AN ILLUSTRATED GUIDE

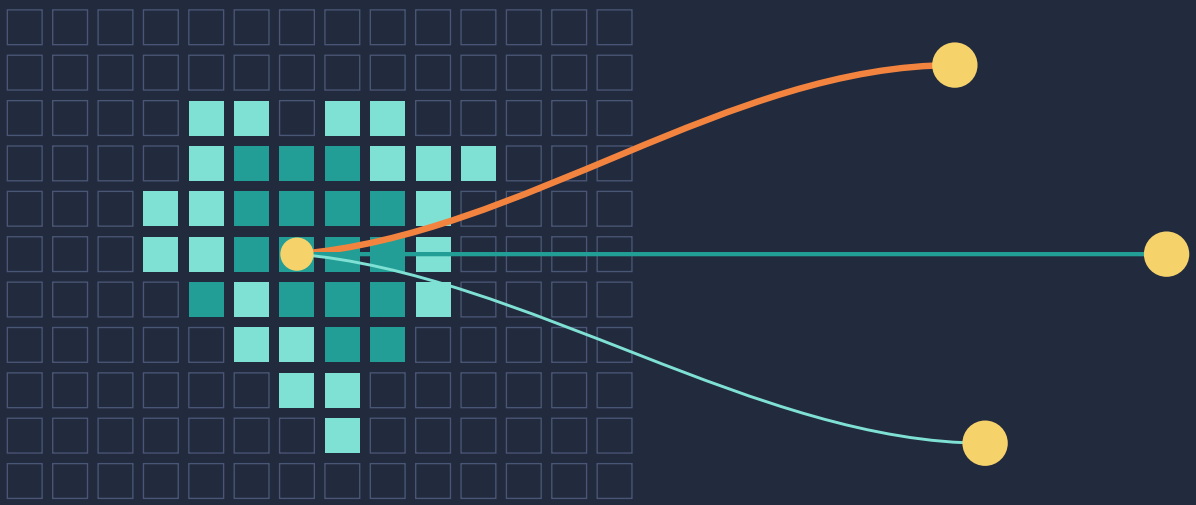
Urban Modelling and Simulation

A friendly guide to building models of cities and running them to test ideas

Plain English, pictures first, seven small models built and run for real, with their numbers, plus a quiz, a glossary and an eight-week plan.

Sixteen short chapters · Quiz · Glossary · Eight-week plan

Prompted and directed by Mahmoud A.
Written and illustrated by Claude (Anthropic)



a town that grows, cell by cell

trips between places

How to Use This Guide

Read it in order, or jump to what you need.

This guide explains **urban modelling and simulation**. A **model** is a simplified copy of a city, or of one part of it, written as rules that a computer can follow. A **simulation** is what happens when you run those rules forward in time and watch. Together they let you test an idea, such as a new rail line or a new housing policy, before anyone builds it. You need no maths beyond school level and no coding.

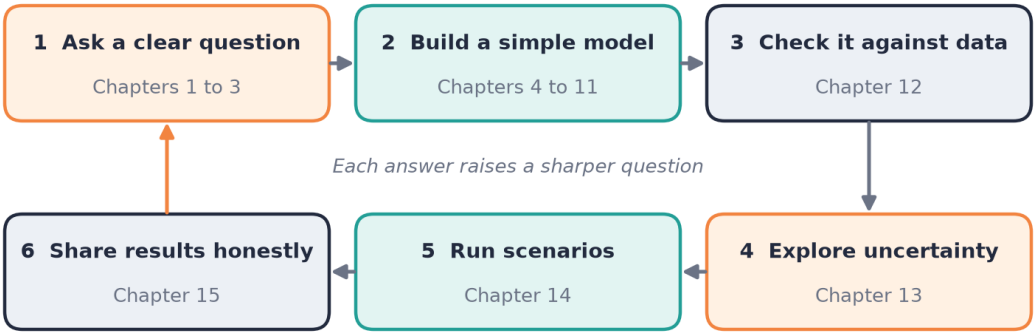


Figure 1. The whole guide in one picture. Building and using a model is a cycle, and each step has its own chapters.

Who it is for

- Curious people who hear words like **agent-based model** or **traffic simulation** and want to know what they mean.
- Planners, architects, engineers, students and community groups who read model results and want to ask good questions about them.
- Anyone who wants to build a first small model of their own.

How each chapter works

- A **banner** tells you the topic in a sentence.
- Plain explanation first, then a picture, then a small example.
- Boxes: **TIP** (a shortcut), **SAY IT LIKE THIS** (a phrase you can use out loud), **WATCH OUT** (a common trap) and **IN ONE LINE** (the chapter in a sentence).
- At the back: myths and facts, a quiz with answers, a glossary, a one-page cheat sheet and further reading.

A note on the numbers

Every number in the worked examples was **computed** by the script that built this guide. The towns, zones and roads are imaginary and small, so you can see every moving part. Random elements use fixed seeds, so the same script always gives the same results. Treat the numbers as teaching examples. They describe no real place.

Contents

Sixteen short chapters and a toolbox at the back.

1 **What Is a Model?**
A simplified city you can run, rewind and run again.

2 **The City as a System**
Stocks, flows, feedback and surprises.

3 **A Map of Model Families**
Seven ways of seeing a city, and when to use each.

4 **System Dynamics**
A tiny housing model, built and run.

5 **Agent-Based Models**
Simple individual rules, surprising city-wide patterns.

6 **Cellular Automata**
Growing a town on a grid, one cell at a time.

7 **Gravity Models**
Big places attract, distance deters.

8 **Networks and Traffic**
Roads as links, drivers as flows, jams as feedback.

9 **Discrete-Event Models**
Arrivals, waiting and the cost of a busy system.

10 **Physics-Based Models**
Energy, heat and air, in brief.

11 **Land Use and Transport Together**
Where we live shapes how we travel, and the other way round.

12 **Calibration and Validation**
Tuning the dials, then testing on data the model has not seen.

13 **Uncertainty and Sensitivity**
How sure are we, and which assumptions matter most?

14 **Scenarios and Forecasts**
Exploring several possible futures instead of predicting one.

15 **Communicating Honestly**
Show the range, state the assumptions, avoid the classic traps.

16 **Try It Yourself**
A short recipe, the tools, and an eight-week plan.

At the back: Myths and Facts · Quick Quiz · Glossary · One-Page Cheat Sheet · Further Reading

1 What Is a Model?

A simplified city you can run, rewind and run again.

A metro map is a model. It leaves out streets, hills and true distances, and keeps only stations and lines. Leaving things out is what makes it useful. If it showed everything, it would be as hard to read as the city itself.

An urban model does the same thing with numbers and rules. It keeps the parts of a city that matter for one question and leaves the rest out. A **simulation** runs the model forward in time, step by step, so you can watch what might happen. You can rewind it, change one thing and run it again. You cannot do that with a real city.

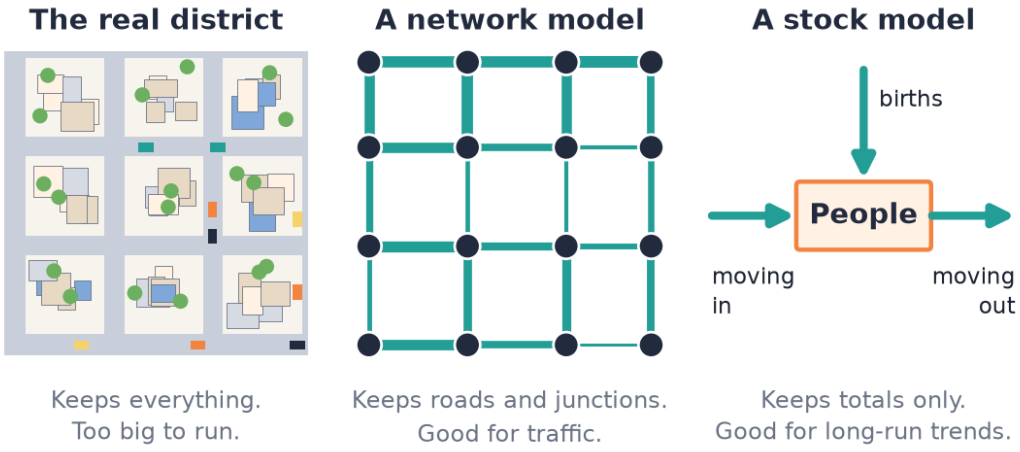


Figure 2. Three models of the same district. Each keeps different details, so each can answer different questions.

