MA621L-Mathematical Modelling

Model-5 : Discrete Models, Leaves on Trees and Fans on Classroom

Panchatcharam Mariappan[1]

[1]Associate Professor
Department of Mathematics and Statistics
IIT Tirupati, Tirupati

November 2, 2026



भारतीय प्रौद्योगिकी संस्थान तिरुपति
IIT
TIRUPATI



Why Discrete?

What You Collected



Last week you went round the campus and counted:

- the **switches**, **fans** and **data ports** in classrooms, and were asked to turn this into a number for the whole institute;

For the exam, you will count the **leaves** on a tree.

Every one of these answers is a whole number. A room cannot have 5.7 fans. A tree is built one fork at a time, and leaves come one season at a time.

> **The question for today**
> In the scaling lecture, we wrote smooth scaling laws such as "fans $\propto$ floor area". What changes when we respect the fact that the quantities are *counts* and the processes happen in *steps*?

That is the subject of discrete models.

What Is a Discrete Model?



Definition 1 (Discrete model)

A discrete model describes a system whose state takes values in a countable set $S = \{s_1, s_2, \dots\}$, or changes in distinct steps $k = 0, 1, 2, \dots$, or both. The step index need not be time: it can be a generation, a season, or a level of branching.

What Is a Discrete Model?



General form

- State $x_k \in S$, input $u_k \in U$
- Deterministic update: $x_{k+1} = F(x_k, u_k)$
- Stochastic update: $P(X_{k+1} = j \mid X_k = i) = p_{ij}$

Representations: difference equation, graph or network, state transition diagram, transition matrix.

What Is a Discrete Model?

Example	Update rule
birth–death	$X_{k+1} = X_k + B_k - D_k$
inventory	$I_{k+1} = \max\{I_k + A_k - D_k, 0\}$
queue	$Q_{k+1} = \max\{Q_k + A_k - \mu_k, 0\}$
logistic map	$x_{k+1} = r\, x_k(1 - x_k)$
cellular automaton	cell_{k+1} = rule(neighbours_k)

Both your campus tasks are discrete: switches, fans and leaves are *counts*. A tree is also built in steps, one fork at a time.

Discrete Models at a Glance

DISCRETE MODELS

Mathematical models for systems with distinct, countable states and events

WHAT ARE DISCRETE MODELS?

- Describe systems that evolve in discrete steps or take values in discrete sets.
- Capture structure, logic, and combinatorics of complex systems.
- Enable exact analysis, efficient algorithms, and computer implementation.

EXAMPLES OF DISCRETE MODELS

Population (Birth–Death) $X_{t+1} = X_t + B_t - D_t$	$B_t, D_t \sim$ Poisson or Binomial
Inventory System $I_{t+1} = \max\{I_t + A_t - D_t, 0\}$	Discrete arrivals and demands
Queueing System $Q_{t+1} = \max\{Q_t + A_t - \mu_t, 0\}$	Customers arrive and are served
Random Walk on a Graph $X_{t+1} = N(X_t)$	Move to a neighboring vertex
Combinatorial Optimization Find $x \in \mathcal{F} \subseteq \{0,1\}^n$	E.g., TSP, Set Cover, Scheduling

GENERAL FORM

A discrete model is defined on a countable set $S = \{s_1, s_2, \ldots\}$

State: $x_t \in S$

Input: $u_t \in U$ (discrete)

Time: $t = 0, 1, 2, \ldots$

State update (dynamical models):

$$x_{t+1} = F(x_t, u_t)$$

or stochastic transition:

$$P(X_{t+1} = j \mid X_t = i)$$

REPRESENTATIONS

Graph / Network



$S = \{1, 2, 3, 4, 5\}$

$E = \{(1,2), (1,3), (2,4), (3,4), (4,5)\}$

State Transition Diagram



Transition Matrix (Markov Chain)

$P =$	1	2	3	4	5
1	0	1	1	0	0
2	0	0	0	1	0
3	0	0	0	1	0
4	0	0	0	0	1
5	0	0	0	1	0

WHY DISCRETE MODELS?

- **Interpretability** — States and transitions are explicit and easy to understand.
- **Computability** — Enable exact algorithms, dynamic programming, and enumerative methods.
- **Flexibility** — Capture logical rules, combinatorial choices, and complex constraints.
- **Scalability** — Leverage graph theory, sparse structures, and decomposition techniques.

