Gas HALE model

```
from numpy import pi
from gpkit import VectorVariable, Variable, Model, units
from gpkit.tools import te_exp_minus1
import gpkit
import numpy as np
gpkit.settings['latex_modelname'] = False

class GasPoweredHALE(Model):
    def setup(self):
        constraints = []
```

Flight segment definitions

```
        # define number of segments
        NSeg = 3
        #Note: NSeg has to be an odd number
        # defining indices of different flight segments
        NLoiter = (NSeg-1)/2
        if NSeg == 3:
            NCruise = [0,2]
        elif NSeg == 7:
            Nclimb = [0,2,4,6]
            NCruise = [1,5]
```

Fuel weight model

```
        MTOW = Variable('MTOW', 'lbf', 'max take off weight')
        W_end = VectorVariable(NSeg, 'W_{end}', 'lbf', 'segment-end weight')
        W_fuel = VectorVariable(NSeg, 'W_{fuel}', 'lbf', 'segment-fuel weight')
        W_zfw = Variable('W_{zfw}', 'lbf', 'Zero fuel weight')
        W_pay = Variable('W_{pay}',10,'lbf', 'Payload weight')
        W_avionics = Variable('W_{avionics}', 2, 'lbf', 'Avionics weight')
        f_airframe = Variable('f_{airframe}', 0.25, '-', 'airframe weight fraction')
        W_airframe = Variable('W_{airframe}', 'lbf', 'airframe weight')
        W_begin = W_end.left # define beginning of segment weight
        W_begin[0] = MTOW
```

Assumptions

- Mass take off weight is greater than the end of the first segment weight plus that segment fuel weight.
- The end of each flight segment weight must be greater than the next end of flight segment weight plus the fuel weight of the next flight segment.
- The end of the last flight segment weight must be greater than the zero fuel weight

```
constraints.extend([MTOW >= W_end[0] + W_fuel[0],
                    W_end[:-1] >= W_end[1:] + W_fuel[1:],
                    W_end[-1] >= W_zfw,
                    W_airframe >= f_airframe*MTOW])
```

Steady level flight model

Assumptions

- Steady state flight
- Angle of attack zero
- Thrust equals drag
- Lift equals weight
- The weight is approximated by the average weight of the flight segment.
- The average weight can be approximated by: $W_{avg} = \sqrt{W_{begin} W_{end}}$. Where W_{begin} is the beginning of the flight segment and W_{end} is the end of the flight segment.

```
CD = VectorVariable(NSeg, 'C_D', '-', 'Drag coefficient')
CL = VectorVariable(NSeg, 'C_L', '-', 'Lift coefficient')
V = VectorVariable(NSeg, 'V', 'm/s','cruise speed')
rho = VectorVariable(NSeg, r'\rho', 'kg/m^3', 'air density')
S = Variable('S', 16, 'ft^2', 'wing area')
eta_prop = VectorVariable(NSeg, r'\eta_{prop}', [0.8,0.6,0.8], '-',
                          'propulsive efficiency')
P_shaft = VectorVariable(NSeg, 'P_{shaft}', 'hp', 'Shaft power')
```

Climb model

```
h_dot = Variable(NSeg, 'h_{dot}', [200,0,0], 'ft/min', 'Climb rate')
```

```
constraints.extend([P_shaft >= V*(W_end+W_begin)/2*CD/CL/eta_prop + W_begin*h_dot/et
                    0.5*rho*CL*S*V**2 >= (W_end+W_begin)/2])
```

Engine Model

```
W_eng = Variable('W_{eng}', 'lbf', 'Engine weight')
W_engtot = Variable('W_{eng-tot}', 'lbf', 'Installed engine weight')
W_engref = Variable('W_{eng-ref}', 4.4107, 'lbf', 'Reference engine weight')
P_shaftref = Variable('P_{shaft-ref}', 2.295, 'hp', 'reference shaft power')
```

Engine Weight constraints

```
constraints.extend([W_eng/W_engref >= 0.5538*(P_shaft/P_shaftref)**1.075,
                    W_engtot >= 2.572*W_eng**0.922*units('lbf')**0.078])
```

Weight breakdown

```
constraints.extend([W_airframe >= f_airframe*MTOW,
                    W_zfw >= W_pay + W_avionics + W_airframe + W_engtot])
```

