```
clear()
clearglobal()

//--- Input and defs ------------------------------------------------------

thispath = "C:\Users\Henry\Desktop\quasi-steady-sim-yogurt\";

exec(thispath+'func-thermo.sce', -1)   // Thermo funcs and consts
exec(thispath+'input_vars.sce', -1)      // Input data (Will change)
//exec(thispath+'func-simul.sce', -1)    // Put all your functions here!!!!

// Initial Values
T = [308.16176   308.16176   316.15001   320.61251   320.2396   308.16176];

// Air Temperatures through Rock Bed
Tr = [T_b0-15 T_b0-10 T_b0-8 T_b0-4];

// Air Temperatures through Yogurt
Ts = [T_y0 T_y0-5 T_y0-7 T_y0-10];

//Yogurt stacks averages temperatures
Ty = [293.15+10 293.15+12 293.15+15];

// Change in Yogurt stacks temperatures
dTy = [ 0 0 0 ];

// rockbed average temperature
Tb = [326.15 326.15-5 326.15-10];

// Change in rockbed temperatures
dTb = [ 0 0 0 ];

// Pressure drop across Rockbed
Pd = 0;

// Pressure drop across yogurt stack
Py = 0;

// number of iteration
num_iteration = 10

// Volume flow rate;
Qr = 3.1734; // 2.647
rpm_set = 900
Qr_ = Qr/n_fan
Q_star = Qr_*rpm_ref/rpm_set
Ab = W_b*H_b;

pressure_pump = -13.649*Q_star^2+73.946*Q_star+1055.2
pressure_drop = 0;
for k=1:num_iteration
    // FAN
    T(2) = T(1);                // no temperature change accross fan
    Tr(1) = T(2);               //
    Pd = 0;                     // Pressure drop accross Rockbed
    Py = 0;                     // Pressure drop accross yogurt
    for j = 1:n_b,
        T_avg = 0.5*( Tr(j)+Tr(j+1) );                         // average temperature
        rho_avg = rho_air( T_avg) ;                            // density of air
        mu_avg = mu_air( T_avg );                              // dynamic viscosity of air
```

```
        cp_avg = cp_air( T_avg );                    // specific heat capacity of air
        k_avg = k_air( T_avg );                      // thermal conductivity of air
        G_avg = rho_avg*(Qr/Ab);                     // mass flow rate per unit area
        Re_avg = G_avg*d_p/(mu_avg);                 // Reynolds Num

 f_avg = 4.466*((Re_avg)^0.2)*((psi)^0.696)*((eps_b)^-2.945)*exp( 11.85*( log(psi) )^2 ); // friction factor

 Nu_avg = 0.437*((Re_avg)^0.75)*((psi)^3.35)*((eps_b)^-1.620)*exp( 29.03*( log(psi) )^2 ); // Nusselt Number
        h_avg = (k_avg*Nu_avg)/(d_p^2);              // Convection coefficient

 Tr(j+1) = Tb(j) - ( Tb(j)-Tr(j) )*exp( (-h_avg*( L_b/n_b ) )/( cp_avg*G_avg ) );

        // change in temperature in rockbed

 dTb(j) = (cp_avg*G_avg*(Tr(j)-Tr(j+1)))/(2640*820*(1-Tr(j+1))*(L_b/n_b))
        // Pressure drop across control volume
        Pd = -f_avg*G_avg^2*(L_b/n_b)/(rho_avg*d_p) + Pd;
        disp(Pd)

    end
    Pd = eta_r*Pd                                // Pressure loss accross rock bed
    T(3) = Tr(n_b+1);

    // Heater
    T_avg = (T(3)+T(4))/2;                       // Average Temperature
    cp_avg = cp_air(T_avg);                      // Heat capacity air
    rho_avg = rho_air(T_avg);                    // density of air
    q_h = KP*(T_set-Ts(1))+KI*0                  // heat added
    T(4) = T(3) + q_h/(cp_avg*rho_avg*Qr)        // temperature change

    // Environemental Losses
    T_avg = (T(4)+T(5))/2;                       // average temperature
    cp_avg = cp_air(T_avg);                      // Heat capacity air
    rho_avg = rho_air(T_avg);                    // density of air
    q = rho_avg*Qr*cp_avg*(T(4)-T(5));

    T(5) = T_env + (T(4)-T_env)/(exp(h_env*A_env/(rho_avg*Qr*cp_avg)));
    Ts(1) = T(5);

    // Yogurt Warming
    for i = 1:n_s,
        T_avg = 0.5*(Ts(i)+Ts(i+1));             // Average Temperature
        rho_avg = rho_air(T_avg);                // density of air
        mu_avg = mu_air(T_avg);                  // dynamic viscosity of air
        cp_avg = cp_air(T_avg);                  // Heat capacity air
        k_avg = k_air(T_avg);                    // thermal conductivity of air
        Pr_avg = Pr_air(T_avg);                  // Prantel Number of Air
        Pr_avg_e = Pr_air(Ts(i+1));              // Prantel Number of air for next control volume
        nu_avg = nu_air(T_avg);                  // Nusselt Number of air
        As = N_Ts*S_Ts*H_y;                      // Effective Area
        mu_s = Qr/As;
        Re_avg = mu_s*D_y/(nu_avg);                  // Reynolds Numbers
```

```
        if Re_avg > 10e3 || Re_avg < 2*10e5 then            // Determine correct Nusselt number equation based on Re
            Nu_d =  0.72*F_s*Re_avg^0.63*Pr_avg^0.365*(Pr_avg/Pr_avg_e)^0.25;  // Nusselt Number
        elseif Re_avg > 2*10e5 || Re_avg < 2*10e6  then
            Nu_d = 0.033*F_s*Re_avg^0.8*Pr_avg^0.4*(Pr_avg/Pr_avg_e)^0.25;  // Nusselt Number
        end,
        m_y = (%pi)/4*rho_avg*(D_y)^2*H_y*N_Ls*N_Ts;
        h_avg = Nu_d*k_avg/(D_y);
        A_y = %pi*D_y*N_Ls*N_Ts;


 Ts(i+1) = Ty(i) - (Ty(i) - Ts(i))*exp( -1*(A_y*h_avg)/(rho_avg*Qr*cp_avg) );

        // change in yogurt stack temperatures
        dTy(i) = (rho_avg*Qr)/(3520*m_y)*(Ts(i)-Ts(i+1))
        // Pressure drop
        // Fix friction factor
        Py = Py + -N_Ls*0.3*(rho_avg*mu_s^2)/2;
    end
    Py = eta_s*Py;
    T(6) = Ts(n_s+1);
    if abs( (T(1)-T(6))/(T(6))) < 1e-8  then
        k =num_iteration;
    else
        T(1) = T(6);                // closed system
    end

    pressure_drop = Py+Pd;
end

disp('k')
disp(k);
disp('T')
disp(T);
disp('Pressure pump')
disp(pressure_pump)
disp('pressure drop')
disp(pressure_drop)
disp('Pressure pump - pressure drop')
disp(pressure_pump + pressure_drop)

disp('change in rockbed temperature')
disp(dTb)
disp('change in yogurt temperature')
disp(dTy)
```