ODTK Flexible State Space

1 Introduction

ODTK Flexible State Space refers to the capability to modify the state space of a continuously running ODTK filter/smoother without the need to reinitialize the filter. This capability enables you, at a filter restart time, to do things such as:

- add/remove satellites and trackers
- add/remove satellite force model states
- add/remove finite maneuver states
- add/remove measurement biases and solved for measurement types
- restart a diverged satellite with a new initial vector
- reset state stochastic model statistics
- reset clock settings
- update satellite mass settings

The smoother provides the capability to smooth across the transition events, adding and dropping states as necessary. The capability to make state changes without filter reinitialization reduces the impact of these changes and allows rapid return to "steady state" performance.

The algorithm to implement the Flexible State Space is fairly straightforward. However, the consequences of the implementation are not always straightforward. For instance, the algorithm, at times, requires either truncation or partial reinitialization of the state covariance matrix. These operations may break correlations that, coupled with numerical precision considerations, can lead to undesirable results. The purpose of this document is to provide insight into the ODTK Flexible State Space algorithm and implementation so that you better understand how to use the capability, what to expect, and the risks involved.

2 Common Flexible State Space issues and tips

This section covers two of the most significant issues are presented here, including tips for dealing with them. Section 4 has more tips woven into the discussion of use cases.

2.1 Deleting highly correlated states

There may be some confusion over when to delete states from ODTK. If a dropped state is highly correlated to a retained state, then the resultant covariance can become unrealistic or even go negative (develop negative eigenvalues). There are no generic rules of thumb for the time it takes for a state estimate to decorrelate from the remaining covariance. The decorrelation time depends on tracking systems and schedules, process noise, and also elapsed time.

TIPS:

- The most reliable approach is to base a decision to delete states on the correlations in the covariance. The StateFileDumper.htm utility will generate a complete sigma-correlation representation of the covariance at the current state epoch. You can inspect the output manually or parse it for automated testing.

- If you use ad hoc rules to decide when to delete states, AGI recommends that you check the rules with this utility.

- You can have the Filter perform automated checking of eigenvalues upon Restart (see Section 3.5). This will trap the case when the covariance goes negative and will apply desperate measures to regain numerical stability, but no automated test exists to trap the case when the covariance becomes unrealistic. If this test fails, AGI recommends backing up to an earlier restart time, running the Filter back to the current state epoch, and using the StateFileDumper.htm utility to determine when to delete the states from Filter State Space.

2.2 Removing obsolete information from scenario object properties

There may be some confusion over when to delete information from scenario object properties and when to retain information that is historical. The Filter and the Smoother will use information from the object property panels to obtain model information, such as maneuver detail or clock resets, while State Space reflects only the corrections to the modeled event. For other states, all of State Space is formed without consulting the scenario, for example position and velocity.

TIPS:

- The key issues are (a) whether the historical information provides event *model* information and (b) whether that event falls within the Filter or Smoother intervals. A finite maneuver event provides *force model* information to the orbit propagator, and therefore contains event *model* information. A clock event has clock model information (event timing and magnitude information). Finally, a Satellite object contains all of the *force model* information for that satellite.

- The Filter does not require historical *model* information. Therefore, if the state has become decorrelated from other states, and safely dropped, then you can drop the historical *model* information.

- The Smoother requires historical *model* information that falls within the Smoother interval. Therefore, if the *model* event occurs during the Smoother interval, you cannot delete the historical *model* information from the scenario object properties.

3 Flexible State Space algorithms

3.1 Adding states at time (t_k)

Define times t_j and t_k and time interval [t_i, t_L] such that $t_i \leq t_j < t_k < t_L$ with the assumption that, for the current state, the filter has processed measurements across [t_i, t_k], t_k is the current restart time, [t_k, t_L] is the next filter update interval, and [t_j, t_L] is the next associated smoother interval. Now consider the case where new states are to be added at time t_k.

At time t_k, the filter will need to add the new state parameters. We will refer to this as Filter State Addition. On the other hand, the smoother, running in reverse direction, will need to reverse the operation and remove these states at time t_k. We will refer to this as Smoother State Deletion.

