Astrodynamics Software and Science Enabling Toolkit (ASSET) Training

NESC Training – Flight Mechanics Tech. Discipline Team

Astrodynamics and Space Research Laboratory

Presenter: Aaron Houin
PI: Dr. Rohan Sood

The University of Alabama, Tuscaloosa AL

Training Overview

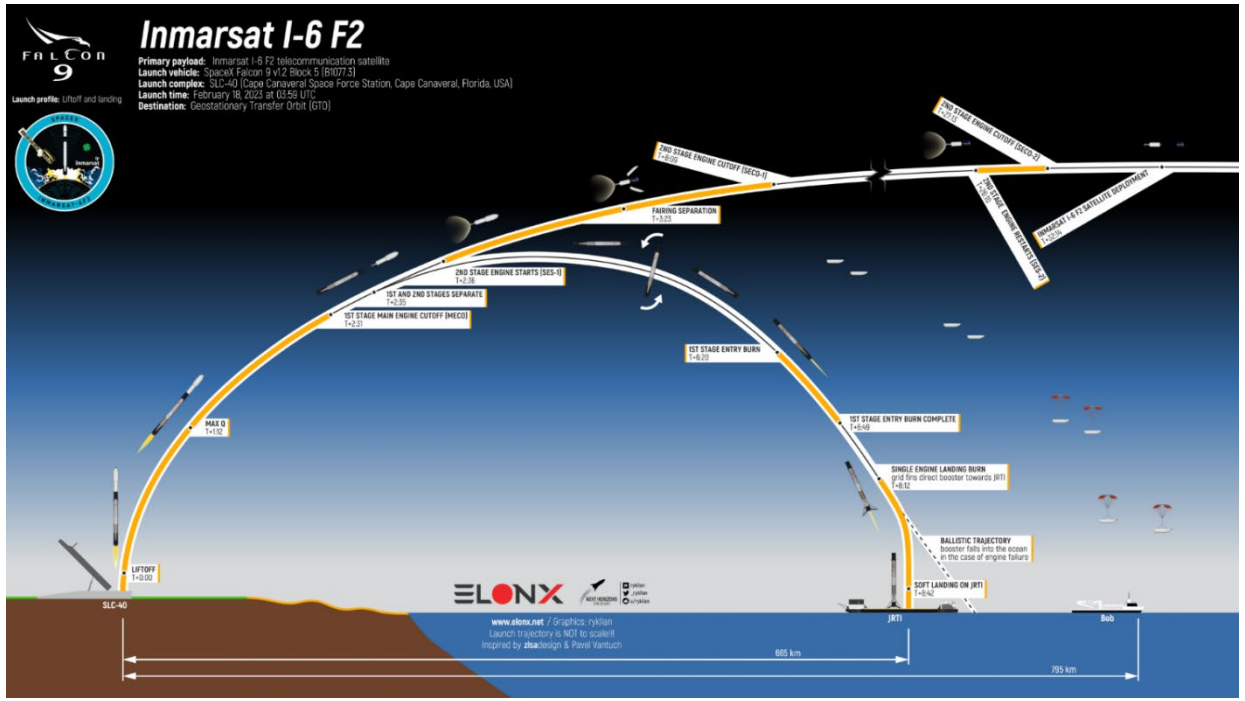
- Day 1: Introduction to ASSET's core functionality
  - Vector Functions - "Basic building blocks used in nearly all ASSET operations"
  - ODEs and Integrators - "Defining solution spaces and integrating the dynamics"
  - Phases - "Setting up optimization problems"
  - Optimal Control Problems (OCPs) - "Configuring complex, multi-phase optimization problems"

- Day 2: Using the "Astro Library" for quick astrodynamics modeling
  - Two-Body Problem Example
  - N-Body Problem Example
  - CR3BP Example
  - Ephemeris Pulsing Rotating Example

Optimal Control Problems

Multi Phase Optimal Control Problems

- Most complex trajectory optimization problems will consist of multiple phases
  - Might need multiple dynamical models
  - Problem might have discontinuous events
    - Ex: Impulsive ΔV, Planetary flyby, drop mass



Optimal

- `oc.OptimalControlProblem` allows us to link together multiple existing phases into a single problem
- Objectives in different phases are implicitly summed

Minimize: $J = \phi_L(\mathbf{y}_{0,f}^{(1)}, \mathbf{s}^{(1)}, \ldots, \mathbf{y}_{0,f}^{(P)}, \mathbf{s}^{(P)}, \mathbf{s}_\mathbf{L}) + \sum_{k=1}^{P} f^{(k)}(\mathbf{y}_{0,f}^{(k)}, \mathbf{s}^{(k)})$

where : $f^{(k)} = \phi^{(k)}(\mathbf{y}_{0,f}^{(k)}, \mathbf{s}^{(k)}) + \int_{t_0^{(k)}}^{t_f^{(k)}} \psi^{(k)}(\mathbf{y}^{(k)}, \mathbf{s}^{(k)})$

Subject to:

Phase Constraints $k = 1, \ldots, P$

Dynamics: $\dot{\mathbf{x}}^{(k)} = \mathbf{f}^{(k)}(\mathbf{x}^{(k)}, t^{(k)}, \mathbf{u}^{(k)}, \mathbf{p}^{(k)}) = \mathbf{f}(\mathbf{y}^{(k)})$

