2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2147]:

using Plots using CSV using DataFrames **using Interpolations**

In [2148]:

*# Import battery performance data (taken from graphs and generated by Engauge)* **csv\_diam**ond\_05C = CSV.File*(*"Foxtech Diamond/Diamond cell test, 0.5C.csv") df\_diamond\_05C = DataFrame(csv\_diamond\_05C) itp\_diamond\_05C = LinearInterpolation(df\_diamond\_05C.Ahr, df\_diamond\_05C.V, extrapolation\_b

**csv\_diam**ond\_3C = CSV. File("Foxtech Diamond/Diamond cell test, 3C.csv") df\_diamond\_3C = DataFrame(csv\_diamond\_3C) itp\_diamond\_3C = LinearInterpolation(df\_diamond\_3C.Ahr, df\_diamond\_3C.V, extrapolation\_bc=L

**csv\_dia**mond\_5C = CSV.File*(*"Foxtech Diamond*/*Diamond cell test, 5C.csv") df\_diamond\_5C = DataFrame(csv\_diamond\_5C) itp\_diamond\_5C = LinearInterpolation*(*df\_diamond\_5C.Ahr, df\_diamond\_5C.V, **extrapolat**ion\_bc=L

*## IMPORTANT TO UPDATE WHENEVER THE BATTERY D*A*TASHEETS ARE CHANGED ## The order is important, it must be done in sequential order by growing discharge rate* **discharge\_curves\_diam**ond = *(*

(0.5, itp\_diamond\_05C), *(*3.0, itp\_diamond\_3C), *(*5.0, itp\_diamond\_5C)

*# Capacity of cells in datasheet, in [Ahr]* **ahr\_datasheet\_diamond\_cell** = 22

*# Individual cell max discharge rate* c\_diamond\_max = 5.5

Out[2148]:

5.5

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

*1/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2149]:

*# Import battery performance data (taken from graphs and generated by Engauge)* **csv\_lipo\_O\_84C = CSV**. File*(*"Molicel/P4*2*A*/*P42A, O\_84C.csv") **df\_lipo\_0\_84C** = DataFrame (c**sv\_lipo\_O\_84C)** itp\_lipo\_0\_84C = LinearInterpolation(df\_lipo\_0\_84C.Ahr, df\_lipo\_O**\_840.V, ext**rapolation\_bc=L

csv\_lipo\_4\_2C = CSV.File("Molicel/P42A*/*P42A, 4\_20.csv") df\_lipo\_4\_2C = DataFrame(csv\_lipo\_4\_20) itp\_lipo\_4\_2C = LinearInterpolation(df\_lipo\_4\_2C.Ahr, df\_lipo\_4\_2C.V, **extrapolat**ion\_bc=Line

csv\_lipo\_10C = CSV.File*(*"Molicel/P*42*A/P42A, 10C.csv") df\_lipo\_10C = DataFrame(csv\_lipo\_10C) itp\_lipo\_10C = LinearInterpolation(df\_lipo\_10C.Ahr, df\_li**po\_10C.V, extrapolat**ion\_bc=Line*(*))

csv\_lipo\_20C = CSV.File*(*"Molicel/P*42*A/P42A, 200.csv") df\_lipo\_20C = DataFrame(csv\_lipo\_200) itp\_lipo\_20C = LinearInterpolation(df\_lipo\_20C. Ahr, df\_lipo\_28C.V, **extrapolati**on\_bc=Line())

csv\_lipo\_30C = CSV.File*(*"Molicel*/*P*42A*/P*42A*, 300.csv") df\_lipo\_30C = DataFrame(csv\_lipo\_30C)

itp\_lipo\_30C = LinearInterpolation(df\_lipo\_30C.Ahr, df\_lipo**\_30C.V, ext**rapolation\_bc=Line*(*))

*## IMPORTANT TO UPDATE WHENEVER THE BATTERY DATASHEETS* A*RE CHANGED ## The order is important, it must be done in sequential order by growing discharge rate* **discharge\_curves\_**P42A = (

*(*0.84, itp\_lipo\_0\_84C), *(*4.2, itp\_lipo\_4\_20), *(*10.0, itp\_lipo\_100), *(*20.0, itp\_lipo\_200), *(*30.0, itp\_lipo\_30C)

*# Capacity of cells in datasheet, in [Ahr]* **ahr\_datasheet\_P42A\_cel**l = 4.2

*# Individual cell mass,* i*n [kg]* **mass\_P42A\_cell = 0.070**

*# Individual cell max continuous discharge rate. This is taken from testing at https://www.* C\_P42A\_max = 45*/*ahr\_**datasheet\_P42A\_cell**

