What Is Running Behind This Page

Six simulations render the backgrounds on this site: Conway's Life, a simulated-annealing graph search, an FM synthesizer, A, and a finite-difference PDE solver. What each one computes and what the controls do.*

september 28, 2025 · Projects

---

the backgrounds on this site are simulations, not video loops. every one is computed per frame in your browser. the landscape icon in the corner opens a panel with the parameters for whichever is on screen, and they cycle every twelve seconds unless you turn that off.

six of them, in cycle order:

background	what it is
cellular automaton	conway's life, plus five other b/s rules
simple waves	three summed sine waves
job scheduling	simulated annealing over a clustered graph
dual fm oscillator	a working fm synth, oscilloscope and spectrogram
shortest path	dijkstra, a*, and greedy search, side by side
pde solver	finite-difference heat and wave equations

cellular automaton

a grid of cells, a rule, and a state buffer stepped one generation at a time. each cell counts its eight neighbors on a wrapping torus, then lives, dies, or is born according to the rule, which is conway's b3/s23 by default, with highlife, maze, coral, day & night and seeds in the dropdown.

the shader only draws; the rule runs on a real buffer. newborn cells take the newborn color and shift toward the established color the longer they survive, and links are drawn between live neighbors, the same eight-cell neighborhood the rule is evaluated over.

random soup under conway settles into still lifes and period-two oscillators within a couple of hundred generations, which is too static for a background. `perturbation rate` flips a small fraction of cells each step to keep things moving. set it to zero to watch it stall.

simple waves

three sine waves summed in a shader. one runs along x, one along y at 0.8× the frequency, one diagonally at 0.6×, each drifting at a different rate. color maps amplitude: bright where they reinforce, dark where they cancel.

it is nothing but superposition. interference is the mechanism behind a great deal of signal processing, and it takes three lines of arithmetic. i spent a couple of years on wavelet bases for audio compression at stony brook, and what stayed with me is how little machinery the underlying physics needs.

turn `wave frequency` up and `wave speed` down to freeze the interference pattern in place.

job scheduling

the most interesting one, and the worst named.

it builds a clustered graph: nodes grouped into tight clusters with dense, high-bandwidth links inside each cluster and sparse, high-latency links between them. roughly the shape of a real datacenter. then it searches for the subgraph of size _n_ with the highest total conductivity, the best-connected group of that size.

that search is simulated annealing with an exponential cooling schedule. it proposes a swap, accepts it outright if it scores better, and accepts it with a temperature-dependent probability if it scores worse. early on, while it is hot, it takes bad trades freely and wanders. as the temperature drops it stops accepting losses and settles. the two highlight colors are separate things: one is the candidate set it is considering right now, the other is the best set it has found so far. early on they diverge constantly. near the end they lock together.

the layout is force-directed and running at the same time, so the graph is still settling while the search runs over it.

set `number of clusters` to 2 and `requested subgraph size` to something close to one cluster's worth of nodes, then watch how long it takes to commit to one side.

dual fm oscillator

a working synthesizer. two oscillators, one modulating the other's phase. that is all fm synthesis is, and it is how a dx7 could make a bell out of two sine waves when a subtractive synth needed a filter bank. the signal then runs through filter, delay, distortion and reverb.

top display is an oscilloscope: amplitude against time. bottom is a spectrogram: frequency low-to-high, scrolling left-to-right, brightness as intensity. every bin is a hann-windowed discrete fourier transform of the same samples the oscilloscope is drawing, so the sidebands, the harmonic series of a square wave, and the sum and difference tones from the ring modulator are all measured rather than drawn. seeing one signal in both domains at once shows what fm does to a spectrum.

push `vco 1 fm amount` up slowly and watch the sidebands appear in pairs either side of the carrier, spaced at the modulator frequency.

it makes sound. hold the speaker button in the settings panel.

i built synthesizers as artist in residence at cewit for a year, mostly analog.

shortest path

seeded random graph, real priority frontier, and one slider that changes which algorithm you are running.

`heuristic weight` scales the heuristic term in `f = g + w·h`:

- **w = 0**: the heuristic vanishes and it is dijkstra. explores outward in every direction equally. always finds the optimal path, looks at far more of the graph than it needed to.
- **w = 1**: a*. the heuristic is admissible, so the path is still optimal, but exploration stretches toward the goal instead of spreading evenly.
- **w > 1**: greedy. it over-trusts the heuristic, drives almost straight at the goal, and gives up the optimality guarantee to do it.

watch the shape of the explored region: a circle, then an ellipse, then a corridor. that is the optimality-versus-effort tradeoff drawn as a shape, and it is why i kept this one in the rotation.

turn `steps per second` down to about 5 to watch the frontier expand node by node.

pde solver

explicit finite differences on a grid, solving either the heat equation $\partial u/\partial t = \alpha\nabla^2 u$ or the wave equation $\partial^2 u/\partial t^2 = c^2\nabla^2 u$. the laplacian is a five-point stencil; heat uses forward-time centered-space, wave uses a centered second difference in time. grid runs at 64^2, 128^2 or 256^2.

the parameters that change the physics rather than the look are `boundary condition` and `initial condition`. dirichlet fixes the edge value, so waves reflect inverted. neumann sets the edge derivative to zero, so they reflect upright. periodic wraps, so anything leaving the right edge arrives at the left. pick a gaussian pulse on the wave equation and switch between the three. the difference in reflection is immediate. it is the clearest demonstration of boundary conditions i know of.

you cannot make it explode. explicit schemes are only conditionally stable. heat needs $\alpha\cdot dt\cdot(1/dx^2 + 1/dy^2) \le 0.5$ and wave needs the cfl condition $c\cdot dt\cdot\sqrt{(1/dx^2 + 1/dy^2)} \le 1$, so the timestep is clamped to the stable maximum before every step. push thermal diffusivity to its limit and the simulation slows down rather than diverging.

on the whole thing

they render to webgl or canvas depending on the background, pause when the tab is hidden, and cycle on a twelve-second timer with a 1.2-second crossfade. every parameter you change is live.

why they exist: a static background is a wasted surface, and i wanted somewhere to put the numerical methods i do not get to write at work.

https://alexnodeland.com/timeline/250928_interactive-algorithm-visualizations/