Solving Recurrence Relations

Table of contents

Two ways of writing the same sequence 1
From a recursive definition to an explicit formula 3
Worked examples 5
The Tower of Hanoi 8
See also 9

recursive-definitions[1] builds a sequence out of its own earlier terms. This note goes the other way: given a recursive definition, find an **explicit formula** for the n^{th} term, and prove the formula is right. See mathematical-induction[2] and strong-induction-and-the-well-ordering-principle[3] for the proof machinery the second method leans on.

Two ways of writing the same sequence

Every sequence here can be described twice over, and the two descriptions are worth naming precisely because the whole topic is moving between them:

	Shape	What it lets you do
Recurrence relation	$a_k = \ldots a_{k-1} \ldots$	get the next term from the ones before it
Explicit formula	$a_n = \ldots n \ldots$	get *any* term without computing the others

The subscript is the tell. A rule written in k that mentions a_{k-1} is a recurrence relation; a rule written in n that mentions only n is an explicit formula.

[1] courses/math1061/recursive-definitions.pdf
[2] courses/math1061/mathematical-induction.pdf
[3] courses/math1061/strong-induction-and-the-well-ordering-principle.pdf

Arithmetic and geometric sequences

The two standard families, in both forms — for a sequence $a_0, a_1, a_2, \ldots$:

	Recurrence relation	Explicit formula
Arithmetic	$a_k = a_{k-1} + d$ for integers $k \geq 1$	$a_n = a_0 + dn$ for integers $n \geq 0$
Geometric	$a_k = r \cdot a_{k-1}$ for integers $k \geq 1$	$a_n = a_0 \cdot r^n$ for integers $n \geq 0$

d is the **common difference** and r the **common ratio**. Both explicit formulas are the recurrence applied n times from a_0, which is why the index of the seed and the exponent agree: a_n is n steps away from a_0.

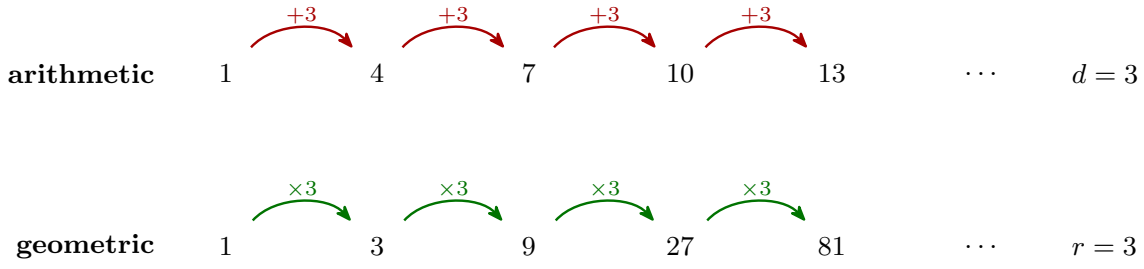


Figure 1: The two families as operations between consecutive terms. Arithmetic adds the same d at every step; geometric multiplies by the same r. Reading a sequence means testing the gaps: constant differences means arithmetic, constant ratios means geometric.

Worked the other way, from terms to a formula: $2, 6, 18, 54, \ldots$ multiplies by 3 each step, so $r = 3$; the seed is $a_0 = 2$; hence $a_n = 2 \cdot 3^n$.

Telling them apart

Deciding which family a run of terms belongs to is a matter of computing the gaps both ways, not of eyeballing.

Do $\frac{5}{8}, \frac{5}{2}, 10$ sit in an arithmetic sequence, a geometric one, both, or neither?

Put everything over the same denominator — $\frac{5}{8}, \frac{20}{8}, \frac{80}{8}$ — and both tests become easy:

Test	First gap	Second gap	Verdict
Differences	$\frac{20}{8} - \frac{5}{8} = \frac{15}{8}$	$\frac{80}{8} - \frac{20}{8} = \frac{60}{8}$	not equal — **not arithmetic**
Ratios	$\frac{5}{2} \div \frac{5}{8} = 4$	$10 \div \frac{5}{2} = 4$	equal — **geometric**, $r = 4$

So the answer is *geometric only*. The common trap is "both": a sequence that passes the ratio test has no obligation to pass the difference test, and here it plainly fails it.

Do $2, 2, 2$? This one is **both**. The differences are all 0, so it is arithmetic with $d = 0$; the ratios are all 1, so it is geometric with $r = 1$. It is **a constant sequence** $a_n = c$, which is what you get from $d = 0$ on one side and $r = 1$ on the other. The constant sequences are the *whole* of the overlap and not merely examples of it: $a_0 + dn$ and $a_0 r^n$ can agree for every n only if $d = 0$, and then both are constant.

