



VisualPDE lets anyone—students, teachers, and researchers—simulate rich partial differential equation (PDE) models directly in the browser, in real time, no coding or installs. Under the hood it turns your device's GPU into a math engine, so patterns bloom and waves ripple as fast as you can drag a slider or draw with your finger. It's "Desmos for PDEs," except the plots are alive.

TL;DR

Open visualpde.com, pick an example (or type your own equations), then paint initial conditions, adjust sliders, and share a link or embed the live sim. It runs on laptops, tablets, and phones, with GPU-accelerated timestepping delivering tens of thousands of steps per second on typical demos.

What is VisualPDE?

PDEs describe how things change across space and time: heat diffuses, waves propagate, chemicals react and spread, populations move. Historically, exploring them required code, licenses, and compute time. VisualPDE removes those barriers: open a link and you're simulating on the spot—no installation, works on most devices, and the entire site is open source if you want to peek or fork.

What to do?

Think interactive PDEs you can play with—edit equations live, watch solutions evolve instantly, and share or embed your setup with one click.

How it works?

GPU + WebGL

VisualPDE treats the simulation grid like an image. Each pixel stores field values; fragment shaders update every pixel in parallel. Your browser's WebGL pipeline does the heavy lifting, stepping the system hundreds of times per frame for smooth, responsive dynamics. Result: real-time PDEs in a tab.

- WebGL + GLSL shaders = parallel updates
- Grid = floating-point textures
- "Solve-as-you-type" responsiveness

Numerics that favor interactivity

Under the hood are familiar methods from a first course in scientific computing:

- Space: Finite difference method on a 1D/2D grid.
- Time: explicit schemes you can choose: Forward Euler, Adams–Bashforth (2-step), Midpoint, and RK4. VisualPDE also detects instability (NaNs/infinities) and nudges you to tweak the timestep or scheme.

Rules of thumb

If a sim “blows up,” halve Δt or switch from Forward Euler to RK4. For pattern-forming systems, a small Δt is usually fine thanks to modern GPUs.

Precision & performance

To maximize compatibility—especially on mobile—VisualPDE currently uses single-precision floats. In practice, that’s enough for teaching and rapid exploration; double precision is on the roadmap as browser support matures. Typical demos achieve tens of thousands of timesteps per second on modern hardware.

Playful interface, serious models

- Paintable initial conditions. Pause and draw on the domain with a brush—seed a ring for a wave equation, scribble a hot patch for diffusion, or drop “prey” into a corner of a predator–prey model.
- Parameter sliders. Declare ranges (e.g., $a = 0.5$ in $[0, 1]$) and explore transitions live.
- Live math. LaTeX rendering confirms your equations are interpreted as intended.
- Flexible domains & boundaries. Use standard rectangular setups, custom masks, and common boundary conditions (periodic, Dirichlet, Neumann, Robin).
- Rich visualization. Toggle 2D colormaps, contours, vector fields, or a 3D surface for 2D sims—instantly.
- Share & embed. Generate a short link that encodes your full setup, or copy an iframe to embed an interactive figure in docs, lectures, or posts.

☰ Minimal embed snippet

```
1 <iframe
2   src="https://visualpde.com/sim/?preset=heatEquation"
3   width="640" height="480" loading="lazy"
4   title="VisualPDE simulation">
5 </iframe>
```

You can also export embed code (tiny or customized), screenshots or short clips for slides and social posts — use the share button at the right hand side of each simulation.

What you can build (and why it's delightful)

Pattern formation: spots, stripes, labyrinths

- Gray–Scott model. Two parameters steer a zoo of behaviors: stationary/moving spots, labyrinths, waves, and more. Exploring this interactively is eye-opening.

[VisualPDE simulation](#)

[Explore the Gray-Scott model on VisualPDE](#) ↗

- Turing systems. Try Schnakenberg or Brusselator; even “Turing on Turing”—use an image/webcam as spatial heterogeneity to watch your face turn into spots and stripes.