Boundary: $\mathbf{c_b}^{(k)}(\mathbf{x}_{0,f}^{(k)}, t_{0,f}^{(k)}, \mathbf{u}_{0,f}^{(k)}, \mathbf{p}^{(k)}, \mathbf{s}^{(k)}) = \mathbf{c_b}^{(k)}(\mathbf{y}_{0,f}^{(k)}, \mathbf{s}^{(k)}) = \mathbf{0}$

$\mathbf{g_b}^{(k)}(\mathbf{x}_{0,f}^{(k)}, t_{0,f}^{(k)}, \mathbf{u}_{0,f}^{(k)}, \mathbf{p}^{(k)}, \mathbf{s}^{(k)}) = \mathbf{g_b}^{(k)}(\mathbf{y}_{0,f}^{(k)}, \mathbf{s}^{(k)}) \leq \mathbf{0}$

Path: $\mathbf{c_p}^{(k)}(\mathbf{x}^{(k)}, t^{(k)}, \mathbf{u}^{(k)}, \mathbf{p}^{(k)}, \mathbf{s}^{(k)}) = \mathbf{c_p}^{(k)}(\mathbf{y}^{(k)}, \mathbf{s}^{(k)}) = \mathbf{0}$

$\mathbf{g_p}^{(k)}(\mathbf{x}^{(k)}, t^{(k)}, \mathbf{u}^{(k)}, \mathbf{p}^{(k)}, \mathbf{s}^{(k)}) = \mathbf{g_p}^{(k)}(\mathbf{y}^{(k)}, \mathbf{s}^{(k)}) \leq \mathbf{0}$

Integral: $s_i^{(k)} - \int_{t_0^{(k)}}^{t_f^{(k)}} \psi_s^{(k)}(\mathbf{y}^{(k)}, \mathbf{s}^{(k)}) = 0$

Link Constraints

$\mathbf{c_L}(\mathbf{y}_{0,f}^{(1)}, \mathbf{s}^{(1)}, \ldots, \mathbf{y}_{0,f}^{(P)}, \mathbf{s}^{(P)}, \mathbf{s}_\mathbf{L}) = \mathbf{0}$

$\mathbf{g_L}(\mathbf{y}_{0,f}^{(1)}, \mathbf{s}^{(1)}, \ldots, \mathbf{y}_{0,f}^{(P)}, \mathbf{s}^{(P)}, \mathbf{s}_\mathbf{L}) \leq \mathbf{0}$

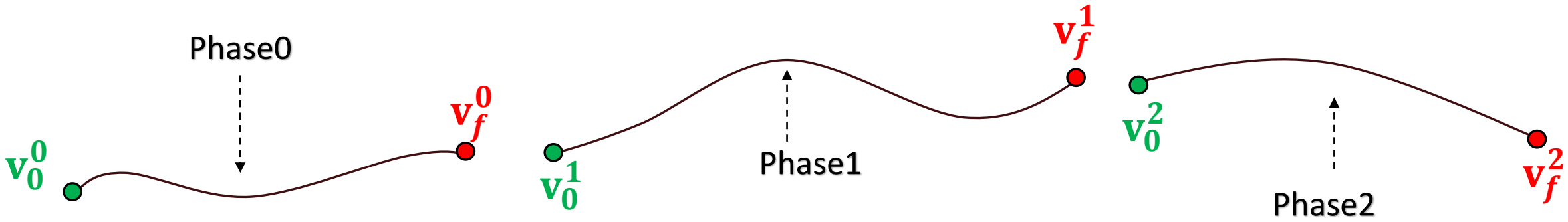
Phases: Initialization

- Create an ocp and then add existing phases
- Phases are named by order of addition
- Reference constituent phases using `ocp.Phase(i)` or with the original object
  - **They are the exact same object**

```
ocp = oc.OptimalControlProblem()

ocp.addPhase(phase0)  # Phase number 0
ocp.addPhase(phase1)  # Phase number 1

print(ocp.Phase(0) == phase0)
print(ocp.Phase(1) == phase1)
print(ocp.Phase(0) == phase1)
```

```
True
True
False
```

Adding Link Constraints and Objectives

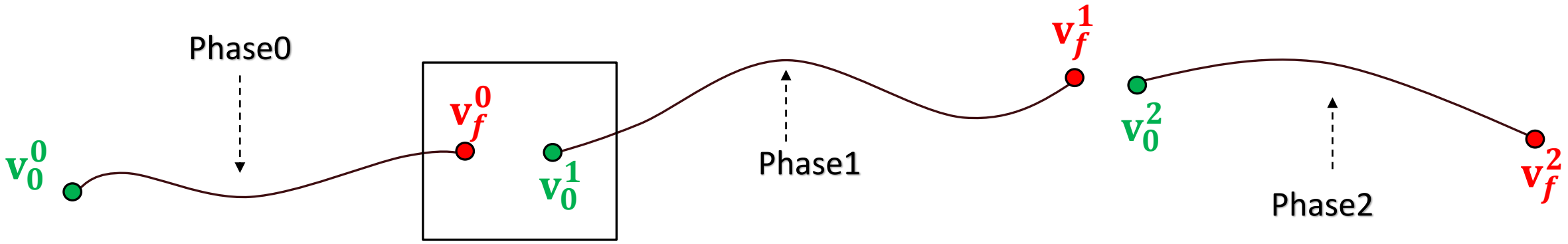


- `ocp.addLink#####(Func,phasei,PhaseRegioni,Vvarsi,phasej, PhaseRegionj, Vvarsj )`
  - `####` = `EqualCon,InequalCon,Objective`
  - Func = an Asset vector function defining the link constraint/objective

$$\mathbf{f}([\mathbf{v}_{0,f}^{i}, \mathbf{v}_{0,f}^{j}])$$

  - phase = phase number or phase name
  - PhaseRegion: “First” or “Last”
  - Vvars: indices of state time control variables

