```
%function [ammonia_kmol] = RunModel(power_required,GasInfo)
%RunModel.m

% Find function for change in efficiency based on ammonia flow rate
% Find out what's wrong with ammonia combustion temperature
% Find a representation for NOx emissions




% Range of flow allowed through the compressor/turbine is



% LOAD DATA
%----------------------------------------------------------------------------
% Initialise a matrix to carry state information
% The matrix is structured with each row representing a different
 stage
% [StageNumber FlowRate StagePressure StageTemp Enthalpy]
% All pressures should be in bar, all temperature should be in K,
 specific
% volume should be in m3/kg, enthalpy is kJ/kg, entropy is kJ/kgK
% Missing values should be flagged with -999
State = zeros(5,6);

% Import gas table structures
load('NH3.mat');
load('O2.mat');
load('N2.mat');
load('H2O.mat');
load('NO.mat');
load('H2.mat');

% Import dissociation/flame temp curve
%load('Dissociation.mat');

Data.NH3 = NH3;
Data.O2 = O2;
Data.N2 = N2;
Data.H2O = H2O;
Data.NO = NO;
Data.H2 = H2;
%Data.Dissociation = Dissociation;

% Physical constants
% Define some physical constants which we have assumed to be constant.
% These should be replaced later with more accurate representations
 that
% vary with temperature, etc
Constant = struct;          % Define a structure for constants
Constant.gamma = 1.4;       % Specific heat ratio
```

```
Constant.cp = 1.005;        % Specific heat at constant pressure, kJ/
kgK

% Define some parameters of the gas turbine powerplant in a structure
Parameter.na = 1;           % 1kmol/s ammonia fuel flow

% Define atmospheric conditions
p_atm = 1.000;                   % Atmospheric pressure in bar
t_atm = 293.13;                  % Atmospheric pressure in Kelvin



ER_min = 0.765;
ER_max = 1.0125;



% INITIAL CONDITIONS
%---------------------------------------------------------------------------
% Initialise stage 1 variables
n1 = 4; % 1 kmol/s
p1 = p_atm;
t1 = t_atm;
v1 = 8315*t1*n1/p1/10^5; %m3
h1 = n1*findProperty(Data.O2,t1,'Dh');

% Add to the State array
State(1,:) = [1 n1 p1 t1 h1 v1];



% INLET
%---------------------------------------------------------------------------
State(2,:) = State(1,:);
State(2,1) = 2;



% COMPRESSION (2-3)
%---------------------------------------------------------------------------
% Run the compressor
State = runCompressor(State,Parameter,Constant,Data);




% COMBUSTION (3-4)
%---------------------------------------------------------------------------
% Run the burner
[State,Parameter] = runBurner(State,Parameter,Constant,Data);
```

```
% EXPANSION (4-5)
%--------------------------------------------------------------------------
% Run the turbine
State = runTurbine(State,Parameter,Constant,Data);
```

```
% POST-PROCESSING
%--------------------------------------------------------------------------
% Something about checking NOx emissions and doing some cogen
% Check NOx emissions

% Run the heat exchanger to partially dissociate the ammonia
% No state output - output probably not that important?
runHeatExchanger(State,Parameter,Constant,Data);
```

```
% POWER GENERATION
%--------------------------------------------------------------------------
% Run the generator
%runGenerator();
```

```
% RESULTS CALCULATIONS
%--------------------------------------------------------------------------
% Run the generator
%runGenerator();
w_t = State(4,5) - State(5,5);
w_c = State(2,5) - State(3,5);
w_out = w_t + w_c
```

```
Compressor successful  Calculated and tabulated enthalpy margin is
 -5.361650e+03 kJ  Compressor work done -6.488723e+04 kJ
Burner successful  T4 temperature 2.006921e+03 K  Equivalence ratio
 0.9
Turbine successful  Calculated and tabulated enthalpy margin is
 -1.035590e+04 kJ  Turbine work done 1.885795e+05
Heat exchanger converges  Temperature difference at burner inlet 443
 K
w_out =

   1.2369e+05
```

Published with MATLAB® R2016b