Hyperloop Documentation

Release 2.0

NASA MARTI

October 14, 2016

Contents

Contents:

CHAPTER 1

hyperloop package

1.1 Subpackages

1.1.1 hyperloop.Hardware package

Module contents

1.1.2 hyperloop.Meshing package

Module contents

1.1.3 hyperloop.Python package

Subpackages

hyperloop.Python.mission package

Submodules

hyperloop.Python.mission.body_frame_acceleration module

class `hyperloop.Python.mission.body_frame_acceleration.`**`BodyFrameAcceleration`**

Bases: `openmdao.core.component.Component`

Super sweet docs, bro

`solve_nonlinear`(*p*, *u*, *r*)

hyperloop.Python.mission.com module

hyperloop.Python.mission.lat_long module

hyperloop.Python.mission.mission_drag module

class `hyperloop.Python.mission.mission_drag.`**`MissionDrag`**

Bases: `openmdao.core.component.Component`

Returns Drag : float

Total drag force acting on pod. Default value is 0.0.

Notes

Computes the total drag force acting on the pod. Will be fed into Mission EOM component

solve_nonlinear(*params*, *unknowns*, *resids*)

Returns Drag : float

Total drag force acting on pod. Default value is 0.0.

Notes

Computes the total drag force acting on the pod. Will be fed into Mission EOM component Evaluates equation D = .5*Cd*rho*(V**2)*S + D_magnetic

hyperloop.Python.mission.mission_thrust module

class `hyperloop.Python.mission.mission_thrust.`**`MissionThrust`**

Bases: `openmdao.core.component.Component`

Params Drag coefficient : float

Drag coefficient of pod. Default value is .2. More accurate results will come from CFD

Reference Area : float

Reference area of the pod. Default value is 1.4 m**2. Value will be pulled from geometry module

Tube Pressure : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

Ambient Temperature : float

Tunnel ambient temperature. Default value is 298 K.

Ideal Gas Constant : float

Ideal gas constant. Default valut is 287 J/(m*K).

Magnetic Drag : float

Drag force from magnetic levitation in N. Default value is 150 N. Value will come from levitation analysis

Pod Thrust : float

Thrust produced by pod compressed air. Default value 3500 N. Will pull value from NPSS

Inclination angle : float

Incline angle of pod in NED frame. Default value is 0.0 rad.

Pod Speed : float

Speed of the pod. Default value is 335 m/s.

Pod mass : float

total mass of pod. Default value is 3100 kg. Value will come from weight component

Gravity : float

Gravitational acceleration. Default value is 9.81 m/s**2

Returns **Thrust** : float

Total thrust force acting on pod. Default value is 0.0.

Notes

Computes the total thrust force acting on the pod assuming 1g acceleration in booster section and constant value of compressor thrust in coasting sections. Will be fed into Mission EOM component

solve_nonlinear (*params*, *unknowns*, *resids*)

Params **Drag coefficient** : float

Drag coefficient of pod. Default value is .2. More accurate results will come from CFD

Reference Area : float

Reference area of the pod. Default value is 1.4 m**2. Value will be pulled from geometry module

Tube Pressure : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

Ambient Temperature : float

Tunnel ambient temperature. Default value is 298 K.

Ideal Gas Constant : float

Ideal gas constant. Default valut is 287 J/(m*K).

Magnetic Drag : float

Drag force from magnetic levitation in N. Default value is 150 N. Value will come from levitation analysis

Pod Thrust : float

Thrust produced by pod compressed air. Default value 3500 N. Will pull value from NPSS

Inclination angle : float

Incline angle of pod in NED frame. Default value is 0.0 rad.

Pod Speed : float

Speed of the pod. Default value is 335 m/s.

Pod mass : float

total mass of pod. Default value is 3100 kg. Value will come from weight component

Gravity : float

Gravitational acceleration. Default value is 9.81 m/s**2

Returns **Thrust** : float

Total thrust force acting on pod. Default value is 0.0.

Notes

Evaluates equation T = m*g*(1+sin(theta)) + .5*Cd*rho*(V**2)*S + D_magnetic

hyperloop.Python.mission.pod_thrust_and_drag module

hyperloop.Python.mission.rhs module

hyperloop.Python.mission.terrain module

hyperloop.Python.mission.usgs_data_converter module

Module contents

hyperloop.Python.pod package

Subpackages

hyperloop.Python.pod.cycle package

Submodules

hyperloop.Python.pod.cycle.comp_len module

class `hyperloop.Python.pod.cycle.comp_len.`**`CompressorLen`**

Bases: `openmdao.core.component.Component`

The CompressorMass class represents a compressor length component in an OpenMDAO model.

A *CompressorMass* models length of a compressor that uses NPSS data to obtain enthalpy data,and mass_flow for a particular pressure ratio. It also uses a correlation derived by Micheal Tong at NASA Glenn Center to obtain Compressor Length.

Params **h_in** : float

Heat in. (kJ/kg)

h_out : float

Heat out. (kJ/kg)

comp_inletArea : float

Compressor Inlet Area. (m**2)

h_stage : float

enthalpy added per stage. (kJ/kg)

Returns **comp_len** : float

Length of Compressor (m)

References

solve_nonlinear (*params*, *unknowns*, *resids*)

hyperloop.Python.pod.cycle.compressor_mass module

class hyperloop.Python.pod.cycle.compressor_mass.**CompressorMass**

Bases: openmdao.core.component.Component

The CompressorMass class represents a compressor mass component in an OpenMDAO model.

A *CompressorMass* models mass of a compressor that uses NPSS data to obtain enthalpy data,and mass_flow for a particular pressure ratio. It also uses a correlation derived by Miceal Tong at NASA Glenn Center to obtain Compressor Mass.

Params **comp_eff** : float

Compressor Efficiency. (unitless)

mass_flow : float

Mass Flow for Compressor. (kg/s)

h_in : float

Heat in. (kJ/kg)

h_out : float

Heat out. (kJ/kg)

comp_inletArea : float

Compressor Inlet Area. (m**2)

Returns **comp_mass** : float

Compressor Mass (kg)

References

[R3], *[R4]*

solve_nonlinear (*params*, *unknowns*, *resids*)

Runs the *CompressorMass* component and sets its respective outputs to their calculated results

Args **params** : *VecWrapper*

VecWrapper containing parameters

unknowns : *VecWrapper*

VecWrapper containing outputs and states

resids : *VecWrapper*

VecWrapper containing residuals

hyperloop.Python.pod.cycle.cycle_group module

hyperloop.Python.pod.cycle.flow_path module

hyperloop.Python.pod.cycle.flow_path_inputs module A group that converts user inputs into format flow_path needs

class `hyperloop.Python.pod.cycle.flow_path_inputs.`**`FlowPathInputs`**

Bases: `openmdao.core.component.Component`

Params **tube_pressure** : float

Ambient temperature inside tube (K)

pod_mach : float

Pod mach number (unitless)

comp_inlet_area : float

Compressor inlet area (m**2)

tube_temp : float

Ambient temperature inside tube (K)

gamma : float

Specific heat ratio

comp_mach : float

Compressor mach number (unitless)

R : float

Ideal gas constant (J/(kg*K))

eta : float

Efficiency of inlet (unitless)

Returns **Pt** : float

Total pressure inside tube (Pa)

Tt : float

Total temperature inside tube (K)

m_dot : float

Mass flow rate (kg/s)

`solve_nonlinear`(*params*, *unknowns*, *resids*)

Module contents

hyperloop.Python.pod.drivetrain package

Subpackages

Submodules

hyperloop.Python.pod.drivetrain.battery module

class `hyperloop.Python.pod.drivetrain.battery.`**`Battery`**

Bases: `openmdao.core.component.Component`

The *Battery* class represents a battery component in an OpenMDAO model.