Why build one?

- **Cities are too big to experiment on.** You cannot build a motorway to see what happens, then remove it.
- **Effects are delayed.** A new rail line may change where people live over decades.
- **Effects are indirect.** A new car park can change bus use, shop rents and air quality at once.
- **People disagree.** A shared model puts assumptions on the table, so the debate can be about evidence.

All models are wrong

The statistician George Box is often quoted as saying that “all models are wrong, but some are useful”. Every model leaves things out, so none is the full truth. The useful question is whether a model is good enough for the job in hand. A model of rush-hour traffic can be useful without knowing anyone's name. The same model would be useless for questions about loneliness.

The parts of a model

Part	What it means
Question	What you want to find out. It decides everything else.
Inputs	Data that describe the starting city, such as population, roads or land use.
State	What the model keeps track of as it runs, such as people per zone or cars per road.
Rules	How the state changes from one moment to the next.
Parameters	Numbers inside the rules that set how strongly things happen, such as a moving rate.
Time step	How far the clock moves each step, from a second for traffic to a year for growth.
Outputs	What you read off at the end, such as travel times, queue lengths or a map of growth.

TIP

Start every model with one written question, such as “How long will drivers queue at the new charging hub on a busy day?” The question tells you what to keep and what to leave out.

SAY IT LIKE THIS

“This model is a thinking tool. It shows what follows from our assumptions, and we can test those assumptions together.”

IN ONE LINE

A model is a deliberate simplification of a city that you can run, and its value depends on the question you ask it.

2 The City as a System

Stocks, flows, feedback and surprises.

A **system** is a set of parts that affect each other. A city is a huge one. Homes, jobs, roads, people, money and weather all push and pull on each other. A few ideas help you see the structure.

Stocks and flows

A **stock** is anything that piles up and can be counted at a moment in time, such as the people living in a town, its homes, or the cars in a car park. A **flow** is a rate that fills or drains a stock, such as births per year, homes built per year or cars arriving per hour. Think of a bathtub. The water in the tub is the stock. The tap and the drain are the flows.

Stocks change slowly, even when flows change fast. This gives cities momentum. If all building stopped tomorrow, the number of homes would barely change this year.

Feedback loops

A **feedback loop** is a chain of cause and effect that comes back to where it started. There are two kinds.

- A **reinforcing loop** (R) amplifies change. More people mean more births, which mean more people.
- A **balancing loop** (B) pushes back towards a limit. More people mean more crowding, which makes the town less attractive, which slows the arrival of newcomers.

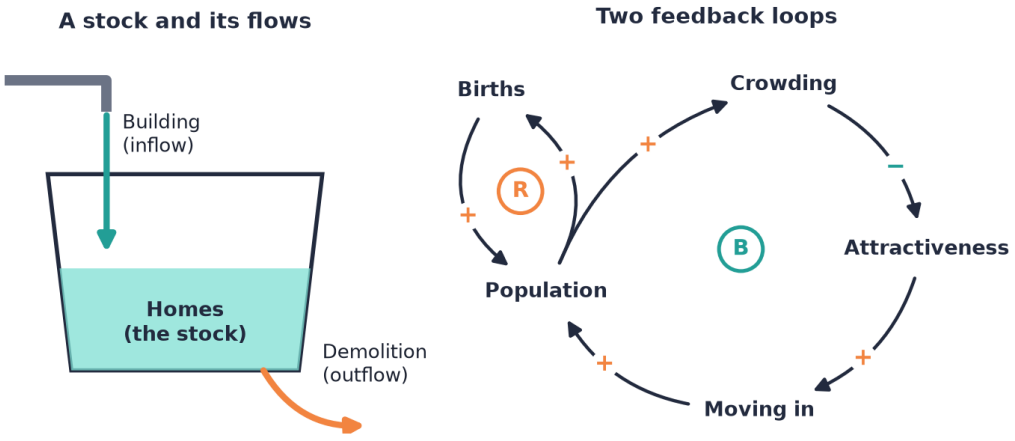


Figure 3. Left: a stock with an inflow and an outflow. Right: a causal loop diagram with a reinforcing loop (R) and a balancing loop (B). A plus sign means “more causes more”. A minus sign means “more causes less”.

Emergence, delays and non-linearity

Emergence means a pattern at the scale of the whole city that nobody planned and no single rule describes. Traffic jams, busy high streets and divided neighbourhoods can all emerge from many small choices. Chapter 5 shows a famous example.

Two more features make cities tricky. **Delays** separate causes from their effects, sometimes by years. **Non-linearity** means that doubling a cause need not double its effect. A road at half its capacity flows freely. Add a few more cars when it is nearly full and it can grind to a halt.

> **WATCH OUT**
> Our minds look for one cause and one effect. In a system, the effect often loops back and changes the cause. Drawing the loops on paper is one of the best ways to catch this.

> **IN ONE LINE**
> A city is a system of stocks, flows and feedback loops, and its most striking patterns often emerge without anyone planning them.

3 A Map of Model Families

Seven ways of seeing a city, and when to use each.

There is no single urban model. There are families of models, and each sees the city in its own way. Some count totals. Some follow individuals. Some track fields of heat or air. After the question, choosing the family is the most important design choice you will make, because it fixes what the model can and cannot say.

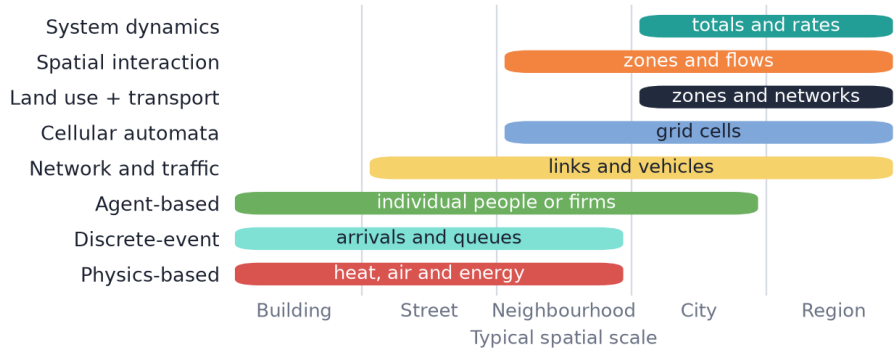


Figure 4. Model families placed by the scales they usually work at. The text on each bar says what the model keeps track of. Land use plus transport (Chapter 11) combines several of the seven families.

Ch	Family	Sees the city as	Reach for it when you ask
4	System dynamics	Stocks, flows and loops	How will totals change over decades?
5	Agent-based	Individuals with rules	How do choices add up to patterns?
6	Cellular automata	A grid of changing cells	Where might the city grow?
7	Spatial interaction	Zones sending trips	How many trips go between places?
8	Network and traffic	Links carrying flows	Where will congestion form?
9	Discrete-event	Arrivals and queues	How long will people wait?
10	Physics-based	Fields of heat and air	How hot, how windy, how much energy?

TIP

Pick the simplest model that can answer your question. Extra detail costs data, time and trust. A simple model you understand is worth more than a complex one you cannot explain.

IN ONE LINE

Match the model family to your question, its scale and the data you have.

4 System Dynamics

A tiny housing model, built and run.

System dynamics models a city as stocks connected by flows, with feedback loops setting how fast each flow runs. The engineer Jay Forrester developed the method, and in his 1969 book *Urban Dynamics* he applied it to a whole city. It works with totals and ignores individuals, so it is quick to build and to run. It suits questions about long-term behaviour over years or decades.

The model

Our imaginary town starts with 50,000 people and 20,000 homes. Each home holds about 2.5 people, so the town starts with exactly enough homes. There are two stocks, people and homes. These rules connect them.

- **Natural growth.** Births minus deaths add 0.5% of the population each year.
- **Moving in.** Newcomers equal to 12% of the population arrive each year, multiplied by **housing adequacy**, which is homes available divided by homes needed. When homes are scarce, fewer people come.
- **Moving out.** 10% of residents leave each year.
- **Building.** Builders try to close any shortage within 2 years and replace old homes. Building slows sharply as land runs out. The land fits about 34,000 homes.
- **Demolition.** 1% of homes are removed each year.