3.1.1 Filter state addition at time (t_k)

Filter state addition is straightforward. The algorithm inputs are, at the restart time, an existing state estimate and its corresponding covariance, and an a priori estimate of the new state parameters together with an associated a priori covariance. The assumption is that existing states and new states are statistically independent. Then the new a priori state estimate and covariance is just augmented to the existing state estimate and covariance with cross-covariances between the new and “old” states set to zero. After the addition process, the states may be reordered within the state vector.

3.1.2 Smoother state deletion at time (t_k)

Smoother state deletion is just a truncation process. The existing smoothed state (estimate and covariance) is reordered such that the states that are to be removed are at the end of the state. The states to be removed are those states that were added by the filter at the restart time. These states are then removed from the state estimate vector and the corresponding covariance block and cross-covariances are removed from the state covariance matrix.

3.2 Removing states at time (t_k)

Again, define times t_j and t_k and time interval [t_i, t_L] such that $t_i \leq t_j < t_k < t_L$ with the assumption that, for the current state, the filter has processed measurements across [t_i, t_k], t_k is the current restart time, [t_k, t_L] is the next filter run time interval and [t_j, t_L] is the next associated smoother interval. Now consider the case where a subset of states is to be removed at time t_k.

At time t_k the filter will need to remove the subset of state parameters. We will refer to this as Filter State Deletion. On the other hand, the smoother, running in reverse direction will need to reverse the operation and add these states at time t_k. We will refer to this as Smoother State Addition.

3.2.1 Filter state deletion at time (t_k)

Filter State Deletion algorithm is a truncation process similar to the Smoother State Deletion algorithm in Section 3.1.2. The existing filtered state (estimate and covariance) is reordered such that the states that are to be removed are at the end of the state. These states are then removed from the state estimate vector and the corresponding covariance block and cross-covariances are removed from the state covariance matrix. Before the filter continues, ODTK tests to make sure that the truncation to the covariance did not cause the covariance to go negative due to numerical precision errors. See Section 3.5.

3.2.2 Smoother state addition at time (t_k)

The Smoother State Addition algorithm is more complicated than the corresponding Filter State Addition algorithm. The algorithm inputs are, at the restart time t_k, the existing smoother state estimate and its corresponding covariance of the reduced state as well as the filter state estimate and covariance of the expanded state. Therefore, to augment the smoother solution at time t_k, it is necessary to produce a smoothed solution of the state parameters to be added back in, together with an associated smoothed covariance, and associated smoothed cross-covariances between these parameters and those in the reduced state. These quantities are computed using the Extended Gauss-Markov Theorem. For a description of this theorem, see reference [1].

3.3 State reinitialization at time (t_k)

Some state changes, such as changing the statistic on a Gauss-Markov state, require a reinitialization of the state, including reinitialization of the state estimate, state estimate (covariance) variance, and state estimate (covariance) cross-covariances. You can consider this as two operations: the first operation to remove the state with the “old” statistic, and the second operation to add the state with the “new” statistic. Consequently, the filter will perform a Filter State Deletion at time t_k, followed by a Filter State Addition at time t_k. Reversing direction, the smoother will perform a Smoother State Deletion at time t_k, followed by a Smoother State Addition at time t_k.

3.3.1 State reinitialization at time (t_k) without updating covariance

ODTK does contain an option to update the constant bias value associated with a Gauss-Markov state without “reinitialization”. Using this option, the current filter state error is “zeroed” and its value is incorporated into the constant term. There is no change to the covariance.

3.4 Adding and removing states at time (t_k)

ODTK has the ability to remove some states at restart time t_k while adding in other states at the same time. The algorithm is to the remove all the states necessary before any new ones are added. This includes those states that are removed as part of State ReInitialization (see Section

3.3). Consequently, the filter will perform *one* Filter State Deletion at Time t_k, followed by *one* Filter State Addition at Time t_k. Reversing direction, the smoother will perform *one* Smoother State Deletion at Time t_k, followed by *one* Smoother State Addition at Time t_k.

3.5 Testing truncated Filter covariance

Truncating the filter covariance may break correlations that, coupled with numerical precision errors, may cause the truncated filter covariance to go negative. Because of this, the filter does an eigenvalue decomposition of the truncated covariance to see if any the eigenvalues have gone negative. If the covariance has gone negative, you have the option to apply a diagonal deweighting matrix in hopes that this will correct for the numerical precision errors and allow the filter to proceed. The elements of the diagonal deweighting matrix are computed as a percentage of the corresponding covariance root-error variance elements. After the filter applies the deweighting matrix, it retests the eigenvalues. If the test fails, the filter will terminate, else the filter will continue using the deweighted covariance.

