Vm433: Advanced Energy Solutions (Fall 2016)
Project 2: Gas-Turbine Engine

Submit via Canvas → Assignments
Part *A*, to be completed **individually**, is due 4 PM, Thur Oct 27
Part *B*, to be completed **individually**, is due 4 PM, Thur Nov 17

(v. 10/30/2016) *This assignment is subject to refinements. You may ask the TA and me clarifying questions on the Canvas discussion forum. All questions and answers will be available for everyone to review. Please do so before firing off your questions, to make sure the questions have not yet been posed and answered. We will stop responding to questions one day before each due date.* **Start working on the project early!**

In the previous project, we dealt with water, which is a common yet useful simple compressible substance. It serves as the working fluid in a steam power (Rankine) cycle. In the current project, we continue the review and exploration of (relatively) basic thermodynamics from Vm235, this time examining a gas-turbine power (Brayton) cycle. The primary working fluid in this case is **air**, which is also a common and useful simple compressible substance, and **an ideal gas** to boot.

For this project, you will build thermodynamic models of a simple-cycle gas-turbine engine, with intercooling. The project will help you

1) review basic thermodynamics from Vm235,
2) gain experience analyzing real engine components (with variable-specific-heat working fluid and non-ideal device performance metrics or figures of merit),
3) gain experience analyzing reacting (combustion) system thermodynamics, by deploying concepts of ideal-gas mixtures, combustion stoichiometry (atom balance), energy conservation (energy balance), and adiabatic flame temperature

THE SYSTEM

The basic design of the General Electric LMS100 intercooled gas turbine is well-publicized, as is its simple-cycle thermal efficiency, which exceeds 46% [1], [2]. Even if the actual device specifications are not known exactly, they can be deduced based on public data of other advanced aeroderivative and frame (land-based industrial) gas-turbine engines, as well as informed state-of-technology guesses and estimates.

The Pio Pico Energy Center (PPEC), currently under construction in the state of California in the United States, will install three of these LMS100 engines, each producing $\sim 100\,\mathrm{MW}$ of dispatchable electricity to support peaking requirements and supplement intermittent renewable (wind and solar) electricity supply to the grid [3]. The description and heat balance diagrams of PPEC, taken from its Application for Certification (Docket no. 2011-AFC-1C) by the California Energy Commission, are available on Canvas.

THE ASSIGNMENT

A. To be completed individually, due 4 PM, Thur Oct 27

1) **10 points.** (a) Develop separate MATLAB functions to determine the changes in (i) specific enthalpy and (ii) specific entropy for an ideal gas, given the gas initial state 1 (T_1, P_1), final state 2 (T_2, P_2), and the specific heat of the gas as a third-degree polynomial of temperature (textbook appendix Table A–2c).

   (b) *The inverse problem.* Develop a related set of MATLAB functions to determine the final temperature T_2 for an ideal gas, given the gas initial state 1 (T_1, P_1), final pressure P_2, the specific heat of the gas as a third-degree polynomial of temperature, and either (i) the change in specific enthalpy, $h_2 - h_1$, for the gas, or (ii) its change in specific entropy, $s_2 - s_1$.

   Notes: (i) Your MATLAB functions should be generally applicable to any one of the common gases listed in Table A–2c, by a simple change of the polynomial coefficients a, b, c, and d.
   (ii) Consider using the MATLAB functions `polyval` and `roots`.