MODEL BUILDING WORKFLOW

1. **Problem Definition** — Identify system, variables, assumptions, and objectives.
2. **State & Parameter Specification** — Define state space S, inputs U, and parameters.
3. **Model Formulation** — Specify transition rules or update equations. $f(x)$
4. **Analysis / Solution** — Use analytical methods or algorithms to study and solve.
5. **Validation & Use** — Validate with data, refine model, and support decisions.

APPLICATIONS

- **Supply Chain & Logistics** — Routing, facility location, inventory control.



- **Social & Communication Networks** — Information spread, influence maximization.



- **Healthcare** — Disease modeling, patient flow, treatment pathways.



- **Cybersecurity** — Attack graphs, intrusion detection, risk assessment.



- **Education** — Student performance, curriculum planning, course scheduling.



- **Engineering & Reliability** — System reliability, fault trees, maintenance scheduling.



- **Economics & Finance** — Market microstructure, credit risk, option lattices.

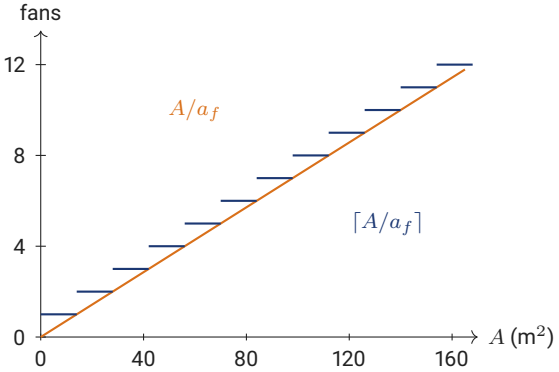


DISCRETE MODELS bridge real-world complexity and mathematical rigor. *From states and rules to insight and impact.*

Real-World System → Discretization → Discrete Model → Computation → Insight & Decision

Same Question, Continuous vs. Discrete

Question: how many ceiling fans does a room of floor area A need, if one fan serves an area a_f?



$a_f = 14$ m^2 (illustrative)

Same Question, Continuous vs. Discrete



Continuous model: $n_f = A/a_f \propto L^2$.

Discrete model:

$$n_f = \left\lceil \frac{A}{a_f} \right\rceil$$

- The staircase never falls below the line: you round *up*, or part of the room is not served.
- The relative gap is up to $1/n_f$: large for small rooms, negligible for halls.
- So the L^2 law is the large-room limit of the discrete model.



Switches, Fans and Data Ports

Model A: One Room, Integer Rules



Room $\ell \times w$: area $A = \ell w$, perimeter $P = 2(\ell + w)$. Design rules (check them against your data):

$$n_f = \left\lceil \frac{A}{a_f} \right\rceil$$ fans: one per area a_f

$$n_\ell = \left\lceil \frac{A}{a_\ell} \right\rceil$$ lights: one per area a_ℓ

$$n_s = n_f + n_\ell$$ switches: one per fan or light

$$n_p = \left\lfloor \frac{P - w_d}{s_p} \right\rfloor$$ wall data ports: spacing s_p, door width w_d

Model A: One Room, Integer Rules



Worked example (illustrative values)
10×8 m room, $A = 80$ m^2, $P = 36$ m; $a_f = 14$ m^2, $a_\ell = 8$ m^2, $s_p = 3$ m, $w_d \approx 0$: $n_f = \lceil 5.7 \rceil = 6$, $n_\ell = 10$, $n_s = 16$, $n_p = 12$.

Fans and switches grow with area (L^2), wall ports with perimeter (L): Chapter 3's exponents as staircases.

Model B: From Rooms to the Institute

Counting walk as a difference equation. Visit rooms $j = 1, 2, \ldots, R$ and keep a running total:

$$N_j = N_{j-1} + c_j, \qquad N_0 = 0 \quad \Longrightarrow \quad N_R = \sum_{j=1}^{R} c_j.$$

Counting every room is a census. A **model** predicts the rooms not visited.