You can control this test by adjusting the following attributes:

- Filter.Restart.OnStateSizeChange.IfCovGoesNegative.Action
- Filter.Restart.OnStateSizeChange.IfCovGoesNegative.PercentSigmaDeweighting.

Search the ODTK Help system for detailed descriptions of these items.

While these actions may recover a positive covariance, you should consider the Tip provided in Section 2.1 above and thereby be assured of no loss of covariance realism.

4 Use cases

Section 4.1 lists several flexible state space use case examples, each of which includes a guide with the steps necessary to implement the use cases. This is not an exhaustive list but is intended to cover the major cases. Section 4.2 lists examples that are specifically related to support of the GPS satellite system. For details about these cases, see reference [2].

There are some steps and considerations common to all use cases. These include the following:

1. You should only make flexible state changes at a restart time.

2. You need to consider backup/recovery in case there are any problems doing the state change. ODTK will automatically make a backup copy of the Restart file and associated Restart folder with the extension “.bak”. You may need to consider saving the current scenario (.sco) file; see item 3.

3. Here is a list of attributes that enable you to make state changes:

    - Change scalar Gauss-Markov states through the Filter.Restart.StochasticModelUpdates.GaussMarkovList attribute

- Change Random Walk states through Filter.Restart.StochasticModelUpdates.RandomWalkList
- Change Vasicek states through Filter.Restart.StochasticModelUpdates.VasicekList
- Change Clock state items through Filter.Restart.ClockUpdates.ClockList
- Make updates to reset the constant values without updating the covariance through ConstantBiasResetList

In these cases, ODTK changes the corresponding object (satellite, facility, etc.) properties to match the Restart value. You should save the scenario to preserve the object properties and restart file settings in case there is a need to repeat the filter runs prior to the state change.

4. ODTK does not implement the changes to the Restart file until the Filter.Go action is initiated. However, once you initiate the "Go" action, ODTK changes the Restart file, and it remains changed even if the Filter terminates before completion.

5. For the smoother to operate successfully, it is necessary that the scenario contain all the objects defining the state parameters at any time during the smoother interval. This means, for example, that if the operation is to remove a satellite, and you need to run the smoother across the transition event, then you need to remove the satellite in the filter by removal from the Filter.SatelliteList and not by deletion of the satellite object from the scenario. Once the smoother interval is past the transition event, then you can remove the object from the scenario.

4.1 Use case examples

The following sections provide some use case examples. There is also a summary of additional steps required to run the filter/smoother because of the changes to the state space. These are only the additional steps and do not include what is already required for a continuously running filter/smoother, such as renaming smoother output file in the filter, redoing the smoother input file list in the smoother, etc.

4.1.1 Add a new satellite

This case supports the launch of a new satellite into a constellation of satellites that are estimated simultaneously in a continuously running ODTK filter/smoother. This case excludes GPS Satellites; see Section 4.2.1 for support of the GPS Satellites. The assumption is that an initial orbit vector and covariance, together with estimates for other satellite state parameters (force model, transponder bias, etc.) are available either through external sources or from ODTK IOD/Least Squares. Given these, the additional steps (additional to a normal filter/smoother run) to add the new satellite are as follows:

1. Add/Build the Satellite object into the scenario. This includes building any related Transponder, GPS receiver, or Antenna subobjects. If an STK *.e ephemeris file is available, then you can use the Initial State Tool to add the satellite initial orbit/covariance.

2. Check to make sure that the satellite is active in the Filter.SatelliteList, either explicitly or by default.

3. Run the Filter.

4. If a Smoother is required, then run the Smoother also.

The Filter will add all the estimated states associated with this satellite and any of its subobjects using the Filter State Addition algorithm at the restart time. This includes orbit, force model, and space-based measurement biases as well as any estimated Transponder, GPS receiver, or Antenna states. The smoother will remove these states at the restart time using the Smoother State Deletion algorithm.

