Converting a state vector/orbital elements to a TLE

This document describes a MATLAB script that can be used to calculate an approximate Two Line Element (TLE) set from a user-provided osculating state vector or classical orbital elements. It is valid for spacecraft that can be modeled using the NORAD SGP4 algorithm.

NORAD Two Line Element Sets and the SGP4 algorithm

The NORAD orbit propagators are popular, accurate and easy to use. They are based on the work of Dirk Brouwer, Lane and Cranford, and others. The Two Line Element (TLE) sets required by these propagators are widely distributed on the Internet and maintained by NORAD. Each algorithm is documented in the classic SpaceTrack Report No. 3, “Models for Propagation of NORAD Element Sets” by Felix R. Hoots and Ronald L. Roehrich. This document also contains Fortran source code and test cases for each propagator.

The following is a classical TLE for the NOAA 14 spacecraft:

```
NOAA 14
1 23455U 94089A   97320.90946019  .00000140  00000-0  10191-3 0  2621
2 23455  99.0090 272.6745 0008546 223.1686 136.8816 14.11711747148495
```

The *mean* orbital elements contained in this data are ECI coordinates with respect to the true equator of date and the mean equinox of date. They do not include the effect of nutation.

The following is a brief description of the data contained in each line of a TLE. Each item must appear in its column field in exactly the format specified.

Line 1

The first line is an up to twenty four characters satellite name. Software that reads a database of TLEs will look for this name to find the correct data.

Line 2

```
Column          Description

01       line number of element data
03-07    satellite number
08       classification (u=unclassified)
10-11    international designator (last two digits of launch year)
12-14    international designator (launch number of the year)
15-17    international designator (piece of the launch)
19-20    epoch year (last two digits of year)
21-32    epoch (day of the year and fractional portion of the day)
34-43    first time derivative of the mean motion
45-52    second time derivative of mean motion (decimal point assumed)
54-61    bstar drag term (decimal point assumed)
63       ephemeris type
65-68    element number
69       checksum (modulo 10)
         (letters, blanks, periods, plus signs = 0; minus signs = 1)
```

Line 3

Column	Description
01	line number of element data
03-07	satellite number
09-16	orbital inclination (degrees)
18-25	right ascension of the ascending node (degrees)
27-33	orbital eccentricity (decimal point assumed)
35-42	argument of perigee (degrees)
44-51	mean anomaly (degrees)
53-63	mean motion (orbits per day)
64-68	revolution number at epoch (orbits)
69	checksum (modulo 10)

All other columns are blank or fixed.

A good Internet source for the latest TLEs is Dr. Thomas Kelso's website located at http://celestrak.com. You can also find a Postscript and PDF version of SpaceTrack Report No. 3 at that location.

Important: Do not use TLEs with other orbit propagators. They are only compatible with the SGP4, SDP4 and other algorithms developed by NORAD.

Important Note

Since a TLE represents the epoch calendar date with only two digits, there is a potential Y2K problem. To avoid this problem the SGP algorithms in this software assume that a TLE data field less than 50 is a calendar date after 2000. The code that "corrects" this problem can be found in the **readtle.m** function and looks like the following

```
if (iyr < 50)
   iyear = 2000 + iyr;
else
   iyear = 1900 + iyr;
end
```

In this source code **iyr** is the two digit calendar date read by **readtle.m** and **iyear** is the integer calendar year actually used by the software.

From the state vector provided by the user, the program will determine the companion classical *osculating* orbital elements (subscript *osc*). The initial guess for the SGP4 compatible *mean* orbital elements (subscript *sgp*) are set to these values according to

$$n_{sgp} = n_{osc} = \text{ mean motion}$$

$$e_{sgp} = e_{osc} = \text{ eccentricity}$$

$$i_{sgp} = i_{osc} = \text{ inclination}$$

$$\omega_{sgp} = \omega_{osc} = \text{ argument of perigee}$$

$$\Omega_{sgp} = \Omega_{osc} = \text{ right ascension of ascending node}$$

$$M_{sgp} = M_{osc} = \text{ mean anomaly}$$

If the orbital eccentricity provided by the user is exactly `0`, the software will set the eccentricity value to `1.0e-8` to avoid numerical problems.

