

Identifying Brown Dwarf candidates in 2MASS and SDSS

Hendrik Heint and Markus Demleitner

February 23, 2022

Abstract

Brown dwarfs are faint objects with a mass below 0.8 solar masses. In their stellar core, hydrogen fusion never starts, and therefore their luminosity is very low. Even those brown dwarfs which are close to the solar system are faint objects, hard to detect and to identify as such. One method to identify brown dwarf candidates is using surveys as 2MASS and SDSS which provide colour photometry. In this use case we will identify brown dwarf candidates by applying the method introduced by Zhang et.al. (2010). We use the VO tools TOPCAT and Aladin, and especially ADQL to access data provided by TAP services for 2MASS and SDSS. We crossmatch our results with SIMBAD to weed out known brown dwarfs. Eventually we run a global discovery (using the VO protocol SSAP) for a spectrum for one of the objects not hitherto in SIMBAD.

Software: [TOPCAT](#) Version 4.7, [Aladin](#) Version 10.0, [SPLAT](#), [TAP](#), [ADQL](#)

1 Starting out

We will use the VO tools TOPCAT, Aladin and Splat-VO. These software packages be found here (or perhaps in the repositories of your distribution):

Aladin: <http://aladin.u-strasbg.fr/>

TOPCAT: <http://www.star.bris.ac.uk/~mbt/topcat/>

Splat-VO: <http://www.g-vo.org/pmwiki/About/SPLAT>

2 ADQL and TAP services

ADQL (Astronomical Data Query Language) is the language used in the VO to query remote TAP services. It is based on SQL and can be learned easily. We will use ADQL to post queries to different TAP services and especially to let the remote services do some of the work, so we receive only the data we are interested in and we do not have to download whole catalogs.

To use TAP (Table Access Protocol) services, one needs a piece of software (a “client”) speaking that protocol. In this use case we use TOPCAT as a TAP client.

Be aware that this is not an introduction to ADQL. Though the steps will be understandable without any ADQL skills, we recommend to have a longer look the GAVO ADQL course at <http://docs.g-vo.org/adql/html/>.

- ▷ 1 *Searching TAP services in the VO registry* – A good starting point for our search for brown dwarfs is 2MASS, because it provides astrometry as well as magnitudes in infrared filter bands. To find an appropriate TAP service for 2MASS (or really, anything in the VO), there is the VO Registry. TOPCAT has interfaces to that Registry, so you don’t have to bother with the details. Just go to [VO](#) → [Table Access Protocol](#). In the TAP Query window at [Keywords](#), enter **2MASS**. In the line below, check [Description](#) (you may need to adjust the window width to see all options). Then click on [Submit Query](#) (that’s querying the Registry behind the scenes).

In the pane below the search field you can now see a list of TAP services related to 2MASS. When searching for services, you may need to spend some time to find the one best suited for your needs. In our case, we already have a good guess that the **GAVO Data Center TAP service** will be the right choice because it provides us with more data we will need in a later step. So in this list, we mark the line by clicking once. Then click [Use Service](#) at the bottom of the window.

Now we have to select the table we want to query. On the left side you see all tables which the GAVO data center TAP service provides. You can either scroll down until you find **twomass.data**, or instead use the search field above the list. Note that you get a first glance of the metadata of this table in the field on the right. Here you find a good overview when you mine for data. We will take a closer look at metadata a few steps ahead.

- ▷ 2 *Submitting a first ADQL Query: cone selection* – Within the VO the term *cone search* references to queries for “things” in an area of the sky defined by a position and a radius around it. You may have performed cone searches already with TOPCAT using a dedicated VO protocol for that. Here is a different approach using TAP and ADQL to give you an idea how these work. Luckily TOPCAT comes with examples for the most common ADQL queries – plus sometimes queries the TAP service maintainers assume to be common or at least useful.