   Deliverables: The following MATLAB functions (as four separate m-files)

- ```
  function [ dhkg, dhkmol ] = findEnthalpyDiff( ...
    cpCoeff, molarMass, T1, T2 )
  % FINDENTHALPYDIFF Find specific enthalpy difference for an ideal
  % gas between two temperatures
  % Inputs:
  %   cpCoeff = constant-pressure specific heat coefficients
  %   molarMass = molar mass of ideal gas species (kg/kmol)
  %   T1 = initial temperature (K)
  %   T2 = final temperature (K)
  % Outputs:
  %   dhkg = mass-specific enthalpy difference (kJ/kg)
  %   dhkmol = mole-specific enthalpy difference (kJ/kmol)
  ```
- ```
  function [ dskg, dskmol ] = findEntropyDiff( ...
    cpCoeff, molarMass, T1, P1, T2, P2 )
  % FINDENTROPYDIFF Find specific entropy difference for an ideal
  % gas between two states
  % Inputs:
  %   ... [see above] ...
  %   P1 = initial pressure (kPa)
  %   P2 = final pressure (kPa)
  % Outputs:
  %   dskg = mass-specific entropy difference (kJ/kg)
  %   dskmol = mole-specific entropy difference (kJ/kmol)
  ```
- ```
  function T2 = findFinalTempFromEnthalpyDiff( ...
    cpCoeff, molarMass, T1, dhkg )
  % FINDFINALTEMPFROMENTHALPYDIFF Find final temperature for an
  % ideal gas given initial temperature and enthalpy difference
  % Inputs/Output:
  %   ... [see above] ...
  ```
  ```

- ```
  function T2 = findFinalTempFromEntropyDiff( ...
    cpCoeff, molarMass, T1, P1, P2, dskg )
  % FINDFINALTEMPFROMENTROPYDIFF Find final temperature for an
  % ideal gas given initial state, final pressure, and entropy
  % difference
  % Inputs/Output:
  %   ... [see above] ...
  ```

2) **20 points.** (a) Write a MATLAB script that incorporates the functions from part (1) above to analyze the LMS100 engine based on its basic cycle parameters given below. In other words, develop a **variable-specific-heat working fluid/ideal-process** (i.e., **air-standard**) model for the engine — the working fluid is air, and the ideal processes are internally reversible (ideal intercooling returns the air that leaves the low-pressure compressor to ambient temperature).

Low-pressure compressor	pressure ratio	4
Overall cycle	pressure ratio	42
Turbine inlet	temperature	1380°C

Use this air-standard model to determine the following performance metrics for the engine: **thermal efficiency**, **heat rate** (in $\mathrm{kJ/kWh}$), **specific work** ($\mathrm{kW/(kg/s)}$), **turbine exhaust temperature** ($^\circ\mathrm{C}$).

The gas-turbine engine performance varies depending on the ambient conditions, which define the thermodynamic state of air entering the engine compressor. In this project, consider the ISO standard conditions of $15\,^\circ\mathrm{C}$, $1.013\,\mathrm{bar}$, which are commonly used in the gas turbine industry. (The ISO standard also stipulates 60% relative humidity, but you need not concern yourself with that for now.)

(b) Plot P-v, T-s, and h-s diagrams for the air-standard LMS100 engine cycle. Connect the state points with solid curves that accurately reflect each process. (Solid curves imply the exact reversible paths are known.) Label each state following General Electric's station schematic (see page 4 of Ref. [2]).

3) **10 points.** Repeat part (2)—determining the thermal efficiency, heat rate, specific work, turbine exhaust temperature, and plotting the P-v, T-s, and h-s diagrams—using a **constant-specific-heat working fluid**/ideal-process (i.e., **cold** air-standard) model based on the same cycle parameters. Consider air with constant specific heat values such that their ratio is $k = 1.4$, and compare the results to those from part (2).

Deliverables (including Part 2 above): (a) A MATLAB script named `airStandardLMS100.m`, which calls the functions developed in Part 1 above to determine the LMS100 engine performance metrics, as well as calls the basic MATLAB plotting functions to generate the required diagrams

- Plot each of the P-v, T-s, and h-s diagrams for both Parts (2b) and (3) on the same set of axes to facilitate comparison between the air standard and cold-air standard results
- Do not neglect proper annotation of the figures using functions such as `title`; `xlabel`, `ylabel` (including units in parentheses); `legend` or `gtext`

(b) A table typeset in Microsoft Word that lists the LMS100 engine performance metrics determined following the air standard versus the cold-air standard assumptions

- Name the Word file `LMS100EnginePerformanceMetrics.docx`
- Do not neglect proper formatting of the table — include a title (or caption), and column headings (with units in parentheses, if needed)
  ```

B. To be completed individually, due 4 PM, Thur Nov 17

4) **20 points.** (a) Develop separate MATLAB functions to determine the work input (output) and exit temperature of an ideal gas upon compression (expansion), given the gas conditions (T, P) at the compressor (turbine) inlet, the gas pressure at the compressor (turbine) exit, and the polytropic ("small-stage") efficiency of the turbomachinery.