*# Individual cell nominal energy capacity, in [kwhr]* **kwhr\_P42A\_cell = .0147**

Out[2149]:

**0.0147**

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

2*/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2150]:

*# Import battery performance data (taken from graphs and generated by Engauge)* **csv\_lipo\_0\_84C = CSV**. File*(*"Molicel/P26A/P26A,Q\_520.csv") **df\_lipo\_O\_84C** = DataFrame(csv\_lipo**\_O\_84C)** itp\_lipo\_0\_84C = LinearInterpolation(df\_lipo\_0\_84C.Ahr, df\_lipo\_O**\_840.V, ext**rapolation\_bc=L

csv\_lipo\_10C = CSV. File("Molicel*/*P26A/P26A, 10C.csv") df\_lipo\_10C = DataFrame(csv\_lipo\_10C) itp\_lipo\_10C = LinearInterpolation(df\_lipo\_10C.Ahr, df\_lipo\_10C.V, extrapolation\_bc=Line())

**csv\_lipo\_12\_6C = CSV.** File*(*"Molicel/P*26*A/P26A, 12\_6C.csv") df\_lipo\_12\_6C = DataFrame (csv\_lipo\_12\_6C) itp\_lipo\_12\_6C= LinearInterpolation(df\_lipo\_12\_6C.Ahr, df\_lipo\_12\_6C.V, extrapolation\_bc=Li

csv\_lipo\_20C = CSV. File*(*"Molicel/P26A/P26A*, 2*00.csv") df\_lipo\_20C = DataFrame(csv\_lipo\_200) itp\_lipo\_20C = LinearInterpolation(df\_lipo\_20C.Ahr, df\_lipo\_20C.V, extrapolation\_bc=Line())

*## IMPORTANT TO UPDATE WHENEVER THE BATTERY D*A*TASHEETS ARE CHANGED ## The order is important, it must be done in sequential order by growing discharge rate* **discharge\_curves\_P**26C=(

(0.84, itp\_lipo\_0\_84C), *(*4.2, itp\_lipo\_4\_2C), *(*10.0, itp\_lipo\_10C), *(2*0.0, itp\_lipo\_200), *(*30.0, itp\_lipo\_30C)

*# Capacity of cells in datasheet, in [Ahr]* **ahr\_datash**eet\_P26\_cell = *2*.6

*# Individual cell mass,* i*n [kg]* **mass\_P26\_cell** = 0.05

*# Individual cell max discharge rate* C\_P26\_max = 35/ahr\_**datasheet\_P26\_cell**

*# Individual cell nominal energy capacity, in [kwhr]* kwhr\_P26\_cell = .095

Out[2150]:

**0.095**

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

*3/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2151]:

*# Import battery performance data (taken from graphs and generated by Engauge) # A1*23Sy*stems cells. Probably shouldn't be changed* **csv\_a123**\_1C = CSV.File*(*"A123Systems*/*ANR26650M1B, 1C.csv") **df\_a123**\_1C = DataFrame(cs**v\_a123\_10)** itp\_a123\_1C = LinearInterpolation(df\_a123\_1C.Ahr, df\_a123\_1C.**V, extrapolati**on\_bc=Line())

**csv\_a123\_**6C = CSV.File*(*"A123Systems*/*ANR26650M1B, 6C.csv") df\_a123\_6C = DataFrame(csv\_a123\_6C) itp\_a123\_6C = LinearInterpolation(df\_a123\_6C.Ahr, df\_a123\_6C.V, extrapolation\_bc=Line())

**csv\_a123\_20**C = CSV.File*(*"A123Systems/ANR26650M1B, 200.csv") df\_a123\_20C = DataFrame(csv\_a123\_200) itp\_a123\_20C = LinearInterpolation(df\_a123\_20C.Ahr, df\_a**123\_20C.V, extrapol**ation\_bc=Line*(*))

*## IMPORT*AN*T TO UPDATE WHENEVER THE BATTERIES DATASHEETS ARE CHANGED ## The order is important, it must be done in sequential order by growing discharge rate* **discharge\_curves\_a123 = *(***

*(*1.0, itp\_a123\_10), (6.0, itp\_a123\_6C), *(*20.0, itp\_a123\_200)

*# Individual cell capacity, in [Ahr]* ahr\_a123\_cell = 2.65

*# Individual cell mass, in [kg]* **mass\_a123\_cell** = .076

*# Individual cell max discharge rate* **c\_a123\_max = 50**

*# Individual cell nominal energy capacity, in [kwhr]* **kwh**r\_a123\_cell = .00825