A *Battery* models battery performance by finding a general battery voltage performance curve (*[R5]*, *[R6]*) based on standard cell properties and can be used to determine the number of cells and cell configuration needed to meet specification.

Params **des_time** : float

time until design power point (h)

time_of_flight : float

total mission time (h)

battery_cross_section_area : float

cross_sectional area of battery used to compute length (cm^2)

des_power : float

design power (W)

des_current : float

design current (A)

q_l : float

discharge limit (unitless)

e_full : float

fully charged voltage (V)

e_nom : float

voltage at end of nominal voltage (V)

e_exp : float

voltage at end of exponential zone (V)

q_n : float

single cell capacity (A*h)

t_exp : float

time to reach exponential zone (h)

t_nom : float

time to reach nominal zone (h)

r : float

resistance of individual battery cell (Ohms)

cell_mass : float

mass of a single cell (g)

cell_height : float

height of a single cylindrical cell (mm)

cell_diameter : float

diamter of a single cylindrical cell (mm)

Outputs **n_cells** : float

total number battery cells (unitless)

battery_length : float

length of battery (cm)

output_voltage : float

output voltage of battery configuration (V)

battery_mass : float

total mass of cells in battery configuration (kg)

battery_volume : float

total volume of cells in battery configuration (cm^3)

battery_cost : float

total cost of battery cells in (USD)

Notes

Battery cost based on purchase of 1000 bateries from ecia (*[R7]*) price listing

References

[R5], *[R6]*, *[R7]*

solve_nonlinear (*params*, *unknowns*, *resids*)

Runs the *Battery* component and sets its respective outputs to their calculated results

Args **params** : *VecWrapper*

VecWrapper containing parameters

unknowns : *VecWrapper*

VecWrapper containing outputs and states

resids : *VecWrapper*

VecWrapper containing residuals

hyperloop.Python.pod.drivetrain.drivetrain module

class `hyperloop.Python.pod.drivetrain.drivetrain.`**`Drivetrain`**

Bases: `openmdao.core.group.Group`

The *Drivetrain* group represents a *Group* of an electric motor, inverter and battery in an OpenMDAO model.

Models the performance and behavior of a combined electric motor, battery, and inverter following previous work from *[R8]*

Params **design_torque** : float

design torque at max rpm (N*m)

design_power : float

desired design value for motor power (W)

motor_max_current : float

max motor phase current (A)

motor_LD_ratio : float

length to diameter ratio of motor (unitless)

motor_oversize_factor : float

scales peak motor power by this figure

inverter_efficiency : float

power out / power in (W)

des_time : float

time until design power point (h)

time_of_flight : float

total mission time (h)

battery_cross_section_area : float

cross_sectional area of battery used to compute length (cm^2)

Outputs **battery_mass** : float

total mass of cells in battery configuration (kg)

battery_volume : float

total volume of cells in battery configuration (cm^3)

battery_cost : float

total cost of battery cells in (USD)

battery_length : float

length of battery (cm)

motor_volume : float

D^2*L parameter which is proportional to Torque (mm^3)

motor_diameter : float

motor diameter (m)

motor_mass : float

mass of motor (kg)

motor_length : float

motor length (m)

motor_power_input : float

total required power input into motor (W)

Components **Motor** : ElectricMotor

Represents a BLDC electric motor

Inverter : Inverter

Represents an Inverter

Battery : Battery

Represents a Battery

References

[R8]

hyperloop.Python.pod.drivetrain.electric_motor module Models the equivalent circuit model of a brushless DC (BLDC) motor to perform motor sizing. Calculates Phase Current, Phase Voltage, Frequency, motor size, and Weight.

class `hyperloop.Python.pod.drivetrain.electric_motor.`**`Motor`**

Bases: `openmdao.core.component.Component`

Represents an electric motor which can calculate output current and voltage based on motor sizing parameters and losses. Used in conjunction with motor_balance to find the correct no load current for this motor

Based on work done by Gladin et. al. (*[R9]*)

Params **w_operating** : float

operating speed of motor (rad/s)

design_torque : float

torque at max rpm (N*m)

I0 : float

motor No-load Current (A)

motor_max_current : float

max motor phase current (A)

pole_pairs : float

number of motor pole pairs (unitless)

n_phases : float

number of motor phases (unitless)

max_torque : float

maximum possible torque for motor (N*m)

power_iron_loss : float

total power loss due to iron core (W)

power_mech : float

mechanical power output of motor (W)

power_windage_loss : float

friction loss from motor operation (W)

winding_resistance : float

total resistance of copper winding (ohm)

Outputs **current** : float

current through motor (A)

voltage : float

voltage across motor (V)

phase_current : float

phase current through motor (A)

phase_voltage : float

phase voltage across motor (V)

frequency : float

Frequency of electric output waveform (Hz)

motor_power_input : float

total required power input into motor (W)

References

[R9]

solve_nonlinear(*params*, *unknowns*, *resids*)

class hyperloop.Python.pod.drivetrain.electric_motor.**MotorBalance**

Bases: openmdao.core.component.Component

Creates an implicit connection used to balance conservation of energy across the motor by computing the residual of power_in - power_out.

Params **motor_power_input** : float

total required power input into motor (W)

current : float

current through motor (A)

voltage : float

voltage across motor (V)

Unknowns **I0** : float

motor No-load Current (A)

apply_nonlinear(*params*, *unknowns*, *resids*)

solve_nonlinear(*params*, *unknowns*, *resids*)

class hyperloop.Python.pod.drivetrain.electric_motor.**MotorGroup**

Bases: openmdao.core.group.Group

MotorGroup represents a BLDC motor in an OpenMDAO model which can calculate size, mass, and various performance characteristics of a BLDC motor based on input paramters

Params **design_power** : float

desired design value for motor power (W)

design_torque : float

desired torque at max rpm (N*m)

motor_max_current : float

max motor phase current (A)

motor_LD_ratio : float

length to diameter ratio of motor (unitless)

motor_oversize_factor : float

scales peak motor power by this figure (unitless)

Outputs **current** : float

current through motor (A)

voltage : float

voltage across motor in (V)

phase_current : float

phase current through motor (A)

phase_voltage : float

phase voltage across motor (V)

frequency : float

Frequency of electric output waveform (Hz)

motor_power_input : float

total required power input into motor (W)

motor_volume : float

D^2*L parameter which is proportional to Torque (mm^3)

motor_diameter : float

diameter of motor (m)

motor_mass : float

mass of motor (kg)

motor_length : float

motor length (m)

motor_power_input : float

total required power input into motor (W)

