GPFA-AB_Phase1UtilizationTask4Main_Program.m

The Matlab code used for the calculation of the LCOH was designed and improved upon by Tim Reber, Lizeta Gkogka, Konstantinos Vilaetis, and Calvin Whealton from Cornell University and Dr. Zach Frone from Southern Methodist University. This MATLAB shell program contains all user-defined input values not provided by the input file. The Microsoft Excel input file is recognized by the GPFA-AB_Phase1UtilizationTask4Main_Program.m by the user entering the main program file and inserting in the name of the completed input Excel file. The user must provide the desired named for the output excel file to be created by the program.

```
%Case 2 means seperate networks for each plant
clear all
close all
clc
% loading packages for octave
pkg load io % for writing xls filesep
pkg load specfun % for lambertw

%%
tic


%% 1 Input Parameters
% -------------------

%Fixed Parameters to be chosen
LifetimePlant = 30;             % Life time plant [yrs]
WellSeparation = 500;           % Seperation of wells [m]
WellFlowRate = 30;              % Max well flow rate [kg/s]
Ramey = 1;                      % 1: Use Ramey's model for temperature drop
in production well and temperature increase in injection well;
                                % 0: Take constant temperature drop in
production well, no temperature inrease in production well
Newcosts = 1;                   % 1: take Maciek's latest drilling costs; 0:
take the 2004 drilling costs
Poweroption = 2;                % 1: 100% electricity; 2: 100% heat (3:CHP
(under construction))
FACR = 0.06;                    % Fixed annual charge rate
Printoutput = 1;                % Print extended output data
BranchDistance = 30;                % Assumed average distance of service
lines to buldings from main distribution network [m]
RoadCoverage = 0.70;            % Proportion of roads in town with main
distribution piping
%maintenance = 3;               % $/MWh used
networkservice = 7.38;          % $/m of network (incl. substation service)
%networkservice = 6.15;         % $/m of network (excl. substation service)
%substationservice = 185;       % $/substation
discount = 0.04;                % discount rate
payback = LifetimePlant;        % payback time
GFmaxDT = 65;                   % Assumed primary fluid maximum delta T
(Tprod - Tinj)
```

```
Tsr = 30;                       % Assumed secondary in temperature at design
conditions (Tsr - return temperature from radiator system)
Tss = 52.5;                     % Assumed secondary out temperature (Tss -
supply temperature to radiators)
Tpinch = 3.0;                   % Assumed HX minimum pinch temperature
(i.e.Tpr - Tsr)
dmult = 1.00;                   % Demand multiplier
GasPrice = 7.51;                % Price of natural gas [$/MCF] (2012 YTD
average for industrial consumers, from EIA)

%%DON'T FORGET TO CHANGE PIPING COSTS BACK TO BASE CASE!!!!%%
outname = 'MAIN_OUTPUT_NyPaWv.csv';        %Output file name

%% 2 ArcGIS Data
% ------------
%Read Tim's Data from ArcGIS (Geothermal Gradient, Length Piping, Average and
Extreme Temperatures
%GradientVector = textread('gradientrasteroutput.txt','%s');
%NoRows = str2double(cell2mat(GradientVector(4)));
%NoCol = str2double(cell2mat(GradientVector(2)));
%Results = zeros(1,NoRows*NoCol);



% setting working directory for the file
cd 'C:/Users/caw324/Desktop/FinalVersion - TJR'
%[INPUTnum,INPUTtxt,INPUTraw] = xlsread('INPUT_TABLE_STATIC_NY_PA_WV.xlsx');
%Read .xlsx - only using Matlab
[INPUTraw] = csvread('INPUT_TABLE_STATIC_NY_PA_WV.csv');
%Read .csv - for use with GNU Octave

NoCases = length(INPUTraw(2:end,1)); % number of cases to run

%%%%%%%%%%%%%%%%%%%%
% MATLAB
%Place_GeoID = cell2mat(INPUTraw(2:end,3));    % Numeric "place" id
%TaveAnnual = cell2mat(INPUTraw(2:end,5));     % Average Annual Temp (degC)
%TminJan = cell2mat(INPUTraw(2:end,6));        % Average Min January Temp
(degC)
%TminAbs = cell2mat(INPUTraw(2:end,7));        % Abs min January Temp (degC)
%vectorRes = cell2mat(INPUTraw(2:end,8));      % Residential sq ft
%vectorAFS = cell2mat(INPUTraw(2:end,9));      % accomodation and food
services sq ft
%vectorASW = cell2mat(INPUTraw(2:end,10));     % administrative and support
waste management sq ft
%vectorAER = cell2mat(INPUTraw(2:end,11));     % arts enertainment and
recreation sq ft
%vectorEdu = cell2mat(INPUTraw(2:end,12));     % Educational Building sq ft
%vectorHCSA = cell2mat(INPUTraw(2:end,13));    % health care and social
assistance sq ft
%vectorInf = cell2mat(INPUTraw(2:end,14));     % information geographic area
sq ft
%vectorMfg = cell2mat(INPUTraw(2:end,15));     % manufacturing sq ft
%vectorPST = cell2mat(INPUTraw(2:end,16));     % professional scientific and
technical sq ft
%vectorRERL = cell2mat(INPUTraw(2:end,17));    % real estate and rental
leasing sq ft
```

```
%vectorRet = cell2mat(INPUTraw(2:end,18));     % retail sq ft
%vectorWhsl = cell2mat(INPUTraw(2:end,19));    % wholesale sq ft
%vectorOth = cell2mat(INPUTraw(2:end,20));     % other sq ft
%P_elec = cell2mat(INPUTraw(2:end,21));        % electricity cost cents/kW-h
%L_road = cell2mat(INPUTraw(2:end,22));        % length of network (??)
%GradientVector = cell2mat(INPUTraw(2:end,23));  % geothermal gradient
(degC/km)
%ResBldgVector = cell2mat(INPUTraw(2:end,26));   % # of Residential Buildings
?
%ResUnitVector = cell2mat(INPUTraw(2:end,27));   % # of Residential Units ?
%ComBldgVector = cell2mat(INPUTraw(2:end,28));   % # of Commercial Buildings
?
%ComUnitVector = cell2mat(INPUTraw(2:end,29));   % # of Commercial Units ?