Adding Link Constraints and Objectives



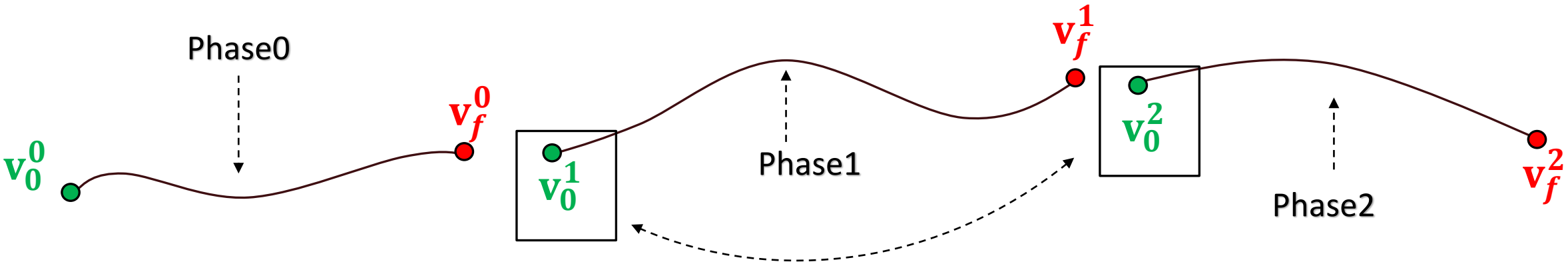
- `ocp.addLink#####(Func,phasei,PhaseRegioni,Vvarsi,phasej, PhaseRegionj, Vvarsj )`
  - Example: Link Last state of Phase0 and first of Phase1 using equality constraint
    - **ocp.addLinkEqualCon(Func,0, "Last",[0,1], 1 , "First", [0,1])**

$$\mathbf{f}\left(\left[x_{0,}^{0}x_{1,}^{0}x_{0,}^{1}x_{1,}^{1}\right]\right) = \mathbf{0}$$

$$\mathbf{i_v} = [0\,,1\,,2\,,3\ ]$$

$$\mathbf{v} = [x_0, x_1, t, u_0] = [\mathbf{x}, t, \mathbf{u}]$$

Adding Link Constraints and Objectives



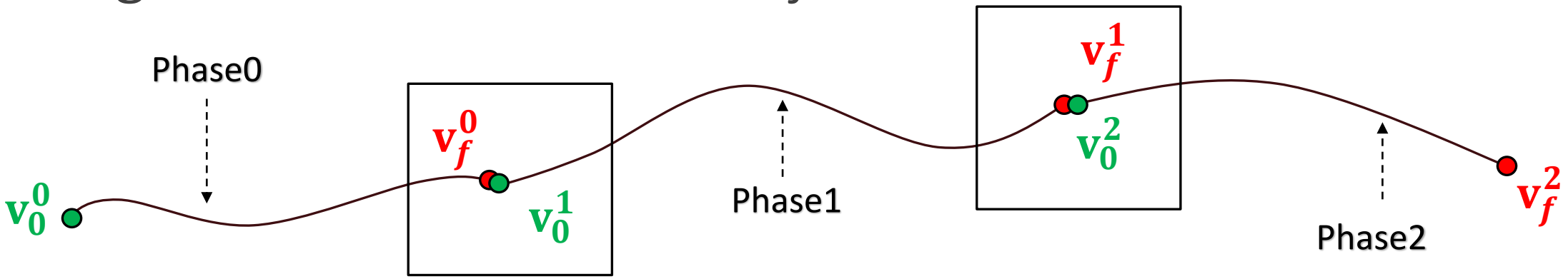
- `ocp.addLink#####(Func,phasei,PhaseRegioni,Vvarsi,phasej, PhaseRegionj, Vvarsj )`
  - Example: Link first time of Phase1 and first of Phase2 using inequality constraint
    - ocp.addLinkInequalCon(Func, 1, “First”,[2], 2 , “First”, [2])

$$\mathbf{f}([t^1, t^2]) < 0$$

$$\mathbf{i_v} = [0\,, 1\,, 2\,, 3\ ]$$

$$\mathbf{v} = [x_0, x_1, t, u_0] = [\mathbf{x}, t, \mathbf{u}]$$

Adding Link Constraints and Objectives



- Most of the time we just want to enforce continuity in state variables
- **`ocp.addForwardLinkEqualCon(phasei, phasej, Vvars)`**
- Example: Enforce continuity in all state and time variables
  - `ocp.addForwardLinkEqualCon(0 , 2, [0,1,2])`

