jj

June 19, 2023

0.1 # Josephson Effect - Jupyter Notebook Plan

1 Introduction

A study of the Josephson effect and the physics related to the application of a Josephson junction.

The Josephson effect is the phenomenon of supercurrent flow between two superconductors separated by a thin insulating layer. The flow of current can occur without resistance due to the quantum mechanical tunneling of Cooper pairs.

The Josephson effect can be modeled using the following equation, known as the Josephson relation:

V = (h / 2e) * dΦ/dt

where V is the voltage across the junction, h is Planck's constant, e is the charge of an electron, Φ is the superconducting phase difference across the junction, and dΦ/dt is the rate of change of the phase difference.

Important Concepts: * Time-dependent behavior * Superconductivity * Quantum Tunneling * Current-phase relations

Basic Principles: * Josephson relation * Critical current of the junction * Josephson Phase - - supercurrent changing with time - - voltage across the junction as function of time

---

2 Application of the Josephson junction

Josephson Junction * simple superconducting circuit

Superconducting Quantum Interference Device (SQUID) * Josephson junctions in a SQUID configuration as a magnetic field sensor

Quantum computing with superconducting qubits * Josephson junction qubits and Josephson energy * Multi-qubit gates

---

3 Exercise

Question:

A Josephson junction is a superconducting circuit consisting of two superconductors separated by a thin insulating layer. Suppose the superconductors have phase differences Φ1 and Φ2, and the insulating layer has a thickness d. Show how the supercurrent flowing through the junction, I, can be calculated using the Josephson relation.

Answer:

The supercurrent flowing through the junction can be calculated using the Josephson relation, which states that the voltage across the junction, V, is given by: V = (h / 2e) * dΦ/dt

The supercurrent, I, is given by the derivative of the phase difference with respect to time: I = (2e / h) * dΦ/dt

Substituting this into the Josephson relation, we get: V = (h / 2e) * (I / (2e / h))

Solving for I, we obtain: I = (2e / h) * V

Thus, the supercurrent flowing through the junction can be calculated by multiplying the voltage across the junction by a constant.

---

4 Demonstration of the Josephson junction effect

Through use of Python coding, we will simulate the behavior of the Josephson junction circuit. The supercurrent and voltage across the junction will be measured and recorded as the phase difference across the junction is varied. This will include modeling the current-phase relation, voltage-phase relation, and time-dependence of the supercurrent and voltage.

The Jospehson relation equation can be used to calculate the supercurrent flowing through the junction as a function of the phase difference and can be used to model the Josephson effect. Additionally, the current-phase relation, which is specific to the type of junction, can also be used to model the Josephson effect.