%DetachedVector = cell2mat(INPUTraw(2:end,30));  % # of detached housing
single family
%AttachedVector = cell2mat(INPUTraw(2:end,31));  % # of attached housing
single family
%Vector2_4 = cell2mat(INPUTraw(2:end,32));       % # of 2-4 unit buildings
%Vector5_19 = cell2mat(INPUTraw(2:end,33));      % # of 5-19 unit buildings
%Vector20_49 = cell2mat(INPUTraw(2:end,34));     % # of 20-49 unit buildings
%Vector50_plus = cell2mat(INPUTraw(2:end,35));   % # number of 50+ unit
buildings
%SqFtUnitVector = cell2mat(INPUTraw(2:end,36));  % # sq ft
%TaveJan = cell2mat(INPUTraw(2:end,37));         % January average
temperature

%%%%%%%%%%%%%%%%%%%%%%%
% OCTAVE
Place_GeoID =INPUTraw(2:end,3);    % Numeric "place" id
TaveAnnual = INPUTraw(2:end,5);     % Average Annual Temp (degC)
TminJan = INPUTraw(2:end,6);        % Average Min January Temp (degC)
TminAbs = INPUTraw(2:end,7);        % Abs min January Temp (degC)
vectorRes = INPUTraw(2:end,8);      % Residential sq ft
vectorAFS = INPUTraw(2:end,9);      % accomodation and food services sq ft
vectorASW = INPUTraw(2:end,10);     % administrative and support waste
management sq ft
vectorAER = INPUTraw(2:end,11);     % arts enertainment and recreation sq ft
vectorEdu = INPUTraw(2:end,12);     % Educational Building sq ft
vectorHCSA = INPUTraw(2:end,13);    % health care and social assistance sq ft
vectorInf = INPUTraw(2:end,14);     % information geographic area sq ft
vectorMfg = INPUTraw(2:end,15);     % manufacturing sq ft
vectorPST = INPUTraw(2:end,16);     % professional scientific and technical
sq ft
vectorRERL = INPUTraw(2:end,17);    % real estate and rental leasing sq ft
vectorRet = INPUTraw(2:end,18);     % retail sq ft
vectorWhsl = INPUTraw(2:end,19);    % wholesale sq ft
vectorOth = INPUTraw(2:end,20);     % other sq ft
P_elec = INPUTraw(2:end,21);        % electricity cost cents/kW-h
L_road = INPUTraw(2:end,22);        % length of network (??)
GradientVector = INPUTraw(2:end,23);  % geothermal gradient (degC/km)
ResBldgVector = INPUTraw(2:end,26);   % # of Residential Buildings ?
ResUnitVector = INPUTraw(2:end,27);   % # of Residential Units ?
ComBldgVector = INPUTraw(2:end,28);   % # of Commercial Buildings ?
ComUnitVector = INPUTraw(2:end,29);   % # of Commercial Units ?
DetachedVector = INPUTraw(2:end,30);  % # of detached housing single family
AttachedVector = INPUTraw(2:end,31);  % # of attached housing single family
Vector2_4 = INPUTraw(2:end,32);       % # of 2-4 unit buildings
```

```
Vector5_19 = INPUTraw(2:end,33);       % # of 5-19 unit buildings
Vector20_49 = INPUTraw(2:end,34);      % # of 20-49 unit buildings
Vector50_plus = INPUTraw(2:end,35);    % # number of 50+ unit buildings
SqFtUnitVector = INPUTraw(2:end,36);   % # sq ft
TaveJan = INPUTraw(2:end,37);          % January average temperature


% initializing matrices to hold output data
%%%%%%%%%%%%%%%%
% MATLAB
%COHMATRIX = cell(NoCases,7);
%MAXPOWERMATRIX = cell(NoCases,7);
%CAPACITYFACTORMATRIX = cell(NoCases,7);
%TOTALNUMBERPLANTSMATRIX = cell(NoCases,7);
%DRILLINGCOSTMATRIX = cell(NoCases,7);
%DISTRICTHEATINGCOSTMATRIX = cell(NoCases,7);
%NETANNUALHEATMATRIX = cell(NoCases,7);
%PRODUCTIONPROPORTION = cell(NoCases,7);
%SYSTEMSETUPMATRIX = cell(NoCases,7);
%LCHMATRIX = cell(NoCases,7);
%TARGETTEMPMATRIX = cell(NoCases,7);

%COHMATRIX(:,1) = num2cell(Place_GeoID);
%MAXPOWERMATRIX(:,1) = num2cell(Place_GeoID);
%CAPACITYFACTORMATRIX(:,1) = num2cell(Place_GeoID);
%TOTALNUMBERPLANTSMATRIX(:,1) = num2cell(Place_GeoID);
%DRILLINGCOSTMATRIX(:,1) = num2cell(Place_GeoID);
%DISTRICTHEATINGCOSTMATRIX(:,1) = num2cell(Place_GeoID);
%NETANNUALHEATMATRIX(:,1) = num2cell(Place_GeoID);
%PRODUCTIONPROPORTION(:,1) = num2cell(Place_GeoID);
%SYSTEMSETUPMATRIX(:,1) = num2cell(Place_GeoID);
%LCHMATRIX(:,1) = num2cell(Place_GeoID);
%TARGETTEMPMATRIX(:,1) = num2cell(Place_GeoID);

%%%%%%%%%%%%%%%%
% OCTAVE
% first column is the geo id
COHMATRIX = [Place_GeoID,zeros(NoCases,6)];
MAXPOWERMATRIX = [Place_GeoID,zeros(NoCases,6)];
CAPACITYFACTORMATRIX = [Place_GeoID,zeros(NoCases,6)];
TOTALNUMBERPLANTSMATRIX = [Place_GeoID,zeros(NoCases,6)];
DRILLINGCOSTMATRIX = [Place_GeoID,zeros(NoCases,6)];
DISTRICTHEATINGCOSTMATRIX = [Place_GeoID,zeros(NoCases,6)];
NETANNUALHEATMATRIX =[Place_GeoID,zeros(NoCases,6)];
PRODUCTIONPROPORTION = [Place_GeoID,zeros(NoCases,6)];
SYSTEMSETUPMATRIX = [Place_GeoID,zeros(NoCases,6)];
LCHMATRIX = [Place_GeoID,zeros(NoCases,6)];
TARGETTEMPMATRIX = [Place_GeoID,zeros(NoCases,6)];

ProdTemp = [80 100];          %Will accept six at a time

for n=1:length(ProdTemp)  %MUST ALWAYS START AT 1!  IF YOU WANT DIFFERENT
TEMPS THEN CHANGE THE ProdTemp VECTOR ABOVE!