$$\mathbf{i_v} = [0\ , 1\ , 2\ , 3\ ]$$

$$\downarrow$$

$$\mathbf{v} = [x_0, x_1, t, u_0] = [\mathbf{x}, t, \mathbf{u}]$$

Solving and Optimizing

- Invoke optimizer using ocp object
  - `ocp.optimize()`
    - Optimize objective subject to constraints
  - `ocp.solve()`
    - Solve just the constraints
  - `ocp.solve_optimize()`
    - Calls .solve then .optimize
  - `ocp.solve_optimize_solve()`
    - Same as above but calls solve if optimize fails
  - `ocp.optimize_solve()`
    - Call optimize then solve if optimize fails
- Retrieve trajectories from constituent phase
  - `ocp.Phase(i).returnTraj()`
  - `phasei.returnTraj()`

```
phase0 = ode.phase("LGL3",TrajIG0,32)
phase0.addBoundaryValue("Front",range(0,4),TrajIG[0][0:4])
phase0.addBoundaryValue("Path",[4],[1])

phase1 = ode.phase("LGL3",TrajIG1,32)
phase1.setControlMode("NoSpline")
phase1.addLUVarBound("Path",4,0.0,1.0,1.0)
phase1.addEqualCon("Path",PathCon(sigma,c,h_ref,Tmag, g),[0,1,2,4])

phase2 = ode.phase("LGL3",TrajIG2,32)
phase2.addBoundaryValue("Path",[4],[0])
phase2.addBoundaryValue("Back",[1,2],[0,mf])
phase2.addLowerDeltaTimeBound(0.0)
phase2.addValueObjective("Back",0,-1.0)

ocp = oc.OptimalControlProblem()
ocp.addPhase(phase0)
ocp.addPhase(phase1)
ocp.addPhase(phase2)

ocp.addForwardLinkEqualCon(phase0,phase2,range(0,4))

## Trigger optimize on whole ocp
flag = ocp.optimize()

## Can return from original phase or the reference inside ocp
Traj0= phase0.returnTraj()
Traj1= ocp.Phase(1).returnTraj()
Traj2= ocp.Phase(2).returnTraj()
```

