**Aircraft Power Modeller (APModeller)**

APModeller is a python library for the modelling of power and energy requirements throughout a flight.

**Version history**

|  |  |  |
| --- | --- | --- |
| 15/09/20 | V0 - Initial release | Alex Pynn (alexpynn@googlemail.com) |
|  |  |  |

**Loading APModeller**

In a new python workspace import the library in the standard way.

The two components needed from the library are the Aircraft class and the atmosphere class:

from APModeller import Aircraft, atmosphere

**Initialising the operating atmosphere**

To create the atmosphere, we need to create an atmosphere object, this can be done by assigning an instance of the atmosphere class to a named variable.

The object will take one optional input which is a temperature shift away from ISA conditions. If this input is not given then no shift will be applied and ISA standard conditions are used.

atm = atmosphere(shift=3)

**Defining your aircraft**

Aircraft properties are defined in dictionaries. The first dictionary contains basic values for use in the simulation. All values are in SI units.

HERAProps = {'TOMass': 22000,

'WingArea': 61.7,

'Cd0': 0.027403,

'Cdi': 0.031,

'GATOREff': 0.98,

'PropEff': 0.85,

'StallSpeed': 47.6}

The next one contains information about the turbo generator in use. Two different modes can be implemented from this information: 1) The PSFC is treated as a constant, 2) The PSFC applies mapping functions to adjust a nominal PSFC value.

To implement mode 2, set: 'PSFC\_Constant': None.

Within the PSFC map are the Power, Velocity and Altitude factors, these factors take a list of co-efficient values that can be either fitted to test data or predicted methods, e.g. [1,0,5] would create a multiplier of .

TurboGenerator = {'PSFC\_Constant': None,

'PSFC\_Map': {'PowerFactor': [0.628, -1.2146, 1.5826],

'VelocityFactor': [1, 0],

'AltitudeFactor': [1],

'NormalPower': 2.2e6,

'NormalPSFC': 0.28}

}

The battery properties are optional but failure to include them will mean the analysis can not be completed with batteries.

Batterys = {'Mass': 3500,

'ED': 500,

'ReserveFraction': 0.325,

'StartingFraction': .95}

**Initialising your aircraft**

To create an aircraft object with the properties above and assign it to the name ‘HERA’, use:

HERA = Aircraft(HERAProps, atm, TurboGenerator, Batteries=Batterys)

**Start a new flight**

To create a blank starting point and clear the workspace (highly recommended) use Aircraft.NewMission()

HERA.NewMission()

**Example 1:**

