ALLEGATO 1:
CODICE MATLAB PER IL CALCOLO DELLA PRESSIONE PER IL FLUSSO DI CO_2

```
%% CODICE PER IL CALCOLO DELLA PRESSIONE PER IL FLUSSO DI CO2

% Temperatura espressa in Kelvin
% Pressione espressa in Pascal
%
% stato liquido : modello fluido incomprimibile (se le condizioni supercritiche
% sono raggiunte attraversando la curva di saturazione, il fluido
% supercritico è trattato come incomprimibile)
% stato gas & supercritico: modello gas comprimibile con fattore di comprimibilità Z
% primo nodo testa pozzo, ultimo nodo sbocco in giacimento
% imponendo una pressione, temperatura e portata il codice calcola:
% evoluzione termodinamica nel pozzo, pressione temperatura e velocità,
% proprietà di interesse e altezza cambio fase
%
% caricamento proprieetà in matrici nella forma:
%      p[MPa] 0,3 . # 148 colonne . 15
% T[K]
% 280
% .
% # 41
% righe
% .
% 320
%
% Codice sviluppato da Francesco Pertuso
%

clear; clc; close all;
load('cp_CO2.mat')
load('density_CO2.mat')
load('th_conductivity_CO2.mat')
load('viscosity_CO2.mat')
load('viscosity_H2O.mat')
%% Input data
z = 2021;              % [m]       profondità pozzo
r_in = 0.0482;         % [m]       raggio interno
r_out = 0.0571;        % [m]       raggio esterno
r_ce = 0.24;           % [m]       raggio casing cemento
D = 2*r_in;            % [m]       diamentro interno
A = pi*r_in^2;         % [m^2]     sezione interna
eps = 0.045e-3;        % [m]       rugosità
g = 9.81;              % [m/s^2]   costante di gravità
k_ss = 50;             % [W/m/K]   conducibilità acciaio
k_ce = 2.30;           % [W/m/K]   conducibilità cemento
R = 8.314462618;       % [J/mol/K] costante universale dai gas
M = 0.0440095;         % [kg/mol]  massa molare CO2
p_crit = 73.8e5;       % [Pa]      pressione critica
T_crit = 30.97+273.15;% [K]       temperatura critica
a = -0.015;            % [K/m]     gradiente geotermico
c = 288.45-0.615;      % [K]       temperatura sul fondale marino - 41m per scambio convettivo
Tgeo = @(z) c - a*z;   % [K]       z=0 profondità reservoir
% figure
% plot(Tgeo(zz),zz)
% title('geothermal gradient')
% grid on
% ylabel('well depth [m] -from surface to bottom-')
% xlabel('temperature [K]')
% set(gca, 'YDir','reverse')
%% Proprietà dell'acqua @ T = 16°C p = 1-2 bar (circa costanti)
T_H2O = 16 + 273.15;   % [K]       temperatura
rho_H2O = 998.95;      % [kg/m3]   densità
cp_H2O = 4187.4;       % [J/kg/K]  calore specifico
mu_H2O = 0.0011078;    % [Pa*s]    viscosità
k_th_H2O = 0.59077;    % [W/m/K]   conducibilità termica
u_H2O = 0.8;           % [m/s]     velocità
Re_H2O = rho_H2O*u_H2O*2*r_out/mu_H2O; % numero di Reynolds
Pr_H2O = cp_H2O*mu_H2O/k_th_H2O;       % numero di Prandlt
%% Property dell'aria @ T = 15°C p = 1 bar
T_air =  15 + 273.15; % [K]       temperatura
rho_air = 1.225;       % [kg/m3]   densità
cp_air = 1004.3;       % [J/kg/K]  calore specifico
mu_air = 1.781e-5;     % [Pa*s]    viscosità
k_th_air = 25.5e-3;    % [W/m/K]   conducibilità termica
u_air = 5;             % [m/s]     velocità
Re_air = rho_air*u_air*2*r_out/mu_air; % numero di Reynolds
Pr_air = cp_air*mu_air/k_th_air;       % numero di Prandlt
%% Discretizzazione dominio computazionale
dz = 1;
zz = (0:dz:z)';
S = 2*pi*dz*r_in;      % [m^2] superficie interna di scambio termico
% preallocazione vettori
p = zeros(size(zz));
T = zeros(size(zz));
v = zeros(size(zz));
rho = zeros(size(zz));
cp = zeros(size(zz));
mu = zeros(size(zz));
```

```matlab
k_th = zeros(size(zz));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%condizioni iniziali%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