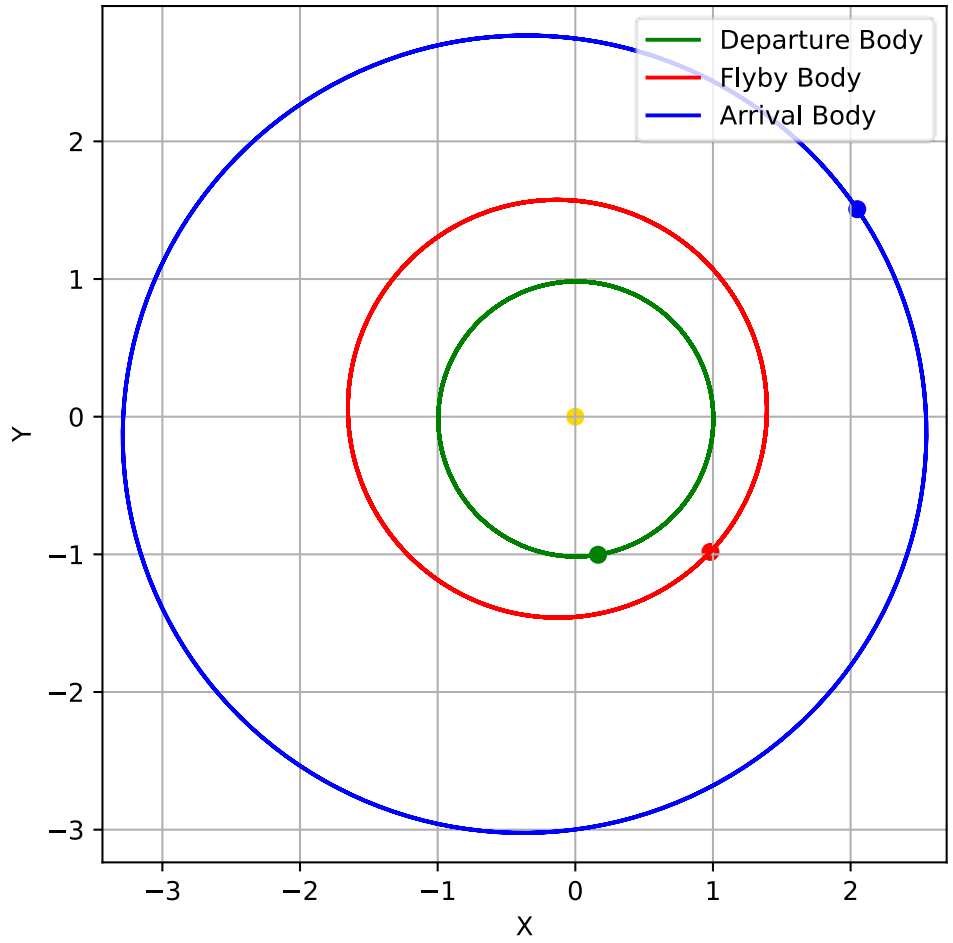
Optimizer Settings

- Each ocp has an optimizer instance as member variable (works same as phase)
  - Use to modify optimizer related settings
  - See PSIOPT tutorial for more details on settings

```
ocp = oc.OptimalControlProblem()
ocp.addPhase(phase1)
ocp.addPhase(phase2)
ocp.addPhase(phase3)
ocp.addPhase(phase4)

## All phases contniuous in everything but mass (var 6)
ocp.addForwardLinkEqualCon(phase1,phase4,[0,1,2,3,4,5, 7,8,9,10])


ocp.optimizer.set_OptLSMode("L1")
ocp.optimizer.set_SoeLSMode("L1")
ocp.optimizer.set_MaxLSIters(2)
ocp.optimizer.set_PrintLevel(1)

ocp.solve_optimize()

Phase1Traj = phase1.returnTraj()  # or ocp.Phase(i).returnTraj()
Phase2Traj = phase2.returnTraj()
Phase3Traj = phase3.returnTraj()
Phase4Traj = phase4.returnTraj()
```

Low-Thrust Flyby Example

- Mass optimal low-thrust transfer leveraging planetary flyby
  - Bounded departure v_∞: $\mathbf{r} - \mathbf{r_d}(t) = \mathbf{0}\,,\ |\mathbf{v} - \mathbf{v_d}(t)| < v_\infty$
  - Rendezvous with target: $\mathbf{r} - \mathbf{r_a}(t) = \mathbf{0}, \mathbf{v} - \mathbf{v_a}(t) = \mathbf{0}$
  - Bound Throttle Vector: $0.001 < |\mathbf{u}| < 1$
  - Mass optimal: $\min: \int_{t_0}^{t_f} |\mathbf{u}| dt$
  - Impulsive flyby approximation
- Discontinuous flyby needs two phases to model
- Requires:
  - TwoBody
  - TwoBodyLTODE



Low-Thrust Flyby Example

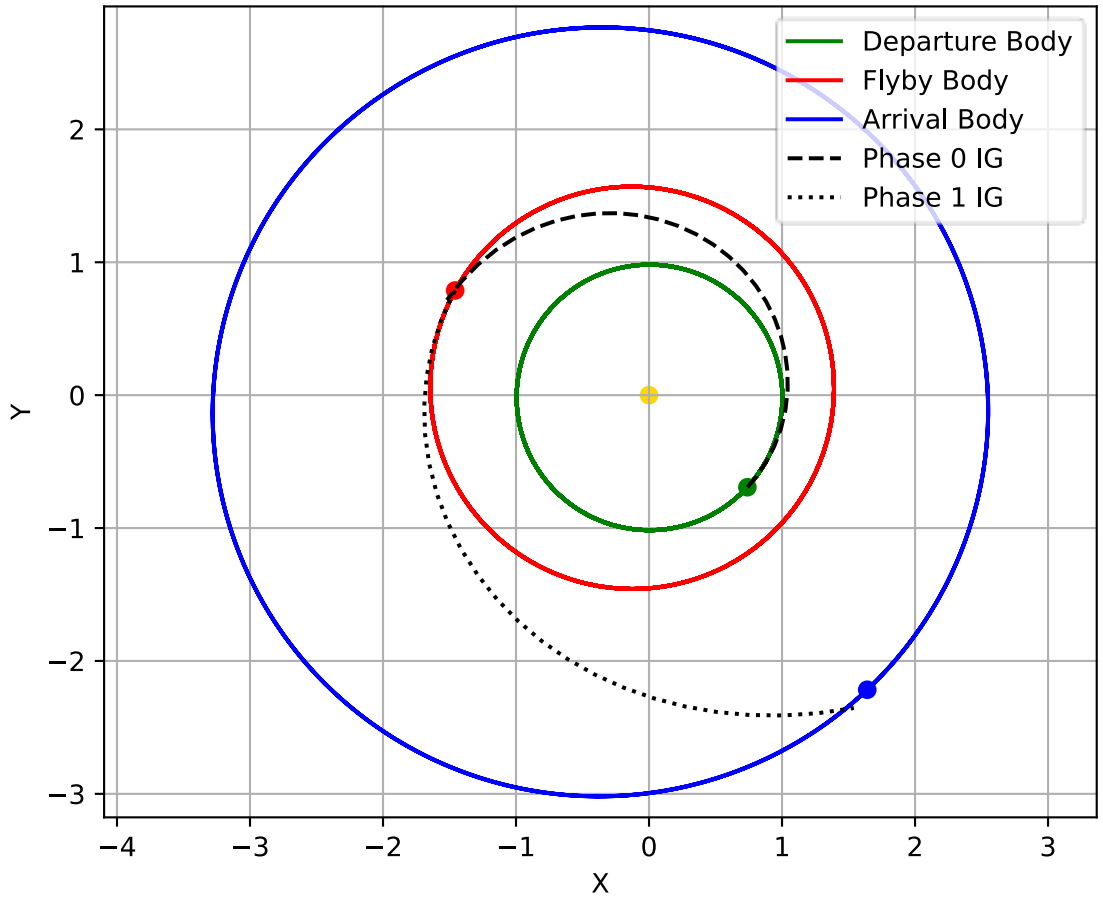
- Setup similar to previous phase example

```
tmax = 100.0 ## Max epoch time
mu = 1.0
alpha = .09  ## low-thrust accelleration
ig_throttle = 0.3  ## throttle level for our ig
vinf = .066

## Initial conditions of  bodies at initial epoch
dep_ig =[ 1.652e-01, -1.0031e+00,  4.425e-05,  9.702e-01,
          1.590e-01,  1.863e-05,  0]
fb_ig= [ 0.981, -0.982, -0.0446,  0.606,  0.644,    -0.00135,
         0.]
arr_ig = [2.048,  1.5072, -0.1259, -0.376,  0.549, -0.0155,
           0.]

fb_rad = 2.599e-05
fb_mu  =3.227e-07

tbode = TwoBody(mu)
tbint = tbode.integrator(.1)

ltode = TwoBodyLT(mu, alpha)
ltig_int = ltode.integrator(.1,TanLaw(ig_throttle),range(0,6))

#################################################################

## Integrate out ics to generate body data
dep_body_traj = tbint.integrate_dense(dep_ig,tmax,20000)
fb_body_traj = tbint.integrate_dense(fb_ig,tmax,20000)
arr_body_traj = tbint.integrate_dense(arr_ig,tmax,20000)

## Load into table for vector functions
dep_body_tab = vf.InterpTable1D(dep_body_traj,tvar = 6)
fb_body_tab  = vf.InterpTable1D(fb_body_traj ,tvar = 6)
arr_body_tab = vf.InterpTable1D(arr_body_traj,tvar = 6)
#################################################################
```

