**Interactive exploration**
and numerical experiments

- **Vortex shedding by a circular barrier**
  (color indicates curl of the velocity)

- **Sound waves**
  (color indicates density)

- **Force on a barrier**
  (color indicates density)

**How it works**
(Lattice-Boltzmann algorithm)

- Discretize two-dimensional space with a square lattice.
- Allow only 9 fundamental displacements and velocities.
- Simulation variables $n_i$ are the 9 densities, at each lattice site, of molecules with the 9 allowed velocities.
- From these we can compute total density $\rho$ and macroscopic flow velocity $\mathbf{u}$:
  \[
  \rho = \sum n_i, \quad \mathbf{u} = \frac{(n_s + n_w) - (n_1 + n_4 + n_7)}{\rho}.
  \]
- To model thermal velocities $\mathbf{\nu}$, discretize the Boltzmann distribution. Weights are determined by equating moments, up to 4th order, of the continuous and discrete distributions.

\[
D(n) = \frac{m}{2\pi kT} e^{-\frac{m|\mathbf{\nu}|^2}{2kT}} \quad n_{\nu} = \frac{4}{9},
\]

- Total (discretized) velocity is flow velocity plus thermal velocity: $\mathbf{\mathbf{\nu}} = \mathbf{u} + \mathbf{\nu}$ ($|\mathbf{\nu}| \ll 1$)

- "Staggered" algorithm: To obtain equilibrium densities:
  \[
  n_i = n_i + \frac{1}{\tau} (n_i^q - n_i)
  \]
  where $n_i^q$ are the equilibrium densities.

- The algorithm is simply to alternate these collisions with "streaming" in which the molecules move into adjacent cells according to their velocities. (When molecules hit a barrier, they bounce back instead.)

The pros code this in Fortran or C, but for $10^4$ to $10^5$ lattice sites, on today’s personal computers, you can get by with an interpreted language. My Python/NumPy code is only 125 lines; Java or JavaScript requires about twice that, not including GUI controls.

See the web site for more details on the theory, code examples, and references. Enjoy!