m = -;              % [kg/s] portata in massa
p_in = - + 1000;    % [Pa] pressione iniziale + 0.001 MPa per interpolazione
T_in = - + 273.15;  % [K] temperatura iniziale
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Condizioni primo nodo
p(1) = p_in;
T(1) = T_in;
v(1) = m/CO2_density(p(1),T(1),density_CO2_m)/A;
i_change = 0;       % ipotesi stato di partenza gassoso, viene corretto se ipotesi sbagliata
% controllo sullo stato termodinamico
if T(1) < T_crit && p(1) > CO2_saturation_pressure(T(1))
    incompressible = 1;
    i_change = 1;
    'stato di partenza liquido' %#ok<NOPTS>
else
    incompressible = 0;
    'stato di partenza gassoso' %#ok<NOPTS>
end
first = 1;  % flag per prima iterazione di U
for ii=1:length(zz)-1
    % calcolo proprietà per il nodo ii
    rho(ii) = CO2_density(p(ii),T(ii),density_CO2_m);
    cp(ii) = CO2_cp(p(ii),T(ii),cp_CO2_m);
    mu(ii) = CO2_viscosity(p(ii),T(ii),viscosity_CO2_m);
    k_th(ii) = CO2_th_conductivity(p(ii),T(ii),th_conductivity_CO2_m);
    % modello fluido incomprimibile Ma<0.3
    if incompressible == 1
        % numero di Reynolds
        Re = m*D/mu(ii)/A;
        % fattore di attrito
        f = (-1.8*log((eps/2/r_in/3.78)^1.11 + 6.9/Re))^(-0.5);
        % equazione pressione -dalla conservazione del momento-
        p(ii+1) = p(ii) + rho(ii)*g*dz - f*dz/D*m^2/rho(ii)/A^2;
        if T(ii) <= T_crit && p(ii+1) <= CO2_saturation_pressure(T(ii))
            warning('cambiamento di fase da liquido a gas')
            incompressible = 0;
            i_change_2 = ii;
        end
    % modello gas reale
    else
        % fattore di comprimibilità
        Z = p(ii)/rho(ii)/R*M/T(ii);
        % numero di Reynolds
        Re = m*D/mu(ii)/A;
        % fattore di attrito
        f = (-1.8*log((eps/D/3.78)^1.11 + 6.9/Re))^(-0.5);
        % equazione pressione -dalla conservazione del momento-
        p(ii+1) = p(ii) + (dz*g*p(ii)*M/R/T(ii)/Z - dz*f*m^2*R*T(ii)/4/r_in*Z/p(ii)/M)...
        /(1-m^2*R*T(ii)*Z/M/p(ii)^2);
    end
    if p(ii) < p_crit % usare correlazione Dittus Boelter correlation
        % numero di Prandlt
        Pr = cp(ii)*mu(ii)/k_th(ii);
        % controllo valitità correlazione Dittus-Boelter
        if Pr < 0.6 || Pr > 160
            error('numero di Prandlt fuori dal range di validità della correlazione di Dittus Boelter')
        else
            if Re < 1e4
                error('numero di Reynolds fuori dal range di validità della correlazione di Dittus Boelter')
            end
        end
        % numero di Nusselt -dalla correlazione di Dittus Boelter-
        Nu = 0.0243 * Re^(4/5) * Pr^(0.4);
        % coefficiente di scambio temrico convettivo interno
        h_in = Nu*k_th(ii)/D;
     else % usare correlazione CO2 supercritica di Gupta
         % calcolo il flusso termico con il coefficiente U dello step precedente
         phi = (T(ii)-Tgeo(zz(ii)))*U;
         % calcolo la temperatura a parete attraverso l'analogia elettrica
         T_wall = T(ii) - phi/h_in;
         % calcolo i le proprietà con la temperatura a parete
         k_th_wall = CO2_th_conductivity(p(ii),T_wall,th_conductivity_CO2_m);
         mu_wall = CO2_viscosity(p(ii),T_wall,viscosity_CO2_m);
         rho_wall = CO2_density(p(ii),T_wall,density_CO2_m);
         cp_wall = CO2_cp(p(ii),T_wall,cp_CO2_m);
         Pr_wall = cp_wall*mu_wall/k_th_wall;
         Re_wall = m*D/mu_wall/A;
         % numero di Nusselt -dalla correlazione di Gupta et al.-
         Nu = 0.0038 * Re_wall^0.96 * Pr_wall^-0.14 * (rho_wall/rho(ii))^0.84 ...