Low-Thrust Flyby Example

- See supplementals for initial guess generation functions
  - Not the best way to do it, should use lambert search

```
t0s = np.linspace(2*np.pi,8*np.pi,400)

traj0,traj1 = Search2(ltode,vinf,ig_throttle,t0s,
                      dep_body_tab,fb_body_tab,arr_body_tab)
```



Low-Thrust Flyby Example

- Setup departure phase

$$|\mathbf{v} - \mathbf{v_d}(t)| < v_\infty$$

```
def VinfFunc(rvtab):
    Vt = Args(4)
    V = Vt.head(3)
    t = Vt[3]
    return (V-rvtab(t).segment(3,3)).norm()
```

```
phase0 = ltode.phase("LGL3",traj0,64)

phase0.addUpperFuncBound("First",VinfFunc(dep_body_tab),[3,4,5,6],vinf)
phase0.addEqualCon("First",PosCon(dep_body_tab),[0,1,2,6])

phase0.addEqualCon("Last",PosCon(fb_body_tab),[0,1,2,6])
phase0.addIntegralObjective(Args(3).norm(),range(7,10))


phase0.addLUNormBound("Path",range(7,10),.001,1.0,1)
phase0.addLowerVarBound("Front",6,traj0[0][6]-2.1,1)
phase0.addUpperVarBound("Front",6,traj0[0][6]+2.1,1)
phase0.addLowerDeltaTimeBound(1.0)
phase0.addLowerVarBound("Back",6,0)
```

$$\mathbf{i_v} = [0\,,1\,,2\,,3\,,4\,,5\,,6\,,7\,,8\,,9\,]$$

$$\mathbf{v} = \left[r_x, r_y, r_z, v_x, v_{y,}\, v_z\,, t, u_x, u_{y,}\, u_z\right] = [\mathbf{x}, t, \mathbf{u}]$$

Low-Thrust Flyby Example

- Setup departure phase

$$\mathbf{r} - \mathbf{r}_d(t) = \mathbf{0}$$

$$\mathbf{r} - \mathbf{r}_{fb}(t) = \mathbf{0}$$

```
def PosCon(rvtab):
    Rt = Args(4)
    R = Rt.head(3)
    t = Rt[3]
    return R-rvtab(t).head(3)
```

```
phase0 = ltode.phase("LGL3",traj0,64)

phase0.addUpperFuncBound("First",VinfFunc(dep_body_tab),[3,4,5,6],vinf)
phase0.addEqualCon("First",PosCon(dep_body_tab),[0,1,2,6])

phase0.addEqualCon("Last",PosCon(fb_body_tab),[0,1,2,6])
phase0.addIntegralObjective(Args(3).norm(),range(7,10))


phase0.addLUNormBound("Path",range(7,10),.001,1.0,1)
phase0.addLowerVarBound("Front",6,traj0[0][6]-2.1,1)
phase0.addUpperVarBound("Front",6,traj0[0][6]+2.1,1)
phase0.addLowerDeltaTimeBound(1.0)
phase0.addLowerVarBound("Back",6,0)
```

$$\mathbf{i_v} = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$$

$$\mathbf{v} = [r_x, r_y, r_z, v_x, v_{y,} v_z, t, u_x, u_{y,} u_z] = [\mathbf{x}, t, \mathbf{u}]$$