Components **motor** : Motor

Calculates the electrical characteristics of the motor

motor_size : MotorSize

Calculates the size, mass, and performance characteristics of the motor

motor_balance : MotorBalance

Calculates the residual in the conservation of energy equation between input power and total power used by the motor from mechanical output and additional losses

class `hyperloop.Python.pod.drivetrain.electric_motor.`**`MotorSize`**

Bases: `openmdao.core.component.Component`

MotorSize models the size of a BLDC motor based on a set of input paramters using data from existing commerical BLDC motors and work done by (*[R10]*)

Outputs **motor_volume** : float

D^2*L parameter which is proportional to Torque (mm^3)

motor_diameter : float

diameter of motor winding (m)

motor_mass :float

mass of motor (kg)

motor_length : float

length of motor (m)

w_base : float

base speed of motor (rad/s)

max_torque : float

maximum possible torque for motor (N*m)

power_iron_loss : float

total power loss due to iron core (W)

power_mech : float

mechanical power output of motor (W)

power_windage_loss : float

friction loss from motor operation (W)

winding_resistance : float

total resistance of copper winding (ohm)

w_operating : float

operating speed of motor (rad/s)

References

[R10]

`calculate_copper_loss`(*motor_diameter*, *motor_max_current*, *n_phases*)

Calculates the resistive losses in the copper winding of a BLDC motor operating at *motor_max_current* and *n_phases* with dimension specified by *motor_diameter*.

Returns float

the total resistive losses of the copper winding (W)

calculate_iron_loss(*motor_diameter*, *motor_speed*, *motor_length*, *core_radius_ratio*, *pole_pairs*)
Calculates the iron core magnetic losses of a BLDC motor with dimensions given by *motor_length* and *motor_diameter* operating at speed *motor_speed*.

Args **motor_diameter** : float

diameter of motor winding (m)

speed : float

desired output shaft mechanical motor_speed (RPM)

motor_length : float

motor length (m)

core_radius_ratio : float

ratio of inner diameter to outer diameter of core (unitless)

Returns float

the total iron core losses of the motor (W)

calculate_windage_loss(*w_operating*, *motor_diameter*, *motor_length*)
Calculates the windage or frictional losses of a BLDC motor with dimensions given by *motor_length* and *motor_diameter* operating at motor_speed *w_operating*.

Args **w_operating** : float

operating speed of motor (rad/s)

motor_diameter : float

diameter of motor winding (m)

motor_length : float

motor length (m)

Returns float

the total windage losses of the motor (W)

solve_nonlinear(*params*, *unknowns*, *resids*)
Runs the MotorSize component and sets its respective outputs to their calculated results in the unknowns *VecWrapper*.

Args **params** : *VecWrapper*

VecWrapper containing parameters

unknowns : *VecWrapper*

VecWrapper containing outputs and states

resids : *VecWrapper*

VecWrapper containing residuals

hyperloop.Python.pod.drivetrain.inverter module

class `hyperloop.Python.pod.drivetrain.inverter.`**`Inverter`**
Bases: `openmdao.core.component.Component`

The *Inverter* class represents a BLDC inverter in an OpenMDAO model

The *Inverter* class models the efficiency loss across a typical BLDC inverter following the example from *[R11]*.

Params **inverter_efficiency** : float

power out / power in (W)

output_voltage : float

amplitude of AC output voltage (A)

output_current : float

amplitude of AC output current (A)

output_frequency : float

frequency of AC output (Hz)

input_voltage : float

amplitude of DC input voltage (V)

Outputs **input_current** : float

amplitude of DC input current (A)

input_power : float

amplitude of DC input current (W)

References

[R11]

solve_nonlinear (*params*, *unknowns*, *resids*)

Runs the *Battery* component and sets its respective outputs to their calculated results

Args **params** : *VecWrapper*

VecWrapper containing parameters

unknowns : *VecWrapper*

VecWrapper containing outputs and states

resids : *VecWrapper*

VecWrapper containing residuals

output_power = params['output_voltage'] * params[

'output_current'] * 3.0 * np.sqrt(2.0 / 3.0)

Module contents

hyperloop.Python.pod.magnetic_levitation package

Submodules

hyperloop.Python.pod.magnetic_levitation.breakpoint_levitation module Current Levitation Code Outputs minimum mass and area of magnets needed for levitation at desired breakpoint velocity. Outputs Halbach array wavelength, track resistance, and inductance can be used to find drag force at any velocity using given pod weight.

class `hyperloop.Python.pod.magnetic_levitation.breakpoint_levitation.`**`BreakPointDrag`**

Bases: `openmdao.core.component.Component`

Current Break Point Drag Calculation very rough. Needs refinement. Default parameters taken from Inductrack I. Calculates minimum drag given at set breakpoint velocity desired with given track parameters.

Params **m_pod** : float

Mass of the hyperloop pod. Default value is 3000.

b_res : float

Residual strength of the Neodynium Magnets. Default value is 1.48.

num_mag_hal : float

Number of Magnets per Halbach Array. Default value is 4

mag_thk : float

Thickness of Magnet. Default value is 0.15.

g : float

Gravitational Acceleration. Default value is 9.81.

l_pod : float

Length of the Hyperloop pod. Default value is 22.

gamma : float

Percent factor used in Area. Default value is 1.

w_mag : float

Width of magnet array. Default value is 3.

spacing : float

Halbach Spacing Factor. Default value is 0.0.

w_strip : float

Width of conductive strip. Default value is .005.

num_sheets : float

Number of laminated sheets. Default value is 1.0.

delta_c : float

Single layer thickness. Default value is .005.

strip_c : float

Center strip spacing. Default value is .0105.

rc : float

Electric resistance. Default value is 1.713*10**-8.

MU0 : float

Permeability of Free Space. Default value is 4*pi*10^-7.

vel_b : float

Breakpoint velocity of the pod. Default value is 23.

h_lev : float

Levitation height. Default value is .01.

d_pod : float

Diameter of the pod. Default value is 1.

track_factor : float

Factor to adjust track width. Default value is .75.

Returns **lam** : float

Wavelength of the Halbach Array. Default value is 0.0.

track_res : float

Resistance of the track. Default value is 0.0

track_ind : float

Inductance of the track. Default value is 0.0.

pod_weight : float

Weight of the Pod. Default value is 0.0.

References

[1] Friend, Paul. Magnetic Levitation Train Technology 1. Thesis. Bradley University, 2004. N.p.: n.p., n.d. Print.

`solve_nonlinear`(*params*, *unknowns*, *resids*)

class `hyperloop.Python.pod.magnetic_levitation.breakpoint_levitation.`**`MagMass`**

Bases: `openmdao.core.component.Component`

Current Magnet Mass Calculation very rough. Needs refinement. Default parameters taken from Inductrack I. Calculates minimum magnet mass needed at set breakpoint velocity desired with given track parameters.

Params **m_pod** : float

Mass of the pod with no magnets. Default value is 3000.0 kg

mag_thk : float

Thickness of Magnet. Default value is 0.15.

rho_mag : float

Density of Magnet. Default value is 7500.

l_pod : float

Length of the Hyperloop pod. Default value is 22.

gamma : float

Percent factor used in Area. Default value is 1.

d_pod : float

Diameter of the pod. Default value is 1.

track_factor : float

Factor to calculate track width. Default value is .75.

w_mag : float

Width of magnet array. Default value is 3.

cost_per_kg : flost

Cost of the magnets per kilogram. Default value is 44.

track_factor : float

Factor to adjust track width. Default value is .75.

