

Single Impulse Orbit Transfer Between Two Intersecting Orbits

This *Numerit* program (`maneuvr2`) can be used to determine the single impulse propulsive maneuver between any two orbits which physically intersect. The orbital intersections are determined numerically and the required ΔV is reported.

The following source code, which is part of a support function called `intrsect`, searches for closest approach conditions. The software cycles through combinations of true anomaly on both the initial and final orbits looking for intersections with the `mdopt` multi-dimensional optimization algorithm. The scalar distance between the two orbits is calculated in a second *Numerit* support function called `ca2sfun2`.

```
nf1 = 0
for i = 0 to 11
  x[1] = 30 * i * dtr
  for j = 0 to 3
    x[2] = 90 * j * dtr
    ` find minimum separation distance
    ` true anomalies
    mdopt(method, gtype, n, eps, maxiter, iflag, niter, f, x)
    ca2sfun2(n, x, rdelta)
    ` check for valid solution
    if (rdelta < 1.0e-4)
      nf1 = nf1 + 1
      x1f1[nf1] = modulo(x[1])
      x2f1[nf1] = modulo(x[2])
```

The check for intersection is performed in the section of code where the tolerance is equal to 0.0001. This procedure finds duplicate solutions which the `intrsect` routine eliminates in subsequent calculations.

The source code for the closest approach objective function `ca2sfun2` is as follows:

```
function ca2sfun2(n, x, f)
` closest approach between two satellites
` objective function
` Orbital Mechanics with Numerit
.....
%oev1[6] = x[1]
%oev2[6] = x[2]
orb2eci(%mu, %oev1, r1, v1)
```

Orbital Mechanics with Numerit

```
orb2eci(%mu, %oev2, r2, v2)
` calculate separation distance
dr = r2 - r1
f = sqrt(vdot(dr, dr))
```

This function accepts the two-dimensional true anomaly vector defined by \mathbf{x} , calculates the position and velocity vectors of both orbits at these points, and evaluates the scalar separation distance f between the initial and final orbits according to

$$\Delta \mathbf{r} = \mathbf{r}_i - \mathbf{r}_f \quad (1)$$

and

$$\Delta r = \sqrt{\Delta r_x^2 + \Delta r_y^2 + \Delta r_z^2} \quad (2)$$

The software will prompt you for the classical orbital elements (except true anomaly) for the initial and final orbits. If the calculations determine that the user-defined orbits do not intersect, it will print the following message and stop.

```
no orbital intersection!!
```

The following is a typical draft output created with this computer program. It illustrates the solution for the *default* orbital elements contained in the code.

```
program manevr2
< one impulse transfer between intersecting orbits >
solution # 1
orbit #1 true anomaly at intersection    288.394561  degrees
orbit #2 true anomaly at intersection    18.39456067  degrees
delta-V at intersection                  2482.020001  meters/second
delta-V LVLH pitch angle                  31.89508582  degrees
delta-V LVLH yaw angle                    0  degrees
solution # 2
orbit #1 true anomaly at intersection    249.4842599  degrees
orbit #2 true anomaly at intersection    339.4842595  degrees
delta-V at intersection                  2489.002297  meters/second
delta-V LVLH pitch angle                  -32.60385183  degrees
delta-V LVLH yaw angle                    0  degrees
```

Orbital Mechanics with Numerit

Notice that the program also provides the local-vertical, local horizontal (LVLH) pitch and yaw angles for each impulsive maneuver. These angles are measured with respect to the initial orbit.

The orbital elements of the initial and final orbits for this example are as follows:

	<u>initial orbit</u>	<u>final orbit</u>	
a	6678.4	18953.14	kilometers
e	0.0075	0.6556	----
i	28.5	28.5	degrees
w	30	300	degrees
Ω	0	0	degrees

For this example the orbital inclination and RAAN of the initial and final orbits are the same (the orbits are coplanar) which explains why the yaw angle at both maneuvers is identically zero.