**discharge\_index\_vector\_a123 = getfield. (discharge\_curves\_a123, 1)**

Out[2151]: *(*1.0, 6.0, *2*0*.*0)

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

**4*/*2*0***

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2152]:

*## Function definitions*

*# Finds the indices in A which bracket the argument, X. This assumes that A is an ordered s* **function findNearestBr**acket(A,x)

*# find the index which is close*s*t to the value* **tmpIdx** = argmin(abs.(A .- x))

*# Find the pair of indices which bracket the value* if tmpIdx == 1

**lowIdx = 1** highIdx = 2 elseif tmpIdx == length*(*A)

lowIdx = length*(A)*-1

highIdx = length*(*A) **else**

if A[tm**pIdx] < x**

lowIdx = tmpIdx highIdx = tmpIdx+1 **else**

lowIdx = tmpIdx-1 highIdx = tmpIdx **end end**

return [lowIdx, highIdx] **end**

*# Calculate by interpolation the instantaneous pack voltage. It takes into account the disc* **function instantaneous\_pack\_voltage (discharge\_vector, discharge\_curves, c\_battery, ahr\_accu**

idx = findNearestBracket(discharg**e\_vector, c\_battery) lowC = discharge\_ve**ctor[idx[1]] **high = discharge\_vec**tor[idx[2]]

***v*\_instantaneous\_lowC = discharge\_curves** [idx[1]][2] **v\_instantaneous\_hig**hc = discharg**e\_curve**s[idx[2]][2] **v\_instantaneous** = batt\_v(c\_battery, lowc, highc, v\_instantaneous\_lowc(ahr\_accum\_cell),

**X**

**return v\_instantaneous end**

*# Linear interpolation of the batt*er*y voltage, with discharge rate as the independent varia* **function batt\_v(x, x1**, x2, y1, y*2)*

**m = (y2-y1)*/(*x2-x1)**

y = m\*(x-1) + y1 **end**

Out[2152]:

**batt\_v *(g*eneri**c function with 1 method)

**User tunable parameters**

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

*5/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2153]:

*# pack = "Diamond"* pack = "Molicel P42A" *#pack = "Molicel P26A"*

*# Number of parallel cells in pack* num\_cols\_a123 =1

*# Number of series rows in pack* **num\_rows\_a123 -19**

if pack == "Molicel P42A"

*# Number of parallel cells in pack* num\_cols\_lipo = 10 *# Number of series rows* i*n pack* num\_rows\_lipo =12

elseif pack == "Molicel P26A"

*# Number of parallel cells in pack*

num\_cols\_lipo = 10

*# Number of series rows in pack* num\_rows\_lipo = 22

**elseif pa**ck == "Diamond"

ahr\_diamond\_cell = 16

*# Can only equal 16,* 22*, or 30*

*# Number of parallel cells in pack* num\_cols\_lipo = 1

*# Number of series rows in pack* **num\_rows**\_lipo = 12

**en*d***

Out[2153]:

**12**

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

6/20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2154]:

if pack == "Molicel P42A"

**discharge\_curves\_lipo = discharge\_curves\_P42A; mass\_lipo\_cell = mass\_P42A\_cell**

**ahr\_datasheet\_lipo\_cell = ahr\_datasheet\_P42A\_cell** ahr\_lipo\_cell = **ahr\_datasheet\_P42A\_cell** kwhr\_lipo\_cell = kwhr\_P42A\_cell

c\_lipo\_max = C\_P42A\_max elseif pack == "Molicel P26A"

**discharge\_curves\_lipo = discharge\_curves\_P26A*;* mass\_lipo\_cell = mass\_P26A\_cell ahr\_datasheet\_lipo\_cell = ahr\_datasheet\_P26A\_cell** ahr\_lipo\_cell = **ahr\_datasheet\_P26A\_cell** kwhr\_lipo\_cell = kwhr\_P26A\_cell

**c\_lipo\_max = c\_P26A\_max elseif pack** == "Diamond"

**mass\_diam**ond\_cell = (1.500*/*6) \* (ahr\_diamond\_cell/16) *# Backed out from the Foxtech webp*

*# Individual cell nominal energy capacity, in [kwhr]* kwhr\_diamond\_cell = .111 \* (ahr\_diamond\_cell / ahr**\_datasheet\_diamond\_cell)**

**discharge\_curves\_lipo = discharge\_curves\_diamond; mass\_lipo\_cell = mass\_diamond\_cell**

**ahr\_datasheet\_lipo\_cell = ahr\_datasheet\_diamond\_cell** ahr\_lipo\_cell = ahr\_**datasheet\_diamond\_cell kwhr\_lipo\_cell = kwhr\_diamond\_cell**

c\_lipo\_max = c\_diamond\_max **end**

discharge\_index\_*v*ector\_lipo = getfield. (discharge\_curves\_lipo, 1)