             * (k_th_wall/k_th(ii))^-0.75 *(mu_wall/mu(ii))^-0.22;
         % coefficiente di scambio temrico convettivo interno
         h_in = Nu*k_th(ii)/D;
    end
    % scambio termico convettivo esterno,
    % nei primi 41 metri il pozzo non è a contatto con la crosta terrestre
```

```
    if zz(ii) < 41
        if zz(ii) <= 21 % fluido esterno: aria
            if first == 1
                % at first step wall and bulk temperature are the same
                U = 0;
            end
            % temperatura a parete dalla conservazione radiale del flusso
            T_w = T(ii) - U*S*(T(ii)-T_air)*log(r_out/r_in)/(2*pi*dz*k_ss);
            mu_w = air_viscosity(T_w);
            % numero di Nusselt -dalla correlazione di E.M.Sparrow et al.-
            Nu1 = 0.025 + (0.4*Re_air^(1/2) + 0.06*Re_air^(2/3))*Pr_air^(0.37)*(mu_air/mu_w)^(1/4);
            % coefficiente convezione esterna per l'aria
            h_out = Nu1*k_th_air/D;
            % coefficiente di scambio termico globale -convezione & conduzione & convezione-
            U = 1/(1/h_in+r_in/k_ss*log(r_out/r_in)+r_in/r_out/h_out);
            % equazione temperatura -dall'equazione del calore, pure advection problem-
            T(ii+1) = T(ii) + S*U/m/cp(ii)*(T_air - T(ii));
        else
            % fluido estrno: acqua
            % temperatura a parete dalla conservazione radiale del flusso
            T_w = T(ii) - U*S*(T(ii)-T_H2O)*log(r_out/r_in)/(2*pi*dz*k_ss);
            mu_w = H2O_viscosity(T_w,viscosity_H2O);
            % numero di Nusselt -dalla correlazione di E.M.Sparrow et al.-
            Nu1 = 0.025 + (0.4*Re_H2O^(1/2) + 0.06*Re_H2O^(2/3))*Pr_H2O^(0.37)*(mu_H2O/mu_w)^(1/4);
            % coefficiente convezione esterna per l'acqua
            h_out = Nu1*k_th_H2O/D;
            % coefficiente di scambio termico globale -convezione & conduzione & convezione-
            U = 1e-2*1/(1/h_in+r_in/k_ss*log(r_out/r_in)+r_in/r_out/h_out);
            % equazione temperatura -dall'equazione del calore, pure advection problem-
            T(ii+1) = T(ii) + S*U/m/cp(ii)*(T_H2O - T(ii));
        end
    else
        % scambio termico conduttivo esterno,
        % pozzo a contatto con la crosta terrestre
        % coefficiente di scambio termico globale -convezione & conduzione-
        U = 1/(1/h_in+r_in/k_ss*log(r_out/r_in)+r_out/k_ce*log(r_ce/r_out));
        % equazione temperatura -dall'equazione del calore, constant surface temperature-
        T(ii+1) = T(ii) + S*U/m/cp(ii)*(Tgeo(zz(ii))-T(ii));
    end
    % equazione della velocità -dalla continuità-
    v(ii+1) = m/rho(ii)/A;
    % controllo massima pressione del reservoir
    if p(ii+1) > 145.1e5
        strcat(['massima pressione del reservoir raggiunta nel pozzo a \' ...