This script solves the following vector system of nonlinear equations

$$\mathbf{r}_{osc} - \mathbf{r}_{sgp} = 0$$

$$\mathbf{v}_{osc} - \mathbf{v}_{sgp} = 0$$

This numerical method is trying to minimize the difference between the components of the osculating state vector input by the user ($\mathbf{r}_{osc}$ and $\mathbf{v}_{osc}$) and the state vector computed by the SGP4 algorithm ($\mathbf{r}_{sgp}$ and $\mathbf{v}_{sgp}$) during each algorithm iteration. The *initial guess* for the SGP4 orbital elements is set equal to the orbital elements input by the user. Since the two state vectors are not dramatically different, the algorithm converges in a reasonable CPU time.

Within the main script, the line of code that solves the system of nonlinear equations using the **`fsolve`** MATLAB algorithm is

```
[xf] = fsolve('tlefunc', x);
```

where **`x`** is the initial guess array and **`xf`** is the solution array.

The following is the MATLAB source code for the function which evaluates the system of nonlinear equations.

```
function y = tlefunc(x)

% TLE objective function

% required by sv2tle.m

% input

%  x = vector of current dependent variables

% output

%  y = function value vector evaluated at x

% Orbital Mechanics with MATLAB

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global iflag ri vi

global xmo xnodeo omegao eo xincl

global xno xndt2o xndd6o bstar

pi2 = 2.0 * pi;

y = zeros(6, 1);
```

```
% "unload" current orbital elements

xincl = x(1);
omegao = mod(x(2), pi2);
xnodeo = mod(x(3), pi2);
eo = x(4);
xmo = mod(x(5), pi2);
xno = x(6);

% zero "unknown" tle elements

bstar = 0;

xndt2o = 0;

xndd6o = 0;

% call sgp4 algorithm and compute state vector

iflag = 1;

tsince = 0;

[rtmp, vtmp] = sgp4(tsince);

% define system of nonlinear equations

y(1) = ri(1) - rtmp(1);
y(2) = ri(2) - rtmp(2);
y(3) = ri(3) - rtmp(3);

y(4) = vi(1) - vtmp(1);
y(5) = vi(2) - vtmp(2);
y(6) = vi(3) - vtmp(3);
```

Since we do not know the value of the first time derivative of the mean motion, the second time derivative of mean motion and the *bstar* drag term, these SGP4 orbital elements are set to zero for all computations. This information could be computed by processing many state vectors and performing some type of *differential correction* or orbit determination process.

The following is a typical user interaction with this MATLAB script. The user can elect to input either an ECI state vector (position and velocity vector components) or the osculating classical orbital elements.

```
demo_sv2tle - convert state vector to TLE

please input the name of this satellite
? mysat

TLE calendar date and universal time

please input the calendar date
(1 <= month <= 12, 1 <= day <= 31, year = all digits!)
? 12,24,2021

please input the universal time
(0 <= hours <= 24, 0 <= minutes <= 60, 0 <= seconds <= 60)
? 0,0,0
```

```
        user input menu

 <1> user input of state vector

 <2> user input of orbital elements

? 2

please input the semimajor axis (kilometers)
(semimajor axis > 0)
? 8000

please input the orbital eccentricity (non-dimensional)
(0 <= eccentricity < 1)
? .015

please input the orbital inclination (degrees)
(0 <= inclination <= 180)
? 28.5

please input the argument of perigee (degrees)
(0 <= argument of perigee <= 360)
? 100

please input the right ascension of the ascending node (degrees)
(0 <= raan <= 360)
? 200

please input the true anomaly (degrees)
(0 <= true anomaly <= 360)
? 45

Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the value of the function tolerance, and
the problem appears regular as measured by the gradient.

osculating state vector

        rx (km)                 ry (km)                 rz (km)                rmag (km)
 +7.45643912752328e+03  -1.53143414665499e+03  +2.16602932328762e+03  +7.91425663201181e+03

        vx (kps)                vy (kps)                vz (kps)               vmag (kps)
 +2.15927484581766e+00  +6.21127434865756e+00  -2.76808218520815e+00  +7.13475128355144e+00

osculating orbital elements

        sma (km)              eccentricity         inclination (deg)         argper (deg)
 +8.00000000000000e+03  +1.50000000000000e-02  +2.85000000000000e+01  +1.00000000000000e+02

       raan (deg)         true anomaly (deg)        arglat (deg)           period (min)
 +2.00000000000000e+02  +4.50000000000000e+01  +1.45000000000000e+02  +1.18684684295007e+02

Two Line Element set

MYSAT
1 XXXXXU XXXXXXXX 121358.00000000 .00000000   00000-0  00000-0
2 XXXXX  28.4958 200.0244 0139902  98.3657  45.4159 12.14276755
```