Deliverables: The following MATLAB functions (as separate m-files)

- ```
  function [ compressWork, exitT ] = gasCompressor( ...
    cpCoeff, molarMass, inT, inP, exitP, polyEff )
  % GASCOMPRESSOR Find ideal gas compressor specific work input
  % and exit temperature
  % Inputs:
  %   ... [see above] ...
  %   inT = compressor inlet gas temperature (K)
  %   inP = compressor inlet gas pressure (kPa)
  %   exitP = compressor exit gas pressure (kPa)
  %   polyEff = compressor polytropic efficiency (percent)
  % Output:
  %   compressWork = mass-specific compressor work input (kJ/kg)
  %   exitT = compressor exit gas temperature (K)
  ```
- ```
  function [ expandWork, exitT ] = gasTurbine( ...
    cpCoeff, molarMass, inT, inP, exitP, polyEff )
  % Function description similar to above, but replace
  % ''compressor'' by ''turbine'', and ''compress'' by ''expand''.
  ```

(b) Repeat part (2)—determining the thermal efficiency, heat rate, specific work, turbine exhaust temperature, and plotting the P-v, T-s, and h-s diagrams—using a variable-specific-heat working fluid/**real-turbomachinery model** this time, based on the same cycle parameters as Part (2) and the performance metrics provided below. (The model is still non-reactive; the working fluid remains air.) Hint: Consider how you might use the results from Part (2) to debug the analysis code here.

LMS100 engine component	Figure of merit	Value
Low-pressure compressor, LPC High-pressure compressor, HPC	Polytropic efficiency	0.90
High-pressure turbine, HPT Intermediate-pressure turbine, IPT Power turbine, PT	Polytropic efficiency	0.90
Intercooler	Pinch temperature diff. Pressure loss ratio	20 K 0.94
Burner	Pressure loss ratio	0.94

Deliverables: (a) A MATLAB script named `realTurbomachineryLMS100.m`, which calls the functions developed in Parts 1 and 4(a) above to determine the LMS100 engine performance metrics, as well as calls the basic MATLAB plotting functions to generate the required diagrams

- Plot each of the P-v, T-s, and h-s diagrams for both Parts 2(b) and 4(b) on the same set of axes to facilitate comparison between the ideal-process (air standard) and real-turbomachinery results. In the latter case, connect the state points on the diagrams with straight, dashed lines (which imply the exact irreversible process paths are not known).

- Do not neglect proper annotation of the figures using functions such as `title`; `xlabel`, `ylabel` (including units in parentheses); `legend` or `gtext`

(b) Append the real-turbomachinery results to the table typeset in the Microsoft Word file that has been submitted in Part A of the project (`LMS100EnginePerformanceMetrics.docx`)

5) **15 points.** (a) Write two MATLAB functions that incorporate (i.e., call) the functions developed in part (1) (`findEnthalpyDiff`, `findEntropyDiff`) to determine the (i) specific enthalpy ($\mathrm{kJ/kg}$) and (ii) specific entropy ($\mathrm{kJ/kg \cdot K}$) for an ideal gas **mixture**, *relative to the standard reference state of* $25\,^{\circ}\mathrm{C}$, $1\,\mathrm{atm}$ (textbook appendix Table A–26), given the mixture temperature, pressure, and chemical composition ($\mathrm{mol\%}$).

Deliverables: The following MATLAB functions (as separate m-files)

- ```
  function [ mixh ] = findMixtureEnthalpy( ...
    mixT, mixP, mix, mixMoleFrac )
  % FINDMIXTUREENTHALPY Find specific enthalpy for an ideal gas mixture
  % Inputs:
  %   mixT = mixture temperature (K)
  %   mixP = mixture pressure (kPa)
  %   mix = cell array of chemical symbols of mixture species
  %     (example: mix = {'O2'; 'N2'; 'Ar'; 'CO2'; 'H2O'})
  %   mixMoleFrac = corresponding mixture species mole fractions (%)
  % Output:
  %   mixh = mass-specific mixture enthalpy (kJ/kg)
  ```
- ```
  function [ mixs ] = findMixtureEntropy( ...
    mixT, mixP, mix, mixMoleFrac )
  % FINDMIXTUREENTROPY Find specific entropy for an ideal gas mixture
  % Inputs:
  %   ... [see above] ...
  % Output:
  %   mixs = mass-specific mixture entropy (kJ/kg-K)
  ```