            ''],num2str(2000 - zz(ii+1)),' metri dallo sbocco')
        ii = ii+1;  %#ok<FXSET>
        break
    end
    % controllo cambiamento di fase gas->liquido
    if i_change == 0
        if  T(ii+1) < T_crit && p(ii+1) >= CO2_saturation_pressure(T(ii+1))
            incompressible = 1;
            i_change = ii;
            warning('cambiamento di fase da gas a liquido')
        else
                % controllo cambiamento di fase sup_gas->sup_liquido
            if p(ii+1) >= CO2_saturation_pressure(T(ii+1))
                i_change = ii;
                warning('cambiamento di fase da gas supercritico a liquido supercritico')
            end
        end
    end
end

%% Plot section
% set defaults for figures
set(0,'defaultlinelinewidth',2)
set(0,'defaultaxesfontsize',20)
set(0,'defaulttextfontsize',20)
set(0,'defaultlegendautoupdate','off')
close all
strcat('pressione finale:',num2str(p(ii)/1e5),' bar')
%% Evoluzione termodinamica nel pozzo
temp = 217:0.1:T_crit; % [K]
temp_1 = T_crit:0.1:340;
p_sat = zeros(size(temp));
p_pscr = zeros(size(temp_1));
for i_sat = 1:length(temp)
    p_sat(i_sat) = CO2_saturation_pressure(temp(i_sat));
end
for i_sat = 1:length(temp_1)
    p_pscr(i_sat) = CO2_saturation_pressure(temp_1(i_sat));
end
figure
tiledlayout('flow')
if i_change == 0    % senza cambiamento di fase
    nexttile
```

```
    hold on
    plot(T(1:ii)-273.15,p(1:ii)/1e5)
    plot(temp-273.15,p_sat/1e5,'k-',temp_1-273.15,p_pscr/1e5,'k-.')
    plot([T_crit-273.15,400-273.15],[p_crit,p_crit]./1e5,'k--')
    plot([T_crit-273.15,T_crit-273.15],[p_crit/1e5,145],'k--')
    title('EVOLUZIONE TERMODINAMICA')
    xlabel('Temperatura [°C]')
    ylabel('Pressione [bar]')
    legend('evoluzione termodinamica','curva saturazione','curva pseudocritica' ...
        ,'condizioni supercritiche','Location','best')
    grid on
    xlim([273-273.15,400-273.15])
    ylim([20,145])
    %% Evoluzione della pressione nel pozzo
    nexttile
    plot(p(1:ii)./1e5,zz(1:ii))
    title('EVOLUZIONE PRESSIONE')
    xlabel('Pressione [bar]')
    ylabel('Profondità pozzo [m]')
    box on
    grid minor
    set(gca, 'YDir','reverse')
    %% Evoluzione della temperatura nel pozzo
    nexttile
    plot(T(1:ii)-273.15,zz(1:ii),Tgeo(zz(41/dz:ii))-273.15,zz(41/dz:ii))
    box on
    title('EVOLUZIONE TEMPERATURA')
    xlabel('Temperatura [°C]')
    ylabel('Profondità pozzo [m]')
    legend('Temperatura fluido','Temperatura crosta terrestre','Location','best')
    grid minor
    set(gca, 'YDir','reverse')
    %% Evoluzione velocità nel pozzo
    nexttile
    plot(v(1:ii),zz(1:ii))
    title('EVOLUZIONE VELOCITÀ')
    xlabel('Velocità [m/s]')
    ylabel('Profondità pozzo [m]')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    %% Evoluzione proprietà nel pozzo
    figure
    tiledlayout('flow')
    % densità
    nexttile
    plot(rho(1:ii),zz(1:ii))
    title('EVOLUZIONE DENSITÀ')
    xlabel('ρ [kg/m^3]')
    ylabel('Profondità pozzo [m]')
    grid minor
    set(gca, 'YDir','reverse')
    % viscosità
    nexttile
    plot(mu(1:ii),zz(1:ii))
    title('EVOLUZIONE VISCOSITÀ')
    xlabel('μ [Pa*s]')
    ylabel('Profondità pozzo [m]')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    % calore specifico
    nexttile
    plot(cp(1:ii),zz(1:ii))
    title('EVOLUZIONE CALORE SPECIFICO')
    xlabel('c_p [J/Kg/K]')
    ylabel('Profondità pozzo [m]')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    % conducibilità termica
    nexttile
    plot(k_th(1:ii),zz(1:ii))
    title('EVOLUZIONE CONDUCIBILITÀ TERMICA')
    xlabel('k [W/m/K]')
    ylabel('Profondità pozzo [m]')
    grid minor
    box on
    set(gca, 'YDir','reverse')
else    % con cambiamento di fase
    strcat('altezza cambio fase:', num2str(zz(i_change)),' m')
    nexttile
    hold on
    %% Evoluzione termodinamica nel pozzo
    plot(T(1:ii)-273.15,p(1:ii)/1e5,T(i_change)-273.15,p(i_change)/1e5,'x')
    plot(temp-273.15,p_sat/1e5,'k-',temp_1-273.15,p_pscr/1e5,'k-.')
    plot([T_crit-273.15,400-273.15],[p_crit,p_crit]./1e5,'k--')
```

```
    plot([T_crit-273.15,T_crit-273.15],[p_crit/1e5,145],'k--')
    title('EVOLUZIONE TERMODINAMICA')
    xlabel('Temperatura [°C]')
    ylabel('Pressione [bar]')
    legend('evoluzione termodinamica','passaggio da gas-like a liquid-like',['curva ' ...