```

```
  ProductionTemperature = ProdTemp(n);                          %Primary supply
temperature (Tps, Tpi) [degrees C]
  %InjectionTemperature = ProductionTemperature - GFDeltaT;     %Primary
return temperature (Tpr, Tpo) [degrees C]

  message = ['Calculating for Tprod = ' num2str(ProductionTemperature)];
  wb = waitbar(0,message);

  for k=1:NoCases % loop over the number of cases (sites, census places)

    waitbar(k/NoCases,wb); % updating the progress bar
    GeoID = Place_GeoID(k);
      Gradienti = GradientVector(k);
    Tave = TaveAnnual(k);
      Tmin = TminJan(k);
    TmeanJan = TaveJan(k);
      Tabsmin = TminAbs(k);
    sfEdu = vectorEdu(k);
      sfFsvc = vectorAFS(k)*0.1375;
      sfHS = vectorHCSA(k);
      sfLgn = vectorAFS(k)*0.8625;
      sfRet = vectorRet(k) + vectorWhsl(k);
      sfOff = vectorInf(k) + vectorRERL(k) + vectorPST(k) + vectorASW(k);
    sfPas = vectorAER(k);
    sfRes = vectorRes(k);
      sfSvc = vectorOth(k);
      length = L_road(k);
      elprice = P_elec(k);
      res_bldgs = ResBldgVector(k);
      res_units = ResUnitVector(k);
      com_bldgs = ComBldgVector(k);
      com_units = ComUnitVector(k);
    detached = DetachedVector(k);
    attached = AttachedVector(k);
    bldg2_4 = Vector2_4(k);
    bldg5_19 = Vector5_19(k);
    bldg20_49 = Vector20_49(k);
    bldg50_plus = Vector50_plus(k);
    SqFtUnit = SqFtUnitVector(k);

    %length = 9408.267938;              %Correlation TBD
    Tps = ProductionTemperature;        %Temperature at inlet district
heating system is assumed same as production well temperature [C]
    %Treturn = InjectionTemperature;    %Temperature at exit district
heating system is assumed same as injection well temperature [C]
    %Gradienti = 30;                    %Fixed geothermal gradient for now
[C/km]
    %elprice = 0.06;                    %Electricity price [$/kWh]

    %Ambient Temperature Inputs
    %Tave = 9;                          %Average yearly temperature
    %Tmin = -5;                         %Minimum daily temperature
    %Tabsmin = -12;                     %Minimum hourly temperature for max
load estimation

    %square feet inputs. (random at the moment)
```

```
    %sfEdu = 9000000;                    %Education square feet
    %sfFsls = 0;                        %Food sales square feet (not
currently used)
    %sfFsvc = 100;                       %Food service squre feet
    %sfHS = 100;                         %Healthcare square feet
    %sfLgn = 10;                         %Lodging square feet
    %sfRet = 100;                        %Retail square feet
    %sfOff = 1000;                       %Office square feet
    %sfPas = 10;                         %Public Assembly square feet
    %sfPOS = 0;                         %Public order and safety (police &
prisons) square feet (not currently used)
    %sfRel = 0;                         %Religious Worship square feet (not
currently used)
    %sfSvc = 10;                         %Service (?) square feet
    %sfRes = 1000;                       %Residential square feet

    %% 3 District Heating Demand and Costs (KV & LG)
    % ------------------------------------------
    %If Poweroption = 2, run Konstantinos Code to calculate estimated heating
demand for district heating system and Lizeta's code to calculate pipe size
and quantity

    [yrdmnd maxdmnd load dmnd PeakUnitDmnd] = DemandFcn(Tave, Tmin,...
        Tabsmin, sfEdu, sfFsvc, sfHS, sfLgn, sfRet, sfOff, sfPas, sfSvc,...
        sfRes, SqFtUnit, TmeanJan);

    dmnd0 = dmnd;                %True demand
    maxdmnd0 = maxdmnd;
    yrdmnd0 = yrdmnd;
    dmnd = dmnd*dmult;           %Adjusted demand
    maxdmnd = maxdmnd*dmult;
    yrdmnd = yrdmnd*dmult;

    %%%%% FLOW CALCULATIONS %%%%%

    %Design parameters for HX
    Tps0 = Tps;                                                %Design primary
supply temp [C]
    Tss0 = Tss;                                                %Design secondary
supply temp [C]
    Tsr0 = Tsr;                                                %Design secondary
return temp [C]
    Tpr0 = max([Tsr0+Tpinch Tps0-GFmaxDT]);                    %Design primary
return temp [C]
    kgsp0 = WellFlowRate;                                      %Design primary
mass flow [kg/s]
    kgss0 = (kgsp0*4200*(Tps0-Tpr0))/(4200*(Tss0-Tsr0));       %Design secondary
mass flow [kg/s]
    Ti0 = 21;                                                  %Indoor design
temperature
    U0 = 5000;                                                 %Design U-value
[W/m2*C]

    Cu = U0*(1/(kgsp0^0.7)+1/(kgss0^0.7));                     %Constant for
heat x-fer coefficient estimation, calculated from design conditions and
optimum U-value
```

```
    maxWth = (maxdmnd*1000*1055)/(86400);                    %Max (design)
community demand [Wth]
    maxWthV(1:365,1) = maxWth;                               %Max power vector
(calc tool) [W]
    LMTDhx0 = ((Tps0-Tss0)-(Tpr0-Tsr0))/log((Tps0-Tss0)/(Tpr0-Tsr0));
%Design heat exchanger LMTD [C]
    LMTDr0 = (Tss0-Tsr0)/log((Tss0-Ti0)/(Tsr0-Ti0));         %Design radiator
LMTD [C]
    dmndWthV = (dmnd'*1000*1055)/86400;                       %Demand vector
(average power each day of year) [W]
    DesignWth = max(dmndWthV);                               %Design condition
power (if sized for average peak) [W]
    DesignWthV(1:365,1) = DesignWth;                         %Design power
vector (calc tool)
    %HXArea = maxWth/(U0*LMTDhx0);                           %Req'd heat
exchnager size (if sized for extreme peak) [m2]
    HXArea = DesignWth/(U0*LMTDhx0);                         %Req'd heat
exchanger size (if sized for average peak) [m2]

    %LMTDrV = ((dmndWthV./maxWthV).^1.3)*LMTDr0;             %Daily radiator
LMTD vector at non-design loads (if designed for extreme peak)
    LMTDrV = ((dmndWthV./DesignWthV).^1.3)*LMTDr0;           %Daily radiator
LMTD vector at non-design loads (if designed for average peak)
    Tss0V(1:365,1) = Tss0;                                   %Daily secondary
supply temp vector
    TsrV = zeros(365,1);                                     %Setup secondary
return temp vector
    kgssV = zeros(365,1);                                    %Setup secondary
mass flow vector
    Ti0V(1:365,1) = Ti0;                                     %Vectorize indoor
temp
    Uvect = zeros(365,1);                                    %Setup U-vector
(may be unused?)