The script steps the model forward a quarter of a year at a time for 60 years. At each step it works out the flows from the current stocks, then updates the stocks.

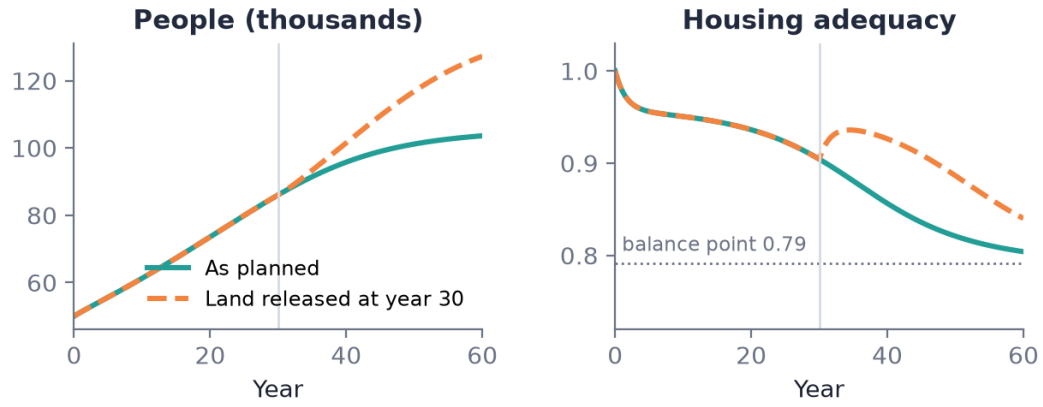


Figure 5. The housing model, computed for this guide. Solid lines: the town as planned. Dashed lines: land for 10,000 more homes is released at year 30. Adequacy is homes divided by homes needed.

Year	People	Homes	Adequacy	People, more land	Homes, more land	Adequacy, more land
0	50,000	20,000	1.00	50,000	20,000	1.00
10	61,300	23,300	0.95	61,300	23,300	0.95
20	73,600	27,600	0.94	73,600	27,600	0.94
30	86,100	31,100	0.90	86,100	31,100	0.90
40	95,800	32,800	0.86	101,600	37,600	0.93
50	101,200	33,200	0.82	116,900	41,400	0.89
60	103,600	33,300	0.80	127,300	42,800	0.84

What the run shows

For about two decades building roughly keeps pace, and adequacy stays near 0.95. Then land runs short, building slows and adequacy falls. By year 60 the town holds 103,600 people and adequacy is 0.80, close to the balance point of 0.79. There, newcomers plus natural growth exactly balance the people leaving. The balancing loop has taken over.

Releasing land at year 30 lifts adequacy to a peak of 0.94 around year 34. Then it drifts down again, reaching 0.84 by year 60, while the population grows to 127,300. In this model, the new homes attract new residents until crowding returns.

SAY IT LIKE THIS

“In this model, extra homes on their own do not end crowding, because newcomers fill them. A lasting change has to alter the feedback loop itself.”

WATCH OUT

This is a toy. Real migration depends on jobs, prices, schools and much more. The point is the structure of stocks and loops. The numbers belong to an imaginary town.

IN ONE LINE

System dynamics shows how stocks, flows and feedback loops shape long-run change, and it often reveals why quick fixes fade.

5 Agent-Based Models

Simple individual rules, surprising city-wide patterns.

An **agent-based model** (ABM) simulates many individual decision-makers, called **agents**. An agent can be a household, a driver, a pedestrian or a firm. Each agent has its own state, looks at its own surroundings and follows its own rules. The modeller then watches what happens when thousands of agents act together.

Schelling's segregation model

In 1971 the economist Thomas Schelling published a model simple enough to play with coins on a chessboard. Two groups of households live on a grid. Each household looks at its eight neighbours. If too few of them are like itself, it is unhappy and moves. In Schelling's version it moved to the nearest spot where it would be content. Our version, like many simple ones, moves it to a random empty spot.

We use a 40 × 40 grid with 1,440 households in two equal groups and 10% empty cells. The grid wraps around at the edges, like the screen of an old arcade game. Each household wants at least 30% of its neighbours to be like itself. That is a mild preference. Each household is content to be in a clear minority.

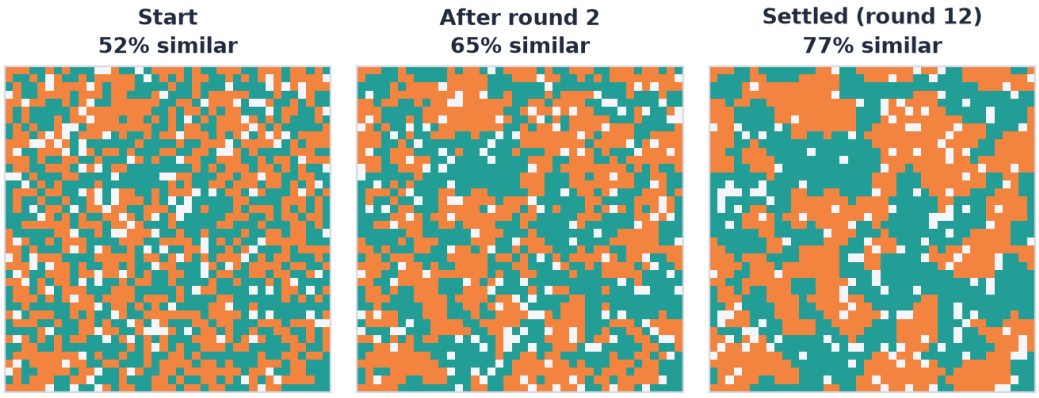


Figure 6. Schelling's model, run for this guide. Teal and orange are the two groups. Light cells are empty. The percentage is the average share of a household's neighbours who belong to its own group.

At the start, a household's neighbours are 52% like itself on average, which is roughly what chance gives with two equal groups. After 12 rounds of moving, everyone is content and the figure has risen to 77%. Nobody wanted a divided town. The pattern emerged from mild preferences and many moves.

Stronger preferences, stronger patterns

We repeated the run 20 times for each preference, each time with a different random seed. Asking for 20% similar neighbours ends between 54% and 59%. Asking for 50% ends between 85% and 89%. Asking for 70% takes between 79 and 289 rounds to settle and ends between 99.4% and 99.7%, an almost complete split.

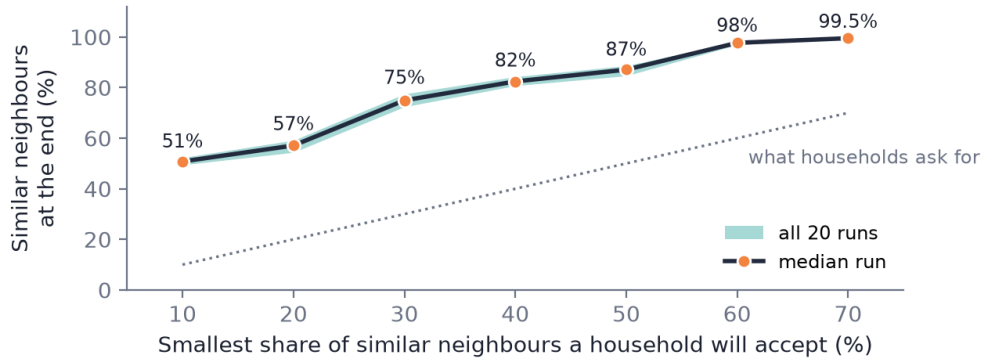


Figure 7. The final pattern is always more segregated than the preference that produced it (above the dotted line). The line is the median of 20 runs per setting and the band spans all of them. Computed for this guide.

When to use agent-based models

- When individuals differ from each other in ways that matter.
- When people react to their local surroundings and to each other.
- When you care about patterns that emerge, such as clusters, queues or crowds.

The costs are real. Agent-based models need many assumptions about behaviour, they can be slow, and their results vary from run to run. Run them many times with different random seeds and report the spread. Describe them with a standard template such as the **ODD protocol** (Overview, Design concepts, Details), so that others can check and rebuild them.