**Urves**

Out[2154]:

**(0.84**, 4.2, 10.0, *2*0*.*0, 30.0*)*

In [2155]:

*# Airplane MTOW* **airplaneMass = 300 climbVelocity** = 55\*.511 **glideratio = 20**

Out[2155]:

**20**

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

7*/2*0

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2156]: *# Diode forward voltage drop # https://www.vishay.com/docs/93804/vs-175bqg030hf4.pdf* v\_drop\_diode = 0.782

**D**

*# Pack nominal energy capacity* **kwhr\_a123\_pack = kwhr\_a123\_cel**l \* num\_cols\_a123 \* num\_r**ows\_a123** kwhr\_lipo\_pack = kwhr\_lipo\_cell \* num\_cols\_lipo \* num\_rows\_lipo

*# Pack amps parallel capacity* **ahr\_a123\_pack = ahr\_a123\_cell \* num\_cols\_a123** ahr\_lipo\_pack = ahr\_lipo\_cell \* num\_c**ols\_lipo**

*# Pack cell mass* **mass\_a123\_pack = mass\_a123\_cell \* num\_cols\_a123 \* num\_rows\_a123 mass\_lipo\_pack = mass\_lipo\_cel**l \* num\_cols\_lipo \* num\_rows\_lipo

**pack\_A\_label** = **pack\_B\_label** =

string("A123 LiFePo4 (", num\_**rows\_a123**, "S", num\_cols\_a123, "P)") string("Li-ion (", num\_**rows\_lipo**, "S, ", num\_cols\_lipo, "P,", round(ahr\_lipo

println*(*"mass [kg]: " \* string(round.([m**ass\_a123\_pack mas**s\_lipo\_pack], digits = 1))) println*(*"capacity [Ahr */* kwHr]: " \* string(round.([ahr\_a123\_pack kwhr\_a123\_pack; ahr\_lipo

**mass** [kg]: [1.4 8.4] **capa**city [Ahr / kwHr]: [2.6 0.2; 42.0 1.8]

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

8*/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2157]:

*# Simulate across time* t = 0 delT = .05

*# Power* s*chedule* P\_TAKEOFF = 18e3 **P\_MAX\_CONTINUOUS = 12e3 P\_SUSTAIN\_CLIMB = 10e3** P\_SUSTAIN = 6e3

*# Plotting variables* **t\_exper**iment = Float64[] I\_motor = Float64[] **V\_a123** = Float64[] V\_lipo = Float64[] V\_bus = Float64[] **soc\_a123\_experiment** = Float64[] soc\_lipo\_experiment = Float64[] c\_a123\_experiment = Float64[] c\_lipo\_experiment = Float64[] **P\_experiment** = Float64[] P\_a123 = Float64[] P\_lipo = Float64[] E\_a123 = Float64[] E\_lipo = Float64[] Soc\_20\_lipo = 0 **soc\_5\_a123 = 0** vel\_airplane = [] **hgt\_airplan**e = [] hgt\_dot\_airplane = []

c\_a123 = 0 c\_lipo = 0

*## Initial conditions # Electrical power Load* **P\_elec = 0**

*# Start off at full charge* Soc\_a123 = 1 soc\_lipo = 1

**v\_0\_a123** = itp\_a123\_20C(0) **V\_O\_lipo = discharge\_curves**\_lipo[1][2](0)

**kwhr\_accum\_a123 = 0** kwhr\_accum\_lipo = 0

**ahr\_accum\_a123 = 0** ahr\_accum\_lipo = 0

**ahr\_accum\_cell\_a123 = ahr\_accum\_a123/num\_cols\_a123** ahr\_accum\_cell\_lipo = ahr\_accum\_lipo/num\_cols\_lipo

kineticEnergy = 0 height = 0 velocity = sqrt(2 \* kineticEnergy */* **airplaneMass)**

**localhost:8888*/*notebooks*/*SEN**IOR DESIGN SIMULATION ALL*/*1*1/*11\_*9/*Hybri**d battery analys**is Final.ipynb

**9*/2*0**

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

**v\_instantaneous\_a123 = v\_0\_a123 v\_instantaneous\_lipo = v\_e\_lipo**

**voltage\_a123\_pack = v\_instantaneous\_a123 \* num\_rows\_a123 voltage\_l**ipo\_pack = (v\_instantaneous\_lipo \* num\_rows\_lipo) - V\_drop\_diode

if abs(voltage\_lipo\_p**ack - voltage\_a123**\_pack) <= .001

**V\_system = voltage\_a123\_pack else**

V\_system = max(voltage\_a123\_pack, *v*oltage\_lipo\_pack ) **end**

I\_a123 = 0 I\_lipo = 0

*# Low-pass variable for changing the power load* alpha = 0.05 P\_setpoint = 0

*# Run the simulation* for i=1:(2000*/*delt)