    kgspV = zeros(365,1);                                    %Setup primary
mass flow vector
    TprV = zeros(365,1);                                     %Setup primary
return temp vector
    Tps0V(1:365,1) = Tps0;                                   %Vectorize
primary supply temp

    TsrV = max((Ti0V-LMTDrV.*lambertw(-((Tss0-Ti0).*exp((Ti0V./LMTDrV)-...
                (Tss0V./LMTDrV))./LMTDrV)),Ti0+Tpinch);   %Secondary return
temp vector
    kgssV = dmndWthV./(4200*(Tss0V-TsrV));                   %Daily secondary
mass flow vector

    x0(1:365,1) = TsrV+Tpinch;                              %Initial guess
vector

    DemandFunction = @(x) dmndWthV.*((kgssV.*(Tss0V-TsrV)./...
            (Tps0V-x)).^-0.7+kgssV.^-0.7).*log(((Tps0V-Tss0V)./(x-TsrV)))-...
            (Tps0V-Tss0V-x+TsrV)*Cu*HXArea;

    %DemandFunction = @(x) dmndWthV.*((dmndWthV./(4200*(Tps0V-x))).^-
0.7+kgssV.^-0.7).*log(((Tps0V-Tss0V)./(x-TsrV))) - (Tps0V-Tss0V-
x+TsrV)*Cu*HXArea;
```

```
    %         for j=100
    %             %DemandFunction = @(x) dmndWthV(j)*((kgssV(j)*(Tss0V(j)-
TsrV(j))/(Tps0V(j)-x))^-0.7+kgssV(j)^-0.7)*log(((Tps0V(j)-Tss0V(j))/(x-
TsrV(j)))) - (Tps0V(j)-Tss0V(j)-x+TsrV(j))*Cu*HXArea;
    %             DemandFunction = @(x) -
(dmndWthV(j)*((dmndWthV(j)/(4200*(Tps0V(j)-x)))^-0.7+kgssV(j)^-
0.7)*log(((Tps0V(j)-Tss0V(j))/(x-TsrV(j)))) - (Tps0V(j)-Tss0V(j)-
x+TsrV(j))*Cu*HXArea);
    %             fplot(DemandFunction,[TsrV(j)-3*Tpinch TsrV(j)+10*Tpinch]);
    %             TprV(j) = real(fminbnd(DemandFunction,TsrV(j)-
Tpinch,TsrV(j)+Tpinch));
    %
    %         end

    options = optimset('Display','off');
    TprV = real(fsolve(DemandFunction,x0,options));              %Solve for
daily average primary return temp
    indices1 = find(TprV<x0);                                    %Primary return
temp cannot be lower than secondary return plus pinch temp
    indices2 = find(TprV>TsrV+25);
    TprV(indices1) = x0(indices1);
    TprV(indices2) = x0(indices2);
    TprV(TprV<(Tps0-GFmaxDT)) = (Tps0-GFmaxDT);             %Assume primary
fluid temperature drop cannot exceed certain value (GFmaxDT)
    %TprV = TprV+(Tps0V-TprV)*0.10;                           %Increase
return temp to account for temperature losses...?
    kgspV = dmndWthV./(4200*(Tps0V-TprV));                  %Calculate daily
mass flow in network

    DailyPower = kgspV.*(Tps0V-TprV)*4200;                  %Thermal power
delivered each day [W]
    DailyMWh = DailyPower.*24/1000000;                      %MWh per day
req'd
    AveragePower = sum(DailyPower)/365;                     %Average power
produced over year
    Averagekgsp = sum(kgspV)/365;                           %Average mass
flow req'd over year
    AverageTpr = Tps - AveragePower/(Averagekgsp*4200);    %Average primary
return temp over year
    DesignInjecTemp0 = Tpr0;                                 %Injection temp
at design conditions
    kgs = kgspV;

    %DesignInjecTemp0 = DesignInjecTemp;                     %Store injetion
temperature
    %DesignInjecTemp = Tpr0+0.10*(Tps0-Tpr0);               %Set injection
temp higher to account for 10% system heat losses

    %Transforming to MW and MWh for cost function
    %Qtot is yearly average heating demand in MWh
    %Qmax is maximum heating demand in MW
    %Qtot = yrdmnd*0.293/(1000*365*24)
    Qtot = yrdmnd*2.931*10^-4;
    Qmax = maxdmnd*1.221*10^-5;
```

```
    %dmnd0 = dmnd*2.931*10^-4;
    %Qmax = 80;
    %Qtot = 80*8760*0.5;

    %InvestmentCost is capital cost for district heating system
    %OperatingCost is yearly O&M costs for district heating system
    %[InvestmentCost,OperatingCost,PipeDiam,NoPipes,Cdist,PumpIC] =
distcostFcn(length,Qtot,Qmax,kgs,Tnetwork,elprice,maintenance,discount,paybac
k,MassFlowRate);

    c = clock;
    disp(['--- Date: ',date,' - Time: ',num2str(c(4)),':',num2str(c(5)),...
