features that will be introduced are how to initialise agents from information in a shapefile and how to work out how many agents of a particular type are within a particular polygon. However, there are also some other useful examples in the model that will not be discussed here, namely how to calculate Moran's I and a segregation index directly in NetLogo.

Creating Agents from a Shapefile. In this model, agents represent virtual households which observe their environment and make residential choices. The model is composed of two vector layers: the environment is represented as a series of polygons created directly from the ESRI shapefile (as with Section 6.4.2); and agents that are represented as points. Section 5.7.2 illustrated how to load this data into a GIS and manipulate the columns in preparation for use in a model. The environment layer (which represents the boundaries of wards in Washington, DC) contains both the attribute data that determines how many agents to create as well as the geometry of each ward.

In the previous Schelling model, a single patch per polygon was used to store all of that polygon's information. That worked well enough because there was only ever one agent per polygon, so the polygon did not need more than one NetLogo patch for its agent to live in. With the more advanced model, however, there will be multiple agents per ward, so it is necessary to devise a method of creating multiple patches (i.e. agent homes) per ward. Fortunately, the `gis:apply-coverage` command does what is needed (see Box 6.7).

Code Example 6.4.9 NetLogo code to move the agents around the environment depending on the number of same-colour neighbours. The rules are very similar to those of the original Schelling model, but have been adapted to take account of the more complicated neighbourhood calculation

```

to move
  ;; Count the number of red or blue turtles in neighbouring polygons
  ask turtles [

```

```

set tneighbors turtles-on [myneighbors] of patch-here
set tneighborpolygons [myneighbors] of patch-here
set bneighbors count tneighbors with [tcolor = "BLUE"]
set rneighbors count tneighbors with [tcolor = "RED"]
]
;; Calculate the percentage of same colour turtles
ask turtles [
  ifelse rneighbors + bneighbors = 0 [set percentage-same 1] [
    if tcolor = "RED" [
      set percentage-same rneighbors / (rneighbors + bneighbors)
    ]
    if tcolor = "BLUE" [
      set percentage-same bneighbors / (rneighbors + bneighbors)
    ]
  ]
  ifelse percentage-same < (Percentage-same-to-be-happy / 100) [
    set happy? false
  ] [
    set happy? true
  ]
]
;; Move to an unoccupied polygon if not happy and change the colour here
ask turtles [
  if happy? = false and count tneighborpolygons with
    [mycolor = "UNOCCUPIED"] > 0 [
    ask patch-here [set mycolor "UNOCCUPIED"]
    move-to one-of tneighborpolygons with [mycolor = "UNOCCUPIED"]
    ask patch-here [set mycolor [tcolor] of myself]
  ]
]
end

```

Box 6.7 gis:apply-coverage

`gis:apply-coverage VectorDataset property-name patch-variable`

Copies values from the given property of the VectorDataset's features to the given patch variable.

Source: [gis:apply-coverage documentation](#).¹⁴

¹⁴ <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:apply-coverage>

Hence the gis:apply-coverage command can be used to identify the patches that intersect the polygons, extract information (attribute data) about the polygons from the shapefile and store this information in patch variables:

```
;; Copy the colour information to patches (converting vector to raster)
gis:apply-coverage dc-dataset "SOC" mycolor
gis:apply-coverage dc-dataset "ID_ID" ID
```

Code Example 6.4.10 Code to update the colours of the polygons to reflect the changes in locations of the agents

```
to update-colors ;; Update polygon colours
  ;; Patches with an ID > 0 are those that represent the centroids of polygons
  ask patches with [ID > 0] [
    ;; Find the polygon that has its centroid within this patch
    let poly item ( ID - 1 ) ( gis:feature-list-of dc-dataset )

    if mycolor = "RED" [
      gis:set-drawing-color red
      gis:fill poly 2.0
    ]
    if mycolor = "BLUE" [
      gis:set-drawing-color blue
      gis:fill poly 2.0
    ]
    if mycolor = "UNOCCUPIED" [
      gis:set-drawing-color grey
      gis:fill poly 2.0
    ]
  ]

  gis:set-drawing-color white
  gis:draw dc-dataset 1
end
```

The code above takes a colour from the ‘SOC’ attribute and a unique polygon identified from an ‘ID_ID’ attribute and stores that information in patch variables called `mycolor` and `ID`, respectively. Then the distribution of the types of agents (representing, for example, ethnic groups), as observed through aggregate census population counts, can form the initial starting conditions for the model. For example, Figure 6.13A illustrates four census areas, each with its own hypothetical attribute information stored in a data table. In this example, Ward 1 has a population of 10 red, 5 blue, 4 green and 2 white agents. The model reads this data and creates an environment polygon for each ward and for the desired agent population based on the data held in the fields – as per Figure 6.13B.

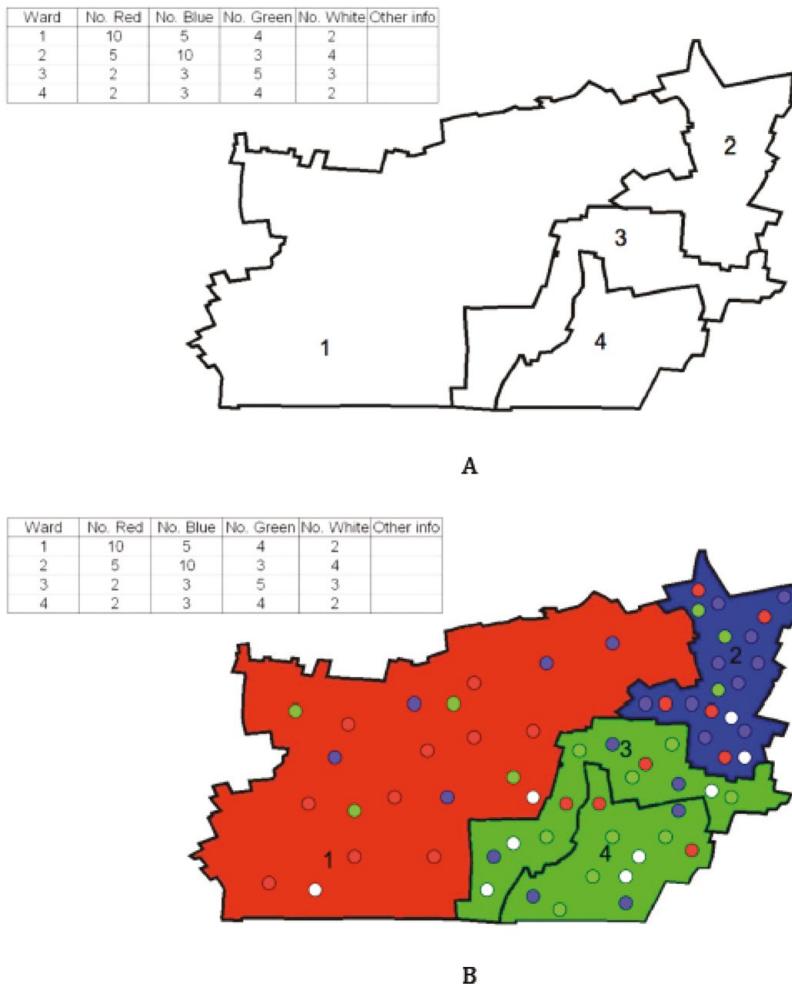


Figure 6.13 Reading in a shapefile and creating agents

The code to create the agents is provided in Code Example 6.4.11. The number of agents to create is determined by the size of the population in the polygon (the `popu` variable), as is the colour of the agent (the `mycolor` variable). Red polygons create 60% red agents and 40% blue agents, and vice versa. This provides a city landscape that is integrated at initialisation. However, these agents could be placed in precise locations if they were known (see Crooks, 2007a, for further detail). Refer to Code Example 6.4.11. Many of the commands are straightforward, but there is one that looks particularly difficult:

```
ask n-of (0.6 * (item 0 popu1 / 10))
  patches with [ID = y and occupied? = false] [
```

Again, reading back from right to left makes this easier to dissect. The `with` part is looking for patches that have the same ID as the current polygon (`y`) and are not occupied:

```
ID = y and occupied? = false
```

In other words, find patches in the current polygon that are available to house an agent. The cryptic:

```
(0.6 * (item 0 popu1 / 10))
```

simply determines how many agents to create. The variable `popu1` is a list that stores the population value of every patch in this polygon. However, we know that every patch in the polygon will have the same population (the `gis:apply-coverage` command made sure of this) so to find the population we simply take any of the values from that list. Here the command `item 0 popu1` takes the first value in the list.¹⁵ That number is divided by 10 to reduce the amount of time needed to run the model (with ten times as many agents it would take hours to produce any results). Finally, it is multiplied by `0.6` because 60% of the agents will be of one colour and 40% will be of the other. Ultimately, `(0.6 * (item 0 popu1 / 10))` calculates a number, which is given to `n-of`, so that we ask for a number of randomly chosen patches that have the same ID as the current polygon and are not occupied.

Finally, when the model is running the agents will move around, and the predominant social group in each ward will change (see Figure 6.14). This means that

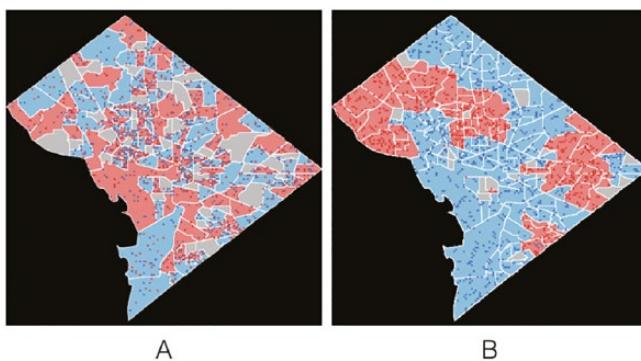


Figure 6.14 (A) Instantiation of a segregation model where each agent is created from attribute information and (B) the resulting end state when all agents are satisfied with their neighbourhood

¹⁵ In NetLogo, as with many programming languages, the first item in a list can be accessed with the number 0, the second with the number 1, etc.

the colour of the polygons in NetLogo needs to be updated. This is accomplished by counting the number of agents of different types within each polygon and working out which one has the largest majority. Code Example 6.4.12 provides the required commands.

There is one more difference between this adapted model and Schelling's original idea. Here, rather than just considering their local neighbourhood (i.e. the surrounding patches), agents also consider the polygon they reside in. If either neighbourhood does not meet their preference then they move to an unoccupied cell in a polygon that is either unoccupied or has the same colour as the agent. Experiments with these two types of neighbourhoods were inspired by O'Sullivan (2009), who discusses how Schelling (1971) explored both a continuous neighbourhood (i.e. Moore) and the bounded neighbourhood (i.e. a common definition of a neighbourhood which remains static over time).

Code Example 6.4.11 NetLogo code to create new agents. The number of agents to create is determined by the size of the population in the polygon, as is the colour of the agent (red polygons create 60% red agents and 40% blue agents, and vice versa)

```
let y 1
while [y <= 188] [
    ;; Find the population and initial colour of all of the patches inside this polygon
    let popul [popu] of patches with [centroid? = true and ID = y]
    let color1 [mycolor] of patches with [centroid? = true and ID = y]

    if color1 = ["RED"] [
        ;; This is a red polygon. Make 60% of the turtles red and the remaining 40% blue
        ask n-of (0.6 * (item 0 popul / 10)) patches with [ID = y and occupied? = false] [
            sprout 1 [
                set tID y
                set tcolor "RED"
                set color red
                set size 2
                ask patch-here [ set occupied? true ]
            ]
        ]
        ask n-of (0.4 * (item 0 popul / 10)) patches with [ID = y and occupied? = false] [
            sprout 1 [
                set tID y
                set tcolor "BLUE"
                set color blue
                set size 2
                ask patch-here [ set occupied? true ]
            ]
        ]
    ]
]
```

```

if color1 = ["BLUE"] [
;; .. Do the same as above, but this time make 60% blue
;; .. and 40% red...
]
set y y + 1
]

```

Box 6.8 Summary: Vector data in agent-based models

In this section, two geographically explicit agent-based models have been discussed. They demonstrated how models can be created using data held within a vector GIS layer and how attributes of the GIS data can be used to create agents. Many more complex examples and more applied models can be created using the same techniques, some of which are outlined in Appendix A. The use of vector data in a GIS will be revisited in Section 8.4, where it will be demonstrated how agents can move around a road network.

6.5 Exporting Data

While the examples above have demonstrated how data can be imported into agent-based models, the means to *extract* data from models have not been considered. There are many instances when information about the model might be useful, for example to make new maps using a GIS, to make detailed visualisations of the model dynamics using bespoke visualisation tools or to do in-depth statistical analysis using statistical software. Some agent-based modelling packages (e.g. Repast and MASON) provide tools to create new shapefiles, while all support some sort of export functionality – the simplest of these being the ability to create plain text (.txt) files.¹⁶

Code Example 6.4.12 Updating the polygon colours depending on the proportions of agents of different colour who are currently living within the polygon

```

to update-colors      ;; Update polygon colors

ask patches with [centroid? = true] [
  let total-here turtles with [ tID = [ID] of myself]

  ifelse count total-here > 0 [
    ;; See if reds make up more than 50%. If not then set the colour to blue
  ]
]
```

¹⁶ The current version of NetLogo only allows for the exportation of raster data, as noted in the documentation (see Box 6.9).

```

ifelse count total-here with [color = red] > (0.5 * count total-here) [
  set mycolor "RED"
]
[
  set mycolor "BLUE"
]
[[ ; This happens if there are no agents in this polygon:
  set mycolor "UNOCCUPIED"
]

;; Now we know what colour to use, draw the polygon correctly:
if mycolor = "RED" [
  gis:set-drawing-color red
  gis:fill item (ID - 1) gis:feature-list-of dc-dataset 2.0
]
;; ... repeat for blue and unoccupied colours

end

```

For example, using Repast, it is possible to export information about the agents (e.g. their current location and attribute values) as a point shapefile. This data can be read into a GIS from which measures such as the population density can be calculated (see Figure 6.15). Although it is not possible to create new shapefiles directly for a vector model in NetLogo, the locations of polygon centroids and their attributes can be written in a plain text file, opened in a GIS package (as was discussed in Section 5.8), and then merged with the existing shapefile of polygon boundaries. Figure 6.16 illustrates this. Both of the models Segregation DC 1 and Segregation DC 2 include a

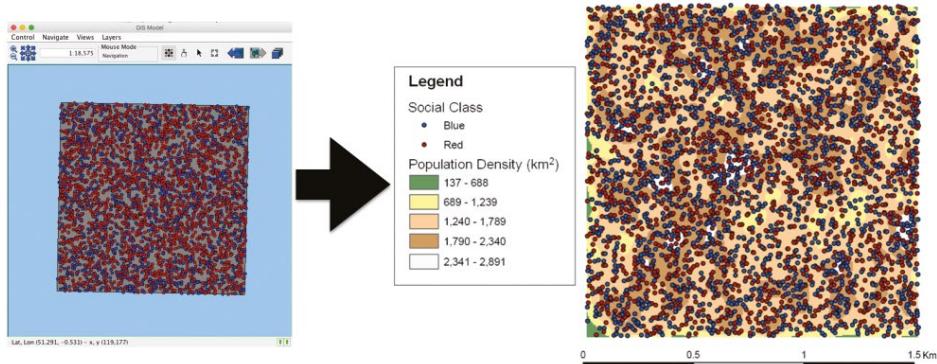


Figure 6.15 An example of taking attribute data from a model and then exploring it in a GIS. Here, agents are exported from a RepastJ model of segregation as a point shapefile. Their locations are then used to calculate the population density using ArcGIS

procedure called `export` that can be used to create a text file of the model data. The code from Segregation DC 1 is included in Code Example 6.5.1 and demonstrates how to make a text file that contains the ID and colour of every polygon (one line per polygon). Note that the file is formatted using ‘comma separated values’ (.csv) format, which is commonly used by many GIS, statistics and spreadsheet packages.

For raster data, NetLogo provides a function to directly export an .asc file called `gis:store-dataset`. For example, Figure 6.17 illustrates how the final land-use pattern can be exported from the simple urban growth model presented in Section 6.4.1 and then visualised in a GIS.

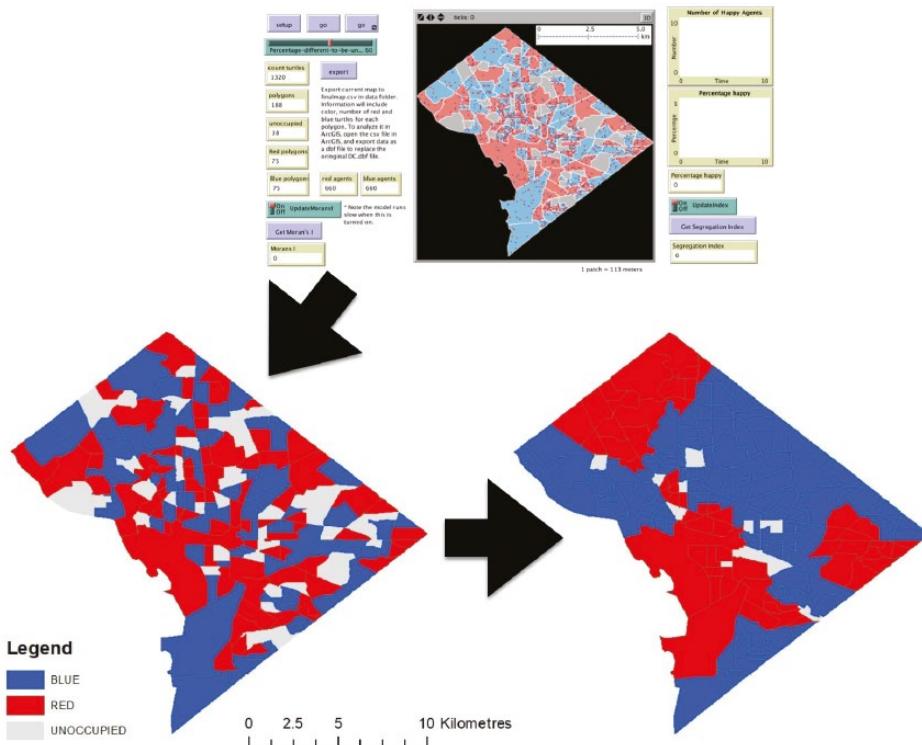


Figure 6.16 An example of taking attribute data from the NetLogo Schelling model and displaying it in a GIS. Agents are exported from the Schelling model as a .txt file at the start and end of the simulation. They are displayed in a GIS. Initially there are 1320 agents distributed over 118 polygons (of which 75 are blue, 75 are red, 38 are unoccupied). Each agent has a preference to be in an area where 60% of their neighbours are of the same colour. Over the course of the simulation this results in agents moving and altering the population and colours of the polygons (114 blue, 58 red, 16 unoccupied)

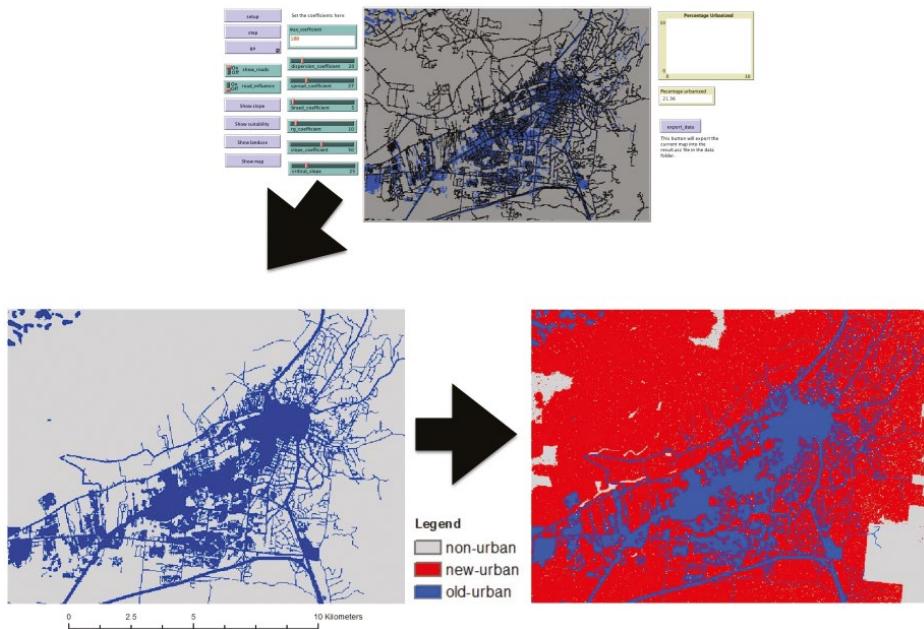


Figure 6.17 Exporting raster data from a simple urban growth NetLogo model and visualising it in a GIS for further analysis. In this example the model ran for 100 time steps and the area urbanised went from 21% to 92%

Code Example 6.5.1 Instructions to write out each polygon's ID and colour to a plain text file. The file is formatted using 'comma separated values' (csv) format, which is commonly used by many GIS, statistics and spreadsheet packages

```

to export

;; This function creates a text file for the colour information of the current map.
file-close
file-delete "data/result.txt" ; Delete the old file, if one exists
file-open "data/result.txt"    ; Open a new file to save the results to
file-print "ID,ID,SCC\r\n"      ; First write the header. This is just the ID and colour

;; Now loop over each polygon and write its ID and colour
let i 1
while [i <= 188] [           ; There are 188 polygons
  file-type i      ; Write the ID
  file-type ","    ; Write a comma to separate ID and the next column
  ;; Ask the corresponding patch to write its colour:
  ask patches with [ID = i] [file-print mycolor]
]

```

```

file-type "\r\n"           ; Write a new line (we want one line per polygon)
set i i + 1
]

file-close ; Close the file, save the results

end

```

Box 6.9 gis:store-dataset

The `gis:store-dataset` command in NetLogo provides the functionality to write export raster data directly. It requires two pieces of information: the data set to write, and the location of the file to write to. The [gis:store-dataset documentation](#)¹⁷ states:

Saves the given dataset to the given file. If the name of the file does not have the proper file extension, the extension will be automatically appended to the name. Relative paths are resolved relative to the location of the current model, or the user's home directory if the current model hasn't been saved yet.

Currently, this primitive only works for RasterDatasets, and it can only save those data sets as ESRI ASCII grid files.

6.6 Discussion

This chapter has highlighted the ability to import and export geospatial data in both raster and vector formats (via a loose coupling approach). Both types of formats have their advantages and disadvantages. For cellular space, it is much easier conceptually and computationally as one can use Moore and von Neumann neighbourhoods. Furthermore, a vast amount of data comes in raster formats, such as remote sensing data. However, using cellular space, there is no way of representing complex geometries. The advantages of using vector space include the ability to give more realism to models by allowing the representation of any geographical shape, moving away from treating the world as a series of square objects. Also, most demographic data come in this format, thus the data does not have to be transformed to a series of cells. However, these advantages come at a price. Topological calculations (such as neighbourhood calculations, the distance between two points

¹⁷ <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:store-dataset>

and point-in-polygon operations) are expensive in terms of execution speed, thus limiting the performance of a vector-based model. Therefore, which space to use for geospatial models depends on the purpose of the model. Benenson et al. (2005) comment that while vector GIS can represent urban objects in spatially explicit models, for theoretical models the points of a regular grid will usually suffice.

By adding real world geographical information to agent-based models we now not only have the ability to model actual decision-making units such as individuals, households and firms but also can account for the influence of specific local conditions. Moreover, we can use data from GIS to create artificial populations and agent descriptions based on data about the area under investigation. Not only does this allow us to set the initial conditions of the model – for example, land use or the predominant social class of the area – but if time series data are available (e.g. census or house price data) we could compare population shifts or housing dynamics to actual places.

Chapter Summary

In previous chapters we have shown how agent-based modelling allows us to explore how individuals interact, giving rise to the emergence of aggregate patterns (e.g. crowds or segregation). This environment can be used as an artificial laboratory to test out ideas and hypotheses. In this chapter we have demonstrated how GIS allows us to create agent-based models that are directly related to actual geographical locations via raster and vector data sets. It was also demonstrated how GIS data can provide a landscape for the agents to inhabit (e.g. a terrain for them to walk upon, or locations to live) as well as deriving attributes from such data to instantiate agents. In the following chapters attention will be focused on how geographical information can be used to compare aggregate outputs from these models to real world phenomena.

6.7 Annotated Bibliography

For more information on GIS functionality in NetLogo, readers are referred to:

- <https://ccl.northwestern.edu/netlogo/docs/gis.html> along with the NetLogo GitHub page: <https://github.com/NetLogo/GIS-Extension>

The Simulating Complexity website provides a good tutorial on integrating GIS into NetLogo:

- <https://simulatingcomplexity.wordpress.com/2014/08/20/turtles-in-space-integrating-gis-and-netlogo/>

For key texts about the integration of GIS and agent-based modelling, readers are referred to:

- Crooks, A.T. and Castle, C. (2012) The integration of agent-based modelling and geographical information for geospatial simulation. In A. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 219–252. New York: Springer.
- Maguire, D.J., Batty, M. and Goodchild M.F. (eds) (2005) *GIS, Spatial Analysis and Modelling*. Redlands, CA: ESRI Press.
- Westervelt, J.D. (2002) Geographic information systems and agent-based modelling. In H.R. Gimblett (ed.), *Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes*, pp. 83–104. Oxford: Oxford University Press.

7

MODELLING HUMAN BEHAVIOUR

Chapter Outline

This chapter explores the most common approaches by which researchers incorporate human behaviour into agent-based models. We explain why it can be necessary to model human behaviour and discuss the main considerations that the researcher needs to be aware of when developing behaviour rules for an agent-based model. We present an overview of the two main approaches: mathematical and conceptual cognitive models. We supplement this discussion with two case studies that provide examples of how these approaches can be implemented. The chapter finishes with a discussion of some of the thorny issues that researchers need to be aware of when attempting to simulate behaviour within agent-based models.

7.1 Introduction

Imagine how difficult physics would be if electrons could think.
(Murray Gell-Mann)

One of the enduring challenges for geographers is understanding the consequences of individual actions over both time and space. As the quote above suggests, capturing human thinking is a challenge, in the sense that unlike the physical sciences where particles are somewhat well behaved under a specific set of conditions, humans react differently to a particular situation or stimulus. Earlier modelling efforts (as discussed previously in Chapters 1 and 2, and later in Chapter 11) were limited to applying statistical models to aggregate-level data – for example, the use of spatial interaction models to simulate the flows of a population between an origin and a destination (Wilson, 1974). This resulted in much of the vibrancy

and rich detail that characterise many populations being ‘aggregated out’ in favour of parsimony. The proliferation of big data has removed an important barrier in our potential to ‘lift the lid’ on what is happening within geographical systems – where we once had to take the average and gloss over detail, we now find ourselves in the position of being able to potentially simulate every individual and all their unique traits. This, combined with a new approach to viewing geographical systems through complexity science (see Chapter 1), in particular through the notion of ‘bottom-up’ modelling, has resulted in an increased sophistication within both our models and the questions that we can ask with them. While this presents new opportunities for researchers, it also presents a new set of challenges to be solved; new tools are needed to allow us to uncover and understand the complexity of our systems. Without these tools we are in danger of building models that are too complex to be understood – see Section 12.2.7 and the work of Batty and Torrens (2005), Crooks et al. (2008), Müller et al. (2013) and Schlüter et al. (2017).

The explosion in individual data, along with increases in computing power, offers some explanation for the rise in popularity of agent-based modelling over the past two decades. However, simulating individuals (human or otherwise) presents a unique challenge for modellers. Whilst the agent-based modelling paradigm perfectly aligns with setting up and creating heterogeneous individuals that move around our ‘artificial’ landscapes, how do we capture and embed the behaviour that makes them unique? As Schlüter et al. (2017) observe, while the importance of capturing the complexity of human behaviour is now acknowledged, integrating this into formal models remains a significant challenge (which is something we come back to in Section 12.2). In particular, these challenges are well summarised by Schlueter et al. (2017): (i) bringing together diverse theories, often complicated by different disciplinary languages from across the social sciences; (ii) choosing a theory with the appropriate level of detail, with theories ranging from detailed representation of a narrow aspect of decision-making to a very broad representation; (iii) understanding and translating the degree of formalisation within each theory; and (iv) reconciling where causal mechanisms are not specified.

This chapter focuses on these challenges associated with modelling human behaviour within agent-based models. Following a general discussion about the issues associated with simulating behaviour (Section 7.2), behavioural models are introduced (Section 7.3). As well as providing an overview of these types of models, we present the main challenges facing researchers when attempting to implement them. We present a more detailed overview of the most common types of mathematical and cognitive models in Sections 7.4 and 7.5, respectively. Then, in Sections 7.6 and 7.7, we use two published case studies to show how these behavioural models have been implemented within agent-based models. The chapter concludes with a short discussion of the main issues associated with embedding human behaviour within agent-based models (Section 7.8).

7.2 The Challenge of Simulating Human Behaviour

Human behaviour flows from three main sources: desire, emotion, and knowledge. (*Plato*)

This quote from Plato nicely articulates why modelling human behaviour presents such a challenge. Humans possess such diverse personality traits, varying levels of knowledge, experience, desires and emotions (Izard, 2007; Bonabeau, 2002), that attempting to simulate any small aspect of this seems a foolish endeavour. Many of these basic emotions are short-lived, but can lead to complex, often unpredictable, longer-term behaviour. Added to this, humans do not live in isolation – we are embedded within dense and interconnected networks encompassing work, social and family ties (Friedkin, 1998) and our actions and behaviour are influenced by what others say and do. It is easy to represent social networks in agent-based models (as we discuss in Chapter 8), but these social networks are embedded in space and, to some extent, time. The challenge for the researcher is to represent not only the social relationships between agents, but also how they vary with space and time and how they influence the behaviours of the agents.

Fortunately, human behaviour is often predictable, as the following example shows:

Box 7.1 Pick a number

Kennedy (2012) challenges the reader to answer the following question: Pick a number between 1 and 4.

The most common response is ‘three’ The second most common answer is ‘two’. Very few people decide to respond with either ‘one’ or ‘four’. Sadly, there is not a serious study of this behaviour but undocumented sources suggest that the response statistics are close to 50% for ‘three’, 30% for ‘two’ and about 10% for the other two answers.

What does Kennedy’s (2012) simple experiment tell us? Simply that human behaviour is not purely random, that there is a degree of predictability to how an individual may behave in a given situation. If we can therefore gather enough accurate information about how an individual will react and behave (e.g. from case studies, newspaper reports, laboratory experiments), we can formulate rules that an agent can use (see Robinson et al., 2007).

Fortunately, when building our models, we are only interested in one or two clearly defined aspects of behaviour that we believe have a strong influence on the system under investigation. Our interest may lie in simulating how pedestrians move through time and space (e.g. Torrens, 2012), urban residential choice

(e.g. Huang et al., 2014), susceptibility to disease (e.g. Augustijn-Beckers et al., 2011b) or human behaviour in reaction to a natural disaster (e.g. Crooks and Wise, 2013). There are an abundance of examples readily available within the agent-based modelling literature that focus on a small aspect of human behaviour. Figure 7.1 shows a representative set of applications. At the top of the figure we have arranged the models based on the temporal or spatial scales of the phenomena they are modelling, while at the bottom we have used axes to represent system/environment complexity and behavioural complexity and grouped the examples accordingly.

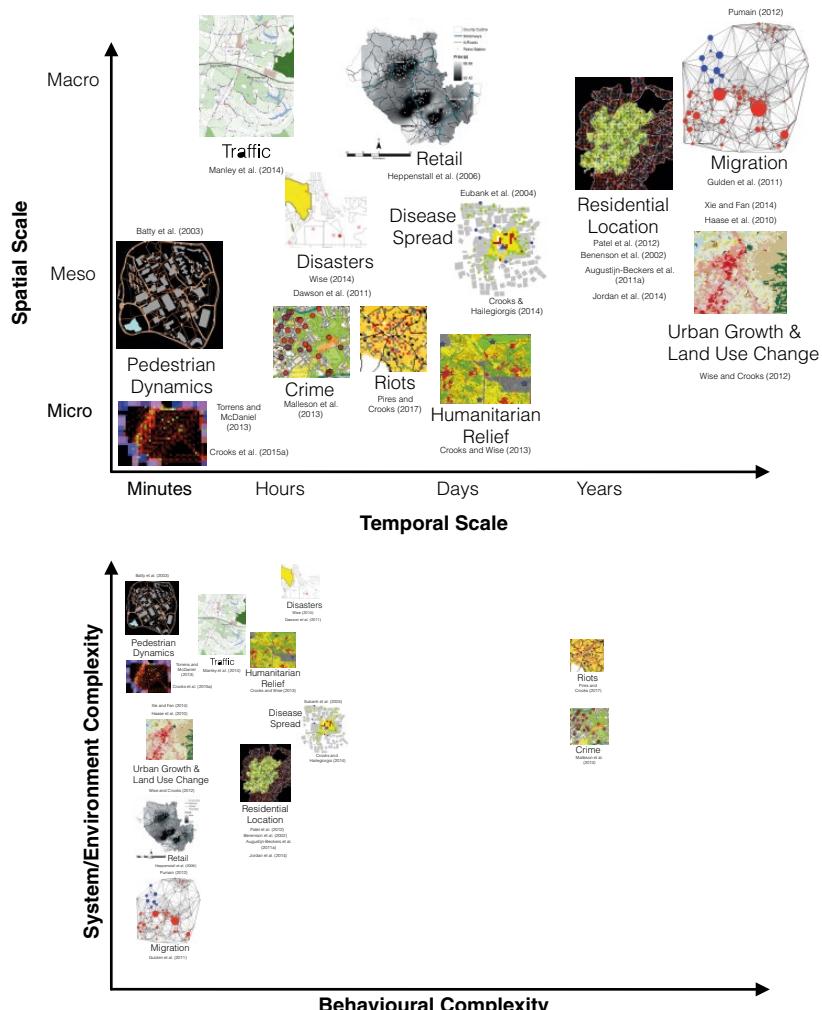


Figure 7.1 In relation to a range of specific example ABMs; above, models are plotted according to spatial and temporal scales, and below behavioural complexity is plotted against environmental complexity

With respect to *system environment complexity* we are referring to how the models represent space from abstract spatial representations – from the Schelling (1971) segregation model to models that capture the basic properties of space such as distance between places (e.g. Gulden et al., 2011; Pumain, 2012) and geographically explicit models that capture the complex geometries of the built environment (e.g. Batty et al., 2003; Torrens and McDaniel, 2013). Turning to *behavioural complexity*, here we try to categorise models from simple behaviours (left) to more complex behavioural representations (right). Simple models would be along the lines of the Schelling model or the pedestrian models of Crooks et al. (2015a), while at the other extreme the models would need to capture all the functions of a ‘real’ person.

There are a couple of points to note. Until recently, applications could be characterised as being either behaviourally complex – for example, Malleson et al.’s (2013) crime model – or environmentally complex, such as Manley et al.’s (2015a) traffic model. Few published agent-based models are both behaviourally and environmentally complex. However, with the increase in new forms of data and processing capabilities, we are beginning to see an increase in these types of applications (see Torrens and McDaniel, 2013, for a discussion). Another noticeable feature is that all the examples of capturing both behavioural and environmental complexity are at the micro level where it is more plausible to capture and represent behaviour. At the larger scale – as evident in the work of Gulden et al. (2011) and Wise and Crooks (2012) – models are neither behaviourally nor environmentally complex, and for the processes that they are simulating they need not be. While these examples focus on social systems, there is a whole body of research that is concerned with coupled human and natural systems to understand the consequences of interactions between the two. An excellent introduction to this literature can be found in An (2012), Filatova et al. (2013) and Schlüter et al. (2017).

7.3 Behavioural Frameworks

There are a whole host of theories in the literature which detail human decision-making that are based on specific applications or problems. One of the largest challenges is selecting an appropriate theory (which is often descriptive) or set of justifications to base decision-making upon, specifically in the construction of the rule set, within agent-based models.

Once an appropriate theory has been selected that best represents the behaviour of interest, the next challenge is to build that behaviour into the agents (we will return to the relationship between theory and model in Section 12.2). This is where behavioural frameworks come in. Broadly speaking, there are two scientific approaches to modelling human behaviour being developed: artificial intelligence using mathematical approaches; and conceptual cognitive approaches that attempt to model all forms of human cognition, including emotions, intuitions and motivations.

As Meyfroidt (2013) comments, full implementations of these frameworks are very limited in empirical agent-based models (Meyfroidt, 2013), while theoretical representations of human decision-making are still weak (Heckbert et al., 2010). In this section, we present an overview of the different types of behavioural framework on offer (Section 7.3.1) before moving to look at the challenges associated with implementing these models (Section 7.3.2). Examples of the different types of behavioural models are given in Sections 7.4 and 7.5.

7.3.1 Types of Behavioural Frameworks

As the number of agent-based modelling applications has increased almost exponentially in the last 20 years (as shown in Section 2.4), the number of available behavioural frameworks is seemingly undergoing similar growth. An agent's architecture determines how the functionality of the agent is organised and how the agent replicates human or biological traits such as reasoning, beliefs, attitudes and behaviour (Singh, 2005). A number of architectures have been proposed to address how these traits should be mimicked; Balke and Gilbert (2014) present a review of available frameworks ranging in diversity from production-rule systems to complex psychologically inspired cognitive ones. From this review, they identify what they see as the five most important dimensions for classifying agent-based modelling work and thus distinguishing their architectures, as shown in Table 7.1. This overview of different dimensions provides a useful checklist for assessing what type of framework is needed for the behaviour that is to be represented. Table 7.2 extends Table 7.1 by listing the key assumptions and application areas of several of the most widely used theories used to represent behaviour in agent-based models.

Table 7.1 The five main dimensions for distinguishing agent architectures

Dimensions	Explanation
Cognitive	What kind of cognitive level does the agent architecture allow for: reactive agents, deliberative agents, simple cognitive agents or psychologically or neurologically inspired agents?
Affective	What degree of representing emotions (if any at all) is possible in the different architectures?
Social	Do the agent architectures allow for agents capable of distinguishing social network relations (and status), what levels of communication can be represented, and to what degree can one use the architectures to represent complex social concepts such as the theory of mind or we-intentionality?
Norm consideration	To what degree do the architectures allow the modelling of agents which are able to explicitly reason about formal and social norms as well as the emergence and spread of the latter?
Learning	What kind of agent learning is supported by the agent architectures?

Source: after Balke and Gilbert (2014).

Table 7.2 Overview of the key assumptions and application areas of popular theories used in representing behaviour

Theory	Origin/description	Key assumptions	Application areas	Key references
Rational choice theory/Homo economicus	Economics	Self-interested utility maximisation; goal orientated; stable and transitive preferences; perfect knowledge; unlimited cognitive capacity for calculating outcomes of all possible behavioural options	Economics; political science; psychology; international relations. Frequently used to model human decision-making in natural resources (e.g. tragedy of the commons)	Simon (1978), Frank (1987), Monroe (2001)
Bounded rationality	Economics, psychology. Rationality is limited by available information and cognitive capacity. Note there are many different versions of bounded rationality	Goal orientated, self-interested; may have cognitive limitations, incomplete or uncertain information about the world, and limited time. The behaviour choice can be realised through maximising utility, reaching an aspiration level (satisficing) or following a heuristic (Gigerenzer and Sellen, 2001)	Economics; political science; psychology; international relations.	Simon (1955), Gigerenzer and Sellen (2001)
Theory of planned behaviour	Environmental psychology. Behaviour is mediated by intentions and perceived behavioural control. Intentions are based on behavioural beliefs (attitudes), normative beliefs (subjective norm) and control beliefs (perceived behavioural control)	Attitudes are aggregated beliefs about the strength of the effect of the behaviour and their normative value. Subjective norms are aggregates of the beliefs of approval/disapproval of the behaviour by important individuals or groups and the motivation to comply with important others. Perceived behavioural controls are aggregates of the beliefs about a control factor (e.g. money) and the perceived power of the control factor (e.g. is money important?)	Environmental psychology. Mostly applied in empirical studies to predict intention and behaviour	Ajzen (1991)

(Continued)

Table 7.2 (Continued)

Theory	Origin/description	Key assumptions	Application areas	Key references
Habitual/reinforcement learning	Biology, psychology. Behavioural learning that originates in the classical Pavlov (1927) model. Habit is a behaviour that we often exhibit without thinking. Reinforcement learning is an approach to representing habitual behaviour	Behaviour is initially deliberate and goal directed. If new behaviour is rewarded, the chances increase that it will be repeated. Repeatedly obtaining satisfactory rewards reinforces the behaviour. The selection of behaviour will be automatic as long as needs are satisfied. The actor will stop automatic behaviour and deliberate about alternative behaviours if need satisfaction drops below a critical level. If the reward devalues or disappears habitual behaviour persists at first, but will go extinct after longer absence of reward	Psychology, neuroscience	Pavlov (1927), Skinner (1953), Graybiel (2008)
Descriptive norm	Social norms are studied within different social science disciplines as a key element affecting decision-making	Observing the behaviour of others can have an impact on a person's behaviour. Observation can take place in an almost subconscious manner, during which the observed behaviour becomes more salient for selection. Or the observation can be more deliberately processed such that other people's behaviour serves as a cue in deciding the proper action to take in a particular situation.	(Experimental) studies of environmentally related behavior (e.g. voting behaviour)	Gerber and Rogers (2009), Cialdini et al. (1991)
Prospect theory	Psychology. Introduces important aspects from cognitive psychology to the rational-actor model, specifically with respect to how people's willingness to seek or avoid risk influences their decisions	Actors bias a rational decision because the context shapes their aversion to risk. Actors have a degree of risk aversion, whereby actors bias decisions towards avoiding loss over chancing a gain. When the stakes are small, actors tend to gamble and seek more risk	International relations, financial, risk management	Kahneman and Tversky (1979)

Source: after Schlüter et al. (2017).

7.3.2 Challenges

There are a number of important challenges that researchers need to address if they are to successfully embed theories on human decision-making into their models; these include finding the relevant theory, formalising the theory and introducing causality (Schlüter et al., 2017). An example of a generic theory is the theory of planned behaviour (Ajzen, 1991) which describes the full process from the formation of different beliefs to the resulting behavioural actions. This includes describing how the behavioural beliefs and attitudes, normative beliefs and subjective norms, together with control beliefs, influence the implementation of the behaviour. A more specific theory that has been widely used in geography is Alonso's (1964) bid-rent theory. This theory states that individuals will always compete for the land nearest to the centre of the city. It does not consider any other forms of behaviour or competition for other types of land. As Schläuter et al. (2017) remark: 'This fragmentation of knowledge makes it challenging to unpack the implications of different theories, particularly in different contexts.'

This leads to the second challenge. It is often the case that the researcher needs to make assumptions to fill logical gaps that appear when trying to formalise the theory. Using the theory of planned behaviour (Ajzen, 1991) as an example again, we find that within this theory, the subjective norm (which describes an individual's perceptions about how significant others will judge the behaviour under consideration) needs to be specified. This involves the modeller making assumptions about what exactly is a significant other, and what this level of significance is.

The final challenge that needs to be considered concerns causality. Simulating human interactions both in time and space requires the specification of causal relationships about how different factors such as social and psychological factors influence an agent's decision-making. However, often empirical studies that are based on one moment in time are used to construct theories. These 'static' empirical studies cannot capture the dynamic nature of the causal relationships between different factors. In place of real understanding about these relationships, the modeller has to embed assumptions within the model about the processes and dynamics within the system under consideration. In the case of simulating the behaviour of a burglar, the modeller may need to make assumptions about the motivations and how the individual moves around her environment and gathers information about the next likely target (e.g. Malleson et al., 2013).

7.4 Mathematical Approaches

Turning to how to represent behaviour in models, perhaps the most widely used approaches are based within mathematics. Mathematical approaches centre on the custom coding of behaviours within the simulation, such as using

random number generators to select a predefined possible choice (e.g. to buy or sell; Gode and Sunder, 1993). Here we present two of the main approaches taken within agent-based models: probabilistic (Section 7.4.1) and threshold rules (Section 7.4.2). However, while it has been argued that these approaches to modelling are appropriate when behaviour can be well specified, they might not be appropriate when more complex behaviours are needed (Kennedy, 2012).

7.4.1 Probabilistic Models

Probabilistic models are used when there is a degree of uncertainty or randomness that needs to be accounted for in the rules that an agent operates. To give a simple example, consider a hypothetical shopper buying their lunch. They might have a choice of sandwiches or noodles, with a probability assigned to each (e.g. a 30% chance of choosing sandwiches and a 70% chance of choosing noodles). Each time the simulation runs, they might choose different food. These probabilities reflect a lack of information about the choice for lunch – we do not know why a person chooses one thing over another, so we take an educated guess at which it might be. With more theoretical and empirical work, the researcher might be able to determine that the choice is heavily influenced by the time of day. Therefore they could adapt their model such that the ratio of probabilities changes throughout the day (before 11am the agent might be much more likely to choose sandwiches, but after 2pm the probability of noodles might be higher). There is still uncertainty about the choice, but under certain conditions we can be more certain about the decision and set the probabilities to reflect this. This probabilism is why agent-based models typically need to be executed a number of times (Chapter 10 will discuss in more detail the need to run models multiple times).

Probabilistic rules can also be used when an event might take place, but not necessarily by choice of the agent. For example, consider a probability of becoming infected with a specific disease. This could be estimated based on the virulence of the disease (its propensity to spread from one person to another) and the likelihood that an infected agent is in the vicinity. If an infected agent meets an uninfected one, there is a chance that the uninfected one will pick up the disease. In some simulations the agent will become infected, in others they will not. This approach has been used in practice for modelling the spread of cholera in a refugee camp (Crooks and Hailegiorgis, 2014). In this model, agents have a probability of becoming infected with cholera bacteria and also a probability of dying if they become infected. At the same time in this model, agents also have a probability (chosen from a normal distribution) of choosing certain activities (e.g. going to visit a friend). More information about this model can be found in Appendix A.1.

The notion of probability is also used in the network models presented in Chapter 8, especially with respect to small-world networks (Section 8.3.2) where nodes (i.e. agents) have a probability of being rewired to each other

or in scale-free networks (Section 8.3.3), and new nodes have a probability of being added to existing nodes based on preferential attachment (Barabási and Bonabeau, 2003). Probabilistic rules are also used for chance encounters or setting agents' opinions on a certain topic (e.g. Alizadeh et al., 2015), and in Oldham's (2016) model of the Battle of Britain, planes had a probability of being shot down if the enemy aircraft were within firing distance.

7.4.2 Threshold Models

One of the most commonly used approaches is that of threshold-based rules. Threshold rules come into operation when a preset value is exceeded. Depending on the value, this will result in behaviour from a predefined set. For example (see Kennedy, 2012):

```
IF <hunger> is below <hungerThreshold1> THEN agent-dies
IF <hunger> is above <hungerThreshold2> THEN address another goal
IF <hunger> is between <hungerThreshold1> and <hungerThreshold2>
THEN search-for-food
```

A simple example of how the threshold rules operate can be found in the Schelling (1971) model first introduced in Chapter 2. Within this model there are only two types of agent: yellow and green. These agents possess a desire to live in a neighbourhood (defined by its eight surrounding cells) with a percentage (the threshold) of neighbours who are identical to themselves. Initially the agents are randomly distributed throughout the environment. As the simulation begins, the following rules come into play, demonstrated here using a threshold level of 50%:

```
IF surrounded by <50\%> of same colour THEN agent-moves
IF surrounded by <50\%> of same colour THEN agent-stays
```

The example in Figure 7.2 shows the agent preference (or threshold) for similar neighbours set at 30%, 40%, 50% and 60% (i.e. at least x% of an agent's neighbours must be of the same type for the agent to be satisfied).

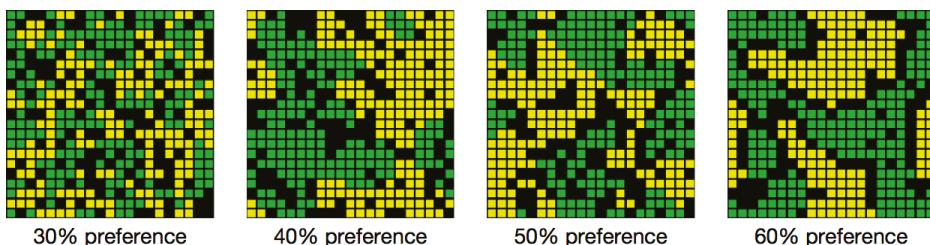


Figure 7.2 Resulting patterns of segregation from different threshold levels

While such rule sets are able to broadly simulate behaviour, one of the main criticisms levelled at this approach is that it cannot easily account for the multiple components of human behaviour, and it should only be used when behaviour can be well specified. For the representation of more sophisticated behaviours, cognitive frameworks are often more appropriate, and it is to these that we turn to next.

7.5 Conceptual Cognitive Models

The second broad class of models are conceptual cognitive models. Within this section we present three examples of cognitive models: Section 7.5.1 gives an overview of the popular beliefs–desires–intentions (BDI) model, Section 7.5.2 briefly gives an overview of the fast and frugal model, and Section 7.5.3 presents the physical conditions, emotional states, cognitive capabilities and social status (PECS) model.

7.5.1 Beliefs–Desires–Intentions Model

The beliefs–desires–intentions architecture (Bratman et al., 1988) is perhaps the most popular architecture and is centred around equipping agents with the cognitive components of beliefs, desires and intentions. This approach follows rational choice ideas because no action is performed without some form of deliberation (Balzer, 2000). The behaviour of a BDI agent is characterised by ‘practical reasoning’: goals are decided upon and then a plan is formed in order to satisfy the goals (Singh, 2005). Beliefs represent the agent’s internal knowledge of the world. The agent has a ‘memory’ of past experiences and the state of the environment as it was last seen. Desires are all the goals which the agent is trying to achieve. These can include short-term goals such as ‘eat food’ and more complex, long-term goals such as ‘raise children’. As some goals might be contradictory, intentions represent the most important goals which the agent chooses to achieve first. Intentions are sometimes viewed as a subset of goals, while at other times they are viewed as the set of plans which will achieve the desired goals (Singh, 2005). Goals (and therefore also intentions) will change throughout time depending on external inputs and the agent’s internal state. A level of caution can be integrated into a BDI agent by specifying how eager the agent is to change its intentions. The BDI architecture has been widely used in a diverse set of applications from air traffic management (e.g. Rao and Georgeff, 1991), the animation of characters in virtual environments (e.g. Torres et al., 2003), studying operations with a shipping container terminal (e.g. Lokuge and Alahakoon, 2004) to transportation (e.g. Horni et al., 2016), geopolitics (e.g. Taylor et al., 2004) and crime (e.g. Brantingham et al., 2005a,b). Readers wishing to know more about the use of BDI are referred to Rao and Georgeff (1995), and to Müller et al. (2013) for

its applications. However, while widely used, it has also suffered some criticism. Fundamentally, the architecture assumes rational decision-making; this is difficult to justify because people rarely meet the requirements of rational choice models (Axelrod, 1997a). Furthermore, Balzer (2000) notes that the core human elements (beliefs, desires and intentions) are difficult to observe directly. Access to them can only be achieved in a laboratory setting which might not relate to real situations. Some criticise the three attitudes which form the core of the architecture (beliefs, desires and intentions) as being either too restrictive or overly complicated (Rao and Georgeff, 1995).

7.5.2 Fast and Frugal Model

Another commonly used cognitive approach is the fast and frugal framework (Gigerenzer and Goldstein, 1996). The fast and frugal approach consists of a school of *heuristic* rule sets, designed to better capture how decision-making is made under uncertainty. The proponents argue that, under many conditions, rational decision-making is too cognitively complex, and that humans will naturally revert to simple rules to guide their behaviour, and these decisions will be based only on the reliable information available to them. Rather than identifying and measuring all alternatives within an environment, decision-makers will instead use simple *cues* on which to base their decisions. From a modelling perspective, the fast and frugal approach resembles a decision tree structure, and unlike other approaches, the model considers the cues affecting a decision sequentially in the order of their importance rather than in parallel (Kennedy and Bassett, 2011).

An example of the use of the fast and frugal decision tree within an agent-based model is presented by Deadman et al. (2004) who explored how individual households made choices with respect to farming practices and how these choices led to deforestation in the Amazon rainforest. This spatially explicit model (including soil types and land cover) produced similar results to observed historical data in the Altamira region of Brazil and provided insights into the factors impacting land-use change in the region from the bottom up. Wise (2014) developed a model of wildfire events using the fast and frugal decision tree for the residents in Colorado Springs to decide whether to go to work, stay at home or evacuate based on how much information they had (more details about the model can be found in Appendix A.9).

Another use of the fast and frugal decision tree can be found in the RiftLand model (Cioffi-Revilla et al., 2011; Kennedy et al., 2014) which was developed in MASON and utilises GeoMASON to simulate coupled social and natural systems. Its purpose was to explore how climate change could impact the inhabitants (herders, farmers, urban populations) of several African countries, and how this results in movement and conflict between people (snapshots of this simulation can be found in Appendix A.14). Within the fast and frugal decision tree,

there are several rules. The first is ‘are animals dying?’ This has a binary (yes/no) response. The second question is ‘is there a conflict nearby?’ A parameter is set for how close is ‘nearby’. This is determined via a probabilistic evaluation (e.g. $X + \text{probabilistic noise value}$).

Multiple heuristic approaches can be combined in defining a complex decision-making framework, as evidenced in Manley et al. (2015b). This framework combines hierarchical models of space and a set of heuristics rules in simulating vehicular route choice under uncertainty. Within this configuration, at each aggregation of space – regional, nodal or at the road segment level (see Figure 7.3A) – a different set of heuristics are applied (Figure 7.3B). This differentiation distinguishes the strategic from localised decision-making, meaning that at the regional level, decisions are made regarding the broad traversal of the city, while at the road-level scale, fine-scale decisions are made within the context of the strategic goals. This decision-making framework is integrated within a large-scale agent-based model, documented in Manley (2014), which outperforms conventional traffic simulation approaches.

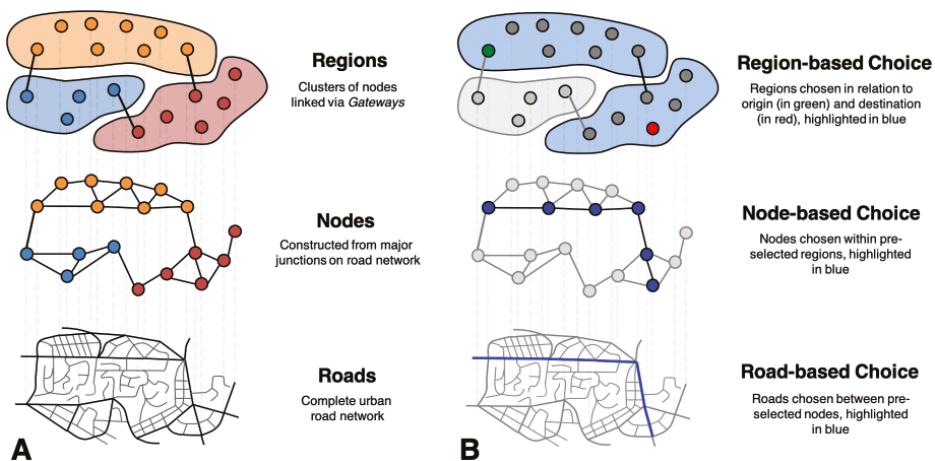


Figure 7.3 Heuristic model of route choice: (A) breakdown of space into a hierarchy, differentiating regional, node-based and road segment-based decision-making; (B) an example route choice process, where different heuristic rule sets are engaged at each level of the hierarchy

How is this framework different from the probabilistic models presented in Section 7.4.1? One of the key differences is that within the fast and frugal decision tree, the decisions are based on heuristic facts (see Gigerenzer and Gaissmaier, 2011) and the order in which the decisions are made relates to ordered facts

(i.e. what are considered the most important to least). Agents decide serially if they have enough information on which to act, and if not they consider the next factor. In probabilistic models an agent's actions are drawn from a probability (e.g. the probability of infection or the probability of attaching to a new node).

7.5.3 Physical Conditions, Emotional State, Cognitive Capability and Social Status Model

The PECS framework, proposed by Schmidt (2000) and Urban (2000), states that human behaviour can be modelled by taking into account physical conditions, emotional states, cognitive capabilities and social status of an agent. The framework is modular, allowing separate components to control each aspect of the agent's behaviour (Martínez-Miranda and Aldea, 2005). Proponents of PECS claim that as rational decision-making is not required and the framework is not restricted to the factors of beliefs, desires and intentions (Schmidt, 2000), it is an improvement on the BDI architecture.

To illustrate the PECS features, an example proposed by Urban (2000) is adapted here. Consider a person in a shop who is contemplating purchasing some goods. They might experience physical needs (such as hunger), emotional states (such as surprise at the available goods), cognition (such as information about current prices) and social status (which will, for example, affect how the agent reacts to the shop assistant). Schmidt (2000) and Urban (2000) argue that every aspect of human behaviour can be modelled using these components, although, depending on the application, it might not be necessary to incorporate all of them (Schmidt, 2002).

Despite documented use of the framework being limited, the applications that have incorporated it are diverse. For example, PECS has been used to build emotions into a virtual learning environment (Ammar et al., 2006; Neji and Ammar, 2007). Here, non-verbal communication was incorporated in the form of emotional facial expressions with the aim of improving the relationship between a human learner and a computer-controlled tutor. In the field of health-care, Brailsford and Schmidt (2003) used the framework to improve a simulation of disease screening. The authors noted that through the use of PECS they were able to incorporate individual behaviour – an important determinant of a patient's attendance at a screening session, a factor that is absent from the majority of models in their field. In another study, Pires and Crooks (2016) used the framework to study the civil war in Sierra Leone and how different levels of security may prevent conflict breaking out (see Appendix A.12 for more details). Pint et al. (2010) used PECS to study how people might turn to organised crime when their needs are unmet based on human needs theory (see Maslow, 1943; Burton, 1979). It has even been used to study the behaviour of burglars (Malleson et al., 2010), allowing different motivations of the burglars to be accounted for.

7.6 Case Study: Simulating Consumer Behaviour Using Probabilistic Rules

The work of Sturley et al. (2018) presents a relatively simple probabilistic model that was built to simulate different patterns of consumer behaviour within a city. This work addressed several key challenges, including: how do we translate observed behaviour into rules that an agent can operate satisfactorily, and which of the many processes involved in this system should be included? These questions were answered through a detailed analysis of a retailer's loyalty card database covering a period of several months. The authors first classified different types of consumer based on key behaviours, including shopping frequency, mission, store choice and spending, as shown in Table 7.3. The resulting seven types of consumer were then linked to geodemographic information to create a picture of where these different consumers lived (for more information about geodemographic information, readers are referred to Harris et al., 2005).

Table 7.3 Summary of customer group characteristics

Consumer type	Store	Frequency type	Time	Distance	Car ownership	Spend
1	Supermarket/ Online	Low	Evening	Average	High	High
2	Supermarket	Low	Weekday daytime	Short	Low	High
3	Convenience	Average	Weekday evening	Short	Average	Low
4	Supermarket/ Convenience	Very high	Weekday	Very Short	Average	Low
5	Supermarket/ Convenience	High	No preference	Average	High	Medium
6	Supermarket	Low	Weekend daytime	Average	Above Average	Medium
7	Supermarket/ Convenience	Low	Weekday evening	Long	Average	Low

Source: after Sturley et al. (2018).

The variables listed in Table 7.3 were converted into simple rules. For example, an agent of consumer type 4 would operate the following rules:

```
IF <need food> = true and <weekday> = true
THEN travel to nearest supermarket or convenience store.
```

And an agent of consumer type 6 would use the following:

```
If <need food> = true and <weekday> = false
THEN travel to nearest supermarket store.
```

To test out the consumer behaviours, a highly abstract representation of the UK, stylised on the city of Leeds, was created within NetLogo as shown in Figure 7.4. This allowed consumers to be located in geographical areas that corresponded to their geodemographic classifications as well as building in real store distributions. These can be seen in Figure 7.4 with the red squares representing different types of stores and the larger squares (in colours other than red) representing different neighbourhoods (based on geodemographics) of Leeds and the different consumer types that reside there.

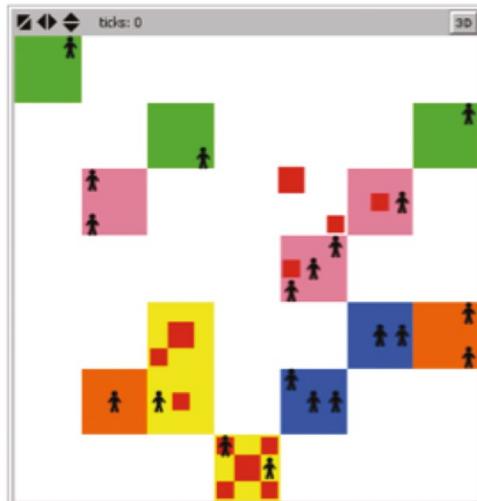


Figure 7.4 Basic spatial environment created within NetLogo that the consumer agents occupy

Another feature of this work is the use of the rich consumer database to calibrate and validate the findings within the model. Whilst a simplified example, this work neatly shows how probabilistic rules can be constructed and tested within a simplified geographical environment constructed within NetLogo (which is available on the accompanying website). This work offers considerable enhancement to the traditionally applied spatial interaction models (as discussed in Chapter 11) which, even after considerable disaggregation, cannot fully capture the complex and individualised spatio-temporal drivers of shopping mission and store choice.

The model (entitled `Store_choice_model`) is available in full in the accompanying online resources.