Stratified estimate. Group rooms into types k (tutorial, hall, lab). Type k has R_k rooms, of which m_k were sampled, with sample mean $\bar{c}_k$ and standard deviation s_k:

$$\hat{N} = \sum_k R_k \, \bar{c}_k, \qquad \mathrm{SE}(\hat{N}) = \sqrt{\sum_k R_k^2 \, \frac{s_k^2}{m_k} \Big(1 - \frac{m_k}{R_k}\Big)}.$$

Model B: From Rooms to the Institute

- The factor $1 - m_k/R_k$ is the finite-population correction: if you count every room, the error is zero.
- Report $\hat{N} \pm 2\,\mathrm{SE}$, not a single number.
- Better still: predict each unvisited room from its area with Model A, then add.



Worked Numbers (Illustrative)



Room type	R_k	sampled m_k	fans/room ($\bar{c}_k \pm s_k$)	switches/room	ports/room
tutorial room	40	5	6 ± 0.7	16	12
lecture hall	10	3	16 ± 2.0	40	24
laboratory	20	4	10 ± 1.5	25	40
Institute total	70	12	**600**	**1540**	**1520**

Worked Numbers (Illustrative)



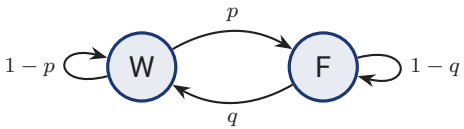
- Fans: $\hat{N} = 40(6) + 10(16) + 20(10) = 600$, $\mathrm{SE} \approx 20$, so 600 ± 40.
- Switches: $40(16) + 10(40) + 20(25) = 1540$. Data ports: $40(12) + 10(24) + 20(40) = 1520$.
- Labs dominate the data ports: one port per workstation, not per metre of wall. Model A's perimeter rule fails for labs, so labs need their own rule.

Replace with your data

These numbers are made up to show the method. Put in your own counts, room numbers per type from the estates office, and your own s_k.

Model C: How Many Fans Are Working Today?

Each fan is in one of two states: **W** (working) or **F** (faulty). Each month a working fan fails with probability p; a faulty fan is repaired with probability q.



Model C: How Many Fans Are Working Today?

$$P = \begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix}$$

Fraction working in month t, w_t:

$$w_{t+1} = (1-p)\, w_t + q\,(1-w_t).$$

Fixed point $w^* = w^*(1-p) + q(1-w^*)$:

$$\boxed{w^* = \frac{q}{p+q}}$$

Error $e_t = w_t - w^*$ obeys $e_{t+1} = (1-p-q)\, e_t$: the approach is geometric, and stable since $|1-p-q| < 1$. Illustrative: $p = 0.02$, $q = 0.5 \Rightarrow w^* \approx 0.96$. Of 600 fans, about 577 work on a typical day. Halving repair time matters more than buying better fans when q is small.

Classroom Model: Ledger and Classification



Dym's questions

Question	Classroom model
Why?	Budget for maintenance and network upgrades.
Find?	Institute totals of switches, fans and data ports, with a range.
Given?	Counts in sampled rooms; list of rooms by type.
How?	Integer design rules, stratified sum, two-state chain.
Valid?	Totals are integers; zero rooms gives zero.
Verified?	Compare with estates/electrical and network inventory records.

Classroom Model: Ledger and Classification



Assumption ledger

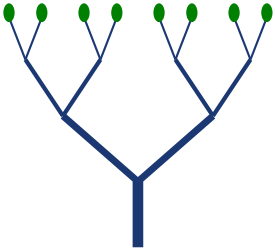
- A1: Rooms of one type are alike (same a_f, a_ℓ, s_p).
- A2: Sampled rooms are typical of their type (no choosing the nearest rooms only).
- A3: A switch is one toggle, not one board. Write your definition.
- A4: Broken and disconnected items are counted (or not) consistently.
- A5: Failure and repair probabilities p, q are the same every month and for every fan.

Velten SQM. S physical, technical; Q phenomenological (design rules), stationary; M discrete, integer-valued, deterministic (A, B) and stochastic (C: Markov chain).



Counting Leaves on a Tree

Leaf Model 1: The Tree as a Branching Recursion



$k = 3, T_3 = 8$

$k = 2, T_2 = 4$

$k = 1, T_1 = 2$

$k = 0, T_0 = 1$

Leaf Model 1: The Tree as a Branching Recursion

State: $T_k =$ number of branches at branching order k (trunk $k = 0$).
Rule: every branch splits into b daughters,

$$T_{k+1} = b\,T_k, \quad T_0 = 1 \implies T_n = b^n.$$

With λ leaves per terminal twig after n levels:

$$\boxed{N = \lambda\, b^n}$$

A linear difference equation in the order k, not in time.