WATCH OUT

Schelling's model shows that mild preferences are *enough* to produce strong separation. It does not show that preferences cause the separation seen in any real city. Housing markets, income, history and discrimination all play their part.

IN ONE LINE

Agent-based models give individuals simple rules, then reveal the city-wide patterns that emerge from their interactions.

6 Cellular Automata

Growing a town on a grid, one cell at a time.

A **cellular automaton** (CA) divides land into a grid of square cells. Each cell has a **state**, such as empty, built-up or water. At each time step every cell looks at its **neighbourhood**, usually the eight cells around it, and a **transition rule** decides its next state. Growth models use this because new building tends to appear next to old.

Our growth model

Our grid has 80 × 80 cells, each a small plot, with a built-up patch in the middle. A river on the west side can never be built on, and a main road runs east from the centre. Each year an empty cell becomes built-up with a chance of 6% for every built-up neighbour. Within two cells of the road that chance is 3 times higher, and new settlements can start there. A tiny chance of growth anywhere allows the odd isolated building.

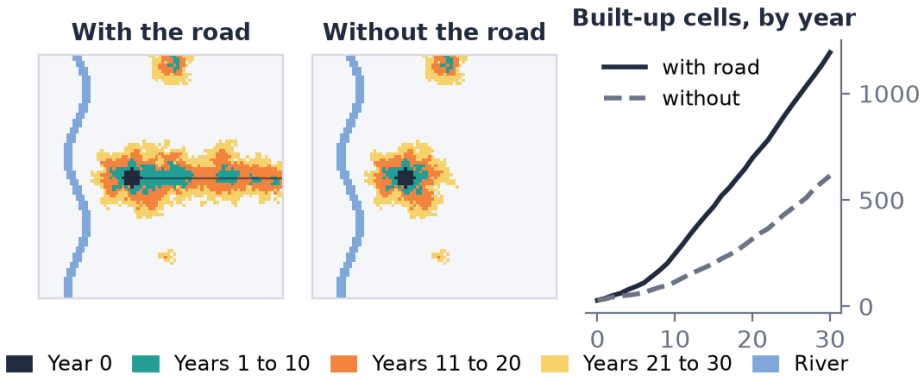


Figure 8. The growth model run for 30 years, coloured by when each cell was built. Left: with the road effect. Middle: the same rule without it. Right: built-up cells over time. Computed for this guide.

With the road, the town grows from 29 cells to 247, 697 and 1,192 at years 10, 20 and 30. The strip along the road covers 4% of the land, yet it takes 20% of all new building in this run, and between 15% and 24% across 20 more runs with other seeds. Without the road effect the town reaches 614 cells and stays rounder. The river blocks growth to the west in both runs.

Real growth models add layers such as slope, protected land, zoning and distance to jobs, and they are calibrated against historic maps. They are good at showing *where* growth might go and say little about *why*, so they are often combined with other models. Before trusting one, test it on the past. Start from an old map and check whether it grows something like today's town.

IN ONE LINE

Cellular automata grow a town cell by cell to explore where it may spread.

7 Gravity Models

Big places attract, distance deters.

Spatial interaction models estimate flows between places, such as commuting trips, shopping trips or migration. The best known is the **gravity model**. It borrows its shape from Newton's law of gravity. The flow between two places grows with their size and shrinks with the distance between them.

In words, the trips from home zone *i* to work zone *j* = (workers living in *i*) × (jobs in *j*) × (a **distance decay** factor) × (balancing factors). Our decay factor is exp(−β × distance), a number that equals 1 at zero distance and shrinks smoothly towards 0 as distance grows. The **parameter** β (beta) says how strongly distance puts people off. A bigger β means shorter trips.

The balancing factors make the totals add up. Every worker makes one trip to work, and every job is filled. A model with both totals fixed is called **doubly constrained**. The script finds the factors by scaling rows and columns in turn until both sets of totals match, a method called **iterative proportional fitting**.

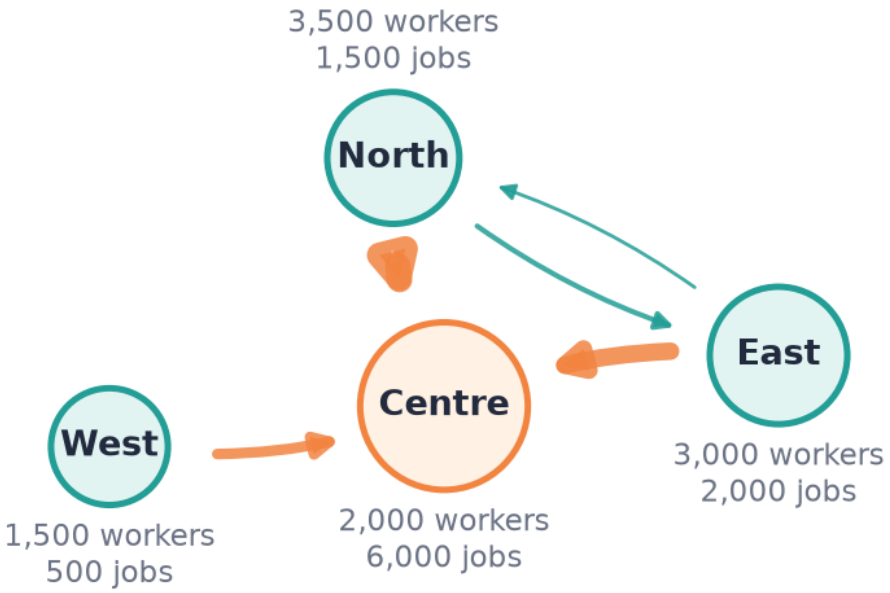


Figure 9. Four imaginary zones on a schematic map, not to scale. Arrow widths show the computed commuting trips (β = 0.15 per km). Only flows of 250 trips or more between different zones are drawn. Trips within a zone are not shown. Orange arrows go to the Centre.

Distances between zones (km)

From \ To	Centre	North	East	West
Centre	2	6	8	10
North	6	2	9	14
East	8	9	2	12
West	10	14	12	3

The diagonal is the typical length of a trip that stays inside a zone.

Trips from home zone (rows) to work zone (columns)

Home \ Work	Centre	North	East	West	Workers
Centre	1,547	198	202	52	2,000
North	2,103	894	431	71	3,500
East	1,461	294	1,155	90	3,000
West	889	114	212	286	1,500
Jobs filled	6,000	1,500	2,000	500	10,000

The Centre has 6,000 of the 10,000 jobs, so it draws most of the trips, even from far away. The average trip is 5.7 km. Doubling β, so that people dislike travel more, cuts the average to 4.8 km. Halving β stretches it to 6.3 km.

SAY IT LIKE THIS

“The gravity model captures a strong regularity. More opportunities attract more trips, and distance puts people off.”

WATCH OUT

β is no law of nature. It differs between places, trip purposes and years, so it must be estimated from local data. Chapter 12 shows how.

IN ONE LINE

Gravity models estimate flows between places from their size and the distance between them, with one parameter for how much distance deters.

8 Networks and Traffic

Roads as links, drivers as flows, jams as feedback.

Transport models represent streets as a **network** of **nodes** (junctions) joined by **links** (road segments). Each link has a length, a speed and a **capacity**, the flow per hour at which delays start to rise steeply. In simple models like this one, the flow can exceed the capacity, but each extra car then adds a lot of delay. Trips, often from a gravity model, are loaded onto the network, and each trip chooses a route. This step is called **traffic assignment**.

Congestion adds feedback. The more cars use a link, the slower it gets. Models capture this with a **volume-delay function**. A widely used one sets travel time = empty-road time × (1 + 0.15 × (flow ÷ capacity)^4). The fourth power makes delay climb steeply as a road fills up.

Two routes, one choice

Every hour 4,000 cars travel from a suburb to the centre. The ring road takes 15 minutes when empty and has a capacity of 2,200 cars per hour. The town road is shorter, 10 minutes when empty, with a capacity of 1,000. Each driver picks whichever route is quicker. Drivers keep switching until neither route is faster. This state is called **user equilibrium**, a principle set out by the transport researcher John Glen Wardrop in 1952.