7.7 Case Study: Simulating Behaviour in Riots Using a Cognitive Model

The work of Pires and Crooks (2017) is an example of the implementation of a cognitive framework, in this case the PECS framework. The work aims to re-create the uprising following the 2007 Kenyan election. Riots are an example of a complex system; they involve a connected, heterogeneous population of individuals coming together to protest for a specific cause. Understanding the motivations behind riots is a complex task. Pires and Crooks (2017) attempted to capture this complexity through the integration of agent-based modelling, GIS and social network analysis. A diverse array of empirical data was used to create the characteristics

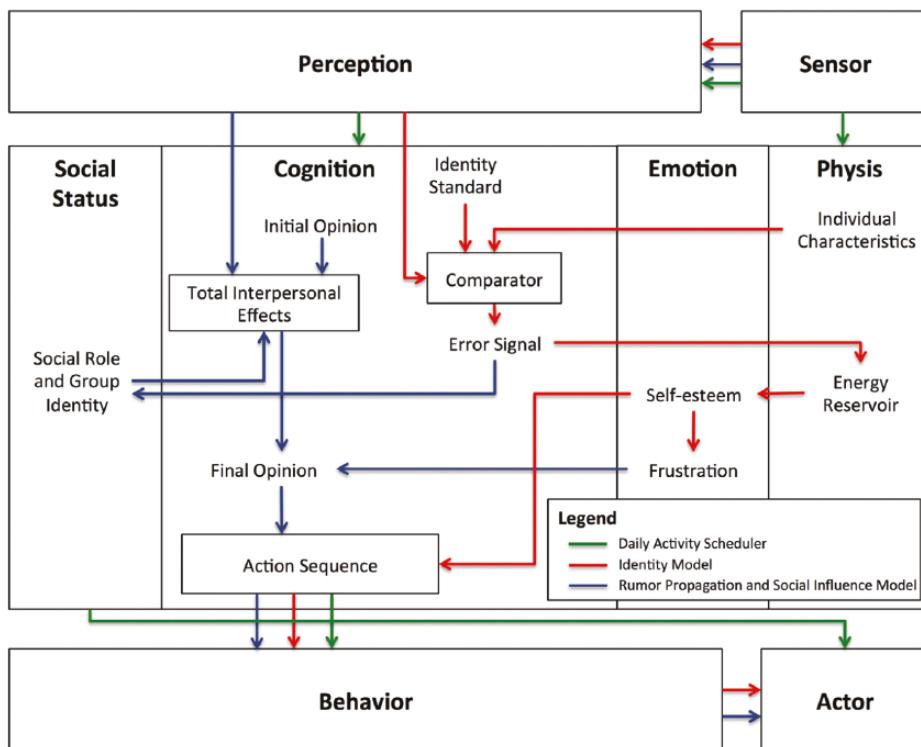


Figure 7.5 A high-level representation of the resident agent behaviour incorporated into the PECS framework (after Pires and Crooks, 2017, which was adapted in turn from Schmidt, 2000)

of the population that were identified as important (such as ethnic diversity), the different agent types (residents and households), and to form the behavioural rules that they operated. In contrast to the simple example in Section 7.6, the behaviours in this example are far more sophisticated and required a behavioural framework that was able to model simple stimulus–response behaviours as well as more elaborate reactive behaviours.

Figure 7.5 shows the three sub-models that were created and embedded within the PECS framework. The aim was to capture the full spectrum of behaviours that theory suggests lead to the emergence of riots: the Daily Activity Scheduler; the Identity Model; and the Rumour and Social Influence Model. These models were selected to simulate the aspects of human behaviour, especially those related to an individual's need for identity and the role rumours play in a person's decision to riot. This provided the foundation to develop the agents' cognitive model, which created a feedback system between the agents' activities in physical space and interactions in social space. By linking agent-based modelling, social network analysis and GIS, the authors were able to develop a cognitive framework for the agents to better represent human behaviour by modelling the interactions that occur over both physical and social space, and capture the nonlinear, reinforcing nature of the emergence and dissolution of riots (further details about this model and a link to the code can be found in Appendix A.3).

7.8 Discussion

One of the central challenges for researchers is understanding the impact of individual human decisions and behaviours over space and time. Until relatively recently, a lack of suitable data has meant that individuals had to be aggregated into large groups to be simulated. This, of course, meant that individuals were assigned the same behaviour and characteristics. For some applications that operate at a larger scale – for example, migration – treating populations as homogeneous groups might be sufficient. However, for many other applications at a micro scale, such as simulating the spread of disease or shopping behaviours, a richer and more detailed understanding of the population is required. In the absence of such data, theories have been developed that try to explain different types of behaviour. These range from general theories such as Maslow's (1943) theory of human motivation or Ajzen's (1991) theory of planned behaviour to more specific theories found within different disciplines – for example, in urban geography there is Christaller's (1933) central place theory and Alonso's (1964) bid-rent theory, while in psychology there is Tajfel and Turner's (1979) social identity theory.

Attempting to understand and then simulate human behaviour is very challenging – so why should we bother? The idea here is related to the notion

that individuals are different from one another and this lies at the heart of agent-based modelling (i.e. such models try to capture this heterogeneity that makes each of us unique). It is not feasible (or desirable) to incorporate every element of human behaviour into a model, as this would give rise to a model that is unwieldy and impossible to explain. However, models need to incorporate important elements of behaviour that drive the phenomenon under investigation; by doing so we can hopefully produce far more realistic simulations that can be used not only by academics, but also by policy-makers. Of course, one of the main challenges is to understand the system well enough to know which behaviours are the most important.

The emergence of big data in the last few years potentially helps shed light on elements of human behaviour and has opened many new research opportunities. It has also allowed researchers to begin to refine existing behavioural frameworks with rich data on individual movements and behaviour. Whilst we are still some way from having robust data at the right resolution, we can begin to revisit the different behavioural frameworks on offer and re-evaluate whether they are adequate for what we need. Many of these frameworks were developed before technologies such as mobile phones and the internet were developed. A rich avenue of study would be the reappraisal of the applicability of these frameworks, taking into account how technological innovations have impacted upon our lifestyles – for example, how we communicate and move around spaces (see Section 12.2.9).

It should also be noted that while this chapter has focused on mathematical and conceptual cognitive models to understand human behaviour, there is a third approach, that of cognitive architectures – for example, Soar (Laird, 2012) and ACT-R (Anderson and Lebiere, 1998) – which focus on abstract or theoretical cognition of one agent at a time, with a strong emphasis on artificial intelligence and cognitive science compared to the other two approaches (Kennedy, 2012). We chose not to focus on them here as such models are rarely applied to more than one or two agents and therefore their utility for large-scale geographically explicit models is currently limited, but they do offer insights into human cognition.

Schlüter et al. (2017) has succinctly outlined many of the challenges associated with implementing behavioural frameworks in agent-based models (which we discussed in Sections 7.1, 7.2 and 7.3.2 and will return to in Section 12.2). Whilst there are several important considerations to be aware of when using these frameworks, they still hold value in providing a way to handle behaviour within our models. One of the biggest challenges with implementing these behavioural models is formalising the theory to be embedded within the model. Here, key assumptions about behaviour have to be made. This is an area where big data might be able to begin to help; careful analysis of such data will allow empirical understanding about individuals to be built in to supplement existing theories.

Chapter Summary

This chapter has provided an overview of the different ways one can incorporate human behaviour within agent-based models. The two main groups of behavioural models that are used to implement behaviour, mathematical and conceptual cognitive, were presented. The most common implementations of these models, such as probabilistic and PECS, were introduced with case studies drawn from published work. Whilst behavioural frameworks provide a useful starting point for implementing behaviour, caution must also be exercised. As discussed, there are numerous challenges that the researcher must address when attempting to simulate human behaviour. The upsurge in new forms of micro-level data (big data) offers a potential solution to several of these challenges and offers a rich vein for understanding behaviour more realistically than has previously been possible.

7.9 Annotated Bibliography

Readers wishing to know more about the issues concerned with modelling human behaviour are referred to:

- Kennedy, W. (2012) Modelling human behaviour in agent-based models. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 167–180. Dordrecht: Springer.

For a discussion of the different types of behavioural frameworks, see:

- Balke, T. and Gilbert, N. (2014) How do agents make decisions? A survey. *Journal of Artificial Societies and Social Simulation*, 17(4), 13. Available at <http://jasss.soc.surrey.ac.uk/17/4/13.html>
- Schlüter, M., Baeza, A., Dressler, G., Frank, K., Groeneveld, J., Jager, W., Janssen, M.A., McAllister, R.R., Müller, B., Orach, K. and Schwarz, N. (2017) A framework for mapping and comparing behavioural theories in models of social-ecological systems. *Ecological Economics*, 131, 21–35.

Readers wishing to know more about how these frameworks have been implemented in published examples are referred to:

- Sturley, C., Newing, A. and Heppenstall, H. (2018) Evaluating the potential of agent-based modelling to capture consumer grocery retail store choice behaviours. *International Review of Retail, Distribution and Consumer Research*, 28(1), 27–46.

- Malleson, N., Heppenstall, A. and See, L. (2010) Crime reduction through simulation: An agent-based model of burglary. *Computers, Environment and Urban Systems*, 34(3), 236–250.
- Pires, B. and Crooks, A.T. (2017) Modeling the emergence of riots: A geosimulation approach. *Computers, Environment and Urban Systems*, 61, 66–80.

Readers wishing to know more about behavioural frameworks in land-use modelling are referred to:

- An, L. (2012) Modeling human decisions in coupled human and natural systems: Review of agent-based models. *Ecological Modelling*, 229, 25–36.
- Filatova, T., Verburg, P.H., Parker, D.C. and Stannard, C.A. (2013) Spatial agent-based models for socio-ecological systems: Challenges and prospects. *Environmental Modelling & Software*, 45, 1–7.

8

NETWORKS

Chapter Outline

Networks play a critical role in our lives. Our movements are constrained to physical networks (e.g. roads), our social networks determine the people we are likely to interact with, and more recently cyber networks (e.g. social media services and mobile telephone networks) constrain our communications. This chapter provides a brief introduction to such networks and shows how they can be integrated into agent-based models. A model is introduced that demonstrates how to navigate agents along a physical road network (this is a common requirement for spatially explicit agent-based models). The chapter concludes with a discussion about how networks can be combined to study real world phenomena.

Box 8.1 Models introduced in this chapter

- Undirected Network Demo and Directed Network Demo: Simple examples illustrating how to create directed and undirected networks (discussed in Section 8.2).
 - Random Network Example: A simple example of how to create random networks (discussed in Section 8.2.2).
 - Networks in NetLogo: A more advanced example of network creation in NetLogo that provides examples of a range of different types of random networks (relevant throughout the discussion in Section 8.3).
 - GMU-Roads: An advanced example of the use of networks, showing how to read in a vector road network and navigate agents around the road network (introduced in Section 8.4).
-

8.1 Introduction

The study and relevance of networks have been brought into much greater focus in the past few years. This is because of an increasing realisation of the role that networks play in driving the processes and dynamics within social and geographical systems. Recent examples include the work of Batty (2013); here Batty explains how cities are the result of potentially finite numbers of networks involving the movement of goods, money, people, etc. With respect to agent-based models, networks provide us with a means to study the connections between people and places. For example, if we just look at census data we see only the numbers and lose the context behind the individuals (Barabási and Bonabeau, 2003) such as who knows whom and the nature and type of interactions. Similarly, if we just look at a map we can see roads, rivers, etc., but it is through network analysis that we can explore these features more deeply – for example, uncovering the optimal route between A and B, or which stream flows into which river.

But what is a network? A network can be roughly defined as a set of components called *nodes* (or vertices) which are somehow connected together by *edges* (or links). It is these edges that allow us to study and explore the interactions between nodes. If we take the nodes and edges together we have a system (i.e. a network or graph), an example of which is shown in Figure 8.1. Here we show a basic network made up of nodes and edges along with two examples: the first is a social network where the nodes are people and the edges are connections between the people (such as one person works with another), while the second example is a depiction of a road network where cars are trying to navigate from one place (i.e. a node) to another via a specific road segment (edge). By using networks we can represent links between agents which are not necessarily located in the same place. The study of networks has been gaining significant interest in agent-based

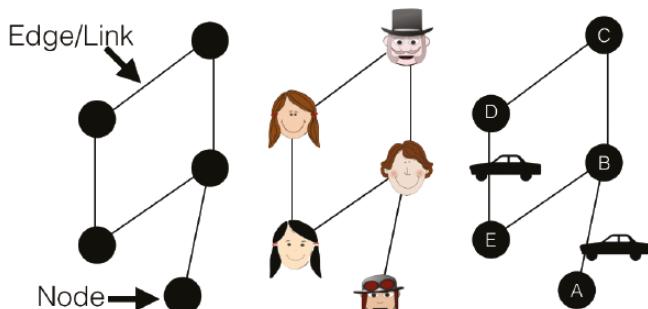


Figure 8.1 From left to right, a basic graph, a social network and a simplified transportation network

modelling as the network typology can impact on the behaviour of agents and the outputs of models (Alam and Geller, 2012).

Throughout this book we have demonstrated the importance of social interactions of agents with respect to the emergence of higher-level phenomena. However, the focus has been on Euclidean interactions (i.e. with the agents' surrounding neighbours). Social networks allow agents to communicate with each other beyond the boundaries of their physical situation (Alam and Geller, 2012). Some researchers refer to such agents as socially embedded (e.g. Granovetter, 1985; Edmonds, 2006): agents' behaviour can be influenced by the network of social connections. Often in agent-based modelling, physical space (i.e. adjacent cells) is used as a proxy for social space, as we have shown in many of the examples in this book. But using networks we can dig deeper into this assumption and specifically explore actual relationships that can be observed in reality.

8.2 Basic Network Properties

8.2.1 Defining Graphs Mathematically

By abstracting a network to a graph we can explore mathematically the connections between people and/or places. A graph (G) is composed of nodes/vertices (V) and edges (E). Edges are a subset of $V \times V$. If $E = V \times V$ the graph is *complete* (i.e. there are the same number of edges as nodes). Often the terms *network* and *graph* are used interchangeably, but here we will attempt to be consistent. The term 'network' will be used when referring to a real system (such as the World Wide Web or a social network) while 'graph' will be used when referring to a mathematical representation of a network (such as links between web pages in web graphs). The mathematical aspects of graphs (i.e. graph theory) date back to Euler (1741) and the seven bridges of Königsberg (now Kaliningrad) (Alexanderson, 2006). The question that Euler considered was whether it was possible to walk across the seven bridges and arrive

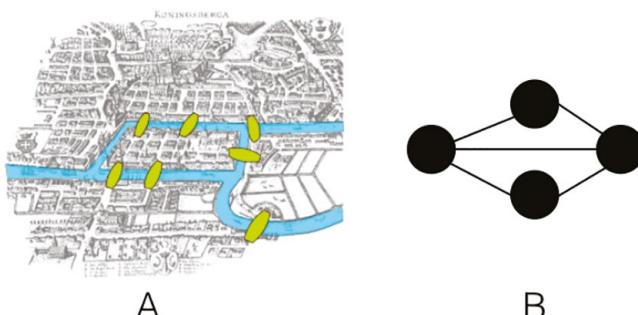


Figure 8.2 The seven bridges of Königsberg problem: (A) the physical depiction of the bridges (Giuşcă, 2017); (B) the graph representation

back at the same starting point without ever crossing a bridge more than once. In this pioneering work, Euler abstracted the seven real world Königsberg bridges (Figure 8.2A) as a graph, whereby the land (the two banks of the river and the islands) are represented as *nodes* and the bridges as *links* (Figure 8.2B). By carrying out such an exercise Euler was able to prove that it was not possible to traverse all the bridges without repeating (walking) over a path (edge).

8.2.2 Building Graphs in NetLogo

What is particularly appealing about graphs is that the *links* between nodes can have different properties. For example, the links can be *undirected*, such that objects or information can flow in both directions (e.g. counties that share a common border, or actor A was in a movie with actor B). Alternatively, they can be *directed* (e.g. cars can only drive one way down a road, or a professor who lectures to students). By understanding the connections (ties) within a graph we can thus explore the nature of the relationships.

Graphs in NetLogo can be created using the concept of *links*. Links are used to connect two turtles and are actually a kind of turtle themselves, but without a spatial location (they are not on any particular patch, and it is not possible to calculate the distance from a link to a patch or turtle). The command `create-link-with` creates undirected links, and `create-link-to` or `create-link-from` create directed links. These are distinguishable in the NetLogo display because undirected links are displayed as lines, whereas

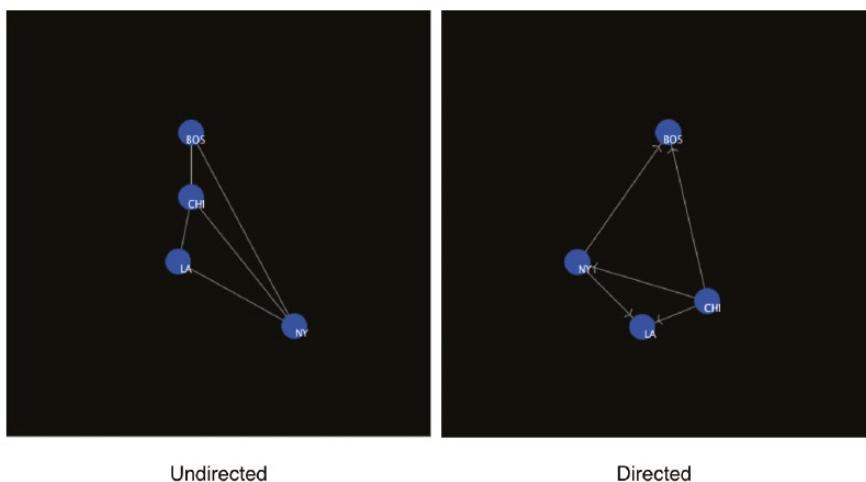


Figure 8.3 Directed and undirected links as displayed in NetLogo. These examples are from the Undirected Network Demo and Directed Network Demo in the accompanying resources

directed links look like arrows. Figure 8.3 illustrates this. The NetLogo [links documentation](#) is very comprehensive and worth reading,¹ but this section will briefly discuss some of the most relevant code that can be used to use links to create networks. The examples are drawn from the two models Undirected Network Demo and Directed Network Demo (both of which are available in the accompanying resources).

The models both begin by creating four turtles at random places in space. These are given identifiers that represent US cities to distinguish them ("LA", "CHI", "BOS", "NY") although these names are not important; the turtles could represent any typical agent-like object. After these turtles have been created, it is possible to create links between them. The code that illustrates how they can be linked is available in [Code Example 8.2.1](#). That example shows how to make undirected edges. To make directed edges, the `create-link-with` commands simply need to be replaced with `create-link-to`. The code itself is straightforward, so needs little explanation. It is, however, worth briefly noting that regardless of which command is being used, we have to use `one-of` to choose a single turtle to make the link with:

```
create-link-with one-of turtles with [ ID = "LA" ]
```

Even though we know that there will only ever be one turtle for each city identifier ("LA", "CHI", "BOS", "NY"), the `turtles with` commands still return an *agent set* (i.e. a list of turtles). The `one-of` command chooses one of those (even though, in this case, there is only ever one turtle that will be chosen).

8.2.3 Adjacency Matrices and Node Degree

To grapple with connections we need to be able to represent the graphs mathematically. We can do this through an adjacency matrix, as shown in Figure 8.4. In this figure we show two graphs (undirected and directed) along with their corresponding adjacency matrix. In these matrices, a 1 denotes a link (i.e. an edge) between two nodes and a 0 shows no connection; i refers to the i th row of the matrix and j refers to the j th column. While the networks might appear similar, they are not. Examining the connections on the directed network shows that they are not symmetrical (i.e. relations are not reciprocal). By summing up the rows and columns a 'degree' can be calculated for each node (i.e. the degree of connections each node has). Referring to Figure 8.4 for the undirected graph, the 'degree' for node 5 is 2, while the degree for node 5 in the directed graph is 1 (as it's only connected to node 2). The degree therefore gives us the size of the neighbourhood for each node.

¹ <https://ccl.northwestern.edu/netlogo/docs/programming.html#links>

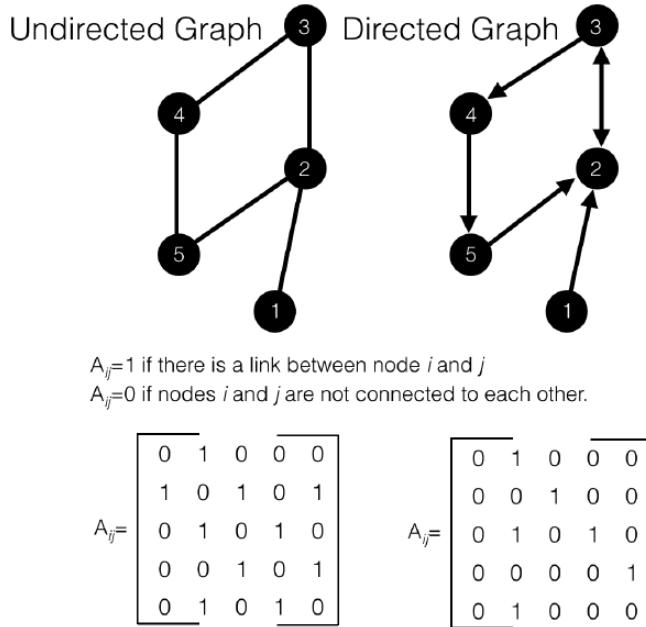


Figure 8.4 Adjacency matrix for undirected and directed graphs

8.2.4 Traversing Graphs

While nodes might be directly connected, they can also be indirectly connected through a series of links. A *walk* is a sequence of nodes that can be visited by following edges. A *trail* is a walk with no repeated edges. Finally, a *path* is a walk with no repeated edges or nodes.

For example in the undirected graph of Figure 8.4, nodes 1 and 5 are not connected directly by an edge, but they are connected by a path: the path 1,2,5 of length 2, or 1,2,3,4,5 of length 4. The *distance* between nodes 1 and 5 is the length of the shortest path between two points (often referred to as the *geodesic*), which in the undirected case is 2 (i.e. 1,2,5).

These concepts can also be applied to directed graphs; however, one also has to consider the direction of the edge, which may be one-directional or bi-directional. The degree in a directed network has two distinct elements, the *indegree* and *outdegree*, which are defined by the number of edges coming into and going out of the node respectively. For example, in the directed graph of Figure 8.4, node 2 has an indegree of 3 (i.e. nodes 1, 3 and 5 flow into it) and an outdegree of 1 (i.e. node 3). This is reflected in the adjacency matrix: the row sum for node 2 is 1 (the outdegree) and the column sum for node 2 is 3 (the indegree). Similarly, paths also change with directed networks; for example, to go from node 1 to 5

as we did in the undirected network we must also consider how the nodes are connected. Moving from 1 to 5 in the directed network example would involve passing through nodes 1,2,3,4,5, which is also the shortest path.

Code Example 8.2.1 An example demonstrating how to create undirected links to join agents. This is from the Undirected Network Demo model. The Directed Network Demo model is almost identical, and shows how to create directed links

```
to setup-links
  ;; This function sets up the links between the turtles

  ;; Chicago will connect to LA, BOS and NY
  ask turtles with [ ID = "CHI" ] [
    create-link-with one-of turtles with [ ID = "LA" ]
    create-link-with one-of turtles with [ ID = "BOS" ]
    create-link-with one-of turtles with [ ID = "NY" ]
  ]

  ;; New York connects to LA and BOS
  ask turtles with [ ID = "NY" ] [
    create-link-with one-of turtles with [ ID = "LA" ]
    create-link-with one-of turtles with [ ID = "BOS" ]
  ]
end
```

8.2.5 Graph Density

Another widely used concept in graph theory is *density*, which describes the overall level of linkages within the graph (ranging from 0 to 1). A complete network would be one where all nodes are connected to all other nodes (i.e. there would be no 0s in the adjacency matrix except on the diagonal, as we assume a node is not connected to itself), which is rare in large networks. In contrast, a sparse network would be one where the nodes had very few connections. Density therefore allows one to summarise the overall distribution of connections compared to a fully connected network. It is calculated as the actual number of connections divided by the potential number of connections:

$$\text{density} = \frac{e}{n(n - 1) / 2}$$

where e is the number of edges and n is the number of nodes present. The undirected network of Figure 8.4 would therefore have a network density of 0.5. In undirected networks, if we sum all the 1s and divide by 2 we get the number of edges. To get a density of 1 in an undirected network, we would need each

pair of nodes to be connected (which would mean 10 edges in the undirected network of Figure 8.4).

In a directed graph, the calculation of density is slightly different as connections are not necessarily symmetrical. Therefore the maximum density in a directed graph relates to the number of pairs of nodes that it contains:

$$\text{density} = \frac{e}{n(n-1)}$$

So for the directed network shown in Figure 8.4 our network density would therefore be $6/(5 \times 4) = 0.3$. However, to calculate the number of edges here from the adjacency matrix all we need to do is sum the number of 1s. To get a density of 1 in the directed network, we would need each pair of nodes to be connected (e.g. 20 edges).

8.2.6 Calculating Node Importance

So far we have focused on the overall network characteristics. However, there are also measures that can be used to describe individual nodes within a network. By focusing on the nodes specifically one can identify, say, the most *influential* person in a social network or the key junction in a road network. This is what we turn to next; the relative *importance* of a node in a graph. There are various ways to identify the importance of a node via various centrality measures, such as degree centrality, betweenness centrality, closeness centrality and eigenvector centrality (see Wasserman and Faust, 1994). Different centrality measures give us different pictures of the network. In this subsection we look at degree centrality and betweenness centrality.

The *degree centrality* is equivalent to the degree of the node, as discussed in Section 8.2.3. In the case of a directed graph, degree centrality is often split between *indegree* (showing how popular the node is) and *outdegree* (how outgoing the node is). For example, node 2 in the directed graph in Figure 8.4 has an indegree of 3 (i.e. the sum of column 2 of the adjacency matrix) and an outdegree of 1 (i.e. the sum of row 2 of the adjacency matrix).

Betweenness centrality (Freeman, 1977) is a calculation of the number of times a node acts as bridge in the shortest path between two other nodes (i.e. how central the node is). Betweenness centrality, $g(n)$, can be calculated as follows:

$$g(n) = \sum_{i \neq n \neq j} \frac{\sigma_{ij}(n)}{\sigma_{ij}}$$

This says that the betweenness centrality of node n is the summation of the geo-desic (shortest) path between any two nodes i and j via n , expressed as a fraction of

the total number of geodesic paths between i and j . Using the networks presented in Figure 8.4, we can calculate the number of paths and geodesics in the graph.

However, as the number of paths in a graph can be affected by the size of the graph we can also normalise the betweenness centrality by the total number of ordered node pairs in the graph. By normalising the betweenness centrality we can then compare centrality values across graphs. To rescale the betweenness centrality we need to rescale the number of pairs of nodes, not including the node in question, so that g ranges between 0 and 1. The division is by $(N - 1) (N - 2)$ for directed graphs and $(N - 1) (N - 2) / 2$ for undirected graphs, where N is the number of nodes in the entire graph.

As the undirected graph in Figure 8.4 contains five nodes, there are 20 possible paths (pairs), 10 distinct paths (as discussed above) and 13 unique geodesics. We now need to find the shortest path (i.e. the geodesic) between each node pair. We record this in the fourth column of Table 8.1. As this graph is relatively small, we can do this by hand, but for larger graphs we might want to consider automation such as breadth-first search, depth-first search or Dijkstra's algorithm (see Tsvetovat and Kouznetsov, 2011, for examples). Based on the fourth column we can calculate the betweenness centrality of any node of the graph. So if we take node 2 as an example, $\sum \sigma_{ij}(n_2)$ is 3.5. Note the 0.5 comes as a result of pair 12, from node 3 to node 5,

Table 8.1 Pairs through the network

Pair	From	To	Geodesic	$\sigma_{ij}(n_2)$
1	1	2	{1,2}	0
2	1	3	{1,2,3}	1
3	1	4	{1,2,3,4; 1,2,5,4}	1
4	1	5	{1,2,5}	1
5	2	1	See pair 1	–
6	2	3	{2,3}	–
7	2	4	{2,3,4; 2,5,4}	0
8	2	5	{2,5}	0
9	3	1	See pair 2	–
10	3	2	See pair 6	–
11	3	4	{3,4}	0
12	3	5	{3,4,5; 3,2,5}	0.5
13	4	1	See pair 3	–
14	4	2	See pair 7	–
15	4	3	See pair 11	–
16	4	5	{4,5}	0
17	5	1	See pair 4	–
18	5	2	See pair 8	–
19	5	3	See pair 12	–
20	5	4	See pair 16	–

where we can go between either nodes 3,4,5 or nodes 3,2,5. If we wish to normalise this undirected graph to get a value between 0 and 1 it would be:

$$g(n_2) = \frac{3.5}{\frac{1}{2}(5-1)(5-2)} = \frac{3.5}{6} = 0.58$$

For directed networks the betweenness centrality for each node can be calculated similarly, while bearing in mind that a path from A to B in a directed network might not be the same as that from B to A.

Betweenness centrality is an important consideration for the flow of communication; a person who has a high betweenness centrality can control the flow of information, or by removing a key node from the network could cause information to flow at a slower rate through it or isolate nodes. For example, if we removed node 2 from the undirected network, node 1 would be isolated.

Box 8.2 Importing network data

NetLogo provides examples of how one can import network data – see the NetLogo Network Import Example (Wilensky, 2017) for an example of how to read in nodes and links from a .txt file and use them for the initialisation of an agent-based model. Alternatively, one could also use NetLogo's Network Extension toolkit² for building networks as we show in Section 8.3.

8.3 Social Networks

Over the past decade social network analysis has grown in popularity within the social sciences (see Borgatti et al., 2009). This is partly due to both increases in the amount of data available to describe social networks (e.g. through social media) and the increase in computational power available to analyse such data. Network analysis makes it possible to explore ties such as *similarities* (e.g. membership in the same clubs), *social relations* (e.g. kinship), *interactions* (e.g. talked with) and *flows* (e.g. information) (see Borgatti et al., 2009). Knowing about the properties of networks, especially in the context of ‘real societies’, and representing them in agent-based models helps to understand not only how resilient they are to change, but also how information (or physical actors such as diseases etc.) can spread through them.

Three well-known models for generating stylised social network structures are: random networks (Rapoport, 1957; Gilbert, 1959; Erdős and Rényi, 1960), small-world networks (Watts and Strogatz, 1998) and scale-free networks

² <https://github.com/NetLogo/Network-Extension>

(Barabási and Albert, 1999). There has been a great deal of attention devoted to the generation of these networks because the artificially created networks share properties with real world networks and are therefore ideal for running experiments (or for including in models). For example, Alizadeh et al. (2017) explored the generation of social networks with respect to agents in a physical geographical space, and Hamill and Gilbert (2009) built social networks based on personal connections that drew on the idea of social circles (Simmel, 1902), that is, a person's social reach. In this section, brief descriptions of each network will be provided, with an account of how they differ.³ The rationale for this discussion is that understanding the basic structure of a network (random, small-world, scale-free) makes it easier to understand its behaviour. For example, a graph's diameter or path length (i.e. the largest number of hops needed to get from one node to another via the shortest possible route) will differ depending on the type of network structure. This in turn will influence how long it takes, for example, a disease to propagate through a given network.

Box 8.3 Networks in NetLogo

Using links (as discussed in Section 8.2.2), it is possible to easily create graphs in NetLogo. However, there are a range of graph structures that might be appropriate for a given domain. For example, random networks (see Section 8.3.1) form a good baseline from which to compare other graph structures, small-world networks (see Section 8.3.2) are suitable for many social networks, and scale-free networks (see Section 8.3.3) are appropriate for modelling networks that grow over time such as the World Wide Web or the electricity grid. The Networks in NetLogo model, available in the accompanying online resources, provides the functionality to visualise a number of different network types, along with the NetLogo code that can be used to create the networks.

8.3.1 Random Networks

Simply stated, a random network is one where nodes are randomly connected based on a constant probability, p , of a link between any two nodes being created.

³ It should be noted that while these networks are used here in the context of social networks, they are also structures that are observed within physical networks. For example, the US highway system resembles a random network while the US airline system is similar to a scale-free network (Barabási and Bonabeau, 2003).

Therefore most nodes will have the same number of links, and this generally follows a Poisson distribution. An example of this is shown in Figure 8.5(A). On the left is the network laid out as a circle containing 25 nodes, and on the right are two charts. The first chart shows the degree centrality, the number of links per node; the second shows the degree distribution, the number of nodes with a specific number of links (this resembles a normal distribution). In random networks, for a given number of nodes, N , the average path length is approximately $\log(N)$ (which is low in comparison to other types of network), while the average degree for any random network is $p(N - 1)$. One of the notable issues of random networks is that they fail to generate high clustering coefficients (Hamill and Gilbert, 2009), so are unlikely to contain groups of highly connected nodes (as you might find in a social network, for example). Generating random networks in NetLogo is straightforward, as illustrated in Code Example 8.3.1.

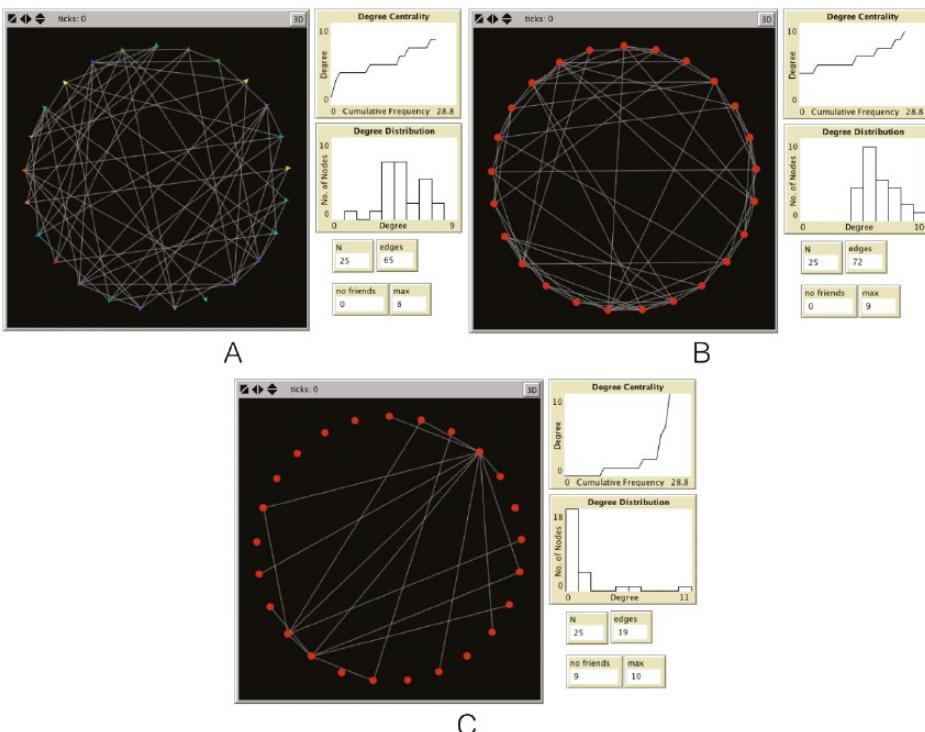


Figure 8.5 Different network structures based on 25 nodes, with 0.2 probability of a link: (A) random; (B) small-world; (C) scale-free

Code Example 8.3.1 A demonstration of how to create simple random networks in NetLogo (from Random Network Example in the NetLogo Models Library)

```
to setup-simple-random
  clear-all

  ;; Make a circle of turtles
  create-turtles num-nodes
  layout-circle turtles (max-pxcor - 1)

  ;; Now make links one at a time until we have enough
  while [count links < num-links] [
    ;; Note that if the link already exists, nothing happens
    ask one-of turtles [ create-link-with one-of other turtles ]
  ]
end
```

To create the random network as devised by Erdős and Rényi (1960), replace the while loop above with the following (full code available in the Random Network Example model):

```
ask turtles [
  ;; we use "self > myself" here so that each pair of turtles
  ;; is only considered once
  create-links-with turtles with [
    self > myself and random-float 1.0 < probability
  ]
]
```

One of the criticisms of random networks is that they are unrealistic (Alam and Geller, 2012). Many social and physical systems cannot be captured using the simple random network structure – this will become clear when the other common structures are discussed. This does not, however, mean that simple random networks have no value. They can act as a baseline (i.e. a test case) with which to compare against other network structures. Both small-world networks (Section 8.3.2) and scale-free networks (Section 8.3.3) can be considered special forms of random networks (see Oldham, 2017, for a discussion), and ones that can be much more appropriate for representing real networks.

8.3.2 Small-World Networks

Small-world networks are a type of graph in which nodes form tight clusters of dense interconnections, with occasional links between these clusters. This results in most nodes not being directly connected, but having relatively short connections between apparently distant nodes. In human societies, there is evidence that any two people on the planet can be connected by six or fewer steps

(i.e. through friends of friends), commonly termed ‘six degrees of separation’ (Milgram, 1967). After Milgram’s work we had to wait three decades until Watts and Strogatz (1998) proposed an algorithm to generate a small-world network that exhibited the correct long-range correlations, high clustering and average shortest path lengths. The algorithm starts from a regular lattice and then rewire each pair of nodes with a probability p . Figure 8.5B shows such a structure. Here you can see that there are certain nodes forming cliques (i.e. clusters of nodes which are well connected), and that there are some nodes that ‘reach across’ to other cliques. This is an important network structure in the study of human systems because it fits many real networks (both physical and social) very well.

8.3.3 Scale-Free Networks

Barabási and Albert (1999) showed how many networks such as the World Wide Web, actor collaboration networks (e.g. the Six Degrees of Kevin Bacon game) and power grids are *scale-free*. These networks are called scale-free because a similar structure would be observed whether they were being observed from near or far; the shape of the network does not change with scale. Their node degrees (i.e. the number of connections into each node) follows a *power law distribution*; there are many nodes with very few connections and a few nodes with a very large number of connections. More formally, the probability that a node has links with k other nodes decays as a power law probability density function. Barabási and Albert (1999) suggest that growth and *preferential attachment* are the two mechanisms that explain the scale-invariant behaviour (*scaling*) of real networks.

In random and small-world networks, *hubs* (nodes with very large numbers of connections) are not possible, but this is not the case for scale-free networks. This can be explained by the mechanism that ultimately constructs the network. In random networks, all nodes are present at the start and links are added iteratively. In contrast, the nodes in scale-free networks are added iteratively, so nodes that have been present in the network for the longest amount of time are more likely to have many more connections than those that have only recently been added. For example, a node can be a webpage, and links can be hyperlinks to that page over time, such that the longest-standing webpages are likely to have the most links (see Barabási and Bonabeau, 2003, for more details).

An example of the growth of a scale-free network is given in Figure 8.6 using NetLogo’s Preferential Attachment model (Wilensky, 2005b).⁴ It starts with two nodes and one link and adds new nodes over time. New nodes prefer to be attached to existing nodes with numerous links. Over time, this preferential attachment will result in hubs developing. Such growth of a network is one

⁴ The model is available at <http://ccl.northwestern.edu/netlogo/models/PreferentialAttachment>

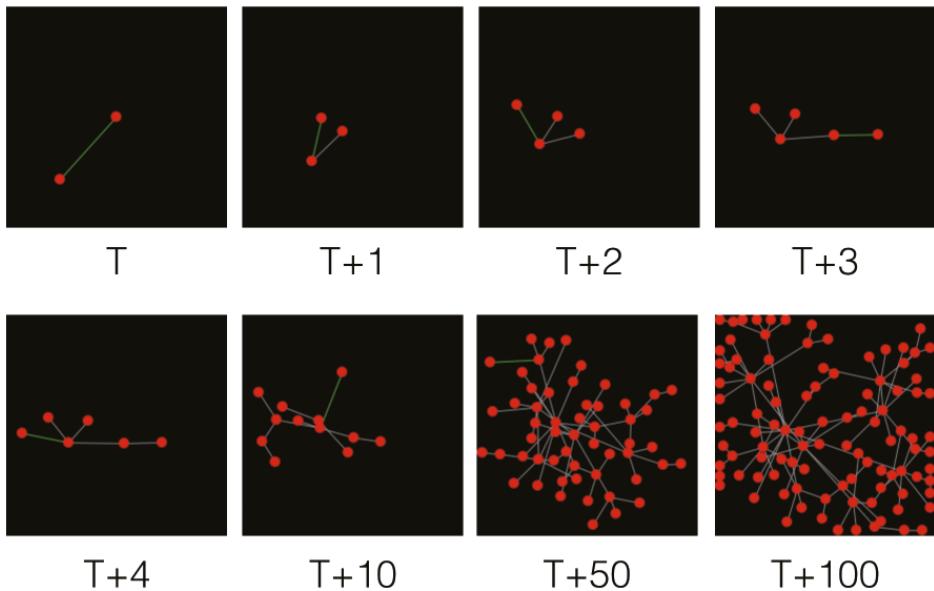


Figure 8.6 Growth of a scale-free network via preferential attachment (Wilensky, 2005b)

reason why well-cited scientific papers acquire more citations over time and why popular websites get more links (Barabási and Bonabeau, 2003). In Figure 8.5(C) the growth of a scale-free network is illustrated. There are a few nodes with a high degree, but most nodes have only one link. This highlights preferential attachment (i.e. the rich get richer) as new agents are added to the simulation.

8.4 Transport Networks: Agents Navigating a Road Network

Connections between places have always been important to human societies and geographers have long studied them (see, for example, Haggett and Chorley, 1969). From the time of the earliest cities, physical and social networks have played a crucial role in how people have moved around. In the introduction to this chapter it was demonstrated how Euler (1741) abstracted the real world into a series on nodes and links to solve the seven bridges of Königsberg problem. More broadly, the ability to formally encapsulate a physical transport infrastructure is critical for many empirical applications. This is especially true for studies of human phenomena, where spatial movements are often restricted to a transport network. As an example, consider a network that represents walkways (i.e. footpaths) for an area as shown in Figure 8.7. To model the flows of pedestrians around the area, an agent-based model will need to incorporate this network of paths and use them to restrict the movements of the agents.

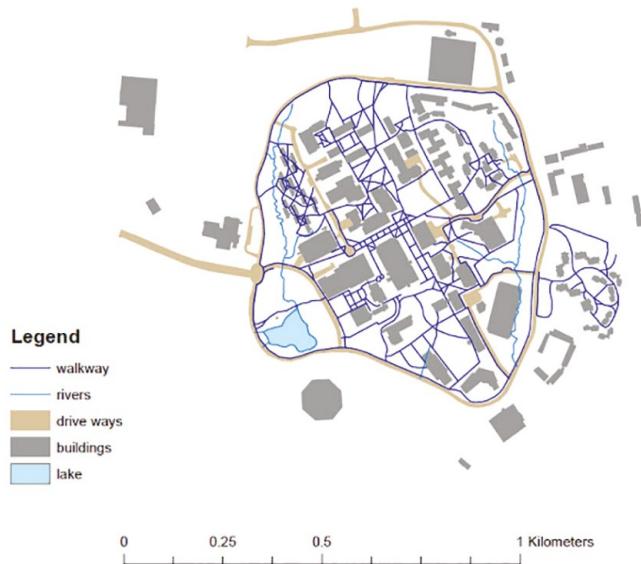


Figure 8.7 Network representation within a GIS, with two types of networks: walkways and rivers

Box 8.4 Routing in GIS and agent-based models

An important aspect of network analysis that is commonly explored in a GIS is that of routing – specifically, finding the shortest path from an origin to a destination that minimises the cost (e.g. in terms of time, distance or financial cost). Finding optimal routes can be computationally very expensive, so the most efficient route-finding algorithms are those that are able to find good routes without having to test every possible route between an origin and a destination. Several algorithms have been designed to solve routing problems. These include Dijkstra's (1959) algorithm, Dantzig's (1960) and others – see de Smith et al. (2009) for more details. Similar routing algorithms are used in the large-scale traffic simulations of MATSims (Horni et al., 2016) and TRANSIMS (Barrett et al., 2009) for trip planning, along with a number of agent-based models exploring humanitarian relief (Crooks and Wise, 2013; see also Appendix A.8 below), evacuation modelling (Dawson et al., 2011), the movement of criminals (Malleson et al., 2013) and navigating taxis through street networks (Manley et al., 2014). The GMU-Roads model, which will be introduced below, uses an algorithm called A-star for routing.

The remainder of this section will discuss the most important parts of the NetLogo code that are required to implement a model based on the vector network of

walkways presented in Figure 8.7. The full model, entitled GMU–Roads (so named because it models the walkways around the George Mason University campus) is available in the accompanying resources. Figure 8.8 illustrates how the model will appear once the data have been loaded into NetLogo.

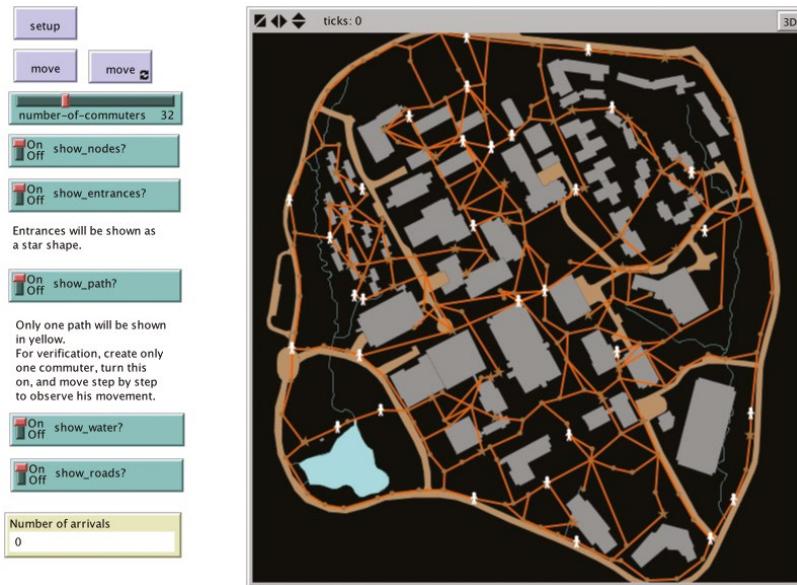


Figure 8.8 A road network imported into NetLogo, from the GMU–Roads model

The GMU–Roads model is one of the most complicated that will be introduced in this book. That said, the code itself is still relatively straightforward so should be understandable by all readers. However, it is recommended to review Chapter 6 first. That chapter discussed, in detail, how to incorporate vector GIS data into NetLogo models.

Introducing the GMU–Roads Model

The model demonstrates how to use the A-star algorithm, implemented in NetLogo, to move agents along a vector road network. The input data represent George Mason University, including the buildings, walkways, driveways and waterways (although the model can be generalised by loading data for a different area). To begin with, each agent (commuter) randomly selects a destination building and then finds the shortest path (in terms of distance) to that destination. The agents cover one node (road junction) in each model iteration. Once their destination has been reached, they remain there for one iteration before choosing a new, random, destination. The purpose of the model is not to create a useful simulation

of movements around a campus, but to demonstrate how agents can find their way using a vector road network, and have their movements restricted to roads.

Breeds and Variables

As with all models that use the **NetLogo GIS extension**,⁵ the first instruction in the model is to activate the extension:

```
extensions [gis]
```

The model defines two different types ('breeds') of agent: *commuters* are the 'people' who will navigate around the environment, and *vertices* represent the nodes in the road network – the junctions/intersections at which two roads meet.

```
breed [commuters commuter]
breed [vertices vertex]
```

The model also defines a range of essential global, patch and turtle variables. The purpose of many of these variables (such as the global variables to store all of the required GIS data) has been discussed previously. Code Example 8.4.1 outlines the key variables that are required to implement the routing behaviour.

Loading GIS Data

The model loads the GIS vector data with `gis:load-dataset` in the usual way, and then displays the layers using the `gis:set-drawing-color`, `gis:fill`, and `gis:draw` commands. Because the agents always choose a building as their destination, it is necessary to find the patch that covers the centroid of each building. In this way, the agents will know where the buildings are. The method to do this is very similar to that provided in Code Example 6.4.12 (page 165), where the patches that intersected the centroids of polygons were found. The line `building ->` tells NetLogo that we will use the word 'building' to refer to the vector objects as we iterate over them (they represent buildings after all). If the syntax of the `foreach` loop is unfamiliar, refer back to Box 6.5 (on page 153).

```
;; identify centroids and assign IDs
foreach gis:feature-list-of gmu-buildings [
  building ->
  let center-point gis:location-of gis:centerid-of building
  ask patch item 0 center-point item 1 center-point [
    set centroid? true
    set id gis:property-value building "Id"
  ]
]
```

⁵ <https://ccl.northwestern.edu/netlogo/docs/gis.html>

Code Example 8.4.1 Variables used in the GMU-roads model to implement routing behaviour

```

patches-own [
    centroid? ;; is it the centroid of a building?
    id          ;; if it is a centroid of a building, it has a building ID
    entrance   ;; nearest vertex on road. only for centroids.
]

commuters-own [
    mynode           ;; a vertex. where the agent begins its trip
    destination      ;; the destination that it wants to arrive at
    destination-entrance ;; the entrance of the destination on the road
    mypath          ;; an agentset containing nodes to visit
    step-in-path    ;; the number of steps taken in the walk
    last-stop       ;; final destination
]

vertices-own [
    myneighbors ;; agentset of neighbouring vertices
    entrance?   ;; if it is an entrance to a building

    ;; The following variables are used and renewed in each path-selection
    dist        ;; distance from original point to here
    done        ;; 0, or 1 if shortest path goes through this point
    lastnode   ;; last node to this point in shortest path
]

```

Creating the Road Network

Code Example 8.4.2 Iterating over the features of the road network and creating links between the nodes

```

;; Iterate over every road in the walkways data
foreach gis:feature-list-of gmu-walkway [
    road-feature ->

    ;; for the road feature, iterate over the vertices that make it up
    foreach gis:vertex-lists-of road-feature [
        v ->

        let previous-node-pt nobody ;; The previous node; used to link nodes together

        ;; for each vertex, iterate over its individual points
        foreach v [
            node ->

```

```

;; Find the location of the node and create a new 'vertex' agent there
let location gis:location-of node
if not empty? location [
  create-vertices 1 [
    set myneighbors n-of 0 turtles ;empty
    set xcor item 0 location
    set ycor item 1 location
    set size 0.2
    set shape "circle"
    set color brown
    set hidden? true

    ;; create a link to previous node
    ifelse previous-node-pt = nobody [
      ;; first vertex in feature, so do nothing
    ] [
      create-link-with previous-node-pt ;create link to previous node
    ]
    ;; remember *this* node so that the next one can link back to it
    set previous-node-pt self
  ]
]
]
]
]

```

After having loaded the GIS data, one of the more complicated tasks can begin: converting the walkways into a usable road network. Code Example 8.4.2 provides the code in full. It looks especially complicated because there are three ‘nested’ `foreach` loops. However, the logic itself is reasonably straightforward.

Line features can be broken down into a list of separate vertices. These represent the individual points that are chained together to make the whole feature. Knowing this makes it possible to break down a line into its constituent parts and represent them in NetLogo. Broadly, the method for converting vector road data into a usable network in NetLogo is as follows:

1. Use `gis:feature-list-of` and a `foreach` loop to iterate over every road feature in the vector data set.
 2. Use `gis:vertex-lists-of` (see Box 8.5) and a second `foreach` loop to split the line into a list of vertices that, together, make up the line.
 3. Use a third `foreach` loop to iterate over each of these individual vertices (points), create a ‘vertex’ agent to represent the point, and link the adjacent points together using `create-link-with`.

Box 8.5 gis:vertex-lists-of

'Reports a list of lists of Vertex values. For point datasets, each vertex list will contain exactly one vertex: the location of a point. For line datasets, each vertex list will contain at least two points, and will represent a "polyline", connecting each adjacent pair of vertices in the list. For polygon datasets, each vertex list will contain at least three points, representing a polygon connecting each vertex, and the first and last vertices in the list will be the same.'

Source: [gis:vertex-lists-of documentation](#).⁶

Now try to read through Code Example 8.4.2, keeping in mind what the purpose of each of the `foreach` loops is.

Moving Agents along the Road Network

Once the road network has been created it is possible to implement the behaviour of the agents such that they only move along roads in the network. In this model, this is accomplished by restricting the movements of the commuters to the locations of the individual `vertex` objects.⁷ There are two stages required to implement this behaviour, and each stage will be explained separately below. Code Example 8.4.3 contains the full code for the `move` procedure.

The first stage is to find the commuters that do not have a destination. This occurs at the beginning of the simulation, or when they have reached their destination during the previous iteration. A commuter's current destination is stored as a variable, so we can simply check whether or not this variable is empty, which is the same as being equal to '`nobody`' (i.e. it has no value):

```
ask commuters [
  if destination = nobody [
```

If the agent has no destination, then it needs to be given a new building destination, chosen randomly. Recall that all patches have a variable called `centroid?`, and when the model was initialised, all patches that intersected with the centroid

⁶ <https://ccl.northwestern.edu/netlogo/docs/gis.html#gis:vertex-lists-of>

⁷ The vertices are technically agents but, as they are designed to represent thoughtless spatial objects, rather than autonomous beings, they will continue to be referred to as 'objects'.

of a building had this set to `true`. Therefore finding a random building is as simple as asking NetLogo to return a patch that has `centroid? = true` as follows:

```
set destination one-of patches with [centroid? = true]
```

There are some other administrative tasks that must be completed (see Code Example 8.4.3). Importantly, these include finding a path from the current location to the new destination. The procedure that does this (called `path-select`) will be discussed shortly.

The second stage in the `move` procedure involves checking whether the agent reached their destination in the last iteration and, if not, moving one step along the path. It is possible to check whether the agent is at its destination by comparing its (x, y) location to that of the destination. If the coordinates are the same then it must have reached the destination during the previous iteration:

```
ifelse xcor != [xcor] of destination-entrance or  
ycor != [ycor] of destination-entrance [
```

If the agent is not at its destination then it can move. The variable `mypath` is a list that contains all of the vertices that the agent must pass through, and `step-in-path` is a counter that records how far through the list the agent has got (i.e. how far along the path they have travelled to their destination). Both of these will be introduced when the routing algorithm is discussed below. The code to move along the path is:

```
move-to item step-in-path mypath  
set step-in-path step-in-path + 1
```

Code Example 8.4.3 Code from the GMU-Roads model that demonstrates how agents can move along a road network, and chose a new destination if they need to

```
to move  
;; Stage 1. For all agents who do not have a destination yet, they need to  
;; pick one and plan a route there  
ask commuters [  
  if destination = nobody [  
    ; This commuter does not have a destination.  
    ; Find a patch that represents a building centroid  
    ; and make this the destination  
    set destination one-of patches with [centroid? = true]  
    ; Also find the closest entrance to the building  
    set destination-entrance [entrance] of destination  
    ; (but make sure the destination isn't the  
    ; same as the agent's current position)  
    while [destination-entrance = mynode] [
```

```

        set destination one-of patches with [centroid? = true]
        set destination-entrance [entrance] of destination
    ]
    ;; Now calculate the shortest path to the destination
    path-select
]
]
;; Stage 2. Move commuters along the path selected
ask commuters [
    ;; See if the agents are at their destination
    ifelse xcor != [xcor] of destination-entrance or
        ycor != [ycor] of destination-entrance [
            ;; They are not at the destination. Move along the path.
            move-to item step-in-path mypath
            set step-in-path step-in-path + 1
        ] [
            ;; They are at the destination. Get ready for the next model iteration.
            set last-stop destination
            set destination nobody
            set mynode destination-entrance
            set got_to_destination got_to_destination + 1
        ]
]
tick
end

```

Route Finding

The final aspect of the model that needs explicit discussion is the route-finding algorithm. This is the algorithm that finds the shortest path (in terms of distance) from an agent's current location to its destination. Here the A-star algorithm is used. This is a heuristic adaptation to Dijkstra's shortest-path algorithm that does not necessarily need to visit *every* node in the graph. To fully understand the algorithm, it is worth first reading about it. The [Wikipedia entry⁸](#) is useful, or for something more comprehensive refer to de Smith et al. (2009). The route-finding algorithm is probably the most complicated aspect of the model. The code for the relevant procedure, `path-select`, is included in full in Code Example 8.4.4. It works by iteratively searching through the different vertices and recording the distance in total from the origin to the vertex (including all of the distances required to reach the previous vertices). The variable called `lastnode` belongs to the `vertex` objects, and this is used to store the previous vertex (or 'node') that was passed on the way. In this manner, once the algorithm has found the

⁸ https://en.wikipedia.org/wiki/A*_search_algorithm

destination node, it can work back through all of the previous nodes that were visited using the `lastnode` variable. Ultimately the agent is provided with two important variables: `mypath` is a list that stores all of the vertices that need to be passed through on the journey to the destination; and `step-in-path` records how far through the list the agent has travelled (i.e. how far along the route the agent is). Although the code is complicated, with an understanding of how the A-star algorithm operates the reader should be in a position to see how the path has been constructed in Code Example 8.4.4.

8.5 Linking Geographical and Social Networks

This chapter has only skimmed the surface of how one can use networks within agent-based models. However, the notion of networks is very important, not only in terms of navigation but also with respect to communication and the spread of information. Perhaps the greatest utility of networks is to bridge social and geographical networks. The route towards achieving this may be found in the mass of new forms of data that link people and places together (see Section 5.6).

We are currently witnessing a massive proliferation of location-aware devices such as smartphones and tablets (with in-built GPS) that allow for information to be easily contributed by users. To give a sense of the scale of this proliferation, it is estimated that in 2015 there were 2.6 billion smartphone users, a figure that is expected to rise to 6.1 billion by 2020 (Lunden, 2015). Coincidng with this growth is the growth in people using social media platforms. Facebook had on average 1.4 billion daily active users in December 2017 (Facebook, 2018), whilst Twitter had an average of 500 million tweets per day in January 2017 (Aslam, 2018). Furthermore, every minute Flickr users upload in excess of 3000 images (Sapiro, 2011). Many of these posts have some sort of geographical information associated with them within their metadata (as shown in Figure 8.9) or which can be derived from their message content, say using a place gazetteer such as GeoNames (see <http://www.geonames.org/>). For example, if we take tweets posted by Twitter users, locational information can be found in the form of a place descriptor, exact coordinates if posted with a GPS-enabled device, or a location provided from the user's profile description which can be extracted from a tweet's JavaScript Object Notation (JSON) file via the Twitter application programming interface.⁹

⁹ More information about Twitter and JSON files can be found at <https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json>

Twitter

```
{
  ...
  "place": {
    "name": "Rochester",
    "attributes": {},
    "full_name": "Rochester, NY",
    "place_type": "city",
    "url": "http://v.api.twitter.com/1/geo/id/2f1fc0d72969452b.json",
    "country_code": "US",
    "bounding_box": {
      "type": "Polygon",
      "coordinates": [[[[-77.701632, 43.183318], [-77.531166, 43.183318], [-77.531166, 43.269045], [-77.701632, 43.269045]]]],
      "id": "2f1fc0d72969452b"
    },
    ...
  },
  "coordinates": { "type": "Point", "coordinates": [-77.68793668, 43.16239715] },
  "geo": { "type": "Point", "coordinates": [43.16239715, -77.68793668] },
  ...
  "user": {
    ...
    "location": "NY x 2",
    ...
  },
  ...
  "text": "..."
}

1) Place descriptor, 2) Exact coordinates, 3) User provided location description
```

Flickr

```
<rsp stat="ok">
  <photo id="..." ...>
    ...
    <location latitude="38.888661" longitude="-77.098792" accuracy="14" context="0" place_id="llPb79UV7JQYFkay0" woeid="55857852">
      <neighbourhood place_id="llPb79UV7J0YFkay0" woeid="55857852">Clarendon-Courthouse</neighbourhood>
      <locality place_id="lta5yP9oP9oP9oP9o" woeid="23559494">Arlington</locality>
      <city place_id="lta5yP9oP9oP9oP9o" woeid="23559494">Arlington</city>
      <region place_id="p9rhG7VU6SbYO" woeid="2347685">Virginia</region>
      <country place_id="nz_0sghTUb4c2WAeCA" woeid="23424977">United States</country>
    </location>
    <geoparms ispublic="1" iscontact="0" isfriend="0" isfamily="0" />
    ...
  </photo>
</rsp>
```

1) Precise coordinates, 2) Descriptive location.

Figure 8.9 Metadata from Twitter tweets (top) and Flickr photos (bottom)

How much of this information has a geographical component? Croitoru et al. (2013) observed that, on average, the percentage of precisely geolocated (at the level of exact coordinates) tweets ranges typically between 0.5% and 3%, but in one study, following the 2012 Fukushima disaster in Japan, this was up to 16% (Stefanidis et al., 2013a). In addition to precisely geolocated tweets, Croitoru et al. (2014) have observed that approximately 40–70% of tweets come with a descriptive toponym related to the location of the user. Flickr metadata is provided in the exchangeable image file (Exif) format. Exif data provides a range of metadata about the contributed image (some of this is shown in Figure 8.9), including detailed information about the date and time the image is taken along with its location (if the camera/device has GPS).¹⁰ One study (Friedland and Sommer, 2010) found that approximately 4.5% of Flickr content is geolocated.