        ':',num2str(round(c(6))),' ---']);

    %% 4 MIT-EGS model
    % -------------
    %Prepare MIT-EGS parameters
    CapacityFactor = 0.5;           %Assume initially EGS capacity factor of
0.5
    MaxPower = 10;                  %Assume initially 10MWth EGS plant
    InvestmentCostMITEGS = 1e6;     %Initial guess for surface plant capital
cost [$]
    OperatingCostMITEGS = 0.5;%e6;    %Initial guess for surface plant O&M
cost [M$/yr]
    Depth = 5;                      %Initial drilling depth [km]. Will be
adjusted in order to get the chosen production temperature [C]
    DesignInjecTemp = DesignInjecTemp0;
    WellSeparation = 500; % Initial Well separation [m]

    %Run MIT-EGS model

    COHVector = zeros(0,1);
    MaxPowerVector = zeros(0,1);
    CapacityVector = zeros(0,1);
    DailyCapacityFactorMatrix = zeros(365,0);
    HeatMatrix = zeros(365,0);
    DrillingCostVector = zeros(0,1);
    DistrictHeatingCostVector = zeros(0,1);
    NetAnnualHeat = zeros(0,1);
    ProductionProportion = zeros(0,1);
    SystemSetupVector = zeros(0,1);
    TotalNumberPlants = 0;
    LCHVector = zeros(0,1);
    ExcessMWVector = zeros(0,1);

    %dmnd = dmnd0;

    counter = 0;
    converged = false;
    while ~converged
      makeInputFile;
      system('Console1.exe');
      [MITEGS_RESULTS] = textread('MATLABSUM.out','%s');
      MaxPower = str2double(cell2mat(MITEGS_RESULTS(4)));
      COH = str2double(cell2mat(MITEGS_RESULTS(6)));
```

```
      Tgeo1 = str2double(cell2mat(MITEGS_RESULTS(8)));
      TgeoEnd = str2double(cell2mat(MITEGS_RESULTS(10)));
      disp([' Depth =                          km    ',num2str(Depth),])
      disp([' TgeoEnd =                        ï¿½C   ',num2str(TgeoEnd),])
      disp([' Capacity Factor =
',num2str(CapacityFactor)])

      CapacityFactorOld = CapacityFactor;

      %Proportion1plant = min((MaxPower/(maxWth/1000000)),1);
      Proportion1plant = min((MaxPower*24/max(DailyMWh)),1);
      %Proportion1plant = WellFlowRate/max(kgspV);
      dmnd1plant = Proportion1plant*(DailyMWh);

      excesspeakMW = Proportion1plant*(maxWth/1000000) - MaxPower;
%Peak power left unmet (per plant service area)
      excesspeakMWh = ...
      Proportion1plant*((maxWth*24/1000000) - max(DailyMWh));   %MWh left
unmet (per plant service area)

      %MaxPower = WellFlowRate*4200*(Tnetwork-max(TprV))/1000000;

      DailyCapacityFactorVector =
min(dmnd1plant,ones(365,1)*24*MaxPower)/(MaxPower*24);
      CapacityFactor = mean(DailyCapacityFactorVector);

      %Account for temperature losses based on total piping distance
(0.25C/km - Ryan 1981) (should subtract from Tps but adding to Tpr will have
same effect...)
      DesignInjecTemp = DesignInjecTemp0 +
0.25*Proportion1plant*(length/1000)*RoadCoverage;

      InvestmentcostMITEGSOld = InvestmentCostMITEGS;

      [InvestmentCostMITEGS,OperatingCostMITEGS,...
       PipeDiam,NoPipes,Cdist,PumpIC,BldgCost,...
       SystemSetup,ShaftP,Energy,pumpingcost,PC,...
       LCOheat,LCOheatnominator,LCOheatdenominator,...
       PeakFuelCost,PeakBoilerCost,TotBldgCost1,...
       TotBldgCost2,BldgCost1,BldgCost2] = ...
                distcostFcn(Proportion1plant,Proportion1plant*length,...
                Proportion1plant*Qtot,Proportion1plant*Qmax,...
                Proportion1plant*kgs,Tps,elprice,discount,payback,...
                WellFlowRate,BranchDistance,res_bldgs,res_units,...
                com_bldgs,com_units,detached,attached,bldg2_4,...
                bldg5_19,bldg20_49,bldg50_plus,SqFtUnit,PeakUnitDmnd,...
                RoadCoverage,Tsr0,Tss0,HXArea,U0,Tpr0,networkservice,...
                GasPrice,excesspeakMWh);  % ZF changed excesspeakMW to
excesspeakMWh to fix issue in function

      %InvestmentCostMITEGS is total investment cost of all surface equipment
(M$)
      %OperatingCostMITEGS is yearly operating cost for surface equipment
(M$/yr)
```

```
      counter = counter + 1;

      if counter > 50
        SystemSetup = {'COUNTER_ERROR'};
        converged = true;
      elseif abs(Tgeo1-TgeoEnd) > 5
        WellSeparation = WellSeparation*1.2;
        converged = false;
      elseif abs(TgeoEnd-ProductionTemperature) > 0.5
        Depth = Depth + (ProductionTemperature-TgeoEnd)/Gradienti;
        converged = false;
      elseif abs(CapacityFactor - CapacityFactorOld) > 0.01
        converged = false;
      elseif abs(InvestmentCostMITEGS - InvestmentcostMITEGSOld) > 100000
        converged = false;
      else
         converged = true;
      end % end of if/else statements

    end % end of while loop

    %Simple discounted cash-flow analysis for estimation of Levelized Cost of
Heat (LCOH)

