SSOXmatch - Manual

Author: Tomás Alonso Albi
Afilliation: Observatorio Astronómico Nacional - Instituto Geográfico Nacional (Leave of Absence); Starion for European Space Agency, (Madrid, Spain)
Contact: talonsoalbi@gmail.com

Contents

1 Introduction and implementation details

The pipeline SSOXmatch is a software package that computes positional cross-matches of Solar System objects (SSO) for a list of observations from different surveys. These cross-matches are later to be confirmed by analysing the images. The software is written in pure Java, with only four dependency libraries: postgresql.jar (more specifically postgresql-9.4-1208.jdbc41.jar), jafama-2.3.2.jar (used to improve the speed of some mathematical functions respect the Java API), jfreesvg-3.4.3.jar (used to create SVG charts of the cross-matches, which are directly ingested in the database), and JNISpice.jar. The postgres dependency allows the program to interact with databases, although it is not mandatory to use databases, since from the configuration of the program (the properties file) the database can be disabled to write text files instead. JNISpice is used to read and write Spice kernels in any supported operating system (Linux, Mac, and Windows). The resources provided with the software include the list of observations for all the surveys directly supported (HST, Herschel, Spitzer, XMM, JWST, and Euclid), the list of orbital elements for asteroids (astorb database from the Lowell observatory, and JPL database of asteroids) and comets (cometpro database from the IMCCE, and the comet database by JPL), and the Spice kernels required to compute the ephemerides (among other less-relevant files, de440s.bsp for JPL DE440 planetary ephemerides, and five kernels to provide the positions for the five observatories already mentioned). The cross-matches can be computed from JPL elements or from Lowell+cometpro databases. The jafama library is used only partially, since most functions are now faster in Java since JRE 11, so it may be removed in a future.

Planetary positions are computed by default using JPL DE440 integration, although other integrations kernels are included (from DE200 to DE432). Reduction algorithms follow the IAU 2006 and 1980 recommendations (selectable, being the algorithms for 1980 the default for JPL elements, to mimic the behavior of Horizons). The positions computed are always astrometric for equinox J2000. Planetary magnitudes are calculated following the model by Mallama and Hilton (2018), although slightly simplified with some corrections implemented but commented. It is basically a revision of the classic formulae, but extending it to observations with a higher range of phase angles, like those typical from observations taken from spacecrafts around the Solar System. When the magnitude cannot be estimated for a given asteroid or comet, the value 100 is used. For eclipsed moons 101 is used, for occulted ones 102, and 103 in both cases.

The ephemerides for natural moons make use of a simple algorithm based on precessing ellipses for most bodies, providing an accuracy around the arcsecond. However, there are some exceptions for bodies in which this approach is not good enough: Hyperion and Iapetus are computed using TASS 1.7 theory (Vienne et al. 1997, again with an accuracy around the arcsecond), while the external moons Phoebe (also around Saturn) and Nereid (satellite of Neptune) are computed using numerical integration, with an accuracy much better than the arcsecond. However, the TASS theory can be forced for all the other Saturn moons (Mimas to Titan) by setting the appropriate property. This also applies to the other analytical theories implemented: Mars07 theory by Lainey et al. (2007) for the satellites of Mars, the L1 theory by Lainey et al. (2004) for the satellites of Jupiter, and GUTS86 by Laskar et al. (1987) for the satellites of Uranus. However, the simple method of precessing ellipses can be accurate enough in most cases and it is much faster. The maximum peak errors observed are around 3" for Callisto and Ganimede in Jupiter, but the median of the errors is only slightly above 1" for Europa and Callisto.

The positions of asteroids and comets are computed using numerical integration, with a selectable Runge-Kutta, by default an order 5th Dormand-Prince integrator (J.R. Dormand and P.J. Prince, 1980). The integration model includes the relativistic effects, J2-J4 oblateness for the Sun, Jupiter, and Earth (restricted in the last case to its effects on the Moon when both bodies are computed independently), and the non-gravitational forces on the comets (following the model by Marsden et al. 1973), and asteroids (using the similar model by Farnocchia et al. 2013). The implementation allows to design an specific Solar System including or not the perturbing effects of some asteroids, and to select the integration schema, including a 3rd order, the classic 4th order Runge-Kutta, The Bogacki-Shampine and Tsitouras methods of order 5, Verner methods of order 6 and 7, the 8th order integrator by Dormand and Prince, and others up to order 12, instead of the default 5th order one, which is overall the most adequate one in terms or accuracy and speed. For propagating the orbits of minor bodies with a fast performance, a Solar System without perturbing asteroids, without Pluto, and with the Earth and Moon as a single body was first used (as in Eproc), while in later revisions the physical model was progresively improved and the code optimized. The current physical model is basically equivalent to the one implemented in the integrator used by the Lowell observatory and Horizons, and much evolved respect Eproc, the Fortran program by IMCCE used to compute ephemerides in the previous pipeline (also used in the SkyBot/Miriade services, see Berthier et al. 2006). The strategy is to first pre-integrate all bodies and to write binary files with their positions five times per year (with an interval of 73.05 days), then the adequate file is read to process the observations around the date of the file, with the possibility to continue with the numerical integration to the instant of the observation, or to use just orbital elements computed from the data in the file (for that interval of +/- 36.5 days), which is faster. Due to the nature of the non-gravitational forces, all bodies with non-null parameters in the Marsden model will be integrated numerically. The preintegrated files contain positions respect the Solar System barycenter, and are the same files for all surveys. The entire set of observations are processed from the most recent to the oldest one, independently of to which survey each observation belongs to, which is an adequate approach to read each binary file just once (of course, surveys can also be processed individually). This same integrator program is also used to propagate the positions of the moons Phoebe and Nereid. Most Runge-Kuttas can be used to estimate the integration error and to adapt the integration step dynamically. The step is automatically reduced when integrating particular bodies in case, for instance, of close encounters with another body with mass, since the interacting forces increases and with them also the error estimate, which is then kept below a selected tolerance by reducing the step when needed.

When the calculation time is very close to the reference time of the elements the position of the minor body is computed using orbital elements instead of numerical integration. This has proven to be accurate and faster than performing numerical integration all the time. However, numerical integration is always used when the body is close to the telescope, or the comet/asteroid has a non-null NGF model, since in that case even a little error can be noticed in the output positions. When a cross-match is found, numerical integration may also be used for the little integrations required to compute the positions of the body around the date of the observation, to estimate the start and end times of the cross-match. In addition, in some situations the jafama library, with a fast and approximate implementation of trigonometric and other functions like computing angular distances, is used to increase the speed. A lot of care has been taken to try to get the best possible performance without sacrifying accuracy. For this, instead of using a Healpix approach to discard bodies that will not have a cross-match with a given observation, a direct approach with a more analytical set of conditions to discard bodies in a fast way was applied. For instance, if an observation is towards 10 degrees or more of ecliptic latitude, no asteroid will have cross-match unless its inclination (corrected to refer it to the Earth's orbit and not the Sun) is of at least of 10 degress. By default the binary files resulting from the pre-integration are created up to five years into the future, so they should be updated with at least this periodicity. Consistency is required between the contents of the binary files and the text file with the orbital elements, so when the text file is updated the binary files should be created again. As a general rule, the current average performance is 1.5 observations per second per thread when computing cross-matches, but it depends strongly on the error tolerance used, and other parameters that controls the accuracy of the integration.

The pipeline will automatically create the database tables, and will ingest the data on them after the calculations are finished. All the files needed (except the pre-integration files, that require 27 GB and are required to compute correct cross-matches) are provided in the resources, but they can be also updated from the program to use more updated files. This includes the list of observations, the kernels, and the orbital elements. The program options are contained in the properties file, that again is included in the resources as a static file, but can be replaced by a custom one from the command-line. The properties include everything related to a specific survey, like the query to perform against the remote database to get the list of observations, so to edit the properties is the only step required to support a different survey, to fill the information required for a new one. One of the parameters in the properties is the local folder in which the program will download possible updated files, creating if requested a folder structure identical to that of the resources, in which these local files will have preference respect those in the resources (which will be typically inside a .jar file of the SSO pipeline distribution, and will be updated only with new releases). The kernels are copied from inside the jar file to the file system because the Spice library needs to read them from the file system. Respect the databases there are two configurations in the properties: one for the local database, and another for the remote one. The remote one is the place to ask for the list of observations (when the query present in the properties is an SQL query to a database, instead of a more common HTML query to a web service), or to copy the local database to the remote one (one of the possible actions available described below). The local database is the one modified, where the ingestion takes place. The local database can still be located in a different machine respect the one in which the calculations are performed. If needed, both local and remote databases can be configured to be the same one. The idea is to read from the remote database and write to the local one, and after checking the resulting cross matches the local database can be copied to the remote one, which is another possibility provided in the pipeline.

2 Differences in the results respect the previous pipeline

This pipeline is an evolution of the previous pipeline SSOXMatch by Elena Racero, available at this software repository. Other useful references are Racero et al. 2021 and this IVOA presentation. Since a first goal (version 0.1.0, see version history below) was to compute everything in a same way as before, but without the need to call a Fortran program, the old pipeline was used as reference to provide an equivalent output in this new one. Besides the technical aspects mentioned above, and the pure Java approach, the main implementation differences are:

- The start/end times for the observations listed in the archive are considered to be in UTC, not in TT as before. Unless this is incorrect, it seems the times in the database tables are indeed in UTC. This means the previous pipeline had a computational error slightly greater than one minute, which, depending on the case, can be noticed sometimes in the Ra/Dec positions, but it is generally very little.
- The position uncertainty of the bodies, computed from the CEU parameters available in the Lowell database, and equivalent values in the JPL one, are corrected from in-orbit calculations to the angular uncertainty of the body as seen from the telescope (from version 0.4.0). This is not done in the old pipeline due to a wrong interpretation of these values. It was not done in the first version because the goal was to mimic it.
- The list of observations are initially filtered to discard those that may contain errors, incoherent information, or will be inadequate to compute cross-matches on them. Some surveys like JWST contain (or at least contained in the first months) a substancial amount of incoherent data, like Ra/Dec pointing locations located far from the field of view (fov) region (details on this will be provided below). On the other hand, the integration in some observations is spread over a long time, like many days, while the total integration time is of few hours or less. This wass the case of 7000 HST observations (among them the famous HST deep fields), and around 13000 Spitzer observations. Many cross-matches would take place in these cases, but the probability to catch a given asteroid in the images is minimum, so these observations can be discarded. There is a parameter in the propertes to adjust the ratio between the time span of the observation and the exposure duration to adjust when the observation is discarded, if at all. As a result, some cross-matches that appeard in the previous pipeline are not present in the new one. Another relevant detail is the entries that can be considered as repeated from the point of view of computing cross-matches (exactly the same pointing direction, start and end times, and fov). For the HST and JWST the percentage goes up to 10% in the entire archive, which means that more than 100 000 HST observations will have the same cross-matches. In this case the observation IDs are joined to compute the cross-matches once, although the cross-match tables will still show individual and repeated entries for all these IDs.

- Instead of applying a Healpix tesselation, a direct approach is used. To avoid problems with fovs close to the poles the sky coordinates of the fov are transformed to pixel coordinates using an stereographic sky projection. Then, the fov and the trajectory of the bodies are drawn on an image kept in memory, checking on it if the body pass or not through the fov, first directly for cross-match types 2 and 3 (see Racero et al. 2021, here types 3 does not exist anymore, it is type 2 also), and then considering also the position error (type 1). Output seems equivalent, but internal implementation is quite simple compared to the previous one, and the two steps required before are not needed here. As described above, some conditions are applied to discard fast objects that will never appear in a particular observation, using criteria like the orbit inclination or the angular distance at the observation end time, combined with the observation duration and angular speed.

- The kernels for some old space telescopes were re-constructed, except for the ones orbiting the Earth like the HST, or those for new spacecrafts that are already optimized, like JWST. The trajectories were retrieved from Horizons and converted into SPICE kernels in a homogeneus way for all telescopes. The purpose was to optimize and reduce the size of them, since the kernels for Herschel and Spitzer alone required almost 1 GB. The original kernels can still be downloaded and used instead, if placed in the right place in the local resources. It was checked that the positions returned are identical compared to the original kernels.

- The cross-match tables include additional columns with the cross-match type (from 1 to 2) following the notation by E. Racero et al. (2021) but without type 3, and the probability to observe the cross-match in the image. Cross-match type 3 was the result of interpolating between two points around the field of view (fov) and finding the trajectory theoretically crosses the field of view, something that it is done internally in this pipeline using accurate ephemerides, to return the event as type 1 (position plus error inside the field) or 2 (body position inside the field). The column observation probability, respect 1 = 100%, basically shows the percentage of the area in which the body could be located that lies inside the fov area. Note that even in case of type 2 the probability of observing the body on the images may be little in case the error circle is high. The estimated error is computed from the CEU parameters of the orbital elements of the asteroids, or assumed to be the angular radius for planets/Sun/Moon (this was not done in the previous pipeline, resulting in missed cross-matches of Jupiter for detailed observations pointing towards their poles), one arcsecond for little natural moons, and 10 arcseconds for comets (as in the previous pipeline in the last case). For comets there is an option to estimate the position error as a percentage of the effect of the non-gravitational forces (eject of material forming the comet tail) on the position of the comet. For this and other entries in the properties extensive comments are given to explain the meaning of the input parameters. The new approach allows to compute additional information like the instant the body was closest to the center of the fov, but this is currently not written to any database column.

- The supported natural moons are Phobos and Deimos on Mars, the four Galilean Jupiter moons, for Saturn all the main nine moons from Mimas to Phoebe, for Uranus the five main moons from Miranda to Oberon, and Triton and Nereid for Neptune. The cross-matches for them are automatically discarded in case the moon is eclipsed or occulted during the entire observation, although it is possible to include them optionally when the moon is just eclipsed. Serendipitious or unintentional cross-matches with moons are unlikely, but could happen for external moons. Cross-matches with the Earth, the Earth's Moon, Pluto, and the Sun are also computed since these bodies are supported by the JPL DE440 kernel. Note Pluto is included in the astorb database and can optionaly be computed from these elements, but by default its position is computed using DE440 and the entry for Pluto in astorb is correctly ignored.

- For objects with close encounters with other bodies with mass the pre-integration to obtain the binary files is performed using kernels. This affects almost 1000 bodies that otherwise would have slightly incorrect positions.

- After version 0.1.0 the physical model of the Solar System is progresively much more evolved compared to Eproc, resulting in more accurate ephemerides and cross-matches for NEOs.

A quick comparison of the results of the XMM and Herschel cross-matches with the previous pipeline showed that around 95% of the cross-matches agree, with the first version of the SSOXmatch pipeline. A very little fraction may appear in the new one due to the time correction, while more frequently others only in the old due to including observations that are now filtered. Positions agree generally to a fraction of an arcsecond. In the rare cases (less than 3%) in which this does not happen and a difference of some arcseconds is visible (even after correcting the 1.1 minute difference between TT and UTC), an external check with Horizons showed the results of the new pipeline are generally more accurate. In a first guess, it seems the reason for this could be related to using a new set of kernels for the satellites XMM and Herschel. The kernels were re-constructed entirely, and this solved some inaccuracies and extrapolations present in the old one, quite evident after comparing the positions returned by the old kernels with Horizons. The elements used are of course different, and a detailed comparison suggest this has also an important impact, explaining in at least most cases the differences of several arcseconds (sometimes even more than one degree) in the positions for some specific bodies. It is also important to notice that the reliability of the results of the SSO pipeline highly depends on the quality of the information in the database. The apparent presence of inconsistencies means that a fraction of the output may not be completely reliable, despite of filtering as much as possible the observations with probably wrong data. The previous pipeline was run 3-4 years before the first run of the new one, and the old list of observations used may contained slightly different data, that would contribute to possible differences. Considering how different both pipelines are, their results are remarkably similar.

The HST archive was run with a different list of observations, around half a million for the old pipeline and twice that for the new one, so their results are difficult to compare (166 000 vs 400 000 cross-matches). Herschel and XMM are more adequate for that. A detailed comparison with Herschel and XMM results in the following conclusions:

- Herschel: around 25% of the cross-matches are different in both pipelines, due probably to little changes in the orbits of the asteroids. From the common 75% of cross-matches (251 000), 96% agree with positional differences below 10", and 93% agree up to 3", even despite the TT-UTC difference. 64 cross-matches were found to differ more than 1 degree, due to major changes in the orbits of a few objects, including asteroids and comets. Tests with Horizons showed the new results are better in at least most cases, both when the discrepancy is very high or just a few arcseconds. This is due to the evolution of the orbital elements and does not mean any kind of problem in the old pipeline, since those results can be reproduced with the new pipeline when using the old set of orbital elements. It is noticeable that these discrepancies cannot be anticipated from the CEU parameters, so CEU parameters can be somewhat meaningless. 35600 bodies with cross-matches in the old pipeline was found to produce no cross-matches in the new pipeline, while those still there basically agree perfectly.

- XMM: the new pipeline filters around 60% of the observations (and cross-matches) due to having the same observation ID and OID. In the new pipeline, when this happens, the observation is considered as repeated, although for XMM it probably represents different images taken on the same field with different filters, despite of having the same start/end times and fovs in the archive (some discrepancies were found when comparing the headers of the fits files with the observations listed in the archive). Other minor differences arise from the 240 observations with inconsistent ra, dec and fov, and the more accurate new kernel. Still, around 60% of the cross-matches in different observations (12850) are present in both pipelines. From them, 90% agree to better than 10", and there is no cross-match with a positional discrepancy above the degree. Both percentages of 90% and 60% are lower than the 96% and 75% found for Herschel, probably due to the differences in the list of observations (the old pipeline have EPIC observations and the list of cross-matches include observation IDs that are not present in the new list of observations, so the list of observations is clearly different) and the new kernel for XMM used, while the Herschel percentages below 100% are probably due to the evolution of the elements in the last four years. The amount of observations in the old pipeline with observation IDs not present in the new list is 1251 (12.3%). This, combined with the 60% of repeated entries with the same obsId and obsOid, suggests the XMM observations should be revised, and the cross-matches maybe repeated with a revised list of observations. This was confirmed and fixed in version 0.2.0.