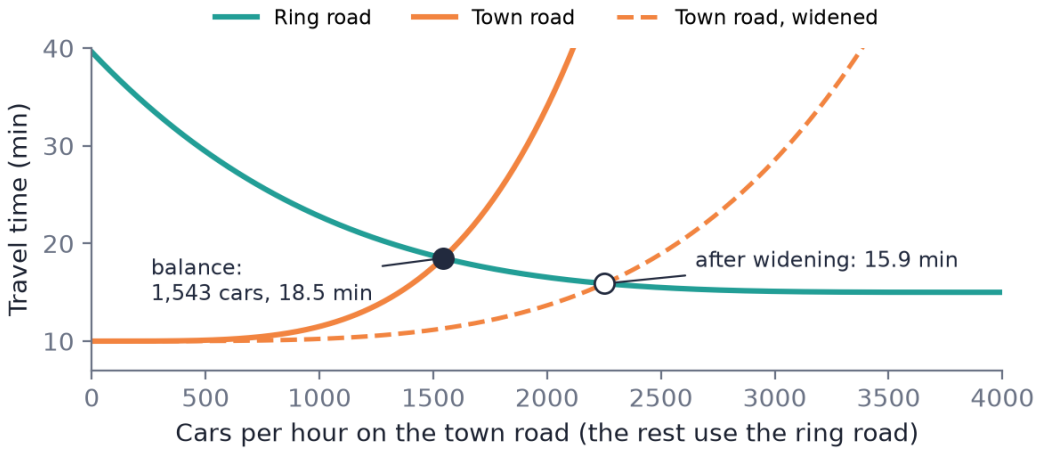


Figure 10. Travel time on each route as drivers shift between them. Where the curves cross, no driver can save time by switching. Computed for this guide.

The script finds the balance at 1,543 cars on the town road and 2,457 on the ring road. Both trips then take 18.5 minutes, longer than on empty roads, because both roads carry more than their capacity. If a planner could assign routes to minimise everyone's total time, the **system optimum**, the average trip would take 18.1 minutes. Each driver choosing their own quickest route costs everyone a little time here. In some networks the gap is large.

Widening the town road

Raise the town road's capacity to 1,600 and the balance moves to 2,253 cars on it, with both routes taking 15.9 minutes. Now suppose the faster trip tempts 10% more people to drive at this hour. The time creeps back up to 16.8 minutes. Extra traffic caused by extra capacity is called **induced demand**. It is one reason a transport model needs a demand side as well as a network.

Bigger tools

Real networks have thousands of links. **Macroscopic** models treat traffic as flows, as in our example. **Microscopic** models, such as SUMO, move each vehicle second by second, with car-following and lane changes. Agent-based transport models, such as MATSim, give each person a plan for the whole day and let them adjust routes and timings over many repeated runs.

> **IN ONE LINE**
> Network models load trips onto roads, where congestion feeds back on route choice until travellers settle into an equilibrium.

9 Discrete-Event Models

Arrivals, waiting and the cost of a busy system.

A **discrete-event simulation** (DES) models a system as a list of events in time, such as a bus arriving or a charger becoming free. The clock jumps from one event to the next. DES suits anything with arrivals, waiting and service, from bus stops and clinics to lifts and charging hubs.

How many chargers?

The owner of a car park plans a charging hub for electric cars. On a typical day 5 cars arrive per hour, at random times, for 12 hours. Each charges for 40 minutes on average. Cars join one queue and take the next free charger. The **offered load** is 5 × 40 ÷ 60 = 3.33 chargers' worth of work, so four chargers should cope on average. The script simulated 1,000 random days for each design, with the same random cars each time for a fair comparison.

Chargers	Demand ÷ capacity	Average wait	Wait over 10 min	1 driver in 20 waits longer than
3	111%	71.4 min	78%	210 min
4	83%	17.0 min	43%	68 min
5	67%	4.6 min	17%	27 min
6	56%	1.4 min	6%	11 min

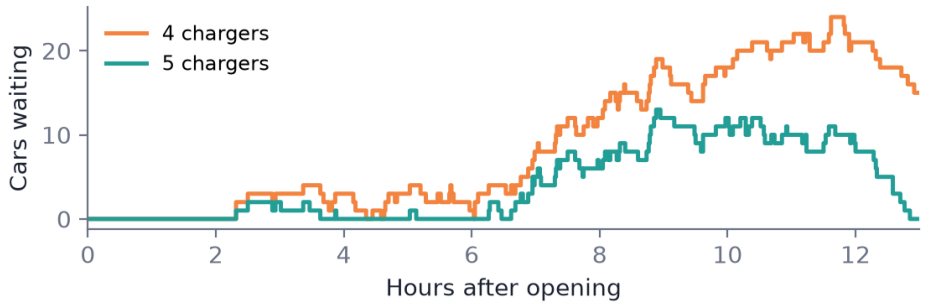


Figure 11. The queue during one simulated day, with four and with five chargers. Computed for this guide.

Three chargers cannot keep up, and the queue grows all day. Demand fills 83% of four chargers' time, which looks efficient, yet 43% of drivers wait more than 10 minutes. A fifth charger cuts the average wait from 17.0 to 4.6 minutes. Queues grow sharply as any system nears capacity, so never size a service by the average alone.

IN ONE LINE

Discrete-event models show how queues and waits grow as a system fills up.

10 Physics-Based Models

Energy, heat and air, in brief.

Some urban questions are about physical processes. How much energy will a building use? How hot will a street get? Where will wind gust around towers, or pollution collect? These need **physics-based models**, which solve equations for heat, air flow and radiation. They form a large field. This chapter gives a short tour.

Type	What it computes	Example question
Building energy simulation	Heat flows through walls, windows and ventilation, hour by hour, and the heating and cooling needed	Will extra insulation pay for itself?
Urban building energy	Energy use of many buildings at once, often from simplified building types	Where would a district heating network make sense?
Urban microclimate	Air and surface temperatures, humidity, wind and comfort in streets and parks	Do street trees make this square more comfortable?
Computational fluid dynamics (CFD)	The detailed flow of air around buildings	Will this tower make the plaza below too windy?

A tiny thermal model

The simplest building energy model treats a whole building as a single lump of thermal mass, like a stone that slowly warms and cools. Heat leaks in and out through the walls and windows. The sun and the people inside add heat. This is called a **lumped** or **resistance-capacitance** model, because it behaves like a simple electrical circuit. The script steps it forward one minute at a time over four hot days and plots the last one.

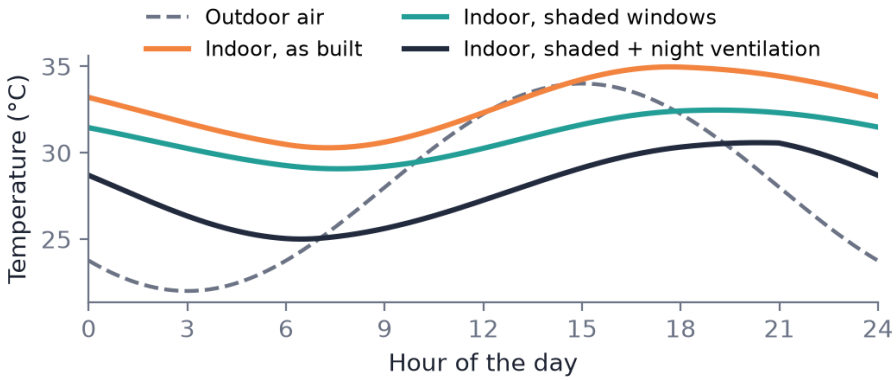


Figure 12. Indoor temperature of a simple lumped building with no air conditioning on a hot day. Computed for this guide.

Outdoors, the air swings between 22 and 34 °C. With no cooling, the indoor peak reaches about 35 °C, hotter than outside because sun through the windows and the people inside add heat. It comes about 3 hours after the outdoor peak, because the heavy building takes time to warm. Shading the windows to block 60% of the sun lowers the peak to about 32 °C. Adding night ventilation, which triples the heat exchange with cooler night air, brings it down to about 31 °C.

Full tools such as EnergyPlus model each room, wall layer and window, with real weather files and schedules of use. Microclimate tools such as ENVI-met, and CFD toolkits such as OpenFOAM, solve the flow of air and heat in three dimensions.