    DrillCost = str2double(cell2mat(MITEGS_RESULTS(12)));
    TotalCap = str2double(cell2mat(MITEGS_RESULTS(22)));
    TotalOp = str2double(cell2mat(MITEGS_RESULTS(32)));
    %SurfaceCost = InvestmentCostMITEGS/1000000;
    %OperatingCost = OperatingCostMITEGS;

    LCOHnominator = zeros(payback+1,1);
    LCOHnominator(1,1) = (TotalCap);
    LCOHdenominator = zeros(payback+1,1);
    for w=1:payback
      LCOHnominator(w+1,1) = TotalOp/((1+discount)^w);
      LCOHdenominator(w+1,1) = (Proportion1plant*Qtot)/((1+discount)^w);
    end

    LCOH0 = sum(LCOHnominator)/sum(LCOHdenominator);
    LCH = LCOH0*1000000/3.413;           %Convert M$/MWh to $/MMBTU

    %Store Data
    COHVector = [COHVector COH];
    MaxPowerVector = [MaxPowerVector MaxPower];
    CapacityVector = [CapacityVector CapacityFactor];
    DailyCapacityFactorMatrix = [DailyCapacityFactorMatrix
DailyCapacityFactorVector];
    TotalNumberPlants = floor(1/Proportion1plant);
    DrillingCostVector = [DrillingCostVector
str2double(cell2mat(MITEGS_RESULTS(12)))];
    DistrictHeatingCostVector = [DistrictHeatingCostVector
str2double(cell2mat(MITEGS_RESULTS(16)))];
    HeatMatrix = [HeatMatrix
DailyCapacityFactorVector.*ones(365,1)*24*MaxPower];
    NetAnnualHeat = [NetAnnualHeat (MaxPower*CapacityFactor*8760)];
```

```
    ProductionProportion = [ProductionProportion
(MaxPower*CapacityFactor*8760)/Qtot];
    SystemSetupVector = [SystemSetupVector SystemSetup];
    LCHVector = [LCHVector LCH];
    ExcessMWVector = [ExcessMWVector excesspeakMW];

    % printing output
    disp(['MITEGS COST OF HEAT OF PLANT ', num2str(TotalNumberPlants),' IS
',num2str(COH)]);
        disp(['The total number of plants is ',num2str(TotalNumberPlants)]);

    %% 5 Postprocessing
    % -----------------
    %     [a,b,c] = textread('SUM.out','%s %s %s');
    %     LCE = cell2mat(c(2));
    %     LCE = str2double(LCE);
    %     if Poweroption == 1
    %         LCEDISP = [' LCE =                        cents/kWh
',num2str(LCE)];
    %     elseif Poweroption == 2
    %         LCEDISP = [' LCH =                        $/MMBTU
',num2str(LCE)];
    %     end
    %     disp(LCEDISP);

    %Results for ArcGIS matrix/table (not active yet)
    % ResultsMatrix = zeros(NoRows,NoCol);
    % for i=1:1:length(Results)
    %     i
    %    r = floor((i-1)/NoCol)+1;
    %    c = mod(i,NoCol);
    %    if c == 0
    %        c = 700;
    %    end
    %    ResultsMatrix(r,c) = Results(i);
    % end
    %
    %
    % dlmwrite('Results.txt', ResultsMatrix, 'delimiter', '\t', 'precision',
2)

    %Store Data in Vector/Matrix

    % storing values
    %%%%%%%%%%%%%%%%%%%%%%
    % MATLAB
    %COHMATRIX(k,n+1) =  num2cell(COHVector);
    %MAXPOWERMATRIX(k,n+1) = num2cell(MaxPowerVector);
    %CAPACITYFACTORMATRIX(k,n+1) = num2cell(CapacityVector);
    %TOTALNUMBERPLANTSMATRIX(k,n+1) = num2cell(TotalNumberPlants);
    %DRILLINGCOSTMATRIX(k,n+1) = num2cell(DrillingCostVector);
    %DISTRICTHEATINGCOSTMATRIX(k,n+1) = num2cell(DistrictHeatingCostVector);
    %NETANNUALHEATMATRIX(k,n+1) = num2cell(NetAnnualHeat);
    %PRODUCTIONPROPORTION(k,n+1) = num2cell(ProductionProportion);
    %SYSTEMSETUPMATRIX(k,n+1) = (SystemSetupVector);
    %LCHMATRIX(k,n+1) = num2cell(LCHVector);
```

```
%EXCESSMWMATRIX(k,n+1) = num2cell(ExcessMWVector);
%TARGETTEMPMATRIX(k,n+1) = num2cell(ProductionTemperature);

%%%%%%%%%%%%%%%%%%%%%%%%%%
% OCTAVE
COHMATRIX(k,n+1) =  COHVector;
MAXPOWERMATRIX(k,n+1) = MaxPowerVector;
CAPACITYFACTORMATRIX(k,n+1) = CapacityVector;
TOTALNUMBERPLANTSMATRIX(k,n+1) = TotalNumberPlants;
DRILLINGCOSTMATRIX(k,n+1) = DrillingCostVector;
DISTRICTHEATINGCOSTMATRIX(k,n+1) = DistrictHeatingCostVector;
NETANNUALHEATMATRIX(k,n+1) = NetAnnualHeat;
PRODUCTIONPROPORTION(k,n+1) = ProductionProportion;
SYSTEMSETUPMATRIX(k,n+1) = -9999;
LCHMATRIX(k,n+1) = LCHVector;
EXCESSMWMATRIX(k,n+1) = ExcessMWVector;
TARGETTEMPMATRIX(k,n+1) = ProductionTemperature;

%Optimal matrix for COH from MITEGS

%%%%%%%%%%%%%%%%%%%%%%
% MATLAB
%if COHVector<cell2mat(COHMATRIX(k,n))
  %BESTMATRIXCOH(k,1) = COHMATRIX(k,1);
  %BESTMATRIXCOH(k,2) = COHMATRIX(k,n+1);
  %BESTMATRIXCOH(k,3) = MAXPOWERMATRIX(k,n+1);
  %BESTMATRIXCOH(k,4) = CAPACITYFACTORMATRIX(k,n+1);
  %BESTMATRIXCOH(k,5) = TOTALNUMBERPLANTSMATRIX(k,n+1);
  %BESTMATRIXCOH(k,6) = TARGETTEMPMATRIX(k,n+1);
%end

%%%%%%%%%%%%%%%%%%%%%%%%%%
% OCTAVE
if COHVector<COHMATRIX(k,n)
  BESTMATRIXCOH(k,1) = COHMATRIX(k,1);
  BESTMATRIXCOH(k,2) = COHMATRIX(k,n+1);
  BESTMATRIXCOH(k,3) = MAXPOWERMATRIX(k,n+1);
  BESTMATRIXCOH(k,4) = CAPACITYFACTORMATRIX(k,n+1);
  BESTMATRIXCOH(k,5) = TOTALNUMBERPLANTSMATRIX(k,n+1);
  BESTMATRIXCOH(k,6) = TARGETTEMPMATRIX(k,n+1);
end