3 Executing the new pipeline

To execute the program the entire package with the dependencies should be exported first as a runnable jar file, selecting the class Main as the default runnable class. The output jar file will have a size around 2 GB. In case the default properties file (local.properties located inside the jar file) needs some modification, a new one can be written from it and provided to the pipeline with the **-environment** parameter. The first basic step is to get the local properties to perform some changes to it if desired, and then to execute the update actions for the database, orbital elements, kernels, and observations. The new files will be downloaded into some sub-directories inside the local folder configured in the properties (local_data parameter). Some care should be taken when updating the observations and the kernels, in case of doubts it may be better to use the files inside the jar file. The next step is to create a set of binary files with the pre-integration of the orbits of all minor bodies, using the **PropagateOrbits** action. This will create around 200 files in the local subfolder nbody, requiring more than 27 GB of space and some days of CPU (depending on how powerful is the machine and the options selected in the properties), and it is a mandatory step to obtain accurate cross-matches. Then, the **IngestAll** action can be called to compute the cross-matches and to write them as text files, or to ingest the results in the local database tables. The **–test** option can be used to execute the pipeline in a selected set of observations for testing purposes (ignoring the observations downloaded from the archive), in which cross-matches for comet ISON or close cross-matches with asteroid Didymos, among others, are computed.

The main program calls one or several actions that will be detailed below. These individual actions can also be called from the IDE (like Eclipse or IntelliJ) directly, since they are written as main methods.

3.1 Command-line options

The JRE recommended to run the pipeline is Java 21 or 23, or in general the latest available. JRE 17 was observed to be 8% faster than JRE 11, and JRE 11 around 20% faster than JRE 8. But JRE 21 is a lot faster when reading and writting binary files. The option of a G1 garbage collector (-XX:+G1GC) will add an additional 5% performance increase for this pipeline respect other collectors. So it is interesting to take advantage of all these performance gains for computation times that may extend for days. A RAM memory of 18 GB seems to be enough to run the pipeline for all surveys at once, with the current number of observations, asteroids, and cross-matches generated (several million). But the process of writting the card-reduction tables, after the ingestion of the cross matches, will be slightly faster if more memory is used, up to about 24 GB. Recommended value is 18 GB or more. The output of the command java -jar SSOXmatch.jar is the basic help written to the console:

```
SSOXmatch v0.6−dev usage information

java −Xmx18G −XX:+UseG1GC −jar SSOXmatch.jar [−action <action> <params>] [−mission <mission>]
[−environment <environment>] [−debugLevel <debugLevel>] [−−update] [−−test]
[JRE >= 8 required. Latest one (JDK >= 21) recommended because it will be the fastest one. Memory and
garbarge collector options above are suggestions, memory used will be usually much less]

Where <action> (optional): one or several actions (case insensitive) between comma. They are executed in
the appropriate (following) order. Some may need a parameter
```

```
 UpdateDatabase – Creates the schema and tables in the local database
 UpdateElements – Downloads orbital elements for asteroids and comets
 UpdateKernels – Downloads kernels for the different missions. Add the parameter 'generate' to create
kernels from Horizons data (when no link is available or the kernel is outdated)
 UpdateObservations – Retrieves the list of observations from the remote database
 IngestElements – Ingests the orbital elements of asteroids into the asteroids table
 IngestBodies – Ingests the complete list of bodies (asteroids, comets, planets) into the identifiers
table
 CrossMatches – Computes and ingest the cross–matches for the missions enabled in the properties
 IngestAll – Ingest all (update database, orbital elements, bodies, cross matches, without downloading
files ) in the local database defined in the properties file
 All – Does everything above in the proper order for the parameters defined in the properties file
 – The following are additional actions to be executed exclusively, not in combination with other
actions
 PropagateOrbits – To pre–integrate the orbits of minor bodies and generate output files to help to
compute accurate cross matches
 IngestFromTextFiles – Ingest from text files obtained when disabling postgres. Add as parameter the
path of the folder with those files
 CopyDatabase – To copy the entire local database to the remote database and schema defined in the
properties
 FixCardReduction – Fix card reduction tables for the local database
 FixCount – Fix count and count_metadata tables for the local database
 Report – Reports a summary of the current configuration and the last ingestion process
 Ephem – Computes ephemerides with comparison against Horizons. See below for examples
 Observation – Process a single custom observation and returns the cross matches in the console
    – Use the parameter –service to launch as a web service in port 8008. Use –port to specify a port
number
        – A service allows to access localhost:port in a browser on the local machine, and query
cross–matches on individual observations more efficiently
    – To pass an observation use the parameter –observation followed by the following parameters
separated
by a space:
        – The mission identifier, like hst or jwst
        – The observation id or name. Multiple ids can be provided between comma. Rest of parameters are
optional if the observation id can be found in the observations file
            – The observation start date and time in UTC, for instance 1997–10–21 12:42:36.49
            – The observation end date and time
            – The observation duration in seconds
            – The right ascension of the field center, in degrees
            – The declination of the field center, in degrees
            – The footprint of the observation in any of the following two formats:
                – (ra1,dec1),(ra2,dec2),(ra3,dec3) ,...  values in radians, to form a polygon in the sky
                – "ra1 dec1 ra2 dec2 ra3 dec3 ..."  values in degrees, to form a polygon in the sky. Do not
forget the ""
    – To read all observations at once (slower startup) but later compute cross–matches faster add the
parameter –all. Useful with –service and –observation with multiple ids
    – To ingest the resulting cross–matches in the database use the parameter –ingest
    – To allow ingesting the resulting cross–matches in the table service_xmatch use the parameter
–ingestService
    – To launch the service in maintenance mode use –maintenance
 – The following are actions related to launching the internal documentation of the package –
 Doc – Launches the PDF documentation using the command 'firefox'. For a different browser, add the
browser command at the end
 Javadoc – Launches the javadoc (technical documentation of the API) using the command 'firefox'. For a
different browser, add the command at the end
```

```
 Paper – Launches the visualization of the PDF Paper using the command 'firefox'. For a different
browser, add the browser command at the end

Where <mission> (optional):
    One or several (between comma) mission identifiers to work with. Default: all those defined in the
properties
Where <environment> (optional):
    Optional environment or complete path to an external properties file. Options: pre_integration,
cross_match, local
    Default: internal properties for the 'local' environment
    In case the environment is the only parameter provided, the content of the properties associated
with that environment will be written to the console
Where <debugLevel> (optional):
    0 – Disable debug messages
    1 – Debug level for debug messages
    2 – Trace level for debug messages
    3 – Normal level for debug messages. Default value unless one is defined in the properties
    4 – Warning level for debug messages
The ––update option will force the update of elements, kernels, or observations even if the local file
is recent enough according to the properties. Default is to not do this
The ––test option will activate the testing observations for the calculation of cross–matches, instead
of those for the survey files

Examples:
 – Export local properties to edit it:
    java –jar SSOXmatch.jar –environment local > custom.properties
 – Force the update of the lists of observations:
    java –jar SSOXmatch.jar –environment custom.properties ––update –action UpdateObservations
 – Initialize the database and ingest elements and bodies:
    java –jar SSOXmatch.jar –environment custom.properties –action
UpdateDatabase,IngestElements,IngestBodies
 – Preintegration of minor bodies:
    java –Xmx18G –XX:+UseG1GC –jar SSOXmatch.jar –environment custom.properties –action PropagateOrbits
 – Do all at once, including computing cross–matches:
    java –Xmx18G –XX:+UseG1GC –jar SSOXmatch.jar –environment custom.properties –action IngestAll
 – Test of computing cross–matches with selected observations (preintegration REQUIRED):
    java –jar SSOXmatch.jar –environment custom.properties –debugLevel 1 ––test
 – Computing cross–matches for a given observation (preintegration REQUIRED):
    java –jar SSOXmatch.jar –environment custom.properties –action Observation –observation jwst
jw02361007001_04104_00004_nrs2 –ingest
 – Test of ephemerides (preintegration NOT required) with comparison against Horizons / Miriade:
      2020 CD3 from geocenter, and Moon from Madrid. Options supported: JD/calendar, UTC/TDB, kernels
(hst/@hst...) and natural moons (Callisto...)
    java –jar SSOXmatch.jar –environment custom.properties –action Ephem "2458184.5, UTC, 2020 CD3,
500@EARTH" "2020–03–06 00:00:00, UTC, Moon, 990@EARTH"
 – Test of ephemerides (preintegration NOT required) using Horizons–like URLs, with comparison against
Horizons:
    java –jar SSOXmatch.jar –environment custom.properties –action Ephem
"https://ssd.jpl.nasa.gov/api/horizons.api?format=text\&COMMAND=%27%3B%27\&MAKE_EPHEM=%27YES%27
\&EPHEM\_TYPE=%27VECTOR%27\&OUT_UNITS=%27KM–S%27\&CENTER=%27@SSB%27\&ANG_FORMAT=
%27DEG%27\&START_TIME=%27JD2450800.5%20TDB%27\&STOP_TIME=%27JD2461000.5%27\&STEP_SIZE=
%2710%20d%27\&QUANTITIES=%271,9,20%27\&EPOCH=2450800.5\&EC=0.1516829576363078\&A=1.097853972438096\&
MA=149.0882670050723\&OM=22.63074081306986\&W=58.07843630667180 4\&IN=1.164548964501213\&ECLIP=
%27J2000%27\&VEC_CORR=%27NONE%27\&OBJ_DATA=%27NO%27\&A1=1.599229127169E–10\&
A2=–5.273644346744E–12\&A3=0.0\&DT=0.0\&R0=1.0\&ALN=1.0\&NM=2.0\&NN=5.093\&NK=0.0\&REF_PLANE=
FRAME\&VEC_TABLE=2"

Quick start (get properties, edit them, read documentation, preintegrate, and compute cross−matches):
 − java −jar SSOXmatch.jar −environment local > custom.properties
 − vi custom.properties
 − java −jar SSOXmatch.jar −action doc
 − java −Xmx15G −XX:+UseG1GC −jar SSOXmatch.jar −environment custom.properties −action PropagateOrbits
 − java −Xmx20G −XX:+UseG1GC −jar SSOXmatch.jar −environment custom.properties −action IngestAll
```

Some example commands are given right above. The card-reduction tables are auxiliary tables for ESA-Sky, to show the trajectory of the minor bodies during an interval of time. They are huge and 'ugly', so when the option alternative_columns is selected they are not written. The options to fix the card-reduction and count tables can be useful when the properties file make use of the parameter incremental_cross_matches_nObs, that allows to compute cross-matches incrementaly, instead of doing all the surveys at once. It can also be useful in case of connectivity or memory problems when writting the card-reduction tables after ingesting successfully all the cross-matches into a local database located in another machine, or to update the count tables when combining the cross-match tables from different schemas. When everything is correct in the local database, it is possible to copy the entire set of tables from the local database to a remote one with the CopyDatabase action.