Returns **mag_area** : float

Total area of the magnetic array. Default value is 0.0

cost : float

Total cost of the magnets. Default value is 0.0.

total_pod_mass : float

Final mass of the pod with magnets. Default value is 0.0.

Notes

[1] Friend, Paul. Magnetic Levitation Train Technology 1. Thesis. Bradley University, 2004. N.p.: n.p., n.d. Print.

solve_nonlinear(*params*, *unknowns*, *resids*)

hyperloop.Python.pod.magnetic_levitation.levitation_group module

class `hyperloop.Python.pod.magnetic_levitation.levitation_group.`**`LevGroup`**

Bases: `openmdao.core.group.Group`

The Levitation group represents a *Group* of the size of magnets required for levitation at breakpoint velocity, and the magnetic drag resulting from this levitation at a given speed. These values are computed in an OpenMDAO model.

Models the levitation parameters following previous work from *[R12]*

Params **m_pod** : float

mass of the pod (kg)

l_pod : float

length of the pod (m)

d_pod : float

diameter of the pod (m)

vel_b : float

desired breakpoint levitation speed (m/s)

h_lev : float

Levitation height. Default value is .01

vel : float

desired magnetic drag speed (m/s)

Outputs **mag_drag** : float

magnetic drag from levitation system (N)

total_pod_mass : float

total mass of the pod including magnets (kg)

Components **Drag** : BreakPointDrag

Represents the drag and magnetic parameters need for levitation at breakpoint speed.

Mass : MagMass

Represents the mass of magnets needed for levitation at breakpoint speed.

MDrag : MagDrag

Represents the magnetic drag acquired from levitation at a given speed.

References

[R12]

hyperloop.Python.pod.magnetic_levitation.magnetic_drag module Magnetic Drag Calculation. Default track and magnet parameters taken from breakpointlev.py. Calculates Magnetic Drag at set velocity desired with given parameters.

class `hyperloop.Python.pod.magnetic_levitation.magnetic_drag.`**`MagDrag`**

Bases: `openmdao.core.component.Component`

Params **vel** : float

Desired velocity of the pod. Default value is 350.

track_res : float

Resistance of the track. Default value is 3.14e-4.

track_ind : float

Inductance of the track. Default value is 3.59e-6.

pod_weight : float

Weight of the Pod. Default value is 29430.0.

lam : float

Wavelength of the Halbach Array. Default value is 0.125658.

Returns **omega** : float

Frequency of Induced current at chosen velocity. Default value is 0.0.

mag_drag_lev : float

Magnetic Drag Force from Levitation. Default value is 0.0.

mag_drag_prop : float

Magnetic Drag Force from Propulsion (TBD). Default value is 0.0.

mag_drag : float

Total Magnetic Drag Force. Default value is 0.0.

Notes

[1] Friend, Paul. Magnetic Levitation Train Technology 1. Thesis. Bradley University, 2004. N.p.: n.p., n.d. Print.

`solve_nonlinear`(*params*, *unknowns*, *resids*)

Module contents

Submodules

hyperloop.Python.pod.drag module

class `hyperloop.Python.pod.drag.`**`Drag`**

Bases: `openmdao.core.component.Component`

Params M_pod : float

Pod mach number. Default value is .8

mach_array : array

Array of mach numbers corresponding to CFD data. Default value is np.zeros((1,7))

cd array : array

Array of drag coefficient values at correponding mach numbers from CFD. Default value is np.zeros((1,7))

Returns Cd : float

Interpolated drag coefficient based on pod mach number.

Notes

Interpolates the drag coefficient of the pod using data mach vs. drag coefficient data from CFD. Component interpolates drag coefficient based on pod mach number.

`solve_nonlinear`(*p*, *u*, *r*)

hyperloop.Python.pod.pod_geometry module

class `hyperloop.Python.pod.pod_geometry.`**`PodGeometry`**

Bases: `openmdao.core.component.Component`

Params A_payload : float

Cross sectional area of passenger compartment. Default value is 2.72

gam : float

Ratio of specific heats. Default value is 1.4

R : float

Ideal gas constant. Default valut is 287 J/(m*K).

p_tunnel : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

M_pod : float

pod Mach number. Default value is .8. value will be set by user

A_p : float

Cross sectional area of passenger compartment. Default value is 2.72 m**2

L_comp : float

length of the compressor. Default value is 1.0 m.

L_bat : float

length of battery. Default value is 1.0 m.

L_inverter : float

length of inverter. Default value is 1.0 m.

L_trans : float

length of transformer. Default value is 1.0 m

L_p : float

length of passenger compartment. Default value is 11.2 m

L_conv : float

length of the converging section of the nozzle. Default value is .3 m

L_div : float

length of the diverging section of the nozzle. Default value is 1.5 m

L_motor : float

length of the motor. Default value is 1.0 m

p_duct : float

Static pressure in the duct. Default value is 6800.0 Pa

p_passenger : float

Static pressure in passenger section. Default value is 101.3e3 Pa

rho_pod : float

Density of pod material. Default value is 2700.0 kg/m**3

n_passengers : float

Number of passengers per pod. Default value is 28

SF : float

Structural safety factor. Default value is 1.5

Su : float

Ultimate strength of pod material. Default value is 50.0e6 Pa

A_duct : float

Area of flow duct within pod. Default value is .3 m**2

dl_passenger : float

Length of passenger compartment per passenger row. Default value is .8 m.

g : float

gravitational acceleration. Default value is 9.81 m/s**2

Returns A_pod : float

Cross sectional area of pod.

D_pod : float

Diameter of pod

S : float

Platform area of the pod

L_pod : float

Length of Pod

t_passenger : float

thickness of structure in passenger section.

t_pod : float

thickness of outer pod section.

BF : float

Pod blockage factor.

beta : float

Duct blockage factor.

Notes

Computes to corss sectional area, length, and planform area of the pod based on the sizes of internal components and the necessary duct area within the pod based on compressor peformance. Assumes isentropic compression and a compressor exit mach number of .3. Also calculate blockage factors based on pressurized cylinder equations.

`solve_nonlinear`(*p*, *u*, *r*)

hyperloop.Python.pod.pod_group module

hyperloop.Python.pod.pod_mach module Estimates tube diameter, inlet diameter, and compressor power Will optimize some sort of cost function based on pod mach number Many parameters are currently taken from hyperloop alpha, pod sizing analysis

class `hyperloop.Python.pod.pod_mach.`**`PodMach`**

Bases: `openmdao.core.component.Component`

Params gam : float

Ratio of specific heats. Default value is 1.4

R : float

Ideal gas constant. Default valut is 287 J/(m*K).