4.1.2 Remove a satellite

This case supports the removal of a satellite from a constellation of satellites that are solved for simultaneously in a continuously running ODTK filter/smoother (excluding the GPS Satellites). The removal may be necessary because the satellite was retired from service, or deorbited, or “super synched”.

The additional steps required to remove the satellite are:

1. If you require a Smoother run across the transition then do the following:

    a. Delete the satellite from the Filter.SatelliteList.

    b. Run the Filter.

    c. Run the Smoother.

    d. When the transition event is no longer contained within the Smoother interval, delete the Satellite object from the scenario.

2. If you do not run the Smoother across the transition event, then proceed as follows:

    a. Delete the Satellite object from the scenario.

    b. Run the Filter.

The filter will remove all the estimated states associated with this satellite and any of its subobjects using the Filter State Deletion algorithm at the restart time. The Smoother will readd these states at the restart time using the Smoother State Addition algorithm

4.1.3 Add a Tracker

The steps necessary to add a Tracker are similar to adding a Satellite. ODTK makes a distinction between Facilities and Trackers. The Tracker may be associated with the Facility object directly (for radars, ground-based optical systems, and SGLS-type systems), or it may be associated with a ground GPS receiver subobject attached to the Facility (for the GPS system). In this case, assume that the tracker is not a ground GPS receiver. The steps to add a Tracker are as follows:

1. Add/Build the Facility object into the scenario.

2. Check to make sure that the Facility/Tracker is active in the Filter.TrackerList, either explicitly or by default.

3. Run the Filter.

4. If you need to use the Smoother, run it now.

The filter will add all the estimated states associated with this facility/tracker and any of its subobjects using the Filter State Addition algorithm at the restart time. This includes measurement biases, facility location, and troposphere scale bias as well as any estimated ground transponder biases. The smoother will remove these states at the restart time using the Smoother State Deletion algorithm.

4.1.4 Remove a tracker

The steps necessary to remove a Tracker object are similar to removing a Satellite.

The additional steps required to remove the Facility/Tracker are:

1. If you are running the Smoother, then proceed as follows:

    a. Delete the tracker from the Filter.TrackerList.

    b. Run the Filter.

    c. Run the Smoother.

    d. When the transition event no longer appears within the Smoother interval, delete the Facility/Tracker object from the scenario.

2. If you are not running the Smoother, then proceed as follows:

    a. Delete the Facility/Tracker object from the scenario.

    b. Run the Filter.

The filter will remove all the estimated states associated with this satellite and any of its subobjects using the Filter State Deletion algorithm at the restart time. The smoother will readd these states at the restart time using the Smoother State Addition algorithm.

4.1.5 Satellite state replacement (one satellite diverged)

This case supports the situation where the orbit state estimate for one of the satellites has diverged, perhaps due to missing maneuver information, and you need to restart that satellite with a new initial vector.

This probably is the most complicated use case because, unlike changing the stochastic model statistics on a state, ODTK has not been designed to drop and then add orbit states at the same time. Therefore, this case requires two filter steps to complete the process. The first filter run removes the satellite (Section 4.1.2) and the second filter run adds the satellite (Section 4.1.1). The first filter run can be as short as possible. If t_k is the epoch time to restart with the new vector, then the first filter run can/should start from the restart record immediately preceding t_k and then end at time t_k. This extra filter run causes you to have to manage an additional smoother ".rough" file relative to the "normal" case.

If you can provide the new initial vector from an STK "*.e" ephemeris file or TLE file, then you can avoid much of the work by using the ODTK "Satellite Replacement Tool" utility. When using this tool, the steps to run the filter and smoother would be as follows:

1. Run the Satellite Replacement Tool.

   This tool replaces the position and velocity state for a satellite in a continuously running filter at the selected epoch. You can also choose to have it replace the satellite covariance, if the covariance is provided in the *.e file. The process does the following for you:

   - Saves the existing scenario and restart file
   - Identifies the restart record immediately preceding the selected epoch
   - Drops the selected satellite from the filter SatelliteList at that restart time
   - Reruns the filter from the restart time to the selected epoch
   - Replaces the satellite position, velocity, and covariance at the selected epoch and adds the satellite to the filter SatelliteList.

2. Complete any necessary changes to the Satellite object, such as enter covariance if not added by the Satellite Replacement Tool.