3.2 Complete local.properties file

Here is an example of a complete properties file. This example file makes use of a tunnel configured locally to connect to the remote database. The database password has been removed.

```
# * Database configuration *

# Local database configuration. Used to ingest the results of the cross matches
# NEVER USE UPPERCASE IN DB NAME OR SCHEMA
local_db_server=localhost
local_db_port=5432
local_db_name=sso
local_db_schema=sso
local_db_user=postgres
local_db_pass=postgres

# Remote database configuration. Direct connection or by tunnel from localhost
# remote_db_port=8300
remote_db_server=localhost
remote_db_port=9002
remote_db_name=
remote_db_schema=
remote_db_user=
remote_db_pass=

# Driver to use
jdbc_driver=org.postgresql.Driver
# Set to true to avoid the requirement of postgres. Cross−matches will be written to text files
disable_postgres=false

# Full path of the logging file . Leave empty to use the console, or to use java −jar ... > file . txt
# The folder selected here will be also used to write any other output file (ingestion files when
# postgres is disabled), replacing the folder selected in local_data below. Use the variable @time to
# select always a different folder in each execution, its value will be replaced by a time stamp
log_file =
#@home@time/log.txt
# OFF, WARNING, NORMAL, TRACE, DEBUG
log_level=NORMAL

# Local folder structure as in resources, so that local files have preference to resources for
# orbital elements and the Spice kernels of the missions. @home for user default home dir (a more
# complex path can be constructed from it). The same with @tmp for the default temporal directory
local_data=@home

# * Properties to download updated elements for asteroids and comets *

# URL to update elements of minor bodies and observatories
url_cometpro=https://ftp.imcce.fr/pub/databases/cometpro/ELTNUM.TXT
url_astorb=https://ftp.lowell.edu/pub/elgb/astorb.dat
url_comet_jpl=https://ssd-api.jpl.nasa.gov/sbdb_query.api?fields=full_name,epoch,a,ma,q,e,i,w,om,tp,M1,K1,M2,K2,PC,A1,A2,A3,DT,equinox,orbit_id,first_obs,last_obs,dat
 url_aster_jpl =https://ssd-api.jpl.nasa.gov/sbdb_query.api?fields=pdes,name,epoch,a,ma,q,e,i,w,om,tp,H,G,A1,A2,A3,DT,equinox,orbit_id,first_obs,last_obs,data_arc,per_y,
url_nong=https://ssd-api.jpl.nasa.gov/sbdb_query.api?fields=full_name,A1,A2,A3,DT,spkid&full-prec=false&sb-cdata=%7B%22OR%22:%5B%22A2%7CNE%7C0%22,%22A
url_eop=https://maia.usno.navy.mil/ser7/finals2000A.all
url_obs_codes=https://minorplanetcenter.net/iau/lists/ObsCodes.html

#  File  names for the elements of minor bodies
fn_cometpro=comet.dat
fn_astorb=aster.dat
fn_comet_jpl=comet_jpl.csv
 fn_aster_jpl=aster_jpl.csv
fn_nong=non_grav.csv

# * SPICE kernels to compute ephemerides *

# Kernels for  DE200 (de200.bsp), DE405 (de405.bsp), DE430 (de430s.bsp), DE432 (de432.bsp), and DE440 (de440s.bsp) are
# supported directly.  Note DE405/430 kernels only supports the year range 1950-2050, while DE440 is valid between
# 1850 and 2150. For longer integrations  the DE432 kernel is provided (1550 to 2650),  while the DE200 kernel is
# valid  between years 1600 and 2169. To support a different kernel or  year  interval  you will need to download it
# externally .  For consistency DE440 should be used when integrating asteroids from both JPL and Lowell elements.
# In  case  the bsp  is  apread over several kernels,  they can be  listed  between a comma
spice.de=de440s.bsp
spice.naif=naif0012.tls
# The SPICE JNI library is  the Java  binding to comunicate with the SPICE C library. It  is  recommended to leave this
# empty so that an automatic value is used for your OS and architecture. Linux, Mac, and Windows are supported for x86
#  architecture ,  and Linux and Mac also for ARM processors. In case you want to use your own binding, set here the name
# of  the  file  and place it  in  the  local  folder  'spice',  inside  the  main folder (along with elements, kernels,   ...)
spice.jni=

# * Configuration properties for  the  integrator  *

# Set  to  true  to  use numerical integration for  computing ephemerides of asteroids and comets. As a general
# rule,  never set  this  to  false
use_numerical_integration=true
# Gravity summation mode. Can be NORMAL, IMPROVED, KAHAN_COMPENSATED_SUMMATION, KLEIN_COMPENSATED_SUMMATION,
        and
# EXTENDED_PRECISION. IMPROVED will minimize round-off errors, but the Kaun mode (compensated summation) is better on
# that,  although slightly  slower.  The method by Klein is an improvement over Kahan, using a 2nd order correction. The
# extended option will use the  partial  implementation of the extended precision mode for computing the gravity
#  accelerations .  This option is  mainly for the  pre-integration of  the  binary  files
gravity_sum_mode=NORMAL
# Number of perturbing asteroids when computing the binary files with the preintegration of the orbits of
# asteroids  and comets. Between 0 and the maximum number of asteroids with masses, currently 343 for DE430-DE440. 16 is
# recommended for consistency with Horizons and Lowell
number_of_perturbers=16
# Consider the masses of the Kuiper belt objects  and the ring model implemented in DE440. Will take no effect
# unless the  property spice.de is  set  to  DE440, and the number_of_perturbers is set to 343
kbo_model_30obj_and_ring=false
# True to  use the  asteroid/comet databases by JPL, false to  use the  Lowell/cometpro ones. The JPL databases contain
# more objects,  like  comet Shoemaker-Levy 9 thay impacted Jupiter, asteroid Oumuamua, or asteroids that impacted Earth.
#  It  also  provides,  in  general,  slightly  better  agreement with observations, although not always
use_jpl_databases=true
# True to  use the JPL parameters to compute non-gravitational forces for asteroids (Yarkovsky, radiation pressure)
use_nong_aster=true
# True to  use the  internal  database of elements of asteroids  that  suffered  close  encounters with other bodies with mass.
# This database was generated with Horizons from a list of close encounter generated with close_encounter_distance=40r,0.001.
# The database is only  available  in  case  use_jpl_databases is  true .  If  true  this  database of elements will be used with
# preference respect  the JPL database for around 850 bodies around the close encounter date and before, to improve the results
# of  the  pre-integration or ephemerides generation. This can introduce discontinuities in  the  PV vector of these bodies when
# they get  close  to  a  body with mass, but the vector will  improve
use_close_encounter_database=true
# Distance between bodies to consider there is  a  close  encounter. Can be specified as  a  function  of  the  radius of
# the  main body, or the absolute distance between them in AU. Do not go beyond 50r and 0.002 AU (Earth-Moon system)
close_encounter_distance=40r,0.001
# Frequency to compute the error estimates. 0 = never,  1  =  all   iterations ,  2  = per  2  iterations ,   ...   Default  is  3
# The effect  is  greater  for  minor bodies that suffer  close  encounters with planets. Values over 5 not recommended
# For high-order ODE solvers 1 is recommended
frequency_of_error_estimate=2
# The factor  to  multiply the default  error  tolerances ,  Lower than 1 means the integration will be more precise than
# with the default  configuration.  Default ( if  commented here) is 1. The error tolerance is  a  value  internally
# selected  as adequate for each solver,  multiplied by  this  factor .  When the error is greater  than  this  tolerance  for
# a  given  iteration ,  the  integration  is  not repeated, so the  error  tolerance  is  not a  strict   limit .  Higher values
# will make the integrations faster , but less accurate. Values above 1E7 not recommended, 1E4 is good to duplicate
# speed while maintaining almost the same accuracy
tolerance_integration_error=1
# ODE solver to use. Can be BOGACKI_SHAMPINE_32, BOGACKI_SHAMPINE_54, CASH_KARP_54, TSITOURAS_54, DORMAND_PRINCE_54,
# TSITOURAS_65, VERNER_65, VERNER_76, DORMAND_PRINCE_87, VERNER_98, ZHANG_10, ONO_1210. The order 10 method
# by Zhang does not have the error control feature. Default is DORMAND_PRINCE_54
solver=DORMAND_PRINCE_54
# Treatment of General Relativity. Can be NONE, DAMOUR_DERUELLE, EINSTEIN_INFELD_HOFFMAN_SIMPLIFIED,
        EINSTEIN_INFELD_HOFFMAN_OPTIMIZED,
# EINSTEIN_INFELD_HOFFMAN. EINSTEIN_INFELD_HOFFMAN_SIMPLIFIED is adequate to replicate the results of Horizons, combined with the
# oblateness option ONLY_J2_EARTH. Default is EINSTEIN_INFELD_HOFFMAN_SIMPLIFIED. The optimized option will skip all but the
# main 3 asteroids, while the EINSTEIN_INFELD_HOFFMAN option will apply all terms
general_relativity =DAMOUR_DERUELLE
# Oblateness mode (J2 to J4) for supported bodies (Sun, Earth, Moon, Jupiter). Can be NONE, ALL, ONLY_J2_EARTH. ALL will use J2-J4 of
# the bodies enabled, by default Sun, Earth, and the Moon. The value ONLY_J2_EARTH can be useful to mimic Horizons. Default is ALL
oblateness=ONLY_J2_EARTH
# Bodies to use in the integration. Only used if the number_of_perturbers=0, otherwise COMPLETE_SOLAR_SYSTEM_WITH_PLUTO is used.
# Can be COMPLETE_SOLAR_SYSTEM_WITH_PLUTO (default), COMPLETE_SOLAR_SYSTEM_WITH_EARTH_MOON_NO_PLUTO, and
# PARTIAL_SOLAR_SYSTEM_EARTH_MOON_BARYCENTER_NO_PLUTO
targets=COMPLETE_SOLAR_SYSTEM_WITH_PLUTO
# Replace during the integration the position of the planets with JPL numerical integration. This prevents deviations of the
# Solar System barycenter, and slows down the integration a bit. Can be NEVER, ALWAYS, ALL_PLANETS_PER_5_ITER,
        ALL_PLANETS_PER_11_ITER,
# ALL_PLANETS_PER_100_ITER (default), ONLY_EARTH_MOON_AND_MERCURY_PER_5_ITER,
        ONLY_EARTH_MOON_AND_MERCURY_PER_11_ITER,
# ONLY_MERCURY_ALL_ITERATIONS, ONLY_MERCURY_PER_5_ITER, ONLY_MERCURY_PER_11_ITER, ONLY_EARTH_MOON_PER_5_ITER,
        and ALWAYS_AND_INSIDE_RK.
# The last value will replace the planetary positions by JPL integration even inside the iterations of the Runge-Kutta
replace_planets_with_jpl=ALL_PLANETS_PER_5_ITER
# Use or not a kernel to solve the positions of the main asteroids that can eventually be used as perturbers, instead of elements.
# Minimum difference in the results, but the kernels are faster because no propagation is needed for the initial conditions of the
# integration. True by default. With the option replace_planets_with_jpl, the asteroid vectors can be replaced after each iteration .
# This option only has effect when use_jpl_databases is true
use_asteroids_kernel=true
# True to adapt the integration step to the error tolerances during the integration, so that it will be slower when needed, or
# faster when possible. True by default
adaptive_step=true
# True to use all bodies to compute if the maximum integration error is exceeded or not. Default is false , to compute
# this automatically depending on the selected option for replace_planets_with_jpl
use_all_bodies_to_estimate_error=false

# * Configuration properties for the computation of cross-matches *

# In the observations folder there is a file named obs_id_list.txt containing a list of observation identifiers for which there are known
# cross-matches in any of the surveys directly supported. When computing cross-matches it is possible to use that list and ignore the
# rest of observations, concentrating computations on the 3% or so of observations that will show cross-matches. The value of this
# property can be NONE, to avoid such filtering, INCLUDE, to use that filtering, and EXCLUDE, to do the opposite and perform computations
# on the rest of 97% of observations that may show no cross-match at all. The list can be customized
filter_observations =NONE
# When computing cross-matches a given one may be missed by a little margin, due to having the positional uncertainty underestimated. In
# additional, the same can happen when the mean motion of the body is not fully correct, so the body may enter the field some minutes
# before or after the observation was taken, while the observation may show that body. An equivalent situation is an inaccurate observation
# footprint. This parameter allows to add time (days) and spatial (degrees) additional uncertainty to the positions of the bodies. Time
# uncertainty will act on the limits of the observation start/end times, while the spatial uncertainty in the uncertainty of the position
# of the body. By default 0 days ( first value) and 0 degrees (second value) is used, which is known to skip some cross-matches. It is
# acceptable to use % to increase both values as a percentage of the observation time and body CEU. Please do not use negative values
add_time_spatial_uncertainty=0,0
# false will use the standard set of columns to mimic the output from the old SSO pipeline, including some additional columns for the
# cross-match probability and images, though. True will use a new set of columns to get a more complete output from the cross-matches
# and the observations, adding observation start/end times and the polygon, and cross-match start/end times. With true the card_reduction
# tables will not be generated
alternative_columns=true
# Set to true to use analytical theories to compute the position of the natural moon always. This will be slower
# but more accurate. Theories are Mars07 for the martian satellites , L1 for the Galilean moons of Jupiter, TASS 1.7
# for Saturn (used if false for Iapetus and Hyperion), and GUST86 for Uranus
use_accurate_moons=true
# If false , magnitude parameters will be read for the comet nucleus, without including the tail. True will
# return the total magnitude including the tail. False is more adequate for the cross matches, but the nuclear
# magnitude is not available for many comets in the JPL database. In those cases, the total magnitude is used
total_magnitude_for_comets=false
# Update interval in days for the orbital elements of asteroids and comets
update_elements_days=180
# Update interval for the lists of observations. Note the interval to update the kernels is the same as for
# the observations
update_observations_days=180
# Will not search for cross-matches if the observation is extended over more days, due
# to an excesive window. This and the next condition are required to skip an observation
max_obs_extension_days=20
# Will not search for cross-matches if duration is very little compare to observation
# extension, so that probability of integrating during asteroid pass is < 5 % (for 20)
max_obs_extension_relative_to_duration=10
# Maximum field of view of the observations. If an observation is read with a greater field , it will be
# rejected (bad input data). If there are many big maps, set to zero or negative to disable this option
max_obs_field_deg=20
# Days to extrapolate the position of the asteroid/comet using its velocity . If an observation
# lasts for more, the cross match search is done in multiple time windows of this size
days_extrapolate_position_using_velocity=1
# Use Pluto from elements. If false it will be used from JPL ephemerides
use_pluto_from_orbital_elements=false
# Set to true to add cross matches of eclipsed moons. Will be probably not visible. Cross matches for occulted moons are
# automatically discarded
include_cross_matches_of_eclipsed_moons=false
# To allow quick discard of cross matches based on the angular distance to observation point
allow_quick_discard_cross_matches=true
# Number of threads to use when computing cross matches and positions for the card_reduction tables
# Set to 0 to use an automatic value, equal to the number of cores in the system minus 1
cross_match_threads=0
# Maximum position uncertainty in degrees for asteroids and comets. If CEU is greater than this
# the object will be discarded and no cross-matches will be computed. Set to <= 0 to disable it.
# Set to 10r (xxr) to set the uncertainty as a function of the observation (footprint) radius.
# 10r means to skip possible cross-matches in which the observation probability is < (1/10)^2 = 1%. This may
# discard real cross-matches when the radius is only around the arcsecond. Both criteria can be
# applied with a comma
max_position_uncertainty=10r,0.01
# CEU is by default hardcoded to 10" for comets, and for asteroids is usually very small or large. For comets
# this is inadequate since the non-gravitational forces (NGF, due to the eject of tail material) can considerably
# change the trajectory for comets once they get close to the Sun. Setting this parameter to 0.01 will replace the
# 10" by the 0.01 fraction (1%) of the difference between the orbit computed with and without the NGF forces in the
# numerical integration (in case this is equivalent to more than 10"). This means to consider that the NGF model
# will improve the computed position by 99% respect the real position of the comet. Depending on the distance to
# the satellite this integration error will be translated to a given angle, different for each comet. Set to 0 to
# ignore this option and use always the CEU hardcoded as the position error for comets. This option has no effect
# for asteroids or comets with no NGF parameters
override_ceu_comets_with_integration_uncertainty=0.01
# Setting this parameter to a value greater than 0 will compute cross matches for this number of observations in
# total . Before starting all observations already present in the cross match tables for all missions will be
# skipped. Using this parameter means that the database tables will not be created unless the UpdateDatabase
# action is called explicitly . This parameter allows to compute the cross matches incrementally. After this, the
# options to fix the card reduction and count tables may be needed.
# In case you want to process in parallel for a given number of observations, you will prefer to skip the first
# set of observations being computed in another process. In that case set the number of observations to skip as a
# second parameter, for instance: 100000 100000 will compute cross matches for one million observations, but
# skipping the first million, being processed in another run with the input 1000000 0
incremental_cross_matches_nObs=0
# Select if images or sketches for each individual cross match should be computed and ingested: none, jpg, svg.
# The format svg is vector graphics and is more optimal, but they may become a problem in non-Linux systems. Format jpg
# is less optimal in terms of image quality and size, but DSS background will show in databases and cross-match services
cross_match_images_format=svg
# Maximum size of the images hold in memory to compute possible cross matches. These images are used internally for
# the algorithm that computes if a cross match is found or not, using the shape of the observation drawn in the image,
# and the trajectory of the minor bodies, to identify a cross match from the color. The pipeline will internally try
# to guarrantee a resolution of at least 0.5" per pixel for this image, but in case the required size is greater than
# this value, this will be the maximum size, and a warning will be thrown. 3840 will guarrantee a resolution of 0.5"
# for observations covering an area of 0.5 degrees in the sky. Losing resolution may return or miss a cross match
# if the object passes just very close to the observation edges of the sensor, but increasing this number may require
# launching the pipeline with more memory
cross_match_images_max_size_pixels=3840
# True to draw in the sketch of the cross-matches a background image with the field of the DSS (Digital Sky Survey)
# Resolution will be 3". Use with care since the size of the sketches in the database can increase considerably
draw_dss_background=false

# * Configuration properties for the missions *

# To add a new mission modify the constants below. Do not change default identifiers, since its observations are
# treated within the code and a correct identification is required for that (jwst and spz). To disable a mission
# remove it from the Ids and Names lists. If you edit these entries , include l2 since it is required in some kernels
missionIds=jwst,hst,hsa,xmm,spz,iue,gaia,kepler,chandra,tess,euclid,l2
missionNames=JWST,HST,Herschel,XMM,Spitzer,IUE,GAIA,Kepler,Chandra,TESS,Euclid,L2

# URLs to update the Spice kernels for the satellites
# Note some kernels are updated from Horizons using this software, since the resulting file is much lighter.
# That is why the url is empty
```

```
url_spice.jwst=https://naif.jpl.nasa.gov/pub/naif/JWST/kernels/spk/jwst_pred.bsp
url_spice.hst=https://naif.jpl.nasa.gov/pub/naif/HST/kernels/spk/hst.bsp
url_spice.hsa=
url_spice.xmm=
#url_spice.xmm=http://spiftp.esac.esa.int/data/SPICE/XMM/kernels/spk/xmm_horizons_19991210_20230223_v01.bsp
url_spice.spz=
#url_spice.spz=https://naif.jpl.nasa.gov/pub/naif/SIRTF/kernels/spk/spk_030825_200134_220101.bsp
url_spice.iue=https://naif.jpl.nasa.gov/pub/naif/IUE/kernels/spk/IUE.bsp
url_spice.gaia=
#url_spice.gaia=http://spiftp.esac.esa.int/data/SPICE/Gaia/kernels/spk/gaia_20131219_20250517_v01.bsp
#url_spice.integral=http://spiftp.esac.esa.int/data/SPICE/INTEGRAL/kernels/spk/integral_sc_ssm_20021017_20230618_v01.bsp
url_spice.kepler=
#url_spice.kepler=https://archive.stsci.edu/pub/k2/spice/spk_2018290000000_2018306220633_kplr.bsp
url_spice.chandra=
url_spice.tess=
url_spice.euclid=https://spiftp.esac.esa.int/data/SPICE/EUCLID/kernels/spk/euclid_flp_00061_20230701_20310907_v01.bsp
url_spice.l2=
#url_spice.l2=https://naif.jpl.nasa.gov/pub/naif/generic_kernels/spk/lagrange_point/L2_de431.bsp

# Limiting magnitudes for particular surveys, and default value for other surveys
# Set to −1 for a given survey to disable limiting magnitude for it and use the global value limiting_magnitude
limiting_magnitude.jwst=34
limiting_magnitude.hst=31
limiting_magnitude.xmm=22
limiting_magnitude.spz=22
limiting_magnitude.hsa=26
limiting_magnitude.euclid=28
limiting_magnitude=34

# File names for the Spice kernels
spice.jwst=jwst.bsp
spice.hst=hst.bsp
spice.hsa=herschel.bsp
spice.xmm=xmm−newton.bsp
spice.spz=spitzer.bsp
spice.iue=iue.bsp
spice.gaia=gaia.bsp
spice.kepler=kepler.bsp
spice.chandra=chandra.bsp
spice.tess=tess.bsp
spice.euclid=euclid.bsp
spice.l2=l2.bsp

# Queries to obtain the dump files
# TAP based
dump_query.hst=https://hst.esac.esa.int/tap−server/tap/sync?LANG=ADQL&REQUEST=doQuery&FORMAT=csv&QUERY=SELECT+TOP+@TOP+obs_id,obs_id,t_min,t_
dump_query.jwst=https://jwst.esac.esa.int/server/tap/sync?LANG=ADQL&REQUEST=doQuery&FORMAT=csv&QUERY=SELECT+TOP+@TOP+obs_id,obs_id,t_min,t_ma
dump_query.xmm=https://nxsa.esac.esa.int/tap−server/tap/sync?REQUEST=doQuery&LANG=ADQL&FORMAT=csv&QUERY=SELECT+TOP+@TOP+observation_oid,ob
# NOTE DURATION IN HSA IS IN MILLISECONDS !
dump_query.hsa=http://archives.esac.esa.int/hsa/whsa−tap−server/tap/sync?REQUEST=doQuery&LANG=ADQL&FORMAT=csv&QUERY=SELECT+observation_oid,obs
# SQL based
dump_query.spz=select oid, obs_id, start_time, stop_time, duration, ra_deg, dec_deg, fov from observations.spitzer_irac_fdw where fov is not null
        order by start_time desc
dump_query.euclid=https://easotf.esac.esa.int/tap−server/tap/sync?REQUEST=doQuery&LANG=ADQL&FORMAT=csv&PHASE=RUN&QUERY=SELECT+observation_id

# Names of the mission dump files
dump_file.jwst=jwst_observations.csv
dump_file.hst=hst_observations.csv
dump_file.hsa=herschel_observations.csv
dump_file.xmm=xmm_observations.csv
dump_file.spz=spitzer_observations.csv
dump_file.euclid=euclid_observations.csv
```

In practice, for a properties file located in the local filesystem it is only necessary to mention explicitly those properties you want to change, since the default file (for the local environment) located inside the jar file is loaded first. However, it is recommended to mention all of them to avoid mistakes with possible changes in the default values in future versions.

The update interval is configured above to 180 days, so that unless the update is forced a new file for orbital elements or observations will not be downloaded if the current one is recent enough. The numerical integration can be disabled to perform fast tests without the need of pre-integrating the orbits in binary files, but this has little interest since the results will be incorrect, except maybe for cross-matches that take place very close to the reference time of the elements. The limiting magnitude for each survey is another parameter that will probably never need any change, but in a future version the limiting magnitude may be computed for each instrument individually. The number of threads is an important parameter that can speed up the computations, but setting it too high can have the opposite effect. It is possible to skip some surveys by removing them from the missionIds and missionNames parameters, but leaving both variables with consistent values, with the same surveys and in the same order.

3.3 Database tables

The database tables are listed below. The asteroid and comet tables contains the orbital elements for asteroids and comets, and they are just informative. The ssoid table contain the list of all body identifiers, names and NAIF codes, including asteroids, comets, planets, and natural moons. The xmatch tables contain the cross-matches for each survey, and the card-reduction ones the Ra and Dec for each body that is present in its corresponding xmatch table, computed every day at 0h UTC during the entire life time of the mission. The card-reduction tables are used by ESA-Sky to draw the trajectory of the body, while marks for specific cross-matches are drawn from the information found in the xmatch tables. These tables are created automatically for each survey listed in the properties file. The count table informs about how many cross-matches a given body have in the different surveys. This database structure is the same as in the previous pipeline, with some additional entries in the xmatch and ssoid tables. The close_encounter table is new in the latest version of the pipeline, listing all objects that have close encounters with other bodies with mass. In case of using the JPL databases, this will include the collisions of asteroids with the Earth, or the Shoemaker-Levy 9 comet with Jupiter. In a future, the tables may be not needed if the cross-matches are computed on-the-fly, without the need to precompute and store them in a database.

```
aster_table
comet_table
ssoid
service_log
service_xmatch
xmatch_euclid
xmatch_herschel
xmatch_hst
xmatch_jwst
xmatch_spitzer
xmatch_xmm
count_metadata
count_table
card_reduction_euclid
card_reduction_herschel
card_reduction_hst
card_reduction_jwst
card_reduction_spitzer
card_reduction_xmm
close_encounter
```

The service log table contains the log messages of the web service included in the pipeline. The service xmatch table includes all the cross-matches of the other tables in one, with an additional column reporting if the calculation were done with JPL elements or not. Card reduction tables are generally no longer used, since they require a lot of space and they are related to ESASky.

3.4 Pipeline execution and summary of the results

With the instructions above it will be straightforward to execute the pipeline for a given action. Multiple properties files can be prepared to execute the pipeline with a given action and a particular set of properties. The logs can be written to the console, as in the properties above, or to an external file. In the logs you will see a report of all observations being procesed by the pipeline, some of them very fast, and others slower in case there are many potential asteroids in that region, or some comet needs to be integrated. An additional little report is written each time another 1000 observations are processed, providing the number of cross-matches found until that moment and an estimate of the time pending to complete the cross-match process for all observations.

