# Heat Exchanger Network Simulation

This document contains the complete Python script for simulating the heat exchanger network based on the provided diagram.

import numpy as np  
from scipy.optimize import fsolve  
  
def heat\_exchanger(shell\_flow\_rate, tube\_flow\_rate, shell\_inlet\_temp, tube\_inlet\_temp, shell\_density, tube\_density, initial\_guess):  
 def equations(vars):  
 shell\_outlet\_temp, tube\_outlet\_temp = vars  
 eq1 = shell\_flow\_rate \* shell\_density \* (shell\_inlet\_temp - shell\_outlet\_temp) - \  
 tube\_flow\_rate \* tube\_density \* (tube\_outlet\_temp - tube\_inlet\_temp)  
 eq2 = shell\_outlet\_temp - tube\_inlet\_temp - 5 # Assuming a 5 degree approach temperature  
 return [eq1, eq2]  
   
 shell\_outlet\_temp, tube\_outlet\_temp = fsolve(equations, initial\_guess)  
 return shell\_outlet\_temp, tube\_outlet\_temp  
  
def forward\_flow(conditions):  
 results = {}  
   
 # E103  
 results["E103"] = heat\_exchanger(\*\*conditions["E103"])  
   
 # E104ABC  
 conditions["E104ABC"]["tube\_inlet\_temp"] = results["E103"][1]  
 results["E104ABC"] = heat\_exchanger(\*\*conditions["E104ABC"])  
   
 # E105  
 conditions["E105"]["tube\_inlet\_temp"] = results["E104ABC"][1]  
 results["E105"] = heat\_exchanger(\*\*conditions["E105"])  
   
 # Continue for other exchangers (E106AB, E107, E108AB, E109, E110, E111ABC)  
 # ...  
   
 return results  
  
def process\_flow(conditions):  
 results = forward\_flow(conditions)  
   
 # Feedback from E111ABC to E108AB  
 conditions["E108AB"]["shell\_inlet\_temp"] = results["E111ABC"][0]  
   
 # Feedback from E108AB to E104ABC  
 conditions["E104ABC"]["shell\_inlet\_temp"] = results["E108AB"][0]  
   
 # Feedback from E105 to E104ABC  
 conditions["E104ABC"]["shell\_inlet\_temp"] = (conditions["E104ABC"]["shell\_inlet\_temp"] + results["E105"][0]) / 2  
   
 return results, conditions  
  
def run\_simulation(initial\_conditions, num\_iterations=5):  
 conditions = initial\_conditions.copy()  
 all\_results = []  
   
 for i in range(num\_iterations):  
 print(f"Iteration {i+1}")  
 results, conditions = process\_flow(conditions)  
 all\_results.append(results)  
   
 return all\_results  
  
# Initial conditions (example values, adjust as needed)  
initial\_conditions = {  
 "E103": {"shell\_flow\_rate": 100, "tube\_flow\_rate": 100, "shell\_inlet\_temp": 150, "tube\_inlet\_temp": 30,  
 "shell\_density": 1000, "tube\_density": 1000, "initial\_guess": (140, 40)},  
 "E104ABC": {"shell\_flow\_rate": 100, "tube\_flow\_rate": 100, "shell\_inlet\_temp": 160, "tube\_inlet\_temp": None,  
 "shell\_density": 1000, "tube\_density": 1000, "initial\_guess": (150, 50)},  
 # Add similar entries for E105, E106AB, E107, E108AB, E109, E110, E111ABC  
}  
  
# Run the simulation  
simulation\_results = run\_simulation(initial\_conditions)  
  
# Print results  
for i, results in enumerate(simulation\_results):  
 print(f"\nIteration {i+1} Results:")  
 for exchanger, (shell\_out, tube\_out) in results.items():  
 print(f"{exchanger}: Shell out = {shell\_out:.2f}, Tube out = {tube\_out:.2f}")