> One convention worth knowing: many texts require $a_0 \neq 0$ and $r \neq 0$ for a geometric sequence, which excludes the all-zeros sequence $0, 0, 0, \ldots$ — every r satisfies $0 = r \cdot 0$, so the common ratio would not be determined. $2, 2, 2$ is unaffected.

From a recursive definition to an explicit formula

The workflow is always the same three steps:

1. **Calculate** the first few terms from the recursive definition.
2. **Guess** an explicit formula that fits them.
3. **Prove** the guess is correct.

Step 3 is not optional, and it is the whole difficulty: any finite run of terms is consistent with infinitely many formulas (see sequences-and-summations[4] on why listing terms never defines a sequence). Two methods are available.

	Method 1	Method 2
What you show	the formula gives the right initial conditions, *and* satisfies the recurrence relation	the formula holds for every n, by (strong) mathematical induction
In a word	quicker	rigorous

[4] courses/math1061/sequences-and-summations.pdf

Why Method 1 is a proof and not just a check

Method 1 can look like it is only testing a couple of cases, so it is worth seeing why it settles the matter. A recursive definition **determines its sequence uniquely**: the initial conditions fix the first terms, and the recurrence relation then forces every later one. So if your formula reproduces the initial conditions and obeys the same recurrence, it generates the same sequence term by term — there is nowhere for the two to diverge. That is the entire content of Method 1, and it is why the two checks it makes are exactly the two halves of the definition.

Method 2 makes that argument explicitly rather than appealing to it, which is why it is the more rigorous of the two and the one to use when a question says "prove by induction".

The Method 1 template

Given a guess $a_n = f(n)$:

- **Initial conditions.** Substitute the seed index into f and check it returns the stated value. One check per initial condition.
- **Recurrence relation.** Write down both sides separately, substituting the guess wherever an earlier term appears, and show they agree:

$$\text{LHS} = a_k = f(k) \qquad \text{RHS} = (\text{the recurrence, with } f(k-1),\ f(k-2),\ \ldots \text{ substituted})$$

Set the two sides out under the labels **LHS** and **RHS** and meet in the middle. That keeps you from the circularity of starting with what you are trying to prove.

The Method 2 template

Take $P(n)$ to be the predicate $a_n = f(n)$ and run an induction. Which induction depends on the recurrence:

The recurrence reaches back	Use	Basis step needs
one term (a_{k-1})	ordinary induction (MI)	one base case
two or more terms ($a_{k-1}, a_{k-2}, \ldots$)	strong induction (SMI)	one base case per initial condition

The rule is not arbitrary: to get $P(k+1)$ from a recurrence mentioning a_{k-1}, you need $P(k-1)$ as well as $P(k)$, and only the strong hypothesis hands you both.

Worked examples

A recurrence reaching back one term

$c_1 = 1$ and $c_k = c_{k-1} + 2^k$ for each integer $k \geq 2$. Is this the sequence $c_n = 2^{n+1} - 3$ for each integer $n \geq 1$?

Method 1.

Initial condition ($n = 1$): $c_1 = 2^{1+1} - 3 = 4 - 3 = 1$

Recurrence relation:

$$\text{LHS} = c_k = 2^{k+1} - 3$$
$$\text{RHS} = c_{k-1} + 2^k = (2^{(k-1)+1} - 3) + 2^k = 2^k + 2^k - 3 = 2(2^k) - 3 = 2^{k+1} - 3$$

LHS = RHS, so the formula satisfies the recurrence. **The statement is true.**

Method 2. Let $P(n)$ be the predicate $c_n = 2^{n+1} - 3$.

Basis step. $c_1 = 1$ by the recursive definition, and $2^{1+1} - 3 = 4 - 3 = 1$, so $P(1)$ is true.

Inductive hypothesis. Suppose $P(k)$ is true for some integer $k \geq 1$, i.e. $c_k = 2^{k+1} - 3$. Then

$$c_{k+1} = c_k + 2^{k+1} = (2^{k+1} - 3) + 2^{k+1} = 2(2^{k+1}) - 3 = 2^{k+2} - 3$$

so $P(k+1)$ is true. Hence by mathematical induction $c_n = 2^{n+1} - 3$ for each integer $n \geq 1$. ■

Note the shape of the inductive step: substitute the recurrence, then the hypothesis, then collect. The two 2^{k+1} terms combining into 2^{k+2} is the only real move.

A recurrence reaching back two terms

$b_0 = 0$, $b_1 = 1$ and $b_k = \frac{1}{2}(b_{k-1} + b_{k-2} + 1) + 3(k-1)$ for each integer $k \geq 2$.