A_pod : float

cross sectional area of the pod. Default value is 1.4 m**2. Value will be taken from pod geometry module

comp_inlet_area : float

Inlet area of compressor. (m**2)

L : float

Pod length. Default value is 22 m. Value will be taken from pod geometry module

prc : float

Pressure ratio across compressor inlet and outlet. Default value is 12.5. Value will be taken from NPSS

p_tube : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

T_ambient : float

Tunnel ambient temperature. Default value is 298 K.

mu : float

Fluid dynamic viscosity. Default value is 1.846e-5 kg/(m*s)

M_duct : float

Maximum Mach number allowed in the duct. Default value is .95

M_diff : float

Maximum Mach number allowed at compressor inlet. Default valu is .6

cp : float

Specific heat of fluid. Default value is 1009 J/(kg*K)

M_pod : float

pod Mach number. Default value is .8

Returns **A_tube** : float

will return optimal tunnel area based on pod Mach number

pwr_comp : float

will return the power that needs to be delivered to the flow by the compressor. Does not account for compressor efficiency

A_bypass : float

will return area of that the flow must go through to bypass pod

A_inlet : float

returns area of the inlet necessary to slow the flow down to M_diffuser

A_duct_eff : float

returns effective duct area which accounts for displacement boundary layer thickness approximation

A_diff : float

returns area of diffuser outlet

Re : float

returns free stream Reynolds number

Notes

Uses isentropic mach-area relationships to determine the cross sectional area of the tube to prevent choking and super sonic flow. Takes pod mach number and tunnel pressure from user, then takes pod area and bloackage factor from geometry.s

`solve_nonlinear`(*params*, *unknowns*, *resids*)

hyperloop.Python.pod.pod_mass module

class `hyperloop.Python.pod.pod_mass.`**`PodMass`**

Bases: `openmdao.core.component.Component`

The PodMass Component sums all the mass to find the total mass of the pod

mag_mass [float] Mass of permanent magnets. (kg)

podgeo_d [float] Diameter of pod. (m)

al_rho [float] Density of the aluminium (kg/m**3)

motor_mass [float] Mass of motor (kg)

battery_mass [float] Mass of battery. (kg)

comp_mass [float] Mass of Compressor. (kg)

pod_len [float] Length of pod. (m)

comp_inletArea [float] Area of compressor (m**2)

BF [float] Blockage factor (unitless)

n_passengers [float] Number of passengers

m_per_passenger [float] mass per passenger (kg)

pod_mass [float] Pod Mass (kg)

`solve_nonlinear`(*params*, *unknowns*, *resids*)

Module contents

hyperloop.Python.tools package

Submodules

hyperloop.Python.tools.io_helper module

class `hyperloop.Python.tools.io_helper.`**`InputHelper`**(*file_name*)

Bases: `object`

`get_config`(*member*)

Module contents

hyperloop.Python.tube package

Submodules

hyperloop.Python.tube.propulsion_mechanics module Estimate power requirements for prouplsion sections Many parameters are currently taken from hyperloop alpha Can currently be used for LSM or LIM systems

class `hyperloop.Python.tube.propulsion_mechanics.`**`PropulsionMechanics`**

Bases: `openmdao.core.component.Component`

Params **p_tube** : float

Pressure of air in tube. Default value is 100 Pa. Value will come from vacuum component

R : float

Ideal gas constant. Default valut is 287 J/(m*K).

T_ambient : float

Tunnel ambient temperature. Default value is 298 K.

g : float

Gravitational acceleration. Default value is 9.81 m/s**2

vf : float

Top pod speed after boosting section. Default value is 335 m/s. Value will be taken from aero module

vo : float

Speed of pod when it enters boosting section. Default value is 324 m/s.

m_pod : float

total mass of pod. Default value is 3100 kg. Value will come from weight component

eta : float

Efficiency of propulsion system. Default value is .8. value will come from propulsion module.

Cd : float

Drag coefficient of pod. Default value is .2. More accurate results will come from CFD

S : float

Reference area of the pod. Default value is 1.4 m**2. Value will be pulled from geometry module

D_mag : float

Drag force from magnetic levitation in N. Default value is 150 N. Value will come from levitation analysis

nozzle_thrust : float

Thrust produced by pod compressed air. Default value 21473.92 N. Will pull value from flow_path.py

ram_drag : float

Drag produced by inlet ram pressure. Default value is 7237.6

Returns **pwr_req** : float

Computes power required by accelerating segment

Notes

Calculate power required to accelerate pod in one boosting section assuming linear acceleration of 1g

solve_nonlinear(*params*, *unknowns*, *resids*)

Evaluate function Preq = (1/eta)*(mg*(1+sin(theta))*(vf-vo)+(1/6)*(Cd*rho*S*(vf^3 - vo^3))+D_mag*(vf-v0)) Can be optimized in the future. Friction and magnetic drag are neglected for now.

hyperloop.Python.tube.steady_state_vacuum module

hyperloop.Python.tube.submerged_tube module

class hyperloop.Python.tube.submerged_tube.**SubmergedTube**

Bases: openmdao.core.component.Component

Params **p_tube** : float

Tube pressure. Default valut is 850 Pa

A_tube : float

Cross sectional area of tube. Default valut is 30 m**2

Su : float

Ultimate strength pf tube material. Default valut is 400.0e6 Pa

SF : float

Tube safety factor. Default valut is 5.0

rho_water : float

Density of sea water. Default value is 1025.0e3 kg/m**3

depth : float

Depth of the tube. Default value is 10.0m

g : float

Gravitational acceleration. Default value is 9.81 m/s**2

Pa : float

Ambient pressure at sea level. Default value is 101.3e3 Pa

unit_cost_tube : float

Cost of tube materials per unit mass. Default value is .3307 USD/kg

Returns **t** : float

Returns tube thickness in m

dF_buoyancy : float

Returns buoyant force on tube per unit length in N/m

material cost : float

Returns material cost of tube per unit length in USD/m

m_prime : float

Returns mass of tube per unit length in kg/m

solve_nonlinear (*p*, *u*, *r*)

t = (p*r)/(Su/SF); p = pa + rho*g*h; F_buoyant/L = rho*A_tube*g

hyperloop.Python.tube.tube_and_pylon module

class `hyperloop.Python.tube.tube_and_pylon.`**`TubeAndPylon`**

Bases: `openmdao.core.component.Component`

Params **tube_area** : float

Inner tube radius. Default is 3.8013 m**2

rho_tube : float

Density of tube material. Default is 7820 kg/m**3

E_tube : float

Young's modulus of tube material. Default value is 200e9 Pa

v_tube : float

Poisson's ratio of tube material. Default value is .3

Su_tube : float

Ultimate strength of tube material. Default value is 152e6 Pa

sf : float

Tube safety factor. Default value is 1.5

g : float

Gravitational acceleration. Default value is 9.81 m/s**2

unit_cost_tube : float

Cost of tube material per unit mass. Default value is .33 USD/kg

p_tunnel : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

p_ambient : float

Pressure of atmosphere. Default value is 101.3e3 Pa.

alpha_tube : float

Coefficient of thermal expansion of tube material. Default value is 0.0

dT_tube : float

Difference in tunnel temperature as compared ot a reference temperature. Default value is 0.0

m_pod : float

total mass of pod. Default value is 3100 kg. Value will come from weight component

r : float

Radius of tube. Default value is 1.1 m. Value will come from aero module

t : float

Thickness of the tube. Default value is 50 mm. Value is optimized in problem driver.

rho_pylon : float

Density of pylon material. Default value is 2400 kg/m**3

E_pylon : float

Young's modulus of pylon material. Default value is 41e9 Pa

v_pylon : float

Poisson's ratio of pylon material. Default value is .2

Su_pylon : float

Ultimate strength of pylon material. Default value is 40e6 Pa

unit_cost_pylon : float

Cost of pylon material per unit mass. Default value is .05 USD/kg

h : float

Height of each pylon. Default value is 10 m.