In the lower window at [ADQL Text](#) click on [Examples](#) → [Cone Selection](#). You see an example ADQL query appearing in the field:

```
SELECT
TOP 1000
*
FROM twomass.data
```

WHERE

```
1=CONTAINS(POINT('ICRS', raj2000, dej2000),
           CIRCLE('ICRS', 189.2, 62.21, 0.05))
```

Now at the first glance that may look confusing, so let's go through it step by step. Each ADQL query starts with `SELECT` followed by specifications of "what" to select, what to do with the selected records and how to return the results. `TOP 1000` means no more than 1000 data records will be returned, regardless of how many actually match. Note that this limit does not depend on any order, though you might think that "top" implies that there might be a "bottom" too. The database will just return the first 1000 results coming in.

With `*` we select all columns of matching data records. If you just want certain columns from a table to be returned, you could specify these here. We will see in a later step how to do this. `FROM twomass.data` specifies the table of the TAP service we want to query. The next lines are the query conditions. In our example we use the ADQL built in functions that define a cone search. The `WHERE` clause extracts those data records that match. In our case these criteria shall be sources in a cone around a certain position.

The `1=CONTAINS` may look a bit odd, but it is simply a simulation of boolean results. A `CONTAINS` result of 1 means "true" whereas 0 means "false". Ignore `POINT`'s first argument; it's going to be deprecated soon. The remaining arguments are the right ascension and the declination in degrees. Our query takes the coordinates from the table using the columns `raj2000` and `dej2000`. Analogously `CIRCLE` expresses a spherical circle; pass the radius as an angle in degrees.

Clicking **OK** will start the cone search and within a few seconds we retrieve the first 1000 data records. Return to the TOPCAT main window and play around with this data (like, do some plots) if you like to.

- ▷ **3 Metadata** – As mentioned above you can have a glance at the metadata of a TAP service in TOPCAT's TAP window. For our hunt of brown dwarfs, we basically need colour indexes and therefore search the broadband photometry. In the TAP window we click on **Table** → **Table columns**. Here we have an overview of the table metadata and get the names of the columns which we need later to specify our ADQL query. We are especially interested in J, H and K magnitudes, which we see are named `hmag`, `jmag` and `kmag`. Knowing the column names, we now can modify the ADQL query accordingly.
- ▷ **4 Searching 2MASS with ADQL** – Now we want to search 2MASS for those objects, which are good candidates for brown dwarfs. Because brown dwarfs are faint objects, we estimate they have a J magnitude of 15.3 or higher (yes, this is a bit of a didactic cheat – accept it for now).

Also, brown dwarfs are very red sources. Hence, we want to search for

sources with a colour index of

$$jmag - kmag > 0.8.$$

Running the query over the whole 2MASS catalog would take a long time so for this exercise we don't search over the whole sky but limit the search to a 2 degrees cone instead. Finally, we change the search cone coordinates.

All this we can do with ADQL by modifying the query as following:

```
SELECT *
FROM twomass.data
WHERE 1=CONTAINS(
POINT('ICRS', raj2000, dej2000),
CIRCLE ('ICRS', 127.0000, 1.2000, 2.0))
AND jmag > 15.3
AND jmag-kmag > 0.8
```

Before we submit the query, we have to set a maximum so we receive all the matching data records. To do so look at **Service Capabilities** → **Max Rows** and select **max** in the drop down menu. We sent the query by clicking on **OK** and the result should be returned in a few seconds.

- ▷ 5 *Crossmatching with SDSS* – With the J, H, and K broadband magnitudes from 2MASS we found first candidates for brown dwarfs. To further weed out non-brown dwarfs we now apply criteria derived from Zhang et al. (2010). Therefore, we need magnitudes in SDSS i, r, and z bands. In the Topcat TAP window we now enter `sdssdr7`, again check [Description](#) and submit the query. While it doesn't really matter, for this exercise choose GAVO's service again.

We could now do a similar search as for 2MASS and then compare the two tables locally. But we rather use a more elegant method and let the remote TAP service do some work for us. As long as both tables are on the very same service, we can merge the two queries into a single one and obtain the result at once; when that is not the case, on most services you can upload local tables and use them remotely for the duration of your query (we will see later how that works).