        'saturazione'],'curva pseudocritica','condizioni supercritiche','Location','best')
    grid on
    box on
    xlim([273-273.15,400-273.15])
    ylim([20,145])
    %% Evoluzione della pressione nel pozzo
    nexttile
    plot(p(1:ii)./1e5,zz(1:ii),p(i_change)/1e5,zz(i_change),'x')
    title('EVOLUZIONE PRESSIONE')
    xlabel('Pressione [bar]')
    ylabel('Profondità pozzo [m]')
    legend('','passaggio da gas-like a liquid-like','Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    %% Evoluzione della temperatura nel pozzo
    nexttile
    plot(T(1:ii)-273.15,zz(1:ii),Tgeo(zz(41/dz:ii))-273.15,zz(41/dz:ii),T(i_change)-273.15,zz(i_change),'x')
    title('EVOLUZIONE TEMPERATURA')
    xlabel('Temperatura [°C]')
    ylabel('Profondità pozzo [m]')
    legend('Temperatura fluido','Temperatura crosta terrestre',['passaggio da gas-like a ' ...
        'liquid-like'],'Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    %% Evoluzione della velocità nel pozzo
    nexttile
    plot(v(1:ii),zz(1:ii),v(i_change),zz(i_change),'x')
    title('EVOLUZIONE VELOCITÀ')
    xlabel('Velocità [m/s]')
    ylabel('Profondità pozzo [m]')
    legend('','passaggio da gas-like a liquid-like','Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    %% Evoluzione proprietà nel pozzo
    figure
    tiledlayout('flow')
    % densità
    nexttile
    plot(rho(1:ii),zz(1:ii),rho(i_change),zz(i_change),'x')
    title('EVOLUZIONE DENSITÀ')
    xlabel('ρ [kg/m^3]')
    ylabel('Profondità pozzo [m]')
    legend('','passaggio da gas-like a liquid-like','Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    % viscosità
    nexttile
    plot(mu(1:ii),zz(1:ii),mu(i_change),zz(i_change),'x')
    title('EVOLUZIONE VISCOSITÀ')
    xlabel('μ [Pa*s]')
    ylabel('Profondità pozzo [m]')
    legend('','passaggio da gas-like a liquid-like','Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    % calore specifico
    nexttile
    plot(cp(1:ii),zz(1:ii),cp(i_change),zz(i_change),'x')
    title('EVOLUZIONE CALORE SPECIFICO')
    xlabel('c_p [J/Kg/K]')
    ylabel('Profondità pozzo [m]')
    legend('','passaggio da gas-like a liquid-like','Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
    % condicibilità termica
    nexttile
    plot(k_th(1:ii),zz(1:ii),k_th(i_change),zz(i_change),'x')
    title('EVOLUZIONE CONDUCIBILITÀ TERMICA')
    xlabel('k [W/m/K]')
    ylabel('Profondità pozzo [m]')
    legend('','passaggio da gas-like a liquid-like','Location','best')
    grid minor
    box on
    set(gca, 'YDir','reverse')
end
figure(1)

```

```
%% FUNZIONI PROPRIETA' TERMODINAMICHE CO2

function density = CO2_density(p,T,density_CO2_m)
% questa funzione interpola linearmente il valore di densità
% sul piano pressione-temperatura
% range pressione: 3  < p [MPa] < 15
%                 30 < p [bar] < 150
% discretizzazione 0.02 MPa
%
% range temperatura: 7  < T [°C] < 127
%                   280 < T [K]  < 400
% discretizzazione 0.5 K
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Questo script lavora con pressione in MPa e temperatura in K %
% è necessario passare le variabili alla funzione in Pa e K    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% basato sul database NIST

% controllo parametri di input
if T < 280 || T > 400
    error('Temperature out of boundaries')
end
if p < 3e6 || p > 15e6
    error('pressure out of boundaries')
end
% conversione Pa in MPa
p = 1e-6 * p;
% trovare i valori di pressione di riferimento nella griglia discretizzata con 0.02 MPa
                                          % example p = 3.023 MPa
p = p*100;                                % 302.3
p_floor = floor(p)/100;                   % 3.020
if mod(p_floor,0.02) ~= 0                 % mod = 0
    p_floor = p_floor - mod(p_floor,0.02);
end
p_ceil = ceil(p)/100;