r_pylon : float

Radius of each pylon. Default value is 1 m. Value will be optimized in problem driver

vac_weight : float

Total weight of vacuums. Default value is 1500.0 kg. Value will come from vacuum component

Returns **m_pylon** : float

mass of individual pylon in kg/pylon

m_prime: float

Calculates mass per unit length of tube in kg/m

von_mises : float

Von Mises stress in the tube in Pa

total_material_cost : float

returns total cost of tube and pylon materials per unit distance in USD/m

R : float

Returns vertical component of force on each pylon in N

delta : float

Maximum deflection of tube between pylons in m

dx : float

outputs distance in between pylons in m

t_crit :

Minimum tube thickness to satisfy vacuum tube buckling condition in m

Notes

[1] USA. NASA. Buckling of Thin-Walled Circular Cylinders. N.p.: n.p., n.d. Web. 13 June 2016.

solve_nonlinear (*params*, *unknowns*, *resids*)

total material cost = ($/kg_tunnel)*m_prime + ($/kg_pylon)*m_pylon*(1/dx) m_prime = mass of tunnel per unit length = rho_tube*pi*((r+t)^2-r^2) m_pylon = mass of single pylon = rho_pylon*pi*(r_pylon^2)*h

Constraint equations derived from yield on buckling conditions

hyperloop.Python.tube.tube_group module

hyperloop.Python.tube.tube_power module

class `hyperloop.Python.tube.tube_power.`**`TubePower`**

Bases: `openmdao.core.component.Component`

Computes the total power requirement for all tube components: Vacuum, TubeAndPylon, PropulsionMechanics

Params **vac_power** : float

Power requirement to run vacuum pumps (kW)

vac_energy : float

Energy requirement to run vacuum pumps for 1 day (kJ)

prop_power : float

Power required to accelerate pod to 1G once (W)

num_thrust : float

Number of propulsion thrusts required for trip (unitless)

time_thrust : float

Time required to accelerate pod to 1G (s)

tube_temp : float

Tube temperature (K)

elec_price : float

Cost of electricity per kiloWatt-hour (USD/(kW*h))

Outputs **tot_power** : float

Total power requirement for tube components (kW)

tot_energy : float

Total energy requirement for tube components (kJ)

cost_pwr : float

Cost for tube power requirements (USD)

Notes

The national average for electricity runs $.13 cents per kilowatt hour. Power requirement to cool the tube is not currently calculated in this component. Params to calculate that power in the future are commented out for the meantime.

References

[1] Laughlin, Robert B., Prof. "Energy Information Administration - Electricity Price." EIA. Stanford University, 30 Dec. 2008. Web. 24 June 2016. <http://large.stanford.edu/publications/power/references/voltprice/> Umrath, Walter, Dr. Fundamentals of Vacuum Technology. N.p.: Oerlikon Leybold Vacuum, n.d. Print.

TODO: add in calculations for refrigeration power requirement?

solve_nonlinear (*params*, *unknowns*, *resids*)

hyperloop.Python.tube.tube_vacuum module Current calculation to determine total number of vacuum pumps needed and their respective cost per year. The National average for Electricity runs $.13 cents per kilowatt hour.

class `hyperloop.Python.tube.tube_vacuum.`**`Vacuum`**

Bases: `openmdao.core.component.Component`

Params **pressure_initial** : float

initial Pressure before the pump down . Default value is 760.2.

pressure_final : float

Desired pressure within tube. Default value is 7.0.

speed : float

Pumping speed. Default value is 163333.3.

tube_area : float

Area of the tube. Default value is 5.0.

tube_length : float

Length of the tube. Default value is 5000.0.

pwr : float

Motor rating. Default value is 18.5.

electricity_price : float

Cost of electricity per kilowatt hour. Default value is 0.13.

time_down : float

Desired pump down time. Default value is 300.0.

gamma : float

Operational percentage of the pump per day. Default value is 0.8.

pump_weight : float

Weight of one pump. Default value is 715.0.

Returns **number_pumps** : float

Number of pumps. Default value is 1.0.

cost_annual : float

Total cost of pumps. The cost of purchasing the pumps and running them per year in USD.

weight_tot: float

Total weight of the pumps throughout the track in kg.

pwr_tot: float

Total power of the pumps in kW.

energy_tot: float

Total energy consumed by the pumps in one day in kJ.

References

[1] Laughlin, Robert B., Prof. "Energy Information Administration - Electricity Price." EIA. Stanford University, 30 Dec. 2008. Web. 24 June 2016. <http://large.stanford.edu/publications/power/references/voltprice/> Umrath, Walter, Dr. Fundamentals of Vacuum Technology. N.p.: Oerlikon Leybold Vacuum, n.d. Print.

solve_nonlinear (*params*, *unknowns*, *resids*)

hyperloop.Python.tube.tube_wall_temp module

hyperloop.Python.tube.tunnel_cost module

Module contents

Submodules

hyperloop.Python.LIM module

Model for a Single Sided Linear Induction Motor(SLIM), using Circuit Model.

Evaluates thrust generated by a single, single sided linear induction motor using the simplified circuit model. Inspired from the paper: DESIGN OF A SINGLE SIDED LINEAR INDUCTION MOTOR(SLIM) USING A USER INTERACTIVE COMPUTER PROGRAM

class hyperloop.Python.LIM.**Thrust**

Bases: openmdao.core.component.Component

acceleration (*P1*, *c_time*, *mass*)

Sub-module used to calculate acceleration using following eq. acceleration = (P1 / (2*mass*c_time))^0.5

omega (*f*, *L*)

Sub-module used to calculate omega using following eq. omega = 2*pi*f*L

phase_angle_calc (*f*, *L*, *R1*)

Sub-module used to calculate phase angle using following eq. phi = arctan((2*pi*f*L) / R1)

reactance_of_inductor (*f*, *L*)
Sub-module used to calculate reactance of inductor using following eq. slip = 2*pi*f*L

slip_ratio (*V_s*, *V_r*)
Sub-module used to calculate slip ratio using following eq. slip = (V_s - V_r) / V_s

solve_nonlinear (*params*, *unknowns*, *resids*)

hyperloop.Python.angular_velocity321 module

class hyperloop.Python.angular_velocity321.**AngularVelocity321**
Bases: openmdao.core.component.Component

Params Yaw : float

Yaw angle (3-axis rotation) of body frame with respect to the inertial NED frame. Default value is 0.0 rad

Pitch : float

Pitch angle (2-axis rotation) of body fram with respect to the inertial NED frame. Default value is 0.0 rad

Roll : float

Roll angle (1-axis rotation) of body fram with respect to the inertial NED frame. Default value is 0.0 rad

Yaw rate : float

Yaw rate of pod body frame. Default value is .01 rad/s

Pitch rate : float

Pitch rate of pod body frame. Default value is .01 rad/s

Roll rate : float

Roll rate of pod body frame. Default value is 0.0 rad/s

Returns Angular velocity : float

Returns the body fame angular velocity of the pod in rad/s

Notes

Evaluates the body frame angular velocity from 321 Euler angles and their derivatives Units are in radians and radians/s

solve_nonlinear (*p*, *u*, *r*)

Params Yaw : float

Yaw angle (3-axis rotation) of body frame with respect to the inertial NED frame. Default value is 0.0 rad