(b) Develop a MATLAB function that determines, for a hydrocarbon fuel C_nH_m, (i) the stoichiometric amount of air for its complete combustion (all carbon oxidized to CO_2, all hydrogen to H_2O), and (ii) the corresponding chemical composition of the combustion product gas mixture, given the air is composed of oxygen O_2, nitrogen N_2, argon Ar, carbon dioxide CO_2, and water H_2O. (Argon is a monatomic gas; at low volume fractions, water can also be taken as an ideal gas.)

Deliverables: The following MATLAB function

- ```
  function [ airStoich, prod, prodStoich ] = ...
    findCnHmStoichiometricCombustion( n, m, air, airMoleFrac )
  % Inputs:
  %   n, m = numbers of fuel carbon and hydrogen atoms
  %   air = cell array of chemical symbols of air mixture species
  %     = {'O2'; 'N2'; 'Ar'; 'CO2'; 'H2O'}
  %   airMoleFrac = corresp. air mixture species mole fractions (%)
  % Outputs:
  %   airStoich = stoichiometric amount of air for the complete
  %     combustion of fuel (in kmol per kmol of fuel)
  %   prod = cell array of chemical symbols of complete-combustion
  ```
  ```

```
%     product species = {'O2'; 'N2'; 'Ar'; 'CO2'; 'H2O'}
%   prodStoich = corresp. stoichiometric amount of complete-
%     combustion product species (in kmol per kmol of fuel)
```

(c) Use these functions to construct the h-T and h-s diagrams for (i) a stoichiometric mixture of propane C_3H_8 and air, and (ii) the corresponding mixture of complete-combustion products.

Deliverables: A MATLAB script named `propaneCombustionPropertyDiagrams.m`, which calls the functions above as well as basic MATLAB plotting functions to generate the required diagrams

- Plot each of the h-T and h-s diagrams for both mixtures on the same set of axes to facilitate comparison.
- Do not neglect proper annotation of the figures using functions such as `title`; `xlabel`, `ylabel` (including units in parentheses); `legend` or `gtext`
- Consider air as composed of the following

Air component	mol%
Oxygen	20.74
Nitrogen	77.30
Argon	0.92
Carbon dioxide	0.03
Water	1.01

Entropy of argon s°_{Ar} at standard reference state is available at http://kinetics.nist.gov/janaf/.

6) **20 points.** (a) Write a MATLAB function, modeled after `findCnHmStoichiometricCombustion.m`, but includes the air-fuel ratio AF as another input parameter, to determine for a hydrocarbon fuel C_nH_m (i) the actual amount of air for its combustion under lean condition, and (ii) the corresponding chemical composition of the combustion product gas mixture (including excess oxygen O_2, nitrogen N_2, etc.)

Deliverables: None required. This part is ungraded, but you would need this MATLAB function in the following sections in any case.

(b) Develop a MATLAB function to determine the adiabatic flame temperature due to the complete combustion of a natural gas fuel in a steady-flow burner, given temperatures of the fuel and air, and the air-fuel ratio AF. Consider the fuel as composed of methane CH_4, ethane C_2H_6, propane C_3H_8, as well as small quantities of oxygen O_2, inert nitrogen N_2 and carbon dioxide CO_2.

(c) *The inverse problem.* Develop a related MATLAB function to determine the air-fuel ratio AF needed for complete combustion of a natural gas fuel in a steady-flow burner to reach some specified adiabatic flame temperature.

Deliverables: The following MATLAB functions (as separate m-files)