3.4.1 Basic commands to access the documentation

The commands below can be used to access the documentation of SSOXmatch: user manual, the Paper by Alonso-Albi (2015), and the technical documentation of the javadoc.

```
    java –jar SSOXmatch.jar –action doc

    java –jar SSOXmatch.jar –action paper

    java –jar SSOXmatch.jar –action javadoc
```

3.4.2 Basic commands to update files

The following command can be used to update the list of observations. This will take considerable time for the Hubble archive. Other update options include UpdateElements, UpdateKernels, UpdateDatabase.

```
    java –jar SSOXmatch.jar –environment custom.properties ––update –action UpdateObservations
```

3.4.3 Basic commands to compute ephemerides

The pre-integration process is not required to compute ephemerides. The following commands will copy the default properties for the local environment to a file named custom.properties, that can later by edited. Then the Ephem action can be called with an input similar to the inputs shown above for the comparison of SSOXmatch with Horizons and Miriade. More than one input between quotes can be added. Another option is to use as input an URL similar to the one accepted by Horizons when the elements of the body are provided, although not all options accepted by Horizons have been implemented in SSOXmatch.

```
    java –jar SSOXmatch.jar –environment local > custom.properties

    vi custom.properties

    java –jar SSOXmatch.jar –environment custom.properties –action Ephem "2458184.5, UTC, 2020 CD3,
        500@EARTH" "2020–03–06 00:00:00, UTC, Moon, 990@EARTH"

    java –jar SSOXmatch.jar –environment custom.properties –action Ephem
"https://ssd.jpl .nasa.gov/api/horizons.api?format=text&COMMAND=%27%3B%27&MAKE_EPHEM=%27YES%27&
EPHEM_TYPE=%27VECTOR%27&OUT_UNITS=%27KM–S%27&CENTER=%27@SSB%27&ANG_FORMAT=
%27DEG%27&START_TIME=%27JD2450800.5%20TDB%27&STOP_TIME=%27JD2461000.5%27&
STEP_SIZE=%2710%20d%27&QUANTITIES=%271,9,20%27&EPOCH=2450800.5&
EC=0.1516829576363078&A=1.097853972438096&MA=149.0882670050723&
OM=22.630740813069860&W=58.07843630667180 4&IN=1.164548964501213&ECLIP=%27J2000%27&
```

```
VEC_CORR=%27NONE%27&OBJ_DATA=%27NO%27&A1=1.599229127169E−10&A2=−5.273644346744E−12&
A3=0.0&DT=0.0&R0=1.0&ALN=1.0&NM=2.0&NN=5.093&NK=0.0&REF_PLANE=FRAME&VEC_TABLE=2"
```

3.4.4 Basic commands to pre-integrate

To preintegrate the action to be used is PropagateOrbits. This may take days of CPU. Pre-integration is not incremental for JPL database and must be completed at once. It is incremental for Lowell database.

```
    java −Xmx18G −XX:+UseG1GC −jar SSOXmatch.jar −environment custom.properties −action PropagateOrbits
```

3.4.5 Basic commands to compute cross-matches

To compute cross-matches the first recommended command is the test over selected observations, to test the ingestion in a database in case this option is used. Another possibility is to compute cross-matches for a given observation (second line in the examples below). The third command will compute all cross-matches for the surveys selected in the properties file.

```
    java −jar SSOXmatch.jar −environment custom.properties −debugLevel 1 −−test

    java −jar SSOXmatch.jar −environment custom.properties −action Observation −observation jwst
         jw02361007001\_04104\_00004\_nrs2 −ingest

    java −Xmx18G −XX:+UseG1GC −jar SSOXmatch.jar −environment custom.properties −action IngestAll
```

3.4.6 Basic command to launch the web service

To launch the web service the action is Observation, but including as additional options -service, and -all to read during the startup all the observations from the supported and selected surveys.

```
    java −jar SSOXmatch.jar −environment custom.properties −action Observation −service −all
```

3.4.7 Summary of the results

Once everything has been computed and ingested sucessfully, maybe including the card-reduction tables, the Report action will be called. This will write to the console a summary of the results and the input used, with details about the rejected observations and the complete number of cross-matches found and in how many individual objects. Here is a report of the pipeline run in which the cross-matches for all surveys were computed:

```
Time cover window of the different kernels:
 GAIA: from JD 2456646.0 (2013−12−19 12:08:28) to JD 2460154.0 (2023−07−28 11:38:28)
 ISO: from JD 2450038.5 (1995−11−17 00:37:46) to JD 2451010.6 (1998−07−16 02:37:46)
 L2: not available
 TESS: from JD 2458227.5 (2018−04−18 23:37:00) to JD 2460153.0 (2023−07−27 11:37:00)
 HST: from JD 2448006.6 (1990−04−25 02:01:40) to JD 2460194.6 (2023−09−07 03:06:18)
 Chandra: from JD 2451397.8 (1999−08−07 07:32:00) to JD 2460153.0 (2023−07−27 11:32:00)
 Euclid: from JD 2460127.2 (2023−07−01 15:53:08) to JD 2463152.5 (2031−10−12 23:03:18)
 Integral: from JD 2452565.2 (2002−10−17 16:48:00) to JD 2460153.0 (2023−07−27 11:48:00)
 Herschel: from JD 2454966.1 (2009−05−14 13:40:11) to JD 2456474.5 (2013−06−30 23:40:11)
 IUE: from JD 2443536.3 (1978−01−27 18:03:44) to JD 2451752.4 (2000−07−26 21:24:39)
 XMM: from JD 2451522.6 (1999-12-10 02:38:00) to JD 2460153.0 (2023-07-27 11:38:00)
 Kepler: from JD 2454897.8 (2009-03-07 06:24:02) to JD 2458847.5 (2019-12-29 23:54:02)
 Spitzer: from JD 2452876.8 (2003-08-25 06:26:43) to JD 2457023.5 (2014-12-31 23:56:43)
 JWST: from JD 2459574.0 (2021-12-25 13:01:09) to JD 2460153.0 (2023-07-27 11:31:09)

Orbital elements of asteroids were updated for the last time for JD 2460100.5 (2023-06-05 00:00:00):
 There are 1297714 asteroids in the file , and 1297714 asteroids ingested in the database
 There are 2205 comets in the file , and 2205 comets ingested in the database
 There are a total of 1299952 body identifiers ingested in table ssoid

Number of observations accepted or rejected:
 HST (resource file): found 1180017 observations: 1059468 were added, 11 have wrong position or fov, 0
are outside kernel cover windows, 113058 were joined because of identical fov and dates, 7458 are too
short integrations over a extended period
 Herschel (resource file ): found 27103 observations: 27103 were added, 0 have wrong position or fov, 0
are outside kernel cover windows, 0 were joined because of identical fov and dates, 0 are too short
integrations over a extended period
 XMM (resource file): found 42339 observations (excluding 27688 duplications): 14400 were added, 251
have wrong position or fov, 0 are outside kernel cover windows, 0 were joined because of identical fov and
dates, 0 are too short integrations over a extended period
 Spitzer (resource file ): found 43788 observations: 30632 were added, 3 have wrong position or fov, 0
are outside kernel cover windows, 52 were joined because of identical fov and dates, 13101 are too short
integrations over a extended period
 JWST (resource file): found 117006 observations: 104506 were added, 0 have wrong position or fov, 0 are
outside kernel cover windows, 12481 were joined because of identical fov and dates, 19 are too short
integrations over a extended period
 Found a total of 1236109 different observations

Number of cross-matches found in the last execution:
 HST: 402719 cross-matches ingested for 31354 individual bodies
 Herschel: 634619 cross-matches ingested for 214698 individual bodies
 XMM: 25393 cross-matches ingested for 20561 individual bodies
 Spitzer: 59177 cross-matches ingested for 12681 individual bodies
 JWST: 19080 cross-matches ingested for 1173 individual bodies

Main configuration parameters:
 Database connection (user postgres): jdbc:postgresql://localhost:5432/sso?searchpahsso_all2
 Local folder: /home/talonso//doc/SSO_asteroids/newPipeline/
 Kernel for JPL DE integration: de440s.bsp
 days_extrapolate_position_using_velocity = 1
 incremental_cross_matches_nObs = 0
 override_ceu_comets_with_integration_uncertainty = 0.01
 limiting_magnitude = 34
 max_obs_extension_days = 30
 max_obs_extension_relative_to_duration = 20
 max_obs_field_deg = 25
 max_position_uncertainty = 20r,1
 use_numerical_integration = true
 use_pluto_from_orbital_elements = false
 allow_quick_discard_cross_matches = true
 cross_match_threads = 15
```

The report corresponds to a complete run on October 22, 2023, with the elements updated up to around July, 10 (asteroids), and September 18 (JPL comets database). It is interesting to see the results of the rejected observations, that shows for HST and Spitzer a minimum percentage of wrong data, and for

HST and JWST an important percentage (10%) of repeated observations, at least from the point of view of computing cross-matches. When the Ra, Dec are 0, 0 a recovery is tried using the information from the polygon field, considering those zero values are just missing information and not errors in the archive data. As a curiosity, there are observations rejected for HST (not visible in the table above) because they are outside the kernel time cover window, among them some observations listed that were taken apparently before the HST was in service or even launched. This is an evidence that the data in the archives is far for being perfect for the purpose of computing reliable cross-matches.

4 Version history

The **first release tagged as version 0.1.0** is marked by the commit 98dde9e93ee on March, 12, 2023. The order 8 Dormand-Prince (DP8) integration scheme was already developed but DP5 was the one tested and used to compute the first set of cross-matches, that were later integrated in ESA-Sky. The target of the first release was to mimic the implementation of Eproc, using a similar physical model for the Solar System (no perturbing asteroids, without Pluto, and Earth-Moon as a single body), and an integrator equivalent in terms of accuracy.

The **second version 0.2.0** is marked by the commit 9b7cc832d39 on July 20, 2023. Includes the following improvements:

- Default integration scheme changed to DP8 (order 8), supporting the variable time step (currently used for asteroids only, since for comets it is not safe enough). Accuracy tests run for all bodies to adjust the time step range for DP8, ensuring good accuracy always. Explicit order 10 methods were explored, but found to be inferior to DP8.
- Improved numerical integration by adding Earth-Moon as separated bodies, Pluto, and the four most massive asteroids as perturbers. DE440 is now used. It was found that 5% of the bodies are considerably affected by the main asteroids, around 15000 km after 20 years of integration. This can lead to several arcseconds of error in the positions from Earth or L2 point. The J2 oblateness calculation was also improved, supported for all bodies except Mars, Saturn and beyond.
- Changed the default comet catalog to that of JPL (cometpro still usable). Among other things, this means to increment the number of comets, get better orbits for many of them, and include cross-matches for the famous comet Shoemaker-Levy 9, that impacted Jupiter in 1994. New elements are more updated, although Horizons provides different elements for the different orbits of the periodic comets, and this is still not supported here.
- Although the first release was fast, this second one features a massive performance improvement. Among other details, the library jafama is used instead of FastMath class to perform fast and accurate trigonometric functions in all cases. The Kepler equation is solved approximately by using a precomputed 2-d matrix for mean anomaly and eccentricity, so that no Newton-Raphson iteration or trigonometric function evaluation is needed for an accuracy of few arcminutes. Only if the cross-match is possible an accurate solution is computed. The preintegration is now faster despite of using asteroids as perturbers, specially for comets and when writting the binary files of the preintegrated orbits. When computing cross-matches the ingestion of data in the card reduction tables is also a lot faster.
- Some kernels (Spitzer, Herschel) for the main missions supported have been reconstructed to reduce the file size around a factor 20, while keeping a similar accuracy. This contributes also to the speed improvement. XMM was also fixed from the methods developed in this software when needed, using also trajectory data from Horizons, since some inconsistencies were found in the original kernel. Added many other missions and kernels: ISO, IUE, Chandra, GAIA, Integral, Kepler, TESS, L2 point, but no cross matches can be computed because there are no observations downloaded

for them. From all them, the HST and IUE kernels are the only ones kept as the original ones downloaded and maintained externally (this is possible because they are orbiting the Earth), while the rest have been downloaded or created from this software. Akari, Suzaku, and CHEOPS are not supported since there is no trajectory publicly available for these missions. These kernels have no observations associated, so no cross matches can be computed, but may be used for a web server in the next version 0.3.0.

- Other minor improvements include to optionally use (from the properties) the analytical theories Mars07, L1, TASS 1.7, and GUST86 to compute the positions of all the moons of Mars, Jupiter, Saturn, and Uranus. The numerical integration for Phoebe and Nereid were improved, with the sampling reduced from 5 to 1 year. A new option in the properties allows to choose between computing the apparent magnitude of the nucleus of the comet, or the total magnitude including the tail. The problem with the missing XMM observations was also fixed.

- Implemented some JUnit tests.

It is important to note that due to these improvements, direct comparisons with the previous pipeline (as well as the Eproc software or the SkyBot service from IMCCE) are no longer meaningful. The integration scheme DP8 (superior to the BS1 method used in Eproc, according to K. Fox, Celestial Mechanics 33, 127-142 (1984), and O. Montenbruck, Celestial Dynamics 53, 59-69 (1992)), the number of bodies used in the numerical integration (specially the perturbing asteroids added), and the default JPL comet catalog used in this software, among other details, are too different. In addition, a new set of cross-matches computed with this package always means to recompute the ephemerides of all bodies, taking advantage of possible orbit improvements for many old bodies (remember the differences in the cross-matches found above for Herschel after only 4 years of evolution in the orbital elements). These new features are superior compared to the version 0.1.0 or the Eproc/SkyBot software packages, and are expected to provide many differences in the cross-matches, mainly the perturbing asteroids and the always updated computations for all bodies. From the ephemerides perspective, Horizons is the reference and some features on it are superior to this package, like the greater list of perturbing asteroids (16 according to the documentation of Horizons, only 4 used in version 0.2.0 due to the performance impact), and the use of different elements for minor bodies depending on the orbit (the NGF forces on comets due to the nucleus behavior can change with time).

The **third version 0.3.0** was finished on June 6, 2024. The practical use in ESA of the project was discarded because the implementation needs first an external verification or validation. Currently the code is not public.

- Default integrator changed back to the order 5th DP5(4). Additional schemas implemented, like some methods by Verner and Ono.

- Cross match detection logic changed from a geometrical approach to a more visual approach (cross match type 3 removed). The algorithm is more robust, and SVG images are generated for each individual cross-match.

- Implemented the visualization of cross-matches using sketch images showing the details of the body and the observation.

- Support for asteroids and comets from JPL database (besides Lowell and cometpro). JPL database includes non-standard NGF models for some asteroids (treated almost like comets), which explained most of the positional differences respect Horizons in some specific bodies. These models (Yarkowski effect) are now supported in the pipeline, very useful to obtain accurate positions for some NEOs.

- Support for collision detection, allowing to obtain cross-matches of the comet Shoemaker-Levy 9, that impacted Jupiter, and some NEOs that impacted Earth. They are only available in the JPL databases.

- Number of perturbers increased by default to 16 for consistency with Lowell and Horizons, after more improvements in speed. Now there is also a limiting magnitude different for each survey, which helps to increase the speed.
- Support for computing close approaches of minor bodies with planets, moons, and asteroids with mass.
- The consistency with Horizons is remarkable when integrating asteroids and comets with JPL elements. Some numerical tests available below. To reach this consistency, some parameters need to be touched depending on if Lowell or JPL elements are used, see some comments in the properties file listed above.
- Support for other numerical integrations, including DE200. Support for the Kuiper belt model introduced in DE440, which improves the overall consistency in raw tests (not perfect because of some lack of detaills about the JPL implementation), but it is not needed for computing cross-matches.
- A simple cross-match web service can be run from the command-line. It is just an example, no special optimizations are done (like reading in memory the files for the preintegration), so it is quite slow.
- Ephemerides for all spacecrafts and bodies from any other body can be computed from Java code or the command line.
- Large number of improvements and minor bug fixes due to massive testing and evolution, impossible to list them all.
- Code quality improved drastically: added many JUnit tests to get a global coverage close to 90%, solved many Sonar issues, javadoc technical documental, and better general documentation available from the command-line.

The **fourth release 0.4.0** was finished on September, 22, 2024. Main changes:

- The positions and the chart shown for cross-matches were sometimes slightly wrong, due to not always using numerical integration in the final step to report the results. This has been fixed.
- Fixed some errors in the calculations of the General Relativity for both the Damour-Deruelle and the complete EIH formulations, thanks to Larry Wasserman from Lowell Observatory for his help.
- Added an oblateness treatment to consider only the J2 term of the Earth, which is the way Horizons computes oblateness. This improved the consistency with Horizons.
- Implemented other integrators Bogacki-Shampine order 5th, Cash-Karp, and others which are better for stiff problems.
- The sketch images showed times in TT, this has been fixed to show them in UTC.
- Solved some memory issues. Now the memory used increase with time very slowly.
- Ephemerides calculations from any observatory on Earth supported. More consistent results with Horizons since the same precession model is now used.
- Fixed a problem when reading the list of observations for XMM. Solved an issue with the units of the duration in the list of Herschel observations. The JPL file of asteroids also had an issue with an empty value of G, now the default value 0.15 is used if empty. Fixed an issue identified with the A1-A3 values of the NGF parameters returned from the JPL Small Body database, slightly different from Horizons. The values downloaded are now automatically updated using Horizons.

- Changed the treatment of the CEU (estimate of the integration error), that was based in a simple calculation in arcseconds, without considering the distance between the body and the telescope. Now a more complete treatment is used based on the rms of the fit and the rate of increment equal to another rms when delta-t is equal to the time-arc the body was observed. This is also scaled for the distance of the body respect the distance it had respect Earth when the orbit was computed. Same treatment for Lowell and JPL elements.
- Compiled a catalog of 850 asteroids that had close encounters with other bodies with mass. For them the SPK kernel was downloaded from Horizons, and they are used for pre-integration and ephemerides, so that the output matches Horizons perfectly for the time interval of validity of the kernels (6 months around the close encounter dates).

The **fifth release 0.5.0** was finished on February, 8, 2025. Main changes:

- Paper in draft status: https://papers.ssrn.com/sol3/papers. cfm?abstract_id=5127618
- Implemented a complete online tool with six services that can be used to query cross-matches and ephemerides. It is available at . It is possible to compare directly the results of the ephemerides with similar external services like Horizons (JPL), Miriade (IMCCE), and the cross-matches with the small-body identification tool (JPL), and SkyBoT (IMCCE).
- Support for ARM archiquecture in SPICE, for Linux and Mac systems.
- Improvements in some computations: the axes of the Earth, Moon, and Sun for the oblateness correction, the General Relativity, particular models for some bodies that were difficult to spot since the model parameters are not present in the JPL Small Body database, and the position uncertainty from the CEU parameters. See Paper for more information.
- Added an specific kernel to solve the vectors for all perturbing asteroids, so the initial conditions of the integrations are computed a lot faster.
- Support for Euclid survey, added a limited set of raw observations.
- Added an optional DSS background image to the sketch images in the cross-matches, showing the background sky on the observation.
- Added more columns in the output to have a more complete information in the cross-match tables.
- Multiple speed and memory optimizations when reading observations and elements, among many minor bug fixes.