**t = t + delt**

**if 1==**

*# Reduce power according to battery limitations* if c\_lipo > c\_lipo\_max

**P\_elec = P\_elec - 1000 end else**

*# Reduce power according to schedule and/or battery limitations* if (t<0.1)

**P\_setpoint = 0 els**eif (t<=60)

**P\_setpo**int = P\_TAKEOFF **elseif (t > 60) && P\_setpoint** == P\_TAKEOFF

P\_setpoint = P\_MAX\_CONTINUOUS elseif (t > 180) && P\_setpoint == P\_MAX\_CONTINUOUS

P\_setpoint = P\_SUSTAIN\_CLIMB elseif (t > 300) && P\_setpoint == P\_SUSTAIN\_CLIMB

**P\_setpoint = P\_SUSTAIN end**

**P\_elec = P\_elec + alpha\* (P\_setpoint-P\_elec) end**

*# Loop enough times to be sure to converge. Convergence is pretty decent after 15 Loops,* **if p\_elec == 0**

**I\_a123 = 0;** I\_lipo = 0;

**C\_a123 = 0;** c\_lipo = 0;

**v\_instantaneous\_a123** = instantaneous\_pack\_voltage(discharge\_i**ndex\_vector\_a123, discharg v\_instantaneous\_li**po = instantaneous\_pack\_voltage (discharge\_**index\_vector\_a1*2*3, discharg**

*# Saturate battery voltages* **if v\_instantaneous\_li**po > 4.2

**v\_instantaneous\_lip**o = 4.2

**e*nd***

**localhost:8888*/*notebooks*/*SEN**IOR DESIGN SIMULATION ALL*/*1*1/*11\_*9/*Hybri**d battery analys**is Final.ipynb

10*/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

**if v\_instantaneous\_a123 > 3.4**

**v\_instantaneous\_a123** = 3.4 **end**

**voltage\_a123\_pack = v\_instantaneous\_a123 \* num\_rows\_a123 voltage\_lipo\_pack = (v\_instantaneous\_lipo** \* num\_rows\_lipo) - v\_drop\_diode

**P\_split\_pct** = 0.5; **else**

*# Initialize some variables before going into the convergence Loop* loopCount = 1 P\_split\_pct = 0 seekScalar = .5 lastRoundLipoHigh = true

**voltage\_a123\_pack = *v*\_@\_a123 \* num\_rows\_a123 voltage\_lipo\_pack** = (v\_0\_lipo \* num\_ro**ws\_lipo) - v\_drop\_diode**

while (loopCount < 100 && abs(v\_**instantaneous\_a123-v\_instantaneous**\_lipo) >= 0.001 )

I\_a123 = (P\_elec \* (1-P\_split\_pct)) */* v**oltage\_a123\_pack** I\_lipo = (P\_elec \* P\_split\_pct) / voltage\_lipo\_pack

*# Calculate cell discharge rate* c\_a123 = I\_a123 , ahr\_a123\_pack c\_lipo = I\_lipo *,* **ahr\_lipo\_pack**

*## Determine* A123 *cell voltages* **v\_instantaneous\_a12**3 = instantaneous\_pack\_voltage (discharg**e\_index\_vector\_a123, discha**

*## Determine LiPo cell voltages* **v\_instantaneous\_li**po = instantaneous\_pack\_voltage **(discharge\_index\_vector\_lipo, discha**

*# Saturate battery voltages* **if v\_instantaneous**\_lipo > 4.2

**v\_instantaneou**s\_lipo = 4.2 **end**

**if v\_instantaneous\_a123 > 3.4**

**v\_instantaneous\_a123 = 3.4 end**

*# Determine pack voltages. This* i*s just the number of* ***batteries in ser*i*es times the v* voltage\_a123\_pack = v\_instantaneous\_a123 \* num\_rows\_a123 voltage\_lipo\_pack = (v\_instantaneous\_lipo \* num\_rows\_lipo) - v\_drop\_diode**

*# Nonlinear search pattern. We're looking for the percentage power split which brings* **if voltage\_lipo\_pack > voltage\_a123\_pack**

if *(*lastRoundLipoHigh == false)