3. Manage the "extra" Smoother .rough file.

4. Run the Filter.

5. Add the "extra" Smoother .rough file to Smoother input list.

6. Run the Smoother.

4.1.6 Add a Force Model parameter

You can change the solution vector to solve for Solar Radiation Pressure (SRP), drag, or density. This just requires that you set the Estimate flag for the appropriate Satellite force model parameter to "true" before running the restarted filter.

4.1.7 Delete a Force Model parameter

You can turn off estimation of SRP, drag, or density. This just requires that you set the Estimate flag for the appropriate Satellite force model parameter to "false" before running the restarted filter.

4.1.8 Change the statistics on a state

You can change the statistics (i.e., half-life or sigma) on any state that is modeled using a stochastic sequence. This includes measurement biases, transponder biases, facility troposphere scale biases, and satellite SRP, drag, and density force model states.

To make any of these changes you must set the Filter.Restart.StochasticModelUpdates.Enabled flag to “true”. This attribute is exposed only when the Filter is in “Restart” or “AutoRestart” mode. Setting this flag to “true” exposes the lists of state parameters modeled using a stochastic sequence. The state parameters are grouped by their associated stochastic sequence model. Parameters modeled using a the scalar Gauss-Markov model are displayed in the Restart.StochasticModelUpdates.GaussMarkovList . Parameters modeled using the scalar Random Walk model are displayed in the Restart.StochasticModelUpdates.RandomWalkList. Parameters modeled using the scalar Vasicek model are displayed in the Restart.StochasticModelUpdates.VasicekList. When double-clicked, each list provides a display of all the states in the Filter modeled using the associated stochastic sequence, including the current state error estimate, covariance root-error variance, and associated statistics. A complete description of this table is given in the ODTK Help. To change the model statistics, change the values in this table to the desired value. Changes to a model statistic will cause a reinitialization of the state; in other words, the state estimate error is set to zero, and the covariance root-error is set to the model sigma. The filter/smoother uses the State Re-initialization algorithm described in Section 3.3. For those models that are n-state models, such as the two-parameter GPS solar pressure model, changing the statistics on one parameter also causes a reinitialization of the other parameter(s). Also, you can update the statistics for more than one stochastic modeled state.

Changes made to stochastic model parameters via the GaussMarkovList, RandomWalkList, and VasicekList tables are pushed back into scenario object properties when the Filter is executed so that the Properties panels reflect the current settings used by the Filter.

The notes in the ODTK Help provide details and implications to consider when exercising this option. They will not be repeated here. The steps to change the statistics on a state are as follows:

1. Set Filter.Restart.StochasticModelUpdates.Enabled to “true”.
2. Edit, as appropriate, one of these:
    a. Filter.Restart.StochasticModelUpdates.GaussMarkovList
    b. Filter.Restart.StochasticModelUpdates.RandomWalkList
    c. Filter.Restart.StochasticModelUpdates.VasicekList
3. Make changes to the selected table.
4. Run the Filter.
5. Run the Smoother.

4.1.9 Remove obsolete maneuver states

You can remove obsolete maneuver states. If you are only concerned with running the filter and not the smoother, the procedure involved is straightforward. You only need to remove the obsolete maneuvers from the satellite maneuver list(s) prior to initiating the filter at the restart time of interest. However, if you need to run the smoother, removing the obsolete maneuver states is a little bit trickier.

Consider this case:

```
|---------- Filter 1 ------|---------- Filter 2 ------|-------- Filter 3 ----|
  |---------M-----------|
```

It represents three consecutive filter runs, where Filter 2 restarts from the end of Filter 1, Filter 3 restarts from the end of Filter 2, and M represents a maneuver time contained within the Filter 1 time interval.

Now consider the situation where, following each Filter run, the Smoother then runs across the last two filter intervals. First consider running the Smoother across [Filter 1, Filter 2]. Clearly, you cannot delete the maneuver at the start of Filter 2 because the smoother needs the maneuver data across the Filter 1 time period. So, assume that Filter 2 was run with no action on the maneuver, i.e., the maneuver is still in the satellite maneuver list and the maneuver is still in state space. Now consider running the smoother across [Filter 2, Filter3]. It would appear that the maneuver could be removed from the maneuver list at the start of Filter 3. But this would be wrong, because the Smoother would still be looking for the associated maneuver inputs since the maneuver is still in the Filter 2 state space.