if mod(p_ceil,0.02) ~= 0                    % mod = 0.01
    p_ceil = p_ceil + 0.02-mod(p_ceil,0.02);  % 3.04end
end
% riportare la pessione in MPa
p = p/100;
% trovare i valori di temperatura di riferimento nella griglia discretizzata con 0.5 K
                                          % example T = 290.5832 K
T = 10*T;                                 % 2905.5832
T_floor = floor(T)/10;                    % 290.5000
if mod(T_floor,0.5) ~= 0                  % mod = 0
    T_floor = T_floor - mod(T_floor,0.5);
end
T_ceil = ceil(T)/10;                      % 290.6000
if mod(T_ceil,0.5) ~= 0                   % mod = 0.1
    T_ceil = T_ceil + 0.5-mod(T_ceil,0.5);  % 291.0
end
% riportare temperatura in K
T = T/10;
% X->p [Mpa] Y->T [K] riferimento
X = [p_floor; p_ceil];
Y = [T_floor; T_ceil];
% riporto i valori numerici di pressione e temperatura in indici nelle
% parentesi ed estraggo dalla matrice di riferimento i valori di riferimento
p_T_floor = density_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_floor-3)*50));
p_ceil_T_floor = density_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_ceil-3)*50));
p_floor_T_ceil = density_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_floor-3)*50));
p_T_ceil = density_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_ceil-3)*50));
% V è la matrice con i valori di output in X,Y
V = [p_T_floor, p_ceil_T_floor; p_floor_T_ceil, p_T_ceil];

density = interp2(X,Y,V,p,T,'linear');
end

function cp = CO2_cp(p,T,cp_CO2_m)
% questa funzione interpola linearmente il valore del calore
% specifico a pressione costante sul piano pressione-temperatura
% range pressione: 3  < p [MPa] < 15
%                 30 < p [bar] < 150
% discretizzazione 0.02 MPa
%
% range temperatura: 7  < T [°C] < 127
%                   280 < T [K]  < 400
% discretizzazione 0.5 K
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Questo script lavora con pressione in MPa e temperatura in K %
% è necessario passare le variabili alla funzione in Pa e K    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% basato sul database NIST

% controllo parametri di input
if T < 280 || T > 400
    error('Temperature out of boundaries')
```

```
end
if p < 3e6 || p > 15e6
    error('pressure out of boundaries')
end
% conversione Pa in MPa
p = 1e-6 * p;
% trovare i valori di pressione di riferimento nella griglia discretizzata con 0.02 MPa
                                        % example p = 3.023 MPa
p = p*100;                              % 302.3
p_floor = floor(p)/100;                 % 3.020
if mod(p_floor,0.02) ~= 0               % mod = 0
    p_floor = p_floor - mod(p_floor,0.02);
end
p_ceil = ceil(p)/100;
if mod(p_ceil,0.02) ~= 0                  % mod = 0.01
    p_ceil = p_ceil + 0.02-mod(p_ceil,0.02);  % 3.04
end
% riportare la pessione in MPa
p = p/100;
% trovare i valori di temperatura di riferimento nella griglia discretizzata con 0.5 K
                                        % example T = 290.5832 K
T = 10*T;                               % 2905.5832
T_floor = floor(T)/10;                  % 290.5000
if mod(T_floor,0.5) ~= 0                % mod = 0
    T_floor = T_floor - mod(T_floor,0.5);
end
T_ceil = ceil(T)/10;                    % 290.6000
if mod(T_ceil,0.5) ~= 0                 % mod = 0.1
    T_ceil = T_ceil + 0.5-mod(T_ceil,0.5);  % 291.0
end
% riportare temperatura in K
T = T/10;
% X->p [Mpa] Y->T [K] riferimento
X = [p_floor; p_ceil];
Y = [T_floor; T_ceil];
% riporto i valori numerici di pressione e temperatura in indici nelle
% parentesi ed estraggo dalla matrice di riferimento i valori di riferimento
p_T_floor = cp_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_floor-3)*50));
p_ceil_T_floor = cp_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_ceil-3)*50));
p_floor_T_ceil = cp_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_floor-3)*50));