5 Future work

The current development version is 0.6.0-dev. It contains a bug fix affecting the topocentric ephemerides of natural moons, and will contain improvements in the readability and coverage of some parts of the code.

The file todoNotes.txt contains some comments with other relevant pending tasks in the project, among them:

- Make the web service VO compliant.
- More consistency with DE440 when using the KBO ring model.
- More consistency with DE432 with tesseral harmonics of the Moon, replacing the secular correction or minimizing it. Use JPL header variables.

- Coverage and Sonar: ObservationElement (drawObservationOutline), ComputeAndIngest, MinorBodyPropagator (main, readElements, ...).
- Explore and discuss other improvements: support observations from more missions, Herschel flux and limiting magnitudes for particular instruments, reduce the size of the card-reduction tables, among others.
- To prepare a tech-talk could be nice to explore and discuss other possible use cases with the community.

6 Visual examples of cross-matches

6.1 Example outputs from the new SSO pipeline

Some examples of sketches of cross-matches are shown in this section.

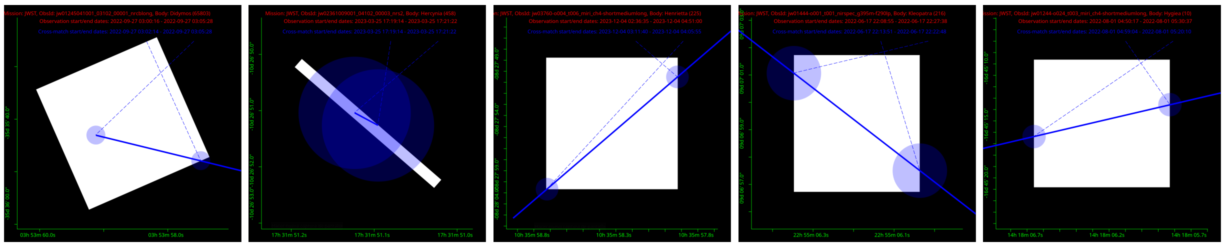


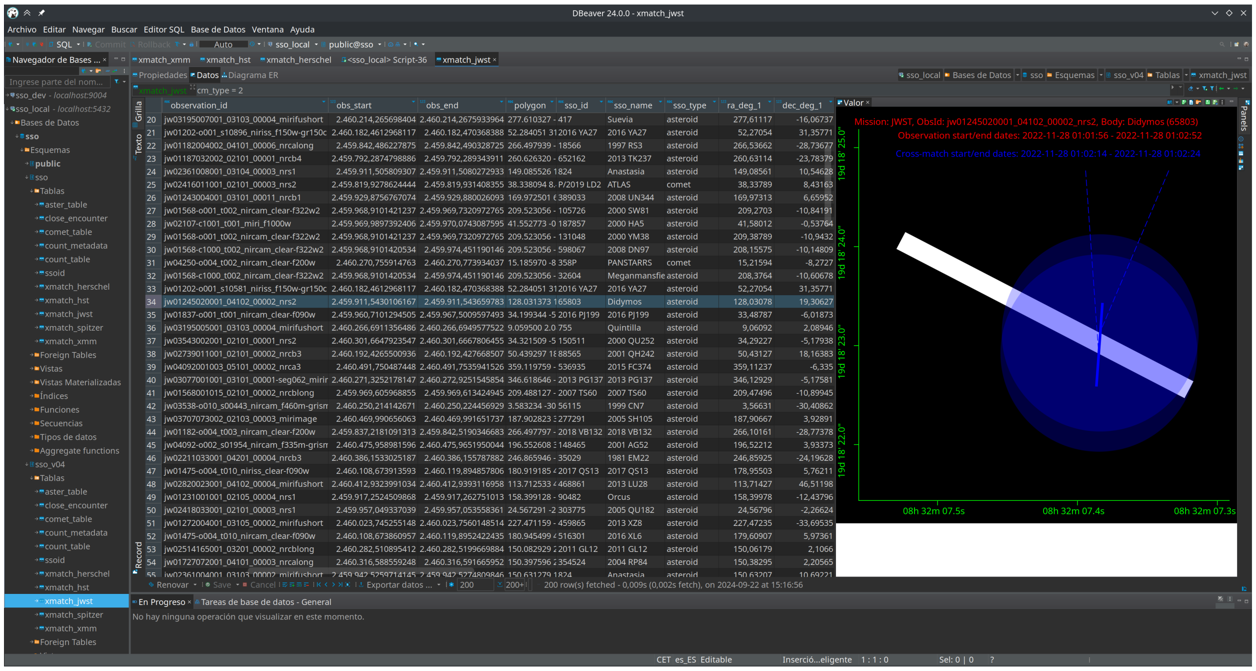
Figure 1: Left: Example of the chart generated with the pipeline of the cross-match of asteroid Didymos with JWST observation jw01245041001_03102_00001_nrcblong. The asteroid passed at only 0.074 AU (11 million km) from the JWST. The blue circles represent the error area estimated for the position of the asteroid, although this estimate is usually excesive. Other examples appear next to the right.

7 Differences between the implemented Java propagator and standard applications

7.1 Propagators available for integrating the Solar System bodies

It is difficult to define what is a standard application in the field of numerical integration, since there are many specialized tools for different purposes, like propagation of Earth satellites, evolution of galaxies, and so on. Respect to the numerical integration of Solar System objects, the tools that were found are shown in the table below.

There is basically no standard, since all these applications have been developed in an independent way, in different languages, and tested mostly internally with unknown accuracy respect the JPL ephemerides. The only possible exception to this could be ASSIST, that includes detailed comparison against the JPL integrator in its reference document. Respect documentation, again ASSIST seems slightly better than the others, but the effort needed to use any of these pieces of software for cross-matching asteroids is probably very high (for instance ASSIST is designed for an unique test particle). The MERCURY integrator is not recommended due to how it deals with some of the corrections to compute ephemerides of minor bodies. Conclusion could be that the option of using the IMCCE integrator (Eproc), basically the old pipeline approach, is the only acceptable alternative to the new pipeline, since others would require a much higher effort. The new Java pipeline is, overall, a better option respect the possible benefit in accuracy from


DBeaver 24.0.0 - xmatch_jwst
Archivo Editar Navegar Buscar Editor SQL Base de Datos Ventana Ayuda
Commit Rollback Auto sso_local public@sso
Navegador de Bases ...
xmatch_xmm xmatch_hst xmatch_herschel <sso_local> Script-36 xmatch_jwst
Propiedades Datos Diagrama ER
sso_local Bases de Datos sso Esquemas sso_v04 Tablas xmatch_jwst
sso_dev - localhost:9004
sso_local - localhost:5432
Bases de Datos
sso
Esquemas
public
sso
Tablas
aster_table
close_encounter
comet_table
count_metadata
count_table
ssoid
xmatch_herschel
xmatch_hst
xmatch_jwst
xmatch_spitzer
xmatch_xmm
Foreign Tables
Vistas
Vistas Materializadas
Índices
Funciones
Secuencias
Tipos de datos
Aggregate functions
sso_v04
Grilla
Texto
Record
Valor
Panels
19d 18' 25.0"
19d 18' 24.0"
19d 18' 23.0"
19d 18' 22.0"
08h 32m 07.5s
08h 32m 07.4s
08h 32m 07.3s
Renovar Save Cancel Exportar datos ... 200 200+
En Progreso
Tareas de base de datos - General
CET es_ES Editable Inserció...eligente 1 : 1 : 0 Sel: 0 | 0

xmatch_jwst cm_type = 2

	observation_id	obs_start	obs_end	polygon	sso_id	sso_name	sso_type	ra_deg_1	dec_deg_1
20	jw03195007001_03103_00004_mirifushort	2.460.214,265698404	2.460.214,2675933964	277.610327 -	417	Suevia	asteroid	277,61117	-16,06737
21	jw01202-o001_s10896_niriss_f150w-gr150c	2.460.182,4612968117	2.460.182,470368388	52.284051 31	2016 YA27	2016 YA27	asteroid	52,27054	31,35771
22	jw01182004002_04101_00006_nrcalong	2.459.842,486227875	2.459.842,490328725	266.497939 -	18566	1997 RS3	asteroid	266,53662	-28,73677
23	jw01187032002_02101_00001_nrcb4	2.459.792,2874798886	2.459.792,289343911	260.626320 -	652162	2013 TK237	asteroid	260,63114	-23,78379
24	jw02361008001_03104_00003_nrs1	2.459.911,505809307	2.459.911,5080272933	149.085526	1824	Anastasia	asteroid	149,08561	10,54628
25	jw02416011001_02101_00003_nrs2	2.459.819,9278624444	2.459.819,931408355	38.338094 8.	P/2019 LD2	ATLAS	comet	38,33789	8,43163
26	jw01243004001_03101_00011_nrcb1	2.459.929,8756767074	2.459.929,880026093	169.972501 6	389033	2008 UN344	asteroid	169,97313	6,65952
27	jw01568-o001_t002_nircam_clear-f322w2	2.459.968,9101421237	2.459.969,7320972765	209.523056 -	105726	2000 SW81	asteroid	209,2703	-10,84191
28	jw02107-c1001_t001_miri_f1000w	2.459.969,9897392406	2.459.970,0743087595	41.552773 -0	187857	2000 HA5	asteroid	41,58012	-0,53764
29	jw01568-o001_t002_nircam_clear-f322w2	2.459.968,9101421237	2.459.969,7320972765	209.523056 -	131048	2000 YM38	asteroid	209,38789	-10,9432
30	jw01568-c1000_t002_nircam_clear-f322w2	2.459.968,9101420534	2.459.974,451190146	209.523056 -	598067	2008 DN97	asteroid	208,15575	-10,14809
31	jw04250-o004_t002_nircam_clear-f200w	2.460.270,755914763	2.460.270,773934037	15.185970 -8	358P	PANSTARRS	comet	15,21594	-8,2727
32	jw01568-c1000_t002_nircam_clear-f322w2	2.459.968,9101420534	2.459.974,451190146	209.523056 -	32604	Meganmansfie	asteroid	208,3764	-10,60678
33	jw01202-o001_s10581_niriss_f150w-gr150c	2.460.182,4612968117	2.460.182,470368388	52.284051 31	2016 YA27	2016 YA27	asteroid	52,27054	31,35771
34	jw01245020001_04102_00002_nrs2	2.459.911,5430106167	2.459.911,543659783	128.031373 1	65803	Didymos	asteroid	128,03078	19,30627
35	jw01837-o001_t001_nircam_clear-f090w	2.459.960,7101294505	2.459.967,5009597493	34.199344 -5	2016 PJ199	2016 PJ199	asteroid	33,48787	-6,01873
36	jw03195005001_03103_00004_mirifushort	2.460.266,6911356486	2.460.266,6949577522	9.059500 2.0	755	Quintilla	asteroid	9,06092	2,08946
37	jw03543002001_02101_00001_nrs2	2.460.301,6647923547	2.460.301,6667806455	34.321509 -5	150511	2000 QU252	asteroid	34,29227	-5,17938
38	jw02739011001_02101_00002_nrcb3	2.460.192,4265500936	2.460.192,427668507	50.439297 18	88565	2001 QH242	asteroid	50,43127	18,16383
39	jw04092001003_05101_00002_nrca3	2.460.491,750487448	2.460.491,7535941526	359.119759 -	536935	2015 FC374	asteroid	359,11237	-6,335
40	jw03077001001_03101_00001-seg062_mirir	2.460.271,3252178147	2.460.272,9251545854	346.618646 -	2013 PG137	2013 PG137	asteroid	346,12929	-5,17581
41	jw01568001015_02101_00002_nrcblong	2.459.969,605968855	2.459.969,613424945	209.488127 -	2007 TS60	2007 TS60	asteroid	209,47496	-10,89945
42	jw03538-o010_s00443_nircam_f460m-grism	2.460.250,214142671	2.460.250,224456929	3.583234 -30	56115	1999 CN7	asteroid	3,56631	-30,40862
43	jw03707073002_02103_00003_mirimage	2.460.469,990656063	2.460.469,991651737	187.902823 3	277291	2005 SH105	asteroid	187,90667	3,92891
44	jw01182-o004_t003_nircam_clear-f200w	2.459.837,2181091313	2.459.842,5190346683	266.497797 -	2018 VB132	2018 VB132	asteroid	266,10161	-28,77378
45	jw04092-o002_s01954_nircam_f335m-grism	2.460.475,958981596	2.460.475,9651950044	196.552608 3	148465	2001 AG52	asteroid	196,52212	3,93373
46	jw02211033001_04201_00004_nrcb3	2.460.386,1533025187	2.460.386,155787882	246.865946 -	35029	1981 EM22	asteroid	246,85925	-24,19628
47	jw01475-o004_t010_niriss_clear-f090w	2.460.108,673913593	2.460.119,894857806	180.919185 4	2017 QS13	2017 QS13	asteroid	178,95503	5,76211
48	jw02820023001_04102_00004_mirifushort	2.460.412,9323991034	2.460.412,9393116958	113.712533 4	468861	2013 LU28	asteroid	113,71427	46,51198
49	jw01231001001_02105_00004_nrs1	2.459.917,2524509868	2.459.917,262751013	158.399128 -	90482	Orcus	asteroid	158,39978	-12,43796
50	jw02418033001_02101_00003_nrs1	2.459.957,049337039	2.459.957,053558361	24.567291 -2	303775	2005 QU182	asteroid	24,56796	-2,26624
51	jw01272004001_03105_00002_mirifushort	2.460.023,745255148	2.460.023,7560148514	227.471159 -	459865	2013 XZ8	asteroid	227,47235	-33,69535
52	jw01475-o004_t010_nircam_clear-f090w	2.460.108,673860957	2.460.119,8952422435	180.945499 4	516301	2016 XL6	asteroid	179,60907	5,97361
53	jw02514165001_03201_00002_nrcblong	2.460.282,510895412	2.460.282,5199669884	150.082929 2	2011 GL12	2011 GL12	asteroid	150,06179	2,1066
54	jw01727072001_04101_00003_nrcalong	2.460.316,588559248	2.460.316,591665952	150.397596 2	354524	2004 RP84	asteroid	150,38295	2,20565
55	jw02361004001_03103_00002_mirifushort	2.459.942,5259714145	2.459.942,5274809846	150.631279	1824	Anastasia	asteroid	150,63207	10,69221

200 row(s) fetched - 0,009s (0,002s fetch), on 2024-09-22 at 15:16:56

No hay ninguna operación que visualizar en este momento.

Mission: JWST, ObsId: jw01245020001_04102_00002_nrs2, Body: Didymos (65803)

Observation start/end dates: 2022-11-28 01:01:56 - 2022-11-28 01:02:52

Cross-match start/end dates: 2022-11-28 01:02:14 - 2022-11-28 01:02:24

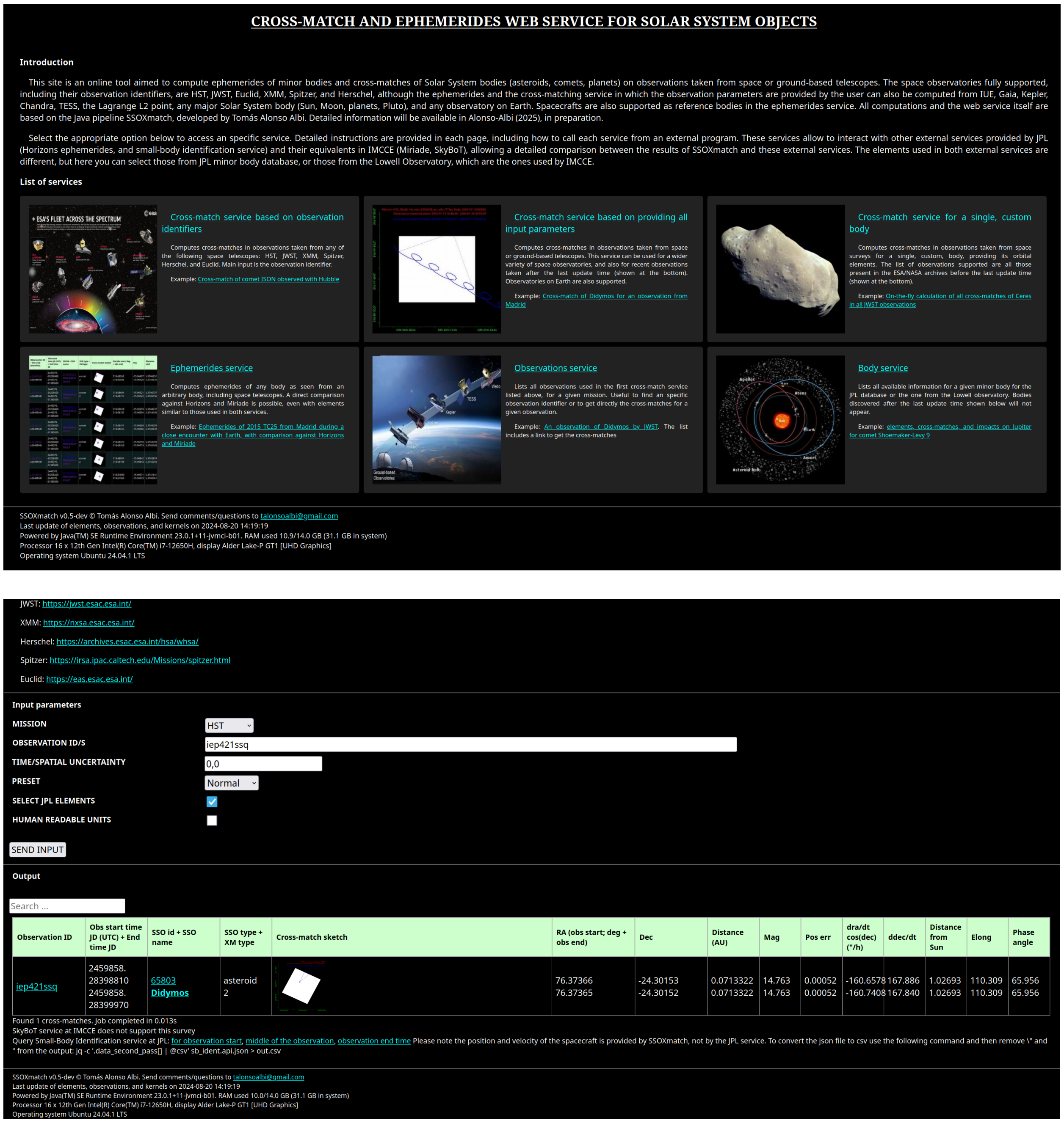
Figure 2: The program DBeaver showing the JWST cross-match table, and details for a cross-match of asteroid Didymos. DBeaver allows to search, filter, and explore the cross-matches found.

using ASSIST, since all the work is already done, and it is unknown how ASSIST would handle the huge magnitude of the computations involved.