¹⁰ More information on how to access Flickr data can be found at www.flickr.com/services/api/flickr.photos.search.html

Code Example 8.4.4 The path-select procedure that implements the A-star algorithm to find the shortest path for a commuter turtle from its current location to its destination

```

to path-select ; Use the A-star algorithm to find the shortest path for a given turtle
  set mypath []      ; A list to store the vertices
  set step-in-path 0 ; Keep track of where we are in the list
  ask vertices [ ; Reset the relevant vertex variables, ready for a new path
    set dist 99999  set done 0  set lastnode nobody  set color brown
  ]
  ask mynode [ set dist 0 ] ; Set the distance to the current node to 0

  ; The main loop! This loops over all of the vertices, marking them as
  ; 'done' once they have been visited.
  while [count vertices with [done = 0] > 0] [
    ; Find vertices that have not been visited:
    ask vertices with [dist < 99999 and done = 0][
      ask myneighbors [
        ; Renew the shortest distance to this point if it is smaller
        let dist0 distance myself + [dist] of myself
        if dist > dist0 [ ; This distance is shorter
          set dist dist0
          set done 0 ; (so that it will renew the dist of its neighbours)
          set lastnode myself ; save the last node to reach here in the shortest path
        ]
      ]
      set done 1 ; set done 1 when it has renewed its neighbours
    ]
  ]
  ; At this point, all of the nodes have been visited, and the vertices that
  ; are part of the shortest path have stored the previous node in the path in
  ; their 'lastnode' variable. The last thing to do is to put the nodes in
  ; shortest path into a list called 'mypath'
  let x destination-entrance
  while [x != mynode] [
    if show_path? [ ask x [set color yellow] ] ; highlight the shortest path
    set mypath fput x mypath
    set x [lastnode] of x
  ]
end

```

Besides social media data sets, we are also seeing growth in the production and availability of spatially embedded communication and interaction networks. Such data take many forms (Schensul et al., 1999) but in general provide information on the actors in the network, along with how network measures could define rules and relationships between individuals and groups. New forms of data can help shed light

on the connections between people and places. For example, when analysing mobile phone data in Belgium, Lambiotte et al. (2008) found that the probability that two customers were connected generally follows a gravity model (i.e. the probability of connection decreases with distance) but at the same time there are some long-distance connections. Such findings are important if we are to model the spread of communication. The geographical content of social media also allows us to explore location-based social networks especially through services such as Waze or implicitly from GPS-enabled devices (e.g. Twitter activity).¹¹ At a more aggregate level, if we were building a model in international relations where each country is an actor we could base the relationship (friendliness) between two countries on different indicators such as who votes with whom in the UN, or who sells arms to whom, or who tweets to whom, as demonstrated by Crooks et al. (2014). If we were to extend and ground the disease model by Yuan and Crooks (2017) (which will be discussed shortly) to actual online debates, we could take insights from social media studies that explore the vaccination debate (e.g. Radzikowski et al., 2016) by using advances in sentiment analysis and machine learning (see Liu, 2012).

Figure 8.10 presents an illustration of how social and geographical networks could be fused together. As geographers, we are extremely good at mapping the spaces around us (e.g. L_1 and L_2 in Figure 8.10), and with network science we can also explore the connections between us (e.g. social networks, shown as L_3 in Figure 8.10). But perhaps what is currently being explored is only L_4 , which is how different types of networks come together and impact our daily lives at different locations (e.g. N_1 , N_2). This merging of networks has great potential for modelling, including the spread of diseases such as HIV and measles. Eubank et al.'s (2004) EpiSims model demonstrated how locations in physical spaces and agents in social networks (in this case a small-world network) can be used to create dynamic contact networks which can impact on the propagation of a disease. Such work has been used to estimate the spread of Ebola in West Africa during 2014 (Waldrop, 2017).

To give a simple example of how we can merge different networks together within an agent-based model, we use the work of Yuan and Crooks (2017).¹² The basic question that the model attempts to explore is how one's online social (i.e. cyber) network influences a person's willingness to be vaccinated, and how this choice impacts the spread of a disease in the physical environment. To answer this question, Yuan and Crooks (2017) have two networks as shown in Figure 8.11. The physical environment acts as one network (based on the agents' eight surrounding neighbours), and this is where the disease transmits (Figure 8.11A). The

¹¹ For more on location-based social networks, see Roick and Heuser (2013).

¹² The actual model and a detailed description of the model following the ODD protocol can be downloaded from www.openabm.org/model/5509/

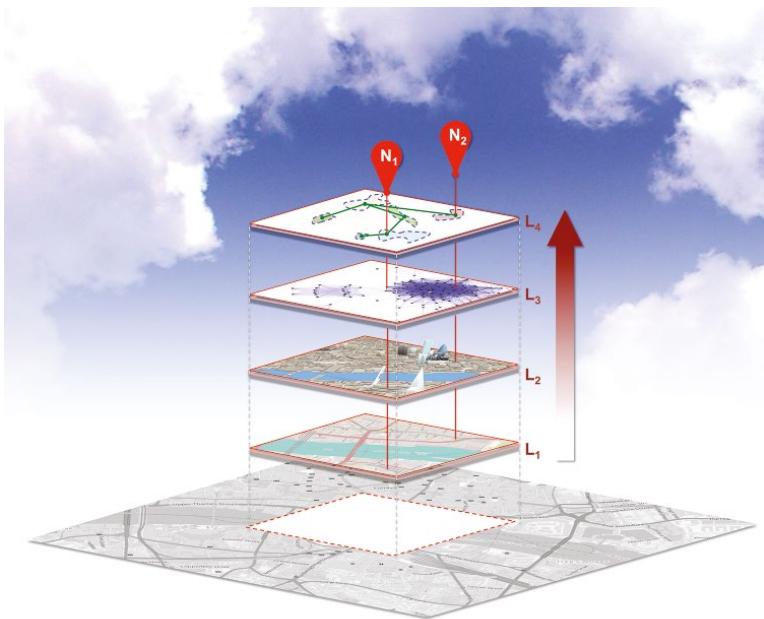


Figure 8.10 Schematic diagram illustrating how different networks can be linked together. From physical (L_1 , L_2), and social (L_3) spaces through to deriving place abstractions (L_4) for different locations (N_1 , N_2) (Crooks et al., 2016)

social network (Figure 8.11B), in this case a scale-free network (Section 8.3.3), acts as a medium for the diffusion of information, and people may be influenced by opinion leaders. The results of running the model show that when a number of cyber space anti-vaccine opinion leaders are present (i.e. those who are against vaccinations and spread messages that discourage people from being vaccinated)

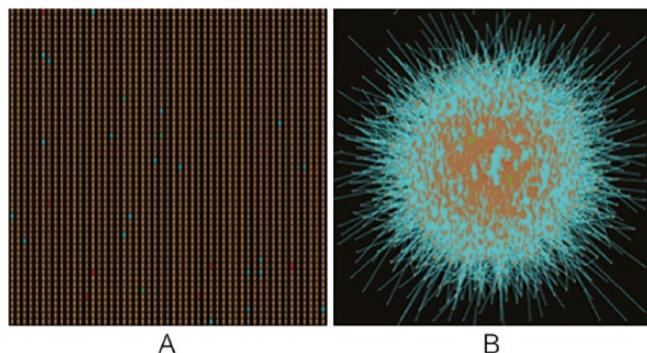


Figure 8.11 Agents and networks: (A) the physical environment; (B) the agents' social (cyber) networks

they can influence people not to be vaccinated, causing disease outbreaks to increase significantly in physical space. While this is only a simple model, one could ground, say, the number of anti-vaccine opinion leaders on data from actual studies of online vaccination debates (e.g. Radzikowski et al., 2016).

There are currently few social network models that combine social and geographical spaces explicitly (Alam and Geller, 2012), but such fusion offers

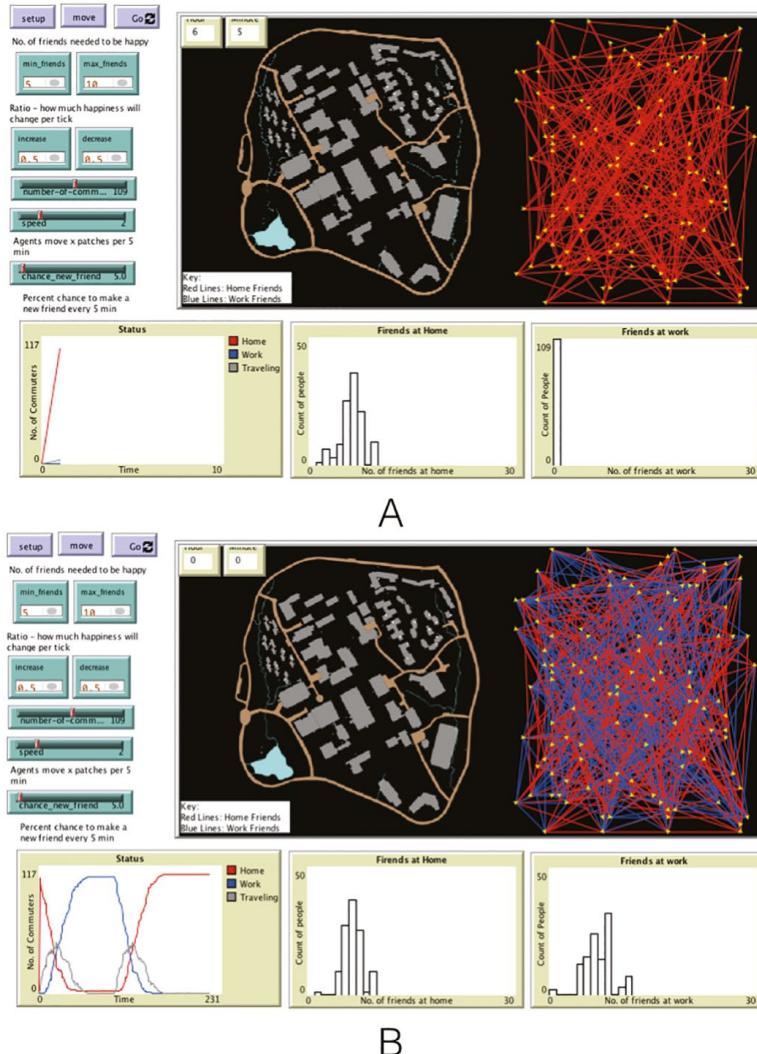


Figure 8.12 Agents and networks related to going to and from work: (A) initial social-network-based home locations; (B) evolution of the social network over time as agents interact with each other

exciting research opportunities. For more detailed work on the fusion of social networks, agent-based models and GIS, interested readers are referred to the work of Pires and Crooks (2017) or Wise (2014) as highlighted in Appendix A.3 and A.9, respectively.

As a pedagogical example of how we can link social and physical networks we show a very simple model created in NetLogo called GMU-Social (which is available in the accompanying online resources) in Figure 8.12. This model extends the GMU-Roads model presented in Section 8.4 to also allow for social networks to evolve over time based on interactions between agents as they go about their daily business. In this model, agents are first initialised and then create their social network based on people who share the same home locations (Figure 8.12A), which could be considered as a family network. Over time, as agents go to and from work, their social networks evolve (Figure 8.12B).

8.6 Discussion

Abstracting the complexities of the real world into a graph allows us to study connections in both physical and social spaces (as discussed in Section 8.1). For agent-based modelling, this allows us to study connections that are based not only on Euclidean distances but also on social or topological distances. While this chapter has only scratched the surface of networks,¹³ it has highlighted how networks (whether physical or social) can be represented and utilised in agent-based models (e.g. either for route planning as discussed in Section 8.4, or for transmission of information across communities as discussed in Section 8.3).

Considering such network connections opens the way to integrating network science into models and for local interactions to lead to more macro patterns emerging which are not just based on the surrounding neighbours of an agent (see Namatame and Chen, 2016, for more discussion). Also, the inclusion of networks directly in agent-based models allows us to explore local interactions and the emerging social structures. Often in spatial models agents' behaviours are only influenced by their surrounding neighbours. By adding networks, information does not just diffuse across the spatial environment (as is seen in cellular automaton models; see Section 11.2.1), but can disperse in many different ways (i.e. dependent on the social network structure itself, as shown in Section 8.5).

Networks also allow us to understand how complex behaviours of agents can be influenced by other agents. One of the advantages of agent-based modelling is

¹³ Interested readers wishing to find out more about networks are encouraged to take a look at the recommended books in the annotated bibliography at the end of this chapter as well as the cited papers.

that, rather than just taking a static view of networks or looking at them through a series of snapshots, we can make such networks dynamic and see how they might evolve. With the growth in data, we are also able to take properties from networks in the real world (e.g. roads and social media) and in turn use them to inform the desired properties in an agent-based model (as in the case of Wise, 2014), then, through simulation, it is possible to see how these networks impact the model and compare model outcomes with actual events (see Fontana and Terna, 2014, for a more detailed discussion). We will return to this idea in Section 12.3.1.

Chapter Summary

In this chapter we have demonstrated how the complex realities of the world around us can be abstracted into a graph and how such a graph can be analysed to shed light on the importance of a node, whether this be a person, a road intersection or something else entirely. We also discussed various types of networks – random, small-world and scale-free – and demonstrated the key properties of each, before discussing how both physical and social networks can be combined using a common method of abstraction, a graph. The symbiosis of network analysis and agent-based modelling allows us to capture the ways network structures influence the behaviour of individual agents based on how they are connected to each other. At the same time new forms of big data are shedding light on people's connections and providing new opportunities to study geographical systems.

8.7 Annotated Bibliography

For a comprehensive overview of social network analysis, methods and applications, readers are referred to:

- Wasserman, S. and Faust, K. (1994) *Social Network Analysis: Methods and Applications*. New York: Cambridge University Press.

For a classic work on the role of networks within geographical systems, readers are referred to:

- Haggett, P. and Chorley, R.J. (1969) *Network Analysis in Geography*. London: Edward Arnold.

For more details on how to instantiate networks and use them within agent-based models created using NetLogo, readers are referred to:

- Wilensky, U. and Rand, W. (2015) *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. Cambridge, MA: MIT Press.

Readers interested in how network science and agent-based models can be fused to understand social networks might find the following interesting:

- Namatame, A. and Chen, S.-H. (2016) *Agent Based Modelling and Network Dynamics*. Oxford: Oxford University Press.

Finally, for those wishing to explore networks there are several software packages, visualisation tools and programming libraries available, including:

- Software Packages: UCINET (Borgatti et al., 2002, URL: <https://sites.google.com/site/ucinetsoftware/>), NodeXL (Hansen et al., 2010, URL: <http://nodexl.codeplex.com/>).
- Network Visualization: Gephi (Bastian et al., 2009, URL: <https://gephi.org/>).
- Programming Packages: igraph – libraries and packages for C/C++, Python and R (Csardi and Nepusz, 2006, URL: <http://igraph.org/>). NetworkX library for Python (Hagberg et al., 2008, URL: <https://networkx.github.io/>).

In NetLogo, complex topological networks (e.g. roads) will make the model slow to run. Therefore it's sometimes good to simplify the network and consider using the online tools such as Mapshaper:

- <http://mapshaper.org>

A large blue circle containing the white number '9'.

SPATIAL STATISTICS

Chapter Outline

This chapter presents a range of statistics and algorithms that can be used to compare two spatial data sets. These are important for modelling because, at some point, it will be necessary to compare a model outcome to some real world data in order to assess the reliability of the model. This chapter examines the statistics themselves, before Chapter 10 elaborates on how to evaluate the success of a model more broadly, part of which includes making use of the methods discussed here.

Box 9.1 Further resources

The statistics used in this chapter were all calculated in the scripting language R using the code and data available in the accompanying online resources.

Box 9.2 Glossary

The following terms will be used throughout the chapter. They will be explained on first use, but this list might provide a useful reference.

- Goodness of fit - a broad description for statistics that quantify how well a model fits a set of real world observations.
- RSS - residual sum of squares; a statistic that estimates the overall difference between two data sets by summing the square of the errors.
- R^2 - 'R squared'; a statistic that estimates the overall difference between two data sets.

- (S)RMSE – (standardised) root mean square error; a statistic that estimates the overall difference between two data sets.
 - KDE – kernel density estimation; a means of estimating point density at a particular location. Often used to draw maps of point patterns.
 - LISA – local indicators of spatial association; statistics that can estimate the locations of local clusters (areas with a higher than expected point density).
 - NNI – nearest neighbour index; measures global spatial uniformity (i.e. the overall amount of clustering in a point pattern). Also known as the Clark and Evans R statistic (Clark and Evans, 1954).
 - Ripley's K function; another measure for global spatial uniformity (i.e. the overall amount of clustering in a point pattern).
 - GI^* – a statistic that estimates the spatial locations of 'hot' (e.g. high) and 'cold' (e.g. low) spots.
-

9.1 Introduction

All models are wrong but some are useful. (Box, 1979)

No model will be perfect, but even so it is still important to be able to quantify a model's imperfection. The processes of *calibration* and *validation* are used for this purpose, both of which will be covered in Chapter 10. However, in order to calibrate or validate a model, a mechanism is needed to determine whether one set of parameters are better than another. In other words, given two proposed models, which is the closest to the real world?

To answer this question, it is necessary to examine how well a model result is able to reflect some real world field data. This is termed *goodness of fit* – how well a model fits a set of observations. Whilst it is possible to use human intuition to qualitatively assess how well a model performs, for example by looking at maps of model results versus real data, a more rigorous assessment can be achieved through the use of statistics. This section will review some well-known methods that can be used to calculate goodness of fit, focusing in particular on the difficulties that arise when these differences need to be accounted for *over space*.

Deciding on which statistics are the most appropriate to compare two data sets in a given situation is difficult. It is necessary to consider the desired *outcome* (e.g. is a visual comparison satisfactory or is something more statistically robust required?), the *type of data* being examined (e.g. points, aggregate data, or something else) and the *scale of the analysis* (e.g. is a global measure of error satisfactory, or is it important to see how error varies in smaller areas?).

The statistics and methods that will be outlined here have been grouped into four categories, as illustrated below and by Figure 9.1:

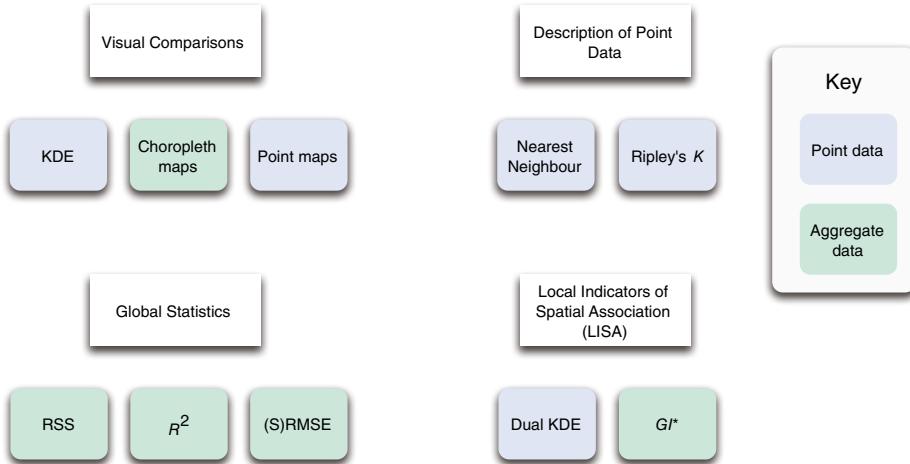


Figure 9.1 Overview of the statistics and methods that will be introduced in this chapter

- *Global statistics* (Section 9.3) – statistics that calculate the *overall* difference between two data sets (e.g. the total error);
- *Visual comparisons* (Section 9.4) – methods that provide visual outputs and allow spatial data sets to be compared visually;
- *Description of point data* (Section 9.5) – statistics that describe the properties of point data, such as the degree of clustering;
- *Local indicators of spatial association* (Section 9.6) – statistics that calculate the difference between two data sets at a local spatial level.

The chapter concludes with an overview of a less commonly used method that spans a number of the above categories which we call *multi-scale error analysis* (Section 9.7).

9.2 Hypothetical Data

The following examples will make use of two hypothetical simulated data sets (i.e. results that might have been produced by model) and a hypothetical observed data set.¹ All three are point data sets, where each point represents

¹ The data is actually derived from publicly available UK sources. For details about where the data have come from and how to recalculate the statistics used here, refer to the R script in the accompanying online material.

some occurrence in space (someone purchasing a product, becoming a victim of a crime, etc.). To illustrate the use of non-spatial statistics, the data have also been aggregated to an administrative area. Figure 9.2 presents a map of the raw points as well as a choropleth map that shows the number of points per area once they have been aggregated. The two modelled data sets are very similar, as would be the case if they were drawn from the same model with slightly different parameter configurations. The observed data is also relatively similar to the modelled results – the hypothetical models are reasonably good – but there are some differences in the spatial distributions.



Figure 9.2 Hypothetical point data used to illustrate the use of statistics in this chapter: two hypothetical model results and an observed ('real') data set. The points are shown in the top row, and in the bottom row are the number of points in each administrative area (dark-red areas have more points)

Figure 9.3 presents a scatter plot that shows the differences in the number of points per administrative area between the two simulated data sets and the observed data set. In this example, a 'perfect'² model would replicate the observed data exactly, so all points would sit on the line $y = x$ (drawn in red). Overall, the

² It is worth noting that, in reality, a 'perfect' model would probably not replicate some observed data exactly. The observed data undoubtedly include some error in their quantification of the underlying phenomenon, so a model that perfectly simulates the data will also include that observational error.

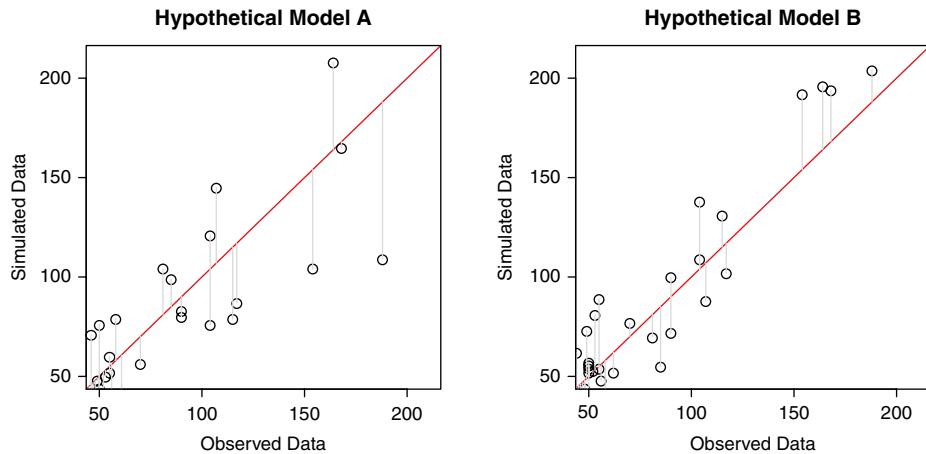


Figure 9.3 The results of two hypothetical models (A and B) plotted against observed data. The lines of best fit ($y = x$) illustrate the locations of ‘perfect’ model results. In this example, model B is a better fit to the observed data. Note that the graphs only illustrate values of $x > 50$ for clarity

results of model B (on the right) are a closer match to the observed data so, in this instance, a goodness-of-fit statistic would report that model B was the better fit (as will be illustrated).

9.3 Goodness of Fit with Global Statistics

Goodness-of-fit statistics are well-established techniques that describe how well modelled outcomes reflect some real world conditions. They are commonly applied to tabular data or matrices. Although they are not inherently spatial, they can often be applied to spatial data. For a model that works with administrative areas, a table can simply be created that compares the simulated data to observed data in each administrative area. For models that output occurrences of some phenomenon as individual points, it is possible to aggregate the points to some administrative area first and then create a table. Table 9.1 presents part of the hypothetical data used in this example.

One of the easiest statistics to use is the residual sum of squares (RSS), which is calculated by adding together the square of the differences between the observed and simulated values:

$$\text{RSS} = \sum_{i=0}^n (\gamma'_i - \gamma_i)^2, \quad (9.1)$$

where γ'_i is the simulated value at matrix point i and γ_i is the observed value at i . Disadvantages with the RSS include the following:

Table 9.1 An example of a table that stores the number of points per area. This can be used to generate goodness-of-fit statistics

Area code	Number of points per area		
	Observed ('real')	Model A	Model B
...		...	
E01032504	2	5	8
E01032503	14	19	17
E01032500	5	10	18
...		...	

- By squaring the errors in individual cells, large errors can be inflated disproportionately.
- The magnitude of the data influences the error statistic. For example, if an observed value was 100, and a model predicted a value of 101 (a 1% difference) the error is 1. However, if the observed value were 2, and the model predicted a value of 3, the error is still 1, but this is a substantially greater *relative* over-prediction – a 50% difference as opposed to 1%.

Hence there are a wealth of more advanced statistics whose aims are to give a more reliable assessment of the goodness of fit. Knudsen and Fotheringham (1986) experimented with a number of such statistics and found that the standardised root mean square error (SRMSE) was the best performing. The lower limit of the SRMSE is 0, which indicates a ‘perfect’ model, and the upper limit is usually 1, but this can be greater when matrices are sparse (Harland, 2008). Although the SRMSE is preferable because it is comparable across spatial systems and is not sensitive to large deviations from the mean (Knudsen and Fotheringham, 1986), the root mean square error (RMSE) is more readily available in statistical packages, so that measure is used here.

A drawback with the RMSE, however, is that it can be difficult to understand what the value actually means. For example, how much better is a model with RMSE = 0.2 than one with RMSE = 0.3? An alternative statistic, R^2 , is much easier to understand because it represents the proportion of agreement between a model and the observed data. The range of the statistic is from 0 (no agreement) to 1 (perfect agreement). For example, a model with $R^2 = 0.8$ is able to represent 80% of the variation exhibited in the observed data. However, Harland (2008) has shown that the R^2 is insensitive to the overall amount of error, predicting a good fit in some circumstances where the (S)RMSE would not, and its reliability is dubious for assessing error in nonlinear models.

Formally, the $RMSE$ and R^2 can be defined as

$$RMSE = \sqrt{\frac{(\sum_{i=0}^n (\gamma'_i - \gamma_i)^2)}{n}}, \quad (9.2)$$

$$R^2 = 1 - \frac{\sum_{i=0}^n (\gamma_i - \bar{\gamma})^2}{\sum_{i=0}^n (\gamma_i - \bar{\gamma})^2}, \quad (9.3)$$

where γ'_i is the simulated value at matrix point i , γ_i is the observed value at i , $\bar{\gamma}$ is the mean value of the predicted values (γ') and n is the total number of values.

Table 9.2 illustrates the values of the different statistics when applied to our two hypothetical models and the hypothetical observed data. Each statistic is in agreement that the results of model B are a closer match to the observed data (recall that *lower* SRMSE values indicate greater agreement, whereas *higher* R^2 values indicate greater agreement). It is beyond the scope of this chapter to explore the intricacies of each statistic in more detail. The main lesson is that *the choice of statistic matters* for error calculation. For example, if only the RSS were used here, one might conclude that model A is significantly weaker than model B. However, the R^2 statistic suggests that model B is only able to account for 7 percentage points more of the variation in the observed data – an important difference but not enormous.

The annotated bibliography at the end of this chapter includes further reading on this area.

Table 9.2 The values of the non-spatial statistics when applied to hypothetical example data

Hypothetical model	RSS	R^2	RMSE
Model A	112,000	0.79	9.0
Model B	86,000	0.86	7.9

A drawback with the statistics outlined above is that they are *aspatial*. Although they can be applied to spatial data by, for example, aggregating points to administrative areas, this aggregation not only introduces error but also is susceptible to the modifiable areal unit problem (MAUP; Openshaw, 1984). Furthermore, the aspatial statistics struggle with the situation where some simulated phenomenon is *close* to its real world counterpart, but not at exactly the same position. For example, a model might predict a disease outbreak within a few hundred metres of a ‘real’ outbreak, which could be considered an excellent prediction. However, if the real and simulated outbreaks occur in different

administrative areas then the statistic will report the same level of error as if they had been many kilometres away.

Ideally, therefore, a disaggregated model (as is often the case with an agent-based model) would leverage an error statistic that works with disaggregated spatial data (e.g. points) directly. Fortunately there are a number of statistics that describe the distribution of point data and can be used to estimate the agreement between two data sets. Furthermore, there are aggregate statistics that have been designed specifically to operate on spatial data and can be more effective. The following sections will outline some alternative methods for comparing observed and modelled spatial data visually and statistically.

9.4 Visual Comparisons

A simple method of comparing model results to observed data is to create a map of the results and then examine the differences visually. This can be a useful exercise, particularly in early stages of results analysis. However, displaying point data on a map is not generally informative of the overall pattern, particularly for large data sets. Instead, it is common to produce thematic maps by aggregating points to (administrative) areas. This approach has the additional advantage that once the points have been aggregated to an area, traditional goodness-of-fit statistics can be applied to the underlying data table in order to quantify error (see Section 9.3). However, maps of this sort are not ideal for visual comparisons because they will suffer from the MAUP and large, bright areas can draw the eye disproportionately which makes the visual comparisons very subjective.

An alternative technique that is commonly used to compare point data is to apply a density estimation algorithm. These begin by placing a regular grid over the study area, and then estimating the density of each cell in the grid by counting the number of cells within a distance (or ‘kernel’), d , from the cell. O’Sullivan and Unwin (2010) describe a ‘naive’ method to estimate the density for a cell, c , as

$$\text{density}_c = \frac{\#(S \in C(c, d))}{\pi d^2}, \quad (9.4)$$

where d is the kernel distance, $C(c, d)$ is a circle of radius d focused on the centre of the cell c , S is the set of all points and $\#$ means the ‘number of’. The size of the kernel (d) can be varied to find a balance between over-smoothing (where multiple separate clusters are merged into one) and under-smoothing (where too few points are merged and the output is not dissimilar to a map of the raw points). It is worth noting that although kernel density estimation (KDE) maps can hint at cluster locations, they do not actually define cluster boundaries. For this purpose, local indicators of spatial association (LISA) statistics can be utilised (see Section 9.6).

These identify areas where the number of points is significantly greater than in surrounding areas and can be useful for comparing two point patterns. For more information the interested reader can refer to Chainey and Ratcliffe (2005).

To summarise, Figure 9.4 compares maps of points, thematic maps, and KDE maps. Of these three methods, mapping the density is the most commonly used and often considered the most appropriate. Comparing simulation results in this way can highlight clear spatial areas in which models are under- or over-predicting the number of occurrences. It is worth noting that an obvious extension to density mapping (and also to the thematic maps) is to map the *difference* between the simulated and observed data in each area. However, this still poses a problem when a simulated hotspot is *close* to its ‘correct’ position, but not exactly overlaid. Section 9.7 introduces a solution to this problem.

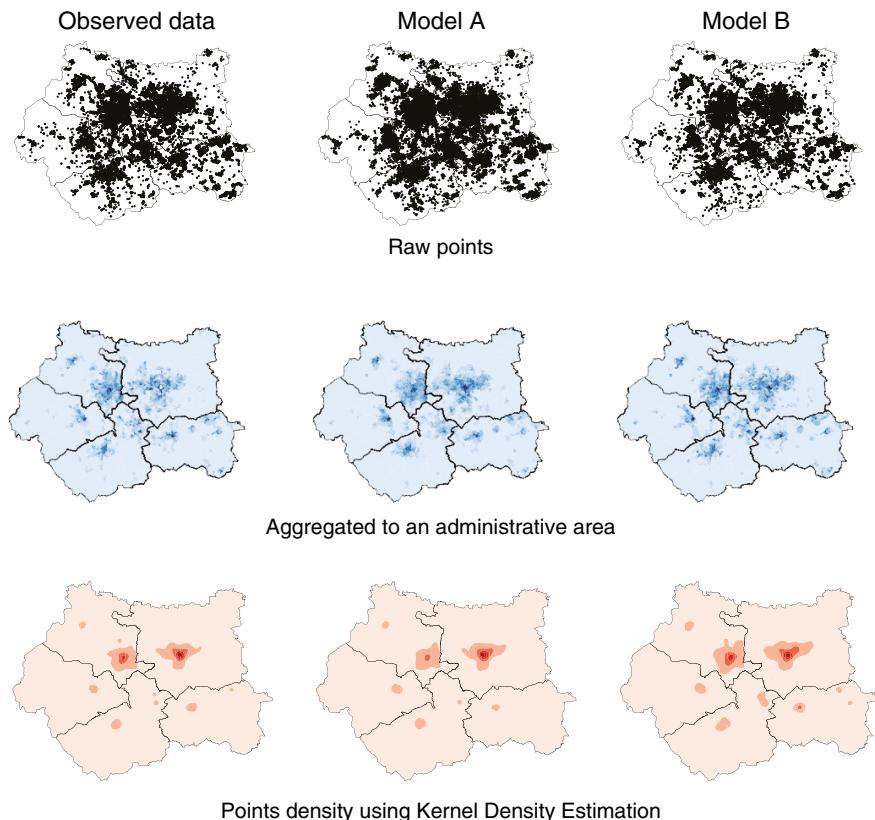


Figure 9.4 Visual comparisons of point pattern similarity: (top) mapping raw points; (middle) aggregating points to an administrative area; (bottom) calculating point density with kernel density estimation

9.5 Description of the Properties of Point Data

As already mentioned, the process of aggregating points to administrative areas introduces error. Therefore it is preferable to assess the similarity of point data sets directly. A common means of doing this is by analysing the degree of clustering in a data set, and one of the simplest ways to assess clustering is through nearest-neighbour methods. These generally analyse point patterns by comparing the distances from any given point to its nearest neighbour, whether these distances are taken from within the same data set or two different ones. In clustered data, points will generally be close to their neighbour, whereas with uniform data the points are generally farther away. The mean minimum nearest-neighbour distance, \bar{d}_{\min} , in a point pattern is calculated by averaging the distances between each point, i , and its nearest neighbour, j :

$$\bar{d}_{\min} = \frac{\sum_{i=1}^n d_{ij}}{n}, \quad (9.5)$$

where n is the number of points. This distance can be used to calculate the *nearest neighbour index* (NNI) – originally formulated as the Clark and Evans R statistic (Clark and Evans, 1954) – by comparing the mean minimum distance between points (\bar{d}_{\min}) to the mean minimum distance that would be generated if the points were produced by an entirely random spatial process ($\bar{\delta}$):

$$NNI = \frac{\bar{d}_{\min}}{\bar{\delta}}, \quad (9.6)$$

where

$$\bar{\delta} = \frac{1}{2\sqrt{A/n}} \quad (9.7)$$

and A is the total area under study (so A/n is the average point density).

In effect, therefore, the NNI is the ratio of the observed mean minimum distance to the expected mean minimum distance under complete spatial randomness.³ The statistic ranges from 0 (all points are at the same location) to 2.15 (entire spatial uniformity). To improve the strength of the analysis, Bailey and Gatrell (1995) offer a significance test. Table 9.3 presents the average NNIs for the three example data sets. As expected, the degree of clustering in the second simulated data set is much closer to that of the observed data. This therefore corresponds closely to the other statistics used throughout the chapter.

³ Complete spatial randomness is an important concept that is used in many point pattern statistics. For a detailed and accessible discussion, see O'Sullivan and Unwin (2010).

Table 9.3 The average NNI values for the three example data sets. The data for model B is closer to the observed data, suggesting a better fit

Data	NNI
Observed data	0.20
Model A	0.53
Model B	0.24

The NNI can offer a useful insight into the degree of clustering in two data sets. If the simulated and observed data sets exhibit very different degrees of clustering then it is likely that the model is representing the underlying phenomenon poorly. There are, of course, some drawbacks with the NNI. One problem is that points that are close to a study area boundary are likely to return a larger minimum distance simply because their nearest neighbours have been excluded – known as *edge effects*. Although some implementations of the statistic attempt to mitigate these problems, they are not ideal (Chainey and Ratcliffe, 2005). Also, it does not highlight the *locations* where clusters in simulated and observed data vary. This has the related disadvantage that two different point patterns might exhibit the same NNI values. Finally, the NNI only considers a single distance in its calculation (that of the closest neighbour), which disregards a considerable amount of information.

Ripley's K function (Ripley, 1977), on the other hand, takes *all* of the neighbours that are within a given distance into account. The index works by counting the number of neighbours within a given distance, d , from each point s_i . The value of K at distance d is then calculated as the mean of all counts divided by the overall point density, defined by O'Sullivan and Unwin (2010) as

$$K(d) = \frac{\sum_{i=1}^n \#(S \in C(s_i, d))}{A / n}, \quad (9.8)$$

where $C(s_i, d)$ is a circle of radius d centred at point s_i and A is the size of the area being studied. Larger values of $K(d)$ are indicative of a greater degree of clustering. Graphs of $K(d)$ reveal information about the clustering of the points at different distances. For example, Figure 9.5 illustrates how the value of K varies with distance for each of the three example data sets. Both the simulated data sources exhibit somewhat different clustering patterns than the observed data, but the second data set appears to be a closer fit (as expected).

There are a number of related statistics that measure slightly different properties of point pattern clustering and provide tests for statistical significance. For a more detailed discussion, the interested reader might begin with O'Sullivan and Unwin (2010). The following section moves beyond simple quantification of clustering and begins outlining methods that can identify clusters at the local level.

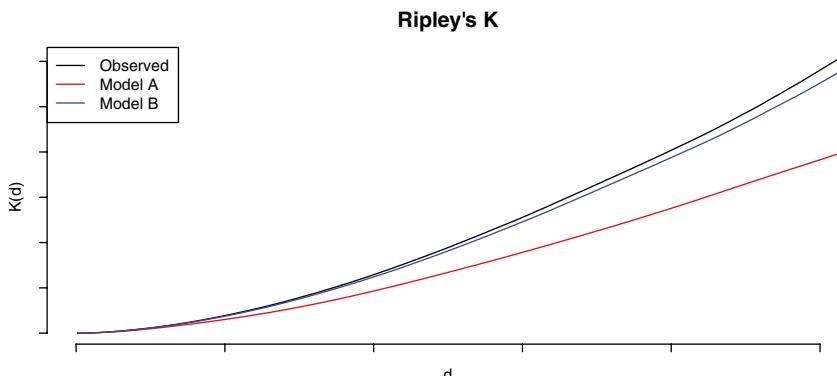


Figure 9.5 Ripley's K with distance ($K(d)$) for the three example data sets. Model B data has a $K(d)$ value that is more similar to the observed data than model A, suggesting a closer fit

9.6 Local Indicators of Spatial Association (LISA)

The methods outlined in the previous section can be used to describe the degree of clustering in a point data set, but provide no information about *where* the clusters might be located. In this section, two methods that provide information about the locations of clusters will be introduced: dual KDE and the GI^* statistic.

9.6.1 Dual KDE

Section 9.4 introduced kernel density estimation as a means of mapping point density. The method worked by placing a regular grid over the study area, and calculating the point density for each cell. This method can be extended for comparing two data sets by simply taking one grid away from the other.⁴ Figure 9.6 illustrates the outcome of calculating the difference between the KDE maps produced by the observed data and those produced by the simulations.

9.6.2 GI^*

The Getis–Ord GI^* statistic (Getis and Ord, 1992; Ord and Getis, 1995) estimates the spatial locations of ‘hot’ and ‘cold’ spots. These are areas where the number of observations is significantly greater, or smaller, than the number of observations in the surrounding areas. The method works with aggregate spatial data rather than directly with points. Figure 9.7 illustrates the GI^* z -scores for each of the

⁴ The density grids that are produced by KDE can actually be compared in numerous ways – for example, by taking the sum, difference or quotient.

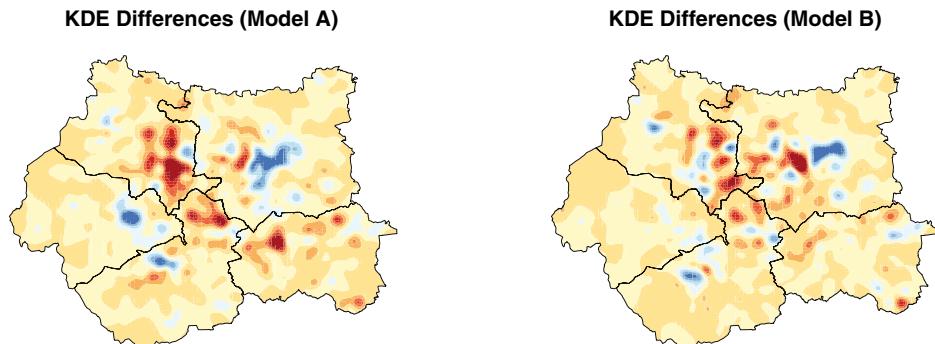


Figure 9.6 The difference between observed point densities and simulated point densities, calculated using kernel density estimation. Red areas highlight negative values (model under-predictions), blue areas highlight high values (model over-predictions)

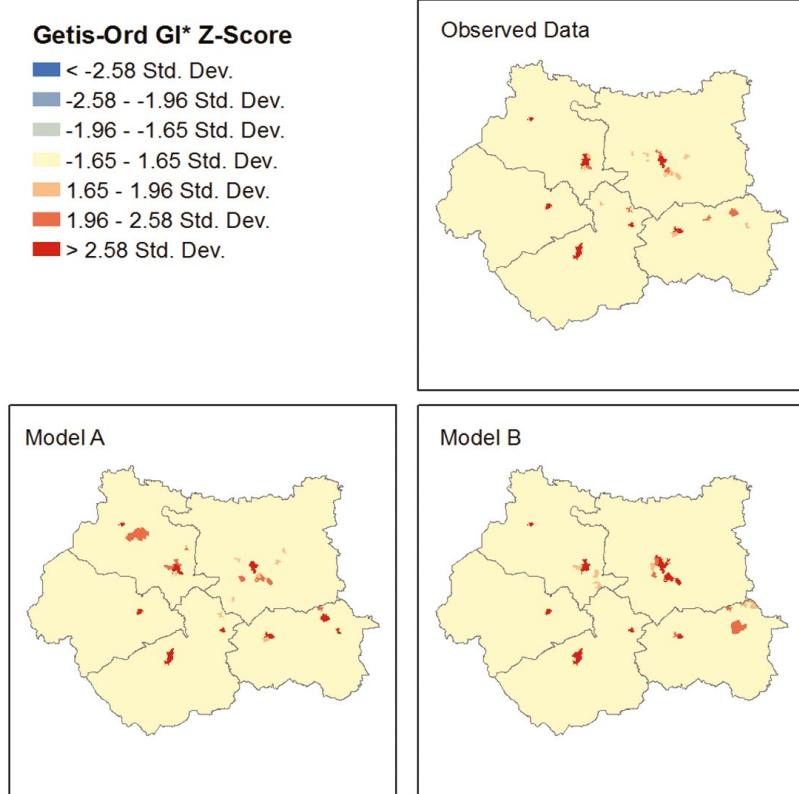


Figure 9.7 Getis–Ord GI^* (Getis and Ord, 1992; Ord and Getis, 1995) hotspots

hypothetical data sets. The data is largely similar, as expected, but there are some locations in the simulated data where hotspot locations vary slightly. In a real setting, it might be interesting to explore why the models are incorrectly simulating some, but not all, of the hotspots. Furthermore, note that in the hypothetical data used here, there are no ‘cold’ spots.

9.7 Multi-scale Error Analysis

So far, this chapter has discussed non-spatial goodness-of-fit methods for comparing data (Section 9.3), followed by comparison through mapping (Section 9.4), and finally through analysing the properties of the underlying point patterns (Section 9.5) and the locations of clusters (Section 9.6). This final section will outline an approach than combines many of the advantages of previous methods by providing an overall error assessment that is spatially explicit and varies by spatial scale. Originally proposed by Costanza (1989) and further explored by Malleson (2010), the procedure involves aggregating points to regular grids of varying resolutions and using goodness-of-fit statistics to measure both the overall error across the data set and the error associated with each cell. The method has not been widely used in practice and is, as far as the authors are aware, not available in any standard statistical packages. The implementation used in this chapter can be found in the R script that accompanies this chapter.

To summarise, the method works as follows:

1. Begin with two point data sets (e.g. a simulation result and an observed data set).
2. Create a regular grid and place it over the points. Initially the grid cells should be large, as the resolution will be reduced in later stages. For example, the initial grid used as a demonstration here consists initially of only one cell.
3. Count the number of points from each data set that fall within each cell.
4. For each cell, i , calculate the relative percentage error between the observed (y_i) and simulated (y'_i) data. This is defined as the difference between the proportions that the cells contribute to the total observation count,

$$100 \frac{y_i}{\sum y} - 100 \frac{y'_i}{\sum y'} . \quad (9.9)$$

This gives the error associated with *each cell*.

5. Calculate aspatial statistics such as R^2 or the RMSE (see Section 9.3) for all cells to give a *global measure of error*.
6. (Optionally) move the grid slightly and repeat step 2 – this reduces the impact of the MAUP (Openshaw, 1984).
7. Repeat step 2 with a grid of a smaller resolution (i.e. more, smaller cells).

After calculating the local and global errors at a number of different resolutions, it is possible to explore the results both spatially and aspatially. To this end, Figure 9.8 presents fuzzy maps of the difference between the results of each simulation and the observed data. In this map, each grid created in step 2 has been drawn simultaneously using a level of transparency such that each grid contributes equally to the overall colour. This makes it possible to begin to identify areas that show the greatest degree of divergence between the model results and the observed data.

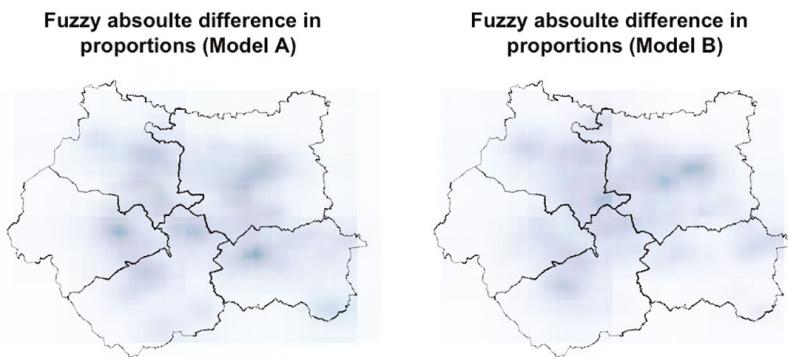


Figure 9.8 Fuzzy maps produced by the multi-scale error analysis process. Darker blue colours indicate areas where the model results show the greatest divergence from the observed data

As well as exploring the places where model errors are the greatest, the multi-scale error analysis algorithm also makes it possible to explore the *overall accuracy* of a model at different resolutions. In a similar manner to the fuzzy map in Figure 9.8, Figure 9.9 graphs the error of model B at different resolutions. Note that for each resolution there are actually three distinct points; this is because in step 6 the grid is moved slightly in order to reduce the impacts of the MAUP. As the size of the cells increases, the error reduces. This is entirely expected because if the data is aggregated to a larger area, it is more likely that the simulated points and the observed points will fall within the same cell. Where these graphs are most useful, however, is in trying to identify the smallest cell size at which the model is still able to perform sufficiently well. Here, once the cell size reaches

approximately 10 square units, the model errors stop decreasing rapidly, so larger cell sizes have a limited impact on the error. Therefore the developer of model B has some evidence to suggest that their model is accurate up to approximately 10 square units, and is not able to reliably represent the phenomenon at resolutions smaller than that.

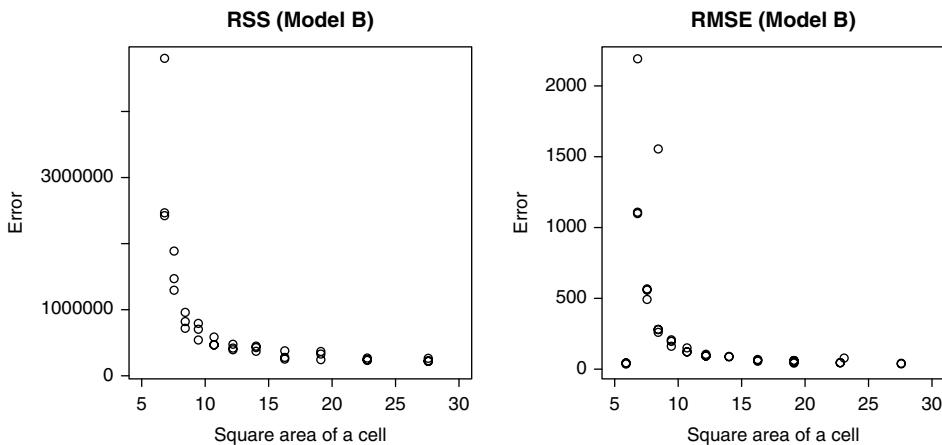


Figure 9.9 The total error between the observed data and model B at different resolutions

9.8 Discussion

When attempting to create models that are grounded in a real world application, it is often necessary to compare model results to some observation data. Choosing the most appropriate methods for making these comparisons is an important decision. Whilst visual methods are useful, often more rigorous statistics are required that can *quantify* the difference between observed data and those produced by models. This is especially important for validating and calibrating models, as will be discussed in Chapter 10. This chapter has therefore laid the necessary groundwork required before robust model evaluation can take place.

The process of applying these methods in earnest is often non-trivial. Here, much of the analysis was performed using the R statistical scripting language (the source file is available from the accompanying online resources), but other libraries or software tools are available. For example, ArcGIS, QGIS and GeoDa all include the functionality to calculate statistics such as GI^* , and to aggregate points using KDE. Although direct instructions are not included here, there are a variety of textbooks and other online resources that demonstrate how such statistics can be calculated. The annotated bibliography below highlights some of these.

Finally, it is worth nothing that the statistics that can be used to describe the properties of point patterns (as discussed in Section 9.5) are not measures of error *per se*, they can be used to contrast the properties of two spatial point patterns and might therefore be indicative of how well the underlying model dynamics is able to account for the behaviour of the real system. For example, if there is less clustering in a model than should be expected, this might highlight a particular flaw in the design of the model or the theories that it has been based on. Broadly, the important lesson that this chapter has attempted to put across is that there are a wide variety of methods that can be used to assess the similarity between two spatial data sets, all of which have pros and cons, and choosing an appropriate one will have a large impact on the perceived accuracy of a model.

Chapter Summary

This chapter has presented a range of statistics and algorithms that can be used to compare two data sets. Three hypothetical data sets were used to illustrate the results of the different methods: a ‘real world’ data set and two others that could have been produced by a model with slightly different parameter configurations. Section 9.3 began with a review of some statistics that can be used to estimate the *overall* difference between two sets of data (i.e. the total model error). Section 9.4 then went on to discuss mapping methods that can be used to compare spatial data visually. Section 9.5 reviewed some methods that describe the *properties* of spatial point patterns. The chapter concluded by illustrating statistics that can be used to find clustering at the local level (Section 9.6) and a method that provides both a fuzzy error estimate at the neighbourhood level and a measure of error at various spatial resolutions (Section 9.7).

Broadly, the methods discussed here are important for modelling because, at some point, it might be necessary to compare a model outcome to some real world data in order to assess the reliability of the model. Chapter 10 will elaborate on how to evaluate the success of a model more broadly, part of which will include making use of the methods discussed here.

9.9 Annotated Bibliography

For a detailed review of a range of spatial statistics and to learn more about the methods presented here, readers are referred to:

- O’Sullivan, D. and Unwin, D. (2010) *Geographic Information Analysis* (2nd edn). Hoboken, NJ: John Wiley & Sons.

For more details about the pros and cons of a number of the goodness-of-fit statistics discussed here, including the root mean square error and R^2 , readers are referred to the seminal paper:

- Knudsen, D.C. and Fotheringham, A.S. (1986) Matrix comparison, goodness-of-fit, and spatial interaction modelling. *International Regional Science Review*, 10(2), 127–147.

While targeted mainly at a crime analyst audience, the following book discusses a number of useful methods that can be used for mapping spatial data:

- Chainey, S. and Ratcliffe, J. (2005) *GIS and Crime Mapping*. Chichester: John Wiley & Sons.

For a broad and up-to-date review of the difficulties in analysing agent-based model outputs and in particular the analysis of time, readers are referred to:

- Lee, J.-S., Filatova, T., Ligmann-Zielinska, A., Hassani-Mahmooei, B., Stonedahl, F., Lorscheid, I., Voinov, A., Polhill, G., Sun, Z. and Parker, D.C. (2015) The complexities of agent-based modelling output analysis. *Journal of Artificial Societies and Social Simulation*, 18(4), 4. Available at <http://jasss.soc.surrey.ac.uk/18/4/4.html>

Much of the statistical analysis in this chapter was derived from the excellent textbook:

- Brunsdon, C. and Comber, L. (2015) *An Introduction to R for Spatial Analysis and Mapping*. London: Sage.

10

EVALUATING OUR MODELS: VERIFICATION, CALIBRATION, VALIDATION

Chapter Outline

Model evaluation is one of the central challenges associated with agent-based models. A key question that all modellers face is how well their model simulates the phenomenon of interest. While there are no universally accepted methods for evaluating agent-based models, researchers often adopt the same three-stage process of verification, calibration and validation. This chapter presents an overview of the methods that are commonly used within each of these stages. The overarching aim of this chapter is to provide the reader with the knowledge to design their own approach to evaluating agent-based models.

10.1 Evaluating Models: An Overview

In the previous chapters, the design and building of spatially explicit agent-based models has been discussed. Chapter 9, in particular, introduced methods that can be used to assess how similar two data sets are. For models to be useful as tools to explore the *real world* they need to replicate the behaviour of their target system to a certain level of accuracy. Of course a model will never be a perfect representation of the system under study – if that were the case then it would cease to be a *model* – but the modeller needs to understand the extent of the accuracy of their simulation. Hence the purpose of this chapter is to introduce the methods that can be used to assess how ‘correct’ a model is, enabling the researcher to quantify the extent to which a model matches some observed conditions in the real world.

An evaluation of model accuracy is an extremely important part of the modelling process, and yet it is a part that is often overlooked or given minimal attention.

Part of the reason for this is that evaluating agent-based models is extremely challenging – see Railsback and Grimm (2011, p. 256) for a discussion – especially when account has to be taken of the patterns that models produce *across space*.

There is no formal methodology for evaluating agent-based models, but many authors have come to settle on a standard process that consists of three separate procedures: *verification*, *calibration* and *validation*. There are variations in the process used to evaluate models and the terminology used to explain the procedures, but broadly most authors are in agreement over the general process. Figure 3.2 from Chapter 3 illustrates how these three procedures fit into the wider modelling process. Each of these concepts will be discussed in detail in the following sections. In summary:

- *Verification* (discussed in Section 10.2) is the process of ensuring that the model *implementation* corresponds to the model *design*. In other words, has the model been programmed correctly?
- *Calibration* (discussed in Section 10.3, also known as ‘parameterisation’) is the process of adjusting the model parameters so that the behaviour of the model closely matches some observed conditions (i.e. ‘fitting’ the model to some data). In effect, we calculate the difference (or ‘error’) between an observed data set and a simulated one, and try to configure the model to reduce the error.
- *Validation* (discussed in Section 10.4) completes the process of evaluating a model by testing it on some new data. Its aim is to demonstrate that the model can produce results to a level of accuracy that is deemed suitable within its domain of applicability. If the model is able to reliably replicate real world conditions that it has not been ‘fitted’ to, then we can be reasonably confident that it can be applied to new (possibly hypothetical) scenarios to explore potential futures or used to better understand system dynamics.

10.2 Verification

Once a model has been designed and implemented it is important to verify that the implementation matches the design. In other words, has the model been *correctly implemented* according to the design? This process is often termed ‘verification’ and broadly involves checking that the model behaves as it is expected to and the results match prior intuition. This can be defined as *internal validity* (Axelrod, 1997a, p. 7). Deciding whether or not the model actually reflects the real world is considered later with calibration and validation. Agent-based models are typically very complicated, with large numbers of behaviours and complex interactions between individual agents across space and time. Hence verification is non-trivial. As outlined below, there are a number of approaches that can allow us to gain insight into the internal validity of our model.

10.2.1 Code Testing

One important source of model error is introduced during the coding of the model. While a model might run without exception and generate a reasonable model output, there is still a chance that an error exists in the code that ultimately means the agent-based model is flawed in some way. These errors can typically consist of mistyped variable names (e.g. assigning a value to the wrong variable), incorrect value assignment (e.g. assigning a parameter value to all agents rather than one) or forgotten quick fixes, among others, that are introduced as the model is coded up. These types of errors are known as ‘semantic’ or ‘logical errors’.

Fixing errors in code can only be achieved through a careful reading of the code itself, and through observation of parameter values. Reading through code is a straightforward task, but should be undertaken systematically. Ideally, it should also be read by someone who did not write the code themselves. ‘Pair programming’ is a common approach in software development whereby two people share the same computer; one person writes the code and another reads it while it is being written. This has been found to reduce the number of errors in code. When reading, each method should be inspected individually, ensuring that the correct variable names are being used, that all assignment functions are in order and brackets close in the correct place (e.g. code is not inadvertently included within `if` and `for` blocks). As the code is being checked, it may be helpful to take the opportunity to incorporate additional comments relating to each part of the model as this allows the model design to be easily explained to others in the future. More examples of the types of code error that might be introduced between model concept and implementation can be found in Galán et al. (2009).

Box 10.1 Unit testing

A more formal approach to testing code, other than just rereading it, is *unit testing*. Unit testing, and similar formal testing approaches, are extremely common in software development but are typically rarely used in academia. Unit tests are procedures that test the smallest units of code. For example, a model might include a function called ‘get-happy-agents’ that returns all agents who have a particular value for an attribute. A unit test for this function might set up the environment in such a way that it knew exactly how many agents should be found (i.e. how many ‘happy’ agents there were), then run the get-happy-agents function to check that the expected number was returned. Although at the time of writing unit tests were not especially well supported in NetLogo, they are a common feature of most major programming languages.

Another way to check the performance of the code and presence of ‘logical errors’ is to watch the variation of particular variables at the model or agent level. NetLogo allows access to these variable values through the `Inspect` feature, as illustrated in Figure 10.1. `Inspect` can be found through a right-click on the model output interface, and can be used to access either global or agent-level variables. As the model proceeds, the `Inspect` monitor will report the evolution of each variable, and allows one to check for strange variances – for example, whether the variables are falling within an expected range. The `Inspect` interface also allows the movement of specific agents to be observed. This allows a check on whether the agents are moving in the directions and speed that are expected. Where unexpected behaviour is found, further attention should be paid to the coding and design of the functions augmenting those specific variables.

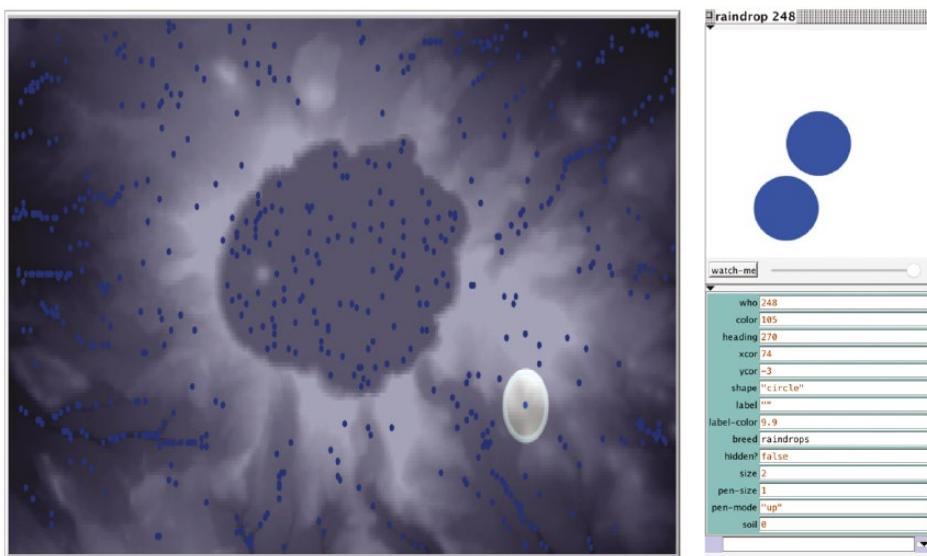


Figure 10.1 Inspecting the current state of an agent, and watching it evolve during the simulation

10.2.2 Simplifying Environments

Another approach that can be taken to verifying a model is through a simplification or limitation of the model environment. By subjecting agents to simplified environmental conditions, it is possible to clearly identify the behaviours that do not align with expectations. Within a spatial-simulation context, simplification of this kind could be achieved through switching complex geographical spaces (in raster or vector form) for abstract spaces, for example. Abstract spaces might consist

of uniform or hypothetical environments, where the extent of environmental variation is constrained. One may also wish to simplify environmental factors in turn, so that the effect of each can be assessed.

An example of this form of verification can be found in the Rainfall model (see Section 6.4.1 for an introduction to this model). This model is designed to operate using a GIS raster input, which defines the behaviour of raindrops in relation to their flow and accumulation within the model environment. An interesting feature of the model is that it allows the testing of agent behaviour in alternative abstract spaces. These include a flat plane, a cone shape and a simple hill. Figure 10.2 illustrates the different patterns that arise when these environments are used. When using these simpler environments, the movement of agents can be easily foreseen, and performance judged accordingly. Deviation from expectations within these environments would clearly present cause for further investigation.

Box 10.2 Exercise

If you have not already done so, experiment with the Rainfall model, which is available in the accompanying online material. If you change the landscape (MapType) to Flat, Cone and Hill, does the water behave as expected?

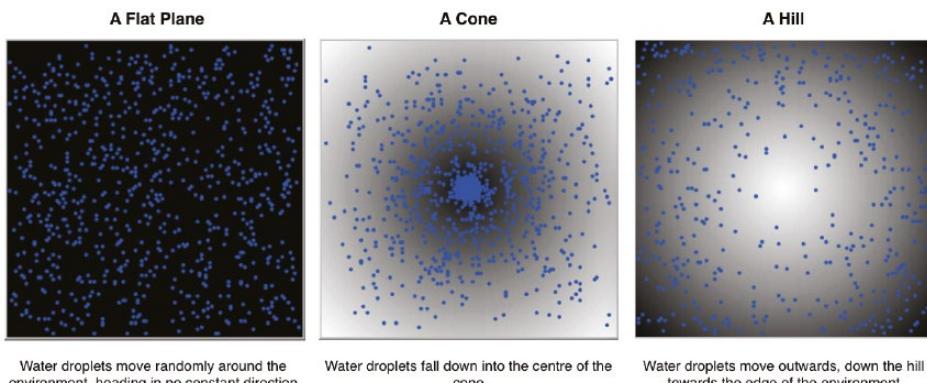


Figure 10.2 Simplified types of environment in the Rainfall model that can be used to test the behaviour of the agents (rain droplets)

The code to implement these different environment types is relatively straightforward. The model contains a parameter called MapType which determines the

type of environment to use. This can either be `CraterLake` (the GIS data that represents the real study is), `Flat`, `Cone` or `Hill`. When the type is `CraterLake`, the GIS raster data is read in as normal (see Section 6.4.1 for details), but with the other three options the environment is created manually. Code Example 10.2.1 demonstrates how this is accomplished.