Environment: landscapes come alive

- [Dryland vegetation \(Klausmeier\)](#): ↗ Add a rainfall gradient on a slope and watch tiger-bush stripes emerge, rotate, or break up.
- [Urban flooding \(Oxford\)](#): ↗ A LiDAR-based elevation map lets you “pour” water, adjust river inflow, and even “dig” diversion channels—an intuitive sandbox for flood intuition.

Health: waves of life and risk

- FitzHugh–Nagumo model. Visualize spiral waves and their competition with homogeneous oscillations—an accessible window into arrhythmia-like dynamics.

[VisualPDE simulation](#)

[FitzHugh–Nagumo model](#) ↗

- [Airborne virus story](#): Toggle ventilation to see how airflow reshapes indoor concentration patterns; an interactive Plus Magazine feature showcased this beautifully.

Physics & beyond

- Kuramoto–Sivashinsky. A classic spatiotemporal chaos demo in a few lines.

[VisualPDE simulation](#)

[Kuramoto–Sivashinsky equation](#)

- Cahn–Hilliard. Phase separation that coarsens over time.

[VisualPDE simulation](#)

[Cahn–Hilliard equation](#)

(All available as ready-to-run examples.)

Why this matters for learning

Immediate, visual feedback builds intuition for symmetry breaking, instability, and nonlinearity—concepts that are hard to “see” in static notes. In classrooms and outreach, embedded sims consistently boost engagement.

Where VisualPDE fits in the tool landscape

- COMSOL-type codes: good for engineering detail and complex 3D geometries, but closed sourced which is often a deal breaker for reproducibility.
- MATLAB / Mathematica: great for coding bespoke solvers and symbolic work; less immediate for shareable, interactive experiments, also closed sourced.
- Open source codes like [Basilisk C](#), [OpenFOAM](#), [Pyoomph](#): Open source, high precision, but heavy weight — not ideal for easily sharing.

VisualPDE fills the gap: frictionless, free, shareable, and powerful enough for research prototyping and teaching—in a browser.

Community, credibility, and what’s next

- The site and solver details are openly documented; the codebase is on [GitHub](#).
- VisualPDE has featured in [Chalkdust](#) and in [Plus magazine](#) (with interactive articles/podcasts), and a [peer-reviewed BMB paper](#) provides technical and pedagogical context.

- Roadmap: continued performance and usability improvements, and work toward a Julia-based companion for larger or more complex studies—all while keeping interactivity at the core.

Get started

1. Visit visualpde.com → Explore an example that catches your eye (waves, epidemics, ecology).
2. Play:
 - Press pause and paint an initial condition.
 - Add a slider for a parameter (a in $[0, 1]$) and watch the behavior shift.
 - Flip the time-stepping scheme (try RK4) and compare stability.
3. Share or embed what you made (short link or iframe), or record a short clip for your slides.

The big picture

VisualPDE democratizes a language used across science—PDEs—by making them visible, tactile, and shareable. You can draw a hypothesis, nudge a parameter, and see the consequences in seconds. That shift—from static math to interactive exploration—is why this tool resonates in classrooms, labs, and public science alike. Open your browser; the canvas is already waiting.

Acknowledgements & sources: this post integrates and cites the VisualPDE paper¹, documentation², and example library (open source, with extensive user-guide material and “Visual Stories”). Also see: <https://visualpde.com/about/>.

☆ Metadata >

 Back to main website

[Home](#), [Team](#), [Research](#), [Github](#), [Blogs](#)

 [Edit this page on GitHub](#)

Footnotes

1. Walker, B. J., Townsend, A. K., Chudasama, A. K., & Krause, A. L. (2023). VisualPDE: rapid interactive simulations of partial differential equations. *Bull. Math. Biol.*, 85(11), 113. [↩](#)

2. <https://visualpde.com/user-guide> ↗ ↵