Calculating terms. Each one uses the two before it:

$$b_2 = \tfrac{1}{2}(1 + 0 + 1) + 3(1) = 1 + 3 = 4$$
$$b_3 = \tfrac{1}{2}(4 + 1 + 1) + 3(2) = 3 + 6 = 9$$
$$b_4 = \tfrac{1}{2}(9 + 4 + 1) + 3(3) = 7 + 9 = 16$$

giving $0, 1, 4, 9, 16, \ldots$ — **guess $b_n = n^2$ for all integers $n \geq 0$.**

Method 1.

Initial conditions: $b_0 = 0^2 = 0$ and $b_1 = 1^2 = 1$ — two conditions, two checks.

Recurrence relation:

$$\text{LHS} = b_k = k^2$$

$$\begin{aligned}
\text{RHS} &= \tfrac{1}{2}\big((k-1)^2 + (k-2)^2 + 1\big) + 3(k-1) \\
&= \tfrac{1}{2}\big(k^2 - 2k + 1 + k^2 - 4k + 4 + 1\big) + 3k - 3 \\
&= \tfrac{1}{2}\big(2k^2 - 6k + 6\big) + 3k - 3 \\
&= \big(k^2 - 3k + 3\big) + 3k - 3 \\
&= k^2
\end{aligned}$$

LHS = RHS, so the explicit formula satisfies the recurrence relation.

Method 2, by strong induction — two terms back, so SMI and two base cases.

Basis step. $b_0 = 0$ and $0^2 = 0$, so $P(0)$ holds; $b_1 = 1$ and $1^2 = 1$, so $P(1)$ holds.

Inductive hypothesis. Suppose $k \geq 1$ is an integer and $P(0), P(1), \ldots, P(k)$ are each true. Then

$$\begin{aligned}
b_{k+1} &= \tfrac{1}{2}\big(b_k + b_{k-1} + 1\big) + 3\big((k+1) - 1\big) \\
&= \tfrac{1}{2}\big(k^2 + (k-1)^2 + 1\big) + 3k \\
&= \tfrac{1}{2}\big(k^2 + k^2 - 2k + 1 + 1\big) + 3k \\
&= \tfrac{1}{2}\big(2k^2 - 2k + 2\big) + 3k \\
&= k^2 - k + 1 + 3k \\
&= k^2 + 2k + 1 = (k+1)^2
\end{aligned}$$

so $P(k+1)$ is true, and by SMI $b_n = n^2$ for each integer $n \geq 0$. ■

The hypothesis is used **twice** here — once for b_k and once for b_{k-1} — which is precisely what ordinary induction could not have supplied.

The same sequence, a different recursive definition

Find a *different* recursive definition of $\{b_n\}_{n\geq 0}$.

Once the explicit formula is known this is easy, and it makes the point that **a sequence does not have one recursive definition**. Look for a one-term recurrence $b_n = b_{n-1} + p$ and solve for the step p:

$$n^2 = (n-1)^2 + p \implies p = n^2 - (n-1)^2 = 2n - 1$$

so $b_0 = 0$ and $b_n = b_{n-1} + (2n-1)$ for all integers $n \geq 1$ defines the same sequence — the familiar fact that the squares are the running totals of the odd numbers. Note it needs only **one** initial condition where the original needed two.

A recurrence that is not a polynomial

$d_0 = 1$ and $d_k = \dfrac{d_{k-1}}{1 + d_{k-1}}$ for each integer $k \geq 1$.

Calculating terms. The useful move is multiplying by a disguised 1 to clear the fraction inside the fraction:

$$d_1 = \frac{1}{1+1} = \frac{1}{2}$$

$$d_2 = \frac{\frac{1}{2}}{1+\frac{1}{2}} \cdot \frac{2}{2} = \frac{1}{2+1} = \frac{1}{3} \qquad d_3 = \frac{\frac{1}{3}}{1+\frac{1}{3}} \cdot \frac{3}{3} = \frac{1}{3+1} = \frac{1}{4}$$

giving $1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots$ — **guess** $d_n = \dfrac{1}{n+1}$ **for all integers** $n \geq 0$**.**

That $\times \frac{k}{k}$ trick is worth keeping: multiplying numerator and denominator by the denominator of the inner fraction turns a compound fraction into a simple one in one step, with no common-denominator bookkeeping.

Method 1.