Recall that the `elevation` patch variable is the one that the agents use to work out which direction is downhill. Therefore creating a flat environment is the most straightforward: the `elevation` of each patch is simply set to 0:

```
ask patches [ set elevation 0 ]
```

For a cone environment, where the lowest elevation is in the centre, the `elevation` of each patch can simply be set to be the distance away from the patch at the centre of the environment – defined as the patch with (x, y) coordinates $(0, 0)$: `patch 0 0`. Hence patches near the centre will have low elevation values, with elevation increasing away from the centre. In this way, agents should travel downhill towards the centre.

```
ask patches [ set elevation round distance patch 0 0 ]
```

Finally, to create a hill, the cone environment needs to be reversed so that the patches with the *greatest* elevation are in the centre. This can be accomplished by simply subtracting the distance from the centre from a large number (300 in this case):

```
ask patches [ set elevation 300 - round distance patch 0 0 ]
```

Code Example 10.2.1 Creating simple environments with which to test the behaviour of the agents

```
if MapType = "Flat" [
  resize-world -71 71 -71 71
  ask patches [ set elevation 0 ]
]
if MapType = "Cone" [
  resize-world -71 71 -71 71
  ask patches [ set elevation round distance patch 0 0 ]
  show_elevation
]
if MapType = "Hill" [
  resize-world -71 71 -71 71
  ask patches [ set elevation 300 - round distance patch 0 0 ]
  show_elevation
]
```

10.2.3 Expected Outcome Alignment

At a higher level of verification, one seeks to ensure that the general patterns of simulation dynamics align with expectations. Given that calibration and validation will improve model fit further, at this stage the only requirement is to verify the nature of the ‘cause and effect’ links that are represented in model design and simulation dynamics. In assessing this aspect of the model, it is good practice to build a verification test plan that incorporates a range of model adjustments with expected model responses. Example tests might include that an expectation of a 10% population increase (model design) leads to a 10% increase in GDP (simulation outcome). Note that these tests only assess simulation processes rather than alignment with real world data (which follows later); however, these tests are important for guarding against complex and unforeseen interactions that lead to unexpected outcomes.

An excellent example of the use of a verification test plan can be seen in Jackson et al. (2008). Set within the context of residential dynamics simulation, the authors implement 13 different verification tests and expected outcomes to measure the performance of different model processes. The tests cover a broad array of starting assumptions in relation to rent prices, population levels and agent behaviour, and these are linked with expected outcomes within the simulation. The authors then assess whether the model adjustment does lead to the expected outcome, and where this is not achieved, they explore and explain this aspect of design. The verification process employed by the authors not only provides openness in terms of model construction, but also improves the parsimony, and likely also the fit, of the final design.

10.2.4 Docking

A reliable method of verification is to implement a model of the same process twice or more, ideally using different programming languages. This process can be termed *docking* (Axtell et al., 1996). In aiming to advance our understanding of the world, progression cannot be assured without equivalent models being confirmed under alternative conditions. By implementing a model through alternative approaches or within a different simulation environment, we gain a basis for direct comparison and potential advancement. A truly verified model of the world should be able to stand up to scrutiny in a variety of forms and implementations.

There are various examples of the use of docking in the literature, usually where macroscopic or system dynamics models have been reimplemented within an agent-based modelling framework. Rahmandad and Sterman (2008) present a direct comparison of agent-based modelling and differential equations in exploring disease diffusion. Another, more direct implementation of docking is found in Brown et al. (2004), where basic mathematical models of urban development are contrasted with agent-based models incorporating heterogeneity and bounded rationality.

Likewise, in Sasaki and Box (2003), the macroscopic von Thünen model of urban development is successfully implemented as an agent-based model, although a direct comparison of the approaches is not made. In the original paper, Axtell et al. (1996) demonstrate equivalence through docking between the well-known SugarScape (Epstein and Axtell, 1996) model and the Axelrod (1995) culture model; however, the authors also note multiple logistic and technical issues with achieving equivalence. In many cases it may be unfeasible to invest the time and resources required to completely reimplement an alternative model.

10.3 Calibration

One of the most common uses of calibration is to manipulate the parameters of the model so that it matches some observed conditions. This can be achieved by running the model with different parameter values and comparing the modelled results to some observed data using the statistics outlined in Chapter 9. Once a model has been configured to match the target data as closely as possible, it might be able to uncover something about the real system. For example, the parameters that have the largest impact on the model might also be those that are the most important in the real world. Calibration can also be used to estimate the value of parameters that are unknown or cannot be directly observed. Finding an optimal value for a parameter that, for example, controls some element of an agent's behaviour might reveal useful information about the theory that underpins the behaviour. For example, Birks et al. (2012) use their model of crime to identify which of three complementary behavioural theories are the most important in representing criminal behaviour. Finally, calibration can be used to better understand the dynamics of a model. By adjusting the values of parameters in some range we are able to identify redundant parameters (those that have a minimal impact on the model outcomes), and those that the model is particularly sensitive to.

The basic process of calibration is illustrated in Figure 10.3. To begin with, the model is executed and synthetic data is generated. This is compared to observations in the real world and the difference between the two data sets, the amount of error, can be calculated. The size of the error is often termed *fitness*, where model configurations that produce the lowest error are said to have the greatest fitness. The model parameters are then adjusted and the model re-executed in order to produce an outcome with a lower error. The process is repeated until the model produces a sufficiently low error. The 'acceptable' level of error is subjective and depends on both the system under study and the motivations of the modeller. However, zero error is not necessarily desirable. The observed data will most likely contain an element of random noise, so if a model replicates the observed patterns too closely it might describe this random noise as well as (or instead of!) the underlying 'true' system dynamics. This 'overfitting' is discussed in Section 10.4.

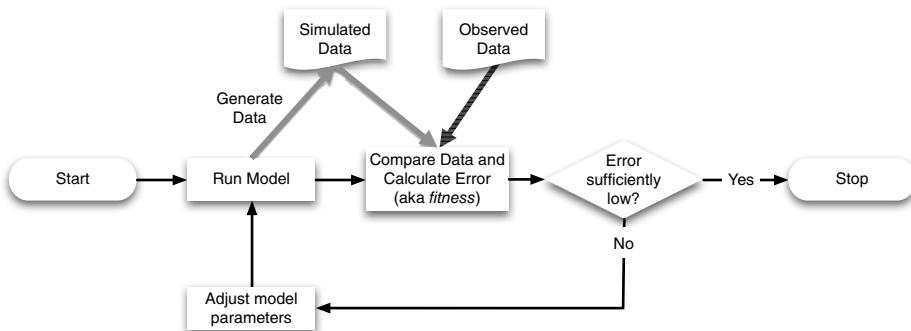


Figure 10.3 An overview of the process of calibration

With simple models, the process of finding model configurations that produce the lowest error is usually straightforward. This is particularly true with models that have few parameters, and those that have a linear relationship between parameter values and model outcomes. However, agent-based models are usually very complicated. They can not only contain a large number of parameters, but also often exhibit nonlinear relationships between parameters and model outcomes. This makes it difficult to trace the path of causality from individual parameter values, to individual agent behaviours, and finally through to the aggregate outcomes. Hence it can be problematic to predict how the model will behave under new parameter configurations. Even if good data is available to calibrate a model with, it might be difficult to know which agent rules to change in order to improve the accuracy of the model!

To further confound the situation, the process of calculating error is non-trivial. Even if a suitable error statistic can be found, which parts of the model should be evaluated? Is it sufficient to compare aggregate outcomes (e.g. counts of occurrences of some phenomenon across space) or should the model be calibrated against some individual-level data that assesses the ‘correctness’ of the behaviour of *individual agents*? Therefore rather than simply fitting the model to some data source, the process of calibration is often as much about using our intuition to tweak model parameters as it is about using quantitative methods to find combinations of parameters that bring the model outcomes closer to the target data that the model is being calibrated against.

This section considers the qualitative and quantitative methods that can be used to calibrate an agent-based model. What will become clear is that model evaluation requires a multidimensional approach. Axtell and Epstein (1994) classify the process into four progressive stages related to improving model performance: building from basic ‘face’ validity, to qualitative agreement with expected macroscopic patterns, to quantitative alignment at the macro scale, and finally through to quantitative confirmation using micro-scale data sets. Within this schema, progression across each

stage provides increased assurance of the validity of a model and can potentially aid understanding of the system of interest.

10.3.1 Qualitative Calibration

Qualitative calibration refers to the process by which a model is adjusted until it looks correct without the need for detailed analysis. This process can involve only the modeller, engaging their own intuition and subject knowledge, or additional assistance from external experts and stakeholders.

The process by which a modeller develops and calibrates through intuition is often called *face validation*. Many modellers will engage in this form of calibration without even knowing it – by maintaining a broad idea of how the simulated process is expected to appear and evolve, a model might be adjusted until something ‘looks correct’ to the modeller. This judgement may be gained through observations of the model or through production of statistics, charts and mapping. In assessing the design of model, the modeller should pay attention to both the behaviour of individual agents (or *micro-validity*; Wilensky and Rand, 2015, p. 332) and the formation of collective patterns of behaviour (*macro-validity*), in addition to how these patterns emerge over time.

In many simulations of real world phenomena, face validation will act as a first step towards a more in-depth empirical calibration and validation process, but in others, where hypothetical behaviours and interactions are the main focus, or there is an absence of suitable calibration data, intuition represents a good approach to model development. There are numerous examples within the NetLogo Models Library where this form of validation has been engaged. This is particularly clear in the simulation of bird flocking, ant line formation and wolf sheep predation, which are conducted in a hypothetical setting and therefore are designed to ignore other potentially informative environmental and behavioural aspects that might be useful for real world prediction. Nevertheless, the models in their basic form are both informative and maintain *face validity* by expressing intuitively believable animal behaviours. In the ant line model, for example, the modeller is able to observe that the movements of individual agents are in line with expectations (e.g. agents walk in a line, and make small individual corrections) and that the formation of macroscopic patterns is as would be expected (e.g. an ant line is formed and then optimised).

In some cases it may be difficult for a modeller to judge the validity of the simulation output, due to either limited contextual knowledge or the available data lacking the suitable richness required for simulating the entire process (Bharathy and Silverman, 2010). In these situations, a modeller may look to external subject-matter experts to provide their view on the representativeness of the simulation. There are a variety of ways in which this involvement can take place. The ‘companion modelling’ approach requires deep interaction, and aims to integrate stakeholders and decision-makers within the modelling process,

in terms of the design of decision-making models and modification of model parameters (Barreteau et al., 2003). It has been argued that this approach adds a deeper richness to simulation development, through a focus on model precision (i.e. alignment with behaviour) rather than accuracy (i.e. alignment with data) (Moss, 2008). Within an alternative framework, the Werker–Brenner approach (Werker and Brenner, 2004), expert testimony is integrated following a stage of empirical validation, helping to identify the most robust model solution from a set of valid alternatives (Bharathy and Silverman, 2010).

10.3.2 Quantitative Calibration

Quantitative forms of calibration focus on identifying a set of parameter values that best reflect the patterns observed in a real world system of interest. The process of undertaking calibration is broken down into the following six steps in Railsback and Grimm (2011, p. 262):¹

1. Calibration should be limited to a few uncertain and important parameters. These parameters can be identified through exploratory sensitivity analysis (discussed below), or through intuitive understanding of the model design. Being highly selective on the number of test parameters leads to a considerably less complex calibration process.
2. Decide on whether calibration should pertain to a *categorical* solution (involving a range of potential calibration values) or a *best-fit* solution (where one value is identified as an optimal setting).
3. Decide a strategy for handling temporal variation in parameter setting performance. If time-series variation is an important part of the model, then a measure that integrates differences over time should be incorporated (e.g. mean difference, maximum error, RMSE; see Section 9.3). Some scenarios may wish to assess model fit once a time step is reached or equilibrium is achieved.
4. Identify a pattern that can be reproduced, and for which observation data exist. Observation data should be generated through the same behavioural mechanism and conditions as captured by the agent-based model.
5. Conduct a series of simulation experiments that explore the settings of parameter values via a *parameter space search*, discussed in more detail below.
6. Analyse the calibration experiment results, and identify the parameter value settings that best represent the observation data.

¹ See the original text for a full explanation.

In the rest of this subsection we will outline some technical methods for calibrating the model within the process defined in Railsback and Grimm (2011).

Parameter Space Search

A parameter space search involves exploring the relationship between different values of calibration parameters and the resulting simulation outcomes. In conducting a search (or *sweep*), each calibration parameter is tested within an expected range of values. For each additional calibration parameter used, an additional dimension is added to the analysis, meaning that each combination of parameters

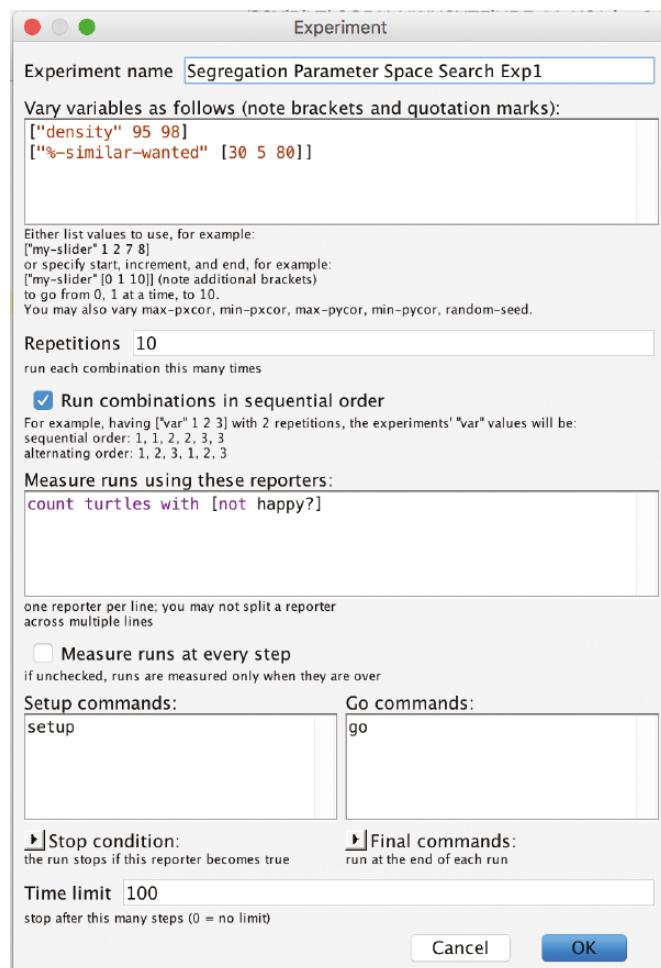


Figure 10.4 NetLogo’s BehaviorSpace interface

is tested by the end of the calibration process. Furthermore, where the model incorporates stochasticity, multiple simulations should be run, and the mean performance and error calculated, to account for variation. Following the completion of the search process, hundreds or thousands of simulations may have been executed, and this is important to consider when planning the simulation design.

Prior to beginning the search process, it is vital that the data being generated through the simulation are both aligned with observation data and relatively easy to interpret. Each of the potentially many simulation outcomes will need to be ranked and compared in some way, and therefore the specification of these measures is of clear importance. The specific metric used will depend entirely on the context, but one should be aware of the need to measure not only correlation between observed and simulated volumes or flows of agents (macro-validity), but also agent-level variation from expected behaviour at the extremes, summarised as errors (micro-validity).

In BehaviorSpace, NetLogo provides an easy-to-use tool for simulating the effect of different parameter values on model outcomes. The toolset enables the setting of parameter values to specific ranges and increments, the execution of repeated runs, and specification of custom reporter functions. Figure 10.4 shows how to set up a parameter space search for the Segregation model (which is available in the Models Library and was introduced in Section 2.4.1):

Variables. In this case we have decided to explore how the variation of density and %-similar-wanted affect the evolution of simulation dynamics. Note that we have set the two search ranges differently – for density we will use two settings, one at 95% and another at 98%, while for %-similar-wanted we will test 5% increments within a range between 30% and 80%. This indicates that we will conduct the simulation 22 times to cover the full extent of the parameter range.

Repetitions. The Segregation model incorporates random elements in the initial placement and movement of agents, and, as such, we should attempt to control for this through execution of multiple model iterations. Here we have chosen 10 each, taking our total number of simulations to 220.

Reporters. Perhaps the most important aspect of the search process is judging the most appropriate reporter to use in reflecting the simulation outcomes. In this case we have chosen to focus on the percentage of agents that are not happy. This measure will summarise the relationship between density, agent preference and overall happiness. Note that the measure is defined in the NetLogo script. Additional measures can be incorporated, and we have the option to record these values at each time step, as noted in the checkbox.

Setup and go. These options tell BehaviorSpace which functions set up and begin the simulation.

Time limit. The time limit specifies the point at which the simulation is stopped and the reporter value exported for that simulation run. This entry can be left at zero to allow the simulation to eventually stop naturally, but, given that the Segregation model can potentially run forever in some cases, we have limited the time extent to 100 time steps.

Once we have finished with design, we can launch our experiment and execute the number of required simulations. NetLogo provides options to export the results to a spreadsheet or text file, and can execute multiple simulations in parallel.

Box 10.3 Exploration

The BehaviorSpace Segregation model experiment has been provided for you to run your own experiments. First of all, execute the experiment that we have provided (and discussed above) and see which combination of parameter values leads to the greatest levels of unhappiness in the simulation. Once you have done that, see if you can identify an improved set of parameter combinations that increases unhappiness even further. You may also wish to experiment with different reporters – is there a better function for measuring the simulation dynamics than simple unhappiness? If you are feeling extra daring, see if you can calculate a measure that accounts for larger and smaller cluster sizes.

Sensitivity and Uncertainty Analysis

Once a set of reference parameter values have been derived through a parameter space search, a secondary stage of calibration is undergone to more deeply explore the sensitivity and robustness of the proposed model to varying conditions. This stage analyses in greater depth the relationship between model parameters and simulated outcomes, highlighting the most sensitive components of the model and their likely operating range, helping us to more deeply understand and explain the model.

Sensitivity analysis is the most widely used method for testing the stability of a simulation. This process introduces a quantitative measure of the effect that small adjustments in parameter values have on a given simulation metric. The approach is typically applied to one parameter at a time, known as the *one factor at a time* approach, with the remaining parameters and conditions held constant. The process evolves through adjustment to the value of a parameter of interest through application of a factor of either 5% or 10%. Once the adjustment is made and the simulation executed, the effect of the adjustment can be reached through an approximation of the partial derivative of the simulation metric with respect to

the parameter. A simplification of this calculation is usually used in the form of the proportional changes S in the simulation measure Y relative to parameter X ,

$$S_{Y,X} = \frac{\delta Y / Y_0}{\delta X / X_0},$$

where δY represents the change in simulation metric and δX the change in parameter value (adapted from O'Sullivan and Perry, 2013). The result is a set of sensitivity measures for each parameter, with values that are relatively high or low compared to other parameters, indicating the impact that the change in the parameter has on the resulting simulation metric. As a result of this process one may be able to identify aspects of the model that could be removed in the interests of parsimony, without severely impacting the simulation.

Additional insight into the performance of the simulation can be gained through uncertainty analysis. This approach is similar to sensitivity analysis, but involves adjustment of multiple parameters in order to more accurately account for uncertainty and address the resulting interactions that are driven by uncertainty. In this context, uncertainty is applied through a probability distribution that is defined for each parameter. The exact distribution depends on the nature of the uncertainty that is expected from that parameter, but can be continuous or discrete, drawn from a Gaussian, log-normal or any other kind of distribution. The hypothesised distribution will ideally be constructed using the means and standard deviations drawn from observation data. Once these uncertainty distributions are defined for a set of parameters (recalling that large numbers of calibration parameters increase computational requirements), the simulation is executed many times (e.g. 500 runs), with new parameter values drawn from the uncertainty distributions at each iteration. As a result of these simulations, it is possible to construct a frequency distribution of simulation output measures which captures variation in likely simulation outcomes. The relative dispersion of the distribution will provide an indication of how uncertain the simulation result can be expected to be, with a wider distribution indicative of larger uncertainty, and a narrow distribution indicating a more deterministic outcome.

Heuristic Searches

Heuristic search methods offer an alternative approach for identifying reference parameter values. While parameter space searches can help us get towards an adequate solution, modern optimisation methods are better adept at precisely and speedily identifying optimal solutions from a vast set of alternatives. Where parameters are particularly important or highly sensitive, precisely identifying the best parameter setting becomes all the more vital. Heuristic

searches operate in the same way as BehaviorSpace worked, through provision of a range of parameter value settings and a simulation outcome measure (in this case, known as an *objective function*).

These approaches fall into two broad categories: genetic algorithms and hill climbing algorithms. Genetic algorithms (GAs) are based on the biological process of natural selection, whereby ‘stronger’ genes are passed on to future generations, with the potential for gene crossover and mutation. Within the GA context for parameter value search, the GA will explore, combine and test different parameter combinations, iteratively progressing towards an optimal configuration. At each stage of ‘evolution’, only the solutions closer to reaching the desired simulation outcome are advanced, but by incorporating randomness and crossover there is natural compensation for the potential of other combinations of parameter values.

Hill climbing (or the converse, hill descending) methods are directed search processes, where the solutions are explored locally in relation to their progression towards the optimisation function. Only solutions which better match the objective function than a current configuration are continued; all other solutions are rejected. If the search algorithm reaches a plateau (e.g. there are no improvements on the existing location in the search space), then a new location is selected randomly and the process continues. Following repeated failed searches for improved locations, the hill climbing algorithm concludes its search. NetLogo provides another set of tools for conducting heuristic search via the BehaviorSearch application.

The interface is similar to BehaviorSpace in that parameter values and ranges are defined; however, rather than using a reporter metric, BehaviorSearch takes an objective function which is to be maximised or minimised during directed search. The search algorithms provided include GA, in addition to hill climbing (a stochastic hill climber) and hill descending (simulated annealing) algorithms. Random search is also included as a baseline for performance measurement. BehaviorSearch is a stand-alone application, found within the NetLogo installation directory, but has the ability to read in the parameters of any NetLogo model. The application includes a detailed tutorial which is recommended prior to use.

10.3.3 Quantitative Calibration Example: WalkThisWay

To conclude the discussion about quantitative calibration, this section will briefly outline a real world quantitative calibration example. The example has been published by Crooks et al. (2015a) and the model itself (entitled ‘WalkThisWay’) was created in MASON (see Appendix A.7 for more details). Here we show the model reimplemented in NetLogo, which is available in the accompanying resources. The aim of the model was to simulate the movement of people as they cross an environment (i.e. a scene) and to calibrate the model against observed pedestrian movement paths derived from trajectory data extracted from CCTV data.

Crooks et al. (2015a) derived information from the CCTV data on which entrances and exits pedestrians took while traversing the scene, along with the routes that were traversed and recorded, in the form of a heatmap which aggregates individual pedestrian traces for each pixel over the course of a day (see Figure 10.5).

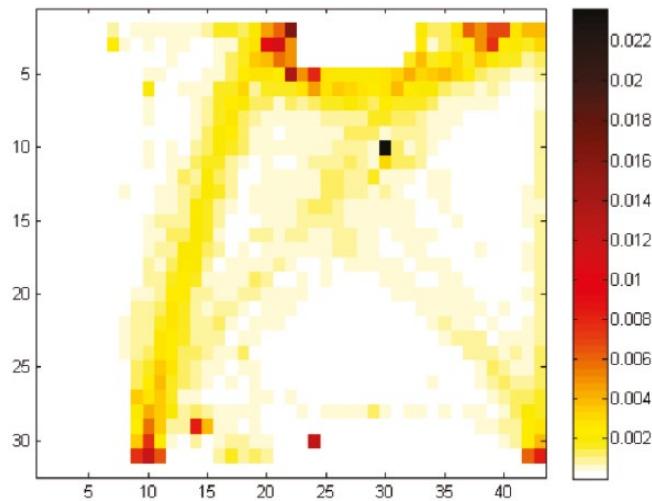


Figure 10.5 A normalised heat map generated from real trajectory data of people's movements (Crooks et al., 2015a)

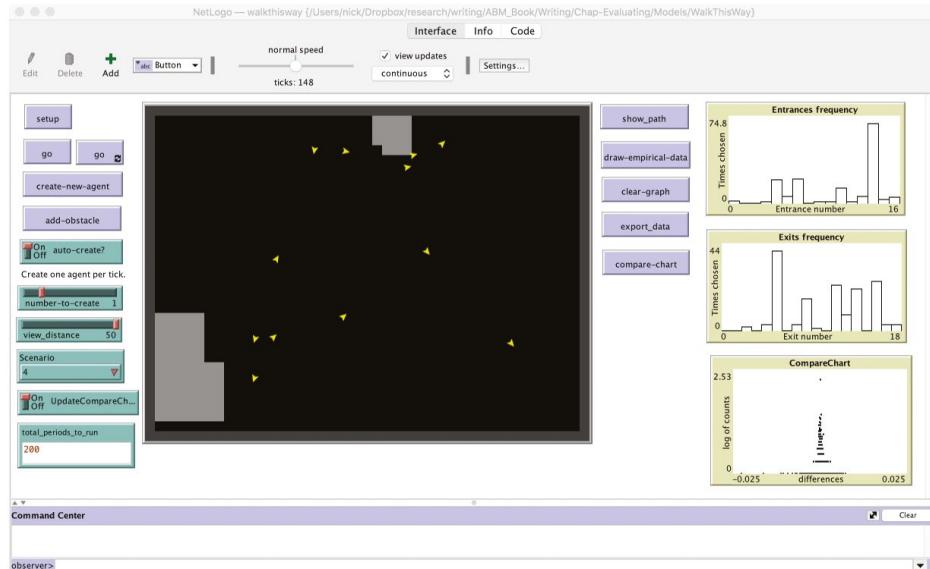


Figure 10.6 The WalkThisWay model, reimplemented in NetLogo

Using such information, Crooks et al. (2015a) were able to calibrate their model. In this reimplementation this information was also used. For instance, on the right-hand side of Figure 10.6 we have information pertaining to which entrances and exits the agents chose along with a chart that compares the differences between over- and under-predication in each simulation run, where the y -axis is the log of the number of pixels having a given error value from the heatmap comparison of the real world movement data. Based on these model outputs, the modeller could change the model logic and increase agents' vision to better reflect patterns observed from the real world (i.e. the heatmap).

10.4 Validation

The previous section hinted at a danger associated with calibration: that of *overfitting*. Overfitting occurs when a model is trained extensively on an observed data set. Henceforth, rather than representing the underlying system dynamics the model simply reproduces the observed data, *including any random noise*. The resulting model would perform poorly were it reconfigured to represent new scenarios. A commonly used solution to this problem is to segregate the available observed data into two discrete sets: *training* data used to calibrate the model, and *test* data used to validate the model. During the validation phase, the model is not reconfigured as it would be during calibration. If the model performs well on the test data then we can be more confident that it has not been overfitted and will perform well on new input data that represent scenarios previously unforeseen (by the model). As illustrated by Figure 3.2, if the model performs poorly in the validation step then it might mean that the design is flawed and the model, in its current form, is not a reliable conceptualisation of the system that it is attempting to reflect. Typical partitions into training and test data sets are 75/25 or 80/20, which in most cases ensure that the model is trained on diverse data. In cases where only a small potential training sample is available, *cross-validation* can be used to assess average performance of the model through testing using multiple random train and test partitions of the data.

Although the motivation for validation is different from that of calibration, the actual processes of conducting calibration and validation are very similar. Like calibration, the model is executed to generate synthetic data and this data is compared to an observed data set. Then the same methods that are used to assess error for calibration can be applied to the new synthetic and observed data sets used for validation.

10.5 Discussion

Robustly evaluating agent-based models remains one of the key challenges within the discipline. The lack of rigorous evaluation methods is an area that is openly

criticised by both proponents and critics of agent-based models (see Balci, 1996; Windrum et al., 2007; Lee et al., 2015). Methods for evaluating agent-based models remain greatly underdeveloped, with the attention of researchers often being diverted to creating new applications powered by new forms of big data. Why is this? Simply, evaluation of agent-based models is extremely difficult. The complex nature of agent-based models (many heterogeneous, interacting agents) makes model evaluation a particular challenge. This challenge is made even more difficult with the introduction of space.

As noted above, there is no formal methodology for evaluating agent-based models, but many authors have come to settle on a standard process that consists of three separate procedures: *verification*; *calibration*; and *validation*. Within this chapter, variations of the different approaches used for evaluating models have been presented. Verification (Section 10.2) of the model can be achieved through an array of methods including code testing, simplifying environments, expected outcome alignment and docking. Calibration, the process of adjusting model parameters to match the behaviour to that of the observed, was introduced in Section 10.3. Before calibration can occur, the key question is what are we calibrating against: pattern, process or some other metric (see Section 10.3)? The two main types of calibration, qualitative (Section 10.3.1) and quantitative (Section 10.3.2), were presented with guidance given about how to undertake them. Finally, validation, the mechanism by which models are tested against unknown data, was introduced in Section 10.4. The methods used for validation are often the same as those employed for calibration. An area of increasing importance that can be facilitated through improved methods of visualisation is that of stakeholder validation (Barnaud et al., 2008). This input from stakeholders creates an important link, not only in evaluating the performance of the models, but also in moving these models towards acceptance within policy arenas.

One of the central issues with creating a robust evaluation of agent-based models is the lack of appropriate data. As has been highlighted throughout the book, agent-based models require a great deal of individual-level data just for constructing appropriate rule sets and parameterising the model. To carry out a rigorous evaluation of how the model is performing, even more individual-level data is required. The emergence of ‘big data’ has opened up the possibility of rich sources of individual-level data that can be used for evaluation. However, there are issues associated with these new data forms that need to be addressed, including data harmonisation and a lack of appropriate tools to mine the data. As Heppenstall et al. (2016) commented, there is some irony in the fact that the continued disaggregation of variables of social and spatial systems to give sufficient heterogeneity to allow for better representation has meant that it is nearly impossible (at present) to rigorously validate and calibrate.

However, one area where progress has been made is within pedestrian modelling, which has experienced a rapid proliferation of data due to motion capture techniques (Torrens, 2012; Crooks et al., 2015a). Despite the progress in this area, there remains room for improvement, especially in unknown conditions such as riots (Torrens and McDaniel, 2013). Another area where evaluation is performed with more rigour is within land-use changed models. Here results are validated by comparison at the smallest unit (normally pixel-by-pixel). However, this form of analysis is restricted to raster formats of data; sadly, validation of models using vector data is less developed (Kocabas and Dragičević, 2009).

Advances are being made in the area of evaluation, but as Axtell remarks, ‘there is a large research program to be done over the next 20 years, or even 100 years, for building good high-fidelity models of human behavior and interactions’ (quoted in Weinberger, 2011). Creating a set of rigorous evaluation methods along with guidance for researchers is key to the future success and uptake of agent-based models.

Chapter Summary

Evaluation remains one of the key challenges within agent-based modelling. The nature of agent-based models means that evaluation often has to be tailored to a particular application and based around available data and knowledge of the system. However, the agent-based modelling community has made efforts to create a process of evaluating these models.

This chapter has presented an overview of the three main stages – verification, calibration and validation – with the key methods commonly used presented. This can be used by the reader as a basis to design evaluation for their own applications.

10.6 Annotated Bibliography

For an excellent chapter summary on ‘Verification, Validation, and Replication’, readers are referred to:

- Wilensky, U. and Rand, W. (2015) *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. Cambridge, MA: MIT Press.

Their work has more detail on some aspects of evaluating models than described here, particularly with respect to replicating models, but focuses less on the spatial aspects of validation.

Railsback and Grimm (2011) has a valuable chapter on ‘Parametrisation and Calibration’ of models:

- Railsback, S.F. and Grimm, V. (2011) *Agent-Based and Individual-Based Modeling: A Practical Introduction*. Princeton, NJ: Princeton University Press.

For a wider theoretical discussion of agent-based modelling, motivation and validation, readers are directed to:

- Axtell, R. and Epstein, J.M. (1994) Agent-based modelling: Understanding our creations. *Bulletin of the Santa Fe Institute* (Winter), 28–32.
- Axtell, R. (2000) Why agents? On the varied motivations for agent computing in the social sciences. Working Paper 17, Center on Social and Economic Dynamics (The Brookings Institute), Washington, DC.

Not covered in this chapter is a discussion of empirical validation methods, a dominant approach in economic agent-based modelling. For coverage of this area, readers are directed to:

- Windrum, P., Fagiolo, G. and Moneta, A. (2007) Empirical validation of agent-based models: Alternatives and prospects. *Journal of Artificial Societies and Social Simulation*, 10(2), 8. Available at <http://jasss.soc.surrey.ac.uk/10/2/8.html>.

For a comparative analysis of ‘empirical’ and ‘companion’ validation approaches, readers are referred to:

- Moss, S. (2008) Alternative approaches to the empirical validation of agent-based models. *Journal of Artificial Societies and Social Simulation*, 11(1), 5. Available at <http://jasss.soc.surrey.ac.uk/11/1/5.html>.

11

ALTERNATIVE MODELLING APPROACHES

Chapter Outline

Agent-based modelling is one of the most popular approaches used in social and spatial simulation. However, there are several other alternative approaches that are commonly used, including cellular automata, microsimulation, discrete event simulation, system dynamics and spatial interaction models. This chapter presents an overview of these other approaches, giving simple examples of how they can be used and summarising the main differences between them. To compare them, the chapter will consider the same scenario throughout: that of the spread of a disease modelled using a susceptible–infected–recovered epidemic model. This shows that while the same general patterns emerge, the mechanics can be very different. The chapter ends with a discussion and summary.

11.1 Introduction

While this book is primarily concerned with agent-based modelling, there are a set of related techniques that have been used to study real world systems (see also Section 2.1). This chapter explores the differences between agent-based models and these comparable aggregate (e.g. spatial interaction and system dynamics models) and individual-based (e.g. cellular automata, microsimulation, discrete event models) approaches.

The chapter gives a broad overview of each of these methods (Section 11.2), before comparing and contrasting different elements (Section 11.3). Each of the techniques is then applied to the scenario of simulating a disease outbreak using a susceptible–infected–recovered epidemic model (Section 11.4). This exercise demonstrates that while the models produce similar patterns, the dynamics of how the disease propagates through the population is very different.

The main aim of this chapter is to highlight the main similarities and differences of these different approaches. By understanding the capabilities of each of the methods, a decision about which approach is the most suitable for a particular research question can be made. Readers wishing to know more about the techniques are referred to Iltanen (2012), Birkin and Wu (2012), Gilbert and Troitzsch (2005), Batty (1976) and Sterman (2000). Further details can be found in the annotated bibliography at the end of the chapter (Section 11.6).

11.2 An Overview of Different Modelling Approaches

11.2.1 Cellular Automata

The first automaton models that focused on modelling geographical phenomena came from the area of *cellular automata* (CA). The basic features of CA have been well documented in the literature (see Wolfram, 2002, for a thorough overview). Essentially, a cellular automaton is a grid (i.e. a lattice) containing cells of equal size, whose behaviour is specified in terms of local relationships between cells. Like an agent in an agent-based model, each cell has a *state* that can encompass many different features (e.g. the amount of grain on a land parcel, the type of building in a city block, the happiness of residents in a neighbourhood, ‘yes’ or ‘no’ answers to a political question). A cell’s state is determined by the state of its neighbours (see Figure 11.1), by a set of local rules and by the cell itself (see Wolfram, 2002; Benenson and Torrens, 2004; Iltanen, 2012). From a geographical perspective, the first notable use of a cellular automaton was for the purpose of urban growth and land use (e.g. Nakajima, 1977; Tobler, 1979). Due to the relative ease of implementation, cellular automaton modelling remains a popular choice for simulating large-scale urban phenomena such as urban sprawl (White and Engelen, 1993; Clarke et al., 1997; Al-Ahmadi et al., 2009).

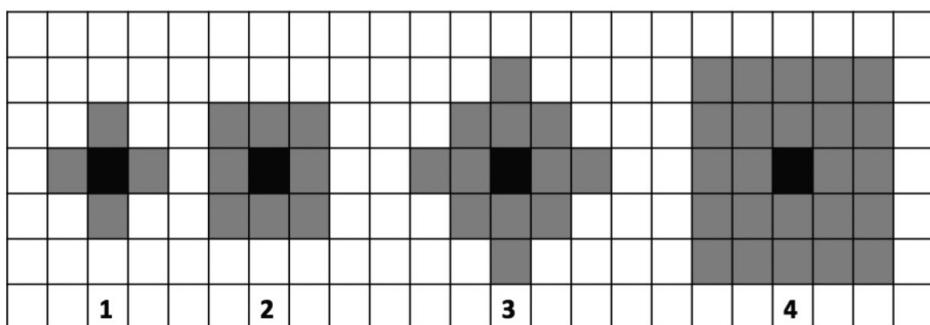


Figure 11.1 Diagrammatic representation of 2D cellular automata and the most commonly used neighbourhoods: (1) von Neumann neighbourhood; (2) Moore neighbourhood; and extended (3) von Neumann and (4) Moore neighbourhoods

To illustrate how cellular automaton models work, we will consider Conway's Game of Life (Gardner, 1970) which was reimplemented by Wilensky (1998). This is a simple model which operates over a rectangular lattice. A cell has only one attribute and one of two states, either 'dead' or 'alive'. The rules of the model are simple. Each cell checks the states of itself and the eight surrounding cells in its Moore neighbourhood and, depending on the states of these cells, it will do one of three things:

1. If the current cell is alive, and two of its neighbours are alive, then the cell remains in the alive state; else it dies.
2. If the cell is currently dead, it will become alive (reproduce) if three of the neighbouring cells are alive.
3. If the cell has fewer than two neighbours in the alive state, or if more than three neighbours are alive, then the cell will die.

To illustrate this, Figure 11.2 shows these rules in action over time. Specifically, it shows how 'dead' cells (white) become 'alive' (black) depending on the number of alive cells around them. Some patterns are stationary – for example, the first four patterns on the left of each image – whilst others oscillate – for example, the three patterns on the right of each image. Others disperse across the space, such as the glider in the bottom of each image. With certain initial starting configurations, it is possible to create patterns of great complexity¹ and, as many might say, beauty. This demonstrates that from simple well-defined rules, complex patterns can emerge over space and time. Each cell processes information

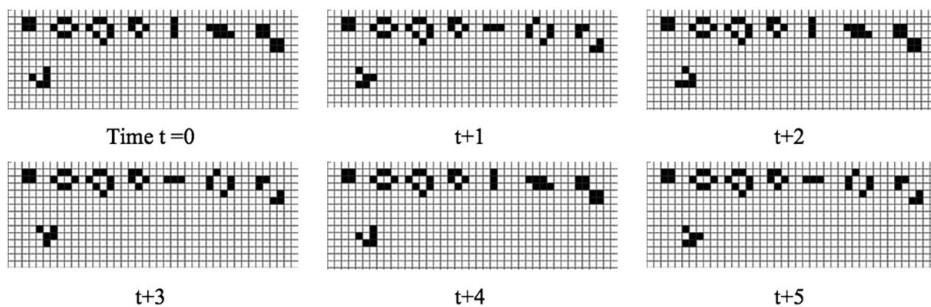


Figure 11.2 Example of cells changing state from dead (white) to alive (black) over time depending on the states of their neighbouring cells (modified from Wilensky, 1998)

¹ For some nice examples of what you can do with the Game of Life, see <https://www.youtube.com/watch?v=C2vgICfQawE>

from external sources and reacts to these sources based on rules. The key message is that, depending on the transition rules and the current state of the cells, different levels of complexity can emerge.

11.2.2 Microsimulation

Related to CA are *microsimulation models* (MSMs), whose origins lie in economics (Orcutt, 1957). MSMs have been widely used to study the impact of social, administrative and environmental changes on populations. This methodology is commonly used to create small-area microdata such as individual-level synthetic populations. Figure 11.3 shows how individual-level data, such as the Sample of Anonymised Records (a sample of individual-level data from the UK Census), for a given spatial area are used to create a representative synthetic population of that area (i.e. a micro-file). Broadly speaking, there are two main types of MSM: static and dynamic. In static MSMs, the individual entities do not change and are often used for short-term predictions, such as the immediate effect of a policy change such as tax reform (Gilbert and Troitzsch, 2005). Dynamic MSMs move the population of individuals through time (e.g. individuals age, get married, etc.) based on transition probabilities to generate future scenarios (see Harland et al., 2012, for a discussion of the different methods used to achieve this) as shown in Figure 11.4. In this example individuals age over time (such as individual *a*), some have a chance of dying (such as individual *g*), while others are born (e.g. individual *h*), the probabilities calculated via transition matrices which update the micro-file.

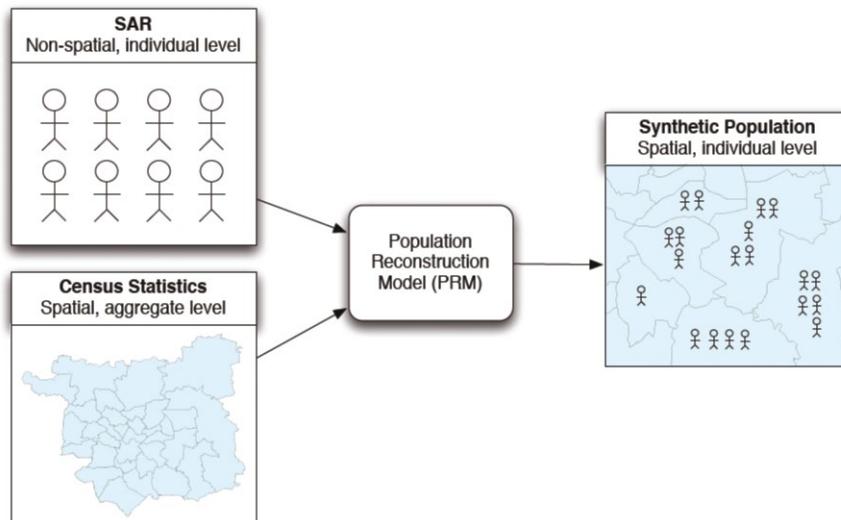


Figure 11.3 Schematic outlining the basic process of creating a synthetic population using microsimulation



Figure 11.4 A simple example of dynamic microsimulation process where individuals change over time (m = married, d = divorced, w = widowed)

Similar to CA, dynamic MSMs operate at the level of the individual to simulate the global and local changes whilst allowing the characteristics of each individual to be tracked over time. Moreover, as transition probabilities are based on observed correlations (e.g. from known data such as average birth rates), the models can be finely calibrated to real sample data. This has resulted in MSMs being widely taken up in geography with applications within transport (e.g. MATSim (Horni et al., 2016) and TRANSIMS (Nagel et al., 1997)),² population dynamics (e.g. SimBritain; Ballas et al., 2005) and health (e.g. Smith, 2012). For interested

² It should be noted that some researchers call individuals in MSMs ‘agents’, but we do not consider these to be agent-based models. Specifically, we note that the line between MSMs and agent-based models is becoming very blurred. There exist MSMs that incorporate minimal social networks to guide their decision-making (e.g. the demand for elder care as modelled in Gilbert and Troitzsch, 2005) and agent-based models in which interaction occurs exclusively through environmental variables (e.g. the most basic, baseline implementation of the SugarScape model of Epstein and Axtell, 1996). The two approaches are moving towards more common ground as MSMs add more behavioural and spatial interaction between individual units and agent-based models add both space and demographic characteristics to their agents (Birkin and Wu, 2012; Horni et al., 2016; Wise et al., 2017).

readers, Birkin and Wu (2012) provide a detailed overview of the diversity of spatial applications of microsimulation and note how such an approach could be leveraged within agent-based models for building robust agent populations based on real world data.

11.2.3 Discrete Event Simulation

A widely used tool for modelling workflow management and traffic dynamics is *discrete event simulation* (DES), also commonly referred to as ‘queuing models’. This style of modelling dates back to the development of queuing theory, which was pioneered by Erlang (1909) in the early part of the twentieth century by modelling the Copenhagen telephone exchange. In such a model, the world is abstracted into three main objects: customers, servers and queues (see Gilbert and Troitzsch, 2005). As shown in Figure 11.5, some *source* generates individual entities (i.e. *customers*) that arrive based on some probability distribution at a queue in front of a *server*; these customers are then served by the server (which is often a stochastic process) and then proceed to a *sink* (or to the next queue and server). In such models, time is neither continuous nor discrete but from one event to the next event; for example, an event could be considered as the customer being served. The aim of such models is to create efficient processes with minimal resources (e.g. number of servers). In the case of a bank, for example, how many tellers (i.e. servers) are needed to serve the customers? If the number of customers is low and the number of tellers is high, some tellers will have nothing to do, which could be considered a waste of resources. Alternatively, if there are a large number of customers and only one teller, the queue might be long and customers become frustrated by the lack of service and decide to use another bank (e.g. Hammond and Mahesh, 1995). DES can therefore be used to explore what the optimal number of servers should be. What is appealing about such a style of modelling is that arrival rates and serving times can be calibrated on known data – for example, flow data if available.

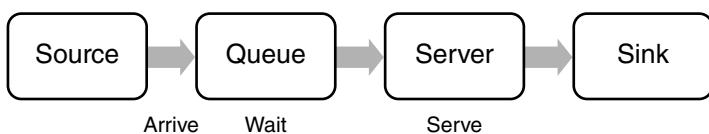


Figure 11.5 Simple example of a discreet event simulation

To further highlight how DES works, let us take the example shown in Figure 11.6 which simulates the waiting time of passengers as they go through a security checkpoint at an airport (Bybee and Eng, 2012). In this example the servers are Transportation Security Administration employees, customers

are passengers and the queue is the security lines. The time it takes to go through security is also a stochastic process, in the sense that different people can pass through at different rates (i.e. someone with little hand luggage versus someone with a lot). The source is the world outside of entrance and the sink is the departure of the customers on the aeroplane. Within the model we can explore how arrival rates and the number of customers that a server can process can impact wait times. DES has been used in a number of applications ranging from logistics of goods and services (e.g. Dong and Chen, 2005) and health delivery (e.g. Aaby et al., 2006), to traffic interactions (e.g. Anokye et al., 2013) and pedestrian models (e.g. Smith, 1991).



Figure 11.6 Airport security line simulation (Bybee and Eng, 2012).

11.2.4 System Dynamics

While the modelling styles presented above focus on individuals, there are a number of modelling styles that look at the world in its aggregate form. One of these is the *system dynamics* (SD) approach which was pioneered by Forrester (1969) and has been widely used in social simulation over the last 50 years for applications ranging from arms races to economic development, tourism and the selling of used cars (see Sterman, 2000; Cioffi-Revilla, 2014). SD is a method for

developing and testing stochastic differential equations via computational simulations. In SD models, the system being modelled is looked at as an indivisible whole in the sense that there are no individuals, but the system is broken down into functional aspects (Gilbert and Troitzsch, 2005). The basic elements are *stocks*, *flows* (e.g. inflow and outflow) and some sort of flow regulator, as shown in Figure 11.7. Similar to DES models, there are *sources* and *sinks*, and inputs flow from a source to a stock based on a specified rate (also known as the flow value).

To give a simple example, imagine the system being modelled is a bathtub. Our source would be the tap; water would flow into the bathtub and the stock would be the amount of water in the bathtub. The rate at which the bathtub fills is dependent on the inflow rate from the tap, and outflow would occur when the bathtub is drained. While this is a simple example, in the sense that there are no feedbacks or time delays (which can also be captured with SD models, as will be shown in the next example), SD models provide means to document the key stocks of the system being modelled and explicitly link (i.e. via flows) how such elements interact over time.



Figure 11.7 Basic elements of a system dynamics model

To build on the bathtub analogy, let us take another simple example, that of wolf–sheep predation as shown in Figure 11.8 (Wilensky, 2005a). Here we have two stocks (sheep and wolves), with the inflow of new sheep dependent upon both birth rate and sheep flow out of the system (if they are eaten by wolves based on a predation rate). Wolves flow into the system dependent on the stock of sheep and their efficiency in capturing sheep (i.e. feedback loops are introduced here). In this example, sheep populations grow and decline over time similarly to the wolf population. However, there is a delay in the system as the wolf stock is dependent on the sheep stock. Various population dynamics are produced which are similar to those produced by Lotka–Volterra equations seen in predator–prey ecosystems.

11.2.5 Spatial Interaction Models

Another widely employed aggregate technique used for geographical simulation is that of *spatial interaction modelling*. Spatial interaction models emerged as a dominant modelling technique for exploring spatial phenomena during the 1960s and 1970s. Such models are usually used to predict the size and direction

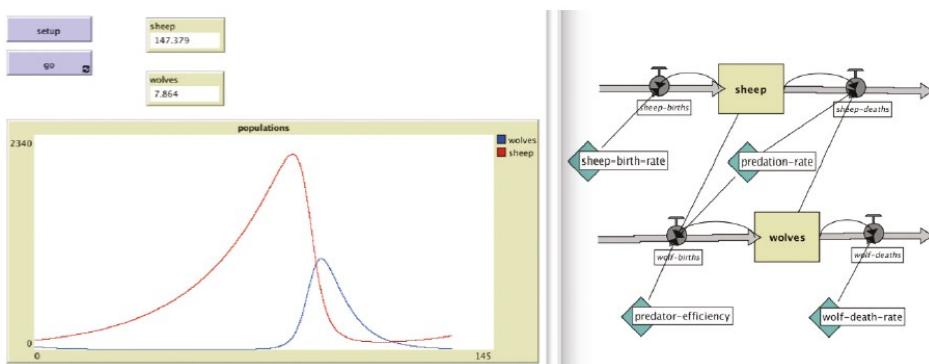


Figure 11.8 An example of a wolf–sheep predation model implemented as a system dynamics model (Wilensky, 2005a)

of spatial flows (e.g. of information, people and goods) between discrete places when changes occur to either the origin or destination of these interactions. This is achieved through the use of independent variables that measure the structural property of the spatial area being modelled. For example, Torrens (2000) highlights how the spatial pattern of journey-to-work flows might be predicted using structural variables such as the distribution of workers, the distribution of employment and the cost of travelling to work.

Early versions of spatial interaction models, also known as ‘gravity models’, include the works of Ravenstein (1885), Reilly (1929) and Zipf (1946) which were used to study migration and retail trade (such as Reilly’s ‘law of retail gravitation’), along with studies of traffic-flow and newspaper circulation (see Haggett et al., 1977). These models were based on mathematical assumptions analogous to Newton’s law of universal gravitational attraction. The gravitational model has a direct analogy in geography: the greater the distance, the less travel occurs because of the distance decay effect which discourages flows over longer distances.

Spatial interaction models are generally used to model the flows between places and thus allocate activity to places, focusing on aggregates. However, they explain little about the behaviour of individuals. Closer examination of individuals is, however, essential to help with our understanding of the underlying processes and dynamics that produce the aggregate patterns that we observe. For example, knowledge concerning cities is socially produced, and the very process of using that knowledge as part of planning is itself a social process – individuals are key to this. As Batty (1979) writes: ‘the nature of the knowledge involved, its origin, and the motivations for using it are quite different from those governing the science of inanimate physical systems, and consequently this difference should be reflected in the form of theory itself’.

However, one cannot discount spatial interaction models altogether. Fotheringham et al. (2000) write that ‘spatial interaction models still get criticised for what they once were, rather than for what they are now’. This is despite the fact that spatial interaction modelling is one of the most applied geographical techniques and the main engine of generic land-use transportation models, with a strong emphasis on the importance of place. They have been used in studies of retail location, including the evaluation of the impacts of new out-of-town shopping centres and the creation of ‘food deserts’, health-care planning/provision, the optimum location of new car dealerships in the motor industry, and the prediction of new accounts being opened at a particular location within the financial service market (see Birkin et al., 1996, for examples in a broad range of applications). Spatial interaction models have also been used in forecasting the demand for housing, in regional population projections, forecasting travel demand between urban areas (Wilson, 2000) and estimating national and subnational migration patterns (Dennett and Wilson, 2013).

11.3 Comparing Different Modelling Approaches

In the previous section we briefly introduced the dominant methods used to simulate geographical systems. In this section we compare and contrast them with agent-based modelling in order to demonstrate what makes the latter unique for the study of such systems. In Table 11.1, which is adapted from Gilbert and Troitzsch (2005), we outline the key differences. Here, the number of levels refers to whether the modelling technique is capable of simulating interactions between levels (e.g. the individual and the society). This is important when studying emergence in geographical systems. For example, how individuals buy and sell houses influences how property markets emerge; conversely, the state of the property market impacts how, when and where individuals buy a house. Communication between agents refers to whether or not it is possible to pass information between agents, while complexity of agents refers to how sophisticated the agents are, whether they are homogeneous (possessing the same characteristics and rule sets) or heterogeneous (different characteristics and rule sets) which we term ‘low’ and ‘high’, respectively. The number of agents refers to how many agents are represented in the system under exploration. System dynamics models, unlike the other modelling techniques, do not model individuals or space *per se* but rather the system as a whole (e.g. the city or the country). In such models only one representative agent is captured, while individual-based approaches can represent anything from one agent (e.g. Schelling, 1971) to millions (e.g. Epstein, 2009) at varying levels of complexity. Similarly, spatial interaction models model only one level but can contain many agents; however, unlike agent-based models, there is no communication between agents.

Table 11.1 A comparison of modelling techniques used to study aspects of geographical systems (adapted from Gilbert and Troitzsch, 2005)

Modelling style	Number of Levels	Communication between agents	Complexity of agents	Number of agents
System dynamics	1	No	Low	1
Spatial interaction	1	No	Low	Many
Microsimulation	2	No	High	Many
Discrete event simulation	1	No	Low	Many
Cellular automata	2	Yes	Low	Many
Agent-based models	2+	Yes	High	Many

Whilst CA, DES and microsimulation allow for the simulation of individuals, they do have several drawbacks compared to agent-based models. For example, CA models tend to be spatially constrained to a grid, and although each cell can be in a different state, they are all driven by identical transition rules, which is also the case for DES. This makes it impossible to capture a range of unique individual behavioural traits (i.e. heterogeneous behaviours as in agent-based models). Microsimulation only allows the modelling of one-directional effects: the impact of the policy on the individuals. With microsimulation there is no behavioural modelling capability (as everything happens through transition matrices) and, perhaps most importantly, individuals do not interact with each other *per se* (Gilbert and Troitzsch, 2005). This is also similar to DES models, where all individual entities are the same. The ability to create individuals with multiple attributes and behaviours who can move freely within a space and interact with other individuals is critical if we are to create tools that allow new insight into the dynamics and processes of geographical systems and to understand the impact of individual patterns of behaviour.

This is exactly what agent-based modelling attempts to do. It emerged at a time when computing power and storage were rapidly increasing, along with new developments in object-orientated computer programming languages that were well suited to the development of agent-based models. It is at this point in the 1990s that agent-based modelling trickled into the social sciences, most notably through the work of Epstein and Axtell (1996) who demonstrated how agent-based models could be extended from modelling people to growing entire artificial societies, an area that they termed *generative social science*. Because agent-based modelling generates emergent phenomena from the bottom up, it raises the issue of what constitutes an explanation of such a phenomenon. According to Epstein and Axtell (1996), agent-based models ‘may change the way we think about explanation in the social sciences. What constitutes an explanation of an observed social phenomenon? Perhaps one day people will interpret the question, “Can you explain it” as asking “Can you grow it?”’.

It is this thinking that has percolated into the agent-based modelling community, leading to new ways of examining real world systems, ‘not from a traditional modelling perspective but from the perspective of redefining the scientific process entirely’ (Bonabeau, 2002). Rather than favouring deduction (testing of assumptions and their consequences) or induction (the development of theories on the basis of observations), there is a third way (Axelrod, 1997a). The researcher starts with a set of assumptions, but then employs experimental methods to generate data that can be analysed inductively (Gilbert and Troitzsch, 2005).

11.4 A Practical Comparison: The SIR Model

To compare and contrast how these models work and how their underlying mechanisms generate outputs, a common problem was selected: the spread of disease modelled using a susceptible–infected–recovered (SIR) epidemic model. The motivation for this comes from the system dynamics model outlined by Shiflet and Shiflet (2014) which was implemented in NetLogo (see the accompanying online resources). For the remaining techniques (i.e. the agent-based model, cellular automata and discrete event simulation), models were created from scratch using NetLogo. A microsimulation and a spatial interaction model were not implemented as they would produce similar results to the DES and SD models, respectively. This is due to how they represent agents and the transition from one state to another, which are very similar. The model-building approach is introduced in Sections 11.4.1–11.4.4, and the results from the four models are compared in Section 11.4.5.

11.4.1 The System Dynamics Approach

In Shiflet and Shiflet’s (2014) SD model, one person is infected at the start of the simulation. Infected people can infect susceptible people. The population of infected individuals will always increase by a quantity given by multiplying together the number infected, the number susceptible, the infection rate and the change in time (dt). Infected individuals may recover. The number of people who will recover in an iteration is always given by multiplying together the number of infected, the recovery rate and dt . Figure 11.9 illustrates the SD process, while Figure 11.10 shows the SIR process as a flowchart.

11.4.2 The Agent-Based Approach

As is the case for the SD model, at the beginning of the simulation, one agent is infected. Agents are randomly distributed on the landscape, and at the beginning of each iteration, they turn to a random direction and move forward by one cell. During each iteration, an infected agent may infect other agents in the same cell. This is different from how the SD model works, specifically with

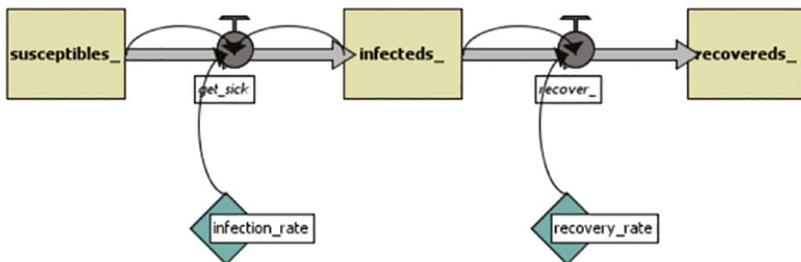


Figure 11.9 System dynamics process (Shiflet and Shiflet, 2014)

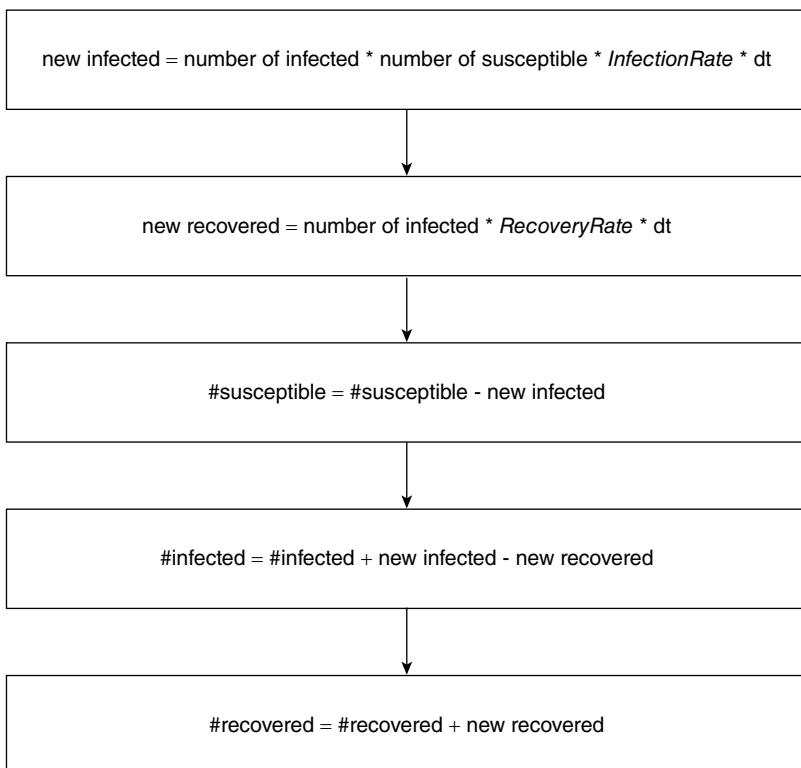


Figure 11.10 System dynamics flowchart

regard to the probability of getting infected. In the SD model, the infection rate is the infection rate in the entire population. In the agent-based model, the probability of becoming infected is equal to the infection rate divided by the probability of an agent being in the same cell, multiplied by the change in time. Each infected agent has a probability of recovering in each time period, which

equals to the recovery rate times the change in time. The equations in the agent-based model are the following:

$$P(\text{get infected}) = \frac{\text{infection rate}}{P(\text{same cell})} dt,$$

$$P(\text{recover}) = \text{recovery rate} \times dt,$$

where $P(\text{same cell})$, the probability of being in the same cell, equals 1 divided by total number of cells; and dt is the change in time. Figure 11.11 illustrates the agent decision process, while Figure 11.12 shows the display of the agent-based model.

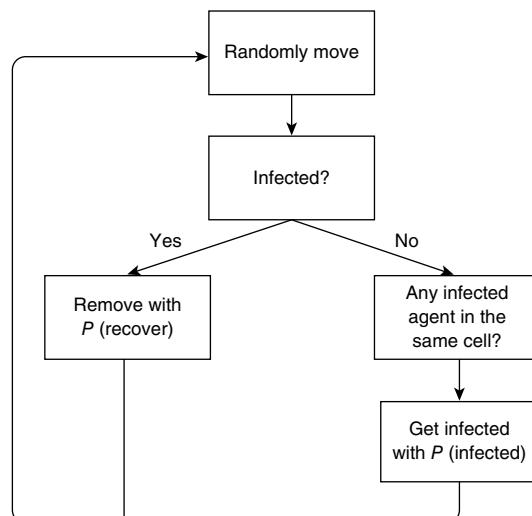


Figure 11.11 Agent-based modelling: agent decision process.