WATCH OUT

Physics models look precise because they use physical laws. Their answers still depend on inputs such as weather data, material properties and how people use the building. Check those before trusting the decimals.

IN ONE LINE

Physics-based models compute energy, heat and air flow from physical laws, for one building or a whole district.

11 Land Use and Transport Together

Where we live shapes how we travel, and the other way round.

Land use and transport are tied together in a loop. Where homes, jobs and shops are located shapes the trips people make. Those trips load the network and set travel times. Travel times decide how easy it is to reach things from each place, which is called **accessibility**. Accessibility, in turn, affects where households and firms choose to locate. **Land-use and transport interaction** (LUTI) models represent the whole loop.

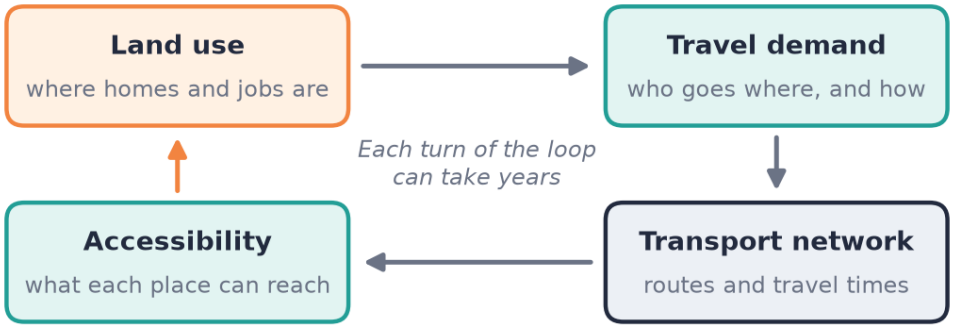


Figure 13. The land-use and transport cycle. A LUTI model goes round it again and again, often one year per turn.

Measuring accessibility

A common measure, based on an idea Walter Hansen published in 1959, adds up the opportunities you can reach from a place and discounts the far ones. Here we use the same exponential distance decay as the gravity model. For zone i, accessibility = the sum over every zone j of (jobs in j) × exp(−β × distance from i to j).

Worked example: a new rail line

Return to the four zones of Chapter 7. A new rail line links the West zone to the Centre and cuts the effective distance from 10 km to 4 km. Then 1,000 new households are placed in the zones in proportion to accessibility multiplied by free land. This is a crude location rule, but it shows the idea.

Zone	Free plots	Accessibility before	Accessibility after	New households before	New households after
Centre	100	5,769	5,932	151	128
North	300	4,130	4,130	325	269
East	300	3,760	3,760	296	245
West	400	2,172	4,126	228	358

West's accessibility rises by 90%, and the number of new households placed there rises from 228 to 358. The Centre gains a little too, because West's jobs are now closer. Because the total of 1,000 is fixed, zones whose accessibility does not change receive fewer. In a full LUTI model those new households would make new trips, which would change travel times, which would change accessibility again. The model keeps going round the loop.

Real LUTI models

Many LUTI models trace back to Ira Lowry's 1964 *Model of Metropolis* and to Alan Wilson's spatial interaction models. UrbanSim is a well-known land-use model whose core code is open source. It simulates where households and firms choose to locate and where developers build, and it is usually linked to a separate travel model.

SAY IT LIKE THIS

“A transport scheme is also a land-use policy. Over the years it changes where people choose to live and work.”

IN ONE LINE

LUTI models link land use and transport in one loop, so a new line changes accessibility, then where people locate, then how they travel.

12

Calibration and Validation

Tuning the dials, then testing on data the model has not seen.

Every model has **parameters**, such as β in the gravity model. **Calibration** means choosing their values so the model reproduces observed data. **Validation** means testing the calibrated model on different data. Calibration without validation is like marking your own homework.

Calibrating the gravity model

The script made a commuting survey for the zones of Chapter 7 from a gravity model with a hidden β of 0.12, plus random errors of about 15%. Then it tried every β from 0.01 to 0.40 and measured how far the model's trips were from the survey.

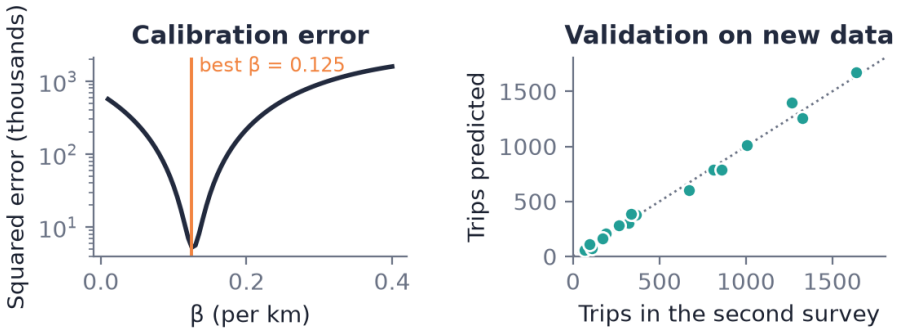


Figure 14. Left: the error is lowest near the hidden value of β. Right: the calibrated model predicts a second survey it never saw. Points near the dotted line are good predictions. Computed for this guide.

The error is smallest at β = 0.125, close to the hidden 0.12. The model then misses the survey by 15 trips per cell on average, about 2.5% of all trips. To validate it, the script made a second survey for a later year, after 1,000 jobs moved from the Centre to the East zone. Given only the new totals, the calibrated model misses it by 37 trips per cell, about 6.3%. Even the hidden true β misses it by 34 trips per cell, so most of that gap is the second survey's own random error and the model carries over well. Judging the error against the noise in the new data is the point of validation.

Good habits

- Split your data. Tune on one part and test on another.
- Match several observed patterns at once, known as **pattern-oriented modelling**.
- Beware of many free parameters. They let a model fit noise, which is **overfitting**.
- A good fit to the past is no guarantee for a future that a policy changes.

IN ONE LINE

Calibrate a model on one set of observations, validate it on another it has never seen, and report both.

13 Uncertainty and Sensitivity

How sure are we, and which assumptions matter most?

Every input to a model is uncertain. **Uncertainty analysis** asks how much the outputs could vary as a result. **Sensitivity analysis** asks which inputs drive that variation.

Monte Carlo runs

The **Monte Carlo** method runs a model many times, each time with inputs drawn at random from plausible ranges. We ran the Chapter 4 housing model 500 times, drawing the moving-in rate between 11% and 13%, the building response time between 1 and 3 years, and the land limit between 30,000 and 38,000 homes.

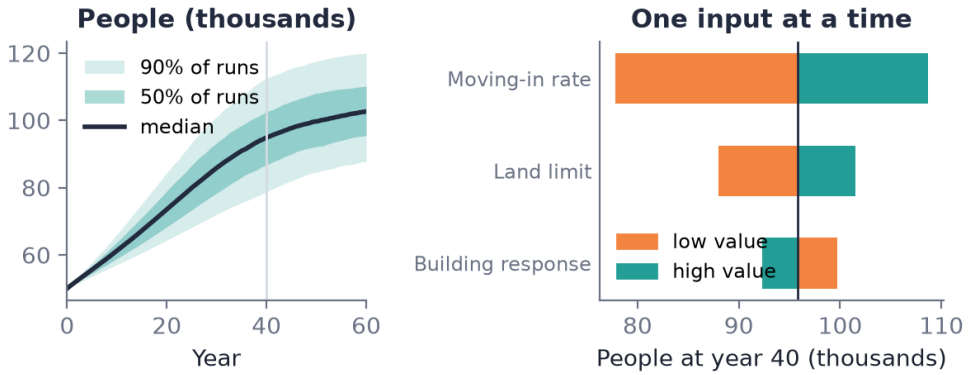


Figure 15. Left: 500 runs of the housing model, shown as bands. Right: a tornado chart for population at year 40. Computed for this guide.

At year 40 the median run has 94,900 people. Half of the runs fall between 86,800 and 102,500, and 90% fall between 78,800 and 112,500.

The tornado chart

A simple sensitivity test moves one input to its low and then its high value while the others stay at their central values. Sorted by length, the bars look like a tornado. Here the moving-in rate matters most and the building response least, so better data on the moving-in rate would pay off most. One-at-a-time tests miss interactions between inputs. **Global** methods, such as variance-based (Sobol) analysis, vary all the inputs together.