You will have noted that we start with `WITH tmandsd AS` This is called Common Table Expressions (CTE) which makes it bit easier to understand complex queries, and also gives you a bit of control over the order at which the database will perform the parts of the queries. We also use several `SELECT` statements, which may look a bit confusing, but it helps structuring the queries: keep in mind that the result of a `SELECT`-statement is always a table and we can perform all table related functions on this table. You can see this in the part of the cone selection on the table `twomass.data`. Using the alias `AS tm` we can refer to this result when we join it with the table `sdss.sources`.

To link the rows from the two tables (2MASS and SDSS), we use the JOIN... ON construct in ADQL to merge the the data. This time we use the CIRCLE function to find matches for the data we received from 2MASS and therefore keep the cone very small at 1”.

Note that we moved the conditions of brightness and the colour cuts to another SELECT-statement. Thus, we tell the databse explicitly to first perform the cone selection in 2MASS, then x-match with SDSS and in the last step check the colours. The reason for this is performance: if we put all in a single SELECT-statement, the database will guess which might be the fastest way and may start with the colour cuts – and of course this computation over more than 500 million sources will take much longer time than performing the cone selection first, and then compute the colours on much fewer sources.

The criteria we use to perform the SDSS colour cuts are taken from Zhang et al. This is the full query:

```

WITH tmandsd AS (
  SELECT *
  FROM (
    SELECT * FROM
    twomass.data
    WHERE
      1=CONTAINS(
        POINT('ICRS', raj2000, dej2000),
        CIRCLE ('ICRS', 127.0000, 1.2000, 2.0)
      )
    ) AS tm

  JOIN sdssdr7.sources as sdss
  ON 1=CONTAINS(
    POINT('ICRS', sdss.ra, sdss.dec),
    CIRCLE('ICRS', tm.raj2000, tm.dej2000,
    1./3600.)
  )
)

SELECT *
FROM tmandsd
WHERE  jmag > 15.3
      AND jmag-kmag > 0.8
      AND i - z BETWEEN 1.5 AND 2
      AND r - i BETWEEN 1.5 AND 4.5
    
```

After submitting the query we receive a table of hopefully good candidates for brown dwarfs, so let’s take a look at the data. You can immediately see that

for some 2MASS identifiers there are two records in our result set. This is due to the TAP search which returned all matches around a source from the 2MASS data, and not the single best match.

To see what is behind these duplications, look at how the objects look like on the sky. Thankfully the VO provides us with the perfect tool for that: Aladin.

- ▷ **6** *Resolving suspicious results with Aladin and TOPCAT* – In this step we want to see the images in 2MASS and SDSS of our brown dwarf candidates, with a particular view to the duplicated SDSS counterparts for 2MASS objects. For this we use the SAMP protocol to send data from TOPCAT to Aladin. We start Aladin and select use 2MASS as a background. Zoom in to a field of view of an arcminute or so as a background. Zoom in to a field of view of an arcminute or so.

Then we go back to the TOPCAT main window and check the table with our candidates from SDSS. We then click in [Current Table Properties](#) → [Activation Action](#) → [Transmit Coordinates](#). That tells TOPCAT to tell all other SAMP clients the sky position of a row (or a point in a plot) when you click on it. Aladin, for instance will then automatically jump to that position, showing the vicinity of that position.

Still in Topcat we open the table view and click on any table row. In Aladin we can see the position change. If we compare the images from our duplicated data records, we see that the sources point to the same object in the catalog. We can switch between 2MASS and SDSS images in Aladin and see the same effect. This is simply because we received all matching data records from our crossmatch with SDSS and in the SDSS catalog the different observations to a single object are not merged (probably – at least one object might actually be a bona fide double star). Anyway, let us clean up our little table so the duplicate matches go away.