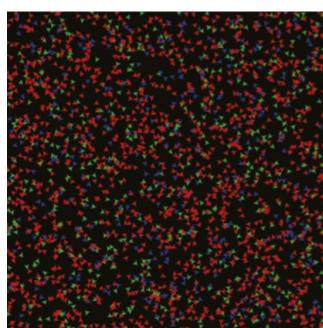


Figure 11.12 Display of the agent-based model: green = susceptible, red = infected, blue = recovered

11.4.3 The Cellular Automata Approach

At the beginning of the simulation, one cell is infected. During each iteration (dt), the infected cell can infect other cells within its Moore neighbourhood (i.e. the eight surrounding cells). The landscape is n by n square, with n being equal to the square root of the number of people to be created at the beginning of the simulation. Wrapping is enabled both horizontally and vertically. The probability of becoming infected is equal to the infection rate divided by the probability of being in the Moore neighbourhood, multiplied by the change in time. Each infected cell has a probability of recovering in each time period, which is based on the recovery rate multiplied by the change in time. This behaviour can be described by the following equations:

$$P(\text{get infected}) = \frac{\text{infection rate}}{P(\text{Moore neighbourhood})} dt,$$

$$P(\text{recover}) = \text{recovery rate} \times dt.$$

Figure 11.13 shows the changing process of the cells, while Figure 11.14 shows the display of the CA model.

11.4.4 The Discrete Event Simulation Approach

In a DES or queuing model there are three abstract types of objects: servers, customers and queues; this approach is different from the CA and agent-based models. To implement a SIR model as a DES, servers are the processes of becoming infected and recovering. Customers are individuals susceptible to infection, and

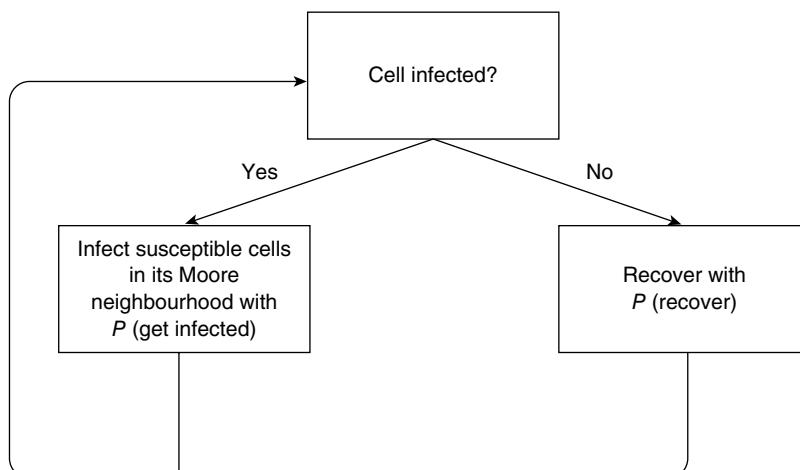


Figure 11.13 Cellular automata cell changing process

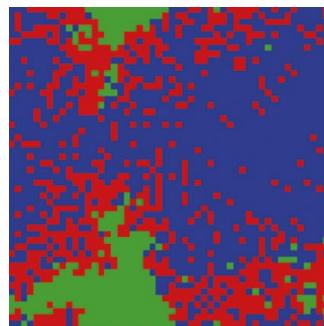


Figure 11.14 Display of the CA model: green = susceptible, red = infected, blue = recovered

infected people are waiting to recover. We assume there are two queues in this model. As susceptible objects (i.e. individuals) are created, queues for infection are formed while people are waiting to be infected. On the other hand, as people get infected, they form a second queue waiting to recover. During each iteration (dt), each object in the queue has a probability of becoming infected and a probability of recovering. After agents recover, they enter the sink of recovered people. The equations can be written as follows:

$$P(\text{get infected}) = \text{infection rate} \times \text{number infected} \times dt,$$

$$P(\text{recover}) = \text{recovery rate} \times dt.$$

The whole process is illustrated in Figure 11.15.

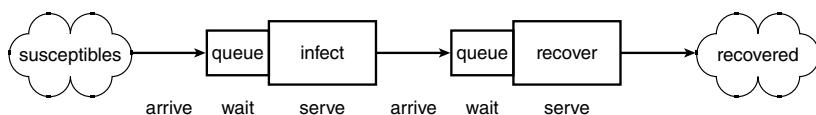


Figure 11.15 Discrete event simulation process

11.4.5 Comparative Analysis

We now examine the results from the four different models. The default parameters used in each model are: number of susceptible people at set-up = 2500, infection rate = 0.002, recovery rate = 0.5, change in time (dt) = 0.001; the number of people in each status are recorded. Since the SD model has no randomness (i.e. deterministic) and will always give the same result, it was run only once. The other three models were run 10 times, the average was taken and the results are shown in Figure 11.16. The stop condition is that there are no individuals remaining to be infected.

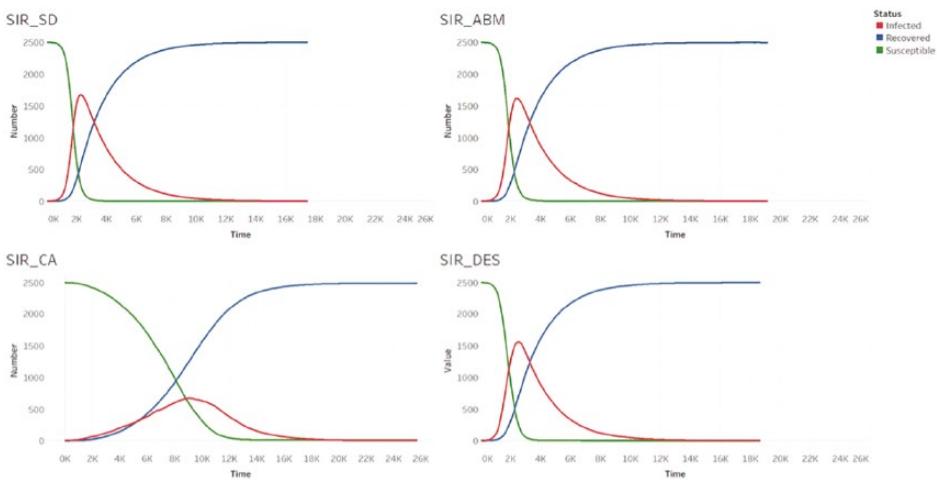


Figure 11.16 Results for the different models. Clockwise from top left: SD model, agent-based model, DES and CA

The same general pattern can be observed in the results of the four models. As the number of susceptible people decreases, the number of infected people first increases before decreasing again, with the number of recovered individuals increasing over time. However, each model realisation also shows differences in how such patterns play out. The SD model has the smallest number of iterations before no one is infected. The number of iterations shown on the graph is the average of the ten runs (except for the SD model which only has one run). The SD model took 17,451 iterations to finish, while the agent-based model took an average of 19,145 iterations and the DES model took 18,645 iterations. The CA model took the longest time for no more individuals to be infected, with 25,680 iterations.

The patterns produced by the SD, agent-based and DES models have several similarities. The number of infected people first increases and reaches a peak of over 1500 over more than 2000 iterations (2272 for SD, 2403 for agent-based model, 2538 for DES). The CA model, however, never reaches this level of infection, due to the slower spread of the disease through the diffusion mechanism of the CA model. An important characteristic of the SD model is that there is no randomness; each simulation will result in the same patterns being produced. In the other models, the likelihood of infection or recovery is dependent on a probability function; this introduces an element of randomness that produces different results each time the models are run. Furthermore, individuals in the SD and DES models are homogeneous, with the same probability of infection or recovery. In the agent-based and CA models, people (represented by moving agents or static cells) are heterogeneous in that they each have different locations. Only susceptible people around an infected individual can be infected. It is interesting to note

that where individuals are mobile – for example, in the agent-based model – the result is similar to the SD model. However, the agent-based model takes more time to recover (19,145 iterations in agent-based model and 17,451 iterations in SD). When people are not mobile and the number of individuals is limited (one individual in one cell in the CA model), the rates of infection and recovery are slower.

To test the sensitivity of the models to a specific parameter, the infection rate in each model is increased from 0.002 to 0.02 and the results shown in Figure 11.17. As would be expected, as the infection rate increases, the number of susceptible people decreases at a much faster rate. However, the SD, agent-based model and DES patterns are still similar to each other, while the infection in the CA model is slower. The average number of iterations for these models are: 15,807 (SD), 15,252 (agent-based model), 16,937 (CA), 16,677 (DES). By increasing the infection rate the total number of iterations of each model has decreased, with the CA model still taking the longest time to converge. The peak numbers of infected people in each model are on average: 2363 people at 255 iterations (SD), 2310 people at 363 iterations (agent-based model), 2035 people at 1019 iterations (CA) and 2340 people at 286 iterations (DES). The CA model takes a longer time and reaches a lower peak.

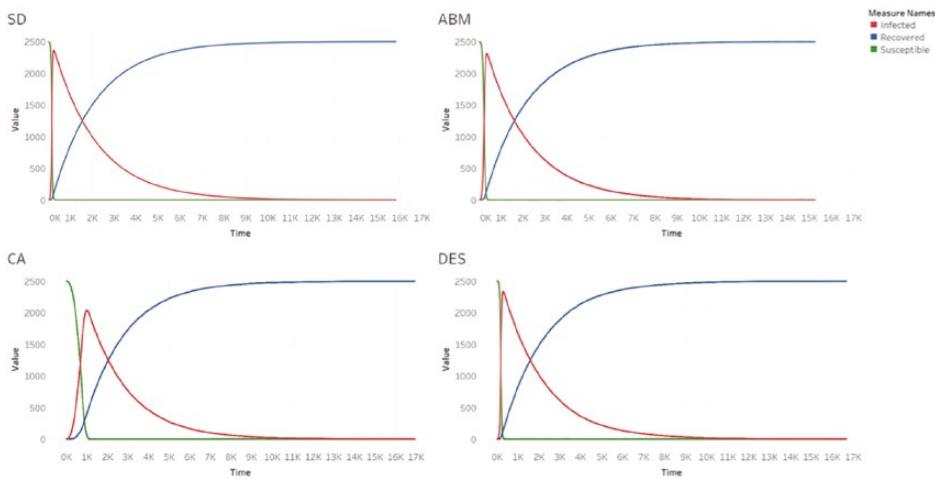


Figure 11.17 Results for the different models with infection rate 0.02. Clockwise from top left: SD model, agent-based model, DES and CA

These models are only simple examples of how a SIR model can be implemented in different modelling techniques, but in reality, if we were to model disease propagation in more detail we would need to consider many other things such as that people could be both moving through space (i.e. travelling to work) or static (i.e. staying at home), and the capacity of each cell is always limited in some respect (see, for example, the work of Crooks and Hailegiorgis, 2014).

11.5 Discussion

This chapter has presented the most commonly used styles of model that can be found within the social sciences. These models typically operate at either the aggregate (e.g. SD and spatial interaction models) or disaggregate (e.g. CA, DES, MSM and agent-based model) level. The purpose of this was to showcase how agent-based models are different from other styles of modelling, as well as to help to solidify the concepts and terms presented in Chapters 2 and 3. Specifically, the ability to incorporate direct agent–agent interaction and create heterogeneous agents makes this modelling approach a unique tool with which to study populations from the bottom up. Section 11.4.5 highlighted that the same phenomena can be explored using a variety of techniques. While each of the approaches produced the same general trends, only through the agent-based model can direct interactions between individuals be simulated.

Chapter Summary

In this chapter agent-based models have been contrasted with other common modelling approaches used in the social sciences. The purpose of this was to give an overview of these approaches and, through experimentation, to highlight their differences from agent-based modelling. In agent-based models, patterns emerge from the direct interaction of multiple heterogeneous agents from the bottom up. In such models we have individual agents, which is not the case in SD or spatial interaction models. The use of the SIR example demonstrated that diseases do not spread via spatial diffusion (i.e. from one cell to another cell as in the CA model) but by direct agent–agent interaction.

11.6 Annotated Bibliography

Readers interested in spatial interaction modelling and how it can be used are referred to:

- Batty, M. (1976) *Urban Modelling: Algorithms, Calibrations, Predictions*. Cambridge: Cambridge University Press.
- Birkin, M., Clarke, G.P., Clarke, M. and Wilson, A.G. (1996) *Intelligent GIS: Location Decisions and Strategic Planning*. Cambridge: Geoinformation Group.

To implement spatial interaction models using R, see the resources collected by Adam Dennett:

- https://rpubs.com/adam_dennett

For a comprehensive review of system dynamics models, readers should see:

- Sterman, J.D. (2000) *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Boston: McGraw-Hill.

For more information on CA models, readers are referred to the works of Stephen Wolfram:

- Wolfram, S. (1994) *Cellular Automata and Complexity*. Reading, MA: Addison-Wesley.
- Wolfram, S. (2002) *A New Kind of Science*. Champaign, IL: Wolfram Media.

For a more geographical take on the use of CA, see:

- Iltanen, S. (2012) Cellular automata in urban spatial modelling. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 69–84. Dordrecht: Springer.

For a good summary of microsimulation, its application and its difference to agent-based models, see:

- Birkin, M. and Wu, B. (2012) A review of microsimulation and hybrid agent-based approaches. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 51–68. Dordrecht: Springer.

Lastly, for a more detailed discussion of all the techniques discussed in this chapter and their history, readers are referred to the following excellent resources:

- Borshchev, A. and Filippov, A. (2004) From system dynamics and discrete event to practical agent based modeling: Reasons, techniques, tools. In M. Kennedy, G.W. Winch, R.S. Langer, J.I. Rowe and J.M. Yanni (eds), *Proceedings of the 22nd International Conference of the System Dynamics Society*. Albany, NY: System Dynamics Society.
- Gilbert, N. and Troitzsch, K.G. (2005) *Simulation for the Social Scientist* (2nd edn). Maidenhead: Open University Press.
- Shiflet, A.B. and Shiflet, G.W. (2014) *Introduction to Computational Science: Modeling and Simulation for the Sciences* (2nd edn). Princeton, NJ: Princeton University Press.

12

SUMMARY AND OUTLOOK

Chapter Outline

This chapter reflects on the current state of the art of agent-based modelling and discusses factors that may shape the future of this discipline. Specifically, we will look at the key challenges for developing robust agent-based models of geographical systems and discuss some potential solutions. We argue that we need to make progress on these challenges if these models are to be used to offer insight into key societal challenges, such as climate change, urban growth and migration.

12.1 Introduction

The intention of this book has been to provide a primer for agent-based models and geographical information systems. As such, we hope that through the chapters we have highlighted how agent-based models can be used to study geographical systems, and through the examples, demonstrated how geographical information can be used as the grounding of our ‘artificial worlds’. Chapter 1 introduced geographical modelling and how we can view such systems through the lens of complex systems. An introduction to agent-based modelling was given in Chapter 2, before a framework for the design and development of agent-based models was presented in Chapter 3. We then began our focus on agent-based model development, through a tutorial on NetLogo in Chapter 4. Following this we outlined the key concepts underlying GIS methods and data (Chapter 5) before discussing how to integrate GIS with agent-based models (Chapter 6).

We then explored two important elements of agent-based model design, namely how we can incorporate human behaviour into agent-based models (Chapter 7) and how networks (both physical and social) can be incorporated into models (Chapter 8). In assessing our models (in Chapters 9 and 10) we outlined methods for understanding and evaluating the trends we find in our data and those produced through our modelling. And in the penultimate chapter (Chapter 11) we compared agent-based modelling to other relevant modelling approaches. In this final chapter

the key challenges for developing and advancing agent-based models of geographical systems (Section 12.2) are discussed. Finally, we look at potential ways to overcome these challenges, provide a roadmap for future research (Section 12.3) and finish with a final discussion of this book (Section 12.4).

12.2 Remaining Challenges

The last 15 years have seen a sharp rise in the development and uptake of agent-based modelling. As discussed in Chapter 1, this is due to a number of factors, including increased computational power and availability of storage and data. There has also been a shift in how geographical and social systems are viewed, with current thinking recognising the social and physical networks that drive these systems. In order to simulate these systems, we need tools that are capable of representing heterogeneous individuals, interactions that occur between them and anything (information, trends, etc.) that emerges from these interactions. In this book we have painted a positive picture of how agent-based modelling and the use of GIS in such models can be used to study emergence. However, the field is not without its challenges (see Cioffi-Revilla, 2002; Axelrod, 2007; Torrens, 2010a; Filatova et al., 2013). Crooks et al. (2008) listed several challenges to the development of agent-based models; these will be revisited and reappraised here to set the scene for future research to make it more scientifically robust and applicable to policy.¹ These challenges range across the spectrum of theory and practice, hypothesis and application.

12.2.1 Reasons for Modelling

In the early days of computer modelling, models were built to test the impacts of policies rather than scientific understanding *per se* (Batty, 2008). The underlying notion was that, given a good theory, a model could be constructed which would then be validated and, if acceptable, implemented within policy-making (Batty, 1976). This notion has been relaxed over the last two decades, and models are now built to explore all stages of the theory-practice continuum (not just for prediction). This is especially the case for agent-based models, which range from exploratory to the predictive (see Parker et al., 2001; Castle and Crooks, 2006; Epstein, 2008). However, a model is only useful for the purpose for which it was constructed, and as modellers we need to be explicit about this and use appropriate verification and validation strategies. For example, if the purpose of

¹ However, readers should remember that agent-based models are already being used outside of academia, as we discussed in Section 2.4.4.

the model is for the discovery of new relationships, as illustrated by Filatova et al. (2009) exploring the evolution of land markets, the validation strategy might be theoretical comparison of price gradients. If the purpose of the model is to predict individual movement, as in the case of Crooks et al. (2015a), one needs to consider quantitative goodness-of-fit measures – for example, comparing the output from the individual interactions against aggregated data collected from the real world.

12.2.2 Theory and Models

The goal of theory is to make the world understandable by finding the right level of abstraction (or simplification; see Miller and Page, 2007). Traditionally in the social sciences, the role of a model was to translate a theory into a form whereby it could be tested, manipulated and refined. However, with agent-based models (along with computational modelling more generally) models are often being used to develop theory (Axelrod, 1997a). That is not to say that agent-based models are theoretically agonistic. There have been several attempts to operationalise existing theories (e.g. Diappi and Bolchi's, 2008, attempt at growing Smith's, 1979, rent gap theory from the bottom up with respect to gentrification) or to test existing theories (e.g. Pires and Crooks, 2016, as we show in Appendix A.12) or to relax some of the restrictive assumptions of past theories (e.g. adding dynamics and heterogeneous agents to the general equilibrium theory in economics; Gintis, 2007). However, a review by Groeneveld et al. (2017) found that out of 134 agent-based models of land use reviewed, only 51 made use of existing theory (the most common being expected utility theory) with respect to agent-based decision-making (which we will return to in Section 12.2.7). From a more pragmatic point of view, especially from the social science perspective, the challenge also relates to which of the many social theories should be tested. It has been noted that for complex systems such theories might not encompass all important aspects (see Schlüter et al., 2017, for examples). While we do not oppose changing the role of models and theory, the challenge with this shift is that in many agent-based models the theoretical implications of the model remain implicit and hidden behind many *ad hoc* assumptions which are often poorly articulated (we will return to how this can be addressed in Section 12.2.7).

12.2.3 Inter-model Comparison

While a growing number of models are being developed for particular applications, these tend to be based on case studies, one-off models or a proof of concept (Filatova et al., 2013; O'Sullivan et al., 2015; Bell et al., 2015). However, there is little in the way of inter-model comparison such as is seen in other areas. Take, for instance, models in the climate-science community such as the Community Ice Sheet Model and inter-comparison exercises based on the same initial conditions

(e.g. Ice Sheet Model Intercomparison Project; see Nowicki et al., 2016). From an agent-based modelling perspective we are a long way from this. Efforts are being made to compare different models applied to the same phenomena such as malaria (Ferris et al., 2015), Ebola (Chowell et al., 2017) or common features of models exploring the same phenomena such as growth along frontier regions (Parker et al., 2008) and slums (Roy et al., 2014) in order to find what constitutes the *must-have* features of models exploring similar issues. However, there are no centralised agent-based modelling repositories that pool together knowledge, code and data. For this to happen, significant community buy-in would be needed. This in itself is a challenge as agent-based models are being developed across multiple scientific fields and are often developed for specific purposes.

Moreover, most modellers do not compare their model to other models (agent-based models or others) that are exploring the same phenomena. Furthermore, unlike in the geospatial realm (e.g. <http://www.opengeospatial.org/standards/cdb>), there is no standard model or protocol on what constitutes an agent or its decision-making process (as discussed in Chapter 3). It is therefore often left to the model developer and the research question being asked to develop the definition of an agent, which further exacerbates the lack of standard practices in developing agent-based models (we will return to the need for better ways of documenting and sharing models in Sections 12.2.4 and 12.2.7). Perhaps the agent-based modelling community could take inspiration from the R and Python communities where packages are shared and widely used, but this would require modellers to choose specific platforms which could also have its drawbacks.

12.2.4 Replication and Experiment

Replication is one of the main principles of the scientific method; however, this is rarely done or is difficult to achieve in the sciences due to the difficulties in controlling all the variables that pertain to a particular situation (see, for example, Baker, 2016). We see this also in the case for agent-based models which have multiple parameters, methods and settings etc. It is often impossible to outline the entire logic of such a model in a paper (due to space constraints) but efforts have been made in this area with attempts at designing ontologies and protocols for providing a more detailed model description and aide comparison. These include the ODD protocol by Grimm et al. (2006, 2010), the ODD+D protocol of Müller et al. (2013) (of which several examples are shown in Appendix A) and the use of UML (Bersini, 2012) when documenting key model processes. But the use of such standards is still not the norm for many agent-based modellers. We would also argue that to aide replication and experimentation modellers should consider sharing their models and data. This is because the

computational model is the full specification of the theory that the model is built upon and without the codebase (and supporting material) it may be difficult to understand, replicate or experiment with. Thankfully, there are efforts to share and make code and data available, such as OpenABM (see <https://www.comses.net/>). However, as noted by An et al. (2014), code is often only understandable by other modellers. It is only through such activities that we can replicate and experiment with agent-based models. Similar efforts with respect to reproducibility of results are also being called on in the geocomputation community more generally (see Brunsdon and Singleton, 2015b).

12.2.5 Verification and Validation

Throughout this book we have discussed verification and validation (especially in Chapter 10). We list them as challenges with respect to agent-based models because if we are to address the challenge of replication and experimentation, our models should also be verified. To do this, we need to ensure that the formal logic of a computer program behaves as expected, which is an aspect that is often taken for granted. Researchers should document steps taken (e.g. through code walk-throughs and parameter testing) to verify the model.

Validation remains one of the biggest challenges (e.g. Batty and Torrens, 2005; Filatova et al., 2013). How can we rigorously evaluate how well the model matches the real world system it is attempting to simulate? From a geographical perspective, Torrens (2010a) notes that this is problematic because the discipline has few techniques for analysing individual agents in complex systems. Which statistics and types of sensitivity analysis should be used is an issue that has dominated the literature on models of land-use and land-cover community, for example (Brown et al., 2005; Pontius et al., 2008; Filatova et al., 2013). Mandelbrot (1983) argues that good models which generate spatial or physical predictions that can be mapped or visualised must ‘look right’. Axelrod (2007) suggests that to understand the output of an agent-based model, it is often necessary to evaluate the details of a specific simulation ‘history’, and this too is usually a qualitative matter. In contrast, researchers such as Axtell and Epstein (1994) see the validation of models on a scale from qualitative to quantitative, with the highest validation being for a model that attains quantitative agreements with both the emergent macro-structures and an individual agent’s micro-behaviour.

The challenge sets agent-based modelling aside from other traditional forms of modelling. Agent-based models embrace heterogeneous systems that evolve over time, where the linkages between dependent and independent variables are difficult, if not impossible, to observe due to their rich model structure (Batty and Torrens, 2005). Finding appropriately rich and detailed data to validate such systems is difficult.

12.2.6 Agent Representation, Aggregation and Dynamics

Throughout this book we have represented agents at a variety of levels (individuals, households, etc.) and have looked at dynamics operating at very different temporal and spatial scales (e.g. from seconds to years). However, little attention has been paid to how we chose such representation or dynamics. Normally modellers choose the agent, representation, dynamics, etc. to meet their research need. And while it might be easy to assign rules and behaviours to individuals, there is little discussion on how we aggregate these rules and behaviours from the individual level to, say, groups or higher aggregations of agents. Often with aggregations we lose details, and the question here is what details we should lose.

Another issue is how many agents and how many attributes for each agent we should account for. Agent-based modellers often use a sample population, but sampling is not yet a well-developed art in agent-based modelling. This representation also raises questions about what are the most appropriate methods for agents to communicate with each other (i.e. via Moore or von Neumann neighbourhoods or social networks). These questions are not new (see, for example, Cioffi-Revilla, 2002) but we would like to see modellers be more explicit with respect to their agent representations, why they chose specific spatial and temporal scales, and what data exists to support their assumptions and validate their outcomes.

12.2.7 Behaviour

The role of theory (as discussed in Section 12.2.2) and the representation of agents are both linked to the actions and behaviour embedded within agents. While Chapter 7 discussed common approaches researchers use to incorporate human behaviour in agent-based models, we (along with others, such as An, 2012; Filatova et al., 2013; Balke and Gilbert, 2014; Groeneveld et al., 2017; Schläter et al., 2017) would argue that modellers do not consider the use of alternative behavioural frameworks or describe in detail why one was chosen over another. There is a growing call to improve our understanding of cognition in human–environment interactions (Meyfroidt, 2013; Manley and Cheng, 2018) and how best to incorporate human decision-making and behaviour in models (e.g. Balke and Gilbert, 2014). Many agent-based models still take an *ad hoc* approach to implementing decisions without reference to appropriate theories (see also Section 12.2.2). Moreover, many researchers have noted that models looking at decision-making do a poor job of describing the decision-making of their agents, and once developed and published are not reused (e.g. Groeneveld et al., 2017; Bell et al., 2015).

This could be related to the fact that human behaviour is still not well understood. Most modellers are still developing their own agent-based models in their own discipline silo, there is still very little in the way of standardisation

or comparison. This is beginning to change with the appearance of frameworks such as ODD + D (Müller et al., 2013) and the Modelling Human Behaviour (Schlüter et al., 2017) framework. These frameworks attempt to provide a means for communicating and comparing different theories of individual human decision-making. This echoes other calls to publish more details on decision models to allow for reuse (e.g. Bell et al., 2015; Groeneveld et al., 2017) through such initiatives as OpenABM. Efforts are being made to develop cognitive frameworks within modelling packages also – for example, the BDI framework in MATSims (Horni et al., 2016) or GAMA (Taillandier et al., 2012) – but this is still rare and not seen in other toolkits/software. The development of standard tools for coding human behaviour into cognitive frameworks does not seem unreasonable and would improve the realism within agent-based models.

Another challenge with decision-making is how to enable agents to learn from past experiences, especially those which might impact their future decision-making (Filatova et al., 2013; Groeneveld et al., 2017). Most agent-based models do not explicitly incorporate learning or memory within their decision-making process (see Magliocca et al., 2011; Crooks and Hailegiorgis, 2014). Notable examples include Bennett and Tang's (2006) elk migration model using evolutionary algorithms, Power's (2009) citizen cooperation model, Bone et al.'s (2011) land-use change model using reinforcement learning, and Bone and Dragičević's (2010) natural resource extraction model where agents' decision-making evolves over time and is based on past experiences.

However, efforts are needed not only in describing how agents make decisions or in the creation of protocols around sharing models, but also for building good high-fidelity models of human behaviour and interactions (Weinberger, 2011). Perhaps data will shed light on this issue, specifically how new sources of data provide new ways to explore how people perceive, use and react to events in the spaces around them. Advances are being made with respect to machine learning and pattern recognition that could help to derive better patterns of human behaviour, but we also need to think carefully how such data can be described in papers.

12.2.8 Sharing and Dissemination of the Model

The last challenge identified by Crooks et al. (2008) involves how we might communicate and share agent-based models with fellow researchers and policy-makers. Traditionally models were the development of intensive and all-pervasive computation, and communicating models was mainly through discussion, simplification and visualisation, and through pedagogy in all its various forms (Batty, 1992). As we have made clear in this book, agent-based models can be overtly

visual, and through such visualisations one can convey the behaviour of the model clearly and quickly over time (Kornhauser et al., 2009). This notion is supported by North and Macal (2007, p. 280) who write that '*visualisation is one of the most effective ways to present key model information to decision-makers*'. Two- and three-dimensional visualisations of agent-based and cellular automata models are commonplace, such as the animation of spatial model results of land-use change (Tobler, 1970; Clarke et al., 2006) which allows users to see the dynamic behaviour of recognisable characteristics of model results, rather than just exploring models through data and statistics. However, further efforts still need to be made in sharing the underlying modelling processes and activities through frameworks such as ODD and UML diagrams (as discussed in Section 12.2.4).

However, if we are really to utilise agent-based models in policy decisions we need to directly involve stakeholders in the research, as Gilbert et al. (2002) note: 'it is frequently the case that policy-makers dismiss academic research as too theoretical, unrelated to the actual problems they are wrestling with, or in other ways irrelevant to their concerns'. One way to circumnavigate this issue is to explore participatory modelling or companion modelling (Barreteau et al., 2003; Etienne, 2014) which directly involves stakeholders in the modelling processes, including role-playing games to develop and validate model rules, thereby keeping the whole process transparent (Barreteau et al., 2001). While such approaches have been around for over two decades they have not been widely adopted by the modelling community, but there is increasing evidence of these approaches being used in resource management and urban planning (e.g. Etienne, 2003; Duijn et al., 2003; Le Page et al., 2015; Semboloni et al., 2004).

Another way of sharing and disseminating models is simply by taking advantage of the internet – not only in disseminating models (as discussed in Section 12.2.4) but also by allowing users to access, run and explore models in their own browsers. While running agent-based models in real time can be a challenge, especially for complex models which require substantial computational resources, for simple models one can easily use web browsers to disseminate them. For example, NetLogo provides functionality for models to be run over the web (see www.netlogoweb.org/), while Agentscript (<http://agentscript.org/>), which is loosely based on NetLogo semantics, uses JavaScript for simple agent-based models which can be deployed over the web. Examples of both are shown in Figure 12.1. There have been attempts to share and disseminate models in virtual worlds such as SecondLife (e.g. Crooks et al., 2009), while others have used video game engines such as Unreal and Crysis (see Crooks et al., 2011).

12.2.9 Data Challenges

Agent-based models are all about the individual, and, akin to all modelling techniques, large quantities of data are required to create models that can robustly test

SUMMARY AND OUTLOOK

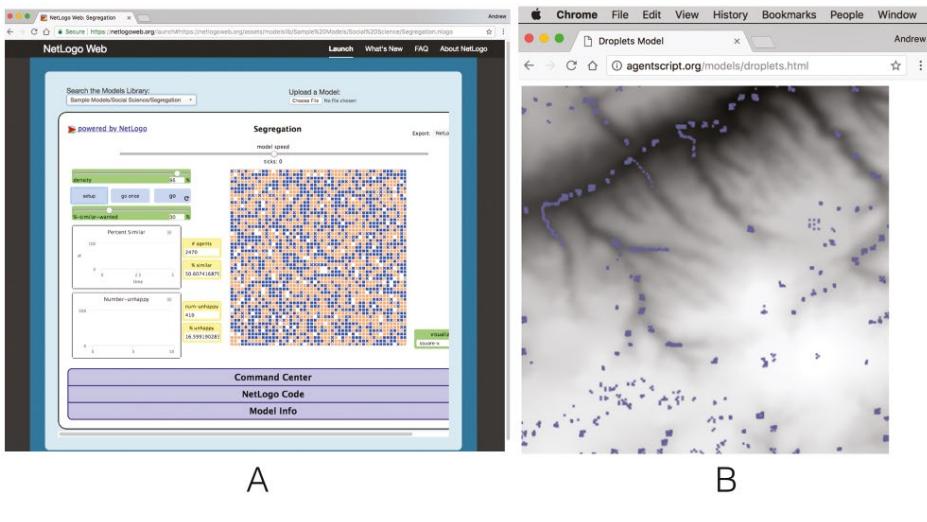


Figure 12.1 Agent-based models running in a web browser: (A) the Segregation model in NetLogo Web; (B) the Droplet model where agents flow down an elevation service in Agentscript

theories, re-create processes and dynamics and make predictions of the future. Despite the data deluge that we are now experiencing in the wake of the big-data movement, there is still a lack of high-quality, linked individual-level data. However, progress is being made here: researchers are increasingly using established approaches such as microsimulation to generate synthetic populations (see Birkin and Wu, 2012, for a review) or examine how the demographics of a population impacts agents' behaviours, an area that is referred to as agent-based computational demography (see Billari and Prskawetz, 2003). We see this as an important next step, but we also recognise the need to connect these synthetic populations to realistic social networks (Burger et al., 2017; Wise, 2014), grounded in observed data (such as average group sizes, number of connections) which can be gained from anthropology and psychology (e.g. Dunbar and Sporns, 1995) or using new sources of data (e.g. social media, mobile phones), to understand the connections and ties between people or to test if findings from previous studies in the social sciences are applicable to larger case studies. For example, Dunbar (1998) proposed that humans can maintain around 150 stable face-to-face relationships based on observations; the advent of new data means that we have the opportunity to test this observation at scale. Researchers have begun to do this; for example, Dunbar et al. (2015) found that online social networks (e.g. Facebook) had similar structures to offline face-to-face networks, while MacCarron et al. (2016) found a similar trend with mobile phone data.

Using spatial data for building the artificial worlds in which our agents operate is quite straightforward, as evidenced by the geographically explicit models

referenced or shown throughout this book. But if we are to reuse models or to build new ones, improved methods are needed for reading data into models for model initialisation. In the applications presented within the book, there is little discussion about the difficulties with and time-consuming nature of preparing data for input into the models (data cleaning, formatting, etc.), limiting such models with respect to rapid prototyping. In addition, there are multiple types of data involved. Many of the geographically explicit models will use both raster and vector data and in some instances will also require networked data (i.e. social connections) in the form of graphs (as we showed in Section 8.5). Furthermore, many papers note that one of the major limitations of their models is lack of fine-scale behavioural (and movement) data sets (e.g. Batty et al., 2003; Torrens, 2014a). For example, it is extremely difficult to get slum locations (Mahabir et al., 2018), population information and social connections between inhabitants of a city along with the current ‘mood’ of the population. Advances with open source libraries like the Python packages of Shapely (<https://pypi.python.org/pypi/Shapely>) and Geopandas (<http://geopandas.org/>) make manipulating data and formatting it for use with models easier, but there is still a significant amount of time needed to prepare data to initialise a model for a new area. Scripting such procedures helps to make it applicable to new areas along with the use of standard data formats (e.g. ESRI shapefiles). But, in general, newcomers to the field of agent-based modelling struggle to get data into models, given limited hands-on training or tutorials available (something this book begins to address – for example, in Chapter 6).

12.3 Looking Ahead

Several challenges for agent-based models have been presented in Section 12.2, along with a discussion of potential ways of overcoming these. In this section we turn to the opportunities that lie ahead for agent-based modelling and geographical systems, especially as computing power, storage capacity and data volume increase. As Torrens (2010a) notes, agent-based modelling promotes spatial thinking in the sciences and has been applied to both human and physical geographical problems² and the social sciences more generally. Such models have been fused with GIS to study a diverse range of applications, including agriculture (e.g. Deadman et al., 2004), avalanches (e.g. Kronholm and Birkeland, 2005), criminology (e.g. Malleson et al., 2010), epidemiology (e.g. Eubank et al., 2004), economics (e.g. Bert et al., 2015), geomorphology (e.g. Favis-Mortlock, 2013), housing markets (e.g. Torrens and Nara, 2007), natural hazards (e.g. Dawson et al., 2011), urban growth

² One such example was the rainfall and erosion model presented in Section 6.4.1.

(e.g. Xie et al., 2007; Xie and Fan, 2014), urban shrinkage (e.g. Haase et al., 2010), the rise of cities and regions (e.g. Pumain, 2012), slums (e.g. Augustijn-Beckers et al., 2011a) and traffic models (e.g. Horni et al., 2016). By utilising GIS we can initialise agent-based models to real world locations and provide spatial methods for relating these objects (agents) based on their proximity, intersection, adjacency or visibility to each other.

12.3.1 Big Data and Agent-Based Modelling

The use of more authoritative or traditional sources of geographical information for building (Chapter 6) and validating (Chapter 10) models has already been discussed. The majority of published applications use these data types, which is most likely down to the fact that modellers are more comfortable manipulating traditional sources of data (e.g. census data; see Robinson et al., 2007) than mining new forms of data, or see these new sources of data as too noisy, biased or inaccurate. It is clear that the rise of big data represents a significant opportunity for agent-based modelling. New forms of data provide us with new avenues with which to explore how people perceive, use and react to events in the spaces around them, and the potential to incorporate these observations into our models in near real time. Moreover, many of these sources of data allow us to examine the connections between people, organisations and space, thus offering a new perspective with which to construct artificial worlds, build environmental layers and derive behaviours that motivate agents to make certain choices and take certain actions. It is clear that big data is making, and will continue to make, a considerable impact on future agent-based modelling.

By building agent-based models with information obtained from big data, we can simulate society across many application areas in terms of who the agents are, where they are located (or where the study area is), what they are doing or what they are responding to, and why this might be the case. For example, there is a growing amount of work on exploring how crowdsourced data can be utilised to aid humanitarian relief efforts after natural disasters or disease outbreaks including the Humanitarian OpenStreetMap Team and Ushahidi (see Meier, 2015, for more information). One of the most notable and early examples was after the 2010 magnitude 7 earthquake in Haiti that killed 230,000 people and left 1.6 million people homeless. Volunteers mapped the devastation and provided a near real-time map of the current situation on the ground. Crooks and Wise (2013) utilised such information as the foundation for their agent-based model of post-disaster relief operations over a span of a week; a screenshot of the model is shown in Figure 12.2. Specifically, they explored the placement of relief centres and how the affected population might get to the aid centres based on the level of devastation seen on the ground, and agents utilised the road network for navigation which was sourced from OpenStreetMap (see Appendix A.8 for more details). As noted

in Section 6.1, spatial data acts as a basis for the artificial world (a base map, so to speak) that agents can inhabit. Crowdsourced data gives a unique insight into refugee camps allowing the exploration of diseases spreading or the outbreaks of riots to be examined at a detailed spatial scale (see Crooks and Hailegiorgis, 2014; Pires and Crooks, 2017).



Figure 12.2 A sample instantiation of the Haiti model, where the roads were initialised from OpenStreetMap data; the blue dots represent aid centres and the red dots represent agents who are seeking out food

Other researchers, such as Malleson and Birkin (2012), have started using social media to explore how people move through space, in this case the city of Leeds (United Kingdom), as shown in Figure 12.3. From mining such data they were able to derive information to classify individual behaviours and begin to develop models of that explored behaviour through space and time that can ultimately be used to model travel-to-work patterns or similar activities.

Building on such movement analysis from social media and OpenStreetMap to build artificial worlds, Wise (2014) created synthetic populations of agents and mined social media to define people's moods (sentiment) during a wildfire

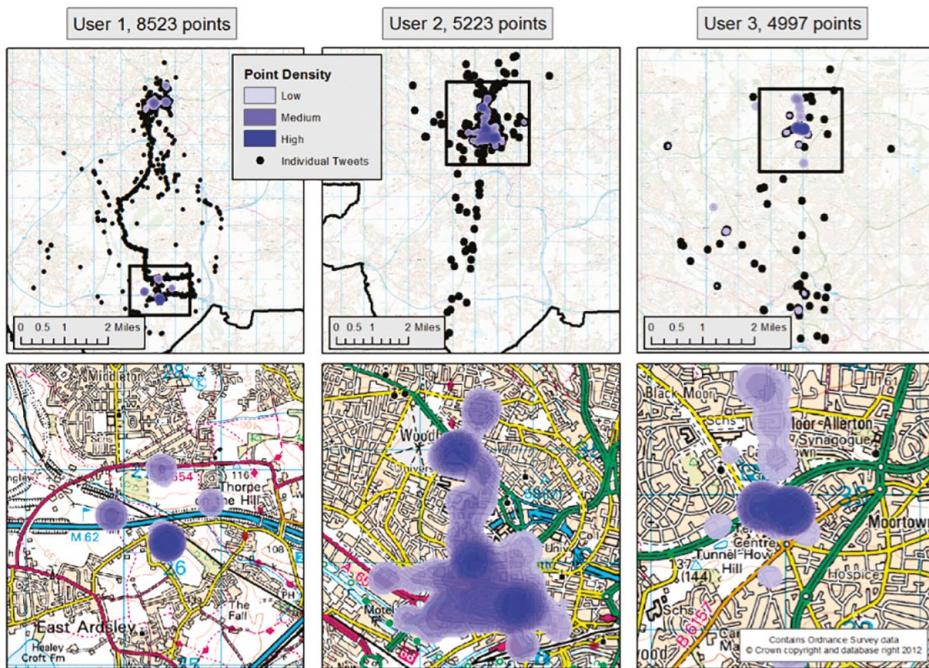


Figure 12.3 Hotspots of activity of Twitter users: tweet locations and associated densities for a selection of prolific users

event and subsequent evacuation in Waldo Canyon, Colorado Springs, in 2012. Using tweets harvested during the event, Wise (2014) first manually classified a number of tweets as of positive, negative or neutral sentiment with respect to the wildfire, with some representative examples being shown in Table 12.1. Once this training data set was created, the remainder of the tweets were analysed to derive the sentiment of all the remaining tweets during the event using the AFINN (Nielsen, 2011) lexicon (dictionary). The results of this are shown in Figure 12.4, where the peak of Twitter activity corresponds to the peak of the wildfire. This social media data set was directly used to inform agent-based decision-making. For example, if one of the agents (i.e. a Colorado Springs resident) knew that the fire was nearby, this information was passed along their social network to other agents who then decided whether to evacuate or not. The movement (evacuation) patterns were then validated using congestion data that was again harvested from the crowd (in this case images) and news reports.

The examples above offer a snapshot of the potential of using big data for building agent-based models focusing on *who* (the agents), *where* (in space and time), *what* (phenomena being modelled) and *why* (agents make decisions the

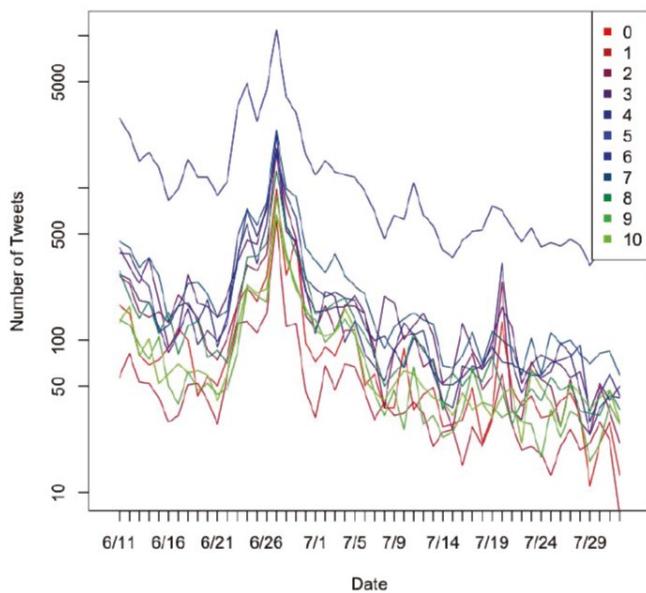


Figure 12.4 Tweet sentiments between 16 June and 12 July for the Waldo Canyon wildfire, log-scaled to give a sense of the change of rank (Wise, 2014)

way they do). However, these applications focus on relatively short time frames (from hours to weeks), while traditionally models have tended to focus on years and decades. Here agents are constantly interacting with each other and their environment at finer spatial and temporal scales than more traditional models. It is in this sense that agent-based modelling is focusing on understanding the bottom-up mechanisms that make use of the physical and social infrastructure to drive the complex systems. To some extent big data, especially that coming from social media, is also generated through a bottom-up approach and offers a valuable means of exploring cities. The data is dynamic, immediate and relates to how people interact in space and time.

12.3.2 Model Integration

While there has been a proliferation of agent-based models, many agent-based models tend to look at only one aspect of a geographical system, as shown in Figure 12.5. Such models can range from the small-scale movement of pedestrians to the spread of diseases or the growth of slums. One reason for this piecemeal approach to looking at geographical systems is limited computational power and the availability of fine-scale data on which to base individual behaviours. This is now changing. In the past, geographically explicit models were limited to small

SUMMARY AND OUTLOOK

Table 12.1 Comparison of sentiment classifications of real tweets

		Negative
0		And it gets more bizarre: Fake #highParkFire fire-fighter pulled a similar scam on #LowerNorthForkFire: http://t.co/MV64BdKV
1		@AKbirder I have my stuff ready. Very uneasy with the current situation w/fire to the north. Most pics are on hard rives. Ready to go.
		Neutral
5	puppy is doin much better =)	
5	Whew. Dropping off the Youngest at camp was fraught with drama! Highway closed behind me, due to forest fire on Hwy50. #Colorado	
		Positive
7	Fire Fighters Rock - Thank You !! http://t.co/Kiqr5EUP	
10	Dear Universe, Hi, hope everything works out & we don't die horribly in fire.. Thanks.. Sincerely, ME	

Source: Wise (2014).

numbers of agents (due to computational and data constraints), but with the growth in computational power and data availability one can simulate a million or more agents. But if we are to address larger societal issues (e.g. climate change, urban change) these individual models will need to be integrated as each just looks at one part of the puzzle.

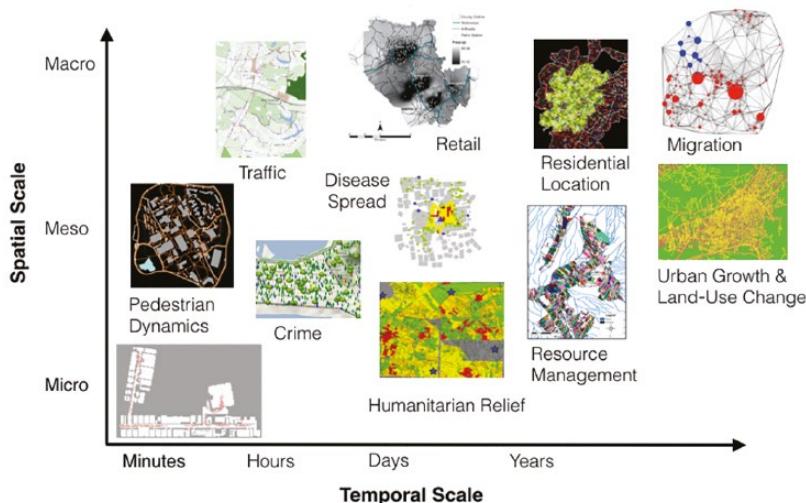


Figure 12.5 A sample of application domains for agent-based models for geographical systems

However, to date there are no geographically explicit agent-based models that simulate a city in its entirety. The challenge is that many models focus on just one aspect (or subsystem) of city life (e.g. travel to work, residential location, the spread of disease, the economy) and treat other urban processes or subsystems as exogenous variables, ignoring the fact that all processes within cities are intricately linked. For example, transportation impacts residential locational decisions or how one navigates around cities. While this piecemeal approach to examining cities is useful, to truly understand cities we need to view them as ‘systems of systems’. Returning to the notion of complexity (as discussed in Chapter 1), we can view cities as hierarchical and composed of interrelated subsystems (parts within parts) in which each subsystem is interdependent but connected to many other subsystems. Such subsystems may plausibly be thought of as self-organising. In economics, for example, national and global markets evolve from locally interacting agents all pursuing their own goals. At a city level, one could consider town centres or sub-centres as system structures, as illustrated in Figure 12.6A. If we look at the connections between these elements we have a hierarchy, as shown in Figure 12.6B. This argument echoes those of near-decomposability (Simon, 1996): here a system (the city) has subsystem components interacting among themselves ‘in clusters or subgraphs, and interactions among subsystems being relatively weaker or fewer but not negligible’ (Cioffi-Revilla, 2014). It is not just the town centres and other elements that are connected but also urban processes. A simple illustrative example of this is shown in Figure 12.6C.

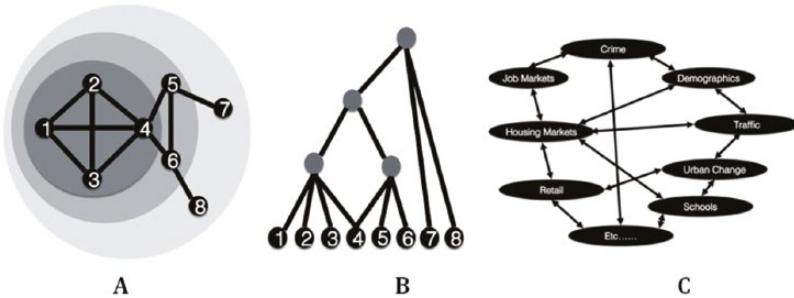


Figure 12.6 (A) System structure; (B) system hierarchy; and (C) related subsystems/processes for urban systems (adapted from Batty, 2013)

This systems approach to understanding cities is not new (see Batty, 2013). For example, Christaller’s (1933) central place theory noted the hierarchical structures of villages, towns and cities. Simon (1996) argues that hierarchy is a fundamental property of how a complex system holds itself together, while Batty (2013) notes that ‘hierarchical organisation from the bottom-up is essential for evolving systems

and that hierarchical structures are the way nature and society develop robust and resilient structures' (Batty, 2013). However, these subsystems do not operate in isolation. In the short term, they might appear to be independent of the rest of the larger system, but in the long run they are indeed dependent on the aggregate system behaviour. For example, most traffic models just explore traffic, while residential location models do not take into account the impact of traffic congestion on residential decisions (Wise et al., 2017).

To model a city or selected subsystems we need to combine/integrate smaller models. Perhaps the best approach is through model integration and coupling. The coupling of various models has been common in more traditional modelling domains, especially in land-use–transport interaction (LUTI) models whose purpose tends to be to test new transportation schemes and/or new land-use developments and their impacts. One example of a LUTI model is the integrated assessment model by Walsh et al. (2011), designed to explore urban sustainability in the context of climate change. This framework couples various models together (e.g. a regional economic model, an employment, population and land-use model, a flood model) to explore climate impacts, adaptation options and greenhouse gas emissions. Such coupling of agent-based models with others is still in its infancy but is being discussed (e.g. Taylor et al., 2004; Kettler and Lautenschlager, 2017), and we would expect this hybrid form of modelling to continue in the future, mixing and matching the best from many different simulation frameworks (styles of models).

The integration of different modelling styles (e.g. microsimulation, system dynamics), as opposed to straightforward coupling, should be considered as the choice of modelling approach depends on the purpose and needs (Martinez-Moyano and Macal, 2016) of the research. For example, agent-based models are good for modelling problems that lack central coordination (Macy and Willer, 2002) and where emergence is key. If emergence is not central to the problem, other approaches might be more suitable – for example, loosely coupling agent-based models of evacuation with flood-risk models to understand and predict the impact of such an event (e.g. Dawson et al., 2011; Wang et al., 2016), or combining a computational fluid dynamics model of a contaminant plume and an agent-based model of movement to explore evacuation design and exposure risks (Epstein et al., 2011). MATSims has been linked to a discrete choice model of residential location (Ziemke et al., 2016), while Park et al. (2017) coupled an agent-based infection model to the Global Change Assessment Model to explore corn production. North et al. (2015) combined a system dynamics energy security model to an agent-based conflict model to explore how changes in oil prices could impact a fragile nation. Finally, Bert et al. (2015) integrated a land rental market with an agent-based agricultural model to explore land-use change in Argentina.

Note that we are not the first to consider this issue (see, for example, Heppenstall et al., 2012a). O’Sullivan et al. (2015) also notes that agent-based models should be integrated as many models can capture complicated patterns and processes, or different modelling styles should at least be compared in the same research domain (as we did in Chapter 11). However, if we are to merge models we need to consider what are the appropriate spatial (e.g. centimetres, metres, kilometres) and temporal scales (e.g. seconds, hours, years) and behavioural process of the actors of each of these subsystems. This poses a significant challenge with respect to capturing behavioural process over such diverse spatial and temporal ranges. Any urban model or model component must account for these dynamic characteristics in time, space and behaviour. We should also be cautious with model integration; the first generation of urban models attempted to capture too many details and were criticised for being highly complicated and too data-intensive, ironically at a time when data and computational power were not available (Lee, 1973).

One major stumbling block with model integration is that, despite advances in computing power, the integration of numerous large, complicated models will still be extremely difficult to run in a reasonable amount of time. There are methods for scaling agent-based models such as distributed computing (e.g. DMASON; Cordasco et al., 2013) or using high-performance computing (e.g. Repast HPC; Collier and North, 2013), but these are still very much under development. There are a growing number of companies specialising in scaling agent-based models such as Improbable whose SpatialOS cloud platform³ could offer solutions to this issue.

For model integration to become a standard part of the agent-based modelling community, more formal protocols are required for model communication. Some progress is being made in this area; for example, NetLogo has an extension called LevelSpace (Hjorth et al., 2015) which allows one to run an arbitrary number of concurrent NetLogo models simultaneously which can communicate with each other. For example, a model of human population growth could be tied to a model of food production which in turn could be linked to a climate model. Each one of these models can potentially influence the other; for example, as the population grows there is more demand for food, and the crop model might need a certain temperature or amount of rainfall, which could be input from the weather model.

We see linking various subsystems as a rich vein of future work. The future of this area will be characterised by big data powering complex agent-based models for large-scale simulations. Moreover, it is only through model integration that we might be able to provide potential solutions for grand societal challenges such as urban growth, climate change, security and sustainability.

³ For more information, see <https://improbable.io/>

12.3.3 Uncertainty and Ensembles

As the agent-based modelling method matures, there are opportunities to begin to adapt useful methods from longer-established fields in order to improve the rigour of agent-based modelling. This is especially true with respect to how agent-based models deal with *uncertainty*. There are a number of reasons why uncertainty can make its way into model results. It might be a result of noise in the input data that is used to parameterise the model or because the model rules themselves are poorly specified (the model is not adequately representing the system that it is designed to represent). Other fields, particularly the environmental sciences such as meteorology and hydrology, have decades of experience in developing methods to quantify and manage uncertainty.

One of these is *ensemble modelling*. An ‘ensemble’ is a group of models that are run simultaneously. The models are probabilistic, so naturally begin to diverge during the course of a simulation. By analysing the range of outcomes across an ensemble of models, it is possible to begin to better understand how uncertain the outputs are. Where most models are broadly in agreement in their results, then there is less uncertainty compared to the situation where the models diverge substantially. Figure 12.7 illustrates the results of two ensembles of numerical weather models.

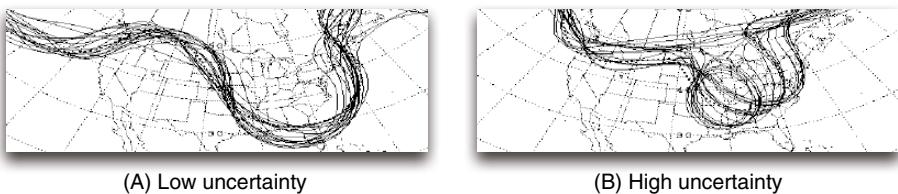


Figure 12.7 A ‘spaghetti plot’ illustrating pressure contours as forecast from numerous models in an ensemble. In (A), the results across different models are similar and hence the forecast has relatively low uncertainty. In (B), however, the results vary substantially in some areas so the forecast is less certain. Adapted from Kalnay (2003)

More rigorous treatment of uncertainty also has the potential to improve the trust that others (including policy-makers) can have in the results of an agent-based model. By presenting results from models alongside their uncertainty, modellers have an opportunity to more rigorously represent the reliability of their predictions. People are generally appreciative of the uncertainty associated with, for example, weather forecasts (“there is a 40% chance of rain today”) so will probably be equally comfortable with the uncertainty generated by an ensemble of agent-based models.

12.3.4 Data Assimilation

Agent-based models are often used to simulate the behaviour of *complex* systems. These systems often diverge rapidly from some initial starting conditions. For example, consider the flocking of birds. The behaviour of the individual birds might be very well understood, and there might be very high-quality information on the precise locations and velocities of all birds in a flock. Nevertheless, a model that attempted to forecast the behaviour of the flock over time would inevitably diverge as small errors in the initial starting conditions lead to very different outcomes over the longer term. One way to prevent a simulation from diverging from reality would be to occasionally incorporate more up-to-date data and adjust the model accordingly. For example, after a few minutes of simulating the bird flock, a new image of the real flock could be taken, with new precise locations of the birds, and this new information could be incorporated into the model to bring it more in to line with reality.

There are a range of techniques that come under the banner of *data assimilation* that are designed for exactly this purpose. However, they have largely evolved from fields such as meteorology (i.e. to incorporate up-to-date environmental data into weather forecasts), and it is not clear whether they are appropriate for use in agent-based modelling. Some have begun to explore this area (Rai and Hu, 2013; Ward et al., 2016) but only with the simplest agent-based models. The marriage of data assimilation methods and agent-based models could be transformative for the ways that some systems (e.g. ‘smart’ cities) are modelled.

12.3.5 Spatially Learning Agents

We know from a range of studies from diverse academic disciplines that the way humans recall, describe and navigate through geographic space is complex. Through studies in behavioural geography, psychology and neuroscience, we have a reasonable understanding that humans remember and recall spaces with bias, error and skew, and are bounded in their ability to navigate through space by their prior experiences. These limitations are due to the way our brains have evolved, with specific brain cells that suggest an evolved spatial learning process based on landmarks (*place cells*), direction (*head direction cells*), regions (*border cells*) and spatial association (*grid cells*).

Despite these findings, our modelling of human behaviour in space tends to assume that agents have ‘perfect’ spatial knowledge and are able to optimise their choices in moving around it. Furthermore, in building on GIS data, models build in an implicit Euclidean representation, which translates into the way agents are modelled to make decisions. Only a handful of simulation models have sought to integrate aspects of spatial cognition and learning (Manley et al., 2015b; Manley and Cheng, 2018).

A promising route forward could be letting the agents learn their environment for themselves. Through methods such as *reinforcement learning*, where positive behaviours are ‘learned’ through a repeated exposure to an environment, deep neural networks, which capture uncertainty and incomplete knowledge representations, and large-scale individual tracking data, it may be possible to teach agents how to navigate spaces as if they were human. These ‘learning’ agents may both better reflect the actual behaviours of humans, and model their behaviour under changing conditions. Progress is rapidly being made elsewhere (e.g. Banino et al., 2018), but integration into geographical modelling remains a challenge.

12.4 Discussion

Understanding the complexity of the geographical systems around us is a fundamentally important challenge. Through a deeper grasp of how our world works, and an improved ability to predict future changes, we can seek to make improvements or militate against threats. As we have shown throughout this book, agent-based modelling is a highly promising methodology for tackling geographical complexity, offering us a framework for analysing the deep systemic interactions that take place within and between socio-technical actors and the wider environment. And yet, in this statement we reveal the ongoing limitations of the methodology. Agent-based modelling remains a relatively immature discipline, and during our discussion we have revealed a vast variety of approaches, languages, theories and methods all in wide use. As such, our verdict of ‘highly promising’ reflects the immaturity of the field, and the need to do much more to fully achieve its potential.

This does not mean a negative outlook for agent-based modelling. The models we have showcased in this book demonstrate a highly active and diverse field, enjoying wide success in a variety of contexts. The increasing availability of data, growing computational resources, deeper multidisciplinary thinking and the simple intuition of the approach all mark agent-based modelling as an approach with a rosy future (see Waldrop, 2017, 2018). Nevertheless, to achieve full maturity and credibility, agent-based modelling for geographical applications requires focus in a few areas.

Perhaps the most pressing need is a deeper focus on scientific method – specifically, replication, validation and progression. As highlighted above, despite the variety of agent-based modelling work being carried out, the field lacks a cohesive direction of progress. New (excellent) ideas are implemented into new (excellent) models, but fail to properly integrate the beneficial work that has gone before. New models do not necessarily need to extend previous modelling work, but some alignment over theory, behavioural framework or development methodology would enable improved interpretation and recognition of progress. This will

not occur in the dark, and modellers must be open with their creations, sharing methods and code openly, allowing and assisting others to investigate and build on what they have done. Replication of results should be encouraged, and integrated as an important element of model development. While the availability of new data means researchers will be drawn to new ideas and approaches, control over how these models are integrated into a wider understanding is needed.

Agent-based modellers should also be braver about presenting this approach as an alternative to other methodologies. Often the intuitive simplicity and methodological diversity of the approach means it is seen less favourably than more mathematical approaches (e.g. regression modelling). But agent-based modelling enables one to capture greater diversity of behaviour than any other method, and lends itself very suitably to integration with new forms of data. However, modellers must be disciplined in how they verify and validate their models, adhering to the now well-established set of methods for testing a model. Agent-based modellers must also talk with those from other modelling disciplines, and be open about the strengths and weaknesses of the approach and routes towards achieving progress, as we are aiming to do here. We should also engage more deeply with the public and with policy-makers, and isolate and develop examples of where agent-based modelling is contributing positively to public life (beyond the conventional examples of pedestrian modelling and movie special effects).

There is an exciting future ahead for agent-based modelling. However, all of us, as developers of agent-based models, must work thoughtfully to deliver the promise it currently offers. We look forward to you joining us in this effort over the coming years.

Chapter Summary

This chapter has summarised some of the remaining challenges and opportunities for agent-based modelling. While the discipline has seen considerable progress over the last two decades, greater focus is needed on model replication, transparency and integration. New forms of data represent a considerable source of promise for agent-based modelling, which is well placed to deal with and integrate this data into the methodology.

12.5 Annotated Bibliography

Readers wishing to know more about running multiple models together in NetLogo are referred to:

- <http://ccl.northwestern.edu/rp/levelscape/>

Readers wishing to know more about companion modelling are referred to:

- <https://www.commod.org/en>

For a more detailed discussion of some of the issues brought up in this chapter and the key challenges in agent-based modelling, see:

- Crooks, A.T., Castle, C.J.E. and Batty, M. (2008) Key challenges in agent-based modelling for geo-spatial simulation. *Computers, Environment and Urban Systems*, 32(6), 417–430.
- Heppenstall, A.J., Crooks, A.T., See, L.M. and Batty, M. (2012) Reflections and conclusions: Geographical models to address grand challenges. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 739–748. Dordrecht: Springer.

For more about big data and geographical information or urban applications, see:

- Singleton, A.D., Spielman, S. and Folch, D. (2017) *Urban Analytics*. London: Sage.