**seekScalar = seekScala**r/2

**end**

P\_split\_pct = P\_split\_pc**t + seekScalar**

**las**tRoundLipoHigh = true **else**

if *(*lastRoundLipoHigh == true)

**seekScalar = seekScalar**/2 **end**

P\_split\_pct = P\_split\_pct - **seekScalar**

localhost:8888*/*notebooks*/*SENIOR DESIGN SIMULATION ALL/1*1/*11\_9/Hybrid b**attery ana**lysis Final.ipynb

11/20

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

**lastR**oundLipoHigh = false **end**

if p\_split\_pct > 1

P\_split\_pct =1 elseif p\_split\_pct < 0

P\_split\_pct = 0 **end**

*# Increment loop variables.*

loopCount = loopCount + 1 **end**

*# Check if voltages successfully converged. Do this check only when there* i*s power bein* if abs(voltage\_lipo**\_pack - voltage\_a123\_pack) > 0.1 && c\_lipo > 0.1 && c\_a12**3 > 0.1

println*(*"Voltages did not converge at t=" \* string(round(t, digits = 2)) \* ", voltage

**break; end**

**end**

*# Update the* sys*tem voltage* if abs(voltage\_lipo\_pack - **voltage\_a123\_pack) <= .001**

**V\_system = voltage\_a123\_pack else**

*# This is the case where the* A*1*23S*ystems pack hasn't discharged down to the Lipo/Li-ion*

**V\_system = max(voltage\_a123\_pack, voltage\_lipo\_pack ) end**

*# Calculate accumulated amp-hour consumption* ahr\_accum\_a123 = ahr\_accum\_a123 + I\_a123 \* delt*/*3600 **ahr\_accum\_lipo = ahr\_accu**m\_lipo + I\_lipo \* delt*/*3600

*# Calculate accumulated discharge from individual cells* **ahr\_accum\_cell\_a123 = ahr\_accum\_a123**/num\_co**ls\_a123** ahr\_accum\_cell\_lipo = ahr\_accum\_lipo/num\_cols\_lipo

*# Calculate instantaneous power* **P\_instantaneous\_a123 = I\_a123 \* V\_system P\_instantaneous**\_lipo = I\_lipo \* V**\_system**

*# Calculate accumulated kw-hour consumption* **kwhr\_**accum\_a123 = kwhr\_accum\_a123 + P**\_instantaneous\_a123 \* del**T*/*360*0/*1000 **kwhr\_ac**cum\_lipo = kwhr\_accum\_lipo + p\_**instantaneou**s\_lipo \* delt/3600/1000

*# Calculate* S*tate of Charge* **soc\_a123** = 1 - ahr\_accum\_a123/ah**r\_a123\_pack** soc\_lipo = 1 - ahr\_accum\_lipo/ahr\_lipo\_pack

*# Stop the experiment if a pack is dead* if (ahr\_accum\_a**123 > ahr\_a123\_pack &&** (num\_cols\_a123 \* nu**m\_rows\_a12**3 > 0)) || ahr\_accum\_1

println("[BATTERY EMPTY] t: " \* string(round(t, digits = 1)) \* ", A123: " \* string(ro **break**

***end***

*# Mark the point in time when the A1*23S*ystems pack's SoC goes below 5*% **if soc\_5\_a123 == 0 && soc\_a123 < 0.05**

soc\_5\_a123 = t **end**

*# Mark the point in time when the Lipo/Li-ion pack's SoC goes below* 20%

**localhost:8888*/*notebooks*/*SEN**IOR DESIGN SIMULATION ALL*/*11*/*11\_9/Hybrid battery analysis Final.ipynb

**12*/*20**

202*1/11/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook** if soc\_20\_lipo == 0 && soc\_lipo < 0.20

soc\_20\_lipo = t **end**

*# End the simulation one minute after the Lipo/Li-ion* if soc\_20\_lipo !=0 && (t-soc\_20\_lipo) > 60

**break end**

*# Simulate airplane. This is an extremely simplified model, but it works fine for these p* **drag = airplaneMass** \* 9.805/glide Ratio \* (velocity*/*climbVelocity)^2 if *(v*elocity < climbVelocity)

*# Below climbVeloci*ty*, we haevn't taken off* y*et* **eff** = .4 kineticEnergy = kineticEnergy + (P\_elec \* eff - velocity \* drag) \* delt **velocit**y = sqrt(2 \* kineticEnergy */* airpl**aneMass)**

h\_dot = 0 **else**

*# Above climbVelocity, we're putting excess energy into climbing* **eff** = .8\*.90 *# Propller efficiency times electrical efficiency* h\_dot = (P\_elec \* eff - velocity \* drag)*/(*airplaneMass\*9.805) height = height + h\_dot\*delt **end**