Low-Thrust Flyby Example

- Setup departure phase

$$\int_{t_0}^{t_f} |\mathbf{u}| dt$$

$$0.001 < |\mathbf{u}| < 1$$

```
phase0 = ltode.phase("LGL3",traj0,64)

phase0.addUpperFuncBound("First",VinfFunc(dep_body_tab),[3,4,5,6],vinf)
phase0.addEqualCon("First",PosCon(dep_body_tab),[0,1,2,6])

phase0.addEqualCon("Last",PosCon(fb_body_tab),[0,1,2,6])
phase0.addIntegralObjective(Args(3).norm(),range(7,10))


phase0.addLUNormBound("Path",range(7,10),.001,1.0,1)
phase0.addLowerVarBound("Front",6,traj0[0][6]-2.1,1)
phase0.addUpperVarBound("Front",6,traj0[0][6]+2.1,1)
phase0.addLowerDeltaTimeBound(1.0)
phase0.addLowerVarBound("Back",6,0)
```

$$\mathbf{i_v} = [0\,,1\,,2\,,3\,,4\,,5\,,6\,,7\,,8\,,9\,]$$

$$\mathbf{v} = \left[r_x, r_y, r_z, v_x, v_{y,}\, v_z\,, t, u_x, u_{y,}\, u_z\right] = [\mathbf{x}, t, \mathbf{u}]$$

Low-Thrust Flyby Example

- Setup arrival phase

$$0.001 < |\mathbf{u}| < 1$$

$$\int_{t_0}^{t_f} |\mathbf{u}| dt$$

$$\mathbf{r} - \mathbf{r_a}(t) = \mathbf{0}, \mathbf{v} - \mathbf{v_a}(t) = \mathbf{0}$$

```
def RendCon(rvtab):
    RVt = Args(7)
    RV = RVt.head(6)
    t = RVt[6]
    return RV-rvtab(t)
```

```
phase1 = ltode.phase("LGL3",traj1,128)

phase1.addLUNormBound("Path",range(7,10),.001,1.0,1)
phase1.addIntegralObjective(Args(3).norm(),range(7,10))

phase1.addEqualCon("Last",RendCon(arr_body_tab),range(0,7))

phase1.addLowerDeltaTimeBound(1.0)
## Important, mass optimal needs bounded time
phase1.addUpperDeltaTimeBound(16.0)
phase1.addUpperVarBound("Back",6,tmax)
```

$$\mathbf{i_v} = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$$

$$\mathbf{v} = [r_x, r_y, r_z, v_x, v_y, v_z, t, u_x, u_y, u_z] = [\mathbf{x}, t, \mathbf{u}]$$

Low-Thrust Flyby Example

- Setup optimal control problem

$$\mathbf{r}^0 = \mathbf{r}^1, t^0 = t^1$$

$$\left|\mathbf{v}^0 - \mathbf{v}_{fb}(t^0)\right| = \left|\mathbf{v}^1 - \mathbf{v}_{fb}(t^1)\right|$$

```
def VinfMatchCon(tab):
    X = Args(8)
    v0 = X.head3()
    t0 = X[3]

    v1 = X.tail(4).head3()
    t1 = X.tail(4)[3]

    vinf0 = (tab(t0).segment(3,3)-v0).norm()
    vinf1 = (tab(t1).segment(3,3)-v1).norm()
    return (vinf0-vinf1)
```

```
ocp = oc.OptimalControlProblem()

ocp.addPhase(phase0)
ocp.addPhase(phase1)

ocp.addForwardLinkEqualCon(phase0,phase1,[0,1,2,6])

ocp.addLinkEqualCon(VinfMatchCon(fb_body_tab)
                    ,phase0,"Last",[3,4,5,6],
                    phase1,"First",[3,4,5,6])

ocp.addLinkInequalCon(FlybyAngleBound(fb_body_tab,fb_mu,fb_rad)
                    ,phase0,"Last",[3,4,5,6],
                    phase1,"First",[3,4,5,6])


ocp.optimize()

traj0 = phase0.returnTraj()
traj1 = phase1.returnTraj()
```

Low-Thrust Flyby Example

- Setup optimal control problem

$$\cos^{-1}(\hat{\mathbf{v}}_\infty^0 \cdot \hat{\mathbf{v}}_\infty^1) - 2\sin^{-1}\left(\frac{\mu_{fb}}{\mu_{fb} + (\mathbf{v}_\infty^0 \cdot \mathbf{v}_\infty^0) r_{min}}\right) < 0$$