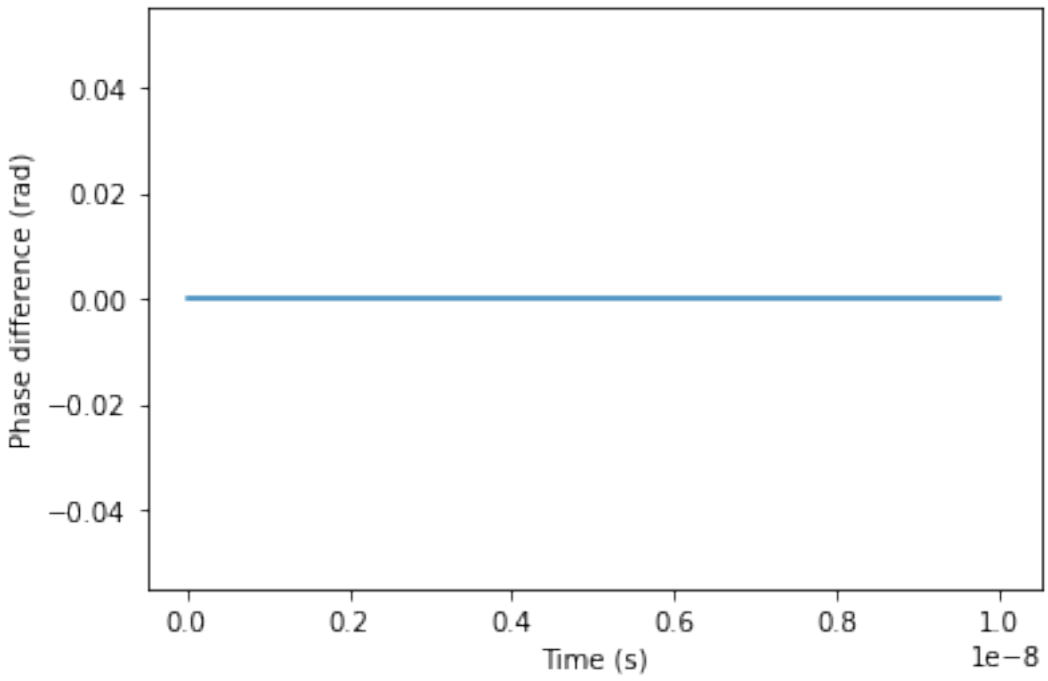
Inputs: * Phase difference * input

Observables: * Voltage across junction * Supercurrent

This code will simulate the behavior of two superconducting junctions connected in series, where the voltage across the junctions will oscillate sinusoidally with a frequency proportional to the phase difference between the junctions. The resulting plot will show the phase difference as a function of time.

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt

# Constants
h = 6.626 * 1e-34  # Planck constant
e = 1.6 * 1e-19  # Charge of an electron
hbar = h / (2 * np.pi)
R = 1e-3  # Resistance of the junction
C = 1e-15  # Capacitance of the junction
Ic = 1e-6  # Critical current of the junction
```

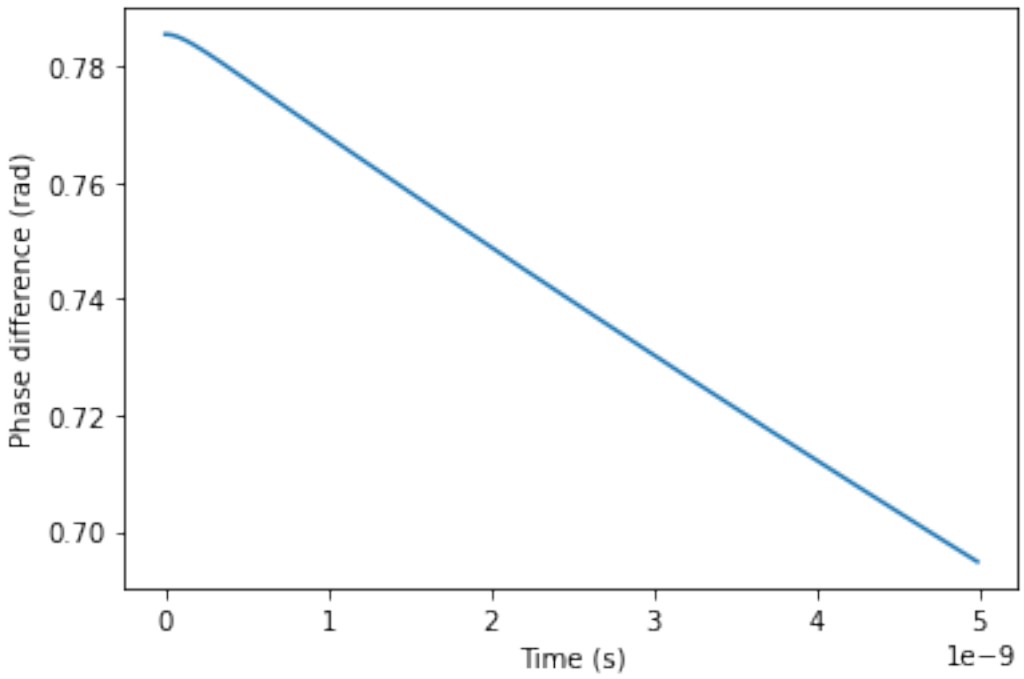
```
phi0 = h / (2 * e)  # Magnetic flux quantum

# Numerical parameters
N = 1000  # Number of time steps
dt = 1e-11  # Time step
T = N * dt  # Total time

# Initial conditions
V0 = 2 * phi0 / (R * C)  # Initial voltage
phi1 = 0  # Initial phase of junction 1
phi2 = 0  # Initial phase of junction 2