Lastly, readers interested in how cities can be viewed as systems of networks and flows through the lens of complexity should see:

- Batty, M. (2013) *The New Science of Cities*. Cambridge, MA: MIT Press.
- Batty, M. (2018) *Inventing Future Cities*. Cambridge, MA: MIT Press.

APPENDICES

A

A GALLERY OF APPLICATIONS

Chapter Outline

While the entirety of this book has given readers a selection of models (mainly in NetLogo) with respect to agent-based modelling more generally and linking such models to real world geographical information, we thought it would be useful for readers to have exposure to agent-based models developed not only with NetLogo but also other toolkits. In this appendix we provide a gallery of applications from a broad section of fields that have been published in the literature or act as exemplars for getting started with agent-based modelling. The criterion for inclusion here is that the source code and data of the model are available and the model is based on real world geographical information. In each case we provide a screenshot of the graphical user interface of the model, a short description of the model, its full citation where possible and where the model (and data) can be downloaded from. Our hope is that the models provide readers with exposure to the possibilities of agent-based models and their potential for analysing a wide array of geographical systems. We also hope to encourage readers to share their own models and data.

A.1 Disease Dynamics in a Refugee Camp

Crooks and Hailegiorgis (2014) developed a cholera disease model implemented in MASON, based on the Dadaab refugee camp in Kenya which is home to approximately 250,000 individuals (see Figure A.1). Poor sanitation and housing conditions within the camp contribute to frequent incidents of cholera outbreaks. The intention of this model is to explore the spread of cholera at the individual level. The progress of cholera transmission is represented through a susceptible–exposed–infected–recovered model which integrates geographical data with agents’ daily activities within a refugee camp. Results from such a model show how cholera infections are impacted by agents’ movements and sources of contamination. For example, one can ask ‘what if’ questions, such as what if a borehole is infected with cholera bacteria – how will it spread over space and time?.

Model available at: <http://css.gmu.edu/Cholera/>



Figure A.1 Graphical user interface of the model. From top left clockwise: legend, simulation display, parameter settings and a clock recording the time dwellers, agents' activities, household size composition, parameter settings and a clock recording the time

A.2 Hiker Movements in the Dolomites UNESCO World Heritage Site

Orsi and Geneletti (2016) developed a model in NetLogo to explore how different transportation modes (e.g. buses and private vehicles) impact visitation patterns in the Dolomites UNESCO World Heritage Site (Italy) as shown in Figure A.2. By modelling individual agents (i.e. visitors), one can see how crowding levels build up along trails (i.e. hiking routes) over the course of a day. By understanding such activities, the model provides a tool for managers to explore or test strategies which preserve natural resources and the quality of recreational experience.

Model available at: <http://dx.doi.org/10.1016/j.compenvurb.2016.07.008>

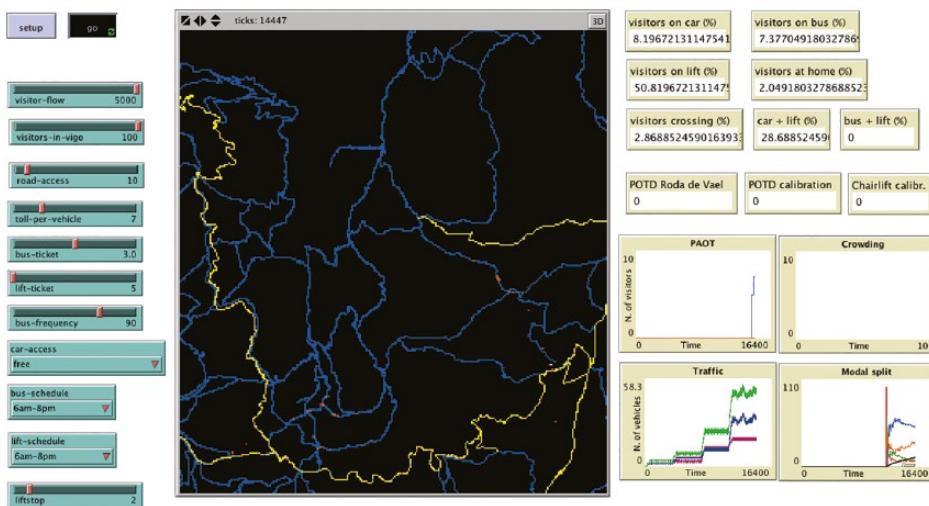


Figure A.2 Graphical user interface of the model. From left to right: input parameters, spatial environment and output statistics such as the amount of traffic and modal split

A.3 Modelling the Emergence of Riots

After the 2007 Kenyan election results were announced, riots erupted in the Kibera slum of Nairobi. Pires and Crooks (2017) created a model in MASON to capture the complex, nonlinear nature of riots as shown in Figure A.3. By using GIS and social network analysis coupled with an agent-based cognitive framework grounded in theory, they were able to model the emergence of riots. Results from the model showed how the youth were most susceptible to rioting, and how increasing education and employment opportunities for the inhabitants of the slum had nonlinear effects on rioting.

Model available at: <https://www.openabm.org/model/4865>



Figure A.3 Graphical user interface of the riot model based on the Kibera slum of Nairobi. From top left clockwise: spatial environment, input parameters, legend and summary of residents' activities.

A.4 The Foothill Yellow-Legged Frog Assessment Model

The Foothill Yellow-legged Frog Assessment Model (FYFAM) developed by Railsback et al. (2015) explores how water temperature and flows affect the breeding success of the foothill yellow-legged frog (*Rana boylii*) in Californian rivers and streams (see Figure A.4). Unlike other frog species, the foothill yellow-legged frog breeds in rivers and streams and not still water, thus eggs and tadpoles are vulnerable to changes in water conditions which are impacted by upstream reservoirs. The model provides an environment to explore the reproductive success of the foothill yellow-legged frog under various amounts of water flow release policies from a dam.

Model available at: <http://www2.humboldt.edu/ecomodel/FYFAM.htm>

A.5 Understanding Artificial Anasazi

While many of the examples in this book focus on the current day, one area of active research for agent-based modelling is in archaeology. Perhaps one of the

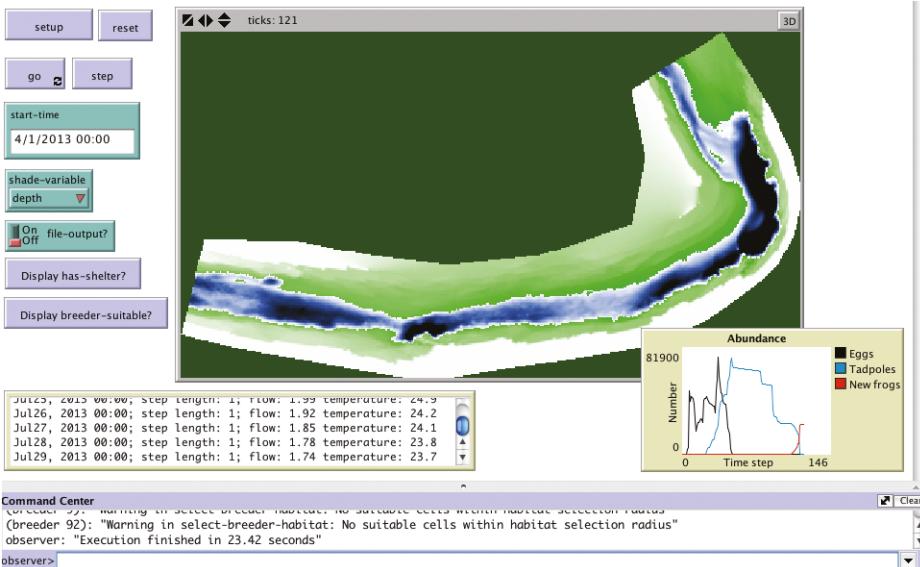


Figure A.4 Graphical user interface of the FYFAM based on South Fork Trinity River site in California

most iconic agent-based models of past societies is the Artificial Anasazi model by Axtell et al. (2002). In this model, the population dynamics of Kayenta Anasazi in Long House Valley, Arizona, between AD 800 and 1350 was simulated. Axtell et al. (2002) were able to closely capture changes in the population along with changing settlement patterns over time which match the archaeological record. While the links to the original model are no longer active, several replications of the model have been attempted. These include one in Ascape (<http://ascape.sourceforge.net/>), a Java-based agent-based modelling toolkit created by Miles Parker, who was also one of the original programmers of the Artificial Anasazi model. However, the one we show in Figure A.5 replicates the model via NetLogo and is from Janssen (2009).

Model available at: <https://www.openabm.org/model/2222/>

A.6 The Spread of Agriculture during the Neolithic Period

Gordó et al. (2015) developed a spatially explicit cellular automaton model in NetLogo to explore the spread of agriculture in the Iberian peninsula during the Neolithic period. Specifically, they classified the landscape (i.e. the cells) based on its suitability for agriculture at a 5×5 km resolution as shown in Figure A.6. They then tested different models of dispersal (e.g. agriculture spreads from one cell to

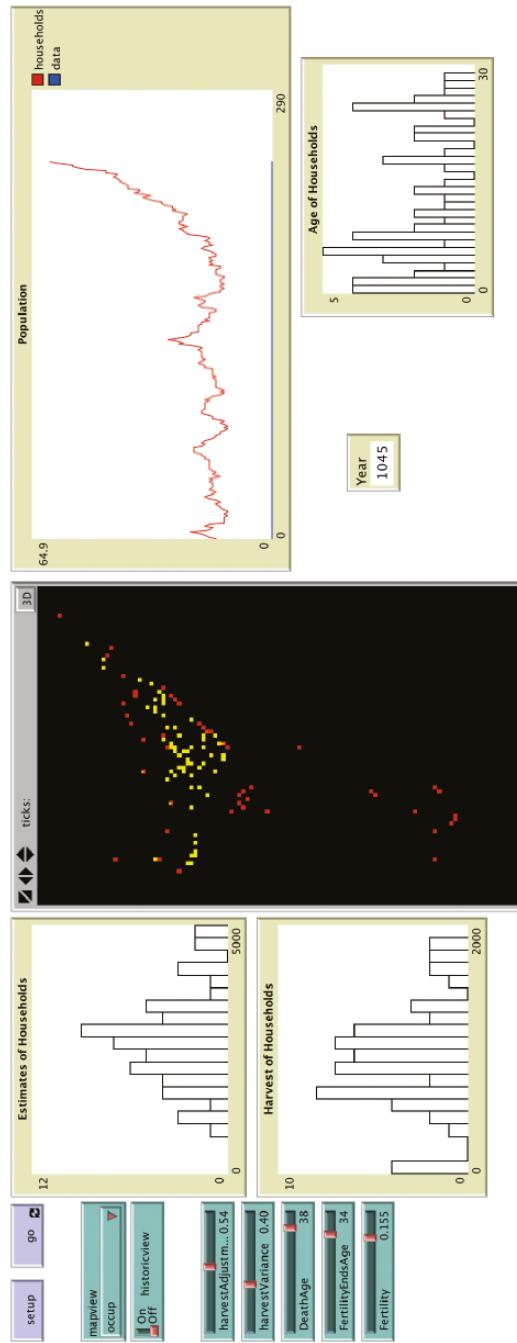


Figure A.5 Graphical user interface of the Understanding Artificial Anasazi model

its neighbour as long as the conditions are suitable for agriculture). By doing so the researchers were able to explore how different starting points for agriculture correlated to actual archaeological records.

Model available at: <https://www.openabm.org/model/4447>

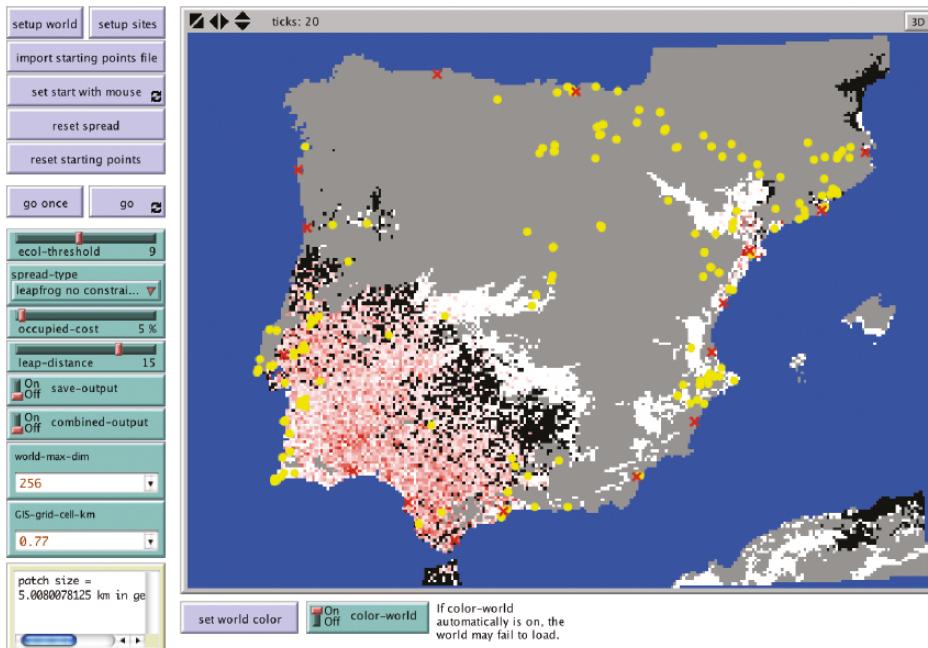


Figure A.6 Exploring the origins of agriculture in the Iberian peninsula during the Neolithic period

A.7 WalkThisWay

Pedestrian modelling has traditionally been faced with the challenge of collecting data to calibrate and validate such models of pedestrian movement. With the increased availability of mobility data sets from video surveillance we are now presented with the opportunity to change the way we build pedestrian models. Using MASON, Crooks et al. (2015a) demonstrated how real scene activity information can be harvested and used in agent-based models (see Figure A.7) to better inform pedestrian models and enhance their predictive capabilities.

Model available at: <https://www.openabm.org/model/4706/>

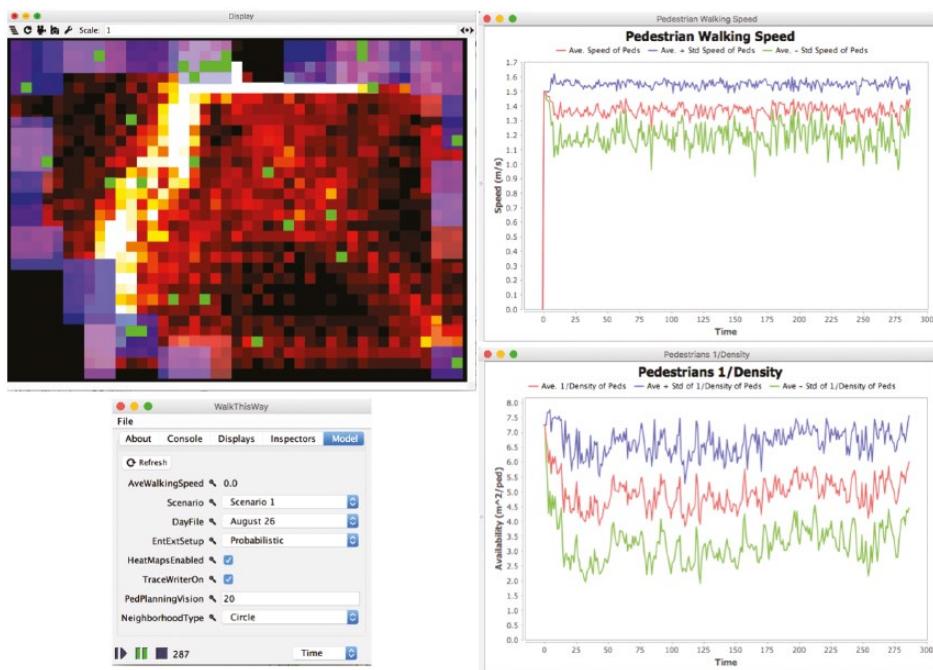


Figure A.7 Graphical user interface of the WalkThisWay pedestrian model. Clockwise from top left: spatial environment, charts recording average walking speed, crowd density and model parameters

A.8 Natural Disasters and Humanitarian Relief

Natural disasters such as earthquakes and tsunamis occur all over the world, altering the physical landscape and often severely disrupting people's daily lives. Recently researchers' attention has focused on using crowds of volunteers to help map the damaged infrastructure and devastation caused by natural disasters.

While such data is extremely useful, as it allows us to assess damage and thus aid the distribution of relief, it tells us little about how the people in such areas will react to the devastation. In the model developed by Crooks and Wise (2013) using MASON and data from several different sources (e.g. OpenStreetMap), they demonstrate how crowdsourced geographic information can be used to study the aftermath of a catastrophic event (Figure A.8). The specific case modelled here is the Haiti earthquake of January 2010. The model demonstrates the importance of exploring of human–environment interactions and the positioning of aid centres after a natural disaster and how different aid centre configurations impact the health of the surviving population.

Model available at: <http://www.css.gmu.edu/haiti/>

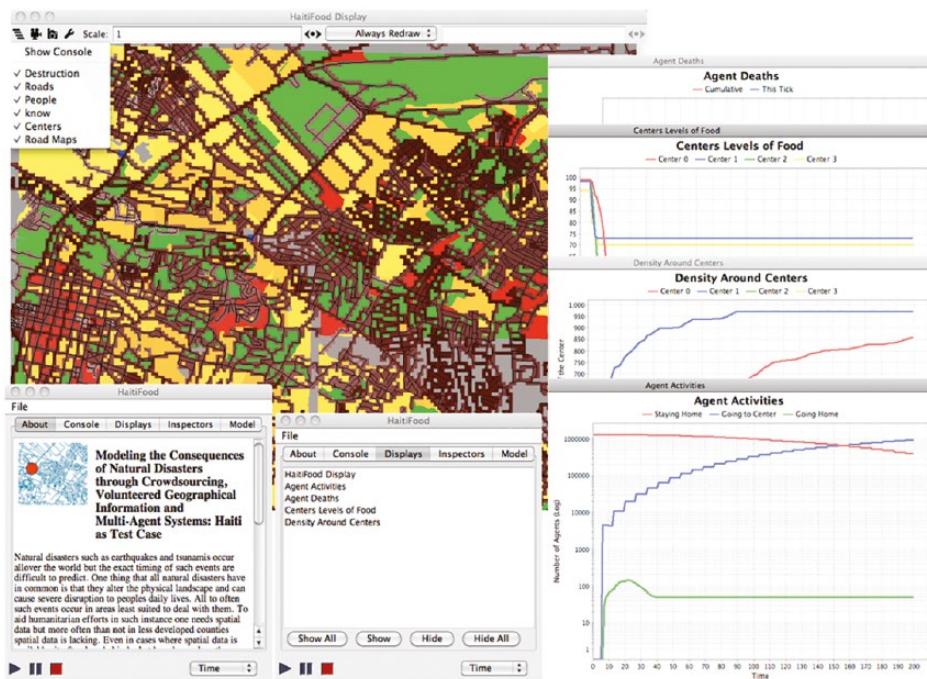


Figure A.8 Graphical user interface of the humanitarian relief model. Clockwise from top left: the spatial environment and its various layers; charts recording agent deaths, food levels and densities at various aid centres and agent activities; model information

A.9 Using Social Media Content to Inform Agent-Based Models for Humanitarian Crisis Response

The Hotspots model (Wise, 2014) was created using MASON and utilises open data to generate a synthetic population with realistically distributed social ties to one another. This synthetic population is then used to populate a model of households being interrupted in their daily activities by the presence of a developing threat. Households make decisions about the wildfire that threatens them based on their own observations, public announcements through conventional media, and communications from their socially linked peers. Traffic flow modelling reflects the emergent result of those individual decisions to stay or evacuate, capturing the location and timing of traffic jams in the emergency situation. Thus, the tool allows researchers to better explore the way interventions might cascade across the environment and translate into different evacuation outcomes.

Model available at: <https://github.com/swise5/Hotspots>

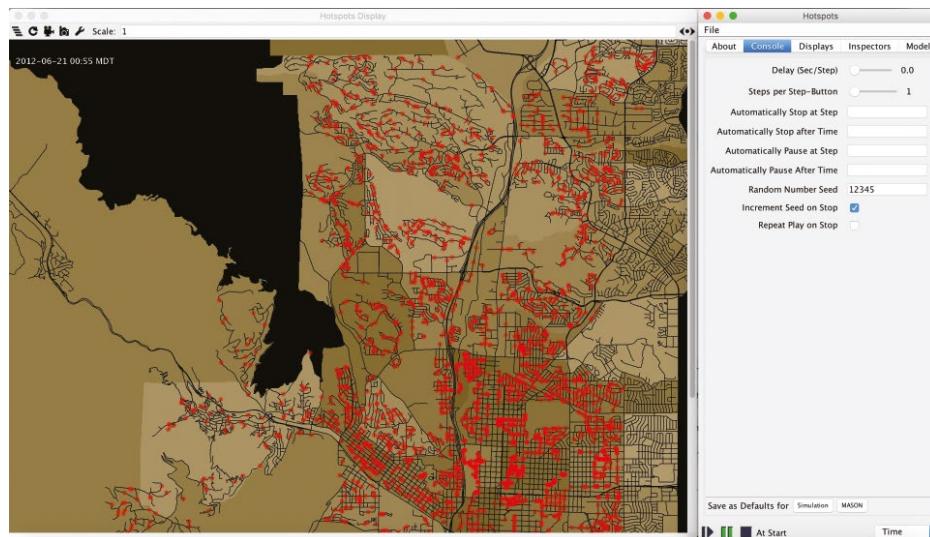


Figure A.9 Graphical user interface of the Hotspots model showing the spatial environment; red dots denote agents

A.10 Modelling Transportation and Development for Reston, VA

Swartz (2015) developed a spatially explicit agent-based model in NetLogo to explore how altering the mode of transportation impacts individual agents' commuting behaviour in Reston, VA, a suburb of Washington, DC (Figure A.10). The model generates typical traffic conditions for the area and explores the potential impact on residential development on travel times as more residents move into the area, including identifying bottlenecks at major intersections from such growth.

Model available at: www.abmgis.org

A.11 Agent-Based Modelling for Community Resource Management

Wise and Crooks (2012) developed an agent-based model in MASON to explore the complex social interactions of water management, which involves land owners (i.e. farmers) collectively maintaining and managing ditches which distribute water among the properties in Taos, New Mexico. This system of the physical ditches and the maintaining organisation together is known as an *acequia*, and the landowners who maintain it are called *parciantes*. Acequias are interesting to researchers because of the developed common property regimes they require to function. The water carried by the ditches is a shared resource, and the complex management system of the acequia has evolved to avoid a

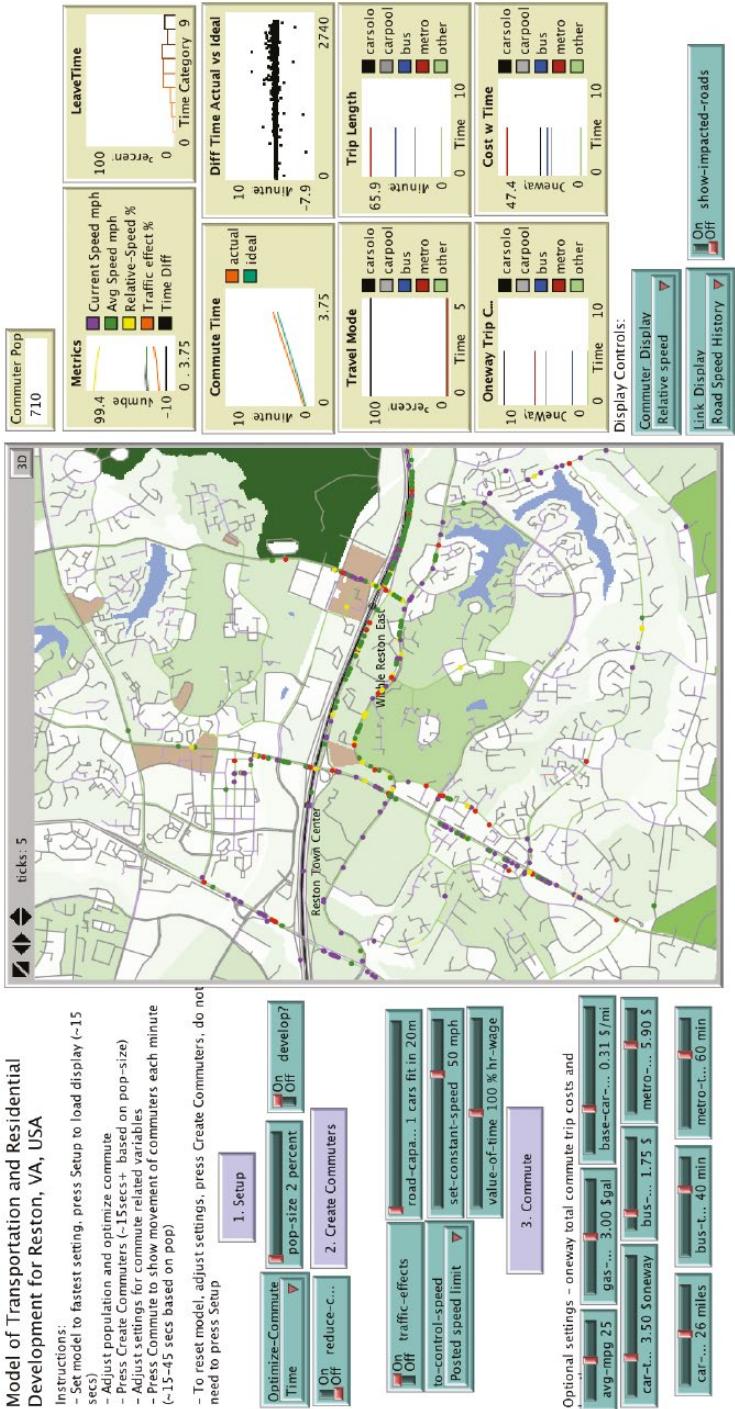


Figure A.10 Graphical user interface of the Reston commuter model. From left to right: input parameters, visual display and model outputs during a representative model run

‘tragedy of the commons’ with regard to natural resources in the sense that it prevents the resource from being overused or under-maintained to the detriment of everyone. Wise and Crooks (2012) investigate the interactions and cumulative impact of the physical water system, local social and institutional structures which regulate water use, and the real estate market on the sustainability of traditional farming in the region. A snapshot of the GUI of the model during a specific simulation is shown in Figure A.11. The results from the model show that, depending on the future patterns of weather and government regulations, acequia-based farming may continue at near current rates, shrink significantly but continue to exist, or disappear altogether.

Model available at: <http://css1.gmu.edu/acequia/>

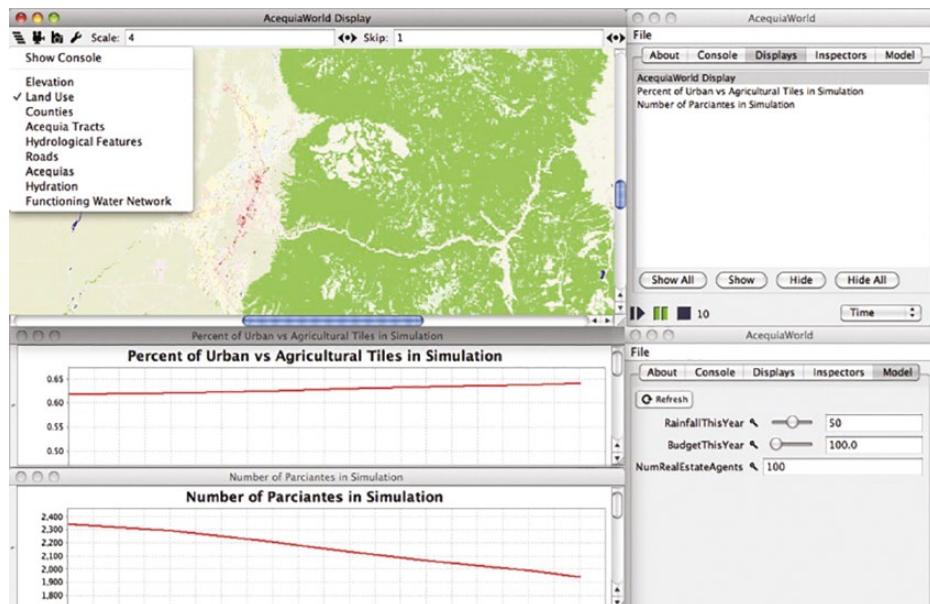


Figure A.11 Graphical user interface of the acequia-based agriculture model showing the spatial environment, including land use, and charts recording the number of farmers (parciantes) and urban-versus-agricultural percentages during a representative model run

A.12 Agent-Based Modelling of Conflict Diamonds

In the early 1990s, Sierra Leone entered into nearly 10 years of civil war. The ease of access to the country’s diamonds is said to have provided the funding needed to sustain the insurgency over the years. According to Le Billon (2001), the spatial dispersion of a resource is a major defining feature of a war. Pires and Crooks (2016) developed an agent-based model in MASON using geographical information to

create a realistic landscape and theory to ground agent behaviour to explore Le Billon's (2001) claim. The GUI of the model is shown in Figure A.12. Different scenarios were explored as the diamond mines were made secure and the mining areas were moved from rural areas to the capital. They found that unexpected consequences can come from minimally increasing security when the mining sites are in rural regions, potentially displacing conflict rather than removing it. On the other hand, minimal security may be sufficient to prevent conflict when resources are found in the city.

Model available at: <https://www.openabm.org/model/4955/>

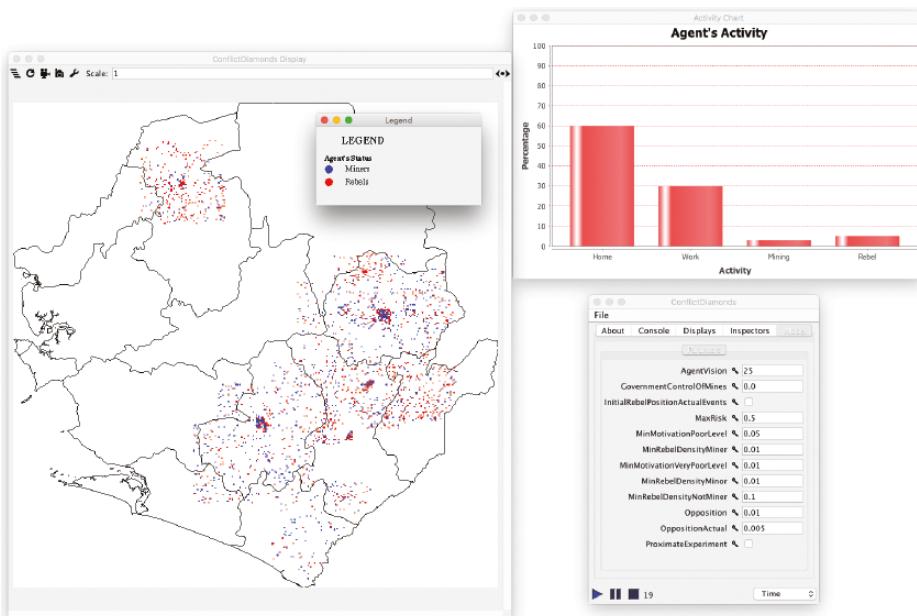


Figure A.12 Graphical user interface of the Geography of Conflict Diamonds model showing the spatial environment centred on Sierra Leone and charts recording agent activities (e.g. working, mining, rebelling) and the model parameters during a representative model run

A.13 Exploring the Growth of Slums

More than 900 million people – one-third of the world's urban population – live in either slum or squatter settlements (UN-Habitat, 2003). Urbanisation rates in developing countries are often so rapid that formal housing development cannot meet the demand, resulting in the rapid proliferation of slums. In order to understand this phenomenon, Patel et al. (2012) first developed a conceptual model

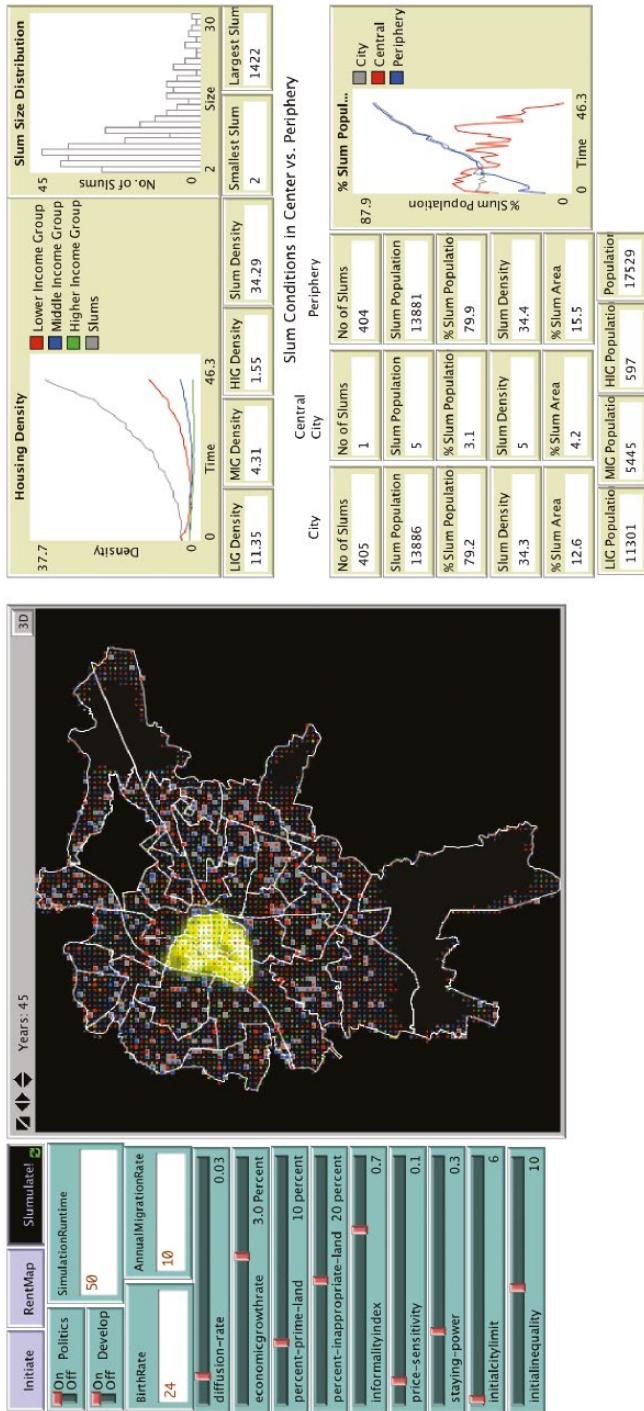


Figure A.13 Graphical user interface of the Slumulation model, including input parameters (left), the spatial environment based on Ahmedabad, India (centre), and various output parameters (right)

of the basic mechanisms that are suggested to cause the growth of slums, before developing a more spatially explicit agent-based model in NetLogo to explore how slums form and expand in Ahmedabad, India, where 41% of its population live in slums (Patel et al., 2018). The GUI of the Slumulation model is shown in Figure A.13. The model shows how the collective effect of many interacting inhabitants of slums as well as non-slum actors (e.g. local government, developers) generates the emergent structure of slums at the macro scale.

Model available at: <http://www.gisagents.org/p/an-agent-based-modeling-approach-to.html>

A.14 RiftLand: Analysing Conflict, Disasters and Humanitarian Crises in East Africa

The RiftLand model (Cioffi-Revilla et al., 2011; Kennedy et al., 2014) was developed in MASON and utilises GeoMASON to simulate coupled social and natural systems. The model is based on several countries in East Africa (roughly 2.6 million square kilometres), including Kenya, Uganda, Rwanda, Burundi, and their respective borderlands with Somalia, Ethiopia, Sudan, Democratic Republic of Congo and Tanzania, as shown in Figure A.14. Its purpose was to explore how

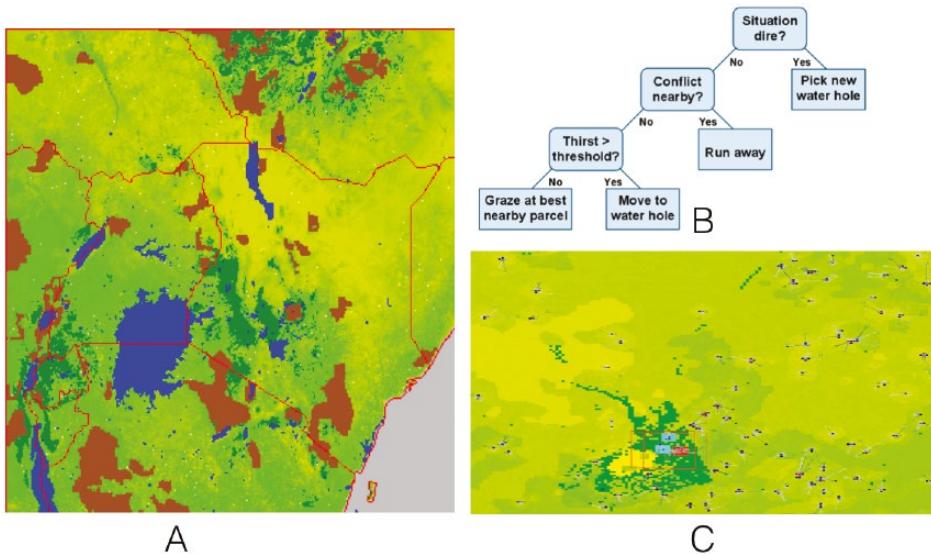


Figure A.14 The RiftLand model of parts of East Africa: (A) the modelled region, where yellow to green represents vegetation, blue is fresh water, grey is the Indian Ocean and brown are nature reserves; (B) an example of decision-making of the household herding model; (C) a zoomed-in section of the model where individual herders and farmers are shown

climate change could impact the inhabitants (herders, farmers, urban populations) of this region, including their decision-making (using a fast and frugal decision tree) as shown in Figure A.14B and how this results in movement and conflict between people (Figure A.14C).

Model available at: <https://github.com/eclab/mason>

A.15 Modelling Forced Migration

Hu (2016) developed a geographically explicit agent-based model using MASON and GeoMASON to explore individual movement paths for the Syrian refugee crisis (Figure A.15). The model was built using open data on migration routes, sources of refugees, numbers, etc. (e.g. United Nations High Commissioner for

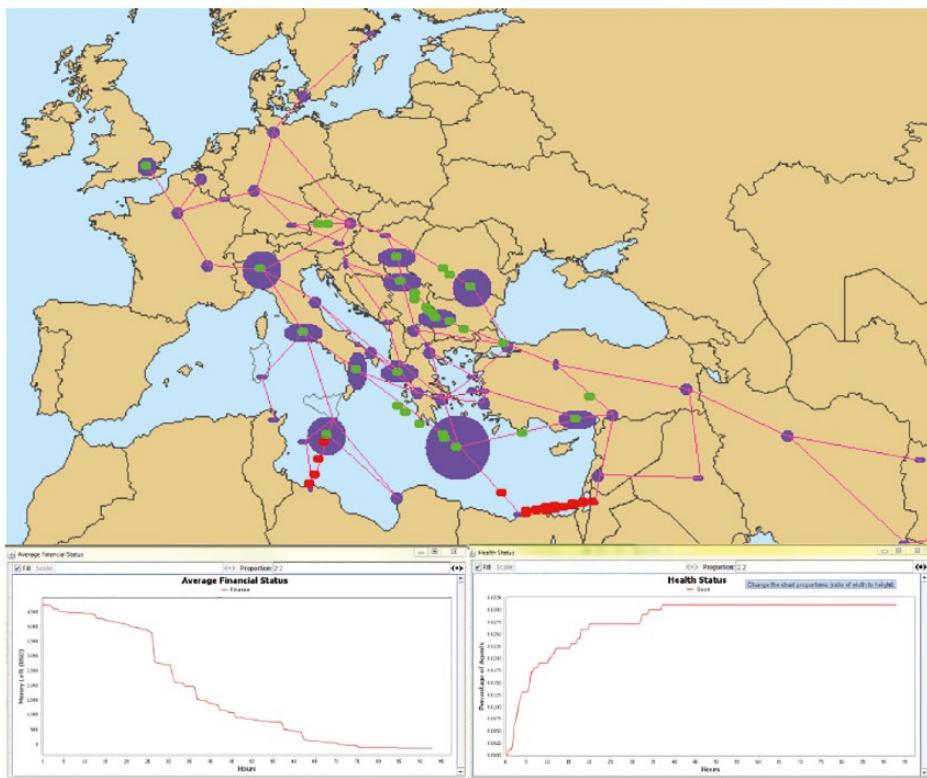


Figure A.15 An agent-based model of forced migration: top: the spatial environment, where lines represent migration routes, and nodes represent numbers of migrants. Purple nodes represent final destinations, red nodes show migrant deaths and green nodes show migrants en route. Bottom: charts recoding average financial status of the migrants and the number that have died en route

Refugees, International Organisation for Migration, OpenStreetMap) and individual agent goal selection in accordance with the theory of intervening opportunities (Stouffer, 1940). Results from such a model could provide guidance for policy and humanitarian decisions in the sense that if past migration routes and number of migrants can be predicated, we can ask what-if questions; for example, if a certain sea route is closed how will the refugees change their routes and thus where should aid centres be placed?

Model available at: <https://github.com/eclab/mason>

A.16 Studying Coupled Human-Artificial-Natural Systems in Boreal and Arctic Regions

The NorthLands model of Cioffi-Revilla et al. (2015, 2016) explores the potential effects of climate change in the boreal and Arctic regions on its inhabitants (over 205 million people in approximately 30 million households). The model integrates two main modules – a physical permafrost module and a movement module – in order to study human migratory movements using MASON and GeomASON. To give a sense of what the model looks like, in Figure A.16 we show its GUI.

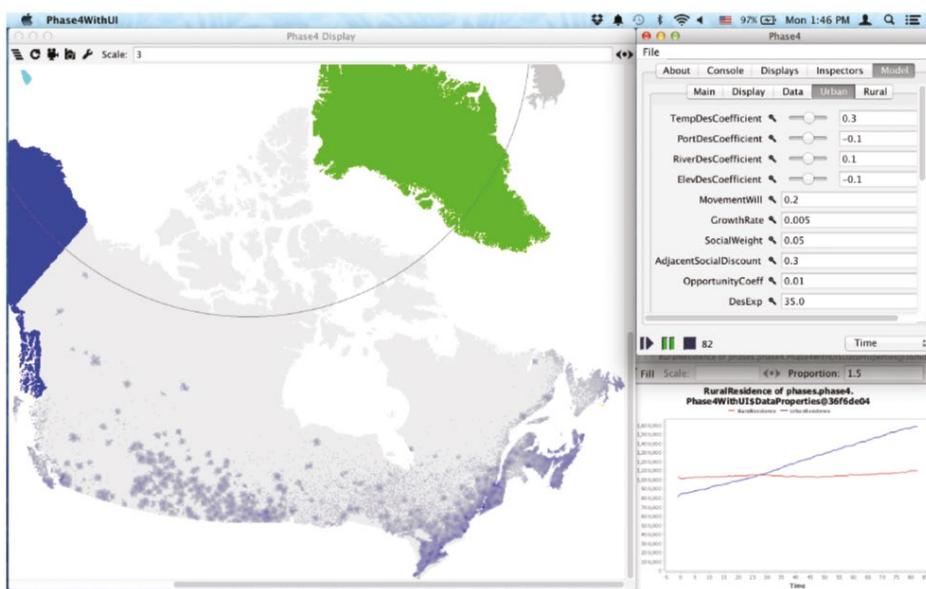


Figure A.16 Graphical user interface of the NorthLands model. Clockwise from left: the spatial environment showing the region of Canada with population distribution in the year 1921, parameter settings and an output time series measuring rural (red) and urban (blue) households

One can explore (via the permafrost module) how the land surface properties (load-bearing capacity, infrastructure stability, etc.) may be impacted due to thawing brought on by climate change, using past records and future projections for temperature from the Community Earth System Model (Hurrell et al., 2013), and how such changes can impact both urban and rural households' decision-making and potentially cause migration (via the movement module). This combination of modules allows one to see how biophysical events in the natural system affect both the physical infrastructure and the human population.

Model available at: <https://github.com/eclab/mason>

REFERENCES

- Aaby, K., Abbey, R.L., Herrmann, J.W., Treadwell, M., Jordan, C.S. and Wood, K. (2006) Embracing computer modeling to address pandemic influenza in the 21st century. *Journal of Public Health Management and Practice*, 12 (4), 365–372.
- Abar, S., Theodoropoulos, G.K., Lemarinier, P. and O'Hare, G.M. (2017) Agent based modelling and simulation tools: A review of the state-of-art software. *Computer Science Review*, 24, 13–33.
- ABC7 News (2010) Chicago links police, private cameras. <http://abc7chicago.com/archive/7370352/> (accessed on 30 March 2017).
- Abrahart, R.J. and See, L.M. (2014) *GeoComputation*. Boca Raton, FL: CRC Press.
- Ahearn, S.C., Smith, J.L.D., Joshi, A.R. and Ding, J. (2001) TIGMOD: An individual-based spatially explicit model for simulating tiger/human interaction in multiple use forests. *Ecological Modelling*, 140, 81–97.
- Ajzen, I. (1991) The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, 50 (2), 179–211.
- Al-Ahmadi, K., See, L., Heppenstall, A. and Hogg, J. (2009) Calibration of a fuzzy cellular automata model of urban dynamics in Saudi Arabia. *Ecological Complexity*, 6 (2), 80–101.
- Alam, S.J. and Geller, A. (2012) Networks in agent-based social simulation. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 199–218. Dordrecht: Springer.
- Alexanderson, G. (2006) About the cover: Euler and Königsberg's bridges: A historical view. *Bulletin of the American Mathematical Society*, 43 (4), 567–573.
- Alizadeh, M., Cioffi-Revilla, C. and Crooks, A.T. (2015) The effect of in-group favoritism on the collective behavior of individuals' opinions. *Advances in Complex Systems*, 18, 02.
- Alizadeh, M., Cioffi-Revilla, C. and Crooks, A.T. (2017) Generating and analyzing spatial social networks. *Computational and Mathematical Organization Theory*, 23 (3), 362–390.
- Alonso, W. (1964) *Location and Land Use: Toward a General Theory of Land Rent*. Cambridge, MA: Harvard University Press.
- Ammar, M.B., Neji, M. and Gouardères, G. (2006) Conversational embodied peer agents in affective e-learning. In G. Rebolledo-Mendez and E. Martinez-Miron (eds), *Workshop on Motivational and Affective Issues in ITS. 8th International Conference on ITS*, pp. 29–37. Taiwan, China.

REFERENCES

- Amouroux, E., Chu, T.Q., Boucher, A. and Drogoul, A. (2007) *GAMA: An Environment for Implementing and Running Spatially Explicit Multi-agent Simulations*, pp. 359–371. Berlin: Springer.
- An, L. (2012) Modeling human decisions in coupled human and natural systems: Review of agent-based models. *Ecological Modelling*, 229, 25–36.
- An, L., Zvoleff, A., Liu, J., and Axinn, W. (2014) Agent-based modeling in coupled human and natural systems (CHANS): Lessons from a comparative analysis. *Annals of the Association of American Geographers*, 104 (4), 723–745.
- Anderson, J.R. and Lebiere, C. (1998) *The Atomic Components of Thought*. Mahwah, NJ: Erlbaum.
- Anokye, M., Abdul-Aziz, A.R., Annin, K. and Oduro, F.T. (2013) Application of queuing theory to vehicular traffic at signalized intersection in Kumasi-Ashanti Region, Ghana. *American International Journal of Contemporary Research*, 3 (7), 23–29.
- Anselin, L., Syabri, I. and Kho, Y. (2005) GeoDa: An introduction to spatial data analysis. *Geographical Analysis*, 38 (1), 5–22.
- Arthur, W.B. (1994) Inductive reasoning and bounded rationality. *American Economic Review*, 84, 406–411.
- Aslam, S. (2018) Twitter by the Numbers: Stats, Demographics & Fun Facts. <https://www.omnicoreagency.com/twitter-statistics/> (accessed on 3 August 2018).
- Atkins (2009) Oxford Circus Diagonal Crossing. <http://www.atkinsglobal.com/en-gb/projects/oxford-circus-diagonal-crossing> (accessed on 3 August 2018).
- Augustijn-Beckers, E., Flacke, J. and Retsios, B. (2011a) Simulating informal settlement growth in Dar es Salaam, Tanzania: An agent-based housing model. *Computers, Environment and Urban Systems*, 35, 93–103.
- Augustijn-Beckers, E., Useya, J., Zurita-Milla, R. and Osei, F. (2011b) Simulation of cholera diffusion to compare transmission mechanisms. In *Proceedings of the 11th International Conference on Geocomputation*, University College London, pp. 39–42.
- Axelrod, R. (1995) The convergence and stability of cultures: Local convergence and global polarization. SFI Working Paper 1995-03-028, Santa Fe Institute, Santa Fe, NM.
- Axelrod, R. (1997a) Advancing the art of simulation in the social sciences. In R. Conte, R. Hegselmann and P. Terna (eds), *Simulating Social Phenomena*, Lecture Notes in Economics and Mathematical Systems 456, pp. 21–40. Berlin: Springer.
- Axelrod, R. (1997b) *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*. Princeton, NJ: Princeton University Press.

REFERENCES

- Axelrod, R. (2007) Simulation in the social sciences. In J. Renard (ed.), *Handbook of Research on Nature Inspired Computing for Economy and Management*, pp. 90–100. Hershey, PA: Idea Group.
- Axtell, R., Axelrod, R., Epstein, J.M. and Cohen, M.D. (1996) Aligning simulation models: A case study and results. *Computational & Mathematical Organization Theory*, 1 (2), 123–141.
- Axtell, R. and Epstein, J.M. (1994) Agent-based modelling: Understanding our creations. *Bulletin of the Santa Fe Institute* (Winter), 28–32.
- Axtell, R., Epstein, J.M., Dean, J.S., Gumerman, G.J., Swedlund, A.C., Harburger, J., Chakravarty, S., Hammond, R., Parker, J. and Parker, M. (2002) Population growth and collapse in a multiagent model of the Kayenta Anasazi in Long House Valley. *Proceedings of the National Academy of Sciences*, 99 (3), 7275–7279.
- Bailey, T. and Gatrell, T. (1995) *Interactive Spatial Data Analysis*. Harlow: Longman Scientific & Technical.
- Baker, M. (2016) 1,500 scientists lift the lid on reproducibility. *Nature*, 533 (7604), 452.
- Balci, O. (1996) Verification, validation, and testing. In J. Banks (ed.), *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*, pp. 335–393. New York: John Wiley & Sons.
- Balke, T. and Gilbert, N. (2014) How do agents make decisions? A survey. *Journal of Artificial Societies and Social Simulation*, 17(4), 13.
- Ballas, D., Clarke, G., Dorling, D., Eyre, H., Thomas, B. and Rossiter, D. (2005) SimBritain: A spatial microsimulation approach to population dynamics. *Population, Space and Place*, 11 (1), 3–34.
- Balzer, W. (2000) SMASS: A sequential multi-agent system for social simulation. In R. Suleiman, K.G. Troitzsch and N. Gilbert (eds), *Tools and Techniques for Social Science Simulation*, pp. 65–82. Heidelberg: Physica-Verlag.
- Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., Pritzel, A., Chadwick, M.J., Degris, T., Modayil, J., Wayne, G., Soyer, H., Viola, F., Zhang, B., Goroshin, R., Rabinowitz, N., Pascanu, R., Beattie, C., Petersen, S., Sadik, A., Gaffney, S., King, H., Kavukcuoglu, K., Hassabis, D., Hadsell, R. and Kumaran, D. (2018) Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705), 429–433.
- Banos, A. and Genre-Grandpierre, C. (2012) Towards new metrics for urban road networks: Some preliminary evidence from agent-based simulations. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 627–642. Dordrecht: Springer.
- Barabási, A. and Albert, R. (1999) Emergence of scaling in random networks. *Science*, 286 (5439), 509–512.

REFERENCES

- Barabási, A. and Bonabeau, E. (2003) Scale-free networks. *Scientific American*, 288 (5), 50–59.
- Barnaud, C., Bousquet, F. and Trebuil, G. (2008) Multi-agent simulations to explore rules for rural credit in a highland farming community of northern Thailand. *Ecological Economics*, 66(4), 615–627.
- Barreteau, O. and others (2003) Our companion modelling approach. *Journal of Artificial Societies and Social Simulation*, 6 (2), 1.
- Barreteau, O., Bousquet, F. and Attonaty, J.M. (2001) Role-playing games for opening the black box of multi-agent systems: Method and lessons of its application to Senegal River Valley irrigated systems. *Journal of Artificial Societies and Social Simulation*, 4 (2), 5.
- Barrett, C., Bisset, K., Holzer, M., Konjevod, G., Marathe, M. and Wagner D. (2009) Implementations of routing algorithms for transportation networks. In C. Demetrescu, A.V. Goldberg and D.S. Johnson (eds), *The Shortest Path Problem: Ninth DIMACS Implementation Challenge*, pp. 309–319. American Mathematical Society, Providence, RI.
- Bastian, M., Heymann, S. and Jacomy M. (2009) Gephi: An open source software for exploring and manipulating networks. In *Proceedings of the Third International Conference on Web and Social Media*, San Jose, CA, pp. 361–362. Menlo Park, CA: AAAI Press.
- Batty, M. (1976) *Urban Modelling: Algorithms, Calibrations, Predictions*. Cambridge: Cambridge University Press.
- Batty, M. (1979) Progress, success, and failure in urban modelling. *Environment and Planning A*, 11 (8), 863–878.
- Batty, M. (1992) Urban modelling in computer-graphic and geographic information system environments. *Environment and Planning B*, 19 (6), 663–685.
- Batty, M. (1995) Cities and complexity: Implications for modelling sustainability. In J. Brotchie, M. Batty, E. Blakely, P. Hall and P. Newton (eds), *Cities in Competition. Productive and Sustainable Cities for the 21st Century*, pp. 469–486. Melbourne: Longman.
- Batty, M. (2001) Polynucleated urban landscapes. *Urban Studies*, 38 (4), 635–655.
- Batty, M. (2005) *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-Based Models, and Fractals*. Cambridge, MA: MIT Press.
- Batty, M. (2008) Fifty years of urban modelling: Macro-statics to micro-dynamics. In S. Albeverio, D. Andrey, P. Giordano and A. Vancheri (eds), *The Dynamics of Complex Urban Systems: An Interdisciplinary Approach*, pp. 1–20. Heidelberg: Physica-Verlag.
- Batty, M. (2012) A generic framework for computational spatial modelling. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 19–50. Dordrecht: Springer.
- Batty, M. (2013) *The New Science of Cities*. Cambridge, MA: MIT Press.

REFERENCES

- Batty, M. (2018) *Inventing Future Cities*. Cambridge, MA: MIT Press.
- Batty, M., Desyllas, J. and Duxbury, E. (2003) Safety in numbers? Modelling crowds and designing control for the Notting Hill carnival. *Urban Studies*, 40 (8), 1573–1590.
- Batty, M., Barros, J. and Alves, S., Jr. (2004a) Cities: Continuity, transformation and emergence. Working Paper 72, Centre for Advanced Spatial Analysis (University College London), London.
- Batty, M., Steadman, P. and Xie, Y. (2004b) Visualisation in spatial modelling. Working Paper 76, Centre for Advanced Spatial Analysis (University College London), London.
- Batty, M. and Torrens, P.M. (2005) Modelling and prediction in a complex world. *Futures*, 37 (7), 745–766.
- Bell, A.R., Robinson, D.T., Malik, A. and Dewal S. (2015) Modular ABM development for improved dissemination and training. *Environmental Modelling & Software*, 73, 189–200.
- Belward, A.S. and Skøien, J. O. (2015) Who launched what, when and why: Trends in global land-cover observation capacity from civilian earth observation satellites. *ISPRS Journal of Photogrammetry and Remote Sensing*, 103, 115–128.
- Benenson, I., Aronovich, S. and Noam, S. (2005) Let's talk objects: Generic methodology for urban high-resolution simulation. *Computers, Environment and Urban Systems*, 29 (4), 425–453.
- Benenson, I., Martens, K. and Birfir, S. (2008) PARKAGENT: An agent-based model of parking in the city. *Computers, Environment and Urban Systems*, 32 (6), 431–439.
- Benenson, I., Omer, I. and Hatna, E. (2002) Entity-based modelling of urban residential dynamics: The case of Yaffo, Tel Aviv. *Environment and Planning B*, 29 (4), 491–512.
- Benenson, I. and Torrens, P.M. (2004) *Geosimulation: Automata-Based Modelling of Urban Phenomena*. Hoboken, NJ: John Wiley & Sons.
- Bennett, D.A. and Tang, W. (2006) Modelling adaptive, spatially aware, and mobile agents: Elk migration in Yellowstone. *International Journal of Geographical Information Science*, 20, 1039–1066.
- Bersini, H. (2012) UML for ABM. *Journal of Artificial Societies and Social Simulation*, 15 (1), 9.
- Bert, F., North, M., Rovere, S., Tatara, E., Macal, C. and Podestá G. (2015) Simulating agricultural land rental markets by combining agent-based models with traditional economics concepts: The case of the Argentine Pampas. *Environmental Modelling & Software*, 71, 97–110.
- Beuck, U., Rieser, M., Strippgen, D., Balmer, M. and Nagel K. (2008) Preliminary results of a multi-agent traffic simulation for Berlin. In S. Albeverio, D. Andrey, P. Giordano and A. Vancheri (eds), *The Dynamics of Complex Urban Systems: An Interdisciplinary Approach*, pp. 75–94. Heidelberg: Physica-Verlag.

REFERENCES

- Bharathy, G.K. and Silverman, B. (2010) Validating agent based social systems models. In *Proceedings of the Winter Simulation Conference*, Baltimore, MD, pp. 441–453. New York: ACM.
- Billari, F.C. and Prskawetz, A. (2003) *Agent-Based Computational Demography: Using Simulation to Improve Our Understanding of Demographic Behaviour*. Heidelberg: Physica-Verlag.
- Birkin, M., Clarke, G.P., Clarke M. and Wilson, A.G. (1996) *Intelligent GIS: Location Decisions and Strategic Planning*. Cambridge: GeoInformation Group.
- Birkin, M. and Wu, B. (2012) A review of microsimulation and hybrid agent-based approaches. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 51–68. Dordrecht: Springer.
- Birks, D., Townsley, M. and Stewart, A. (2012) Generative explanations of crime: Using simulation to test criminological theory. *Criminology*, 50 (1), 221–254.
- Birks, D., Townsley, M. and Stewart, A. (2014) Emergent regularities of interpersonal victimization: An agent-based investigation. *Journal of Research in Crime and Delinquency*, 5 (1), 119–140.
- Blei, D.M., Ng, A.Y. and Jordan M.I. (2003) Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 993–1022.
- Bloomquist, K. (2011) Tax compliance as an evolutionary coordination game: An agent-based approach. *Public Finance Review*, 39 (1), 25–49.
- Bonabeau, E. (2002) Agent-based modelling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99 (3), 7280–7287.
- Bonabeau, E. (2003a) Don't trust your gut. *Harvard Business Review*, 81, 116–123.
- Bonabeau, E. (2003b) Predicting the unpredictable. *Harvard Business Review*, 80, 109–116.
- Bone, C. and Dragićević, S. (2010) Simulation and validation of a reinforcement learning agent-based model for multi-stakeholder forest management. *Computers, Environment and Urban Systems*, 34 (2), 162–174.
- Bone, C., Dragićević, S. and White R. (2011) Modeling-in-the-middle: Bridging the gap between agent-based modeling and multi-objective decision making for land use change. *International Journal of Geographical Information Science*, 25 (5), 717–737.
- Borgatti, S.P., Everett, M.G. and Freeman L.C. (2002) UCINET For Windows: Software for social network analysis – user's guide. Technical report, Analytic Technologies, Boston.
- Borgatti, S.P., Mehra, A., Brass, D.J. and Labianca, G. (2009) Network analysis in the social sciences. *Science*, 323 (5916), 892–895.
- Box, G.E.P. (1979) Robustness in the strategy of scientific model building. MRC Technical Summary Report no. 1954, Mathematics Research Center, University of Wisconsin–Madison.

REFERENCES

- Box, P. (2001) Kenge GIS-CA class template for Swarm. *Natural Resources and Environmental Issues*, 8.
- Brail, R.K. and Klosterman, R.E. (eds) (2001) *Planning Support Systems: Integrating Geographic Information Systems, Models and Visualisation Tools*. Redlands, CA: ESRI Press.
- Brailsford, S. and Schmidt, B. (2003) Towards incorporating human behaviour in models of health care systems: An approach using discrete event simulation. *European Journal of Operational Research*, 150 (1), 19–31.
- Brantingham, P.L., Glasser, U., Kinney, B., Singh, K. and Vajihollahi, M. (2005a) A computational model for simulating spatial aspects of crime in urban environments. In *Proceedings of the 2005 International Conference on Systems, Man and Cybernetics*, Waikoloa, HI, Vol. 4, pp. 3667–3674. Piscataway, NJ: Institute of Electrical and Electronics Engineers.
- Brantingham, P.L., Glasser, U., Kinney, B., Singh, K. and Vajihollahi, M. (2005b) Modeling urban crime patterns: Viewing multi-agent systems as abstract state machines. In D. Beauquier, E. Börger and A. Slissenko (eds), *Proceedings of the 12th International Workshop on Abstract State Machines*, Paris, France, pp. 101–117.
- Bratman, M.E., Israel, D.J. and Pollack M.E. (1988) Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4 (3), 349–355.
- Brown, D.G., Page, S.E., Riolo, R. and Rand, W. (2004) Agent-based and analytical modeling to evaluate the effectiveness of greenbelts. *Environmental Modelling & Software*, 19 (12), 1097–1109.
- Brown, D.G., Page, S.E., Riolo, R., Zellner, M. and Rand, W. (2005) Path dependence and the validation of agent-based spatial models of land use. *International Journal of Geographical Information Science*, 19 (2), 153–174.
- Brunsdon, C. and Comber, A. (2015) *An Introduction to R for Spatial Analysis and Mapping*. London: Sage.
- Brunsdon, C. and Singleton, A. (eds) (2015a) *Geocomputation: A Practical Primer*. London: Sage.
- Brunsdon, C. and Singleton, A. (2015b) Reproducible research: Concepts, techniques and issues. In C. Brunsdon and A. Singleton (eds), *Geocomputation: A Practical Primer*, pp. 254–264. London: Sage.
- Burger, A., Oz, T., Crooks, A.T. and Kennedy, W.G. (2017) Generation of realistic mega-city populations and social networks for agent-based modeling. In *Proceedings of the Computational Social Science Society of Americas Conference*, article no. 15. Santa Fe, NM. New York: ACM.
- Burton, J.W. (1979) *Deviance, Terrorism and War: The Process of Solving Unsolved Social and Political Problems*. Canberra: Australian National University Press.
- Bybee, G. and Eng, L.A. (2012) Airport Security Line Simulation. <http://beyondbitsandatomsblog.stanford.edu/spring2012/assignments/assignment-4-creating-netlogo-models/airport-security-line-simulation-netlogo-abm/> (accessed on 10 April 2017).