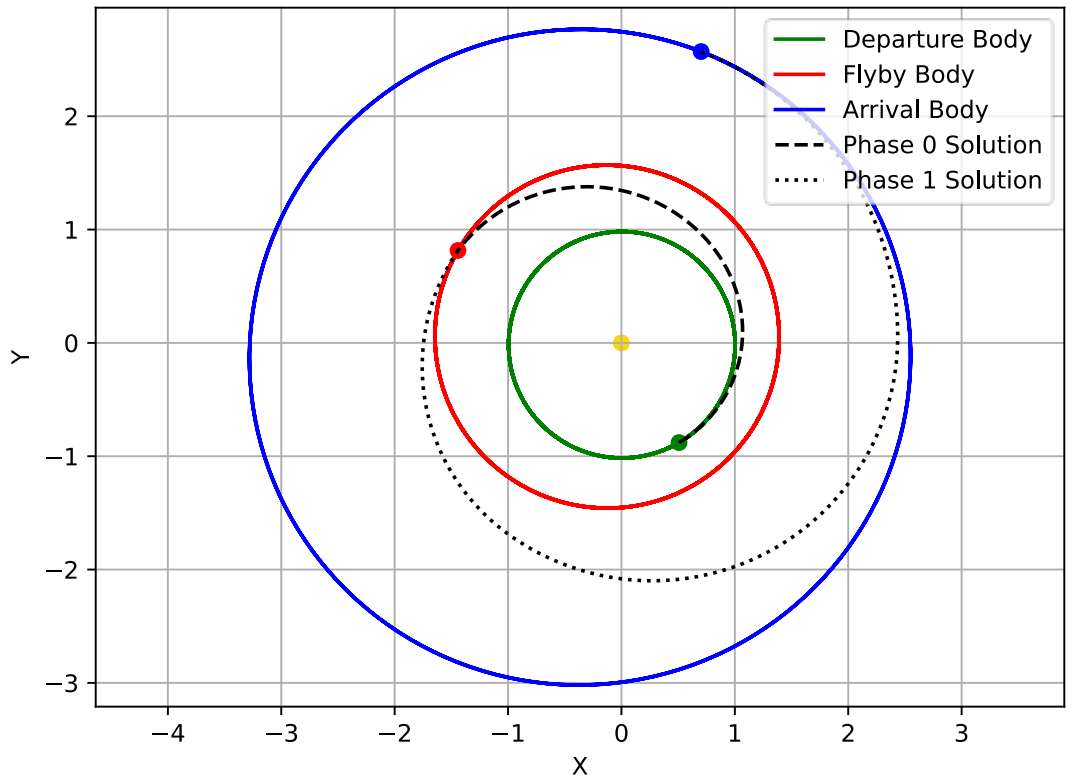
$$\mathbf{v}_\infty^i = \mathbf{v}^i - \mathbf{v}_{fb}(t^i)$$

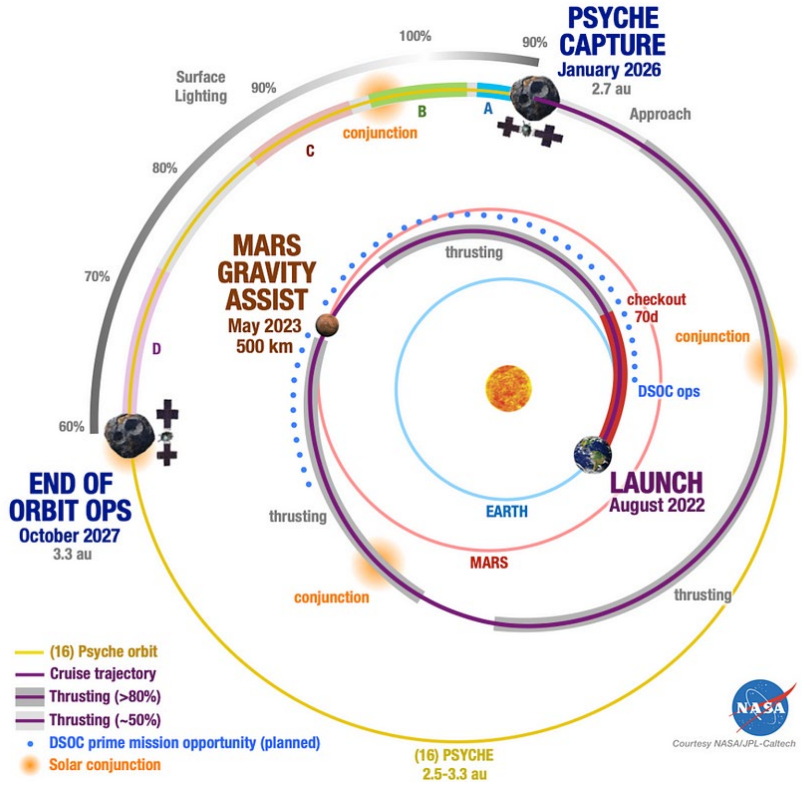
```
def FlybyAngleBound(tab, mubod, minrad):
    X = Args(8)
    v0 = X.head3()
    t0 = X[3]

    v1 = X.tail(4).head3()
    t1 = X.tail(4)[3]

    v0dir = (tab(t0).segment(3,3)-v0).normalized()
    v1dir = (tab(t1).segment(3,3)-v1).normalized()

    vInf2 = (tab(t0).segment(3,3)-v0).squared_norm()

    delta = vf.arccos(vf.dot(v0dir, v1dir))
    deltaMax = 2*vf.arcsin(mubod/(mubod + minrad*vInf2))
    return delta - deltaMax
```

```
ocp = oc.OptimalControlProblem()

ocp.addPhase(phase0)
ocp.addPhase(phase1)

ocp.addForwardLinkEqualCon(phase0,phase1,[0,1,2,6])

ocp.addLinkEqualCon(VinfMatchCon(fb_body_tab)
                    ,phase0,"Last",[3,4,5,6],
                    phase1,"First",[3,4,5,6])

ocp.addLinkInequalCon(FlybyAngleBound(fb_body_tab,fb_mu,fb_rad)
                    ,phase0,"Last",[3,4,5,6],
                    phase1,"First",[3,4,5,6])


ocp.optimize()

traj0 = phase0.returnTraj()
traj1 = phase1.returnTraj()
```

Low-Thrust Flyby Example

- Optimizes in <1s
- Low fidelity approximation of original Psyche trajectory (origin of random constants)





Conclusion

- See tutorial online for more in-depth details

Tutorials - ASSET 0.4.0 documentation (alabamaasrl.github.io)