Breguet Range

Assumptions

- Constant speed during each flight section
- Constant BSFC
- The ln can be approximated using a Taylor-series expansion

```
z_bre = VectorVariable(NSeg, 'z_{bre}', '-', 'breguet coefficient')
BSFC = VectorVariable(NSeg,'BSFC', [0.5,.55,0.6], 'lbf/hr/hp', 'brake specific fuel
t = VectorVariable(NSeg, 't', 'days', 'time on station')
R = Variable('R', 200, 'nautical_miles', 'range to station')
g = Variable('g', 9.81, 'm/s^2', 'Gravitational acceleration')

constraints.extend([z_bre >= V*t*BSFC*CD/CL/eta_prop,
```

```
R == V[NCruise]*t[NCruise],
t[NLoiter] == 5*units('days'),
W_fuel/W_end >= te_exp_minus1(z_bre, 3)])
```

Aerodynamics model

Assumptions

- The wing is a box shape.
- The non-wing drag is a constant
- The stall factor is based off standard airfoil polar.
- Reference length for Reynolds number is teh chord.
- The skin friction is based off of Blasius flat plate.
- The form factor for the wing is constant.

```
CdO = Variable('C_{d0}', 0.02, '-', 'Non-wing drag coefficient')
CLmax = Variable('C_{L-max}', 1.5, '-', 'Maximum lift coefficient')
e = Variable('e', 0.9, '-', 'Spanwise efficiency')
AR = Variable('AR', '-', 'Aspect ratio')
b = Variable('b', 'ft', 'Span')
mu = Variable(r'\mu', 1.5e-5, 'N*s/m^2', 'Dynamic viscosity')
Re = VectorVariable(NSeg, 'Re', '-', 'Reynolds number')
Cf = VectorVariable(NSeg, 'C_f', '-', 'wing skin friction coefficient')
Kwing = Variable('K_{wing}', 1.3, '-', 'wing form factor')
cl_16 = Variable('cl_{16}', 0.0001, '-', 'profile stall coefficient')

constraints.extend([CD >= CdO + 2*Cf*Kwing + CL**2/(pi*e*AR) + cl_16*CL**16,
                    b**2 == S*AR,
```

In place of an actual structural model, we impose $AR \leq 20$.

```
AR <= 20,
CL <= CLmax,
Re == rho*V/mu*(S/AR)**0.5,
Cf >= 0.074/Re**0.2])
```

Atmosphere model

Assumptions

- Valid only to the top of the troposphere.

References

- wp:Density of Air

```
h = VectorVariable(NSeg, 'h', 'ft', 'Altitude')
gamma = Variable(r'\gamma',1.4,'-', 'Heat capacity ratio of air')
p_sl = Variable('p_{sl}', 101325, 'Pa', 'Pressure at sea level')
T_sl = VectorVariable(NSeg, 'T_{sl}', [288.15,288.15,288.15], 'K',
                      'Temperature at sea level')
L_atm = Variable('L_{atm}', 0.0065, 'K/m', 'Temperature lapse rate')
T_atm = VectorVariable(NSeg, 'T_{atm}', 'K', 'Air temperature')
a_atm = VectorVariable(NSeg, 'a_{atm}','m/s', 'Speed of sound at altitude')
R_spec = Variable('R_{spec}', 287.058,'J/kg/K', 'Specific gas constant of air')
TH = (g/R_spec/L_atm).value.magnitude  # dimensionless

constraints.extend([#h <= [20000, 20000, 20000]*units.m,
                    T_sl >= T_atm + L_atm*h,  # Temp decreases w/ altitude
                    rho == p_sl*T_atm**(TH-1)/R_spec/(T_sl**TH),
                    h[NLoiter] >= 15000*units('ft'), # makes sure that the loiter a
                    ])
```

Wind speed model

```
V_wind = VectorVariable(NSeg, 'V_{wind}', 'm/s', 'wind speed')
wd_cnst = Variable('wd_{cnst}', 0.0015, 'm/s/ft',
                   'wind speed constant predicted by model')
                    #0.002 is worst case, 0.0015 is mean at 45d
wd_ln = Variable('wd_{ln}', 8.845, 'm/s',
                 'linear wind speed variable')
                #13.009 is worst case, 8.845 is mean at 45deg
h_min = Variable('h_{min}', 11800, 'ft', 'minimum height')
h_max = Variable('h_{max}', 20866, 'ft', 'maximum height')

constraints.extend([V_wind >= wd_cnst*h + wd_ln,
```

```
                               V >= V_wind,
                               h[NCruise] >= h_min])
```