*# Some useful console spe*w **if 1==0**

println*(*

t, ": ",

" [",

round(v\_i**nstantaneous\_a123**, digits=3), " "*,* round(voltage\_a123\_pack, digits=2), " ", round(C\_a123, digits=1), ",", round(ahr\_accum\_a123, digits=4), ",", round**(soc\_a123\*100, digits=1),** "], [", round(v\_instantaneous\_lipo, digits=3), "*"* round(voltage\_lipo\_pack, digits=2), ", ", round(c\_lipo, digits=1), ", ", round(ahr\_accum\_lipo, digits=4), ",",

round(soc\_lipo\*100, digits=1), "], :", round(**voltage\_a123\_pack\*I\_a123 + voltage\_lip**o\_pack\*I\_lipo)) **end**

append!(t\_experiment, t) append! (c\_**a123\_experiment, c\_a123)** append! (c\_lip**o\_experiment, c\_lipo)** append! **(soc\_a123\_experiment, soc\_a123)** append! (soc\_lipo\_experiment, soc\_lipo) append! (V\_**a123, voltage\_a123\_pack)** append! (V\_lipo, voltage\_lipo\_pack) append! (V\_**bus, V\_system)** append! (P\_experiment, P\_elec) append! (P\_a**123, p\_instantaneous\_a123)** append! (P\_lipo, p**\_instantaneous\_lipo)** append! (E**\_a123, kwhr\_accum\_a123)** append! (E\_lipo, kwhr\_accum\_lipo) append!(**vel\_airplane, velocity)**

localhost:8888/notebooks/SENIOR DESIGN SIMULATION ALL*/*11/11\_9/Hybrid **battery analysis** Final.ipynb

**13/20**

**Hybrid battery analys**is Final - Jupy**ter Notebook**

202*1/11/*15 E41:46

append! (hgt\_airplane, height) append! (hgt\_dot\_airplane, h\_dot)

**end**

[soc\_5\_a123, soc\_20\_lipo]

Out[2157]:

2-element Vector{Float64}:

**115.2499999999956 633,7499999999714**

In [2158]: plot(t\_experiment, P\_experiment./1000, xlabel = "Time", ylabel = "kw", ylims=[0, Inf], title

plot!(t\_experiment, [P\_a123 P\_lipo]./1000, label=[pack\_A\_label pack\_B\_label])

hline! ([P\_TAKEOFF, P\_MAX\_CONTINUOUS, P\_SUSTAIN\_CLIMB, P\_SUSTAIN]./1000, lw=0.5, label="") vline!([60, soc\_5\_a123, soc\_*2*0\_lipo], lw=0.5, label="")

Out[2158]:

Input electrical power A123 LiFeP*o*4 (19S1P)/Li-ion (125, 10P,42.0Ahr)

**Total power A123 LiFePo4 (19S1P)** - Li-ion (129, 10P,42*.*0Ahr)

*M*Y

***u***

**200**

**400**

**600**

Time

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

***1*4*/*20**

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2159]:

colorRightAxis = [:blue :red] linestyleRightAxis = :solid

plot(t\_experiment, [V\_bus], xlabel = "Time", title=*(*"Pack voltage\n [Nominal] LiFePO4: " \* $

*# Plot the times when the packs are empty* vline!([soc\_5\_a123, soc\_20\_lipo], lw=0.5, label="")

*# Ugly hack to get all the graph Labels on the same Legend* plot!(1, [NaN NaN], label=["A123 cell voltage" "Li-ion cell voltage"], grid=false, linestyl

p = twinx()

plot!(p, t\_**experi**ment, [V\_a123./num\_**rows\_a12**3 V\_lipo./num\_**rows\_lipo], legend=false, xticks=**

hline! (P, [2.0 3.0], label="", linestyle=:dash, color=[:blue :red]) *# Plot minimum cell vo*

**2**

Out[2159]:

Pack voltage [Nominal] LiFePO4: 68.4V, Li-lon: 50.4V

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

**-**

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

---

---

---

-

-**---**

**----**

**--**

**V bus** A123 cell voltage Li-ion cell voltage

**200**

**600**

400 **Time**

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

15*/*20

2021*/*1*1/*15 EF1:46

**Hybrid battery analysis** Final - Jup**yter Notebook**

In [2160]: plot(t\_experiment, [c\_a123\_experiment c\_lipo\_experiment], title="Discharge rate", **xlabel =**

vline! ([soc\_5**\_a123, so**c\_20\_lipo], lw=0.5, label="") hline! ([c\_a123\_max, c\_lipo\_max], label="", lw=0.5) *# Max nominal discharge rates # savefig("plot.png")*