7.2 Differences between the integrators

Although there is no standard, there is a standard set of physical effects/corrections that need to be implemented properly to get good results. Depending on the algorithms implemented, and if properly tested, the accuracy will vary in the different implementations, and also the propagator speed. The development of the Java integrator was done taking in consideration both points, with the goal of getting the maximum possible accuracy, but sacrifying speed as less as possible. So the purpose was not to always implement a given physical effect with the maximum possible accuracy, or at least to make that physical effect selectable in the implementation, with a default mode looking for an equilibrium between speed and accuracy. Other implementations may focus on accuracy, or on none of the two points, but speed is usually not a factor. If the purpose is to compute cross-matches of millions of bodies against millions of observations, the speed should be an important consideration.

Here we list the differences in the implementation between the different integrators, as far as the information is available. The list is restricted to the tools for which, directly or indirectly, some service, information or files were required to develop a new pipeline in Java. In bold face appear the features in which there are noticeable differences that can affect the results, and in red the critical practical features for the purpose of computing cross matches in the comming years.



CROSS-MATCH AND EPHEMERIDES WEB SERVICE FOR SOLAR SYSTEM OBJECTS

Introduction

This site is an online tool aimed to compute ephemerides of minor bodies and cross-matches of Solar System bodies (asteroids, comets, planets) on observations taken from space or ground-based telescopes. The space observatories fully supported, including their observation identifiers, are HST, JWST, Euclid, XMM, Spitzer, and Herschel, although the ephemerides and the cross-matching service in which the observation parameters are provided by the user can also be computed from IUE, Gaia, Kepler, Chandra, TESS, the Lagrange L2 point, any major Solar System body (Sun, Moon, planets, Pluto), and any observatory on Earth. Spacecrafts are also supported as reference bodies in the ephemerides service. All computations and the web service itself are based on the Java pipeline SSOXmatch, developed by Tomás Alonso Albi. Detailed information will be available in Alonso-Albi (2025), in preparation.

Select the appropriate option below to access an specific service. Detailed instructions are provided in each page, including how to call each service from an external program. These services allow to interact with other external services provided by JPL (Horizons ephemerides, and small-body identification service) and their equivalents in IMCCE (Miriade, SkyBoT), allowing a detailed comparison between the results of SSOXmatch and these external services. The elements used in both external services are different, but here you can select those from JPL minor body database, or those from the Lowell Observatory, which are the ones used by IMCCE.

List of services

Cross-match service based on observation identifiers

Computes cross-matches in observations taken from any of the following space telescopes: HST, JWST, XMM, Spitzer, Herschel, and Euclid. Main input is the observation identifier.

Example: Cross-match of comet ISON observed with Hubble

Cross-match service based on providing all input parameters

Computes cross-matches in observations taken from space or ground-based telescopes. This service can be used for a wider variety of space observatories, and also for recent observations taken after the last update time (shown at the bottom). Observatories on Earth are also supported.

Example: Cross-match of Didymos for an observation from Madrid

Cross-match service for a single, custom body

Computes cross-matches in observations taken from space surveys for a single, custom, body, providing its orbital elements. The list of observations supported are all those present in the ESA/NASA archives before the last update time (shown at the bottom).

Example: On-the-fly calculation of all cross-matches of Ceres in all JWST observations

Ephemerides service

Computes ephemerides of any body as seen from an arbitrary body, including space telescopes. A direct comparison against Horizons and Miriade is possible, even with elements similar to those used in both services.

Example: Ephemerides of 2015 TC25 from Madrid during a close encounter with Earth, with comparison against Horizons and Miriade

Observations service

Lists all observations used in the first cross-match service listed above, for a given mission. Useful to find an specific observation identifier or to get directly the cross-matches for a given observation.

Example: An observation of Didymos by JWST. The list includes a link to get the cross-matches

Body service

Lists all available information for a given minor body for the JPL database or the one from the Lowell observatory. Bodies discovered after the last update time shown below will not appear.

Example: elements, cross-matches, and impacts on Jupiter for comet Shoemaker-Levy 9

SSOXmatch v0.5-dev © Tomás Alonso Albi. Send comments/questions to talonsoalbi@gmail.com
Last update of elements, observations, and kernels on 2024-08-20 14:19:19
Powered by Java(TM) SE Runtime Environment 23.0.1+11-jvmci-b01. RAM used 10.9/14.0 GB (31.1 GB in system)
Processor 16 x 12th Gen Intel(R) Core(TM) i7-12650H, display Alder Lake-P GT1 [UHD Graphics]
Operating system Ubuntu 24.04.1 LTS

JWST: https://jwst.esac.esa.int/

XMM: https://nxsa.esac.esa.int/

Herschel: https://archives.esac.esa.int/hsa/whsa/

Spitzer: https://irsa.ipac.caltech.edu/Missions/spitzer.html

Euclid: https://eas.esac.esa.int/

Input parameters

MISSION	HST
OBSERVATION ID/S	iep421ssq
TIME/SPATIAL UNCERTAINTY	0,0
PRESET	Normal
SELECT JPL ELEMENTS	✓
HUMAN READABLE UNITS	

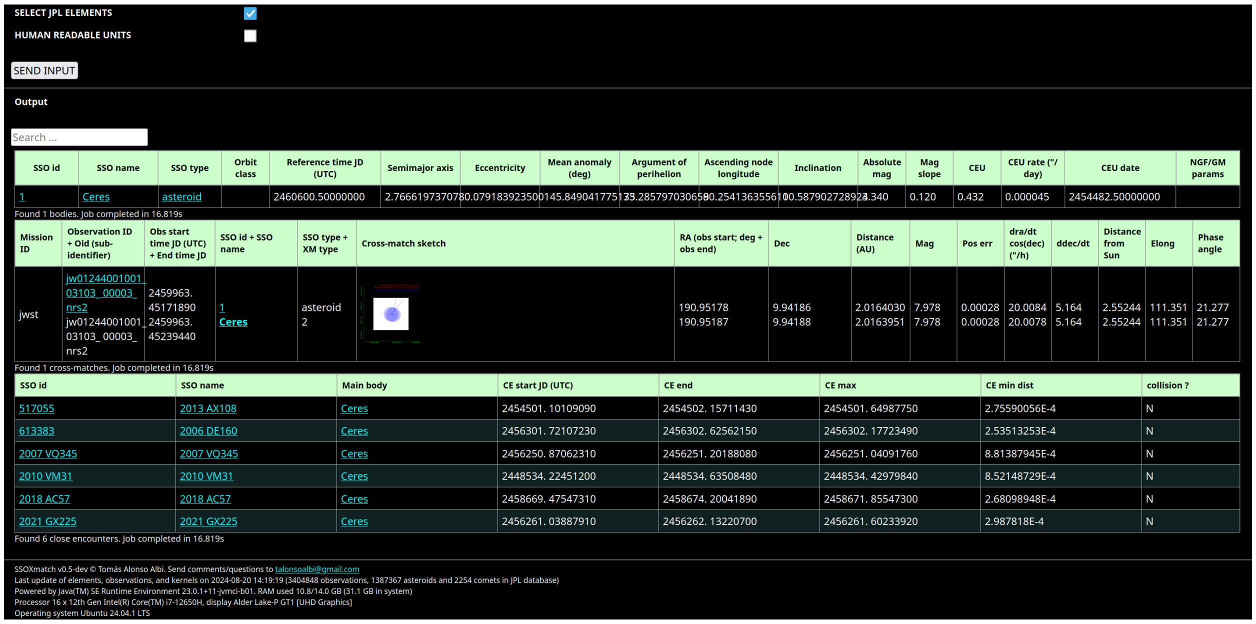
Output

Observation ID	Obs start time JD (UTC) + End time JD	SSO id + SSO name	SSO type + XM type	Cross-match sketch	RA (obs start; deg + obs end)	Dec	Distance (AU)	Mag	Pos err	dra/dt cos(dec) ("/h)	ddec/dt	Distance from Sun	Elong	Phase angle
iep421ssq	2459858. 28398810 2459858. 28399970	65803 **Didymos**	asteroid 2		76.37366 76.37365	-24.30153 -24.30152	0.0713322 0.0713322	14.763 14.763	0.00052 0.00052	-160.6578 -160.7408	167.886 167.840	1.02693 1.02693	110.309 110.309	65.956 65.956

Found 1 cross-matches. Job completed in 0.013s
SkyBoT service at IMCCE does not support this survey
Query Small-Body Identification service at JPL: for observation start, middle of the observation, observation end time Please note the position and velocity of the spacecraft is provided by SSOXmatch, not by the JPL service. To convert the json file to csv use the following command and then remove \" and " from the output: jq -c '.data_second_pass[] | @csv' sb_ident.api.json > out.csv

SSOXmatch v0.5-dev © Tomás Alonso Albi. Send comments/questions to talonsoalbi@gmail.com
Last update of elements, observations, and kernels on 2024-08-20 14:19:19
Powered by Java(TM) SE Runtime Environment 23.0.1+11-jvmci-b01. RAM used 10.0/14.0 GB (31.1 GB in system)
Processor 16 x 12th Gen Intel(R) Core(TM) i7-12650H, display Alder Lake-P GT1 [UHD Graphics]
Operating system Ubuntu 24.04.1 LTS

Figure 3: Top: main page of the online web tool (http://talonsoalbi.noip.me/XmMain) to compute cross-matches and ephemerides. Bottom: Cross-match of Didymos on a HST observation, the screenshot shows the bottom of the web page.

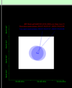


SELECT JPL ELEMENTS	✓
HUMAN READABLE UNITS	☐

Output

SSO id	SSO name	SSO type	Orbit class	Reference time JD (UTC)	Semimajor axis	Eccentricity	Mean anomaly (deg)	Argument of perihelion	Ascending node longitude	Inclination	Absolute mag	Mag slope	CEU	CEU rate ("/ day)	CEU date	NGF/GM params
1	Ceres	asteroid		2460600.50000000	2.766619737078	0.079183923500	145.849041775135	73.28579703065	80.25413635561	10.58790272892	3.340	0.120	0.432	0.000045	2454482.50000000	

Found 1 bodies. Job completed in 16.819s

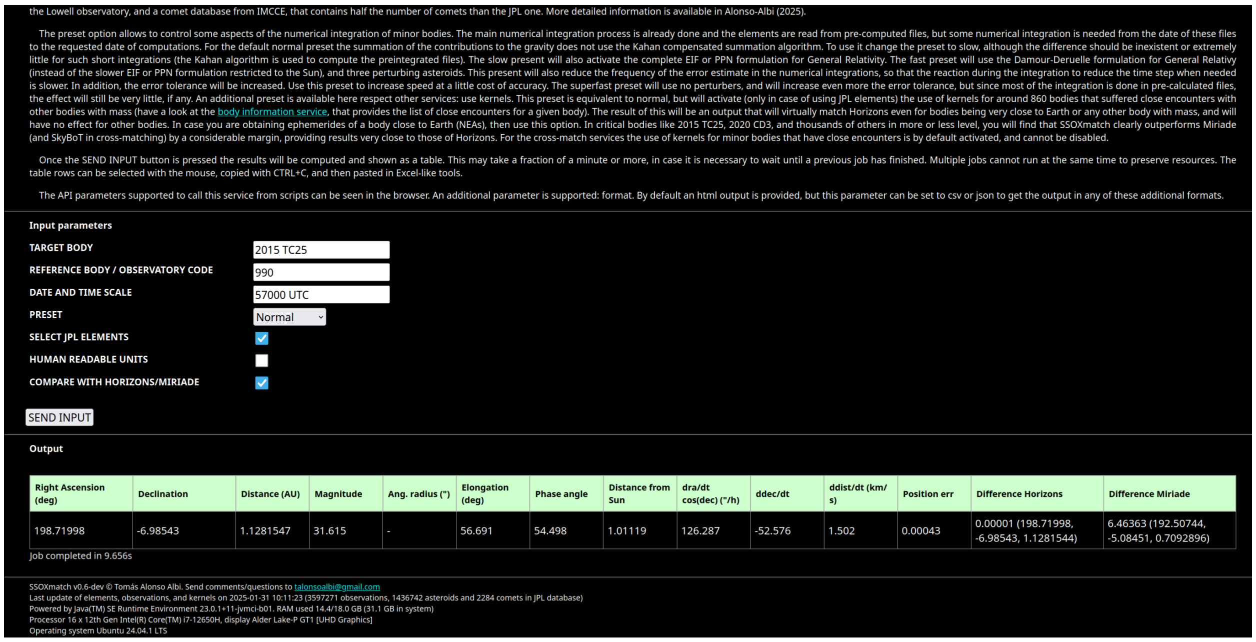
Mission ID	Observation ID + Oid (sub-identifier)	Obs start time JD (UTC) + End time JD	SSO id + SSO name	SSO type + XM type	Cross-match sketch	RA (obs start; deg + obs end)	Dec	Distance (AU)	Mag	Pos err	dra/dt cos(dec) ("/h)	ddec/dt	Distance from Sun	Elong	Phase angle
jwst	jw01244001001_03103_00003_nrs2 jw01244001001_03103_00003_nrs2	2459963.45171890 2459963.45239440	1 Ceres	asteroid 2		190.95178 190.95187	9.94186 9.94188	2.0164030 2.0163951	7.978 7.978	0.00028 0.00028	20.0084 20.0078	5.164 5.164	2.55244 2.55244	111.351 111.351	21.277 21.277

Found 1 cross-matches. Job completed in 16.819s

SSO id	SSO name	Main body	CE start JD (UTC)	CE end	CE max	CE min dist	collision ?
517055	2013 AX108	Ceres	2454501. 10109090	2454502. 15711430	2454501. 64987750	2.75590056E-4	N
613383	2006 DE160	Ceres	2456301. 72107230	2456302. 62562150	2456302. 17723490	2.53513253E-4	N
2007 VQ345	2007 VQ345	Ceres	2456250. 87062310	2456251. 20188080	2456251. 04091760	8.81387945E-4	N
2010 VM31	2010 VM31	Ceres	2448534. 22451200	2448534. 63508480	2448534. 42979840	8.52148729E-4	N
2018 AC57	2018 AC57	Ceres	2458669. 47547310	2458674. 20041890	2458671. 85547300	2.68098948E-4	N
2021 GX225	2021 GX225	Ceres	2456261. 03887910	2456262. 13220700	2456261. 60233920	2.987818E-4	N

Found 6 close encounters. Job completed in 16.819s

SSOXmatch v0.5-dev © Tomás Alonso Albi. Send comments/questions to talonsoalbi@gmail.com
Last update of elements, observations, and kernels on 2024-08-20 14:19:19 (3404848 observations, 1387367 asteroids and 2254 comets in JPL database)
Powered by Java(TM) SE Runtime Environment 23.0.1+11-jvmci-b01. RAM used 10.8/14.0 GB (31.1 GB in system)
Processor 16 x 12th Gen Intel(R) Core(TM) i7-12650H, display Alder Lake-P GT1 [UHD Graphics]
Operating system Ubuntu 24.04.1 LTS



the Lowell observatory, and a comet database from IMCCE, that contains half the number of comets than the JPL one. More detailed information is available in Alonso-Albi (2025).

The preset option allows to control some aspects of the numerical integration of minor bodies. The main numerical integration process is already done and the elements are read from pre-computed files, but some numerical integration is needed from the date of these files to the requested date of computations. For the default normal preset the summation of the contributions to the gravity does not use the Kahan compensated summation algorithm. To use it change the preset to slow, although the difference should be inexistent or extremely little for such short integrations (the Kahan algorithm is used to compute the preintegrated files). The slow present will also activate the complete EIF or PPN formulation for General Relativity. The fast preset will use the Damour-Deruelle formulation for General Relativy (instead of the slower EIF or PPN formulation restricted to the Sun), and three perturbing asteroids. This present will also reduce the frequency of the error estimate in the numerical integrations, so that the reaction during the integration to reduce the time step when needed is slower. In addition, the error tolerance will be increased. Use this preset to increase speed at a little cost of accuracy. The superfast preset will use no perturbers, and will increase even more the error tolerance, but since most of the integration is done in pre-calculated files, the effect will still be very little, if any. An additional preset is available here respect other services: use kernels. This preset is equivalent to normal, but will activate (only in case of using JPL elements) the use of kernels for around 860 bodies that suffered close encounters with other bodies with mass (have a look at the body information service, that provides the list of close encounters for a given body). The result of this will be an output that will virtually match Horizons even for bodies being very close to Earth or any other body with mass, and will have no effect for other bodies. In case you are obtaining ephemerides of a body close to Earth (NEAs), then use this option. In critical bodies like 2015 TC25, 2020 CD3, and thousands of others in more or less level, you will find that SSOXmatch clearly outperforms Miriade (and SkyBoT in cross-matching) by a considerable margin, providing results very close to those of Horizons. For the cross-match services the use of kernels for minor bodies that have close encounters is by default activated, and cannot be disabled.

Once the SEND INPUT button is pressed the results will be computed and shown as a table. This may take a fraction of a minute or more, in case it is necessary to wait until a previous job has finished. Multiple jobs cannot run at the same time to preserve resources. The table rows can be selected with the mouse, copied with CTRL+C, and then pasted in Excel-like tools.

The API parameters supported to call this service from scripts can be seen in the browser. An additional parameter is supported: format. By default an html output is provided, but this parameter can be set to csv or json to get the output in any of these additional formats.

Input parameters

TARGET BODY	2015 TC25
REFERENCE BODY / OBSERVATORY CODE	990
DATE AND TIME SCALE	57000 UTC
PRESET	Normal
SELECT JPL ELEMENTS	✓
HUMAN READABLE UNITS	☐
COMPARE WITH HORIZONS/MIRIADE	✓

Output

Right Ascension (deg)	Declination	Distance (AU)	Magnitude	Ang. radius (")	Elongation (deg)	Phase angle	Distance from Sun	dra/dt cos(dec) ("/h)	ddec/dt	ddist/dt (km/s)	Position err	Difference Horizons	Difference Miriade
198.71998	-6.98543	1.1281547	31.615	-	56.691	54.498	1.01119	126.287	-52.576	1.502	0.00043	0.00001 (198.71998, -6.98543, 1.1281544)	6.46363 (192.50744, -5.08451, 0.7092896)

Job completed in 9.656s

SSOXmatch v0.6-dev © Tomás Alonso Albi. Send comments/questions to talonsoalbi@gmail.com
Last update of elements, observations, and kernels on 2025-01-31 10:11:23 (3597271 observations, 1436742 asteroids and 2284 comets in JPL database)
Powered by Java(TM) SE Runtime Environment 23.0.1+11-jvmci-b01. RAM used 14.4/18.0 GB (31.1 GB in system)
Processor 16 x 12th Gen Intel(R) Core(TM) i7-12650H, display Alder Lake-P GT1 [UHD Graphics]
Operating system Ubuntu 24.04.1 LTS

Figure 4: Top: Body information service showing the elements, cross-matches, and close-encounters with massive bodies of Ceres. Bottom: Ephemerides for the close encounter with Earth of 2015 TC25, including the discrepancies respect Horizons and Miriade on the last columns of the table.