REFERENCES

- Cailiou, P., Gaudou, B., Grignard, A., Truong, C.Q. and Taillandier P. (2017) A simple-to-use BDI architecture for agent-based modeling and simulation. Groningen, The Netherlands. In W. Jager, R. Verbrugge, A. Flache, G. de Roo, L. Hoogduin and C. Hemelrijk (eds), *Advances in Social Simulation 2015*, Advances in Intelligent Systems and Computing 528, pp. 15–28. Cham: Springer.
- Carrella, E. (2014) Zero-knowledge traders. *Journal of Artificial Societies and Social Simulation*, 17 (3), 4.
- Castle, C.J.E. (2007a) Agent-based modelling of pedestrian evacuation: A study of London's King's Cross underground station. Ph.D. thesis, University College London.
- Castle, C.J.E. (2007b) Guidelines for assessing pedestrian evacuation software applications. Working Paper 115, Centre for Advanced Spatial Analysis (University College London), London.
- Castle, C.J.E. and Crooks, A.T. (2006) Principles and concepts of agent-based modelling for developing geospatial simulations. Working Paper 110, Centre for Advanced Spatial Analysis (University College London), London.
- Cederman, L.E. (2001) Agent-based modelling in political science. *The Political Methodologist*, 10 (1), 16–22.
- Chainey, S. and Ratcliffe, J. (2005) *GIS and Crime Mapping*. Chichester: John Wiley & Sons.
- Chen, Z. and Schintler, L.A. (2015) Sensitivity of location-sharing services data: Evidence from American travel pattern. *Transportation*, 42 (4), 669–682.
- Chowell, G., Viboud, C., Simonsen, L., Merler, S. and Vespignani, A. (2017) Perspectives on model forecasts of the 2014–2015 Ebola epidemic in West Africa: Lessons and the way forward. *BMC Medicine*, 15, 42.
- Chrismann, N. (2006) *Charting the Unknown: How Computer Mapping at Harvard Became GIS*. Redlands, CA: ESRI Press.
- Christaller, W. (1933) *Die Centralen*. Jena: Gustav Fischer.
- Cialdini, R., Kallgren, C. and Reno, R.R. (1991) A focus theory of normative conduct: A theoretical refinement and reevaluation of the role of norms in human behavior. In M. Zanna (ed.), *Advances in Experimental Social Psychology*, Vol. 24, pp. 201–234. New York: Academic Press.
- Cioffi-Revilla, C. (2002) Invariance and universality in social agent-based simulations. *Proceedings of the National Academy of Sciences*, 99 (3), 7314–7316.
- Cioffi-Revilla, C. (2014) *Introduction to Computational Social Science: Principles and Applications*. New York: Springer.
- Cioffi-Revilla, C., Gulden, T., Kennedy, W. and Coletti, M. (2011) MASON RiftLand: An agent based model for analyzing conflict, disasters, and humanitarian crises in East Africa. Technical report, George Mason University, Fairfax, VA.

REFERENCES

- Cioffi-Revilla, C., Rogers, J.D., Schopf, P., Luke, S., Bassett, J., Hailegiorgis, A., Kennedy, W., Froncek, P., Mulkerin, M., Shaffer, M. and Wei, E. (2015) MASON NorthLands: A geospatial agent-based model of coupled human-artificial-natural systems in boreal and Arctic regions. In W. Jager, R. Verbrugge, A. Flache, G. de Roo, L. Hoogduin and C. Hemelrijck, (eds), *Proceedings of the Eleventh Conference of the European Social Simulation Association*, Groningen, Netherlands.
- Cioffi-Revilla, C., Rogers, J.D., Schopf, P., Luke, S., Bassett, J., Hailegiorgis, A., Kennedy, W., Revay, P., Mulkerin, M., Shaffer, M. and Wei, E. (2016) MASON NorthLands: A geospatial agent-based model of coupled human-artificial-natural systems in boreal and Arctic regions. In *Proceedings of the Computational Social Science Society of Americas Conference*, Santa Fe, NM.
- Cioffi-Revilla, C. and Rouleau, M. (2010) MASON ReBeLand: An agent-based model of politics, environment, and insurgency. *International Studies Review*, 12 (1), 31–46.
- Clark, P.J. and Evans, F.C. (1954) Distance to nearest neighbor as a measure of spatial relationships in populations. *Ecology*, 35 (4), 445–453.
- Clark, W.A.V. (1991) Residential preferences and neighbourhood racial segregation: A test of the Schelling segregation model. *Demography*, 28(1), 1–19.
- Clarke, K.C., Gazulis, N., Dietzel, C.K. and Goldstein, N.C. (2006) A decade of SLEUTHing: Lessons learned from applications of a cellular automaton land use change model. In P. Fisher (ed.), *Classics from IJGIS: Twenty Years of the International Journal of Geographical Information Science and Systems*, pp. 413–426. Boca Raton, FL: Taylor & Francis.
- Clarke, K.C., Hoppen, S. and Gaydos, L.J. (1997) A self-modifying cellular automaton model of historical urbanization in the San Francisco Bay area. *Environment and Planning B*, 24 (2), 247–261.
- Collier, N. and North, M.J. (2004) Repast for Python Scripting. In *Proceedings of the Agent 2004 Conference on Social Dynamics: Interaction, Reflexivity and Emergence*, pp. 231–237. Washington, DC: US Department of Energy, Office of Science.
- Collier, N. and North M. (2013) Parallel agent-based simulation with Repast for high performance computing. *Simulation*, 89, 1215–1235.
- Cordasco, G., De Chiara, R., Mancuso, A., Mazzeo, D., Scarano, V. and Spagnuolo C. (2013) Bringing together efficiency and effectiveness in distributed simulations: The experience with D-MASON. *Simulation* 89, 1236–1253.
- Costanza, R. (1989) Model goodness of fit: A multiple resolution procedure. *Ecological Modelling*, 47 (3–4), 199–215.
- Couclelis, H. (1992) People manipulate objects (but cultivate fields): Beyond the raster-vector debate in GIS. In A.U. Frank, I. Campari and U. Formentini (eds), *Theories and Methods of Spatio-temporal Reasoning in Geographic Space*, pp. 65–77. Berlin: Springer.

REFERENCES

- Couclelis, H. (2002) Modelling frameworks, paradigms, and approaches. In K.C. Clarke, B.E. Parks and M.P. Crane (eds), *Geographic Information Systems and Environmental Modelling*, pp. 36–50. Upper Saddle River, NJ: Prentice Hall.
- Cranshaw, J., Schwartz, R., Hong, J.I. and Sah N.M. (2012) The Livehoods Project: Utilizing social media to understand the dynamics of a city, *Proceedings of the Sixth International AAAI Conference on Weblogs on Social Media*, Dublin, Ireland, pp. 58–65 Palo Alto, CA. AAAI Press.
- Croitoru, A., Crooks, A.T., Radzikowski, J. and Stefanidis, A. (2013) GeoSocial Gauge: A system prototype for knowledge discovery from geosocial media. *International Journal of Geographical Information Science*, 27, 2483–2508.
- Croitoru, A., Crooks, A.T., Radzikowski, J., Stefanidis, A., Vatsavai, R.R. and Wayant, N. (2014) Geoinformatics and social media: A new big data challenge. In H.A. Karimi (ed.), *Big Data Techniques and Technologies in Geoinformatics*, pp. 207–232. Boca Raton, FL: CRC Press.
- Croitoru, A., Wayant, N., Crooks, A., Radzikowski, J. and Stefanidis, A. (2015) Linking cyber and physical spaces through community detection and clustering in social media feeds. *Computers, Environment and Urban Systems*, 53, 47–64.
- Crooks, A.T. (2006) Exploring cities using agent-based models and GIS. Working Paper 109, Centre for Advanced Spatial Analysis (University College London).
- Crooks, A.T. (2007a) Experimenting with cities: Utilizing agent-based models and GIS to explore urban dynamics. Ph.D. thesis, University College London.
- Crooks, A.T. (2007b) The Repast simulation/modelling system for geospatial simulation. Working Paper 123, Centre for Advanced Spatial Analysis (University College London), London.
- Crooks, A.T. (2010) Constructing and implementing an agent-based model of residential segregation through vector GIS. *International Journal of GIS* 24 (5), 661–675.
- Crooks, A.T. and Castle, C. (2012) The integration of agent-based modelling and geographical information for geospatial simulation. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 219–252. Dordrecht: Springer.
- Crooks, A.T., Castle, C.J.E. and Batty, M. (2008) Key challenges in agent-based modelling for geo-spatial simulation. *Computers, Environment and Urban Systems*, 32 (6), 417–430.
- Crooks, A.T., Croitoru, A., Jenkins, A., Mahabir, R., Agouris, P. and Stefanidis, A. (2016) User-generated big data and urban morphology. *Built Environment*, 42 (3), 396–414.
- Crooks, A.T., Croitoru, A., Lu, X., Wise, S., Irvine, J.M. and Stefanidis, A. (2015a) Walk this Way: Improving pedestrian agent-based models through scene activity analysis. *ISPRS International Journal of Geo-Information*, 4(3), 1627–1656.
- Crooks, A.T., Croitoru, A., Stefanidis, A. and Radzikowski, J. (2013) #Earthquake: Twitter as a distributed sensor system. *Transactions in GIS*, 17(1), 124–147.

REFERENCES

- Crooks, A.T. and Hailegiorgis, A. (2014) An agent-based modeling approach applied to the spread of cholera. *Environmental Modelling and Software*, 62, 164–177.
- Crooks, A.T. and Heppenstall, A. (2012) Introduction to agent-based modelling. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 85–108. Dordrecht: Springer.
- Crooks, A.T., Hudson-Smith, A. and Dearden J. (2009) Agent Street: An environment for exploring agent-based models in Second Life. *Journal of Artificial Societies and Social Simulation*, 12 (4), 10.
- Crooks, A.T., Hudson-Smith, A. and Patel, A. (2011) Advances and techniques for building 3D agent-based models for urban systems. In D.J. Marceau and I. Benenson (eds), *Advanced Geosimulation Models*, pp. 49–65. Hilversum: Bentham Science Publishers.
- Crooks, A., Masad, D., Croitoru, A., Cotnoir, A., Stefanidis, A. and Radzikowski, J. (2014) International relations: State-driven and citizen-driven networks. *Social Science Computer Review*, 32(2), 205–220.
- Crooks, A.T., Pfoser, D., Jenkins, A., Croitoru, A., Stefanidis, A., Smith, D. A., Karagiorgou, S., Efentakis, A. and Lamprianidis, G. (2015b) Crowdsourcing urban form and function. *International Journal of Geographical Information Science*, 29, 720–741.
- Crooks, A.T., Schechtner, K., Dey, A.K. and Hudson-Smith, A. (2017) Creating smart buildings and cities. *IEEE Pervasive Computing*, 16 (2), 23–25.
- Crooks, A.T. and Wise, S. (2013) GIS and agent-based models for humanitarian assistance. *Computers, Environment and Urban Systems*, 41, 100–111.
- Csardi, G. and Nepusz, T. (2006) The Igraph software package for complex network research. *InterJournal, Complex Systems*, 1695 (5).
- Dantzig, G. B. (1960) On the shortest route through a network. *Management Science*, 6 (2), 187–190.
- Darley, V. and Outkin, A.V. (2007) *NASDAQ Market Simulation: Insights on a Major Market from the Science of Complex Adaptive Systems*. River Edge, NJ: World Scientific Publishing.
- Daume, S. (2016) Mining Twitter to monitor invasive alien species – an analytical framework and sample information topologies. *Ecological Informatics*, 31, 70–82.
- Dawson, R.J., Peppe, R. and Wang, M. (2011) An agent-based model for risk-based flood incident management. *Natural Hazards*, 59 (1), 167–189.
- de Smith, M.J., Goodchild, M.F. and Longley, P.A. (2009) *Geospatial Analysis: A Comprehensive Guide to Principles, Techniques and Software Tools* (3rd edn). Winchelsea: Winchelsea Press.
- Deadman, P. J., Robinson, D. T., Moran, E. and Brondizio E. (2004) Effects of colonist household structure on land use change in the Amazon rainforest: An agent based simulation approach. *Environment and Planning B*, 31 (5), 693–709.

REFERENCES

- Deadman, P.J. and E. Schlager (2002) Models of individual decision making in agent-based simulation of common-pool-resource management institutions. In H.R. Gimblett (ed.), *Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes*, pp. 137–169. Oxford: Oxford University Press.
- Defense Threat Reduction Agency (2001) *The HPAC User's Guide*, Version 4.0.3. Prepared for Defense Threat Reduction Agency, Contract DSWA01-98-C-0110. McLean, VA: Science Applications International Corporation.
- Dempsey, D. (2012) GIS Industry Trends and Outlook. <https://www.gis-lounge.com/gis-industry-trends/> (accessed on 3 August 2018).
- Demšar, U., Reades, J., Manley, E. and Batty, M. (2018, May) Revisiting the past: Replicating fifty-year-old flow analysis using contemporary taxi flow data. *Annals of the American Association of Geographers*, 108 (3), 811–828.
- Dennett, A. and Wilson, A. (2013) A multilevel spatial interaction modelling framework for estimating interregional migration in Europe. *Environment and Planning A*, 45(6), 1491–1507.
- Diappi, L. and Bolchi, P. (2008) Smith's rent gap theory and local real estate dynamics: A multi-agent model. *Computers, Environment and Urban Systems*, 32 (1), 6–18.
- Dijkstra, E. W. (1959) A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269–271.
- Dong, M. and Chen, F.F. (2005) Performance modeling and analysis of integrated logistic chains: An analytic framework. *European Journal of Operational Research*, 162, 83–98.
- Duijn, M., Immers, L., Waaldijk, F. and Stoelhorst, H. (2003) Gaming Approach Route 26: A combination of computer simulation, design tools and social interaction. *Journal of Artificial Societies and Social Simulation*, 6(3), 7.
- Dunbar, R.I. (1998) The social brain hypothesis. *Evolutionary Anthropology*, 6 (5), 178–190.
- Dunbar, R. I., Arnaboldi, V., Conti, M. and Passarella, A. (2015) The structure of online social networks mirrors those in the offline world. *Social Networks*, 43, 39–47.
- Dunbar, R. I. and Spoors, M. (1995) Social networks, support cliques, and kinship. *Human Nature*, 6 (3), 273–290.
- Edmonds, B. (2006) How are physical and social spaces related? Cognitive agents as the necessary 'glue'. In F.C. Billari, T. Fent, A. Prskawetz and J. Scheffran (eds), *Agent-Based Computational Modelling Applications in Demography, Social, Economic and Environmental Sciences*, pp. 195–214. New York: Springer.
- Edmonds, B. and Moss, S. (2004) From KISS to KIDS – an 'anti-simplistic' modelling approach. In P. Davidsson, B. Logan and K. Takadama (eds), *Multi-Agent and Multi-Agent-Based Simulation*, pp. 130–144. Berlin: Springer.

REFERENCES

- Eidelson, B. M. and Lustick, I. (2004) VIR-POX: An agent-based analysis of small-pox preparedness and response policy. *Journal of Artificial Societies and Social Simulation*, 7 (3), 6.
- EMC2 (2014) The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. <https://www.emc.com/collateral/analyst-reports/idc-digital-universe-2014.pdf>.
- Epstein, J.M. (1999) Agent-based computational models and generative social science. *Complexity*, 4 (5), 41–60.
- Epstein, J.M. (2008) Why model? *Journal of Artificial Societies and Social Simulation*, 11 (4), 12.
- Epstein, J.M. (2009) Modelling to contain pandemics. *Nature*, 460, 687.
- Epstein, J.M. and Axtell, R. (1996) *Growing Artificial Societies: Social Science from the Bottom Up*. Cambridge, MA: MIT Press.
- Epstein, J.M., Pankajakshan, R. and Hammond, R.A. (2011) Combining computational fluid dynamics and agent-based modeling: A new approach to evacuation planning. *PLoS ONE*, 6 (5), e20139.
- Erdős, P. and Rényi, A. (1960) On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5 (1), 17–60.
- Erlang, A.K. (1909) The theory of probabilities and telephone conversations. *Nyt Tidsskrift for Matematik*, 20, 33–39.
- Etienne, M. (2003) SYLVOPAST: A multiple target role-playing game to assess negotiation processes in sylvopastoral management planning. *Journal of Artificial Societies and Social Simulation*, 6 (2), 5.
- Etienne, M. (ed.) (2014) *Companion Modelling: A Participatory Approach to Support Sustainable Development*. New York: Springer.
- Eubank, S., Guclu, H., Kumar, A.S., Marathe, M., Srinivasan, A., Toroczkai, Z. and Wang, N. (2004) Modelling disease outbreaks in realistic urban social networks. *Nature*, 429, 180–184.
- Euler, L. (1741) Solutio problematis ad geometriam situs pertinentis. *Commentarii Academiae Scientiarum Petropolitanae*, 8, 128–140.
- Facebook (2018) Company Info. <https://newsroom.fb.com/company-info/>.
- Favis-Mortlock, D. (2013) Non-linear dynamics, self-organization and cellular automata models. In J. Wainwright and M. Mulligan (eds), *Environmental Modelling: Finding Simplicity in Complexity* (2nd edn), pp. 45–68. Chichester: John Wiley & Sons.
- Ferris, C., Raybaud, B. and Madey, G. (2015) OpenMalaria and EMOD: A case study on model alignment. In *Proceedings of the Conference on Summer Computer Simulation*. San Diego, CA: Society for Computer Simulation International.
- Filatova, T., Parker, D. and van der Veen, A. (2009) Agent-based urban land markets: Agent's pricing behavior, land prices and urban land use change. *Journal of Artificial Societies and Social Simulation*, 12 (1), 3.

REFERENCES

- Filatova, T., Verburg, P.H., Parker, D.C. and Stannard, C.A. (2013) Spatial agent-based models for socio-ecological systems: Challenges and prospects. *Environmental Modelling & Software*, 45, 1–7.
- Flache, A. and R. Hegselmann (2001) Do irregular grids make a difference? Relaxing the spatial regularity assumption in cellular models of social dynamics. *Journal of Artificial Societies and Social Simulation*, 4 (4), 6.
- Fontaine, C.M. and Rounsevell, M. (2009) An agent-based approach to model future residential pressure on a regional landscape. *Landscape Ecology*, 24 (9), 1237–1254.
- Fontana, M. and Terna, P. (2014) From agent-based models to network analysis (and return): The policy-making perspective. *Journal of Policy and Complex Systems*, 1 (2), 77–92.
- Forrester, J. W. (1969) *Urban Dynamics*. Cambridge, MA: MIT Press.
- Fotheringham, A.S., Brunsdon, C. and Charlton, M. (2000) *Quantitative Geography: Perspectives on Spatial Data Analysis*. London: Sage.
- Fotheringham, A.S. and O’Kelly, M.E. (1989) *Spatial Interaction Models: Formulations and Applications*. New York: Springer.
- Frank, R. H. (1987) If *Homo Economicus* could choose his own utility function, would he want one with a conscience? *American Economic Review*, 77 (4), 593–604.
- Franklin, S. and Graesser, A. (1996) Is it an agent, or just a program? A taxonomy for autonomous agent. In *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, pp. 21–35. Berlin: Springer.
- Freeman, L.C. (1977) A set of measures of centrality based on betweenness. *Sociometry*, 40 (1), 35–41.
- Friedkin, N.E. (1998) *A Structural Theory of Social Influence*. New York: Cambridge University Press.
- Friedland, G. and Sommer, R. (2010) Cybercasing the joint: On the privacy implications of geotagging. In *Proceedings of the Fifth USENIX Workshop on Hot Topics in Security (HotSec 10)*, Washington, DC. New York: ACM.
- Galán, J.M., Izquierdo, L.R., Izquierdo, S.S., Santos, J.I., del Olmo, R., López-Paredes, A. and Edmonds, B. (2009) Errors and artefacts in agent-based modeling. *Journal of Artificial Societies and Social Simulation*, 12 (1), 1.
- GAMA (2016) GAMA Modeling and Simulation Development Environment. <http://gama-platform.org/> (accessed on 2 October 2016).
- Gardner, M. (1970) Mathematical games: The fantastic combinations of John Conway’s new solitaire game. *Scientific American*, 223 (4), 120–123.
- Geanakoplos, J., Axtell, R., Farmer, D., Howitt, P., Conlee, B., Goldstein, J., Hendrey, M., Palmer, N. and Yang, C. (2012) Getting at systemic risk via an agent-based model of the housing market. *American Economic Review*, 102 (3), 53–58.
- Gerber, A.S. and Rogers, T. (2009) Descriptive social norms and motivation to vote: Everybody’s voting and so should you. *Journal of Politics*, 71 (1), 178–191.

REFERENCES

- Gerrard, G. and Thompson, R. (2011) Two million cameras in the UK. *CCTV Image*, 42, 10–12.
- Getis, A. and Ord, J.K. (1992) The analysis of spatial association by use of distance statistics. *Geographical Analysis*, 24 (3), 189–206.
- Gigerenzer, G. and Gaissmaier, W. (2011) Heuristic decision making. *Annual Review of Psychology*, 62, 451–482.
- Gigerenzer, G. and Goldstein, D.G. (1996) Reasoning the fast and frugal way: Models of bounded rationality. *Psychological Review*, 103 (4), 650–669.
- Gigerenzer, G. and Selten, R. (eds) (2001) *Bounded Rationality: The Adaptive Toolbox*. Cambridge, MA: MIT Press.
- Gilbert, E.N. (1959) Random graphs. *Annals of Mathematical Statistics*, 30 (4), 1141–1144.
- Gilbert, N. and Banks, S. (2002) Platforms and methods for agent-based modelling. *Proceedings of the National Academy of Sciences of the USA*, 99 (3), 7197–7198.
- Gilbert, N., Maltby, S. and Asakawa, T. (2002) Participatory simulations for developing scenarios in environmental resource management. In C. Urban (ed.), *Third Workshop on Agent-Based Simulation*, pp. 67–72. Passau: SCS European Publishing House.
- Gilbert, N. and Terna, P. (2000) How to build and use agent-based models in social science. *Mind and Society*, 1(1), 57–72.
- Gilbert, N. and Troitzsch, K.G. (2005) *Simulation for the Social Scientist* (2nd edn). Maidenhead: Open University Press.
- Gimblett, H.R. (ed.) (2002) *Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes*. Oxford: Oxford University Press.
- Gintis, H. (2007) The dynamics of general equilibrium. *Economic Journal*, 117 (523), 1280–1309.
- Giuşcă, B. (2017) The Königsberg bridge problem. https://en.wikipedia.org/wiki/Graph_theory#/media/File:Kon
- Gode, D.K. and Sundern S. (1993) Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of Political Economy*, 101, 119–137.
- Goodchild, M.F. (1992) Geographical information science. *International Journal of Geographical Information Systems*, 6 (1), 31–45.
- Goodchild, M.F. (2007) Citizens as sensors: The world of volunteered geography. *GeoJournal*, 69 (4), 211–221.
- Goodchild, M.F., Yuan, M. and Cova, T. (2007) Towards a general theory of geographic representation in GIS. *International Journal of Geographical Information Science*, 21 (3), 239–260.
- Gordó, S.P., Aubán, J.B., Puchol, O.G., Barton, M. and Bergin S.M. (2015) The origins of agriculture in Iberia: A computational model. *Documenta Praehistorica*, 42, 117–131.

REFERENCES

- Government Accountability Office (2015) Geospatial data: Progress needed on identifying expenditures, building and utilizing a data infrastructure, and reducing duplicative efforts. Technical Report GAO-15-193, United States Government Accountability Office, Washington, D.C.
- Granovetter, M. (1985) Economic action and social structure: The problem of embeddedness. *American Journal of Sociology*, 91 (3), 481–510.
- Graybiel, A.M. (2008) Habits, rituals, and the evaluative brain. *Annual Review of Neuroscience*, 31 (1), 359–387.
- Grignard, A. (2016) Agent-based visualization: A simulation tool for the analysis of river morphosedimentary adjustments. In B. Gaudou and J.S. Sichman (eds), *Multi-agent Based Simulation XVI*, pp. 109–120. Cham: Springer.
- Grignard, A., Taillandier, P., Gaudou, B., Vo, D.A., Huynh, N.Q. and Drogoul A. (2013) GAMA 1.6: Advancing the art of complex agent-based modeling and simulation. In G. Boella, E. Elkind, B.T.R. Savarimuthu, F. Dignum and M.K. Purvis (eds), *PRIMA 2013: Principles and Practice of Multi-agent Systems*, pp. 117–131. Heidelberg: Springer.
- Grimm, V. (2002) Visual debugging: A way of analyzing, understanding, and communicating bottom-up simulation models in ecology. *Natural Resource Modeling*, 15 (1), 23–38.
- Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S., Huse, G., Huth, A., Jepsen, J., Jørgensen, C., Mooij, W., Müller, B., Pe'er, G., Piou, C., Railsback, S., Robbins, A., Robbins, M., Rossmanith, E., Ruger, N., Strand, E., Souissi, S., Stillman, R., Vabo, R., Visser, U. and DeAngelis, D.L. (2006) A standard protocol for describing individual-based and agent-based models. *Ecological Modelling*, 198 (1–2), 115–126.
- Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J. and Railsback, S.F. (2010) The ODD protocol: A review and first update. *Ecological Modelling*, 221(23), 2760–2768.
- Grimm, V. and Railsback, S.F. (2012) Designing, formulating, and communicating agent-based models. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 361–378. Dordrecht: Springer.
- Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W.M., Railsback, S.F., Thulke, H., Weiner, J., Wiegand, T. and DeAngelis, D.L. (2005) Pattern-oriented modelling of agent-based complex systems: Lessons from ecology. *Science*, 310 (5750), 987–991.
- Groeneveld, J., Müller, B., Buchmann, C.M., Dressler, G., Guo, C., Hase, N., Hoffmann, F., John, F., Klassert, C., Lauf, T. and Liebelt, V. (2017) Theoretical foundations of human decision-making in agent-based land use models – a review. *Environmental Modelling & Software*, 87, 39–48.

REFERENCES

- Guerin, S. and Carrera, F. (2010) Sand on fire: An interactive tangible 3D platform for the modeling and management of wildfires. *WIT Transactions on Ecology and the Environment*, 137, 57–68.
- Guerrero, O. A. and Axtell, R. (2011) Using Agentization for Exploring Firm and Labor Dynamics. In S. Osinga, G. Hofstede and T. Verwaart (eds), *Emergent Results of Artificial Economics*, pp. 139–150. New York: Springer.
- Gulden, T., Harrison, J.F. and Crooks A.T. (2011) Modeling cities and displacement through an agent-based spatial interaction model. In *Proceedings of the Computational Social Science Society of America Conference (2011)*, Santa Fe, NM.
- Haase, D., Lautenbach, S. and Seppelt, R. (2010) Modeling and simulating residential mobility in a shrinking city using an agent-based approach. *Environmental Modelling & Software*, 25, 1225–1240.
- Hagberg, A., Swart, P. and Schult, D. (2008) Exploring network structure, dynamics, and function using NetworkX. Technical report no. LA-UR-08-05495, Los Alamos National Laboratory, Los Alamos, NM.
- Haggett, P. and Chorley, R.J. (1969) *Network Analysis in Geography*. London: Edward Arnold.
- Haggett, P., Cliff, A.D. and Frey, A. (1977) *Locational Analysis in Human Geography: Locational Models*. London: Edward Arnold.
- Hahnel, D., Burgard, W., Fox, D., Fishkin, K. and Philipose, M. (2004) Mapping and localization with RFID technology. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1015–1020. Piscataway, NJ: IEEE.
- Haklay, M., O'Sullivan, D., Thurstain-Goodwin, M. and Schelhorn, T. (2001) 'So go downtown': Simulating pedestrian movement in town centres. *Environment and Planning B*, 28 (3), 343–359.
- Hamill, L. and Gilbert, N. (2009) Social circles: A simple structure for agent-based social network models. *Journal of Artificial Societies and Social Simulation*, 12 (2), 3.
- Hammond, D. and Mahesh, S. (1995) A simulation and analysis of bank teller manning. In *Proceedings of the Winter Simulation Conference*, pp. 1077–1080. Washington, DC: IEEE Computer Society.
- Hansen, D., Shneiderman, B. and Smith, M.A. (2010) *Analyzing Social Media Networks with NodeXL: Insights from a Connected World*. Burlington, MA: Morgan Kaufmann.
- Harland, K. (2008) Journey to learn: Geographical mobility and education provision. Ph.D. thesis, School of Geography, University of Leeds.
- Harland, K., Heppenstall, A.J., Smith, D. and Birkin M. (2012) Creating realistic synthetic populations at varying spatial scales: A comparative critique of population synthesis techniques. *Journal of Artificial Societies and Social Simulation*, 15 (1), 1.
- Harper, S. J., Westervelt, J. D. and Trame, A. (2002) Management application of an agent-based model: Control of cowbirds at the landscape scale.

REFERENCES

- In H.R. Gimblett (ed.), *Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes*, pp. 105–123. Oxford: Oxford University Press.
- Harris, R., O'Sullivan, D., Gahegan, M., Charlton, M., Comber, L., Longley, P., Brunsdon, C., Malleson, N., Heppenstall, A. and Singleton, A. (2017) More bark than bytes? Reflections on 21+ years of geocomputation. *Environment and Planning B*, 44 (4), 598–617.
- Harris, R., Sleight, P. and Webber, R. (2005) *Geodemographics, GIS and Neighbourhood Targeting*. Chichester: John Wiley & Sons.
- Harrower, M.A. and Brewer, C.A. (2003) ColorBrewer.org: An online tool for selecting color schemes for maps. *Cartographic Journal*, 40 (1), 27–37.
- Heckbert, S., Baynes, T. and Reeson, A. (2010) Agent-based modeling in ecological economics. *Annals of the New York Academy of Sciences*, 1185 (1), 39–53.
- Helbing, D. (2001) Traffic and related self-driven many-particle systems. *Reviews of Modern Physics*, 73 (4), 1067–1141.
- Helbing, D. and Baitette, S. (2011) How to do agent-based simulations in the future: From modeling social mechanisms to emergent phenomena and interactive systems design. Working Paper 11-06-024, Santa Fe Institute, Santa Fe, NM.
- Helbing, D., Farkas, I. and Vicsek , T. (2000) Simulating dynamical features of escape panic. *Nature* , 407, 487–490.
- Helbing, D. and Molnár, P. (1995) Social force model for pedestrian dynamics. *Physical Review E*, 51 (5), 4282–4286.
- Helft, M. (2016) The Godfather of Digital Maps. <https://www.forbes.com/sites/miguelhelft/2016/02/10/the-godfather-of-digital-maps/#47f3bc614da9> (accessed on 3 August 2018).
- Henein, C.M. and White, T. (2005) Agent-based modelling of forces in crowds. In P. Davidsson, B. Logan, and K. Takadama (eds), *Multi-Agent and Multi-Agent-Based Simulation*, pp. 173–184. Berlin: Springer.
- Heppenstall, A.J., Crooks, A.T., See, L.M. and Batty, M. (2012a) Reflections and conclusions: Geographical models to address grand challenges. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 739–748. Dordrecht: Springer.
- Heppenstall, A.J., Crooks, A.T., See, L.M. and Batty, M. (eds) (2012b) *Agent-Based Models of Geographical Systems*. Dordrecht: Springer.
- Heppenstall, A.J., Evans, A.J. and Birkin, M.H. (2006) Using hybrid agent-based systems to model spatially-influenced retail markets. *Journal of Artificial Societies and Social Simulation*, 9 (3), 2.
- Heppenstall, A.J., Evans, A.J. and Birkin, M.H. (2007) Genetic algorithm optimisation of a multi-agent system for simulating a retail market. *Environment and Planning B*, 34 (6), 1051–1070.

REFERENCES

- Heppenstall, A., Malleson, N. and Crooks, A.T. (2016) 'Space, the final frontier': How good are agent-based models at simulating individuals and space in cities? *Systems*, 4 (1), 9.
- Heywood, I., Cornelius, S. and Carver, S. (2006) *An Introduction to Geographical Information Systems* (3rd edn). Harlow: Pearson Education.
- Hjorth, A., Brady, C., Head, B. and Wilensky, U. (2015) Thinking within and between levels: Exploring reasoning with multi-level linked models. In O. Lindwall, P. Häkkinen, T. Koschmann, P. Tchounikine and S. Ludvigsen (eds), *11th International Conference on Computer Supported Collaborative Learning*, Volume 1, Gothenburg, Sweden, pp. 797–798. International Society of the Learning Sciences, Inc.
- Hollis, C. (2011) 2011 IDC Digital Universe Study: Big data is here, now what? <http://bit.ly/kouTgc>
- Horni, A., Nagel, K. and Axhausen, K.W. (2016) *The Multi-Agent Transport Simulation MATSim*. London: Ubiquity.
- Howe, T.R., Collier, N.T., North, M.J., Parker, M.T. and Vos, J.R. (2006) Containing agents: Contexts, projections, and agents. In D. Sallach, C.M. Macal, C.M. and M.J. North, (eds), *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, University of Chicago and Argonne National Laboratory, Chicago, IL, pp. 107–113.
- Hu, E. (2016) An agent-based model of the European refugee crisis. <https://github.com/elizabethhu/refugee-abm> (accessed on 3 August 2018).
- Huang, Q., Parker, D.C., Filatova, T. and Sun, S. (2014) A review of urban residential choice models using agent-based modeling. *Environment and Planning B*, 41 (4), 661–689.
- Hurrell, J.W., Holland, M.M., Gent, P.R., Ghan, S., Kay, J.E., Kushner, P.J., Lamarque, J.-F., Large, W.G., Lawrence, D. and Lindsay, K. (2013) The community earth system model: A framework for collaborative research. *Bulletin of the American Meteorological Society*, 94 (9), 1339–1360.
- Iltanen, S. (2012) Cellular automata in urban spatial modelling. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 69–84. Dordrecht: Springer.
- Izard, C.E. (2007) Basic emotions, natural kinds, emotion schemas, and a new paradigm. *Perspectives on Psychological Science*, 2 (3), 260–280.
- Jackson, J., Forest, B. and Sengupta, R. (2008) Agent-based simulation of urban residential dynamics and land rent change in a gentrifying area of Boston. *Transactions in GIS*, 12 (4), 475–491.
- Jacobs, J. (1961) *The Death and Life of Great American Cities*. New York: Vintage Books.
- Janssen, M.A. (2009) Understanding artificial Anasazi. *Journal of Artificial Societies and Social Simulation*, 12 (4), 13.

REFERENCES

- Jenkins, A., Croitoru, A., Crooks, A.T. and Stefanidis, A. (2016) Crowdsourcing a collective sense of place. *PLoS ONE* 11 (4), e0152932.
- Jjumba, A. and Dragičević, S. (2016) A development of spatiotemporal queries to analyze the simulation outcomes from a voxel automata model. *Earth Science Informatics*, 9 (3), 343–353.
- Johansson, A., Batty, M., Hayashi, K., Al Bar, O., Marcozzi, D. and Memish Z.A. (2012) Crowd and environmental management during mass gatherings. *Lancet Infectious Diseases*, 12 (2), 150–156.
- Johansson, A. and Kretz, T. (2012) Applied pedestrian modeling. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 451–462. Dordrecht: Springer.
- Johnson, P.E. (2002) Agent-based modeling: What I learned from the artificial stock market. *Social Science Computer Review*, 20 (2), 174–186.
- Johnston, K.M. (ed.) (2013) *Agent Analyst: Agent-Based Modeling in ArcGIS*. Redlands, CA: ESRI Press.
- Jordan, R., Birkin, M. and Evans, A. (2014) An agent-based model of residential mobility assessing the impacts of urban regeneration policy in the EASEL district. *Computers, Environment and Urban Systems*, 48, 49–63.
- Kahneman, D. and Tversky, A. (1979) Prospect theory: An analysis of decision under risk. *Econometrica*, 47 (2), 263–291.
- Kalnay, E. (2003) *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge: Cambridge University Press.
- Kaul, D., Bruno, J. and Roberts, J. (2004) *CATS User's Manual* (No. SAIC-00/1010). San Diego, CA: Science Applications International Corporation.
- Keeler, B.L., Wood, S.A., Polasky, S., Kling, C., Filstrup, C.T. and Downing J.A. (2015) Recreational demand for clean water: Evidence from geotagged photographs by visitors to lakes. *Frontiers in Ecology and the Environment*, 13 (2), 76–81.
- Kennedy, W.G. (2012) Modelling human behaviour in agent-based models. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 167–180. Dordrecht: Springer.
- Kennedy, W.G. and Bassett, J.K. (2011) Implementing a ‘fast and frugal’ cognitive model within a computational social simulation. In *Proceedings of The Computational Social Science Society of America Conference* (2011), Santa Fe, NM.
- Kennedy, W. G., Cotla, C.R., Gulden, T.R., Coletti, M. and Cioffi-Revilla, C. (2014) Towards validating a model of households and societies of East Africa. In S.H. Chen, I. Terano, H. Yamamoto and C.C. Tai (eds), *Advances in Computational Social Science: The Fourth World Congress*, pp. 315–328. New York: Springer.
- Kennedy, W.G., Hailegiorgis, A.B., Rouleau, M., Bassett, J.K., Coletti, M., Balan, G.C. and Gulden, T. (2010) An agent-based model of conflict in East Africa and the effect of watering holes. In *Proceedings of the 19th Annual Conference on*

REFERENCES

- Behavior Representation in Modeling and Simulation*, BRIMS Society, Charleston, SC, pp. 274–281. Suffolk, VA: BRIMS Committee.
- Kettler, B. and Lautenschlager, J. (2017) Expeditionary modeling for population-centric operations in megacities: Some initial experiments. In S. Schatz and M. Hoffman (eds), *Advances in Cross-Cultural Decision Making*, pp. 3–15. New York: Springer.
- Knudsen, D.C. and Fotheringham, A.S. (1986) Matrix comparison, goodness-of-fit, and spatial interaction modeling. *International Regional Science Review*, 10 (2), 127–147.
- Kocabas, V. and Dragićević, S. (2009) Agent-based model validation using Bayesian networks and vector spatial data. *Environment and Planning B*, 36 (5), 787–801.
- Kornhauser, D., Wilensky, U. and Rand, D. (2009) Design guidelines for agent based model visualization. *Journal of Artificial Societies and Social Simulation*, 12(2), 1.
- Kravari, K. and N. Bassiliades (2015) A survey of agent platforms. *Journal of Artificial Societies and Social Simulation*, 18 (1), 11.
- Kronholm, K. and Birkeland, K.W. (2005) Integrating spatial patterns into a snow avalanche cellular automata model. *Geophysical Research Letters*, 32 (19), L19504.1–L19504.4.
- Krugman, P. R. (1996) *The Self-Organizing Economy*. Oxford: Blackwell Publishers.
- Krzysztof, K., Dzwinel, W. and Yuen, D.A. (2005) Nonlinear development of bacterial colony modelled with cellular automata and agent objects. *International Journal of Modern Physics C*, 14 (10), 1385– 1404.
- Laird, J.E. (2012) *The Soar Cognitive Architecture*. Cambridge, MA: MIT Press.
- Lambiotte, R., Blondel, V.D., de Kerchove, C., Huens, E., Prieur, C., Smoreda, Z. and Dooren P. (2008) Geographical dispersal of mobile communication networks. *Physica A*, 387 (21), 5317–5325.
- Landis, J.D. (2001) CUF, CUFII, and CURBA: A family of spatially explicit urban growth and land-use policy simulation models. In R.K. Brail and R.E. Klosterman (eds), *Planning Support Systems: Integrating Geographic Information Systems, Models and Visualisation Tools*, pp. 157–200. Redlands, CA: ESRI Press.
- Landsat (2018) Landsat 8. <https://landsat.gsfc.nasa.gov/> (accessed on 3 August 2018).
- Langran, G. (1992) *Time in Geographic Information Systems*. London: Taylor & Francis.
- Łatek, M.M., Mussavi Rizi, S.M., Crooks, A.T. and Fraser, M. (2012) Social simulations for border security. In *Proceedings of the 2012 European Intelligence and Security Informatics Conference*, Odense, Denmark, pp. 340–345. Los Alamitos, CA: IEEE Computer Society.
- Laver, M. and Sergenti, E. (2012) *Party Competition: An Agent-Based Model*. Princeton, NJ: Princeton University Press.

REFERENCES

- Lazer, D., Pentland, A., Adamic, L., Aral, S., Barabási, A., Brewer, D., Christakis, N.A., Contractor, N., Fowler, J., Gutmann, M., Jebara, T., King, G., Macy, M., Roy, D. and Van Alstyne, M. (2009) Computational social science. *Science*, 323 (5915), 721–723.
- Le Billon, P. (2001) The political ecology of war: Natural resources and armed conflicts. *Political Geography*, 20 (5), 561–584.
- Le Page, C., Bobo, K.S., Kamgaing, T.O.W., Ngahane, B.F. and Waltert, M. (2015) Interactive simulations with a stylized scale model to codesign with villagers an agent-based model of bushmeat hunting in the periphery of Korup National Park (Cameroon). *Journal of Artificial Societies and Social Simulation*, 18 (1), 8.
- Leavesley, G.H., Markstrom, S.L., Brewer, M.S. and Viger, R.J. (1996) The Modular Modeling System (MMS) – the physical process modeling component of a database-centered decision support system for water and power management. In W. Chow, R. W. Brocksen and J. Wisniewski (eds), *Clean Water: Factors That Influence Its Availability, Quality and Its Use*, pp. 303–311. New York: Springer.
- Lee, D.B. (1973) Requiem for large-scale models. *Journal of the American Institute of Planners*, 39 (3), 163–178.
- Lee, J.-S., Filatova, T., Ligmann-Zielinska, A., Hassani-Mahmooei, B., Stonedahl, F., Lorscheid, I., Voinov, A., Polhill, G., Sun, Z. and Parker, D.C. (2015) The complexities of agent-based modelling output analysis. *Journal of Artificial Societies and Social Simulation*, 18 (4), 4.
- Li, J. and Wilensky U. (2009) NetLogo Sugarscape 3 Wealth Distribution Model. <http://ccl.northwestern.edu/netlogo/models/Sugarscape3WealthDistribution> (accessed on 3 August 2018).
- Lim, K., Deadman, P.J., Moran, E., Brondizio, E. and McCracken, S. (2002) Agent-based simulations of household decision making and land use change near Altamira, Brazil. In H.R. Gimblett (ed.), *Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes*, pp. 277–310. Oxford: Oxford University Press.
- Liu, B. (2012) Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5 (1), 1–167.
- Lokuge, P. and Alahakoon, D. (2004) BDI agents using neural network and adaptive neuro fuzzy inference for intelligent panning in container terminals. In *Neural Information Processing: 11th International Conference, ICONIP 2004*, Calcutta, India, pp. 941–946. Berlin: Springer.
- Longley, P.A. and Batty M. (2003) Prologue: Advanced spatial analysis: Extending GIS. In P.A. Longley and M. Batty (eds), *Advanced Spatial Analysis: The CASA Book of GIS*, pp. 1–17. Redlands, CA: ESRI Press.
- Longley, P.A., Goodchild, M.F., Maguire, D.J. and Rhind D.W. (2005) *Geographical Information Systems and Science* (2nd edn). Hoboken, NJ: John Wiley & Sons.
- Longley, P.A., Goodchild, M.F., Maguire, D.J. and Rhind D.W. (2010) *Geographical Information Systems and Science* (3rd edn). Hoboken, NJ: John Wiley & Sons.

REFERENCES

- Lowry, I.S. (1965) A short course in model design. *Journal of the American Institute of Planners*, 31 (2), 158–165.
- Luke, S. (2015) Multiagent simulation and the MASON Library. Technical report, George Mason University, Fairfax, VA.
- Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K. and Balan G. (2005) MASON: A multi-agent simulation environment. *Simulation*, 81(7), 517–527.
- Lunden, I. (2015) 6.1B smartphone users globally by 2020, overtaking basic fixed phone subscriptions. <https://techcrunch.com/2015/06/02/6-1b-smartphone-users-globally-by-2020-overtaking-basic-fixed-phone-subscriptions/#.brp6v1:RPIH> (accessed on 3 August 2018).
- Lyttinen, S.L. and Railsback, S.F. (2012) The evolution of agent-based simulation platforms: A review of NetLogo 5.0 and ReLogo. In R.M. Bichler, S. Blachfellner and W. Hofkirchner (eds), *Proceedings of the Fourth International Symposium on Agent-Based Modeling and Simulation (21st European Meeting on Cybernetics and Systems Research)*, Vienna, Austria.
- MacCarron, P., Kaski, K. and Dunbar, R. (2016) Calling Dunbar's numbers. *Social Networks*, 47, 151–155.
- Macal, C.M. and North, M.J. (2005) Tutorial on agent-based modelling and simulation. In M.E. Euhl, N.M. Steiger, F.B. Armstrong and J.A. Joines (eds), *Proceedings of the 2005 Winter Simulation Conference*, Orlando, FL, pp. 2–15. Piscataway, NJ: IEEE.
- Macatulad, E.G. (2014) 3DGIS-based multi-agent geosimulation and visualization of building evacuation using GAMA platform. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40, 87–91.
- Mackenzie, D. and Tzar, J. (2002) The science of surprise: can complexity theory help us understand the real consequences of a convoluted event like September 11? *Discover Magazine*, 23, 59–63.
- Macy, M.W. and Willer, R. (2002) From factors to factors: Computational sociology and agent-based modeling. *Annual Review of Sociology*, 28 (1), 143–166.
- Magliocca, N.R. and Ellis, E.C. (2013) Using pattern-oriented modeling (POM) to cope with uncertainty in multi-scale agent-based models of land change. *Transactions in GIS*, 17 (6), 883–900.
- Magliocca, N., Safirova, E., McConnell, V. and Walls, M. (2011) An economic agent-based model of coupled housing and land markets (CHALMS). *Computers, Environment and Urban Systems*, 35, 183–191.
- Maguire, D.J. (1995) Implementing spatial analysis and GIS applications for business and service planning. In P.A. Longley and G. Clarke (eds), *GIS for Business and Service Planning*, pp. 171–191. Cambridge: GeoInformation International.
- Maguire, D.J. (2005) Towards a GIS platform for spatial analysis and modelling. In D.J. Maguire, M. Batty and M.F. Goodchild (eds), *GIS, Spatial Analysis and Modelling*, pp. 19–39. Redlands, CA: ESRI Press.

REFERENCES

- Mahabir, R., Croitoru, A., Crooks, A.T., Agouris, P. and Stefanidis, A. (2018) A critical review of high and very high resolution remote sensing approaches for detecting and mapping slums: Trends, challenges and emerging opportunities. *Urban Science*, 2(1), 8.
- Mahdavi Adrestani, B., O'Sullivan, D. and Davis, P. (2018) A multi-scaled agent-based model of residential segregation applied to a real metropolitan area. *Computers, Environment and Urban Systems*, 69, 1–16.
- Malleson, N. (2010) Agent-based modelling of burglary. Ph.D. thesis, School of Geography, University of Leeds.
- Malleson, N. and Andresen, M.A. (2015) The impact of using social media data in crime rate calculations: Shifting hot spots and changing spatial patterns. *Cartography and Geographic Information Science*, 42 (2), 112–121.
- Malleson, N. and Andresen, M.A. (2016) Exploring the impact of ambient population measures on London crime hotspots. *Journal of Criminal Justice*, 46, 52–63.
- Malleson, N. and Birkin, M. (2012) Estimating individual behaviour from massive social data for an urban agent-based model. In A. Koch and P. Mandl (eds), *Modeling Social Phenomena in Spatial Context*, pp. 23–30. Berlin: Lit Verlag.
- Malleson, N., Heppenstall, A. and See, L. (2010) Crime reduction through simulation: An agent-based model of burglary. *Computers, Environment and Urban Systems*, 34 (3), 236–250.
- Malleson, N., Heppenstall, A., See, L. and Evans, A. (2013) Using an agent-based crime simulation to predict the effects of urban regeneration on individual household burglary risk. *Environment and Planning B*, 40 (3), 405–426.
- Mandelbrot, B.B. (1983) *The Fractal Geometry of Nature*. New York: W.H. Freeman.
- Manley, E. (2014) Modelling driver behaviour to predict urban road traffic dynamics. Ph.D. thesis, University College London.
- Manley, E., Addison, J.D. and Cheng, T. (2015a) Shortest path or anchor-based route choice: A large-scale empirical analysis of minicab routing in London. *Journal of Transport Geography*, 43, 123–139.
- Manley, E. and Cheng, T. (2018) Exploring the role of spatial cognition in predicting urban traffic flow through agent-based modelling. *Transportation Research Part A: Policy and Practice*, 109, 14–23.
- Manley, E., Cheng, T. and Emmonds, A. (2011) Understanding route choice using agent-based simulation. In *Proceedings of the 11th International Conference on Geocomputation*, University College London, pp. 54–59.
- Manley, E., Cheng, T., Penn, A. and Emmonds, A. (2014) A framework for simulating large-scale complex urban traffic dynamics through hybrid agent-based modelling. *Computers, Environment and Urban Systems*, 44, 27–36.
- Manley, E. and Dennett, A. (2018) New forms of data for measuring urban activity and interaction in developing countries. *Applied Spatial Analysis and Policy*, doi: 10.1007/s12061-018-9264-8.

REFERENCES

- Manley, E., Orr, S. and Cheng, T. (2015b) A heuristic model of bounded route choice in urban areas. *Transportation Research Part C: Emerging Technologies*, 56, 195–209.
- Manley, E., Zhong, C. and Batty, M. (2018) Spatiotemporal variation in travel regularity through transit user profiling. *Transportation*, 45 (3), 703–732.
- Manson, S.M. (2001) Simplifying complexity: A review of complexity theory. *Geoforum*, 32 (3), 405–414.
- Manson, S.M. (2007) Challenges to evaluating models of geographic complexity. *Environment and Planning B*, 34 (2), 245–260.
- Martínez-Miranda, J. and Aldea, A. (2005) Emotions in human and artificial intelligence. *Computers in Human Behavior*, 21, 323–341.
- Martinez-Moyano, I.J. and Macal, C.M. (2016) A primer for hybrid modeling and simulation. In T.M.K. Roeder, P.I. Frazier, R. Szechtmann, E. Zhou, T. Huschka and S.E. Chick (eds), *Proceedings of the 2016 Winter Simulation Conference*, Arlington, VA, pp. 133–147. Piscataway, NJ: IEEE.
- Maslow, A.H. (1943) A theory of human motivation. *Psychological Review*, 50 (4), 370–396.
- MASON (2016) Multi Agent Simulation of Neighbourhood. Available at <http://cs.gmu.edu/~eclab/projects/mason/> (accessed on 12 July 2016).
- Massive (2017) Film Gallery. Available at <http://www.massivesoftware.com/gallery.html> (accessed on 30 March 2017).
- Mazur, M. and Manley, E. (2016) Exploratory models in a time of big data. *Interdisciplinary Science Reviews*, 41 (4), 366–382.
- McKenna, B. (2013) What does a petabyte look like? <https://www.computerweekly.com/feature/What-does-a-petabyte-look-like> (accessed on 3 August 2018).
- Mearian, L. (2011) World's data will grow by 50X in next decade, IDC study predicts. *Computer World*, 28 June.
- Meier, P. (2015) *Digital Humanitarians: How Big Data Is Changing the Face of Humanitarian Response*. Boca Raton, FL: CRC Press.
- Mena, C.F., Walsh, S.J., Frizzelle, B.G., Xiaozheng, Y. and Malanson G.P. (2011) Land use change on household farms in the Ecuadorian Amazon: Design and implementation of an agent-based model. *Applied Geography*, 31(1), 210–222.
- Meyfroidt, P. (2013) Environmental cognitions, land change, and social–ecological feedbacks: An overview. *Journal of Land Use Science*, 8 (3), 341–367.
- Milgram, S. (1967) The small-world problem. *Psychology Today*, 1 (1), 61–67.
- Miller, J.H. and Page, S.E. (2007) *Complex Adaptive Systems*. Princeton, NJ: Princeton University Press.
- Minar, N., Burkhart, R., Langton, C. and Askenazi, M. (1996) The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations, Santa Fe Institute (SFI Working Paper: 1996-06-042), Santa Fe, NM. Available at <https://www.santafe.edu/research/results/working-papers/>.

REFERENCES

- Monroe, K.R. (2001) Paradigm shift: From rational choice to perspective. *International Political Science Review*, 22 (2), 151–172.
- Moss, S. (2008) Alternative approaches to the empirical validation of agent-based models. *Journal of Artificial Societies and Social Simulation*, 11 (1), 5.
- Müller, B., Bohn, F., Dreßler, G., Groeneveld, J., Klassert, C., Martin, R., Schlüter, M., Schulze, J., Weise, H. and Schwarz, N. (2013) Describing human decisions in agent-based models – ODD + D, an extension of the ODD protocol. *Environmental Modelling & Software*, 48, 37–48.
- Mysore, V., Narzisi, G. and Mishra, B. (2006) Agent modelling of a sarin attack in Manhattan. In N.R. Jennings, M. Tambe, T. Ishida and S.D. Ramchurn (eds), *First International Workshop on Agent Technology for Disaster Management*, Future University, Hakodate, Japan, pp. 108–115.
- Nagel, K. (2003) Traffic networks. In S. Bornholdt and H. Schuster (eds), *Handbook of Graphs and Networks: From the Genome to the Internet*, pp. 248–272. Weinheim: Wiley-VCH.
- Nagel, K., Stretz, P., Pieck, M., Donnelly, R. and Barrett C.L. (1997) TRANSIMS Traffic Flow Characteristics. arXiv:adap-org/971003.
- Nakajima, T. (1977) Application de la théorie de l'automate à la simulation de l'évolution de l'espace urbain. *Congrès sur la Méthodologie de l'Aménagement et du Développement, Association Canadienne-Française pour l'Avancement des Sciences et Comité de Coordination des Centres de Recherches en Aménagement, Développement et Planification (CRADEP)*, Montreal, Canada, pp. 154–160.
- Namatame, A. and Chen, S.-H. (2016) *Agent Based Modelling and Network Dynamics*. Oxford: Oxford University Press.
- Narain, A. (2018) Economic impact of geospatial industry surges to US\$ 2210.7 billion. <https://www.geospatialworld.net/blogs/economic-impact-of-geospatial-industry/> (accessed on 3 August 2018).
- NatureServe Vista (2016) Decision Support for Better Planning. Available at http://www.natureserve.org/conservation-tools/nature_serve-vista (accessed on 12 April 2016).
- Neji, M. and Ammar, M.B. (2007) Agent-based collaborative affective e-learning framework. *Electronic Journal of e-Learning*, 5 (2), 123–134.
- Nguyen, N.-A.T., Zucker, J.-D., Nguyen, H.-D., Drogoul, A. and Vo, D.A. (2011) A hybrid macro-micro pedestrians evacuation model to speed up simulation in road networks. In F. Dechesne, H. Hattori, A. ter Mors, J.M. Such, D. Weyns and F. Dignum (eds), *Advanced Agent Technology: AAMAS 2011 Workshops*, pp. 371–383. Berlin: Springer.
- Nielsen, F.A. (2011) A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In M. Rowe, M. Stankovic and M. Hardey (eds), *Proceedings of the 1st Workshop on Making Sense of Microposts (#MSM2011): Big Things Come in Small Packages*, Heraklion, Greece, pp. 93–98.

REFERENCES

- Nikolai, C. and Madey, G. (2009) Tools of the trade: A survey of various agent based modeling platforms. *Journal of Artificial Societies and Social Simulation*, 12 (2), 2.
- Norris, C., McCahill, M. and Wood D. (2004) The growth of CCTV: A global perspective on the international diffusion of video surveillance in publicly accessible space. *Surveillance & Society*, 2, 110–135.
- North, M.J., Collier, N.T., Ozik, J., Tatara, E.R., Macal, C.M., Bragen, M. and Sydelko P. (2013) Complex adaptive systems modeling with Repast Simphony. *Complex Adaptive Systems Modeling*, 1, 3.
- North, M.J., Collier, N.T. and Vos, J.R. (2006) Experiences creating three implementations of the Repast agent modelling toolkit. *ACM Transactions on Modeling and Computer Simulation*, 16 (1), 1–25.
- North, M.J., Howe, T.R., Collier, N.T. and Vos, J.R. (2005a) The Repast Simphony Development Environment. In C.M. Macal, M.J. North and D. Sallach (eds.), *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, Chicago, IL, pp. 159–166.
- North, M.J., Howe, T.R., Collier, N.T. and Vos, J.R. (2005b) The Repast Simphony Runtime System. In C.M. Macal, M.J. North and D. Sallach (eds.) *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, Chicago, IL, pp. 151–158.
- North, M.J. and Macal, C.M. (2007) *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford: Oxford University Press.
- North, M.J., Macal, C.M., St Aubin, J., Thimmapuram, P., Bragen, M., Hahn, J., Kerr, J., Brigham, N., Lacy, M.E. and Hampton, D. (2010) Multiscale agent-based consumer market modeling. *Complexity*, 15, 37–47.
- North, M.J., Murphy, J.T., Sydelko, P., Martinez-Moyano, I., Sallach, D.L. and Macal, C.M. (2015) Integrated modeling of conflict and energy. In L. Yilmaz, W.K.V. Chan, I. Moon, T.M.K. Roeder, C. Macal and M.D. Rossetti (eds), *Winter Simulation Conference*, Huntington Beach, CA, pp. 2499–2510. Piscataway, NJ: IEEE.
- Nowicki, S.M., Payne, A., Larour, E., Seroussi, H., Goelzer, H., Lipscomb, W., Gregory, J., Abe-Ouchi, A. and Shepherd, A. (2016) Ice Sheet Model Intercomparison Project (ISMIP6) contribution to CMIP6. *Geoscientific Model Development*, 9 (12), 4521–4545.
- Oldham, M. (2016) To big wing, or not to big wing, now an answer. In N. Osman and C. Sierra (eds), *Autonomous Agents and Multiagent Systems*, pp. 73–89. Cham: Springer.
- Oldham, M. (2017) Market fluctuations explained by dividends and investor networks. *Advances in Complex Systems*, 20 (08), 1750007.
- Openshaw, S. (1984) *The Modifiable Areal Unit Problem*, Volume 38 of *Concepts and Techniques in Modern Geography (CATMOG)*. Norwich: Geo Books.

REFERENCES

- Orcutt, G. H. (1957) A new type of socio-economic system. *Review of Economics and Statistics*, 39 (2), 116–123.
- Ord, J.K. and Getis, A. (1995) Local spatial autocorrelation statistics: Distributional issues and an application. *Geographical Analysis*, 27 (4), 286–306.
- Orellana, D. and Wachowicz, M. (2011) Exploring patterns of movement suspension in pedestrian mobility. *Geographical Analysis*, 43, 241–260.
- Ormerod, P. (1998) *Butterfly Economics: A New General Theory of Economic and Social Behaviour*. London: Faber and Faber.
- Orsi, F. and Geneletti, D. (2016) Transportation as a protected area management tool: An agent-based model to assess the effect of travel mode choices on hiker movements. *Computers, Environment and Urban Systems*, 60, 12–22.
- O’Sullivan, D. (2001) Exploring spatial process dynamics using irregular cellular automaton models. *Geographical Analysis*, 33 (1), 1–18.
- O’Sullivan, D. (2004) Complexity science and human geography. *Transactions of the Institute of British Geographers*, 29 (3), 282–295.
- O’Sullivan, D. (2009) Changing neighborhoods – neighborhoods changing: A framework for spatially explicit agent-based models of social systems. *Sociological Methods and Research*, 37 (4), 498–530.
- O’Sullivan, D., Evans, T., Manson, S.M., Metcalf, S., Ligmann-Zielinska, A. and Bone C. (2015) Strategic directions for agent-based modeling: Avoiding the YAAWN syndrome. *Journal of Land Use Science*, 11 (2), 177–187.
- O’Sullivan, D. and Haklay, M. (2000) Agent-based models and individualism: Is the world agent-based? *Environment and Planning A*, 32 (8), 1409–1425.
- O’Sullivan, D., Millington, J., Perry, G. and Wainwright, J. (2012) Agent-based models – because they’re worth it? In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 109–123. Dordrecht: Springer.
- O’Sullivan, D. and Perry, G.L. (2013) *Spatial Simulation: Exploring Pattern and Process*. Chichester: John Wiley & Sons.
- O’Sullivan, D. and Unwin, D. (2010) *Geographic Information Analysis* (2nd edn). Hoboken, NJ: John Wiley & Sons.
- Ozik, J., Collier, N., Combs, T., Macal, C.M. and North M. (2015) Repast Simphony statecharts. *Journal of Artificial Societies and Social Simulation*, 18, (3), 11.
- Ozik, J., Collier, N.T., Murphy, J.T. and North M.J. (2013) The ReLogo agent-based modeling language. In R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill and M.E. Kuhl, (eds), *Proceedings of the 2013 Winter Simulation Conference*, Washington, DC, pp. 1560–1568. Piscataway, NJ: IEEE.
- Panteras, G., Lu, X., Croitoru, A., Crooks, A.T. and Stefanidis, A. (2016) Accuracy of user-contributed image tagging in Flickr: A natural disaster case study. In A. Gruzd, J. Jacobson, P. Mai, E. Ruppert and D. Murthy (eds), *Proceedings of the 7th International Conference on Social Media and Society*. New York: ACM.