Errors grow with every level
$\frac{\delta N}{N} \approx n\,\frac{\delta b}{b}$. With $n = 6$ and $b = 3 \pm 0.3$ (10%), the count is uncertain by about 60%.

Stochastic version: if each branch has a random number of daughters with mean μ, this is a Galton–Watson branching process, $E[T_n] = \mu^n$. Simulate it to get a range, not one number.

Leaf Model 2: Add a Conservation Rule



Leonardo da Vinci's rule: at each fork, the cross-sectional area of the parent equals the total area of the daughters,

$$d_k^2 = \sum_{\text{daughters}} d_{k+1}^2 \qquad \xrightarrow{b \text{ equal daughters}} \qquad d_{k+1}^2 = \frac{d_k^2}{b}.$$

This is again a difference equation.

Leaf Model 2: Add a Conservation Rule

Apply it n times from the trunk ($d_0 = D$) to the terminal twig ($d_n = d$):

$$d^2 = \frac{D^2}{b^n} \quad \Longrightarrow \quad b^n = \left(\frac{D}{d}\right)^2 \quad \Longrightarrow \quad \boxed{N = \lambda\left(\frac{D}{d}\right)^2}$$

- b and n cancel. Only two diameters (tape and vernier) and λ are needed.
- The girth^2 law (pipe model) is now *derived*, not assumed. A scaling law with exponent 2 comes out of a discrete recursion.
- Illustrative: $D = 30$ cm, $d = 0.5$ cm, $\lambda = 20 \Rightarrow 3600$ twigs, about $72{,}000$ leaves.

Assumption traded: we no longer need b and n, but we need area conservation at forks. Check it by measuring ten forks.

Leaf Model 3: Leaves Through the Seasons



Now the step t is a growing season. Let T_t be the number of growing tips.

- Every tip branches each season: $T_{t+1} = r\,T_t$ (geometric growth).
- A new shoot rests one season before it branches: $T_{t+1} = T_t + T_{t-1}$. Fibonacci, so $T_t \sim \varphi^t$, $\varphi \approx 1.618$.
- Leaves are made and shed (fraction s per season): $N_{t+1} = (1-s)N_t + \lambda$ (new tips).
- Inner leaves are shaded, so the count saturates:

$$N_{t+1} = N_t + r\,N_t\Big(1 - \frac{N_t}{K}\Big), \qquad K \approx \frac{\text{LAI} \times \text{crown ground area}}{\text{area of one leaf}}.$$

Leaf area index (LAI): leaf area per unit ground area, typically 3–6 for trees. K gives an independent check on Model 2.

Leaf Model 3: Leaves Through the Seasons



Same equation as the bifurcation diagram

Rescaled, this is the logistic map $x_{t+1} = r\,x_t(1-x_t)$. For large r it oscillates and becomes chaotic: a warning that discrete models can behave very differently from their continuous versions.

Building the Leaf Model: Workflow and Ledger

1. Problem definition	Estimate N for one campus tree, with an uncertainty range.
2. States, parameters	T_k, d_k; parameters b, n, λ, D, d.
3. Formulation	$T_{k+1} = bT_k$; $d_{k+1}^2 = d_k^2/b$; $N = \lambda(D/d)^2$.
4. Analysis	Compute N both ways; propagate errors; simulate Galton–Watson.
5. Validation	Count one sapling or one whole branch completely and compare.

Building the Leaf Model: Workflow and Ledger



Ledger: A1 equal splitting at forks; A2 area conserved at forks; A3 similar λ on all twigs; A4 no dead or broken branches; A5 unit is a leaf, not a leaflet; A6 one season only.

Velten SQM: S physical, natural; Q mechanistic (branching) with phenomenological λ; M discrete, linear difference equation, deterministic or stochastic.

Field procedure

Measure parent and daughter diameters at $8{-}10$ forks (test A2); count daughters per fork (b); count leaves on $10{-}15$ twigs (λ); measure D and d. Compute N from λb^n and from $\lambda(D/d)^2$, and explain why they differ.