7.3 Comparison tests

7.3.1 Comparing the results of the Java pipeline with JPL integration

Here is a partial raw output of the integrator implemented, with a comparison with JPL DE432 ephemerides. The pipeline generally uses DE440, but DE432 has some advantages when comparing raw output with

Name	Programming language	Web page/service	References	Source code
JPL ephemerides / Horizons	Fortran 77	Horizons, Small body identification tool	Park (2021)	No
IMCCE's Eproc library / SkyBot service	Fortran 77	SkyBoT Eproc	Berthier (2006)	No (login required, but we have it due to a collaboration)
Propagator of the Lowell Observatory (ASTORB)	Fortran 77	https://asteroid.lowell.edu/	https://www.sciencedirect.com/science/article/pii/S2213133722000750?via%3Dihub	No
ASSIST	C and Python	ASSIST is based on REBOUND	Holman (2023), Rein (2012)	ASSIST repository REBOUND repository
OpenOrb	Fortran	No	https://digitalcommons.library.arizona.edu/objectviewer?o=uadc://azu_maps/Volume44/Number12/31a2f109-42d4-4e69-8ae8-c47fdb041370	https://github.com/oorb/oorb
FindOrb	C++	No	No	https://github.com/Bill-Gray/find_orb/tree /master
Augmented MERCURY integrator	Fortran	No	https://articles.adsabs.harvard.edu/pdf/2022SerAJ.204...51F	https://github.com/Fenu24/mercury
Integrator by S.L. Moshier	C++	http://www.moshier.net/ssystem.html	https://articles.adsabs.harvard.edu/pdf/1992A%26A...262..613M	http://www.moshier.net/de118i-2.zip

JPL/Horizons:

- The DE432 kernel that can be downloaded allows to integrate for several centuries, while the DE440 one is limited to around 100 years (unless a huge file is downloaded).
- DE432 does not include a model for the Kuiper belt and no trans-neptunian object is considered to have mass. So the physical model of the Solar System is more simple. This pipeline supports the physical model of DE440, but the implementation is still not fully consistent with JPL, so DE432 is a better choice to compare the accuracy of the integrator implemented.

```
Raw output of 250-yr integration (from JD 2460400.5) using the solver
BOGACKI_SHAMPINE_54, a tolerance parameter 1, and including all 343
asteroids as perturbers. The GR model EINSTEIN_INFELD_HOFFMAN_OPTIMIZED
was used, and the ACCURATE option for oblateness. Results labeled as
NBody are those from the Java pipeline. The integration time was 30h
with one thread. For consistency, the elements used were from JPL.

– Comparison with JPL DE432 (ecliptic J2000 coordinates) –
JD (TDB): 2551713.0 (91312.500 days from starting time)
– Comparison with JPL DE432 (ecliptic J2000 coordinates) –
JD (TDB): 2542857.5
SUN
x–y–z (JPL):        0.000301455538 −0.002652718141 −0.000036911274
vx–vy–vz (JPL):     0.000005505288 −0.000002334417 −0.000000106763
x–y–z (NBody):      0.000301455538 −0.002652718141 −0.000036911274
vx–vy–vz (NBody): 0.000005505288 −0.000002334417 −0.000000106763
MERCURY
x–y–z (JPL):        −0.119940511532 −0.452736949299 −0.025991574783
vx–vy–vz (JPL):     0.021530697492 −0.005889017358 −0.002444728685
x–y–z (NBody):      −0.119940510141 −0.452736949680 −0.025991574937
vx–vy–vz (NBody): 0.021530697515 −0.005889017274 −0.002444728680
VENUS
x–y–z (JPL):        0.526948518078 −0.504048408356 −0.037523110583
vx–vy–vz (JPL):     0.013823519209 0.014572757350 −0.000585579952
```

Feature	**Java pipeline**	**JPL ephemerides / Horizons**	**IMCCE (Eproc) + old pipeline (E. Racero)**	**Propagator of Lowell Observatory**
Integration schema	Selectable explicit Runge-Kutta with order between 3 and 12. Order 5 (Dormand-Prince 5(4)) by default (both accurate and fast) with error estimate to adapt the time step	Order 14th Adams-Bashforth-Moulton predictor-corrector, or 8th order Gauss-Jackson multi-step predictor-corrector ?	Bulirsh-Stoer Extrapolation BS1	Similar to the 8th order Gauss-Jackson multi-step predictor-corrector according to the paper
General relativity	Selectable between PPN and Damour-Deruelle. By default, an optimized version of the PPN formulation is used to better match Horizons without sacrifying speed	Einstein-Infeld-Hoffman (EIH or PPN formulation), restricted to the Sun for Horizons	Damour-Deruelle	Applied, but method not mentioned in the paper
Bodies with mass	Selectable, by default all planets, Moon, Pluto, and 16 asteroids. DE440 by default, but other JPL integrations can be used. All 343 asteroids are selectable. Kuiper belt objects and model are optional	All planets, Moon, Pluto, and 16 asteroids (Horizons service). 343 + 40 Kuiper belt objects (JPL ephemerides). DE440	Earth, Moon, and Pluto are not considered (the barycenter Earth-Moon is used). No asteroids. INPOP06 ?	All planets, Moon, Pluto, and 16 asteroids. DE440
Oblateness	Selectable, by default J2-J4 of Sun, Earth, Moon, and Jupiter. Approximations taken in the orientation of the bodies. Implemented a simple model to correct for secular oblateness effects, based on the Marsden one. Option for only J2 of Earth supported	Horizons only applies the J2 of Earth in the ephemerides	Not considered	Only J2 term for Earth
Non-gravitational forces (affect comets and a few hundred asteroids)	Marsden Model for comets including DT parameter (time offset for comet tail activity) Radiation pressure + Yarkovsky for asteroids using A values (model by Farnocchia et al. 2013 = Marsden model with $1/r^2$ law). NGF parameters from the JPL Small Body database, but corrected to match those of Horizons Simple model for Earth tides (DE200) to improve the position of the Moon. Planetary positions can be replaced by DE440 ones each n iterations, so this is not important	Marsden Model for comets including DT parameter Radiation pressure + Yarkovsky for asteroids using A values (Farnocchia et al. 2013 model) Evolved models for Earth tides, mantle cores, and some other minor effects	Marsden model for comets without DT parameter No No	No comets considered, Radiation pressure + Yarkovsky for asteroids using A values (Marsden model with $1/r^2$ law), No
Orbital elements	Selectable: astorb (Lowell) for asteroids + cometpro (IMCCE) for comets, or JPL for both	JPL for both	Astorb (Lowell) + cometpro (IMCCE)	No (fitted observations)
Close encounters / collisions computed ? (Shoemaker-Levy 9 with Jupiter)	Yes (when using JPL elements), but accuracy is below JPL in close encounters due to the incomplete (but fast) oblateness implementation. But a catalog of elements and specific kernels files can be used to avoid discrepancies with Horizons for 1000 bodies with close encounters	Yes	No. Comet Shoemaker-Levy 9 and many others not present in the cometpro database. Model not suitable for close encounters, many NEAs will deviate	No. Asteroids that collided with Earth are skipped from the astorb file. Oumuamua is not present.
Support for minimizing round-off errors (Kahan compensated summation)	Yes, different options to sum interactions ordered by magnitude, use Kahan summation, and Klein algorithm. Method by Kaun is the default	Not needed since extended precision arithmetics is used for JPL (Horizons ?)	No	? Probably no
Support for extended precision arithmetics	Started, limited to the main loop in the nbody code. Too slow for practical use, just to test the effects (DE405-like), although using the very fast Quadruple precision library https://github.com/m-vokhm/Quadruple	Yes for JPL ephemerides, but no for Horizons ?	No	?
Support for spacecraft kernels	Yes	Yes	Partial. HST supported, but others like JWST requires to modify the library with a customized Fortran code	Not applicable
Support for moons	Yes for the positions (only the most important moons: 2 for Mars, 4 for Jupiter, 9 for Saturn, 5 for Uranus, 2 for Neptune) Gravitational effects not supported	Yes (all) for positions and gravitational effects	Yes for the positions (same analytical methods used in the Java pipeline, and/or numerical integration for most dwarf satellites, not clear)	No (not considered in the integration)
Ready for future needs (speed, flexibility)	Yes	?	No	Evolving for this reason, but currently no
Preintegration needed	When elements are updated	When numerical integration version is updated	No (body by body based, very slow)	Not applicable
Support for different element sets	Two databases	Yes (time evolution)	No	No (unique fit / integration)
GPU acceleration	Discarded for now after tests with TornadoVM. Many iGPUs do not support double precision calculations, and cross-platform feature of Java is lost. Thread synchronization can be problematic	No	No	Currently no
Development tools, easy compilation, JUnit tests, profiling/optimization applied, Sonar quality, easy to maintain	Yes (scientifically/technically state-of-the-art algorithms, cross-platform, extremely optimized, >85% coverage with >100 JUnit tests, green quality gate, revised code style)	No (old algorithms)	No (old algorithms, bulky library applied for cross-matching, but not developed for it), hard and slow to use (manual steps)	No (old algorithms)

```
x−y−z (NBody):      0.526948518777 −0.504048407619 −0.037523110611
vx−vy−vz (NBody): 0.013823519188 0.014572757369 −0.000585579950
EARTH
x−y−z (JPL):          −0.148966604506 0.969459308732 −0.000576715289
vx−vy−vz (JPL):     −0.017284016923 −0.002681040292 0.000002034638
x−y−z (NBody):      −0.148966603285 0.969459308108 −0.000576714926
vx−vy−vz (NBody): −0.017284016987 −0.002681039773 0.000002034700
Moon
x−y−z (JPL):          −0.148987296167 0.966889887984 −0.000764523913
vx−vy−vz (JPL):     −0.016693349318 −0.002711627521 0.000030421627
x−y−z (NBody):      −0.148987489751 0.966889924075 −0.000764540695
vx−vy−vz (NBody): −0.016693343916 −0.002711671359 0.000030416572
MARS
x−y−z (JPL):          1.118329103786 −0.817816168146 −0.044053272297
vx−vy−vz (JPL):     0.008761443191 0.012512729639 0.000052578900
x−y−z (NBody):      1.118329104004 −0.817816167835 −0.044053272296
vx−vy−vz (NBody): 0.008761443188 0.012512729641 0.000052578900
JUPITER
x−y−z (JPL):          2.083883842450 4.578463400487 −0.066092421657
vx−vy−vz (JPL):     −0.006964242797 0.003486026238 0.000139905808
x−y−z (NBody):      2.083883841961 4.578463400731 −0.066092421647
vx−vy−vz (NBody): −0.006964242798 0.003486026237 0.000139905808
SATURN
x−y−z (JPL):          −6.279829816488 −7.670115537515 0.382580343680
vx−vy−vz (JPL):     0.004002589087 −0.003554732002 −0.000100171764
x−y−z (NBody):      −6.279829816370 −7.670115537625 0.382580343678
vx−vy−vz (NBody): 0.004002589087 −0.003554732002 −0.000100171764
URANUS
x−y−z (JPL):          12.344806827223 −15.535089248872 −0.216661059525
vx−vy−vz (JPL):     0.003051184220 0.002261517434 −0.000031083403
x−y−z (NBody):      12.344806827299 −15.535089248818 −0.216661059526
vx−vy−vz (NBody): 0.003051184220 0.002261517434 −0.000031083403
NEPTUNE
x−y−z (JPL):          −20.110016306032 22.346529391584 0.003390989223
vx−vy−vz (JPL):     −0.002351384621 −0.002078595949 0.000097028569
x−y−z (NBody):      −20.110016306125 22.346529391504 0.003390989227
vx−vy−vz (NBody): −0.002351384621 −0.002078595949 0.000097028569
Pluto
x−y−z (JPL):          −7.604875253136 −29.023152028746 5.302469098915
vx−vy−vz (JPL):     0.003103250146 −0.001321697646 −0.000756433106
x−y−z (NBody):      −7.604875252967 −29.023152028819 5.302469098873
vx−vy−vz (NBody): 0.003103250146 −0.001321697646 −0.000756433106
```

Before starting with the integration, with the setup completed, the relativistic position and velocity of the barycenter is computed, and, at the end, recomputed again. The positions of all bodies are then adjusted so that the barycenter of the entire system will remain at the same point as when the integration started. This fix is necessary to directly compare against JPL ephemerides with cartesian coordinates, and does not affect the output right ascension and declinations of the bodies or the cross-matches. When the physical model is identical and the numerical behavior of the code is optimal, the barycenter position should be the same before and after. In the numbers above it is evident the good behavior of the pipeline, since the position of the barycenter at the end of the integration is very similar. The change for DE440 is much larger, specially when the Kuiper belt bodies and rings are not included.

The maximum discrepancies with JPL integration after 250 yr are, in summary:

- Around 0.1 meter for the Sun.

- Around or less than 100 meter for all bodies except the Moon.
- Around 30 km for the Moon.

In the previous numbers the DE200 model for the Earth tides has been applied, without it the discrepancies for the Earth and Moon grow one order of magnitude (to 300 km and 1 km). The median of the discrepancies is around 70 m after 250 years of integration. These differences are in the level of some minor effects not considered in the model, like the Lense–Thirring effect, radiation pressure, the approximate treatment of oblateness (no precession/nutation), or to integrate the Moon with coordinates respect the Earth, among some others. In general, since the positions of all planets are computed from the JPL integration instead of propagated, the propagation is stricly limited to minor bodies. But if the option to propagate all the bodies from the physical model implemented is selected (without 'tricks'), the results will be similar to the ones above, and probably much better than what other packages would provide. In practice, also, only a few asteroids (16 by default, even in Horizons) are considered to have mass, not 343 as in the complete model, so calculations are faster and accurate enough. Considering the perturbations of at least the most massive asteroids is important for some particular bodies that get perturbed by them, including some comets. Regarding this point the old pipeline based on Eproc lacks some relevant elements of the physical model, among others neglecting the mass of the asteroids or considering the Earth-Moon as a single body, which limits its applicability to near-Earth asteroids. When no asteroids are considered in the previous test, the discrepancies with JPL increase between 3x and 10x depending on the body, which can be very acceptable, considering that the integration is completed in less than two minutes.

7.3.2 Comparing the results of the Java pipeline with SkyBot and Horizons

A comparison between this pipeline and the old one using Eproc was given above for Herschel and Spitzer observations. No more details can be provided until the cross-matches are re-computed with the old pipeline using an updated set of observations and orbital elements, something that will take a lot of time, or even may not happen. There is a JPL service to compute cross-matches, but it works slow, frequently with timeout errors for observations in the past.

Raw output of ephemerides can be obtained for Horizons, Miriade, and this pipeline. Some specific tests has been done with the following input and output:

```
// LIST OF TESTS
// Date, Time scale, body, origin/reference body, tolerance (deg), use elements in horizons (optional)
"2458184.5, UTC, 2020 CD3, 500@EARTH, 1E−2, true", // 2458184.5 = 2018 3 7. 2" = 0.0005 deg consistency
! for JPL elements and useElementsInHorizons
"2457000.5, UTC, 2015 TC25, 500@EARTH, 1E−2, true", // 1E−5 deg consistency !
"2449550.3, TDB, D/1993 F2−A, @JUPITER, 1E−4, true", // Same output from Horizons with true and false,
which is fine
"2020−03−06 00:00:00, UTC, Moon, 990@EARTH, 1E−5, false", // 990 = Madrid
"2020−03−06 00:00:00, UTC, Saturn, @hst, 1E−5",
"2020−03−06 00:00:00, UTC, Jupiter, @hst, 1E−5",
"2020−03−06 00:00:00, UTC, Callisto, @hst, 5E−5", // Set to use analythical theories for the moons
"2020−03−06 00:00:00, UTC, Callisto, @JUPITER, 0.005", // Only moons and minor bodies not supported as
reference bodies
"2020−03−06 00:00:00, UTC, Sun, 500@EARTH, 1E−5", // 500@EARTH = Geocenter, EARTH = center of mass
"2458184.5, UTC, hst, 500@EARTH, 3E−4",

* USING LOWELL ELEMENTS: first line propagating the elements of Horizons, second line propagating the
elements in the program database
Object     RA Horizons DEC (°) Distance (AU) RA SSOXmatch DEC  Distance   Discrepancy   RA Miriade DEC        Distance  Discrepancy
2020 CD3   313.39205   30.26124   0.00700        314.22982  29.95629   0.00679    0.78627        256.91092   −20.71962  0.29193   74.46518
2020 CD3   314.23376   29.95479   0.00678        314.22982  29.95629   0.00679    0.00373        256.91092   −20.71962  0.29193   74.87810

* USING JPL ELEMENTS: first line propagating the elements of Horizons, rest of lines propagating the
elements in the program database
Object     RA Horizons DEC (°) Distance (AU) RA SSOXmatch DEC  Distance   Discrepancy   RA Miriade DEC        Distance  Discrepancy
2020 CD3   313.39205   30.26124   0.00700        314.15880  29.98357   0.00681    0.71898        256.91092   −20.71962  0.29193   74.46518

2020 CD3   314.15996   29.98312   0.00681        314.15880  29.98357   0.00681    0.00110        256.91092   −20.71962  0.29193   74.84238
2015 TC25  198.71862   −6.98416   1.12813        198.71857  −6.98415   1.12813    0.00005        192.83818   −5.20331   0.73798   6.11213
D/1993 F2−A 22.14036  −79.49996  0.00075        22.14034   −79.49996  0.00075    0.00000
```

```
Moon       115.83099  22.49144  0.00246      115.83100  22.49144  0.00246   0.00000      115.83100  22.49144  0.00246   0.00001
Saturn     300.48224  −20.51250 10.67448     300.48224  −20.51250 10.67448  0.00000      300.48226  −20.51237 10.67452  0.00013
Jupiter    291.82474  −21.93695 5.70209      291.82474  −21.93695 5.70209   0.00000      291.82484  −21.93671 5.70212   0.00026
 Callisto  291.81494  −21.94168 5.71461      291.81495  −21.94168 5.71461   0.00001      291.81505  −21.94144 5.71465   0.00027
 Callisto  288.06910  −23.92641 0.01257      288.07434  −23.92545 0.01257   0.00489      291.81505  −21.94144 5.71465   3.97984
Sun        346.75057  −5.67404  0.99206      346.75057  −5.67404  0.99206   0.00000      346.75057  −5.67404  0.99206   0.00000
hst        149.78865  28.13512  0.00005      149.78891  28.13514  0.00005   0.00023

* With USE_CLOSE_ENCOUNTER_DATABASE (=> fix the integration with elements from the close encounter =>
use kernels and compare with raw output from Horizons):
2020 CD3   313.39205  30.26124  0.00700      313.39205  30.26124  0.00700   0.00001      256.91092  −20.71962 0.29193   74.46518
2015 TC25  198.71862  −6.98416  1.12813      198.71862  −6.98416  1.12813   0.00001      192.83818  −5.20331  0.73798   6.11213
```