Conclusion

```
        objective = MTOW
        return objective, constraints

if __name__ == "__main__":
    M = GasPoweredHALE()
    M.solve()
    with open("sol.tex", "w") as f:
        f.write(M.solution.table(latex=True))
```

Solution

Cost —- 196 [lbf]

Free Variables	Value	Units	Description
AR	20		Aspect ratio
$MTOW$	196	$[lbf]$	max take off weight
$W_{airframe}$	49	$[lbf]$	airframe weight
$W_{eng-tot}$	6.381	$[lbf]$	Installed engine weight
W_{eng}	2.679	$[lbf]$	Engine weight
W_{zfw}	67.38	$[lbf]$	Zero fuel weight
b	17.89	$[ft]$	Span
C_D	[0.0619 0.0579 0.0538]		Drag coefficient
C_L	[1.21 1.14 1.12]		Lift coefficient
C_f	[0.00532 0.0055 0.00426]		wing skin friction coefficient
P_{shaft}	[2.5 2.29 2.5]	$[hp]$	Shaft power
Re	[5.2e+05 4.39e+05 1.58e+06]		Reynolds number
T_{atm}	[265 258 265]	$[K]$	Air temperature
V	[33.5 31.3 102]	$[m/s]$	cruise speed
V_{wind}	[29.8 31.3 52]	$[m/s]$	wind speed
W_{end}	[192 68.9 67.4]	$[lbf]$	segment-end weight
W_{fuel}	[3.81 123 1.51]	$[lbf]$	segment-fuel weight
ρ	[0.854 0.771 0.854]	$[kg/m^3]$	air density
h	[1.18e+04 1.5e+04 1.18e+04]	$[ft]$	Altitude
t	[0.128 5 0.042]	$[day]$	time on station
z_{bre}	[0.0196 1.05 0.0221]		breguet coefficient

Constants	Value	Units	Description
C_{L-max}	1.5		Maximum lift coefficient
C_{d0}	0.02		Non-wing drag coefficient
K_{wing}	1.3		wing form factor

L_{atm}	0.0065	$[K/m]$	Temperature lapse rate
$P_{shaft-ref}$	2.295	$[hp]$	reference shaft power
R	200	$[nmi]$	range to station
R_{spec}	287.1	$[J/K/kg]$	Specific gas constant of air
S	16	$[ft^2]$	wing area
$W_{avionics}$	2	$[lbf]$	Avionics weight
$W_{eng-ref}$	4.411	$[lbf]$	Reference engine weight
W_{pay}	10	$[lbf]$	Payload weight
μ	1.5e-05	$[N * s/m^2]$	Dynamic viscosity
cl_{16}	0.0001		profile stall coefficient
e	0.9		Spanwise efficiency
$f_{airframe}$	0.25		airframe weight fraction
h_{dot}	3	$[ft/min]$	Climb rate
h_{min}	1.18e+04	$[ft]$	minimum height
p_{sl}	1.013e+05	$[Pa]$	Pressure at sea level
wd_{cnst}	0.0015	$[m/ft/s]$	wind speed constant predicted by model
wd_{ln}	8.845	$[m/s]$	linear wind speed variable
$BSFC$	[0.5 0.55 0.6]	$[lbf/hp/hr]$	brake specific fuel consumption
T_{sl}	[288 288 288]	$[K]$	Temperature at sea level
η_{prop}	[0.8 0.6 0.8]		propulsive efficiency

Sensitivities	Value	Units	Description
$f_{airframe}$	5.373		airframe weight fraction
wd_{cnst}	4.916		wind speed constant predicted by model
C_{d0}	2.816		Non-wing drag coefficient
$BSFC$	[0.135 7.21 0.163]		brake specific fuel consumption
K_{wing}	1.999		wing form factor
wd_{ln}	1.933		linear wind speed variable
W_{pay}	1.096		Payload weight
R_{spec}	0.7217		Specific gas constant of air
$W_{eng-ref}$	0.6451		Reference engine weight
μ	0.3997		Dynamic viscosity
L_{atm}	0.3118		Temperature lapse rate
R	0.2985		range to station
$W_{avionics}$	0.2193		Avionics weight
h_{min}	0.1374		minimum height
T_{sl}	[0.224 0.182 0.00438]		Temperature at sea level
cl_{16}	0.1257		profile stall coefficient
S	-0.5218		wing area
$P_{shaft-ref}$	-0.6935		reference shaft power
p_{sl}	-0.7217		Pressure at sea level
η_{prop}	[-0.822 -7.21 -0.171]		propulsive efficiency
e	-3.251		Spanwise efficiency