A Simple Cellular Automaton

Problem: A Row of Students with Cards



17 students stand in a row. Each holds a card: **up** (1) or **down** (0). At first only the middle student's card is up. Every time the teacher claps, *all students at once* look at their left and right neighbours and apply one rule: **Rule:** card up if **exactly one** of my two neighbours has a card up; otherwise down.

Definition 2 (Elementary cellular automaton)
Cells $i = 1, \ldots, n$ on a line, states $x_i^t \in \{0, 1\}$, neighbourhood (x_{i-1}, x_i, x_{i+1}), and one rule $x_i^{t+1} = f(x_{i-1}^t, x_i^t, x_{i+1}^t)$ used by every cell at the same time.

Problem: A Row of Students with Cards

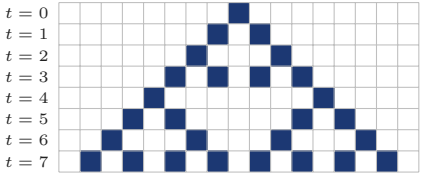
Our rule ignores the student's own card:
$x_i^{t+1} = x_{i-1}^t \oplus x_{i+1}^t = (x_{i-1}^t + x_{i+1}^t) \bmod 2.$

The full rule table (all $2^3 = 8$ neighbourhoods; top: left, self, right; bottom: new state):



Bottom row 01011010 in binary is 90: **Rule 90** (Wolfram's numbering), one of $2^8 = 256$ rules.

Running Rule 90 by Hand



each row is one clap; time runs downwards **Cards up after each clap:**

$$1,\ 2,\ 2,\ 4,\ 2,\ 4,\ 4,\ 8$$

Running Rule 90 by Hand



- The pattern is the Sierpiński triangle: a fractal from a two-neighbour rule.
- Number of cards up at clap t is $2^{s(t)}$, where $s(t)$ is the number of 1s in t written in binary (e.g. $t = 5 = 101_2$ gives $2^2 = 4$).
- **Boundary:** we assume a “down” card beyond each end; within 7 claps the pattern does not reach the ends.

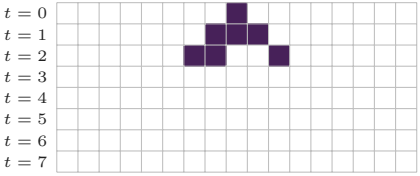
Why this is a model

State space: $\{0,1\}^{17}$, finite and countable. Update: $x^{t+1} = F(x^t)$, deterministic and discrete in space and time. The global pattern is nowhere in the local rule: it emerges.

Your Turn: Rule 30

Same row, same start, new rule table:





first three rows given; fill in the rest

Exercise 1: Rule 30 by hand

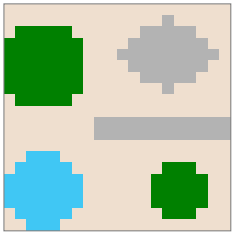
Complete rows $t = 3$ to 7. Count the cards up in each row. Does a pattern like $2^{s(t)}$ appear?

Exercise 2: Play it

Act out Rule 90 with 17 students in class. How many claps until the ends matter?

Land Cover: Water, Built, Forest, Open Land

Problem: What Is the Campus Made Of?



water built forest open

20×20 grid, illustrative

Classes (a countable state set): $S = \{\mathrm{W}, \mathrm{B}, \mathrm{F}, \mathrm{O}\}$

- W: water (lake, ponds)
- B: built (buildings, roads, parking)
- F: forest (tree canopy)
- O: open land (grass, scrub, bare soil)

Problem: What Is the Campus Made Of?



Getting the data. Take a Google Maps or Google Earth screenshot of a campus area, lay a 20×20 grid over it (cells 50×50 m, so 1 km^2), and label each cell by the class covering most of it.

State: $x_i \in S$ for cell $i = 1, \dots, 400$. **Counts:** $n_c = \#\{i : x_i = c\}$.

	W	B	F	O
cells	37	59	66	238
fraction π_c	0.092	0.148	0.165	0.595
area (ha)	9.25	14.75	16.5	59.5

Majority rule is the discretisation: a cell half tree, half road is forced into one class.