The object 2020 CD3 is a critical example because it is a temporal natural satellite of Earth. The test corresponds to a date for which the body already had two close encounters with Earth when integrating with Lowell elements to the past during more than 2 years. Discrepancy with Horizons is around 0.8 degrees with Lowell elements, but goes down to 0.004 degrees when the elements of Lowell are used in the Horizons server to propagate them. This is a bad test, since Horizons uses a particular model for the gravity of Earth to compute the trajectory of 2020 CD3, not supported in this pipeline (that is the reason to force to use the elements of Lowell in Horizons, to remove this particular gravity model). When using the more Horizons-compatible JPL elements, the discrepancies are lower. The last results for 2020 CD3 and 2015 TC25 are computed replacing the JPL elements by a BSP kernel that reproduces the results from Horizons, including the particular gravity model (from a compiled catalog of close encounters), showing a perfect match with Horizons for all problematic bodies. To reach this level of consistency with Horizons, some options in the pipeline need to be specially tunned, for instance to disable the oblateness of all bodies except for the J2 of Earth. Note Miriade fails to compute a correct position, since the Earth/Moon system is considered a single body and the physical model of the Solar System used in that service is inadequate for other reasons also. This happens with other bodies since most of the new objects are little asteroids discovered when they are close to the Earth, so the physical model in Eproc is inadequate for such close asteroids. This is especially true when using the elements of Lowell as happens in Skybot/Miriade, since the elements are propagated by Lowell using a good physical model into the future (although the technical details of such model are not clear), and later propagated back in those services using a bad one, producing an inconsistent result when comparing with observations taken when the body was discovered. The elements of asteroids in the JPL database are kept for some critical bodies in the past, producing good results for calculations around when the body was discovered, independently of the integrator (provided the Earth-Moon system is described as two bodies on it). Note the flexibility and quality of this pipeline allows to derive details about the differences between the integrators used by Lowell observatory and by Horizons, although the specific options selected to better match their outputs are not, necessarily, the best values from the physical point of view.

Object D/1993 F2-A is the A-fragment of the Shoemaker-Levy 9 comet, with the ephemerides computed here right before the collision with Jupiter and after 1 year of integration moving close to it in a trajectory very different respect that of most other bodies. The result is a perfect match with Horizons (if the oblateness of Jupiter is disabled, considering only the J2 of Earth), quite remarkable considering how close the object was to Jupiter, with the comet entering from -45° of ecliptic latitude, and that the perturbations of the Jupiter moons are not included in the pipeline. Note Miriade cannot compute this event.

The calculations of the ephemerides of other bodies shows in general a perfect agreement of the Java pipeline with Horizons. Even the positions of the moons of Jupiter from Jupiter itself can be computed with nice accuracy.

8 Inputs required to execute the SSO pipeline

Most of the inputs required are included in the src/main/resources folder. They are listed in the table below.

File	Source/s
Orbital elements of asteroids	Lowell database ('astorb' file aster.dat): https://ftp.lowell.edu/pub/elgb/astorb.dat; JPL database (aster_jpl.csv): https://ssd-api.jpl.nasa.gov/sbdb_query.api?fields=pdes, name,epoch,a,ma,q,e,i,w,om,tp,H,G,A1,A2,A3,DT,equinox, orbit_id,first_obs,last_obs, data_arc,per_y,condition_code,rms,two_body,producer, spkid&amp;full-prec=true&amp;sb-kind=a&amp;www=1
Orbital elements of comets	Cometpro database (comet.dat): https://ftp.imcce.fr/pub/databases/cometpro /ELTNUM.TXT; JPL database (comet_jpl.csv): https://ssd-api.jpl.nasa.gov/ sbdb_query.api?fields=full_name,epoch,a,ma,q,e,i,w,om,tp,M1,K1,M2,K2,PC,A1,A2, A3,DT,equinox,orbit_id,first_obs,last_obs,data_arc,per_y,condition_code,rms,two_ body,producer, spkid&amp;full-prec=true&amp;sb-kind=c&amp;www=1
Other files in the elements folder of the resources	ObsCodes.txt: list of IAU observatory codes; eop.txt: Earth Orientation Parameters, used for an accurate UT1-UTC calculation, to compute ephemerides respect a given observatory
Non-gravitational parameters of the asteroids	asteroidNonG.csv. Can be updated from https://ssd-api.jpl.nasa.gov/sbdb_query.api?fields=full_name,A1,A2,A3,DT&amp;full−prec=false &amp; sb - cdata=%7B%22OR%22:%5B%22A2%7CNE%7C0%22,%22A3%7CNE%7C0 %22,%22A1%7CNE%7C0%22%5D%7D &amp;sb-kind=a&amp;www=1
Spice kernels for spacecrafts	Included in the resources, or downloaded externaly jwst.bsp, hst.bsp, ...
Observations for each mission supported currently (Spitzer, Herschel, XMM, HST, JWST, Euclid)	Different files that were included in the resources, but now downloaded externally. There are multiple URLs included in the properties file to download these files, or to query them from a database in the required format
Spice kernels for ephemerides	de440s.bsp, header.440, naif0012.tls. Kernels for other integrations supported: de200, de405, de430, de432
Dependencies (included in the lib folder, mandatory for Spice)	The Spice JNI library (JNISpice.jar, libJNISpice.so for Linux), that cannot be configure with Maven; jafama-2.3.2.jar for faster maths (may be removed in the future); postgresql-9.4-1208.jdbc41.jar to read/write from databases; jfreesvg-3.4.3.jar, to produce SVG images of the cross-matches; JUnit5 is also needed for the tests

When working with the pipeline the input parameters are contained in a properties file, and some examples of them are provided also in the resources. One of the parameters is the local folder to work in. On that folder, a directory structure identical to the one in the resources may be created, with the files mentioned above downloaded or copied from other sources, for instance updated with the provided URLs. The update of the files can be requested through the pipeline using the command line, or they are done automatically when the already available file is too old (configurable from a property, by default 180 days). In case any of those files are present in the local folder, that file will have preference if it is more recent. In case it is not present, the one in the resources (inside the jar file of the pipeline) will be read instead, except if it is too old and an update is required. The kernels need to be copied from inside the jar file to the file system, otherwise the Spice library cannot read them.

Respect the database, there is a file named tables.sql in the resources that is used by the pipeline to automatically create the required tables in the database configured in the properties file. This is done automatically or can be requested through the command line. The command line includes many other useful options (class Main.java in the pipeline). For instance, it is possible to copy the entire database to another server, useful when the cross-matches are computed in the local machine, validated, and then the task is to uppload the results to a development server.

Once the pipeline is configured with all required input files, the execution has two main steps: the pre-integration of all bodies to generate a set of binary files (in intervals of 73.05 days by default), and the calculation of the cross-matches for a number of missions. With the default configuration (favouring accuracy and not speed) and the latest list of observations (around 5 million in total), the first step requires around one week for Lowell elements (two for JPL ones) in a mini-pc working with 15 threads (of which only half of them contributes linearly to increase the speed of the machine), and the second around one month for the five missions supported: HST, XMM, Spitzer, Herschel, JWST, and Euclid. These numbers are for the default options detailed in the previous page, for instance using 16 asteroids with mass as perturbers, which is something that strongly affects the performance of an nbody code. The grid machine was found to be 3.5x faster, but allowing to execute two jobs or more at the same time, so the total time is around 3 and 10 days for the two steps. Another output of the first step is a set of close encounters and collisions that took place during the integration. The second step includes the ingestion of the results in the database, with the structure already detailed above.

There is another file named start.jsh in the resources, that shows how to start the pipeline and call some functions using JShell scripts. JSchell is equivalent to what Python provides, but with the advantages of the speed of Java. JShell scripts can be used to execute the pipeline with different .properties files, computing the cross-matches or whatever is needed and provided in the pipeline for different environments. JShell can be used with cron jobs for a simple automatic computation of cross-matches or ephemerides for different environments. The file ObsCodes.txt included in the elements folder, inside the resources, contains the list of IAU observatories, and from Java code or JShell it is possible to compute ephemerides respect any of them.

9 Cross-match requirements

9.1 Filtering of observations

The input list of observations for each mission is filtered to discard wrong or non-useful information. This process, apparently, was not done in the old pipeline, but it was found that some observations retrieved from the archive contain wrong or confusing information. Also, respect the purpose of computing cross-matches, two observations with the same fov and start/end times are, in practice, the same since the same cross-matches will be in both. In those cases the observations are grouped to make the calculations faster, ingesting later the cross-matches found for the group in each observation, repeated. This happens with 10% of the Hubble and JWST observations. Also, there is a tolerance angle so that all observations with the same start/end times pointing to a region very close in the sky, with a difference below 0.1 degrees, are also grouped. This includes images taken with different instruments with almost the same fov. Most of the processing for these observations are equivalent, so time can be saved on them. Here is a table with the summary of the actions done when reading and filtering the observations. In the section 3.4 a summary appear with the results of the observations skipped or grouped for each mission.

There are other details when trying to interpret the input data of an observation to try to fix possible wrong data. For instance, the list of observations for JWST contained, at least in the first months in which they were retrieved using a SQL query, Ra = Dec = 0 for all observations. In this case, those values were assumed to be lack of information, and computed to take the central point of the fov.

9.2 Filtering of minor bodies during the computation of cross-matches

The cross-matches are computed from the most recent observation to the oldest one. The process use a set of files with the preintegrated positions of all bodies, that needs to be generated each time the elements are updated. The preintegration is done up to five years into the future, so at least they should be updated in this interval, but usually the interval will/should be few months. This preintegration is unique for all missions (that is not the case in SkyBot, for instance), and an interval of 73.05 days (5 times per year) is selected by default. This means that the elements for a given minor body will be always referred to a time within +/- 37 days respect the instant of the observation. All the files/data preintegrated are around 27 GB in size, so although they are sequentially read one by one during the computation of the cross-matches, a web service may have all of them read into the RAM memory to provide an almost instantaneous output.

Another thing necessary to know to understand some details in the table below is that the calculation of the cross-matches is separated in windows of 1 day by default (configurable in the properties). In case an observation not filtered is spread over 3 days, the calculation of the cross-matches is done for each body 3 times, one for each day interval. The algorithm later accounts for the movement of the body in that time window to find all the cross-matches, if happened, and when within that window. It is important to note that SkyBot (or the equivalent service by NASA at https://ssd.jpl.nasa.gov/tools/sb_ident.html#/) does not allow that. In those services you search for cross-matches is a given circle of the sky for a given

Properties parameter	**Default value**	**Action**	**Reason**
max_obs_extension_days max_obs_extension_relative_to_duration	20 10	• Discard observations taken during many days • Discard observations in which the exposure time is much shorter than the period in which is spread	Both conditions applied with an AND logic. Filter observations that would contain many cross-matches with a very low probability to be present in the images. Example: Hubble Deep Fields, and many Spitzer observations
max_obs_field_deg	20	• Discard observations with a huge field of view	Those observations may contain an error in the fov. They may belong to mosaics, so the observing interval may be wrong or referred to the entire mosaic, when each part of the image was taken in different subintervals. Inadequate information to compute cross-matches.
		• Discard observations not covered by the kernel of the mission	If the kernel cannot compute the position of the telescope for the time of a given observation, for instance because the kernel was not updated, then discard that observation
		• Discard repeated observations	The observations contain an id and, sometimes also, an oid string value to identify them. When two or more observations are loaded with the same two values, only the first is considered (in order of time from older to newer ones)
		• Parsing errors	When the Ra, Dec, duration, start/end times, or fov contains invalid data, the observation is discarded
		• Coordinate errors	When the Ra, Dec reported in the observation is not a point located inside the fov, and the field radius is greater than 0.1 degrees (this failure is frequent in observations with very little fields, like planets). In practice, this error only occurs in observations with a fov <1 deg, since when constructing the observation object the Ra, Dec are corrected when needed for fovs <1 deg.

unique instant. Here, in the Java pipeline, the observation is considered to be spread between the start and end times, and all cross-matches on that time window are reported. In case the window is 1 day, in SkyBot it is necessary to call the service many times and elaborate the list of cross-matches manually, with the possibility of missing some if they happened during a little period. Respect the old pipeline, this limitation of computing the position of one body in one instant forced a complex two-step process to compute cross-matches, explained in the Section 3 of the paper by E. Racero et al. A HEALPix tesselation was used to calculate globaly the positions of the bodies, even if they would never appear on that field, and then in a second step the positions of the cross-matching candidates are computed in detail to confirm or discard the cross-match, using a complex set of geometrical algorithms. The process worked just fine, but could be considered, along with the practical limitations of calling Eproc through the command line, too slow to be adequate for the future, even in a grid machine. The approach in the Java pipeline is more 'direct', applying all possible speed improvements from the mathematical and geometrical point of view of the orbits, since absolute control is possible in the internal process. The cross-match is confirmed or discarded in case the body is found inside the field of view, whatever its shape. For that, the sky coordinates of the fov are transformed to 'pixel' coordinates using an stereographic sky projection, like if the intention is to draw everything on the screen. Then, a Java object is constructed to represent that shape, and native Java functions are used to calculate if the body is inside the shape or not, applying all necessary time subintervals to find if there is cross-match and the closest time, or if the position plus the uncertainty lies partially within the field (cross-match type 1 in the E. Racero paper). This approach solves pending issues in the old pipeline related to having the minor body close to the poles. Also, this approach means that cross-match type 3 (line between two positions crossing the fov) as defined in the paper by E. Racero does not exist in this pipeline, is already integrated in type 2 (direct detection of the cross-match in the field). In the tables generated by the pipeline new fields are included respect the old one, mentioning for each cross-match its type (1 or 2), and the observation probability, based on the fraction of the circle representing the position of the minor body plus its position uncertainty contained inside the fov. Tests done as the elements from Lowell evolve shows that the CEU uncertainty parameter is not always useful to estimate the real uncertainty, but when the probability value is close to 1 the image will probably show the track of the minor body.

Here there is a list of the criteria applied to discard minors bodies for a specific calculation (observation) during the main loop to compute cross-matches, listed in the order they appear in the implementation. Some criteria are selectable, which means they can be enabled or disabled through the properties, with the parameter allow_quick_discard_cross_matches.

To complete the description, during the calculation of the precise trajectory of a minor body that may show a cross-match in a given time window, an additional set of conditions are used to select if the minor body needs numerical integration to return the accurate position to check for a cross-match, or if the required accuracy can be provided with just solving the orbit with the elements. For elements referred to more than 37 days respect the calculation time, objects with a set of non-gravitational parameters, or located within 0.5 AU from the Sun, the integration is done always. Then a new orbit is computed so that in later observations, processed backwards by time, an integration over the same body may not be needed.

Criteria	**Selectable**	**Reason**
Orbital elements not applicable	No	The set of elements can be null if the object collided, so it does not exist on that date The elements object may have a orbitDisabled flag set to true. For objects that collided in the past, it is done when reading the elements. The flag was also set when a minor body is beyond 50 AU from the Sun, but this condition was later removed since comet Hale-Bopp is around that point and still being observed with JWST.
Ecliptic latitude of the observation too high	Yes	When the minor body is always further from the Sun than the telescope + 1 AU (so the distance > 1 AU always towards the outer Solar System), and the ecliptic latitude of the body as seen from the telescope (computed with its orbit inclination) is always higher than the minimum ecliptic latitude of the observation, the minor body will never, ever, could be on that field of view
Quick approximate angular distance of the minor body to the observation center too high	Yes	The solution to the Kepler equation is precalculated in a 2d matrix (a grid of values for eccentricity and mean anomaly) to get a fast approximate position of the minor body in case the eccentricity is below 0.75 and the body does not have non-gravitational parameters (A1-A3 values in the Marsden model for non-G forces, used in some comets and asteroids). The output is a quick estimate of the angular distance to the observation with an error of 3 arcminutes at most. The criteria used is to discard the minor body is if the distance is greater than the radius of the observation + 10 degrees (to account easily with any possible gravitational effect that may affect the orbit since the reference time of the elements). In the strange case the object is closer than 0.1 AU to the telescope, this criteria is not applied. The idea is that these gravity perturbations will never reach 10 degrees in an interval of 37 days, even for objects at 0.1 AU from the telescope
Position uncertainty (CEU) for asteroids in the Lowell database	No	[This and later conditions are applied to the accurate position of the minor body] The orbital elements include a rms of the fit (CEU parameter), the arc of time the body was observed, and the intermediate date when the elements were fitted. The criteria used is to consider that the uncertainty will increase at a rate equal to the rms for an additional interval of time equal to the observed arc time. In addition, the distance to Earth is computed for the fitted date and used to scale the uncertainty for the distance of the body to the telescope, to get the real angular uncertainty, so in other words the uncertainty in arcseconds is translated to a unit of distance. The CEU rate parameter of the Lowell elements is ignored since it is not adequate to compute the real uncertainty. For both Lowell and JPL the same criteria is used. If the computed uncertainty of the body for the date is higher than a value selected in the properties, the object is discarded. The default value in the properties is 0.01 degree and 10 times the radius of the observation, and both conditions should pass. If an observation is only within a field of 1" and the error is more than 10", it is very unlikely to find the object in the image, but in this case probably that body was the purpose of the observation, so the absolute condition 0.01 deg will allow the cross-match to be computed unless the uncertainty is even above this value. So this value should not be very low. In the old pipeline the criteria was only absolute, not relative to the size of the field of view.
Angular distance of the minor body too high / velocity too little	Yes	Discarded if the radius of observation < 3 deg, and distance body to telescope > 0.5 AU, and the angular distance to observation center is > 18 deg. Basically, in one day interval it is imposible for a minor body located far from the telescope to move several degrees Discarded if previous angular distance is higher than the radius of the observation + the angular distance the minor body may move in 1 day at most (time window of the cross-match search) + 0.5 deg (as a safe margin)