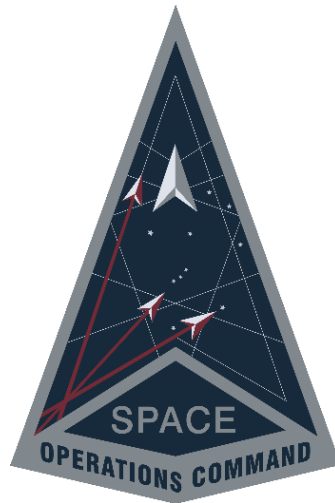


Astrodynamics Standards
Shared Library



Simplified General Perturbations #4
Propagator
(SGP4Prop)

Version 9.4

May 2024

Contents

1.	INTRODUCTION	1
2.	PREREQUISITES.....	1
3.	GETTING STARTED	1
4.	UNDERSTANDING SGP4PROP	1
5.	CALLING SGP4PROP	2
6.	USE CASE	2
6.1.	SET UP ARRAYS FOR PROPAGATION DATA	2
6.2.	LOAD THE LIBRARIES	2
6.3.	INITIALIZE LIBRARIES	3
6.4.	CREATE TLE INPUTS.....	3
6.5.	LOAD THE SATELLITE	4
6.6.	PROPAGATE THE SATELLITE.....	4
6.7.	CLEAN UP	5

1. Introduction

SGP4Prop is an analytic method based on a general perturbation theory for generating ephemerides for satellites in earth-centered orbits. It is the proper means for correctly propagating a Joint Space Operations Center TLE.

Sgp4Prop provides the users with library functions to propagate a satellite state vector/element set to the requested time using Simplified General Perturbations Theory #4 (SGP4). Sgp4Prop only works with SGP and SGP4 Two Line Element Sets (TLEs).

SGP4Prop is publically releasable and can be distributed/redistributed to any agency/organization/person but it must include the SGP4 license agreement file (**SGP4_Open_License.txt**). This file is located in each package distribution.

If you are on Windows, the shared library files will end in ".dll". For example, "Sgp4Prop.dll". If you are on Linux, the shared library will begin with "lib" and end in ".so", and will be all lowercase. For example, libsgp4prop.so.

2. Prerequisites

The following libraries MUST be loaded and initialized before using Sgp4Prop:

- AstroFunc
- DllMain
- EnvConst
- TimeFunc
- TLE

3. Getting Started

To get started, please read the README.txt file that came in the root directory of your distribution. In addition to an overall description contained in the distribution, it has a description of a "**wrapper**".

To get started with **SGP4Prop**, there is a "wrapper" specific to SGP4Prop, under the **SampleCode** directory. Under your language of choice, you will see a "**DriverExample/wrapper**" subdirectory. The files under this directory will have all the Application Programming Interfaces (APIs) available. For SGP4Prop specific APIs, you should see a source file labelled with "SGP4Prop" in the file name. This will be where you will find all the APIs for that specific library. The "DriverExample" directory will also contain several examples of applications that should run by simply running the runExample.bat or runExample.sh script. You can use these examples as a starting point for building your application.

If you do not see your programming language under "SampleCode", look in the HTML documentation for the APIs. Open a browser to the "Documentation/APIDocs/index.html" file. This document will show all the APIs regardless of programming language.

The Astrodynamics Standards libraries should work with any language capable of using Dynamic Link Library (on Windows) or Shared object (on Linux) files.

4. Understanding SGP4Prop

Internally, Sgp4Prop works with the data loaded by Tle. When a satellite is to be propagated, it first needs to be initialized. At initialization time, the programmer will provide the satellite's satKey. The satKey will be looked up in the current set of loaded TLE's (see Tle.doc). If the satKey is found, the associated TLE data is retrieved and used to initialize a satellite in Sgp4Prop.

After the initialization step, when the programmer wants to propagate the satellite to the requested time, the Sgp4Prop will look up the satKey in its set of loaded satellites. If the satKey is found, the associated satellite data is retrieved and used to compute the satellite's new state.

5. Calling Sgp4Prop

Follow these steps to implement the SGP4 propagator:

1. Load and initialize all the prerequisite libraries.
2. Load and initialize Sgp4Prop. Once the library is loaded (the process for doing this is language specific), initialization is accomplished using Sgp4Init().
3. Use the routines in TLE to load satellite state vectors, SGP and/or SGP4 TLEs (see TLE document).
4. Use Sgp4InitSat() to initialize the satellites to be propagated.
5. Propagate the initialized satellites to the desired time using Sgp4PropMse(), Sgp4PropDs50UTC(), etc..
6. Convert the results of step 5 to the desired format if necessary using the routines provided in AstroFunc. (See AstroFunc documentation)

Note: many SGP4 routines such as Sgp4InitSat(), Sgp4PropMse(), and Sgp4PropDs50UTC() return an integer status code. A return value of 0 (zero) indicates success. A non-zero return value indicates an error. Use the GetLastErrMsg() routine in DllMain to retrieve the text of the error message.


6. Use Case

The following use case is taken from the C_Sgp4Prop_Simple.c sample code. It demonstrates the steps for using the SGP4 propagator to propagate a single satellite. This example will illustrate how to:

- Load and initialize all the required libraries.
- Load a TLE.
- Initialize the loaded TLE.
- Propagate the initialized TLE to the requested time.
- Deallocate loaded libraries.

