Numerical Simulation of Orbital Defense/Attack Game using Semi-Direct Collocation with Nolinear Programming Algorithm

1 GA Preprocessing

In the GA preprocessing, the terminal time t_f and the unknown values of the costates at initial time t_0 are chosen as parameters and coded in each individual:

$$\begin{aligned}&\{\lambda_{11,0}, \lambda_{12,0}, \lambda_{13,0}, \lambda_{14,0}, \lambda_{15,0}, \lambda_{16,0},\\ &\lambda_{21,0}, \lambda_{22,0}, \lambda_{23,0}, \lambda_{24,0}, \lambda_{25,0}, \lambda_{26,0},\\ &\lambda_{31,0}, \lambda_{32,0}, \lambda_{33,0}, \lambda_{34,0}, \lambda_{35,0}, \lambda_{36,0}, t_f\}\end{aligned}$$

The objective function in the genetic algorithm as a weighted sum of constraint violations is explicitly given as

$$\tilde{J} = \sum_{j=1}^{16} {c_j}^2$$

where the 16 constraints are given as

$$x_{11,f} - x_{21,f} = 0$$
$$x_{12,f} - x_{22,f} = 0$$
$$x_{13,f} - x_{23,f} = 0$$
$$\lambda_{11,f} + \lambda_{22,f} + \lambda_{33,f} = 2\big(x_{21,f} + x_{31,f}\big)\big[\cos(x_{24,f})\cos(x_{34,f})\cos(x_{23,f} - x_{33,f}) + \sin(x_{24,f})\sin(x_{34,f}) - 1\big]$$
$$\lambda_{12,f} = \lambda_{22,f} = \lambda_{32,f} = 0$$
$$\lambda_{13,f} + \lambda_{23,f} + \lambda_{33,f} = 0$$
$$\lambda_{14,f} + \lambda_{24,f} + \lambda_{34,f} = 2x_{21,f}x_{31,f}\sin(x_{24,f} + x_{34,f})\big[1 - \cos(x_{23,f} - x_{33,f})\big]$$
$$\lambda_{15,f} = \lambda_{25,f} = \lambda_{35,f} = 0$$
$$\lambda_{16,f} = \lambda_{26,f} = \lambda_{36,f} = 0$$
$$H_f = 0$$

The state and costate equations are numerically integrated toward the terminal time t_f to evaluate the constraint violations above for each individual, after which all the individuals evolve to produce a new population according to the GA mechanism. After appropriate amounts of evolution, the optimal individual survived such that the initial guess for all the state and costate variables in the time interval $[t_0, t_f]$ are determined. The open source genetic algorithm toolbox developed by researchers from Sheffield University is employed to provide the initial guess that will yield convergent solution in the Semi-DCNLP algorithm.

2 Semi-DCNLP Algorithm

The extended state vector now contains the state variables of the pursuer, the evader and the target, costate variables of the evader and can be written as

$$\tilde{\boldsymbol{x}} = \left\{ r_P, v_P, \theta_P, \phi_P, \gamma_P, \psi_P, r_E, v_E, \theta_E, \phi_E, \gamma_E, \psi_E, r_T, v_T, \theta_T, \phi_T, \gamma_T, \psi_T, \hat{\lambda_{21}}, \hat{\lambda_{22}}, \hat{\lambda_{24}}, \hat{\lambda_{25}}, \hat{\lambda_{26}} \right\}^T$$

Note that the state variables of the target at any time can be determined due to its prescribed initial state and no active control being imposed, amongst all the possible finial states, the optimal one minimizes the objective function of the game, making terminal time t_f the implicit effect of the target on the game.

In the semi-DCNLP algorithm, the time interval $[t_0, t_f]$ is divided into N subintervals, $[t_{i-1}, t_i], i = 1, 2, \cdots, N$, then the discrete state variables are solved in parallel by the NLP solver.

3 Program Description

The MATLAB script file "$SemiDCNLP\_GA.m$" is the main program that generates the initial guess, which is output in the text file "$NLP\_Input.txt$", as the input of the NLP solver.