%Optimal matrix for LCH calcualted using discounted cash-flow
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MATLAB
%if LCHVector<cell2mat(LCHMATRIX(k,n))
  %BESTMATRIXLCH(k,1) = LCHMATRIX(k,1);
  %BESTMATRIXLCH(k,2) = LCHMATRIX(k,n+1);
  %BESTMATRIXLCH(k,3) = MAXPOWERMATRIX(k,n+1);
  %BESTMATRIXLCH(k,4) = CAPACITYFACTORMATRIX(k,n+1);
  %BESTMATRIXLCH(k,5) = TOTALNUMBERPLANTSMATRIX(k,n+1);
  %BESTMATRIXLCH(k,6) = TARGETTEMPMATRIX(k,n+1);
%end

%%%%%%%%%%%%%%%%%%%%%%
% OCTAVE
```

```
    if LCHVector<LCHMATRIX(k,n)
      BESTMATRIXLCH(k,1) = LCHMATRIX(k,1);
      BESTMATRIXLCH(k,2) = LCHMATRIX(k,n+1);
      BESTMATRIXLCH(k,3) = MAXPOWERMATRIX(k,n+1);
      BESTMATRIXLCH(k,4) = CAPACITYFACTORMATRIX(k,n+1);
      BESTMATRIXLCH(k,5) = TOTALNUMBERPLANTSMATRIX(k,n+1);
      BESTMATRIXLCH(k,6) = TARGETTEMPMATRIX(k,n+1);
    end

  end % end of inner for loop

  close(wb)

  %Write results to output spreadsheet

  Titles = {'GeoID' 'LCH($/MMBTU)Calc' 'COH($/MMBTU)MITEGS' 'MaxPower(MW)'
'CapacityFactor' 'NumberPlants' 'DrillCost(M$)' 'NetworkCost(M$)'
'ProdProportion' 'NetAnnualHeat(MWh)' 'UnmetPeakMW' 'SystemSetup' ' '
'LCHPlotting:' ' ' ' ' ' ' ' ' ' 'COHPlotting'};

  %%%%%%%%%%%%%%%%%%%%%%%%
  % MATLAB
  xlswrite(outname,Titles,['Tprod ' num2str(ProductionTemperature)],'A1:S1');
  xlswrite(outname,COHMATRIX(1:end,1),['Tprod '
num2str(ProductionTemperature)],'A2');
    xlswrite(outname,COHMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'C2');
  xlswrite(outname,MAXPOWERMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'D2');
    xlswrite(outname,CAPACITYFACTORMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'E2');
    xlswrite(outname,TOTALNUMBERPLANTSMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'F2');
    xlswrite(outname,DRILLINGCOSTMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'G2');
    xlswrite(outname,DISTRICTHEATINGCOSTMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'H2');
    xlswrite(outname,PRODUCTIONPROPORTION(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'I2');
  xlswrite(outname,NETANNUALHEATMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'J2');
  xlswrite(outname,SYSTEMSETUPMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'L2');
  xlswrite(outname,LCHMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'B2');
  xlswrite(outname,EXCESSMWMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'K2');

  %%%%%%%%%%%%%%%%%%%%%%%%%%
  % OCTAVE
  xlswrite(outname,Titles,['Tprod ' num2str(ProductionTemperature)],'A1:S1');
  xlswrite(outname,COHMATRIX(1:end,1),['Tprod '
num2str(ProductionTemperature)],'A2');
    xlswrite(outname,COHMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'C2');
```

```
  xlswrite(outname,MAXPOWERMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'D2');
    xlswrite(outname,CAPACITYFACTORMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'E2');
    xlswrite(outname,TOTALNUMBERPLANTSMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'F2');
    xlswrite(outname,DRILLINGCOSTMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'G2');
    xlswrite(outname,DISTRICTHEATINGCOSTMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'H2');
    xlswrite(outname,PRODUCTIONPROPORTION(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'I2');
  xlswrite(outname,NETANNUALHEATMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'J2');
  xlswrite(outname,SYSTEMSETUPMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'L2');
  xlswrite(outname,LCHMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'B2');
  xlswrite(outname,EXCESSMWMATRIX(1:end,n+1),['Tprod '
num2str(ProductionTemperature)],'K2');

  %Plot COH results as supply curve
  CumCapac =
cell2mat(MAXPOWERMATRIX(:,n+1)).*cell2mat(TOTALNUMBERPLANTSMATRIX(:,n+1));

  LCECapac0 = real([cell2mat(COHMATRIX(:,n+1)) CumCapac]);

  LCECapac1 = sortrows(LCECapac0,1);

  LCECapac2 = [LCECapac1(:,1) cumsum(LCECapac1(:,2))];

  [rows cols] = size(LCECapac2);

  LCECapac3 = zeros(rows+1,2);
  LCECapac3(1:rows,1) = LCECapac2(:,1);
  LCECapac3(rows+1,1) = LCECapac2(rows,1);
  for i=1:(rows);
    LCECapac3(i+1,2) = LCECapac2(i,2);
    LCECapac3(i,3) = LCECapac3(i+1,2)-LCECapac3(i,2);
    LCECapac3(i+1,4) = LCECapac3(i+1,1)-LCECapac3(i,1);
  end

  LCECapac3(1,2) = 0;
  LCECapac3(rows+1,3) = 0;
  LCECapac3(1,4) = 0;

  %     stairs(LCECapac3(:,2),LCECapac3(:,1));
  %     hold all;
  %     xlabel('Cumulative Capacity (MW)');
  %     ylabel('LCH ($/MMBTU)');
  %     title('Estimated Supply Curve');
  %     axis([0 max(LCECapac3(:,2)) 0 100])

  xlswrite(outname,{'COH' 'CumCapac' 'x-step' 'y-step'},['Tprod '
num2str(ProductionTemperature)],'S2:V2');
```

```