We solve this in TOPCAT. In the main window we mark the table and click on [Joins](#) → [Internal match](#). In the new window at [Table](#) we chose our current table and below we check [Eliminate All But First of Each Group](#). Of course you should think twice before blindly throwing away records in a real science project. For the current discovery purpose, it's just fine since we don't lose any positions. Now we reduced our brown dwarfs candidates table to 11 rows and it's about time to check the accuracy of our method – and if perhaps we've discovered new brown dwarfs.

- ▷ **7** *TAP upload to SIMBAD* – Now we want to know how many of our candidates are already confirmed brown dwarfs. A good location to check for this of course is SIMBAD. We again go to the TOPCAT registry search window and now search for [Simbad](#). From the few options listed we select the Simbad service.

We select the table `public.basic`. This time we don't want to get all data from SIMBAD but only the column that contains the object type. Checking the service metadata we find out that we will need the column `otype.txt`. The

special feature of the query we want to perform is the TAP upload though. TAP lets you upload local data and treat it the same way as any other table on the remote service. We will use this to perform a crossmatch between our brown dwarf candidates and the SIMBAD database.

In the topcat TAP window click on [Examples](#) → [Upload](#) → [Upload Join](#). This will generate something like

```
SELECT
  TOP 1000
  tc.*, db.otype_txt
FROM basic AS db
RIGHT OUTER JOIN TAP_UPLOAD.t1 AS tc
ON 1=CONTAINS(
  POINT('ICRS', db.ra, db.dec),
  CIRCLE('ICRS', tc.raj2000, tc.dej2000, 3./3600.))
```

You may notice that our JOIN command now is different to the one we used above. A normal (“inner”) JOIN will only return records with matches in both tables. In this case you would only receive rows *with* matches in Simbad.

But it this step we actually are interested in keeping those of our local records that do not match records in SIMBAD: They could be good candidates for further research, because apparently they have not been identified as brown dwarfs in the published literature yet. RIGH OUTER JOIN roughly translates to “Join table1 with table2 where the following conditions are met. Where the conditions are not met, fill any columns from table1 with NULLs.” Analogously there is the construct LEFT OUTER JOIN implemented in ADQL.

Now let’s look at our output table: we have an additional column `otype_txt` in which we find the object type. You see that 5 of our objects are listed in SIMBAD as brown dwarfs. The other 6 objects may be candidates worth further research.

▷ **8** *Obtaining spectra with Splat-VO and SSAP* –

We now want to use Splat-VO and the Simple Spectrum Access Protocol (SSAP) to search for spectra that might give us further hints of what the object might be. We open Splat and in there open the SSAP window: [File](#) → [SSAP_](#). In the left part of the SSAP window we see the server selection. Here- one can preselect services according to the spectra you are interested in. For us, it’s fine to not further constrain the waveband and to check all services by clicking on [select all](#).

As search parameters we will give the position of one of our candidates from the topcat table: **RA: 126.918696**, **Dec -0.146868** for the position and **1 arcminute** as **Radius**. We submit the query by clicking on the green button **SEND QUERY**. Splat-VO now searches on all SSAP services that we selected for spectra within the range of 1 arcminute of our given position.

After a few moments we will receive results from three services and we can select and download spectra. Thanks to the metadata provided by the services we can figure out which spectra are of interest for us. Unfortunately, this is not always the case: please complain to the operators (you can find contact information in the registry) if you find missing metadata – every bug fixed improves the VO for those using it after you.

3 References

- Demleitner, M. <http://docs.g-vo.org/adql/html/>
- Fernique, P. <http://aladin.u-strasbg.fr/java/AladinManual6.pdf>
- Ortiz, I., Lusted, J., Dowler, P. et al. <http://www.ivoa.net/documents/REC/ADQL/ADQL-20081030.pdf>
- Taylor, M. <http://www.star.bris.ac.uk/~mbt/topcat/#docs>
- Zhang, Z. H., Pinfield, D. J., Day-Jones, A. C., et al. 2010, MNRAS, 404, 1817, 2010MNRAS.404.1817Z