Land Cover Model 1: Markov Chain (Cells Act Alone)

Each year every cell changes class with fixed probabilities $q_{xy} = P(x_{t+1} = y \mid x_t = x)$, independent of other cells. Fractions $\pi_t = (\pi_{\mathrm{W}}, \pi_{\mathrm{B}}, \pi_{\mathrm{F}}, \pi_{\mathrm{O}})$ obey

$$\pi_{t+1} = \pi_t\, Q, \qquad Q = \begin{array}{c} \\ \mathrm{W} \\ \mathrm{B} \\ \mathrm{F} \\ \mathrm{O} \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{W} & \mathrm{B} & \mathrm{F} & \mathrm{O} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.995 & 0 & 0.005 \\ 0 & 0.01 & 0.97 & 0.02 \\ 0 & 0.04 & 0.03 & 0.93 \end{pmatrix} \end{array}$$ (illustrative, per year)

Estimating Q: label the same grid on two dates (Google Earth Pro has historical imagery) and set $\hat{q}_{xy} = n_{x\to y}/n_x$, the share of cells of class x that became y.

Land Cover Model 1: Markov Chain (Cells Act Alone)

	W	B	F	O
now, π_0	0.092	0.148	0.165	0.595
after 10 years, $\pi_0 Q^{10}$	0.092	0.339	0.240	0.329
long run (share of non-water land)	–	0.833	0.083	0.083

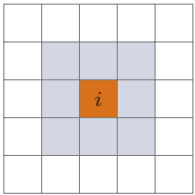
Water never changes (an absorbing state). Because built land almost never goes back ($q_{\mathrm{BB}} = 0.995$), the long run is mostly built: keeping green space needs a policy that raises q_{BF} or lowers q_{OB}.

Is This a Cellular Automaton?



Definition 3 (Cellular automaton)

A lattice of cells, each in one state from a finite set, all updated together at each step by the same *local rule*: a cell's next state depends on its own state and the states of its neighbours.



More neighbourhood: 8 cells

Is This a Cellular Automaton?



- **Model 1 is not a CA.** Each cell flips its own coin and ignores its neighbours.
- **Real land cover is.** New buildings appear next to roads and buildings; trees spread from existing forest by seeds; open land surrounded by buildings is the first to go.
- Add neighbour dependence and Model 1 becomes a **stochastic cellular automaton**. Urban-growth models used in planning (e.g. SLEUTH) work this way.

Rule 90 and Rule 30 were 1-D, 2 states, deterministic. Land cover is 2-D, 4 states, random.

Land Cover Model 2: The Cellular Automaton Rule

Let $\nu_{\mathrm{B}}(i)$ and $\nu_{\mathrm{F}}(i)$ be the fractions of the 8 neighbours of cell i that are built and forest. Each year, all cells together:

Current state	Becomes	With probability
W (water)	W	1
B (built)	B	1
O (open)	B	$a + b\,\nu_{\mathrm{B}}(i)$ (building spreads)
O (open)	F	$c\,\nu_{\mathrm{F}}(i)$ (seeds from nearby trees)
F (forest)	B	$e\,\nu_{\mathrm{B}}(i)$ (cleared next to buildings)
F (forest)	O	d (felling, tree death)

Land Cover Model 2: The Cellular Automaton Rule



Otherwise the cell keeps its state. Illustrative parameters: $a = 0.005$, $b = 0.15$, $c = 0.08$, $d = 0.01$, $e = 0.03$.

- a: building on open land far from anything (rare); b: extra chance per built neighbour.
- An open cell fully surrounded by buildings ($\nu_{\mathrm{B}} = 1$) is built within a year with probability 0.155; an isolated one with probability 0.005. That $31\times$ difference is what Model 1 cannot express.

Land Cover: Ledger and Classification



Dym. *Why?* Plan green cover as the campus grows. *Find?* Class counts now and in 10 years, and where change happens. *Given?* Two labelled maps. *How?* Markov chain; neighbour-dependent CA. *Valid?* Fractions sum to 1; water unchanged. *Verified?* Predict a past year from an older one and compare with the real map.

Assumption ledger

- A1: Majority rule labels each 50 m cell with one class.
- A2: Four classes are enough (lawn and bare soil are both “open”).
- A3: Transition probabilities stay the same every year.
- A4: Only the 8 nearest cells matter (no roads or master plan far away).
- A5: Built land never reverts; water never changes.
- A6: The two image dates are in the same season.

Velten SQM. S physical, natural and technical; Q phenomenological, instationary, distributed (a map, not one number); M discrete in space and time, stochastic, nonlinear (CA probabilities depend on neighbours).