# Create arrays for time, voltage, and phase
t = np.arange(dt, T, dt)
V = np.ones(N) * V0
phi = np.zeros(N)

# Define the update function for the simulation
def update(num):
    global V, phi1, phi2
    # Calculate the current through the junctions
    I1 = Ic * np.sin(phi1)
    I2 = Ic * np.sin(phi2)
    # Calculate the voltage across the junctions
    V1 = hbar / (2 * e) * (phi2 - phi1) / dt - I1 * R
    V2 = hbar / (2 * e) * (phi1 - phi2) / dt - I2 * R
    # Calculate the voltage across the entire circuit
    V[num] = V1 - V2
    # Calculate the change in phase of the junctions
    dphi1 = 2 * np.pi * V1 / (hbar / (2 * e))
    dphi2 = 2 * np.pi * V2 / (hbar / (2 * e))
    # Update the phase of the junctions
    phi1 += dphi1 * dt
    phi2 += dphi2 * dt
    # Calculate the total phase across the junctions
    phi[num] = phi2 - phi1

# Run the simulation
for i in range(N):
    update(i)

# Plot the results
plt.figure()
plt.plot(t, phi)
plt.xlabel('Time (s)')
plt.ylabel('Phase difference (rad)')
plt.show()
```



In this modified version of the code, we set the initial value of phi to np.pi/4 (45 degrees) and the initial value of V to 0.

[ ]:
```python
import numpy as np
import matplotlib.pyplot as plt

# Simulation parameters
L = 1e-6  # Inductance
C = 1e-12  # Capacitance
R = 50  # Resistance
Ic = 1e-6  # Critical current

# Time parameters
N = 500  # Number of time steps
t_max = 5e-9  # Total simulation time
dt = t_max / N  # Time step

# Initial conditions
phi = np.zeros(N)
V = np.zeros(N)
phi[0] = np.pi/4
V[0] = 0

# Simulation loop
for i in range(N-1):
```

```
    phi[i+1] = phi[i] + dt/2 * (V[i] + V[i+1])
    V[i+1] = V[i] + dt/2 * (Ic*np.sin(phi[i+1])/C - V[i]/(R*C) - phi[i+1]/(L*C))

# Plot the results
fig, ax = plt.subplots()
ax.plot(np.arange(N)*dt, phi)
ax.set_xlabel('Time (s)')
ax.set_ylabel('Phase difference (rad)')
ax.set_title('')
plt.show()
```



The next plot will show the voltage across the junctions as a function of time, and displays the time evolution of the voltage.

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation

# Constants
h = 6.626 * 1e-34  # Planck constant
e = 1.6 * 1e-19  # Charge of an electron
hbar = h / (2 * np.pi)
R = 1e-3  # Resistance of the junction
C = 1e-15  # Capacitance of the junction
Ic = 1e-6  # Critical current of the junction
```

```
phi0 = h / (2 * e)  # Magnetic flux quantum

# Numerical parameters
N = 1000  # Number of time steps
dt = 1e-11  # Time step
T = N * dt  # Total time

# Initial conditions
V0 = 2 * phi0 / (R * C)  # Initial voltage
phi1 = 0  # Initial phase of junction 1
phi2 = 0  # Initial phase of junction 2

# Create arrays for time, voltage, and phase
t = np.arange(dt, T, dt)
V1 = np.zeros(N)
V2 = np.zeros(N)

# Define the update function for the simulation
def update(num):
    global V1, V2, phi1, phi2
    # Calculate the current through the junctions
    I1 = Ic * np.sin(phi1)
    I2 = Ic * np.sin(phi2)
    # Calculate the voltage across the junctions
    V1[num] = hbar / (2 * e) * (phi2 - phi1) / dt - I1 * R
    V2[num] = hbar / (2 * e) * (phi1 - phi2) / dt - I2 * R
    # Calculate the change in phase of the junctions
    dphi1 = 2 * np.pi * V1[num] / (hbar / (2 * e))
    dphi2 = 2 * np.pi * V2[num] / (hbar / (2 * e))
    # Update the phase of the junctions
    phi1 += dphi1 * dt
    phi2 += dphi2 * dt

# Create the animation function
fig, ax = plt.subplots()
line, = ax.plot(t, V1 - V2)

def animate(num):
    update(num)
    line.set_ydata(V1 - V2)
    return line,

# Set the animation parameters and start the animation
ani = animation.FuncAnimation(fig, animate, frames=N, interval=1, blit=True)

# Set the plot parameters and display the plot
plt.xlabel('Time (s)')
```

```
plt.ylabel('Voltage (V)')
plt.show()
```

[ ]: