Week Nine PHY-480

Lewis

October 2025

Q17.1 – What is Percolation and the Percolation Threshold?

A percolation process is when parts of a system are randomly filled or left empty and as more parts are filled, small clusters start connecting together. The **percolation threshold** (p_c) is the point where a cluster first connects throughout the system.

The **infinite cluster** is the large and connected group that spans from one side of the grid to the other.

Q17.2 – Scaling Formulas for Cluster Probability and Conductivity

When p gets close to p_c from above, both the size of the infinite cluster and the system's conductivity follow power laws:

$$P_\infty \sim (p - p_c)^\beta, \quad \sigma \sim (p - p_c)^t$$

where:

- P_∞ = chance that a site belongs to the infinite cluster,
- σ = electrical conductivity,
- $\beta \approx 0.14$, $t \approx 1.3$ (for 2D grids).

This means both P_∞ and σ go to zero smoothly as p approaches p_c.

Q17.3 – Simple Algorithm to Find the Infinite Cluster

To find the infinite cluster in a random grid:

1. Make a grid and randomly remove some edges or sites using probability p.

2. Pick a filled (conducting) site and check all connected neighbors using a search.
3. Mark all connected sites as one cluster.
4. Keep doing this until every site belongs to a cluster.
5. The largest cluster that reaches from one side of the grid to the other is the **infinite cluster**.

Q17.4 – Finding Edge Currents and Basic Code Idea

To find edge currents in the random resistor network:

1. Each node follows Kirchhoff's law which just means that the total current going in will equal the current going out.
2. So to write the equation for each node:

$$\sum_j G_{ij}(V_i - V_j) = 0$$

   where G_{ij} is the conductance between node i and j.

3. Solve all these equations at once to get the voltages V_i.
4. Find each edge's current using:

$$I_{ij} = G_{ij}(V_i - V_j)$$

Simple Python Example

```
import numpy as np

def solve_currents(G, left_nodes, right_nodes, V0=1):
    N = len(G)
    A = np.zeros((N, N))
    b = np.zeros(N)

    # Kirchhoff equations
    for i in range(N):
        for j in range(N):
            if G[i, j] != 0:
                A[i, i] += G[i, j]
                A[i, j] -= G[i, j]

    # Set boundary voltages
```

```
    for n in left_nodes:
        A[n, :] = 0
        A[n, n] = 1
        b[n] = V0
    for n in right_nodes:
        A[n, :] = 0
        A[n, n] = 1
        b[n] = 0

    V = np.linalg.solve(A, b)
    I = G * (V[:, None] - V[None, :])
    return V, I
```

Which will solve for the voltage at each node and then calculates the current on each edge.