Pitch : float

Pitch angle (2-axis rotation) of body fram with respect to the inertial NED frame. Default value is 0.0 rad

Roll : float

Roll angle (1-axis rotation) of body fram with respect to the inertial NED frame. Default value is 0.0 rad

Yaw rate : float

Yaw rate of pod body frame. Default value is .01 rad/s

Pitch rate : float

Pitch rate of pod body frame. Default value is .01 rad/s

Roll rate : float

Roll rate of pod body frame. Default value is 0.0 rad/s

Returns **Angular velocity** : float

Returns the body fame angular velocity of the pod in rad/s

Notes

omega = [[s(psi)*s(theta), c(psi), 0], [c(psi)*s(theta), -s(psi), 0], [c(theta), 0,1]] * [[phi], [theta], [psi]]

hyperloop.Python.boundary_layer_sensitivity module

class `hyperloop.Python.boundary_layer_sensitivity.`**`BoundaryLayerSensitivity`**

Bases: `openmdao.core.component.Component`

Params **gam** : float

Ratio of specific heats. Default value is 1.4

R : float

Ideal gas constant. Default valut is 287 J/(m*K).

A_pod : float

cross sectional area of the pod. Default value is 1.4 m**2. Value will be taken from pod geometry module

comp_inlet_area : float

Inlet area of compressor. (m**2)

L : float

Pod length. Default value is 22 m. Value will be taken from pod geometry module

prc : float

Pressure ratio across compressor inlet and outlet. Default value is 12.5. Value will be taken from NPSS

p_tube : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

T_ambient : float

Tunnel ambient temperature. Default value is 298 K.

mu : float

Fluid dynamic viscosity. Default value is 1.846e-5 kg/(m*s)

M_duct : float

Maximum Mach number allowed in the duct. Default value is .95

M_diff : float

Maximum Mach number allowed at compressor inlet. Default valu is .6

cp : float

Specific heat of fluid. Default value is 1009 J/(kg*K)

M_pod : float

pod Mach number. Default value is .8

length_calc : bool

True calculates boundary layer thickness. False takes boundary layer thickness as an input. Default value is false.

Returns **A_tube** : float

will return optimal tunnel area based on pod Mach number

pwr_comp : float

will return the power that needs to be delivered to the flow by the compressor. Does not account for compressor efficiency

A_bypass : float

will return area of that the flow must go through to bypass pod

A_inlet : float

returns area of the inlet necessary to slow the flow down to M_diffuser

A_duct_eff : float

returns effective duct area which accounts for displacement boundary layer thickness approximation

A_diff : float

returns area of diffuser outlet

Re : float

returns free stream Reynolds number

Notes

Component is not a part of the system model, but is instead intended to analyze the sensitivity of tube area to the bounday layer thickness over the pod. Can be made to calculatee based on Reynolds number accourding to flat plate assumption or to vary boundary layer to account for boundary layer suction or some other version of flow control.

`solve_nonlinear`(*params*, *unknowns*, *resids*)

hyperloop.Python.sample_mission module

class `hyperloop.Python.sample_mission.`**`SampleMission`**

Bases: `openmdao.core.component.Component`

Params **p_tunnel** : float

Tunnel pressure. Default value is 850 Pa.

T_tunnel : float

Tunnel temperature. Default value is 320 K.

Cd : float

Drag coefficient. Default value is .2

m_pod : float

Pod mass. Default value is 10000.0 kg

D_mag : float

magnetic drag. Default value is (10000.0*9.81)/200.0 N

ram_drag : float

Inlet ram drag. Default value is 1855.44 N

nozzle_thrust : float

Gross thrust from the nozzle. Default value is 5503.12 N

M_pod : float

Pod mach number. Default value is .8

g : float

Gravitational acceleration. Default value is 9.81 m/s**2

R : float

Ideal gas constant. Default value is 287 J/kg/K

gam : float

Ratio of specific heats. Dafault value is 1.4

theta : float

Elevation angle of track. Default value is 0.0 rad

track_length : float

Total length of track. Default value is 600.0e3 m

Returns **dx_start** : float

Start up distance in m

dx_boost : float

length of individual booster in m

boost_time : float

Time in individual booster section in s

prop_period : float

distance between propulsion sections in m

num_thrust : float

number of booster sections

coast_time : float

Time coasting between boosters in s

Notes

This component outputs relevant mission parameters assuming a flat trajectore from LA to SF

`solve_nonlinear`(*p*, *u*, *r*)

hyperloop.Python.structural_optimization module

class `hyperloop.Python.structural_optimization.`**`StructuralOptimization`**

Bases: `openmdao.core.component.Component`

Params **tube_area** : float

Inner tube radius. Default is 3.8013 m**2

rho_tube : float

Density of tube material. Default is 7820 kg/m**3

E_tube : float

Young's modulus of tube material. Default value is 200e9 Pa

v_tube : float

Poisson's ratio of tube material. Default value is .3

Su_tube : float

Ultimate strength of tube material. Default value is 152e6 Pa

sf : float

Tube safety factor. Default value is 1.5

g : float

Gravitational acceleration. Default value is 9.81 m/s**2

unit_cost_tube : float

Cost of tube material per unit mass. Default value is .33 USD/kg

p_tunnel : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

p_ambient : float

Pressure of atmosphere. Default value is 101.3e3 Pa.

alpha_tube : float

Coefficient of thermal expansion of tube material. Default value is 0.0

dT_tube : float

Difference in tunnel temperature as compared ot a reference temperature. Default value is 0.0

m_pod : float

total mass of pod. Default value is 3100 kg. Value will come from weight component

r : float

Radius of tube. Default value is 1.1 m. Value will come from aero module

t : float

Thickness of the tube. Default value is 50 mm. Value is optimized in problem driver.

rho_pylon : float

Density of pylon material. Default value is 2400 kg/m**3

E_pylon : float

Young's modulus of pylon material. Default value is 41e9 Pa

v_pylon : float

Poisson's ratio of pylon material. Default value is .2

Su_pylon : float

Ultimate strength of pylon material. Default value is 40e6 Pa

unit_cost_pylon : float

Cost of pylon material per unit mass. Default value is .05 USD/kg

h : float

Height of each pylon. Default value is 10 m.

r_pylon : float

Radius of each pylon. Default value is 1 m. Value will be optimized in problem driver

vac_weight : float

Total weight of vacuums. Default value is 1500.0 kg. Value will come from vacuum component

Returns **m_pylon** : float

mass of individual pylon in kg/pylon

m_prime: float

Calculates mass per unit length of tube in kg/m

von_mises : float

Von Mises stress in the tube in Pa

total_material_cost : float

returns total cost of tube and pylon materials per unit distance in USD/m

R : float

Returns vertical component of force on each pylon in N

delta : float

Maximum deflection of tube between pylons in m

dx : float

outputs distance in between pylons in m

t_crit :

Minimum tube thickness to satisfy vacuum tube buckling condition in m

Notes

[1] USA. NASA. Buckling of Thin-Walled Circular Cylinders. N.p.: n.p., n.d. Web. 13 June 2016.

solve_nonlinear(*params*, *unknowns*, *resids*)

total material cost = ($/kg_tunnel)*m_prime + ($/kg_pylon)*m_pylon*(1/dx) m_prime = mass of tunnel per unit length = rho_tube*pi*((r+t)^2-r^2) m_pylon = mass of single pylon = rho_pylon*pi*(r_pylon^2)*h