p_T_ceil = cp_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_ceil-3)*50));
% V è la matrice con i valori di output in X,Y
V = [p_T_floor, p_ceil_T_floor; p_floor_T_ceil, p_T_ceil];

cp = interp2(X,Y,V,p,T,'linear');
end

function th_conductivity = CO2_th_conductivity(p,T,th_conductivity_CO2_m)
% questa funzione interpola linearmente il valore della conducibilità
% termica sul piano pressione-temperatura
% range pressione: 3  < p [MPa] < 15
%                 30 < p [bar] < 150
% discretizzazione 0.02 MPa
%
% range temperatura: 7  < T [°C] < 127
%                   280 < T [K]  < 400
% discretizzazione 0.5 K
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Questo script lavora con pressione in MPa e temperatura in K %
% è necessario passare le variabili alla funzione in Pa e K    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% basato sul database NIST

% controllo parametri di input
if T < 280 || T > 400
    error('Temperature out of boundaries')
end

if p < 3e6 || p > 15e6
    error('pressure out of boundaries')
end
% conversione Pa in MPa
p = 1e-6 * p;
% trovare i valori di pressione di riferimento nella griglia discretizzata con 0.02 MPa
                                        % example p = 3.023 MPa
p = p*100;                              % 302.3
p_floor = floor(p)/100;                 % 3.020
if mod(p_floor,0.02) ~= 0               % mod = 0
    p_floor = p_floor - mod(p_floor,0.02);
end
p_ceil = ceil(p)/100;
if mod(p_ceil,0.02) ~= 0                  % mod = 0.01
    p_ceil = p_ceil + 0.02-mod(p_ceil,0.02);  % 3.04
end
% riportare la pessione in MPa
p = p/100;
% trovare i valori di temperatura di riferimento nella griglia discretizzata con 0.5 K
```

```
                                        % example T = 290.5832 K
T = 10*T;                               % 2905.5832
T_floor = floor(T)/10;                  % 290.5000
if mod(T_floor,0.5) ~= 0                % mod = 0
    T_floor = T_floor - mod(T_floor,0.5);
end
T_ceil = ceil(T)/10;                    % 290.6000
if mod(T_ceil,0.5) ~= 0                 % mod = 0.1
    T_ceil = T_ceil + 0.5-mod(T_ceil,0.5);  % 291.0
end
% riportare temperatura in K
T = T/10;
% X->p [Mpa] Y->T [K] riferimento
X = [p_floor; p_ceil];
Y = [T_floor; T_ceil];
% riporto i valori numerici di pressione e temperatura in indici nelle
% parentesi ed estraggo dalla matrice di riferimento i valori di riferimento
p_T_floor = th_conductivity_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_floor-3)*50));
p_ceil_T_floor = th_conductivity_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_ceil-3)*50));
p_floor_T_ceil = th_conductivity_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_floor-3)*50));
p_T_ceil = th_conductivity_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_ceil-3)*50));
% V è la matrice con i valori di output in X,Y
V = [p_T_floor, p_ceil_T_floor; p_floor_T_ceil, p_T_ceil];

th_conductivity = interp2(X,Y,V,p,T,'linear');
end

function viscosity = CO2_viscosity(p,T,viscosity_CO2_m)
% questa funzione interpola linearmente il valore della viscosità
% sul piano pressione-temperatura
% range pressione: 3  < p [MPa] < 15
%                 30 < p [bar] < 150
% discretizzazione 0.02 MPa
%
% range temperatura: 7  < T [°C] < 127
%                   280 < T [K]  < 400
% discretizzazione 0.5 K
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Questo script lavora con pressione in MPa e temperatura in K %
% è necessario passare le variabili alla funzione in Pa e K    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% basato sul database NIST

% controllo parametri di input
if T < 280 || T > 400
    error('Temperature out of boundaries')
end
if p < 3e6 || p > 15e6
    error('pressure out of boundaries')
end
% conversione Pa in MPa