Exercises: Land Cover



Exercise 3: Label your map

Pick a 1 km^2 part of campus that includes the lake. Grid it 20×20, label W, B, F, O, and give counts, fractions and areas. Have two students label it independently: how many cells disagree?

Exercise 4: Estimate Q

Label the same grid from the oldest Google Earth image available. Estimate $\hat{q}_{xy}$ and project 10 years ahead. Find the long-run fractions.

Exercise 5: Build the CA

Code the CA rule in Python. Fit a and b so that the CA reproduces where new buildings actually appeared between your two dates. Does it beat the Markov chain at predicting *which* cells changed?

Exercises: Rooms, Fans and Leaves



Exercise 6: Your rooms

Measure A and P for every room you counted. Fit a_f, a_ℓ, s_p from your data and test the ceiling rules. Which rooms break them, and why?

Exercise 7: Institute total

Get the number of rooms of each type. Compute $\hat{N} \pm 2\,\mathrm{SE}$ for fans, switches and data ports. Which room type contributes most to the uncertainty?

Exercises: Rooms, Fans and Leaves (contd.)



Exercise 8: Fans in a month

Starting from all fans working ($w_0 = 1$), iterate $w_{t+1} = (1-p)w_t + q(1-w_t)$ for $p = 0.02$ and $q = 0.1, 0.5$. How many months to get within 1% of w^*?

Exercise 9: Two leaf counts

For one campus tree, compute N from λb^n and from $\lambda(D/d)^2$. Validate on one fully counted branch. Explain the difference using the error-growth formula.

From Space: The Vegetation Index (NDVI)

Chlorophyll absorbs red light; leaf tissue reflects near-infrared (NIR). With ρ the fraction of light reflected,

$$\mathrm{NDVI} = \frac{\rho_{\mathrm{NIR}} - \rho_{\mathrm{R}}}{\rho_{\mathrm{NIR}} + \rho_{\mathrm{R}}} \in [-1, 1].$$

Data: Sentinel-2 (Copernicus Browser, free), red band B4, NIR band B8, 10 m pixels. Google Maps cannot give NDVI: it has no NIR band.

Surface (typical)	ρ_{R}	ρ_{NIR}	NDVI
forest canopy	0.05	0.45	0.80
grass	0.08	0.35	0.63
bare soil	0.25	0.30	0.09
concrete, roofs	0.30	0.35	0.08
water	0.05	0.02	-0.43

From Space: The Vegetation Index (NDVI)



Is NDVI a discrete model? The formula itself is a static algebraic map. The data are discretised (pixels, bands, integer readings, dates), and the formula is applied pixel by pixel. It becomes part of a discrete model when thresholds turn values into classes:

$$\text{NDVI} < 0 \to \text{W}, \qquad 0 \le \text{NDVI} < 0.2 \to \text{B or bare}, \qquad 0.2\text{–}0.5 \to \text{O}, \qquad > 0.5 \to \text{F}.$$

This automates the hand labelling of the land-cover problem, with one weakness: concrete and bare soil have the same NDVI, so a satellite map still needs Google Maps or a field visit to tell built from bare.

Summary

Model	Discrete structure	Key result
one room	integer rules $\lceil A/a_f \rceil$, $\lfloor P/s_p \rfloor$	staircase over the L^2, L laws
institute	running sum $N_j = N_{j-1} + c_j$; strata	$\hat{N} = \sum R_k \bar{c}_k \pm 2\,\mathrm{SE}$
fan status	two-state Markov chain	$w^* = q/(p+q)$
tree	$T_{k+1} = bT_k$, $d_{k+1}^2 = d_k^2/b$	$N = \lambda(D/d)^2$
seasons	Fibonacci, logistic map	saturation at K; possible chaos
row of cards	elementary CA, Rule 90	Sierpiński pattern, $2^{s(t)}$ cards
land cover, cells alone	4-state Markov chain $\pi_{t+1} = \pi_t Q$	long-run fractions
land cover, neighbours	stochastic cellular automaton	clustered growth
NDVI	algebraic map per pixel + thresholds	automatic classes

Take-home

A discrete model is not a rough version of a continuous one. It respects counts and steps, and it shows effects the smooth model hides: rounding up, error growth through levels, and oscillation in the logistic map.



Thank You