IN ONE LINE

Run the model across plausible inputs to show the range of outcomes, and use sensitivity analysis to find the assumptions that matter.

14

Scenarios and Forecasts

Exploring several possible futures instead of predicting one.

A **forecast** is a statement about what will most likely happen. A **projection** says what would happen if current trends continued. A **scenario** is a coherent, plausible story about the future, used to ask “what if?”. Urban futures depend on the economy, technology, climate and politics, which no model can predict over decades. So modellers usually build a few contrasting scenarios and test their options against all of them.

Options across scenarios

We ran the housing model under three scenarios for the wider region. In a **boom**, the moving-in rate is 13.5%. In **steady** conditions it is 12%. In a **slowdown** it is 10.5%. In each scenario we compared doing nothing with two options. One releases land for 10,000 more homes at year 20. The other speeds up planning approvals, so builders close a shortage in one year instead of two.

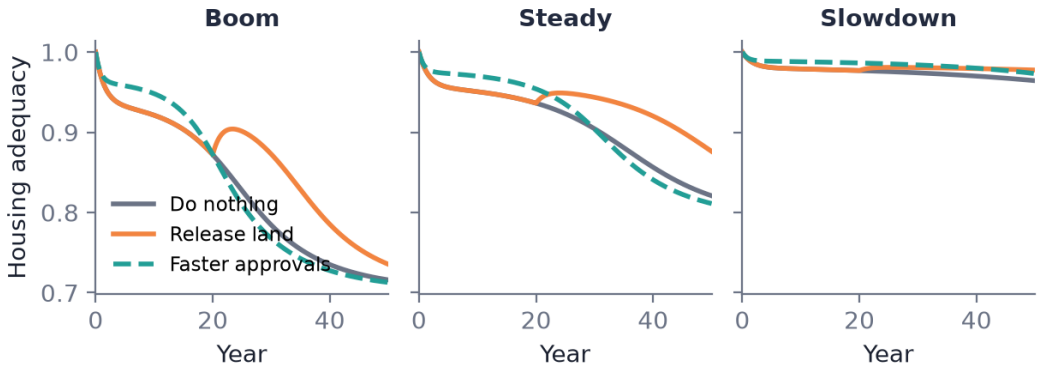


Figure 16. Housing adequacy under three scenarios and three options. Computed for this guide.

Results at year 50

Option	Boom	Steady	Slowdown
Do nothing	117,300 people adequacy 0.72	101,200 people adequacy 0.82	72,400 people adequacy 0.96
Release land	147,500 people adequacy 0.74	119,500 people adequacy 0.88	74,200 people adequacy 0.98
Faster approvals	118,500 people adequacy 0.71	103,700 people adequacy 0.81	75,800 people adequacy 0.97

Releasing land improves adequacy at year 50 in every scenario, by 0.02 in the boom, 0.06 in steady conditions and 0.02 in the slowdown. In the boom and steady scenarios faster approvals barely change the year-50 result, because land is the real limit. In the slowdown, where land is not yet short, faster approvals do about as much as

releasing land. The scenarios matter more than either option. No option stops the boom from bringing crowding, so a planner would also look at options this model leaves out.

Robust choices

An option that performs reasonably well in every scenario is called **robust**. It is often a better bet than the option that is best in one scenario and poor in the others. Planning approaches built on this idea are known as **decision making under deep uncertainty**.

> **SAY IT LIKE THIS**
> “These runs explore several plausible futures. They show how each option holds up across all of them.”

> **IN ONE LINE**
> Use scenarios to explore several plausible futures, and prefer options that hold up well across them all.

15 Communicating Honestly

Show the range, state the assumptions, avoid the classic traps.

Model results travel. A chart from a technical report ends up in a slide, then in a headline, losing its caveats on the way. Honest communication protects your audience and your decision.

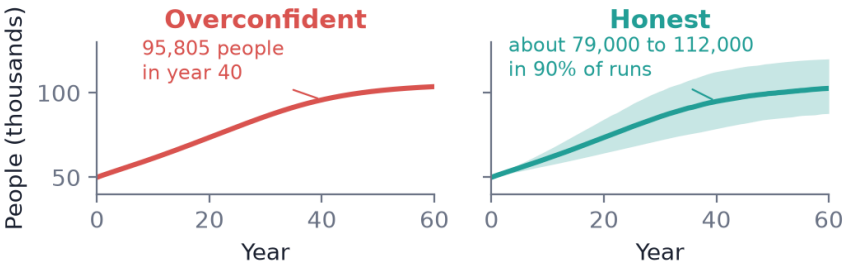


Figure 17. The same housing model shown two ways. Left: one run with a precise-looking number. Right: 500 runs with a range. Computed for this guide.

A checklist for sharing results

- State the question the model was built for, and the questions it cannot answer.
- List the main assumptions in plain words.
- Show ranges and say where they come from. Round to the precision you have. “About 95,000, probably between 79,000 and 112,000” beats “95,805”.
- Say how the model was calibrated and validated, and share the code and data where you can.

Classic pitfalls

Pitfall	What it looks like	Remedy
Wrong question	A traffic model judging health	Write the question first
False precision	Results quoted to the last person	Round and show ranges
Too much detail	Months on inputs that barely matter	Start simple
One run only	A random model reported once	Run many seeds
Forecast confusion	“The population will be…”	Say “under this scenario”
Missing feedback	Fixed demand after road widening	Model the loop, or flag it

IN ONE LINE

Share results with ranges, plain assumptions and honest limits, so they inform decisions without pretending to certainty.

16

Try It Yourself

A short recipe, the tools, and an eight-week plan.

The best way to understand modelling is to build a small model yourself. You need a laptop, a clear question and a free afternoon. Here is a way to start.

A recipe for your first model

1. Write one question in one sentence.
2. Sketch the system on paper, with its stocks and flows, agents or zones, and the loops between them.
3. Pick the simplest model family that can answer the question.
4. Build the smallest version that runs. Test it on cases where you know the answer.
5. Compare it with real data. Calibrate, then validate on separate data.
6. Run scenarios and a sensitivity analysis, with many random seeds if the model is random.
7. Write up the question, assumptions, ranges and limits. Share the code if you can.

Tools you can use

Tool	Family	What it is
NetLogo	Agent-based, cellular automata	Free modelling environment with a simple language and a large library of example models, including a segregation model
Mesa	Agent-based	Python library for building agent-based models
GAMA	Agent-based with maps	Open-source platform that works well with map data
Vensim, Stella	System dynamics	Desktop tools for drawing and running stock-and-flow models. Check the licence terms
PySD	System dynamics	Python library that runs system dynamics models
SimPy	Discrete-event	Python library for process-based discrete-event simulation
AnyLogic	Several	Commercial tool that combines system dynamics, agent-based and discrete-event models
SUMO	Traffic	Open-source microscopic simulator of vehicles, pedestrians and public transport
MATSim	Transport, agent-based	Open-source framework that simulates the daily plans of many travellers

Tool	Family	What it is
UrbanSim	Land use	Land-use modelling framework with open-source code for household, firm and development location models
EnergyPlus	Building energy	Free, open-source whole-building energy simulation engine
OpenFOAM	CFD	Open-source toolbox for simulating fluid flow, including wind around buildings
QGIS	Maps and data	Free geographic information system for preparing inputs and mapping results
Python with NumPy and Matplotlib	Any	Every example in this guide was built this way

An eight-week plan

Week	Focus	Do this
1	Systems thinking	Draw stocks, flows and causal loops for your own street or town. Find one reinforcing and one balancing loop.
2	System dynamics	Rebuild the housing model of Chapter 4 in a spreadsheet or in Python. Predict the effect of changing one rate, then run it.
3	Agent-based models	Play with a segregation model in NetLogo. Change the preference and the share of empty cells.
4	Cellular automata	Code a growth grid. Add a road and a protected park, and compare the maps.
5	Gravity models	Build a four-zone trip model with iterative proportional fitting. Vary β and watch the average trip length.
6	Networks and queues	Find the equilibrium on two routes. Then simulate a queue and size a service.
7	Calibration and uncertainty	Fit β to a made-up survey and test it on a second one. Run the model many times with random inputs (Monte Carlo) and draw the range of results as shaded bands.
8	A small project	Pick a real local question. Build, test, run scenarios and write a two-page honest summary.

