Table of contents

Lecture 11

Oscillations and Waves

Python Concepts

- Solving coupled differential equations for oscillatory systems (e.g., damped harmonic oscillators).
- Using Fourier analysis to study waveforms.

Application

- Simulating simple harmonic motion and damped oscillations.
- Visualizing the motion and analyzing the frequency components of the oscillation.
- Homework: Extend the simulation to include coupled oscillators or waves on a string.

```
//| echo: false
d3 = require("d3@7")
```

```
//| echo: false

// Define the Vicsek model simulation
{


function vicsekModel(width, height, particleCount, noise) {
  const particles = d3.range(particleCount).map(() => ({
    x: Math.random() * width,
    y: Math.random() * height,
    angle: Math.random() * 2 * Math.PI
  }));

  function step(currentRadius) {
```

```
    particles.forEach(particle => {
      // Find neighbors within radius
      const neighbors = particles.filter(p =>
        (p !== particle) &&
        (Math.hypot(p.x - particle.x, p.y - particle.y) <= radius)
      );

      // Calculate average direction of neighbors
      let avgAngle = particle.angle;
      if (neighbors.length > 0) {
        avgAngle = d3.mean(neighbors, d => d.angle);
      }

      // Update particle angle with noise
      particle.angle = avgAngle + (Math.random() - 0.5) * noise;

      // Move particle
      particle.x += speed*Math.cos(particle.angle);
      particle.y += speed*Math.sin(particle.angle);

      // Wrap around boundaries
      particle.x = (particle.x + width) % width;
      particle.y = (particle.y + height) % height;
    });
  }

  return {
    particles,
    step
  };
}

// Set up the visualization
const width = 600;
const height = 600;
const particleCount = 400;
const noise = 2;
let speed = 5;
let radius = 20;

// Create container div
const container = d3.create("div");
```

```
// Create slider
const slider = container.append("input")
  .attr("type", "range")
  .attr("min", 1)
  .attr("max", 50)
  .attr("value", radius)
  .style("width", "50%")
  .on("input", function() {
    radius = +this.value;
    d3.select("#radius-value").text(radius);
  });

  const slider1 = container.append("input")
    .attr("type", "range")
    .attr("min", 1)
    .attr("max", 10)
    .attr("value", speed)
    .style("width", "50%")
    .on("input", function() {
      speed = +this.value;
      d3.select("#speed-value").text(speed);
    });

// Display current radius value
container.append("div")
  .text("Radius: ")
  .append("span")
  .attr("id", "radius-value")
  .text(radius);

// Create SVG
const svg = container.append("svg")
  .attr("width", width)
  .attr("height", height)
  .style("background", "#f0f0f0");

const model = vicsekModel(width, height, particleCount, noise);

const particles = svg.selectAll("circle")
  .data(model.particles)
  .join("circle")
  .attr("r", 2)
```

```
  .attr("fill", "blue");

// Animation function
function animate() {
  model.step(radius);
  particles
    .attr("cx", d => d.x)
    .attr("cy", d => d.y);
  requestAnimationFrame(animate);
}

// Start the animation
animate();

// Return the container div
return container.node();}
```