p = 1e-6 * p;
% trovare i valori di pressione di riferimento nella griglia discretizzata con 0.02 MPa
                                        % example p = 3.023 MPa
p = p*100;                              % 302.3
p_floor = floor(p)/100;                 % 3.020
if mod(p_floor,0.02) ~= 0               % mod = 0
    p_floor = p_floor - mod(p_floor,0.02);
end
p_ceil = ceil(p)/100;
if mod(p_ceil,0.02) ~= 0                  % mod = 0.01
    p_ceil = p_ceil + 0.02-mod(p_ceil,0.02);  % 3.04
end
% riportare la pessione in MPa
p = p/100;
% trovare i valori di temperatura di riferimento nella griglia discretizzata con 0.5 K
                                        % example T = 290.5832 K
T = 10*T;                               % 2905.5832
T_floor = floor(T)/10;                  % 290.5000
if mod(T_floor,0.5) ~= 0                % mod = 0
    T_floor = T_floor - mod(T_floor,0.5);
end
T_ceil = ceil(T)/10;                    % 290.6000
if mod(T_ceil,0.5) ~= 0                 % mod = 0.1
    T_ceil = T_ceil + 0.5-mod(T_ceil,0.5);  % 291.0
end
% riportare temperatura in K
T = T/10;
% X->p [Mpa] Y->T [K] reference
X = [p_floor; p_ceil];
Y = [T_floor; T_ceil];
% riporto i valori numerici di pressione e temperatura in indici nelle
% parentesi ed estraggo dalla matrice di riferimento i valori d  riferimento
p_T_floor = viscosity_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_floor-3)*50));
p_ceil_T_floor = viscosity_CO2_m(int16(1+(T_floor-280)*2),int16(1+(p_ceil-3)*50));
p_floor_T_ceil = viscosity_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_floor-3)*50));
p_T_ceil = viscosity_CO2_m(int16(1+(T_ceil-280)*2),int16(1+(p_ceil-3)*50));
```

```
% V è la matrice con i valori di output in X,Y
V = [p_T_floor, p_ceil_T_floor; p_floor_T_ceil, p_T_ceil];

viscosity = interp2(X,Y,V,p,T,'linear');
end

function viscosity = air_viscosity(T)
% questa funzione calcola la viscosità dell'aria per una pressione
% p = 1 bar ed un range di temperatura: 100 < T [K]  < 3000
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% lo script lavora in Kelvin, passare la temperatura in Kelvin %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% basato sulla correlazione di A.I. Zografos et al.

if T < 100 || T > 3000
    error('air temperature out of limits')
end
viscosity = 2.5914e-15 * T^3 - 1.4346e-11 * T^2 + 5.0523e-8 * T + 4.1130e-6;
end

function viscosity = H2O_viscosity(T,viscosity_H2O)
% questa funzione calcola la viscosità dell'acqua per una pressione
% p = 1 bar ed un range di temperatura: 5  < T [°C] < 150
%                                       278 < T [K]  < 423
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% lo script lavora in Kelvin, passare la temperatura in Kelvin %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% basato sul database NIST

if T < 5+273 || T > 150+273
    error('water temperature out of limits')
end
T = T - 273.15;
TT = 5:0.5:150;
viscosity = interp1(TT,viscosity_H2O,T);
end

function p_sat = CO2_saturation_pressure(T)
% questa funzione calcola la pressione di saturazione per la CO2 per
% temperature maggiori a quella del punto triplo T = 216.59 K ed inferiori a quella
% critica T = 304.12 K
% temperatura output in Kelvin
%
% basato sulla correlazione del paper: "A New Equation of state for Carbon Dioxide"

if T < 216.59
    error('temperature below triple point boundary')
% else
%     if T > 304.12
%         warning('temperature above critical point, pseudocritical line')
%     end
end
p_c = 7.3773e6; % [Pa]  pressione critica
T_c = 304.1282; % [K]  temperatura critica
a_1 = -7.0602087; a_2 = 1.9391218; a_3 = -1.6463597; a_4 = -3.2995634;
t_1 = 1.0; t_2 = 1.5; t_3 = 2.0; t_4 = 4.0;
p_sat = p_c * exp(T_c/T * ( a_1*(1-T/T_c)^t_1 + a_2*(1-T/T_c)^t_2 + a_3*(1-T/T_c)^t_3 + a_4*(1-T/T_c)^t_4));
p_sat = real(p_sat);
end
```