Initial condition ($d_0 = 1$): $d_0 = \dfrac{1}{0+1} = 1$

Recurrence relation:

$$\text{LHS} = d_k = \frac{1}{k+1} \qquad \text{RHS} = \frac{d_{k-1}}{1+d_{k-1}} = \frac{\frac{1}{k}}{1+\frac{1}{k}} \cdot \frac{k}{k} = \frac{1}{k+1}$$

LHS = RHS.

Method 2, by ordinary induction — the recurrence reaches back one term, so MI and one base case.

Basis step. $d_0 = 1$ from the definition and $\frac{1}{0+1} = 1$, so $P(0)$ is true.

Inductive hypothesis. Suppose $P(k)$ is true for some integer $k \geq 0$. Then

$$d_{k+1} = \frac{d_k}{1+d_k} = \frac{1/(k+1)}{1+1/(k+1)} = \frac{1/(k+1)}{(k+2)/(k+1)} = \frac{1}{k+1} \cdot \frac{k+1}{k+2} = \frac{1}{k+2}$$

which is $\frac{1}{(k+1)+1}$, so $P(k+1)$ is true. By MI, $d_n = \dfrac{1}{n+1}$ for each integer $n \geq 0$. ■

The Tower of Hanoi

> Given a tower of 8 discs of decreasing size on one of three pegs, transfer the whole tower to another peg, subject to two rules: **move only one disc at a time**, and **never move a larger disc onto a smaller one**. Is there a solution, and if so what is the fewest number of moves?

This is the standard example of a recurrence that comes out of an argument rather than out of a formula. For $n \geq 1$ let m_n be the **minimum number of moves** needed to transfer a tower of n discs from one peg to another.

Small cases first: $m_1 = 1$ — move the single disc. $m_2 = 3$ — small disc to the spare peg, large disc to the target, small disc back on top.

The general case is the recursive insight:

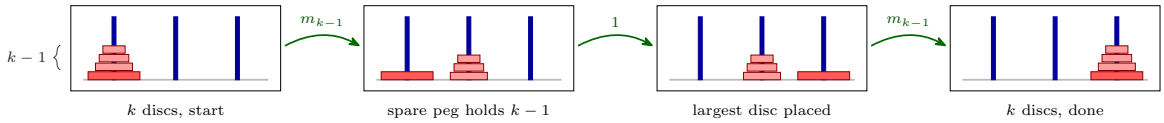


Figure 2: Why the recurrence doubles and adds one. Each box is one position of the three pegs, with discs as bars, the widest at the bottom, and the moves run left to right. To move k discs you must at some point move the largest, and it can only go onto an empty peg — so the other $k-1$ discs have to be parked on the spare peg first, then brought back on top afterwards. Two sub-transfers of $k-1$ discs, plus the single move of the largest. The $k-1$ smaller discs are drawn as a stack of three for legibility; nothing in the argument depends on how many there are.

Reading the picture off as arithmetic gives, for example, $m_4 = m_3 + 1 + m_3$, and in general

$$m_1 = 1 \qquad \text{and} \qquad m_k = 2m_{k-1} + 1 \text{ for all integers } k \geq 2$$

The explicit formula. The textbook (p. 310 in the 4th edition, p. 346 in the 5th) guesses $m_n = 2^n - 1$ for all integers $n \geq 1$. By Method 1:

Initial condition ($m_1 = 1$): $m_1 = 2^1 - 1 = 2 - 1 = 1$

Recurrence relation:

$$\text{LHS} = m_k = 2^k - 1$$
$$\text{RHS} = 2m_{k-1} + 1 = 2(2^{k-1} - 1) + 1 = 2 \cdot 2^{k-1} - 2 + 1 = 2^k - 1$$

LHS = RHS, so the guess is correct. The textbook proves the same result by mathematical induction — either method is acceptable.

So the eight-disc tower takes $2^8 - 1 = 255$ moves, and the answer to "is there a solution?" is yes, constructively, because the recursion *is* the algorithm.

See also

- recursive-definitions[5] — the other direction, and recursively defined sets
- sequences-and-summations[6] — sequence notation, and why initial terms never define a sequence
- mathematical-induction[7] · strong-induction-and-the-well-ordering-principle[8] — Method 2's machinery
- 2026-08-31-solving-recurrence-relations[9] — Lecture 16
- practice-problems[10] — §20 Solving Recurrence Relations, with full worked solutions

[5] courses/math1061/recursive-definitions.pdf
[6] courses/math1061/sequences-and-summations.pdf
[7] courses/math1061/mathematical-induction.pdf
[8] courses/math1061/strong-induction-and-the-well-ordering-principle.pdf
[9] courses/math1061/2026-08-31-solving-recurrence-relations.pdf
[10] courses/math1061/practice-problems.pdf