So how do you get rid of the maneuver? The trick is to not remove the maneuver from the maneuver list at the start of Filter 3, but to mark the maneuver as “disabled”. This will remove it from state space, but the maneuver information will still be available for the Smoother across the Filter 2 time interval. You could then remove the maneuver from the maneuver list after the Smoother runs and before the start of the next Filter.

4.1.10 Change constant bias and bias sigma

The procedure to change the constant bias and corresponding bias sigma for a state parameter modeled using a stochastic sequence is essentially identical to changing the statistics on the state given in Section 4.1.8. The same list attributes that display and provide the changes to the statistics also display and provide changes to the constant bias and bias sigmas. Also, you use the same State Reinitialization algorithm. Reference Section 4.1.8 for steps involved.

4.1.11 Change constant bias only without updating covariance

ODTK 6.1.2 added the capability to update the constant value associated with a bias state without a full reinitialization (i.e., drop/add) of that state.

To make these changes, you must set the Filter.Restart. StochasticModelUpdates.Enabled flag to “true”. This attribute is exposed only when the Filter is in “Restart” or “AutoRestart” mode. Setting this value to “true” exposes the “ConstantBiasResetList”. Double-click to the right of the

ConstantBiasResetList attribute to display a List dialog that enables you to add and remove any or all of the state parameters that are listed. The state parameter entry is a string of the form <ObjectName>.<StateName>. Example ConstantBiasResetList entries are TrackingSystem1.Facility1.Range, Satellite1.BallisticCoefficient.B, Satellite1.SolarPressure.Cp, and Satellite1.Transponder1.TransponerBias.

For the parameters in the list, ODTK will reset the associated constant bias value by zeroing the current filter state error estimate and adding the error into the constant value. Unlike modifying the constant bias using the GaussMarkov, Random Walk, or Vasicek lists, the filter state will not be "reinitialized" using a state drop/add procedure. However, the FIS smoother and VLS state change processing is unchanged. Also, for the Vasicek parameter, it is the bias error estimate in the "long term" state that is set to zero, with the "short term" error estimate adjusted accordingly.

4.1.12 Update satellite mass

You can update the satellite mass at a restart time if and when a better mass estimate becomes available during normal operations. ODTK only updates the mass; it does not change any state parameters, including those associated with force models that depend on the mass (i.e, solar pressure and drag). The onus is left on you to change the force model state parameters appropriately for a sufficiently large mass change. Failure to do so could lead to degraded performance, including divergence. Updating the mass, in and of itself, does not result in any adding, removing, or reinitializing of any states. It is just that both the filter and smoother will use the new mass estimate after the restart time, and the old mass estimate(s) before the restart time.

To make a mass change, you must set the Filter.Restart.SatelliteMassUpdates.Enabled flag to "true". This attribute is exposed only when the Filter is in "Restart" or "AutoRestart" mode. Setting this flag to "true" exposes the Filter.Restart.SatelliteMassUpdates.SatelliteMassList attribute, which provides a display of all the satellites and current mass estimates contained on the Restart file. A complete description of this table is given in the ODTK Help. To change the satellite mass, change the mass value in this table to the desired value. Unlike the stochastic model tables, changes to the mass do not result in changes to the corresponding object property, which in this case is Satellite.PhysicalProperties.Mass, since this value is intended to reflect the "initial" on-orbit mass, consistent with the initial position and velocity information.

4.2 Use Case Examples Specific to the GPS Satellite System

4.2.1 Add a Separate Partition for "Sick" Satellites

See reference [2] Section 10.1.

4.2.2 Add New GPS Satellites

See reference [2] Section 10.2.

4.2.3 Remove GPS Satellites

See reference [2] Section 10.3.

4.2.4 Change SV PRN Assignments

See reference [2] Section 10.4.

4.2.5 Replace Clocks

See reference [2] Section 10.5.

References

[1] McReynolds, S.R., *Multipoint Smoothing Algorithm for Discrete Linear Systems,* Journal of Guidance, Control, and Dynamics, Volume 12, Number 6, Nov-Dec 1989, Pages 920-924.

[2] *ODTK Applied to GPS Satellite Orbits and Clocks* (go to ODTK LaunchPad → Resources → Handy References)