  xlswrite(outname,LCECapac3,['Tprod ' num2str(ProductionTemperature)],'S3');

  %Plot LCH results as supply curve

  CumCapac =
cell2mat(MAXPOWERMATRIX(:,n+1)).*cell2mat(TOTALNUMBERPLANTSMATRIX(:,n+1));

  LCECapac0 = real([cell2mat(LCHMATRIX(:,n+1)) CumCapac]);

  LCECapac1 = sortrows(LCECapac0,1);

  LCECapac2 = [LCECapac1(:,1) cumsum(LCECapac1(:,2))];

  [rows cols] = size(LCECapac2);

  LCECapac3 = zeros(rows+1,2);
  LCECapac3(1:rows,1) = LCECapac2(:,1);
  LCECapac3(rows+1,1) = LCECapac2(rows,1);
  for i=1:(rows);
    LCECapac3(i+1,2) = LCECapac2(i,2);
    LCECapac3(i,3) = LCECapac3(i+1,2)-LCECapac3(i,2);
    LCECapac3(i+1,4) = LCECapac3(i+1,1)-LCECapac3(i,1);
  end

  LCECapac3(1,2) = 0;
  LCECapac3(rows+1,3) = 0;
  LCECapac3(1,4) = 0;

  %     stairs(LCECapac3(:,2),LCECapac3(:,1));
  %     hold all;
  %     xlabel('Cumulative Capacity (MW)');
  %     ylabel('LCH ($/MMBTU)');
  %     title('Estimated Supply Curve');
  %     axis([0 max(LCECapac3(:,2)) 0 100])

  xlswrite(outname,{'LCH' 'CumCapac' 'x-step' 'y-step'},['Tprod '
num2str(ProductionTemperature)],'N2:Q2');
  xlswrite(outname,LCECapac3,['Tprod ' num2str(ProductionTemperature)],'N3');

end % end outer for loop

% figure(1)
% legend(['Tprod ' num2str(ProdTemp(1)) ' Celsius'],['Tprod '
num2str(ProdTemp(2)) ' Celsius'],['Tprod ' num2str(ProdTemp(3)) '
Celsius'],['Tprod ' num2str(ProdTemp(4)) ' Celsius'],['Tprod '
num2str(ProdTemp(5)) ' Celsius'],['Tprod ' num2str(ProdTemp(6)) '
Celsius'],'Location','NorthWest')


%Prepare supply curve for best matrix for cash-flow LCH
CumCapac = cell2mat(BESTMATRIXLCH(:,3)).*cell2mat(BESTMATRIXLCH(:,5));

LCECapac0 = real([cell2mat(BESTMATRIXLCH(:,2)) CumCapac]);
```

```
LCECapac1 = sortrows(LCECapac0,1);

LCECapac2 = [LCECapac1(:,1) cumsum(LCECapac1(:,2))];

[rows cols] = size(LCECapac2);

LCECapac3 = zeros(rows+1,2);
LCECapac3(1:rows,1) = LCECapac2(:,1);
LCECapac3(rows+1,1) = LCECapac2(rows,1);
for i=1:(rows);
  LCECapac3(i+1,2) = LCECapac2(i,2);
  LCECapac3(i,3) = LCECapac3(i+1,2)-LCECapac3(i,2);
  LCECapac3(i+1,4) = LCECapac3(i+1,1)-LCECapac3(i,1);
end

LCECapac3(1,2) = 0;
LCECapac3(rows+1,3) = 0;
LCECapac3(1,4) = 0;

OptimalTitles = {'LCH calc using discounted cash-flow' ' ' ' ' ' ' ' ' ' ' ' ';
'GeoID' 'LCH($/MMBTU)' 'MaxPower(MW)' 'CapacityFactor' 'NumberPlants'
'OptimalTemp'};

xlswrite(outname,OptimalTitles,'OptimalTemp','A1:F2');
xlswrite(outname,BESTMATRIXLCH,'OptimalTemp','A3');
xlswrite(outname,{'LCH' 'CumCapac' 'x-step' 'y-step'},'OptimalTemp','H2:K2');
xlswrite(outname,LCECapac3,'OptimalTemp','H3');

%Prepare supply curve for best matrix for MITEGS COH

CumCapac = cell2mat(BESTMATRIXCOH(:,3)).*cell2mat(BESTMATRIXCOH(:,5));

LCECapac0 = real([cell2mat(BESTMATRIXCOH(:,2)) CumCapac]);

LCECapac1 = sortrows(LCECapac0,1);

LCECapac2 = [LCECapac1(:,1) cumsum(LCECapac1(:,2))];

[rows cols] = size(LCECapac2);

LCECapac3 = zeros(rows+1,2);
LCECapac3(1:rows,1) = LCECapac2(:,1);
LCECapac3(rows+1,1) = LCECapac2(rows,1);
for i=1:(rows);
  LCECapac3(i+1,2) = LCECapac2(i,2);
  LCECapac3(i,3) = LCECapac3(i+1,2)-LCECapac3(i,2);
  LCECapac3(i+1,4) = LCECapac3(i+1,1)-LCECapac3(i,1);
end

LCECapac3(1,2) = 0;
LCECapac3(rows+1,3) = 0;
LCECapac3(1,4) = 0;
```

```
OptimalTitles = {'GeoID' 'COH($/MMBTU)' 'MaxPower(MW)' 'CapacityFactor'
'NumberPlants' 'OptimalTemp'};

xlswrite(outname,OptimalTitles,'OptimalTemp','R2:W2');
xlswrite(outname,BESTMATRIXCOH,'OptimalTemp','R3');
xlswrite(outname,{'COH calc from MITEGS using fixed charge rate method' ' ' '
' ' '; 'COH' 'CumCapac' 'x-step' 'y-step'},'OptimalTemp','M1:P2');
xlswrite(outname,LCECapac3,'OptimalTemp','M3');

%Print parameters

ParameterMatrix(1,:) = {'Date', datestr(floor(now)), datestr(rem(now,1))};
ParameterMatrix(2,:) = {'LifetimePlant' num2str(LifetimePlant) 'yrs'};
ParameterMatrix(3,:) = {'WellSeparation', num2str(WellSeparation), 'm'};
ParameterMatrix(4,:) = {'MassFlowRate', num2str(WellFlowRate), 'kgs'};
ParameterMatrix(5,:) = {'FixedAnnualCharge', num2str(FACR), ' '};
ParameterMatrix(6,:) = {'BranchDistance', num2str(BranchDistance), 'm'};
ParameterMatrix(7,:) = {'RoadCoverage', num2str(RoadCoverage), 'percent'};
ParameterMatrix(8,:) = {'NetworkService', num2str(networkservice),
'$/meter'};
ParameterMatrix(9,:) = {'DiscountRate', num2str(discount), ' '};
ParameterMatrix(10,:) = {'RadiatorSupply', num2str(Tss0),
'degreesC(design)'};
ParameterMatrix(11,:) = {'RadiatorReturn', num2str(Tsr0),
'degreesC(design)'};
ParameterMatrix(12,:) = {'HXPinchTemp', num2str(Tpinch), 'degreesC'};
ParameterMatrix(13,:) = {'MaxDeltaT', num2str(GFmaxDT), 'degreesC'};
ParameterMatrix(14,:) = {'DmndMult', num2str(dmult) '(portion)'};

xlswrite(outname,ParameterMatrix,'Parameters');

toc
```