IN ONE LINE

Start with one question and the smallest model that can answer it, then test, explore and share it honestly.

Myths and Facts

Common misunderstandings, cleared up.

Myth	Fact
A model predicts the future.	A model shows what follows from its assumptions. Over long horizons it is best used to compare options across scenarios.
More detail always makes a model better.	Detail needs data, adds run time and adds places for errors to hide. The right level depends on the question.
If a model fits past data, it is right.	A good fit is a start. The model must also do well on data it has never seen, and the future may differ from the past.
Simulation results are objective.	Every model contains choices about what to include and leave out. Making those choices visible is part of the job.
Agent-based models are always more realistic.	They can represent individuals, but each behaviour rule is an assumption. For totals, simpler aggregate models are often as good and easier to check.
Wider roads always cure congestion.	Extra capacity can attract extra trips, known as induced demand. A model needs a demand side to show this.

Quick Quiz

Ten questions. Answers are on the next page.

1. What is the difference between a stock and a flow?
2. Name the two kinds of feedback loop and say what each does.
3. In the housing model, why did releasing land not end crowding?
4. In our Schelling run, households wanted at least 30% similar neighbours. What happened?
5. Which model family would you use to estimate waiting times at a new bus interchange?
6. In a gravity model, what happens to the average trip length when β increases?
7. What is user equilibrium?
8. Why must validation use different data from calibration?
9. What does a tornado chart show?
10. What is the difference between a scenario and a forecast?

Quiz Answers

No peeking until you have tried.

1. A stock is an amount at a moment in time, such as the homes in a town. A flow is a rate that changes it, such as homes built per year.
2. A reinforcing loop amplifies change. A balancing loop pushes back towards a limit or a goal.
3. The extra homes made the town more attractive, so more people moved in until crowding returned towards the balance point.
4. The average share of similar neighbours rose from about 52% to 77%. A mild preference produced a strongly divided pattern.
5. A discrete-event simulation, because the question is about arrivals, queues and service.
6. It gets shorter, because distance puts people off more strongly.
7. The state of a road network in which no driver can save time by switching to another route.
8. Otherwise you only test whether the model can reproduce the data it was tuned on, which says little about its predictions.
9. How much an output changes when each input is moved, one at a time, from its low to its high value. The longest bars matter most.
10. A forecast claims what is most likely to happen. A scenario is a plausible “what if” story used to test options.

Glossary

Plain definitions of the words in this guide.

Term	Meaning
Accessibility	How many opportunities, such as jobs or shops, can be reached from a place, with far ones counting less.
Agent-based model (ABM)	A model built from many agents that follow rules and interact with each other.
Balancing loop	A feedback loop that pushes a system back towards a limit or goal.
Calibration	Choosing parameter values so that a model reproduces observed data.
Capacity	The flow on a road at which delays start to rise steeply.
Cellular automaton (CA)	A grid of cells whose states change according to rules based on neighbouring cells.
Computational fluid dynamics (CFD)	Computer simulation of how air or water flows, for example around buildings.
Discrete-event simulation (DES)	A simulation that jumps from one event, such as an arrival, to the next.
Distance decay	The tendency for interaction between places to fall as distance grows.
Doubly constrained	A trip model in which both the trips leaving each zone and the trips arriving in each zone are fixed.
Emergence	A pattern at the scale of the whole system that arises from many small interactions.
Feedback loop	A chain of causes and effects that returns to its starting point.
Flow	A rate that fills or drains a stock, such as births per year.
Forecast	A statement of what is most likely to happen.
Gravity model	A model in which flows grow with the size of places and fall with the distance between them.
Induced demand	Extra travel that appears when travel becomes faster or cheaper.
Iterative proportional fitting	Scaling the rows and columns of a table in turn until both sets of totals match.
LUTI model	A land-use and transport interaction model, linking where people locate with how they travel.

Term	Meaning
Monte Carlo method	Running a model many times with randomly drawn inputs to see the range of outcomes.
Offered load	The work arriving at a service, counted as the number of servers it would keep busy on average.
Overfitting	Tuning a model so closely to past data that it fits the noise and predicts poorly.
Parameter	A number in a model that sets how strongly something happens, such as β.
Projection	What would happen if current trends continued.
Random seed	A starting number for a random generator. The same seed gives the same random sequence.
Reinforcing loop	A feedback loop that amplifies change in the same direction.
Robust option	An option that performs reasonably well in every scenario.
Scenario	A coherent, plausible story about the future, used to explore “what if” questions.
Sensitivity analysis	Testing how much each input affects a model's outputs.
Simulation	Running a model forward in time to see how it behaves.
Stock	An amount that accumulates and can be counted at a moment, such as the homes in a town.
System dynamics	A modelling method built from stocks, flows and feedback loops.
System optimum	The route split that minimises everyone's total travel time.
Tornado chart	Bars showing how much an output changes as each input moves from its low to its high value.
Traffic assignment	Loading trips onto a network so that each one chooses a route.
User equilibrium	A traffic state where no traveller can save time by changing route.
Validation	Testing a calibrated model against data it was not tuned on.
Volume-delay function	A formula linking traffic flow on a road to its travel time.

One-Page Cheat Sheet

Pin it next to your screen.

Idea	In short
Model	A simplification you can run, built to answer one question.
Stocks and flows	Amounts that accumulate, and the rates that fill and drain them.
Feedback	Reinforcing loops amplify change. Balancing loops push back towards limits.
System dynamics	Totals over years and decades. Good for loops, delays and quick fixes that fade.
Agent-based	Individuals with rules. Good for emergent patterns. Run many seeds.
Cellular automata	Cells change with their neighbours. Good for maps of possible growth.
Gravity model	Trips grow with size and fall with distance. One parameter, β, sets the decay.
Traffic assignment	Trips choose routes. Congestion feeds back until user equilibrium.
Discrete-event	Arrivals, queues and service. Waits grow steeply as a system fills up.
Physics-based	Energy, heat and air from physical laws. Inputs decide the quality.
LUTI	Land use shapes travel, travel shapes accessibility, accessibility shapes land use.
Calibrate, validate	Tune on one dataset. Test on another. Report both.
Uncertainty	Many runs, shown as a range. A tornado chart finds the inputs that matter.
Scenarios	Several plausible futures. Prefer options that are robust across them.

Further Reading

Where the ideas come from.

These references are written from memory. Please check titles, years and details before you cite them.

- Box, G. E. P. (1976). Science and statistics. *Journal of the American Statistical Association*, 71(356), 791–799.
- Box, G. E. P. (1979). Robustness in the strategy of scientific model building. In R. L. Launer and G. N. Wilkinson (Eds.), *Robustness in Statistics* (pp. 201–236). Academic Press.
- Forrester, J. W. (1969). *Urban Dynamics*. MIT Press.
- Meadows, D. H. (2008). *Thinking in Systems: A Primer*. Chelsea Green.
- Sterman, J. D. (2000). *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Irwin/McGraw-Hill.
- Schelling, T. C. (1971). Dynamic models of segregation. *Journal of Mathematical Sociology*, 1(2), 143–186.
- Grimm, V. et al. (2006). A standard protocol for describing individual-based and agent-based models. *Ecological Modelling*, 198(1–2), 115–126.
- Batty, M. (2005). *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-Based Models, and Fractals*. MIT Press.
- Lowry, I. S. (1964). *A Model of Metropolis* (RM-4035-RC). RAND Corporation.
- Wilson, A. G. (1970). *Entropy in Urban and Regional Modelling*. Pion.
- Hansen, W. G. (1959). How accessibility shapes land use. *Journal of the American Institute of Planners*, 25(2), 73–76.
- Wardrop, J. G. (1952). Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers, Part II*, 1(3).
- Ortúzar, J. de D. and Willumsen, L. G. (2011). *Modelling Transport* (4th ed.). Wiley.
- Railsback, S. F. and Grimm, V. (2019). *Agent-Based and Individual-Based Modeling: A Practical Introduction* (2nd ed.). Princeton University Press.
- Saltelli, A. et al. (2008). *Global Sensitivity Analysis: The Primer*. Wiley.