- ```
  function AFT = findAdiabaticFlameTemp( ...
    fuel, fuelMoleFrac, fuelT, air, airMoleFrac, airT, AF )
  % FINDADIABATICFLAMETEMP Find adiabatic flame temperature
  % for complete combustion of fuel in air
  % Inputs:
  ```
  ```

```
%   fuel/air = cell array of chemical symbols of fuel/air
%     mixture species
%   fuel/airMoleFrac = corresponding fuel/air mixture
%     species mole fractions (%)
%   fuel/airT = fuel/air mixture temperature (K)
%   AF = air-fuel ratio (kg air / kg fuel)
% Output:
%   AFT = adiabatic flame temperature (K)
```

- ```
  function AF = findAirFuelRatio( ...
    fuel, fuelMoleFrac, fuelT, air, airMoleFrac, airT, AFT )
  % FINDAIRFUELRATIO Find air-fuel ratio based on complete-
  % combustion adiabatic flame temperature of fuel in air
  % Inputs/Output:
  %   ... [see above] ...
  ```

7) **20 points.** Repeat part (4b)—determining the thermal efficiency, heat rate (both on the LHV° basis), specific work, turbine exhaust temperature, and plotting the P-v, T-s, and h-s diagrams—but accounting for the thermodynamics of ideal-gas reacting (combusting) mixtures this time.

   The fuel (natural gas) and oxidizer (environmental air) are composed of the following:

Fuel component	mol%	Air component	mol%
Methane	90.7	Oxygen	20.74
Ethane	3.6	Nitrogen	77.30
Propane	1.9	Argon	0.92
Nitrogen	1.8	Carbon dioxide	0.03
Carbon dioxide	1.0	Water	1.01
Oxygen	1.0		

   In order to promote fuel/air mixing, inject the fuel into the combustor through nozzles at a pressure that is twice the combustor inlet pressure. A fuel compressor is therefore needed, though *not* a fuel intercooler. Otherwise, the cycle parameters (LPC and overall cycle pressure ratios, turbine inlet temperature) and device performance metrics (polytropic efficiencies, pinch temperature difference, and pressure loss ratios) are to remain unchanged from parts (2) and (5b).

   A note on the cycle T-s and h-s diagrams: One has to keep in mind that the mass flow rate and composition of the working fluid vary over the engine operating cycle—first as separate streams of air and fuel, then as air-fuel mixture, and lastly as a mixture of combustion products. Therefore, for the T-s and h-s diagrams to make sense, the enthalpy and entropy values being plotted on the h and s axes should be scaled to the same basis. (No mixing of apples and oranges.) For this project, plot working fluid entropy per unit mass of air intake by the low-pressure compressor.

   Deliverables: (a) A MATLAB script named `reactiveLMS100.m`, which calls the functions developed in Parts 4(a), 5, and 6 above to determine the LMS100 engine performance metrics, as well as calls the basic MATLAB plotting functions to generate the required diagrams

   - Plot each of the P-v, T-s, and h-s diagrams for both Parts 4(b) and 7 on the same set of axes to facilitate comparison between the results using the non-reactive and reactive models. In both cases, connect the state points on the diagrams with straight, dashed lines (which imply the exact irreversible process paths are not known).
  ```

- Do not neglect proper annotation of the figures using functions such as `title`; `xlabel`, `ylabel` (including units in parentheses); `legend` or `gtext`

(b) Append the reactive model results to the table in `LMS100EnginePerformanceMetrics.docx`

Analysis Requirements and Hints

See requirements and hints laid out in Project 1 description.

Deliverables

A formal write-up is not necessary for this assignment. Please turn in a ZIP or RAR folder containing the MATLAB functions, scripts, and Word documents listed in the assignment.

Your grade will be assigned based on the analysis approach (whether the code makes sense), the numerical results, and its presentation quality (whether the tables and figures are formatted neatly and professionally), as well as the performance of the code when exercised in real time.

References

[1] M. J. Reale, "New high efficiency simple cycle gas turbine: GE's LMS100™," GE Energy Rep. GER-4222A, 2004. Available on Canvas.

[2] General Electric, "GE LMS100 energy efficiency and DLE2 combustion technology," at the International Energy & Environment Fair and Conference, June 2011. Available on Canvas.

[3] California Energy Commission, *Energy Facility Status: Power Plant Projects Since 1996*, accessed Sept. 1, 2015. Available: http://energy.ca.gov/sitingcases/all_projects.html