Constraint equations derived from yield on buckling conditions

hyperloop.Python.ticket_cost module

class hyperloop.Python.ticket_cost.**TicketCost**

Bases: openmdao.core.component.Component

Params length_cost : float

Cost of materials per unit length. Default value is 2.437e6 USD/km

pod_cost : float

Cost per individual pod. Default value is 1.0e6 USD.

capital_cost : float

Estimate of overhead capital cost. Default value is 1.0e10 USD.

energy_cost : float

Cost of electricity. Default value is .13 USD/kWh

ib : float

Bond interest rate. Default value is .04

bm : float

Bond maturity. Default value is 20.0 years.

operating_time : float

operating time per day. Default value is 16.0*3600 s

JtokWh : float

Convert J to kWh. Default value is J/kWh

m_pod : float

Pod mass. Default value is 3100 kg

n_passengers : float

Number of passengers. Default value is 28.0

pod_period : float

Time in between pod departures. Default value is 120.0 s

avg_speed : float

average pod speed. Default value is 286.86 m/s

track_length : float

length of the track. Default value is 600e3 m

pod_power : float

Power consumption of the pod. Default value is 1.5e6 W

prop_power : float

power of an individual propulsion section. Default value is 350e3 W

vac_power : float

Power of the vacuum pumps. Default value is 71.049e6 W

alpha : float

percent of vacuum power used in steady state. Default value is .0001

vf : float

Pod top speed. Default value is 286.86 m/s

g : float

Gravity. Default value is 9.81 m/s/s

Cd : float

Pod drag coefficient. Default value is .2

S : float

Pod planform area. Default value is 40.42 m**2

p_tunnel : float

Tunnel pressure. Default value is 850.0 Pa

T_tunnel : float

Tunnel temperature. Default value is 320 K

R : float

Ideal gas constant. Default value is 287 J/kg/K

eta : float

Efficiency of propulsion system

D_mag : float

Magnetic drag. Default value is (9.81*3100.0)/200.0 N

thrust_time : float

Time spent during a propulsive section. Default value is 1.5 s

prop_period : float

distance between pripulsion sections. Defualt value is 25.0e3 km

Returns **ticket_cost** : float

cost of individual ticket. Default value is 0.0 USD

prop_energy_cost : float

cost of energy used by propulsion section per year. Default value is 0.0 USD

Notes

This Component takes into account various cost figures from the system model and combines them to estimate tickt cost per passenger.

solve_nonlinear (*p*, *u*, *r*)

hyperloop.Python.tube_and_pod module

hyperloop.Python.underwater_optimization module

class `hyperloop.Python.underwater_optimization.`**`UnderwaterOptimization`**

Bases: `openmdao.core.component.Component`

Params **tube_area** : float

Inner tube radius. Default is 3.8013 m**2

rho_tube : float

Density of tube material. Default is 7820 kg/m**3

E_tube : float

Young's modulus of tube material. Default value is 200e9 Pa

v_tube : float

Poisson's ratio of tube material. Default value is .3

Su_tube : float

Ultimate strength of tube material. Default value is 152e6 Pa

sf : float

Tube safety factor. Default value is 1.5

g : float

Gravitational acceleration. Default value is 9.81 m/s**2

unit_cost_tube : float

Cost of tube material per unit mass. Default value is .33 USD/kg

p_tunnel : float

Pressure of air in tube. Default value is 850 Pa. Value will come from vacuum component

p_ambient : float

Pressure of atmosphere. Default value is 101.3e3 Pa.

alpha_tube : float

Coefficient of thermal expansion of tube material. Default value is 0.0

dT_tube : float

Difference in tunnel temperature as compared ot a reference temperature. Default value is 0.0

m_pod : float

total mass of pod. Default value is 3100 kg. Value will come from weight component

r : float

Radius of tube. Default value is 1.1 m. Value will come from aero module

t : float

Thickness of the tube. Default value is 50 mm. Value is optimized in problem driver.

rho_pylon : float

Density of pylon material. Default value is 2400 kg/m**3

E_pylon : float

Young's modulus of pylon material. Default value is 41e9 Pa

v_pylon : float

Poisson's ratio of pylon material. Default value is .2

Su_pylon : float

Ultimate strength of pylon material. Default value is 40e6 Pa

unit_cost_pylon : float

Cost of pylon material per unit mass. Default value is .05 USD/kg

h : float

Height of each pylon. Default value is 10 m.

r_pylon : float

Radius of each pylon. Default value is 1 m. Value will be optimized in problem driver

vac_weight : float

Total weight of vacuums. Default value is 1500.0 kg. Value will come from vacuum component

Returns **m_pylon** : float

mass of individual pylon in kg/pylon

m_prime: float

Calculates mass per unit length of tube in kg/m

von_mises : float

Von Mises stress in the tube in Pa

total_material_cost : float

returns total cost of tube and pylon materials per unit distance in USD/m

R : float

Returns vertical component of force on each pylon in N

delta : float

Maximum deflection of tube between pylons in m

dx : float

outputs distance in between pylons in m

t_crit :

Minimum tube thickness to satisfy vacuum tube buckling condition in m

Notes

[1] USA. NASA. Buckling of Thin-Walled Circular Cylinders. N.p.: n.p., n.d. Web. 13 June 2016.

`solve_nonlinear` (*params*, *unknowns*, *resids*)

total material cost = ($/kg_tunnel)*m_prime + ($/kg_pylon)*m_pylon*(1/dx) m_prime = mass of tunnel per unit length = rho_tube*pi*((r+t)^2-r^2) m_pylon = mass of single pylon = rho_pylon*pi*(r_pylon^2)*h

Constraint equations derived from yield on buckling conditions

Module contents

1.2 Module contents

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

[R1] Michael Tong Correlation used.

[R2] NASA-Glenn NPSS compressor cycle model.

[R3] Michael Tong Correlation used.

[R4] NASA-Glenn NPSS compressor cycle model.

[R5] Gladin, Ali, Collins, "Conceptual Modeling of Electric and Hybrid-Electric Propulsion for UAS Applications" Georgia Tech, 2015

[R6] 4. (a) Mavris, "Subsonic Ultra Green Aircraft Research - Phase II," NASA Langley Research Center, 2014

[R7] eciaauthorized.com/search/HHR650D

[R8] Gladin, Ali, Collins, "Conceptual Modeling of Electric and Hybrid-Electric Propulsion for UAS Applications" Georgia Tech, 2015

[R9] "J. Gladin, K. Ali, K. Collins, "Conceptual Modeling of Electric and Hybrid-Electric Propulsion for UAS Applications," Georgia Tech, 2015.

[R10] "J. Gladin, K. Ali, K. Collins, "Conceptual Modeling of Electric and Hybrid-Electric Propulsion for UAS Applications," Georgia Tech, 2015.

[R11] Gladin, Ali, Collins, "Conceptual Modeling of Electric and Hybrid-Electric Propulsion for UAS Applications" Georgia Tech, 2015

[R12] Friend, Paul. Magnetic Levitation Train Technology 1. Thesis. Bradley University, 2004. N.p.: n.p., n.d. Print.

h

A

B

C

D

F

G

H

I

L

M

O

P

R

S

T

U

V