6.1. Set up arrays for propagation data

The first step is to set up the variables and arrays to be used in the propagation:

	 Copy
<pre>int errCode; double mse, ds50UTC, startTime, endTime; __int64 satKey; // Arrays that store propagation data double pos[3]; // Position (km) in TEME of Epoch double vel[3]; // Velocity (km/s) in TEME of Epoch double llh[3]; // Latitude(deg), Longitude(deg), Height above Geoid (km)</pre>	

6.2. Load the libraries

The next step is to load the Astro Standards libraries. In the C_Sgp4Prop_Simple example code, this is done by calling the LoadAstroStdDlls function which calls the Load method in the wrapper of each required library:

	 Copy
--	--

```

// Load MainDll dll
LoadDllMainDll(libPath);

// Load EnvConst dll and assign function pointers
LoadEnvConstDll(libPath);

// Load TimeFunc dll and assign function pointers
LoadTimeFuncDll(libPath);

// Load AstroFunc dll and assign function pointers
LoadAstroFuncDll(libPath);

// Load TLE dll and assign function pointers
LoadTleDll(libPath);

// Load Sgp4Prop dll and assign function pointers
LoadSgp4PropDll(libPath);

```

6.3. Initialize Libraries

Once the libraries are loaded, they must be initialized. In the C_Sgp4Prop_Simple example code, this is done by calling the InitAstroStdDlls function with calls the Init method in the wrapper for each required library. For the AstroFunc and TimeFunc libraries, the initialization requires a pointer to DLLMain.



```

fAddr apPtr;
int errCode;

// Get pointer to the global data
apPtr = DllMainInit();

errCode = EnvInit(apPtr);
if (errCode != 0)
    ShowMsgAndTerminate();

errCode = AstroFuncInit(apPtr);
if (errCode != 0)
    ShowMsgAndTerminate();

errCode = TimeFuncInit(apPtr);
if (errCode != 0)
    ShowMsgAndTerminate();

errCode = TleInit(apPtr);
if (errCode != 0)
    ShowMsgAndTerminate();

errCode = Sgp4Init(apPtr);
if (errCode != 0)
    ShowMsgAndTerminate();

```

6.4. Create TLE inputs

The C_Sgp4Prop_Simple example code uses a single TLE that is created using the TleAddSatFrLines function in the TLE wrapper:



```

// load a TLE using strings (see TLE dll document)

```

```
satKey = TleAddSatFrLines(
    "1 90021U RELEAS14 00051.47568104 +.00000184 +00000+0 +00000-4 0 0814",
    "2 90021 0.0222 182.4923 0000720 45.6036 131.8822 1.00271328 1199");
```

Alternatively, TLEs can be loaded from an external file (the TleLoadFile function) or by passing its data fields (the TleAddSatFrFieldsGP function).

6.5. Load the satellite

With one or more TLEs loaded, the next step is to initialize the TLE(s). This must be done before the TLE(s) can be propagated.



```
// initialize the loaded TLE before it can be propagated
errCode = Sgp4InitSat(satKey);

// check to see if initialization was successful
if (errCode != 0)
    exit(1);
```

6.6. Propagate the satellite

With the libraries and TLEs loaded and initialized, the actual propagation can be performed. This is done by specifying a start and end time and then iterating the propagation calculation:



```
// propagate using specific date, days since 1950 UTC
//(for example using "2000 051.051.47568104" as a start time)

// convert date time group string "YYDDD.DDDDDDDD" to days since 1950, UTC
startTime = DTGToUTC("00051.47568104");

// from start time propagate for 10 days
endTime = startTime + 10;


// propagate for 10 days from start time with 0.5 day step size
for (ds50UTC = startTime; ds50UTC < endTime; ds50UTC += 0.5)
{
    Sgp4PropDs50UTC(satKey, ds50UTC, &mse, pos, vel, llh);
}

// propagate using minutes since satellite's epoch
// propagate for 30 days since satellite's epoch
// with 1 day (1440 minutes) step size
for (mse = 0; mse < (30 * 1440); mse += 1440)
{
    // propagate the initialized TLE to the specified time
    //in minutes since epoch
    Sgp4PropMse(satKey, mse, &ds50UTC, pos, vel, llh);
}
```

There are alternative propagation methods such as Sgp4PropDs50UtcLLH and Sgp4PropDs50UtcPos. See the Sgp4Prop documentation for details.

6.7. Clean Up

The last step after using any libraries is to unload and deallocate the libraries:

	 Copy
<pre>// Remove loaded satellites if no longer needed TleRemoveSat(satKey); // remove loaded TLE from memory Sgp4RemoveSat(satKey); // remove initialized TLE from memory // Free MainDll dll FreeDllMainDll(); // Free EnvConst dll FreeEnvConstDll(); // Free TimeFunc dll FreeTimeFuncDll(); // Free AstroFunc dll FreeAstroFuncDll(); // Free TLE dll FreeTleDll(); // Free Sgp4Prop dll FreeSgp4PropDll();</pre>	

While this use case demonstrates a very simplified use of the SGP4 propagator, all of the important steps from initialization to deallocation are covered and provide a guide for more complex uses.