Out[2160]:

Discharge rate

**- A123 LiFePo4 (19S1P)**

Li-ion (129, 10P,42.0Ahr)

-

**200**

400

**600**

Time

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

**16*/*20**

2021*/*1*1/*15 EF1:46

**Hybrid battery analysis** Final - Jup**yter Notebook**

In [2161]:

b = plot(t\_experiment, [soc**\_a123\_experiment soc\_lipo\_exp**eriment]. \*100, title="State of Char

vline! ([soc\_5**\_a123, Soc\_20\_lip**o], lw=0.5, label="") hline! ([5,20], label="", lw=0.5) *# SOC Limits*

**old\_xticks** = xticks(b[1]) *# grab xticks of the 1st subplot* **new\_xticks** = (round.([soc\_5\_a123, Soc\_20\_lipo]), string. (Int.(round.([soc\_5**\_a123, Soc\_20\_li** vline! (new\_xticks[1]) keep\_indices = findall(x -> all(**x . new\_xt**icks[1]), old\_xticks[1]) **merged\_xti**cks = (old\_xticks [1][k**eep\_indices] U new\_xt**icks[1], old\_xticks[2][ke**ep\_indices ]** xticks! (merg**ed\_xticks)**

Out[2161]:

State of Charge

*100 L*

**A123 LiFePo4 *(*19S1P)** Li-ion (129, 10P,42.0Ahr)

***y5***

**75**

%

50

**25**

**115**

**200**

400

**600634**

*11*5

Time

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

**17*/*20**

2021*/*1*1/*15 EF1:46

**Hybrid battery analys**is Final - Jupy**ter Notebook**

In [2162]:

title = "Energy *(*" \* string(round(kwhr\_a123\_pack + kwhr\_lipo\_pack, digits = 2)) \* "kwHr)" plot(t\_experiment, [E\_a123 E\_lipo], title=title, xlabel = "Time", ylabel = "kwHr", label=[p

vline! ([soc\_5\_a123, soc\_20\_lipo], lw=0.5, label="") hline! ([**kwhr\_a123\_pack], label**="A123 pack energy", linestyle=;dashdot) hline! ([kwhr\_lipo\_pack], label="Li-ion pack energy", linestyle=:dashdot)

lingstyte ze da su

Out[2162] :

Energy (1.92kwHr)

-II-II-III

A123 LiFePo4 (1951) Li-ion (129, 10P,42.0Ahr)

**A123 pack energy** -.-.-.-.- Li-ion pack ene**rgy**

**1.5**

kwHr

0.5

IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII

200

200

4

400 Time

00

**600**

600

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

**18*/2*0**

2021*/*1*1/*15 EF1:46

**Hybrid battery analysis** Final - Jup**yter Notebook**

In [2163]:

plot(t\_experiment, hgt\_airp**lane, xlabel** = "Time", ylabel = "height (m)", xlims=[0,Inf], tit

hline! ([300\*12\*.0254], label="Min safe height", linestyle=:dashdot) hline! ([800\*12\*.0254), label="Pattern height", **linestyle=**:dashdot) hline! ([1000\*12\*.0254), label="Departure height", linestyle=:dashdot) hline! ([1500\*12\*.0254], label="Sustain height", linestyle=:dashdot)

vline! ([soc\_5\_a123, soc\_20\_lipo], lw=0.5, label="")

Out[2163]:

Height AGL

**400**

Min safe height .-.. Pattern height .-.. Departure height

Sustain height

-I-II-III-II-IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII

IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII

height (m)

-I-II-

II-II-

IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII-II-II-II-II-II-III-II-IIIII

-

**200**

200

400

**600**

Time

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

**19*/*20**

2021*/*1*1/*15 EF1:46

**Hybrid battery analysis** Final - Jup**yter Notebook**

In [2164]: plot(t\_**experiment, vel\_airplane, xlabel** = "Time", xlims=[0,30], ylabel = "Speed (m*/s)*", yli

hline!([42\*.511], label="Stall speed", linestyle=:dashdot)

Out[2164]:

Velocity

--------- Stall speed

E

-I-1-1-1-1-1-1-1-1-1-1-1-1-1

-I-I-

-I-I-I-I-I-I-

-I-

-I-

-I-

-I-

-I-I-

-I-

-I

Speed (m*/s)*

**20**

**30**

**Time**

In [ ]:

In [ ]:

In [ ]:

**localhost:8888*/*notebooks*/*SE**NIOR DESIGN SIMULATION ALL*/*11*/*11\_*9*/Hybrid **battery analysis** Final.ipynb

20*/*20