REFERENCES

- Panteras, G., Wise, S., Lu, X., Croitoru, A., Crooks, A.T. and Stefanidis, A. (2015) Triangulating social multimedia content for event localization using Flickr and Twitter. *Transactions in GIS*, 19 (5), 694–715.
- Park, B.H., Allen, M.R., White, D., Weber, E., Murphy, J.T., North, M.J. and Sydelko, P. (2017) MIRAGE: A framework for data-driven collaborative high-resolution simulation. In D.A. Griffith, Y. Chun and D.J. Dean (eds), *Advances in Geocomputation*, pp. 395–403. New York: Springer.
- Parker, D.C., Berger, T. and Manson S.M. (eds) (2001) *Meeting the Challenge of Complexity: Proceedings of an International Workshop on Agent-Based Models of Land-Use and Land-Cover Change*, Irvine, CA.
- Parker, D.C., Brown, D.G., Filatova, T., Riolo, R., Robinson, D.T. and Sun, S. (2012) Do land markets matter? A modeling ontology and experimental design to test the effects of land markets for an agent-based model of ex-urban residential land-use change. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 525–542. Dordrecht: Springer.
- Parker, D.C., Entwistle, B., Rindfuss, R., Vanwey, L., Manson, S.M., Moran, E., An, L., Deadman, P.J., Evans, T., Linderman, M., Mussavi Rizi, M. S. and Malanson G. (2008) Case studies, cross-site comparisons, and the challenge of generalization: Comparing agent-based models of land-use change in frontier regions. *Journal of Land Use Science*, 3 (1), 41–72.
- Parker, D.C., Manson, S.M., Janssen, M.A., Hoffmann, M.J. and Deadman, P. (2003) Multi-agent systems for the simulation of land-use and land-cover change: A review. *Annals of the Association of American Geographers*, 93 (2), 314–337.
- Parrett, C.M., Crooks, A.T. and Pike, T. (2018) The future of GEOINT: Data science will not be enough. In *The State and Future of GEOINT 2018 Report*, pp. 12–15. Herndon, VA: United States Geospatial Intelligence Foundation.
- Patel, A., Crooks, A.T. and Koizumi N. (2012) Slumulation: An agent-based modeling approach to slum formations. *Journal of Artificial Societies and Social Simulation*, 15 (4), 2.
- Patel, A., Crooks, A.T. and Koizumi N. (2018) Spatial agent-based modeling to explore slum formation dynamics in Ahmedabad, India. In J.-C. Thill and S. Dragičević (eds), *Geocomputational Analysis and Modeling of Regional Systems*, pp. 121–141. New York: Springer.
- Pavlov, P. I. (1927) *Conditioned Reflexes*. London: Oxford University Press.
- Peuquet, D. (2002) *Representations of Space and Time*. New York: Guilford Press.
- Peuquet, D.J. (2005) Time in GIS and geographical databases. In P.A. Longley, M.F. Goodchild, D.J. Maguire and D.W. Rhind (eds), *Geographical Information Systems: Principles, Techniques, Management and Applications* (abridged edition), pp. 91–103. Hoboken, NJ: John Wiley & Sons.
- Pheasant, S. and Haslegrave, C.M. (2006) *Bodyspace: Anthropometry, Ergonomics and the Design of Work* (3rd edn). London: Taylor & Francis.

REFERENCES

- Pint, B., Crooks, A.T. and Geller A. (2010) Exploring the emergence of organized crime in Rio de Janeiro: An agent-based modeling approach. In G.P. Dimuro, A.C. da Rocha Costa, J.S. Sichman, P. Tedesco, D.F. Adamatti, J. Balsa and L. Antunes (eds), *Advances in Social Simulation: Proceedings of the 2nd Brazilian Workshop on Social Simulation*, Sao Bernardo do Campo, Brazil, pp. 7–14. Piscataway, NJ: IEEE.
- Pires, B. and Crooks, A.T. (2016) The geography of conflict diamonds: The case of Sierra Leone. In K.S. Xu, D. Reitter, D. Lee and N. Osgood (eds), *Social, Cultural, and Behavioral Modeling: 9th International Conference*, Washington, DC, pp. 335–345. Cham: Springer.
- Pires, B. and Crooks, A.T. (2017) Modeling the emergence of riots: A geosimulation approach. *Computers, Environment and Urban Systems*, 61, 66–80.
- Polhill, G.J., Parker, D.C., Gotts, N.M. and Edmonds, B. (2007) Effects of land markets on competition between innovators and imitators in land use: Results from FEARLUS-ELMM. In B. Edmonds, C. Hernandez and K.G. Troitzsch (eds), *Social Simulation: Technologies, Advances and New Discoveries*, pp. 81–97. Hershey, PA: IGI Global.
- Pontius, R.G., Boersma, W., Castella, J.C., Clarke, K.C., Nijs, T., Dietzel, C.K., Duan, Z., Fotsing, E., Goldstein, N.C., Kok, K., Koomen, E., Lippitt, C.D., McConnell, W., Sood, A.M., Pijanowski, B., Pithadia, S., Sweeney, S., Trung, T.N., Veldkamp, A. and Verburg P.H. (2008) Comparing the input, output, and validation maps for several models of land change. *Annals of Regional Science*, 42 (1), 11–37.
- Power, C. (2009) A spatial agent-based model of N-Person prisoner's dilemma cooperation in a socio-geographic community. *Journal of Artificial Societies and Social Simulation*, 12 (1), 8.
- Pumain, D. (2012) Multi-agent system modelling for urban systems: The series of SIMPOP models. In A.J. Heppenstall, A. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 721–738. Dordrecht: Springer.
- Radzikowski, J., Stefanidis, A., Jacobsen, K.H., Croitoru, A., Crooks, A.T. and Delamater, P.L. (2016) The measles vaccination narrative in Twitter: A quantitative analysis. *JMIR Public Health and Surveillance*, 2, e1.
- Rahmandad, H. and Sterman, J. (2008) Heterogeneity and network structure in the dynamics of diffusion: Comparing agent-based and differential equation models. *Management Science*, 54(5), 998–1014.
- Rai, S. and Hu, X. (2013) Behavior pattern detection for data assimilation in agent-based simulation of smart environments. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, Volume 2, pp. 171–178. Los Alamitos, CA: IEEE Computer Society.
- Railsback, S.F. and Grimm, V. (2011) *Agent-Based and Individual-Based Modeling: A Practical Introduction*. Princeton, NJ: Princeton University Press.

REFERENCES

- Railsback, S.F. and Harvey, B.C. (2002) Analysis of habitat selection rules using an individual-based model. *Ecology*, 83 (7), 1817–1830.
- Railsback, S.F., Harvey, B.C., Kupferberg, S.J., Lang, M.M., McBain, S. and Welsh, H.H., Jr (2015) Modeling potential river management conflicts between frogs and salmonids. *Canadian Journal of Fisheries and Aquatic Sciences*, 73, 773–784.
- Railsback, S.F. and Johnson, M.D. (2014) Effects of land use on bird populations and pest control services on coffee farms. *Proceedings of the National Academy of Sciences of the United States of America*, 111(16), 6109–6114.
- Railsback, S.F., Lytinen, S.L. and Jackson, S.K. (2006) Agent-based simulation platforms: Review and development recommendations. *Simulation*, 82(9), 609–623.
- Raney, B., Çetin, N., Völlmy, A., Vrtic, M., Axhausen, K. and Nagel, K. (2003) An agent-based microsimulation model of Swiss travel: First results. *Networks and Spatial Economics*, 3 (1), 23–42.
- Rao, A.S. and Georgeff, M.P. (1991) Modeling rational agents within a BDI-architecture. In J. Allen, R. Fikes and E. Sandewall (eds), *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, San Mateo, CA, pp. 473–484. San Francisco, CA: Morgan Kaufmann.
- Rao, A.S. and Georgeff, M.P. (1995) BDI agents: From theory to practice. In V. Lesser (ed.), *Proceedings of the First International Conference on Multiagent Systems*, San Francisco, CA, pp. 312–319. Palo Alto, CA: AAAI.
- Rapoport, A. (1957) Contribution to the theory of random and biased nets. *Bulletin of Mathematical Biology*, 19 (4), 257–277.
- Ravenstein, E.G. (1885) The laws of migration. *Journal of the Royal Statistical Society*, 48, 167–227.
- Reardon, S.F. and O’Sullivan, D. (2004) Measures of spatial segregation. *Sociological Methodology*, 34 (1), 121–162.
- Reilly, W.J. (1929) Methods for the study of retail relationships. Bulletin Number 2944, University of Texas, Austin, TX.
- Repast (2016) Recursive Porous Agent Simulation Toolkit. Available at <http://repast.sourceforge.net/> (accessed on 7 October 2016).
- Rey, S.J. and Anselin, L. (2010) PySAL: A Python library of spatial analytical methods. In M.M. Fischer and A. Getis (eds), *Handbook of Applied Spatial Analysis*, pp. 175–193. New York: Springer.
- Reynolds, C. (1987) Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21, 25–34.
- Ripley, B.D. (1977) Modelling spatial patterns. *Journal of the Royal Statistical Society B*, 39 (2), 172–212.
- Robertson, D.A. (2005) Agent-based modeling toolkits. *Academy of Management Learning and Education*, 4 (4), 525–527.
- Robinson, D.T., Brown, D., Parker, D.C., Schreinemachers, P., Janssen, M.A., Huigen, M., Wittmer, H., Gotts, N., Promburom, P., Irwin, E., Berger, T.,

REFERENCES

- Gatzweiler, F. and Barnaud, C. (2007) Comparison of empirical methods for building agent-based models in land use science. *Journal of Land Use Science*, 2 (1), 31–55.
- Roick, O. and Heuser, S. (2013) Location based social networks – definition, current state of the art and research agenda. *Transactions in GIS*, 17, 763–784.
- Roy, D., Lees, M.H., Palavalli, B., Pfeffer, K. and Sloot M.P. (2014) The emergence of slums: A contemporary view on simulation models. *Environmental Modelling & Software*, 59, 76–90.
- Sapiro, G. (2011) Images everywhere: Looking for models: Technical perspective. *Communications of the ACM*, 54, 108–108.
- Sasaki, Y. and Box, P. (2003) Agent-based verification of von Thünen's location theory. *Journal of Artificial Societies and Social Simulation*, 6(2), 9.
- Schelling, T.C. (1971). Dynamic models of segregation. *Journal of Mathematical Sociology*, 1 (1), 143–186.
- Schelling, T.C. (1978) *Micromotives and Macrobbehavior*. New York: W.W. Norton.
- Schensul, J.J., Gavelty, M.D., Trotter, R.T., Cromley, E.K. and Singer, M. (1999) *Mapping Social Networks, Spatial Data, and Hidden Populations*. Lanham, MD: AltaMira Press.
- Schlüter, M., Baeza, A., Dressler, G., Frank, K., Groeneveld, J., Jager, W., Janssen, M.A., McAllister, R.R., Müller, B., Orach, K. and Schwarz, N. (2017) A framework for mapping and comparing behavioural theories in models of social-ecological systems. *Ecological Economics*, 131, 21–35.
- Schmidt, B. (2000) *The Modelling of Human Behaviour*. Ghent: SCS-Europe.
- Schmidt, B. (2002) The modelling of human behaviour: The PECS reference model. In A. Verbraeck and W. Krug (eds), *Proceedings of the 14th European Simulation Symposium and Exhibition (ESS '02)*, pp. 13–18. Ghent: SCS-Europe.
- Seibel, F. and Thomas, C. (2000) Manifest destiny: Adaptive cargo routing at Southwest Airlines. *Perspectives on Business Innovation*, 4, 27–33.
- Selva, D. and Krejci, D. (2012) A survey and assessment of the capabilities of cubesats for Earth observation. *Acta Astronautica*, 74, 50–68.
- Semboloni, F., Assfalg, J., Armeni, S., Gianassi, R. and Marsoni, F. (2004) CityDev, an interactive multi-agents urban model on the web. *Computers, Environment and Urban Systems*, 28 (1–2), 45–64.
- Shaham, Y. and Benenson, I. (2018) Modeling fire spread in cities with non-flammable construction. *International Journal of Disaster Risk Reduction*, doi: 10.1016/j.ijdrr.2018.03.010.
- Shiflet, A.B. and Shiflet, G.W. (2014) *Introduction to Computational Science: Modeling and Simulation for the Sciences* (2nd edn). Princeton, NJ: Princeton University Press.
- Simmel, G. (1902) The number of members as determining the sociological form of the group. I. *American Journal of Sociology*, 8 (1), 1–46.

REFERENCES

- Simon, H.A. (1955) A behavioral model of rational choice. *Quarterly Journal of Economics*, 69, 99–118.
- Simon, H.A. (1978) Rationality as process and as product of thought. *American Economic Review*, 68 (2), 1–16.
- Simon, H.A. (1996) *The Sciences of the Artificial* (3rd edn). Cambridge, MA: MIT Press.
- SimTable (2017) SimTable Home Page. <https://www.simtable.com> (accessed on 3 August 2018).
- Singh, K. (2005) An abstract mathematical framework for semantic modeling and simulation of urban crime patterns. PhD thesis, University of Delhi.
- Skinner, B. (1953) *The Behavior of Organisms*. New York: Appleton-Century-Crofts.
- Smith, D.M. (2012) Simulating spatial health inequalities. In A.J. Heppenstall, A.T. Crooks, L.M. See and M. Batty (eds), *Agent-Based Models of Geographical Systems*, pp. 499–510. Dordrecht: Springer.
- Smith, J.M. (1991) State-dependent queueing models in emergency evacuation networks. *Transportation Research Part B: Methodological*, 25, 373–389.
- Smith, N. (1979) Toward a theory of gentrification: A back to the city movement by capital not people. *Journal of the American Planning Association*, 45 (4), 538–548.
- Stefanidis, A., Cotnoir, A., Croitoru, A., Crooks, A.T., Radzikowski, J. and Rice, M. (2013a) Demarcating new boundaries: Mapping virtual polycentric communities through social media content. *Cartography and Geographic Information Science*, 40, 116–129.
- Stefanidis, A., Crooks, A.T. and Radzikowski, J. (2013b) Harvesting Ambient Geospatial Information from Social Media Feeds. *GeoJournal*, 78 (2), 319–338.
- Stefanidis, A., Vraga, E., Lamprianidis, G., Radzikowski, J., Delamater, P.L., Jacobsen, K.H., Pfoser, D., Croitoru, A. and Crooks, A.T. (2017) Zika in Twitter: Temporal variations of locations, actors, and concepts. *JMIR Public Health and Surveillance*, 3 (2), e22.
- Sterman, J.D. (2000) *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Boston: McGraw-Hill.
- Stouffer, S.S. (1940) Intervening opportunities: A theory relating mobility and distance. *American Sociological Review*, 5 (6), 845–867.
- Sturley, C., Newing, A. and Heppenstall, H. (2018) Evaluating the potential of agent-based modelling to capture consumer grocery retail store choice behaviours. *International Review of Retail, Distribution and Consumer Research*, 28 (1), 27–46.
- Sugiyama, Y., Fukui, M., Kikuchi, M., Hasebe, K., Nakayama, A., Nishinari, K., Tadaki, S.-i. and Yukawa, S. (2008) Traffic jams without bottlenecks – experimental evidence for the physical mechanism of the formation of a jam. *New Journal of Physics*, 10 (3), 033001.

REFERENCES

- Sullivan, K., Coletti, M. and Luke, S. (2010) GeoMason: GeoSpatial Support for MASON. Technical report, Department of Computer Science, George Mason University, Fairfax, VA.
- Swarm (2016) Swarm: A platform for agent-based models. Available at <http://www.swarm.org/> (accessed on 7 September 2016).
- Swartz, M. (2015) Evaluating impacts of future development on commuters: A model of transportation and development for Reston, VA. Master's Project, George Mason University, Fairfax, VA.
- Taillandier, P. (2014) Traffic Simulation with the GAMA Platform. In *Eighth International Workshop on Agents in Traffic and Transportation, International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014)*, Paris, France.
- Taillandier, P., Banos, A., Drogoul, A., Gaudou, B., Marilleau, N. and Truong Q.C. (2016) Simulating urban growth with raster and vector models: A case study for the city of Can Tho, Vietnam. In N. Osman and C. Sierra (eds), *Autonomous Agents and Multiagent Systems*, pp. 154–171. Cham: Springer.
- Taillandier, P., Therond, O. and Gaudou B. (2012) A new BDI agent architecture based on the belief theory. Application to the modelling of cropping plan decision-making. In R. Seppelt, A.A. Voinov, S. Lange and D. Bankamp (eds), *Managing Resources of a Limited Planet: Pathways and Visions under Uncertainty: Proceedings of the Sixth Biannual Meeting of the International Environmental Modelling and Software Society*, pp. 2471–2479. Leipzig: International Environmental Modelling and Software Society.
- Tajfel, H. and Turner, J.C. (1979) An integrative theory of intergroup conflict. In W.G. Austin and S. Worcher (eds), *The Social Psychology of Intergroup Relations*, pp. 7–24. Monterey, CA: Brooks Cole Publishing.
- Takama, T. and Preston, J. (2008) Forecasting the Effects of road user charge by stochastic agent-based modelling. *Transportation Research Part A: Policy and Practice*, 42 (4), 738–749.
- Taylor, G., Frederiksen, R., Vane, R.R. and Waltz E. (2004) Agent-based simulation of geo-political conflict. In *Nineteenth National Conference on Artificial Intelligence*, pp. 884–891. Menlo Park, CA: AAAI Press.
- Tenkanen, H., Di Minin, E., Heikinheimo, V., Hausmann, A., Herbst, M., Kajala, L. and Toivonen, T. (2017) Instagram, Flickr, or Twitter: Assessing the usability of social media data for visitor monitoring in protected areas. *Scientific Reports*, 7 (1), 17615.
- Tenkanen, H., Saarsalmi, P., Järv, O., Salonen, M. and Toivonen, T. (2016) Health research needs more comprehensive accessibility measures: Integrating time and transport modes from open data. *International Journal of Health Geographics*, 15 (1), 23.
- Thiele, J.C. (2014) R marries NetLogo: Introduction to the RNetLogo package. *Journal of Statistical Software*, 58 (2), 1–41.

REFERENCES

- Thiele, J.C. and Grimm, V. (2010) NetLogo meets R: Linking agent-based models with a toolbox for their analysis. *Environmental Modelling & Software*, 25, 972–974.
- Tobler, W. (1970) A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 46, (2), 234–240.
- Tobler, W.R. (1979) Cellular geography. In S. Gale and G. Olsson (eds), *Philosophy in Geography*, pp. 379–386. Dordrecht: Reidel.
- Tomlin, C.D. (1994) Map algebra: One perspective. *Landscape and Urban Planning*, 30 (1–2), 3–12.
- Torrens, P.M. (2000) How land-use-transportation models work. Working Paper 20, Centre for Advanced Spatial Analysis, University College London.
- Torrens, P.M. (2003) Automata-based models of urban systems. In P. A. Longley and M. Batty (eds), *Advanced Spatial Analysis: The CASA Book of GIS*, pp. 61–81. Redlands, CA: ESRI Press.
- Torrens, P. (2008) Wi-fi geographies. *Annals of the Association of American Geographers*, 98 (1), 59–84.
- Torrens, P. (2010a) Agent-based modeling and the spatial sciences. *Geography Compass*, 4 (5), 428–448.
- Torrens, P.M. (2010b) Geography and computational social science. *GeoJournal*, 75 (2), 133–148.
- Torrens, P.M. (2012) Moving agent-pedestrians through space and time. *Annals of the Association of American Geographers*, 102, 35–66.
- Torrens, P. (2014a) High-fidelity behaviors for model people on model streetscapes. *Annals of GIS*, 20 (3), 139–157.
- Torrens, P.M. (2014b) High-resolution space–time processes for agents at the built–human interface of urban earthquakes. *International Journal of Geographical Information Science*, 28 (5), 964–986.
- Torrens, P.M. and McDaniel, A.W. (2013) Modeling geographic behavior in riotous crowds. *Annals of the Association of American Geographers*, 103, 20–46.
- Torrens, P.M. and Nara, A. (2007) Modelling gentrification dynamics: A hybrid approach. *Computers, Environment and Urban Systems*, 31 (3), 337–361.
- Torres, J.A., Nedel, L.P. and Bordini, R.H. (2003) Using the BDI architecture to produce autonomous characters in virtual worlds. In T. Rist, R.S. Aylett, D. Ballin and J. Rickel (eds), *Intelligent Virtual Agents*, pp. 197–201. Berlin: Springer.
- Tsvetovat, M. and Kouznetsov, A. (2011) *Social Network Analysis for Startups*. Sebastopol, CA: O'Reilly.
- UN-Habitat (2003) *The Challenge of Slums: Global Report on Human Settlements 2003*. London: United Nations Human Settlements Programme.
- Ungerer, M.J. and Goodchild, M.F. (2002) Integrating spatial data analysis and GIS: A new implementation using component object model (COM). *International Journal of Geographic Information Science*, 16 (1), 41–53.

REFERENCES

- Urban, C. (2000) PECS: A reference model for the simulation of multi-agent systems. In R. Suleiman, K.G. Troitzsch and N. Gilbert (eds), *Tools and Techniques for Social Science Simulation*, pp. 83–114. Heidelberg: Physica-Verlag.
- van der Wal, C.N., Formolo, D., Robinson, M.A., Minkov, M. and Bosse, T. (2017) Simulating crowd evacuation with socio-cultural, cognitive, and emotional elements. In J. Mercik (ed.), *Transactions on Computational Collective Intelligence XXVII*, pp. 139–177. Cham: Springer.
- van Vliet, J., Hagen-Zanker, A., Hurkens, J. and van Delden, H. (2013) A fuzzy set approach to assess the predictive accuracy of land use simulations. *Ecological Modelling*, 261–262, 32–42.
- Vatsavai, R.R. and Bhaduri, B. (2013) Geospatial analytics for big spatiotemporal data: Algorithms, applications, and challenges. In *NSF Workshop on Big Data and Extreme-scale Computing*, Charleston, SC.
- Vieweg, S., Hughes, A.L., Starbird, K. and Palen L. (2010) Microblogging during two natural hazards events: What Twitter may contribute to situational awareness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1079–1088. New York: ACM.
- von Thünen, J. H. (1966) *Von Thünen's Isolated State: An English Version of Der Isolierte Staat*. Oxford: Pergamon Press.
- Vos, J.R. and North, M.J. (2004) Repast .NET. In C.M. Macal, D. Sallach and M.J. North (eds), *Proceedings of the Agent 2004 Conference on Social Dynamics: Interaction, Reflexivity and Emergence*, pp. 239–254. Washington, DC: US Department of Energy, Office of Science.
- Wagoum, A.U.K., Seyfried, A. and Holl, S. (2012) Modeling the dynamic route choice of pedestrians to assess the criticality of building evacuation. *Advances in Complex Systems*, 15 (7), 1250029.
- Waldrop, M.M. (2017) News feature: Special agents offer modeling upgrade. *Proceedings of the National Academy of Sciences*, 114 (28), 7176–7179.
- Waldrop, M.M. (2018) Free agents. *Science*, 360 (6385), 144–147.
- Walsh, C.L., Dawson, R.J., Hall, J.W., Barr, S., Batty, M., Bristow, A.L., Carney, S., Dagoumas, A.S., Ford, A.C., Harpham, C. and Tight, M.R. (2011) Assessment of climate change mitigation and adaptation in cities. *Proceedings of the Institution of Civil Engineers – Urban Design and Planning*, 164 (2), 75–84.
- Wang, H., Mostafizi, A., Cramer, L.A., Cox, D. and Park, H. (2016) An agent-based model of a multimodal near-field tsunami evacuation: Decision-making and life safety. *Transportation Research Part C: Emerging Technologies*, 86, 86–100.
- Ward, J.A., Evans, A.J. and Malleson, N.S. (2016) Dynamic calibration of agent-based models using data assimilation. *Royal Society Open Science*, 3, 150703.
- Wasserman, S. and Faust, K. (1994) *Social Network Analysis: Methods and Applications*. New York: Cambridge University Press.

REFERENCES

- Watts, D. and Strogatz, S.H. (1998) Collective dynamics of ‘small-world’ networks. *Nature*, 393 (6684), 440.
- Wegener, M. (2000) Spatial models and GIS. In A.S. Fotheringham and M. Wegener (eds), *Spatial Models and GIS: New Potential and New Models*, pp. 3–20. London: Taylor & Francis.
- Weinberger, S. (2011) Web of war: Can computational social science help to prevent or win wars? *Nature*, 471, 566–568.
- Werker, C. and Brenner, T. (2004) Empirical calibration of simulation models. Papers on Economics and Evolution 0410, Max-Planck-Institut für Ökonomik, Jena.
- Westervelt, J.D. (2002) Geographic information systems and agent-based modeling. In H.R. Gimblett (ed.), *Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes*, pp. 83–104. Oxford: Oxford University Press.
- Westervelt, J. (2004) GRASS roots. In *Proceedings of the FOSS/GRASS Users Conference*, Bangkok, Thailand.
- White, R. and Engelen, G. (1993) Cellular automata and fractal urban form: A cellular modelling approach to the evolution of urban land use patterns. *Environment and Planning A*, 25 (8), 1175–1199.
- Wilensky, U. (1997a) Segregation. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. Available at <http://ccl.northwestern.edu/netlogo/models/Segregation> (accessed on 3 August 2018).
- Wilensky, U. (1997b) Wolf Sheep Predation. <http://ccl.northwestern.edu/netlogo/models/WolfSheepPredation> (accessed on 3 August 2018).
- Wilensky, U. (1997c) Traffic Basic. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. Available at <http://ccl.northwestern.edu/netlogo/models/TrafficBasic> (accessed on 3 August 2018).
- Wilensky, U. (1998) Life. <http://ccl.northwestern.edu/netlogo/models/Life> (accessed on 12 March 2017).
- Wilensky, U. (1999) NetLogo. Center for Connected Learning and Computer-Based Modeling. Northwestern University, Evanston, IL. <http://ccl.northwestern.edu/netlogo> (accessed on 12 March 2017).
- Wilensky, U. (2005a) Wolf Sheep Predation (System Dynamics). [http://ccl.northwestern.edu/netlogo/models/WolfSheepPredation\(SystemDynamics\)](http://ccl.northwestern.edu/netlogo/models/WolfSheepPredation(SystemDynamics)) (accessed on 30 March 2017).
- Wilensky, U. (2005b) Preferential Attachment. <http://ccl.northwestern.edu/netlogo/models/PreferentialAttachment> (accessed on 12 March 2017).

REFERENCES

- Wilensky, U. (2017) Network Import Example.nlogo. <https://github.com/NetLogo/models/blob/master/Code%20Examples/Network%20Import%20Example.nlogo>
- Wilensky, U. and Rand, W. (2015) *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. Cambridge, MA: MIT Press.
- Wilson, A.G. (1974) *Urban and Regional Models in Geography and Planning*. Chichester: John Wiley & Sons.
- Wilson, A.G. (2000) *Complex Spatial Systems: The Modelling Foundations of Urban and Regional Analysis*. Harlow: Pearson Education.
- Windrum, P., Fagiolo, G. and Moneta, A. (2007) Empirical validation of agent-based models: Alternatives and prospects. *Journal of Artificial Societies and Social Simulation*, 10 (2), 8.
- Wise, S. (2014) Using social media content to inform agent-based models for humanitarian crisis response. PhD thesis, George Mason University, Fairfax, VA.
- Wise, S. and Crooks, A.T. (2012) Agent based modelling and GIS for community resource management: Acequia-based agriculture. *Computers, Environment and Urban Systems*, 36 (6), 562–572.
- Wise, S., Crooks, A.T. and Batty, M. (2017) Transportation in agent-based urban modelling. In M. Namazi-Rad, L. Padgham, P. Perez, K. Nagel and A. Bazzan (eds), *Agent Based Modelling of Urban Systems*, pp. 129–148. New York: Springer.
- Wolfram, S. (2002) *A New Kind of Science*. Champaign, IL: Wolfram Media.
- Wood, S.A., Guerry, A.D., Silver, J.M. and Lacayo, M. (2013) Using social media to quantify nature-based tourism and recreation. *Scientific Reports*, 3, 2976.
- Wooldridge, M. and Jennings, N.R. (1995) Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10 (2), 115–152.
- Xie, Y. (1996) A generalized model for cellular urban dynamics. *Geographical Analysis*, 28, 350–373.
- Xie, Y., Batty, M. and Zhao, K. (2007) Simulating emergent urban form: Desakota in China. *Annals of the Association of American Geographers*, 97 (3), 477–495.
- Xie, Y. and Fan, S. (2014) Multi-city sustainable regional urban growth simulation – MSRUGS: A case study along the mid-section of Silk Road of China. *Stochastic Environmental Research and Risk Assessment*, 28, 829–841.
- Yuan, X. and Crooks, A.T. (2017) From cyber space opinion leaders and the spread of anti-vaccine extremism to physical space disease outbreaks. In D. Lee, Y. Lin, N. Osgood and R. Thomson (eds), *Proceedings of the 2017 International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pp. 114–119. New York: Springer.

REFERENCES

- Zhao, Y. and Sadek, A.W. (2012) Large-scale agent-based traffic micro-simulation: Experiences with model refinement, calibration, validation and application. *Procedia Computer Science*, 10, 815–820.
- Zhong, C., Batty, M., Manley, E., Wang, J., Wang, Z., Chen, F. and Schmitt, G. (2016) Variability in regularity: Mining temporal mobility patterns in London, Singapore and Beijing using smart-card data. *PLoS ONE*, 11(2), e0149222.
- Ziemke, D., Nagel, K. and Moeckel, R. (2016) Towards an agent-based, integrated land-use transport modeling system. *Procedia Computer Science*, 83, 958–963.
- Zipf, G.K. (1946) The P1P2/D hypothesis: On the intercity movement of persons. *American Sociological Review*, 11, 677–686.

INDEX

Tables and Figures are indicated by **bold** print. The letter “b” after a page number indicates bibliographical information in a Annotated Bibliography section.

- Abdou, M. et al 60b
Abernathy, D. 124b
accuracy of a model 244–5
see also evaluation
ACT-R 191
actions of agents 51–2
 prompted by decisions 52
 repeated actions 51–2
 segregation models 58
adaptivity of agents 17
adjacency matrices 198–9
advantages of agent-based modelling 14–15, 20–22, 275–6
AFINN 297
Africa Health Research Institute, South Africa 122
agent learning 177, 179
agents 15–19, 49–52
 actions 51–2, 57–8
 attributes 16–17, **19**
 adaptation/learning 17, 21
 autonomy 16, 17, **18**
 cross-correlated 49
 goal-directed 16
 heterogeneity 16, 21
 interactive/communicative 17
 mobility 17
 rationality 16–17
 reactive/perceptive 16
changing environment 125
characteristics 49–50
 in segregation models 56–7
communication between 274
complexity 274, 275
core elements 32
definition of 288
detail in modelling 49
inspecting current state **247**
mobility 22
in object-orientated environment **18**
representation 290
rules 18–19, **20**
Agentscript 292
aggregate complexity 2–3
aggregate data 40, 42–3, 290
airport security line simulation **271**
Ajzen, I. 180, 190
Alizadeh, M. et al 204
Alonso, W. 180, 190
Amazon rainforest deforestation 184
ambient geographical information (AGI) 106
An, L. 193b
An, L. et al 289
applications of agent-based modelling 14–15, 23–31, 294–5, **299**
 examples of models 311–28
ArcGIS 103, 104, 241
archaeology: Understanding Arifcial Anasazi 314, **314–15, 315, 316**
artificial intelligence 176
'artificial laboratories' 8, 15
'artificial worlds' 17, 19–20, 26, 126, 293–4
 using big data 295
 using vector data 149
autonomy of agents 16, 17
Axelrod, R. 36, 289
 culture model 251
 KISS (keep it simple stupid) 49
Axtell, R. 263, 264b
Axtell, R. and Epstein, J.M. 252, 264b, 289
Axtell, R. et al 251, 315
Balke, T. and Gilbert, N. 177, 192b
Barabási, A. and Albert, R. 207
bathtub (system dynamics models) **272**
Battle of Britain model 182
Batty, M. 3, 6, 8, 13b, 33b, 195, 273, 283b, 307b
BDI (beliefs, desires, intentions) model 183–4, 291
behaviour *see* human behaviour
Benenson, I. et al 130
Benenson, I. and Torrens, P.M. 33b
biases 41, 41–2
bid-rent theory of behaviour 180

- BiosGroup 30
 bird flocking 43, 47, 253, 304
 Birkin, M. et al 283b
 Birkin, M. and Wu, B. 270, 284b
 Birks, D. et al 251
 Bokeh 122
 Bonabeau, E. 21
 Borschev, A. and Filippov, A. 284b
 bounded rationality 178
 brains 304
 Bratman, M.E. et al 183
 Brewer, Cynthia 121
 Brown, D.G. 124b
 Brown, D.G. et al 250
 Brundson, C. and Comber, L. 243b
- calibration 32, 227, 251–61, 262
 process 252
 qualitative 253–4
 quantitative 254–61
 Canada Geographic Information System 97
 causality in human behaviour 180
 cellular automata models 6, 8, 266–8, 279, 280, 281, 282
 2D cellular automata 266
 cell changing process 279
 and cellular space models 139
 comparison with other models 275
 Conway's Game of Life 267–8
 urban land-use development (White and Engelen) 6
 cellular space 43, 44, 54, 139, 169
 Center for Connected Learning and Computer-Based Modeling 63
 central place theory 300
 Chainey, S. and Ratcliffe, J. 243b
 characteristics of agents 49–50
 choropleth maps 117, 121, 122
 Christaller, W. 190, 300
 Cioffi-Revilla, C. 13b
 Cioffi-Revilla, C. et al 325
 cities:
 and complexity theory 4
 economies of agglomeration and dispersion 8
 hierarchy 300–301
 human movement 296
 and individual human behaviour 273
 and individuals 273
 interrelated subsystems 300
 networks 195
 as 'people systems' 4
 residential choice 175
 slums 313, 323–5, 324
 system structure 300
 traffic 28
 urban growth modelling 21, 140–41, 140, 251, 266
 urban sustainability and climate change 301
 Clark, P.J. and Evans, F.C. 235
 climate and weather 95–6, 145, 287–8, 301, 328
 boreal and Arctic regions 327–8
 Riftland: analysing conflict, disasters, and humanitarian crises 184, 325–6
 clustering in a data set 235–7
 code errors 246, 247
 code testing 246, 247
 coding behaviours 180–81
 cognition 176, 177, 183, 186, 290, 304
 cognitive architectures 191
 cognitive frameworks 291
 Colorado Springs wildfire events 184, 296–7
 Colorbrewer2 121
 colour in visualisations 120–21
 communications 48
 and resource exchange 48–9
 community cohesion 150
 Community Earth System Model 328
 Community Ice Sheet Model 287–8
 companion modelling 253
 comparing data sets 226–42
 choosing statistics 227
 description of point data 228
 global statistics 228
 goodness of fit 226, 227, 230–33
 hypothetical data 228–30
 local indicators of spatial association 228, 237–9
 multi-scale error analysis 239–41
 overview of statistics and methods 228
 properties of point data 235–7
 visual comparisons 228, 233–4
 comparison of models 287–8
 comparisons of modelling approaches 274–83, 275, 301
 comparative analysis 280–82, 281
 SIR (susceptible–infected–recovered) model 276–82
 see also modelling approaches by name
 complex systems 304, 305
 complexity of models 51, 58, 176, 245, 252
 complexity theory 2–5
 hierarchies 4, 5
 key terms 3
 feedback 3
 nonlinear 3
 path dependence 3
 self-organisation 3, 8
 computational social science (CSS) 7
 conflict:
 conflict diamonds 322–3

INDEX

- emergence of riots 189–90, 313–**14**
and forced migration **326**–7
Riftland: analysing conflict, disasters, and humanitarian crises **325**–6
Consequences Assessment Tool Set 129
consumer behaviour **187**–8
continuous space **43**, **44**, 54
Conway's Game of Life **3**, **267**–8
Costanza, R. 239
coupling **127**–9, 301
and integration of modelling styles 301
coupling GIS and agent-based models **128**
tight coupling 128
crime:
 crime locations 9
 criminal behaviour 251
Croitoru, A. et al 218
Crooks, A.T. and Castle, C. 171b
Crooks, A.T. et al 176, 220, 260, 261, 286, 287, 291, 307b
Crooks, A.T. and Hailegiorgis, A. 138, 181
Crooks, A.T. and Heppenstall, A.J. 16, 21
Crooks, A.T. and Wise, S. 295, 318
cross-correlated attributes 49–50
cross-validation 261
crowd scenes in films 14
crowds: interactions 45
crowdsourcing 106, 295
CubeSats 108
- D3.js 121, 122
data:
 ambient geographical information (AGI) 106
 assimilation 304
 big data 15, 22, 39, 106, 108, 109, 293, 295–8
 challenges 292–4
 and computer power 7–8
 crowdsourcing 106, 295, 296
 ‘data deluge’ 7
 data sets 19, 39
 development of new forms 1–2
 for evaluation 262
 generation flexibility 40
 for GIS **105**–110
 implicit sources 107
 new data sources 7, 293, 295
 observation data 42–3
 online source **105**
 open data 7, 111, 112, 319, 326
 quality of 22–3
 quantity of 22, 32
 remotely sensed data 108–**110**
 sharing 288–9
 social media data 106–8
volunteered geographical information (VGI) 106, **107**
data collection methods 39–40
 choosing 40
Dawn of the Planet of the Apes (film) 14
de Smith, M.J. et al 124b
Deadman, P.J. et al 184
decision models 30–31, 32, 50–51
 access to information 51
decision-making 184, 185, 254, **278**, 288, 290
 learning and memory 291
deduction 36, 275, 276
Demšar, U. et al 120
density estimation algorithms 233
density in graphs 200–201
density maps 119
descriptive models 5–6
design 22, 32, 35–59
 agents 49–52
 appropriateness 39
 biases 41
 data collection 39–40
 decision-making 50–51
 development 40
 interactions 46–9
 overview 37–8, 42
 pattern-orientated models 58
process 39
purpose 38–9
segregation models 52–8
software 40
visualisations 40–41
world 42–6
detail, amount of 49
development of agent-based modelling 275–6, 286
digital data 7
Digital Globe 108
Dijkstra, E.W. 202, 209, 216
discrete event simulation **270**–**71**, 275, 279–**80**, 281, 282
examples
 SIR **279**–**80**
 waiting time at airport security checkpoint **270**–**71**
disease 30, 276
 disease dynamics in a refugee camp 138, 181, **311**–**12**
 SIR models 276, 281–2
dissemination 291–2
docking 250–51
dot density maps **118**–19
Dual Independent Map Encoding (DIME) program 97
Dual KDE 237

- Dunbar, R.I. 293
 dynamic MSMs **268, 269**
 dynamic nature of agent-based models 21
 dynamics 290, 293, 390
- Ebola in West Africa 220
 Eclipse 137
 edges (networks) 195
 Edmonds, B. and Moss, S. 49
 embedding GIS 127, 128
 emergence 8, 125, 274, 286
 and modelling styles 301
 emergence of knowledge/ideas 17, **19**
 emergency evacuation 297
 emotions 177, 186
 ensemble modelling **303**
 environments 19–20
 complexity 176
 simplifying 247–9, **248**
 Epstein, J.M. 16
 Epstein, J.M. and Axtell, R. 26–8, 36, 269n2, 275
 Growing Artificial Societies 26
 Erdős, P. and Rényi, A. 206
 Erlang, A.K. 270
 errors, size of 251
 ESRI shapefiles 101, 102, 103–4, 149–50
 Euler, L.: Seven Bridges of Königsberg **196–7**, 208
 evaluation 22, 32, 50, 241, 244–63
 calibration 227, 251–61
 difficulty of 262
 need of big data 262
 validation 32, 227, 245, 261, 289
 verification 245–51, 289
 expected outcome alignment 250
 Exploring the Growth of Slums **324**
 exporting data 165–9
 external systems 42–3
- face validation 252
 Facebook 217
 fast and frugal model of behaviour 184–6
 feedback (negative and positive) 3
 field-based views 98
 Filatova, T. et al 193b, 287
 fitness 251
 Flickr 106, 217, **218**
 Forrester, J.W. 271
 Fotheringham, A.S. et al 274
 Franklin and Graesser 16
 Fuzzy-Kappa Simulation 138
- gal files 104
 Galán, J.M. et al 246
 Gama 131
- Gama (GIS Agent-Based Modelling Architecture) **137–8**
 applications 137
 genetic algorithms (GAs) 259
 gentrification 32, 135, 287
 geo-atom theory 102n6
 geocomputation 8
 GeoDa **104, 241**
 geographical information, proliferation 217–18
 geographical information systems (GIS) 95–123
 abstraction from real world **126**
 with agent-based modelling 9–10, 125–70
 complementarity 10
 coupling 127–9, **128**
 embedding 128
 toolkits 129–37
 commercial applications 97
 commercial value 98
 data sources **105–110**
 decision-making 97
 growth in uses 97–8
 history 96–8
 meaning of 'GIS' 96n1
 modelling space–time changes 125–6
 open source GIS 97
 planning 123
 representing the world 98–102
 software 103–5
 time and change 102–3
 tools 103
 uses 9–10, 95–6, 97–8
 uses and purposes 9–10
 using QGIS 110–117
 visualising results 117–22
 classes of data 121
 colour 120–21
 interactivity 121–2
 maps 117–20
 three[3]D visualisation 119
 geographical space 43, **44**
 GeoMASON 184
 GeoNames 217
 Geopandas 294
 geosocial analysis **108**
 get-happy-agents 246
 Getis-Ord GI* 227, 237–9, **238**
 Gilbert, N. et al 292
 Gilbert, N. and Troitzsch, K.G. 33b, 35, 36, 269n2,
 274, 284b
 Global Change Assessment Model 301
 goal-directed agents 16, 183
 goodness of fit 226, 227, 230–33, **231, 233**
 Google Earth 109
 Google Scholar 23

INDEX

- Google Street View 106
Gordó, S.P. et al 315
graphs 196–203
 adjacency matrices 198–9
 density 200–201
 links: directed or undirected 197
 in NetLogo 197–8
 nodes
 betweenness centrality 201–2, 203
 degree centrality 201
 importance of 201–3
 pairs through network 202
path 199
trail 199
traversing graphs 199–200
walk 199
- Graser, A. 124b
GRASS 104
gravity models 273
Grimm, V. et al 22, 58, 61b, 288
Grimm, V. and Railsback, S.F. 17
Groeneveld, J. et al 287
growth of agent-based modelling 15–16, 24
Gulden, T. et al 176
- Haggett, P. and Chorley, R.J. 224b
Hamill, L. and Gilbert, M. 204
Harris et al 8
Hazard Prediction and Assessment Capability system
 129
health 106
Heppenstall, A.J. et al 18, 262, 307b
heterogeneity of agents 16, 21, 39
heuristic models 184, 185
heuristic searches 258–9
hierarchical structures 51, 300–301
hill climbing algorithms 259
Hotspots model 319
Hu, E. 326
hubs (networks) 207
human behaviour 14–15, 172–92, 290–91
 beliefs–desires–intentions model 183–4
 causality 180
 cognitive architectures 191
 complexity 173, 174, 175, 176, 245
 conceptual cognitive models 183–6
 diversity 306
 embedding in models 22
 fast and frugal model 184–6
 focus on aspects 175
 frameworks 176–9, 290, 291
 key assumptions and theories 178–9, 180
 main dimensions 177
- individuals 273
mathematical approaches 180–83
PECS framework 186
predictability 174
probabilistic models 181–2
social relationships 174
spatial knowledge 304
theories 290
threshold models 182–3
using big data 173, 191
Hurricane Sandy 95–6
hypothetical data 228–30
 hypothetical models plotted against observed data
 230
scatter plot 229
- Iberian Peninsula 315, 317
ImportRasterSample model 99
individuals 274, 275
 and aggregations 290
 availability of data 173
 behaviour 14
 decisions 20
 diversity 190–91
 need for study 273
 representations of 15, 274
 simulation of 7–8, 14
 induction 36, 275, 276
 information, access to 51
 information storage 96, 294
 integration of modelling styles 301
interactions (agents) 17, 46–9
 communications 48
 complexity 21
 physical 46–7
 resource exchange 48–9
interactive visualisations 121–2
internal validity 245
internet for sharing and dissemination 292, 293
invasive species 106
Itanen, S. 284b
- Jackson, J. et al 250
JavaScript 292
Jenkins, A. et al 109, 110
- KDE (kernel density estimation) 227, 233,
 234, 238
Kennedy, W. 174, 192b
Kennedy, W. et al 325
Kenya 311, 312, 313
Knudson, D.C. and Fotheringham, A.S. 231, 243b
Kornhauser, D. et al 60b

- land-use 97, 100, 140–**41**, 184, 301, 320, 322
 community resource management 320, **322**
 evaluation 263
 spread of agriculture in Neolithic period 315, **317**
- land-use modelling, and theory 287
- land-use–transport interaction (LUTI) 301
- Landsat data 100, 103
- Landsat program 97
- layers of data 102, **103**
- Lazer, D. et al 13b
- Le Billon, P. 322, 323
- learning agents 305
- learning and memory 291
- Lee, J.-S. et al 243b
- Li, J. and Wilensky, U. 26, 27
- limitations of agent-based modelling 22–3, 39
- line maps 119
- links (networks) **197**, 198
- LISA (local indicators of spatial association) 227, 233–4
- location-aware devices 217
- Longley, P.A. et al 95, 96n1, 100n5, 121
- loose coupling 127
- Macal, C.M. and North, M.J. 16
- Magliocca, N.R. and Ellis, E.C. 58
- Maguire, D.J. 124b
- Maguire, D.J. et al 171b
- Malleson, N. 239
- Malleson, N. and Birkin, M. 296
- Malleson, N. et al 17, 193b
- Mandelbrot, B.B. 289
- Manley, E. et al 185
- Manson, S.M. 2
- Mapillary 106, **107**
- mapping agencies 17
- maps 117–**20**
 elements:
 graphs 120
 labels 119
 legend 119
 north arrow 119–20
 scale bar 119
 text 120
 thematic 233, 234
 types 117–19
- Maps.Me **107**
- Maslow, A.H. 190
- MASON (Multi Agent Simulation of Neighbourhood) 129, 131, 132, 137, 141, 184, 259
- spatial models 133
- Massive 14
- mathematical approaches to behaviour modelling 180–81, 191
- MATSims 209, 291, 301
- merging models 302
- micro-level interactions 4
- microsimulation models (MCMs) **268**–70, **269**, 275
- and agent-based models 269n2
 agents 269n2
- migration 326–7, **326**–7
- Milgram, S. 206
- Miller, J.H. and Page, S.E. 13b
- mobility 17, 22
- model development software 40
- model integration 298, 299, 302
- modelling:
 comparisons of modelling approaches 274–83
 and geographical information systems (GIS) 9–10, 97
 purposes 286–7
 and theories 287
 types of models 5–6
- Modelling Human Behaviour framework 291
- models:
 levels 6
 purposes 5–6
 and simulation 6
 and theories 5
- Moss, S. 264b
- Müller, B. et al 61b, 288, 291
- multi-scale error analysis 239–**41**, **240**
- Namatame, A. and Chen, S.-H. 225b
- NASA Earth Observing System Data and Information System 109
- National Land Cover Database 105
- natural disasters 175
 Haiti earthquake 295, **296**, 318–**19**
 and humanitarian relief 318–**19**
 wildfires 106, 109, 296–7
 California Sand Fire **31**
 social media content to inform agent-based models for humanitarian crisis response 319–**20**
 training 30–31
- Natural Earth Data 105, **111**
- natural resources 313
- NatureServe Vista 129
- near-decomposability 4
- negative feedback 3
- neighbourhoods 56, 138, 139, 169
 gentrification 32, 135, 287
- hierarchies 4, 5

INDEX

- Moore/von Neumann neighbourhoods 56, 138, 139, 169, 266
perceptions of 109
segregation *see* Schelling's *Segregation* model
see also cities
- NetLogo 40, 62–93, 135
ask command 70–72
 color variable 70–73
 pcolor variable 70
 set pcolor blue 71
ask patches 249
ask and with 72–3
BehaviorSearch 259
BehaviorSpace 255, 256, 257, 259
button and associated code 77
clear-all command 79
Code tab 76
consumer behaviour modelling 188
contexts 68, 69
count command 67
dictionary 154
environment 74
exporting data 166–7, 168
foreach loops 154
geographically explicit agent-based models 136
gis:fill 155
gis:set-drawing-color 155
gis:store-dataset 167, 169
GIS extension 138, 146, 151
go button 63
go once button 63
graphs 197–8
 create-link-with/create-link-to/from 197, 198
 links documentation 198
importing network data 203
Info tab 64
Inspect feature 247, 247
interface: main components 66
LevelSpace 302
model building 73–93
 simple model 73–83
 creating a button 75–7
 environment configuration 74–5
 final version 83
 go button 80
 interface 81
 setting up the model 76–7
 sliders 77–8
 turtles and patches 78–80
advanced model 83–90
 final model 84
 giving birth 89–90
 grass growth 87–9, 88
 code 89
new functionality 83
variables
 energy 85
 grass-regrow-time 88
 time-since-eaten 85
graph creation 91–3, 92
Models Library 63, 139, 253
models listed 11–12
models on the web 292
multi-level modelling 135
neighbors command 139
neighbours4 command 139
Network Extension toolkit 203
observer 68, 69
patches 63, 67, 68, 79–80, 139, 148
 elevation 249
 size 151, 152
Preferential Attachment model 207–8
procedures 76–7
R extension 105, 135
random networks 204–6
raster data 139–49, 167, 168
Schelling's *Segregation* model 11, 12, 24–6, 52–8, 62–5, 64, 167
set command 68
setup button 63
show command 65
sliders and variables 77–8
toolkits 131
transport networks 209–217
 GMU-Roads Model 210–217
 breeds and variables 211
 creating the road network 212–13, 214
 loading GIS data 211–12
 moving agents along the road network 214–16
 route finding 216–17
 vertex values 214
 turtles 63, 67, 68, 78–80, 139–40
variables 68, 69, 84–5
vector data 149–65, 167
Virus model 63, 65, 70
visualisations 41
WalkThisWay model 259–61, 260
with command 67
Wolf Sheep Predation model 63, 65–8, 68
NetLogoWeb 137
network space *see* topological space
networks 194–224
 definition 195
Directed Network Demo 194, 200
edges 195
geographical linked with social networks 217–23, 222

- GMU-Roads 194
 graphs 196–203
 hubs 207
 linking 221
 merging 220–21
Networks in NetLogo 194
 nodes 195, 201–3, 207–8
 preferential attachment 207–8
 probability 181–2
Random Network Example 194
 random networks 203, 204–6
 scale-free networks 207–8
 small-world networks 181–2, 203, 206–7
 transport networks 208–217
Undirected Network Demo 194, 200
see also social networks; transportation, networks
- New York City: tweets relating to entertainment 109–110
 NNI (nearest neighbour index) 227, 235–6
 nodes 195, 207
 calculating importance 201–3
 nonlinear complexity 3
 norm consideration 177
 North, M.J. et al 301
 North, M.J. and Macal, C.M. 292
- object-based views 98
 objective functions 259
 observation data 9, 10, 39, 42–3, 102, 241, 254, 256, 258
see also satellites; surveillance systems
- ODD+D protocol (Overview, Design concepts and Details)+Decision 36, 58–59, 288, 291
- ODD (Overview, Design concepts and Details) 36, 58–9, 288
- Oldham, M. 182
- open data 7, 111, 112, 319, 326
- open source GIS 97
- open source libraries 294
- OpenABM 289, 291
- OpenStreetMap 19, 105, 106, 137, 295
- Orsi, E. and Geneletti, D. 59, 313
- O’Sullivan, D. 2, 302
- O’Sullivan, D. and Haklay, M. 41
- O’Sullivan, D. and Perry, G.L. 93b
- O’Sullivan, D. and Unwin, D. 233, 236, 242b
- overfitting 251, 261
- OYSSEY GIS 97
- pair programming 246
- parameters 50, 251
 space search 254, 256
- Park, B.H. et al 301
- Parker, Miles 315
- Patel, A. et al 323, 324
- path dependence 3
- pattern-orientated models (POM) 36, 58
 applications 58
- PECS framework 186
 simulating behaviour in riots 189–90
- pedestrian modelling 47, 49, 141–5, 263
 evacuation model 142–5, 143
 evaluation 263
- WalkThisWay model 259–61, 260, 317–18
- photos as data 106
- physical interactions 46–7
 agent-to-agent 47
 agent-to-physical environment 47
 indirect agent-to-agent 47
- physical rules 46
 segregation models 55
- Pires, B. and Crooks, A.T. 59, 186, 189, 193b, 223, 313
- Planet 108
- planned behaviour theory 178, 180
- point maps 118
- populations 45–6, 49, 290
 segregation models 55
- populations at risk 109
- positive feedback 3
- predictive models 6
- probabilistic models of behaviour 181–2
 simulating consumer behaviour 187–9, 188
- properties of point data 235–7
- prospect theory 179
- protest movements 107, 109
- proximity 19
- Python (software) 104, 122, 288, 294
- qualitative calibration 253–4
- quantitative calibration 254–61
 example: WalkThisWay model 259–61, 260
 heuristic searches 258–9
 parameter space search 256
 sensitivity analysis 50, 257–8
 six steps 254
 uncertainty analysis 258
- quantitative and qualitative methods 9
- Quantum GIS (QGIS) 103, 104, 110–117, 241
 attribute table 115–16
 ‘clipping’ vector data 114
 new field 115–16
 open data 111, 112
 spatial operations 111–13
 vector data 113–14
- queueing models 270
- R (software) 104, 105, 288
- R squared 226

INDEX

- Rahmandad, H. and Sterman, J. 250
Railsback, S.F. et al 314
Railsback, S.F. and Grimm, V. 93b, 254, 264b
Railsback, S.F. and Johnson, M.D. 58
random networks 204–**6**
 criticism 206
raster data 98, **99–100**, 138, 139–49
 cells 100, 100n5, 139, 147
 pedestrian modelling 141–5
 Rainfall model example 145–8, **146**
 urban growth modelling 140–1
 and vector formats 169
rational choice theory 178
rationality of agents 16–17
Ravenstein, E.G. 273
recreational sites 106
Reilly, W.J. 273
reinforcement learning 305
remotely sensed data 108–**110**
Repast:
 exporting data **166**
 Repast Symphony 131, 132, 133, 133–4, 137
 vector agent-based models **134**
replication 288–9, 306
resource exchange 48–9
riots *see* conflict
Ripley's K function 227, 236, **237**
routeing 209
RSS (residual sum of squares) 226, 230–31
rules 18–19, 46
- Sample of Anonymised Records 268
sampling 290
Sasaki, Y. and Box, P. 251
satellites 100, 103, 108–9
scale 41
scale-free networks 203, 207–**8**
Schelling's *Segregation* model 6, 11, 12, 19–20, **23–6**, **25**, 125
 changing patterns of segregation 24–5, **26**
example:
 agents:
 actions 57–8
 characteristics 56–7
 decisions 57
 interactions:
 communication 56
 physical interactions 56
 resource exchange 56
 overview 52–4
 assumptions 54
 biases 54
 development and software 53
purpose and process 52
visualisations **53**
world:
 external systems 54
 physical rules 55
 populations 55
 space 54
 time 55
manipulating vector data in QGIS 113–**14**
progression of segregation over time **25**
as raster data model 139
threshold rules **182**
vector data variations 150–65
 creating agents from a shapefile 159–65, **162**, **164**
 drawing spatial features 153–6
 moving the agents 158
 multiple agents per polygon 158–9, **158**
 patch size 151
 polygon neighbours 155
 reading data 153
 storing values of polygons 156–7
Schlüter, M. et al 173, 180, 191, 192b, 291
Schmidt, B. 186
scientific process: induction/deduction 276
SeeClickFix 106, **107**
self-organisation 3, 4, 8
sensitivity analysis 50, 256, 257–8, 282, 289
shapefile format 102
Shapely 294
sharing models 288–9, 291–2
Shiflet, A.B. and Shiflet, G.W. 276, 277, 284b
Shiny 122
Sierra Leone civil war 186, 322–3
Simon, H.A. 4, 5, 13b, 23, 300
simplified environments 247–9, **248**
SimTable 30–**31**
simulations 6, 35–**6**, 40
 extent 42
Singleton, A.D. et al 307b
SIR (susceptible-infected-recovered) model 276–82
 agent-based model 276, 277–**8**, 281, 282
 agent decision process **278**
 cellular automata model 279, **280**, 281, 282
 discrete event simulation 279–**80**, 281, 282
 system dynamics (SD) approach 276, **277**, 281, 282
SLEUTH model **140–1**
 layers **140**
small-world networks 181–2, 203, 206–7
smartphones 217
Soar 191
social circles 204

- social media 106–8, 217, 296–7
 informing agent-based models for humanitarian crisis response 319–20
- social networks 48, 174, 177, 195, 196, 203–4, 293
 linked with geographical networks 217–23, 222
 models:
 random networks 203
 scale-free networks 203–4
 small-world networks 203
- random networks 203
 random networks 203
 scale-free networks 203–4
 and social media 293
 ties, similarities, social relations, interactions, flows 203
- social norms 179
- social sciences 275, 287
- socially embedded agents 196
- software:
 for GIS 103–5
 for Mac 104
- space 47, 138
 cellular models 139, 148, 169
 purpose in agent-based models 127
- scale 21, 45
 segregation models 54
 start location 50
 types 43–5, 44
 visualising 117–20
- spatial data 293–4
 spatial interaction models 272–4, 275
 applications 274
 spatially embedded communication 219
 spatially learning agents 304–5
Spectre (film) 14
- SRMSE standardised root mean square error 227, 231
- stakeholder validation 262
- stakeholders 292
- start location 50
- Sterman, J.D. 284b
- stochasticity 50
- store location 96
- Sturley, C. et al 187, 192b
- subsystems 4, 300, 301
SugarScape model 23, 26–8, 251, 269n2
 wealth distribution model 27
- surveillance systems 109
- surveys 39
- Swarm 131, 132, 137
- Swartz, M. 320
- SYMAP project 96–7
- Syrian refugees 32
- system dynamics (SD) models 42, 271–2, 273, 274, 275, 276, 281, 282
 docking 250
 flowchart 277
 and integration of modelling styles 301
 process 277
 and validation 245, 261
 wolf-sheep predation 272, 273
- system structure 300
- system/environment complexity 176
- systems hierarchies 300–301
- Tajfel, H. and Turner, J.C. 190
- Taos, New Mexico 320
- TasP trial dashboard 122
- theories, testing 293
 theories of human behaviour 178–9, 180
 theory development 287
 theory testing 287
 3ds Max 41
 3D visualisation 119, 135
 threshold rules 182–3
 time and change 45, 102–3, 268–9
 segregation models 55
 snapshot approach 102, 103
 time-scales 21
 time-series variation 254
 time-step intervals 45
 toolkits 129–37, 131
 topological space 44
 Torrens, P.M. 12b, 13b, 289, 294
- TRANSIMS 209
- transportation 28–30, 109, 300, 312
 flow of pedestrians 208–9
 GMU-Roads Model 210–217
 route finding 216–17
- hiker movements in the Dolomites UNESCO World Heritage Site 313
- networks 195, 208–217
 routeing 209
 traffic flow 28–30, 29, 40, 42
 transportation and development for Reston, VA 320, 321
- Twitter 106, 109, 217, 218, 297, 298, 299
- Uganda 111, 112
- UML diagrams 292
- uncertainty analysis 258, 303
- Understanding Arificial Anasazi 317
- unit testing 246
- Unity 41
- Urban, C. 186

INDEX

- US Federal Emergency Management Agency
(FEMA) 95–6
- USGS Earth Explorer 105
- Ushahidi 295
- validation 32, 227, 245, 261
challenge of 289
and system dynamics (SD) models 245
- vector data 98, **99**, 101–2, 135, 138, 149–65
applications 149
basic building blocks **101**
example: Schelling's *Segregation* model 150–65
toolkits 149n8, 149–50
- vector formats, and raster data 169
- verification 145, 245–51, 262, 289
test plan 250
- visual comparisons 233–4
- visualisation 40–41, 117–22, 262, 291–2
colour 120–21
three[3]D visualisation 119
- volunteered geographical information (VGI) 106, **107**
- von Thünen model of urban development 6, 251
- Walsh, C.L. et al 301
- Wasserman, S. and Faust, K. 224b
- water flow:
community resource management 320, **322**
and erosion 135, 145
- Foothill Yellow-Legged Frog Assessment Model 314
- Rainfall model: Crater Lake 145–8, **146**, **248**–9
- Watts, D. and Strogatz, S.H. 206–7
- wealth distribution 26–8
- Web of Science 23
- Wegener, M. 95
- Werker, C and Brenner, T. 254
- Westervelt, J.D. 127, 128, 171b
- 'what if' scenarios 142
- White, R. and Engelen, G. 6
- Wilensky, U. 24, 29, 52, 267, 273
- Wilensky, U. and Rand, W. 34b, 93b, 225b, 263b
- Wilson, A.J. 6
- Windrum, P. et al 264b
- Winnipeg, Canada: income per annum **118**
- Wise, S. 184, 296
- Wise, S. and Crooks, A.T. 320, 322
- wolf-sheep predation 253, **273**
- Wolfram, S. 284b
- Wooldridge, M. and Jennings, N.R. 16
- world 19–20, 42–6
abstracting from **126**
external systems 42–3
physical rules 46
populations 45–6
space 43–5, **44**
time 45
- World War Z* (film) 14
- WorldPop 105, **111**
- Yuan, X. and Crooks, A.T. 220
- Zipf, G.K. 273