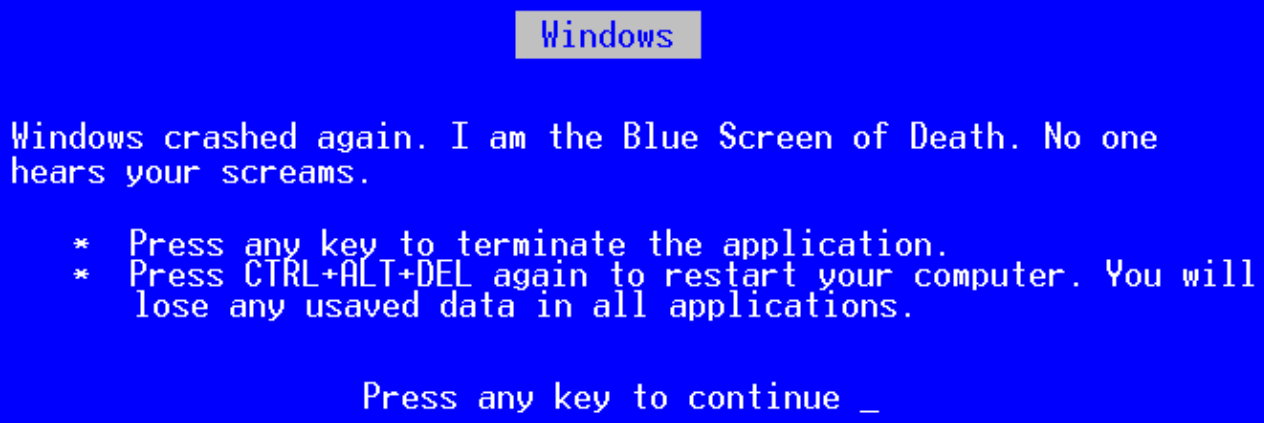
Heidelberg Institute for Theoretical Studies

HITS

Is VO ready for machine learning?



Challenges

Heidelberg Institute for Theoretical Studies

HITS

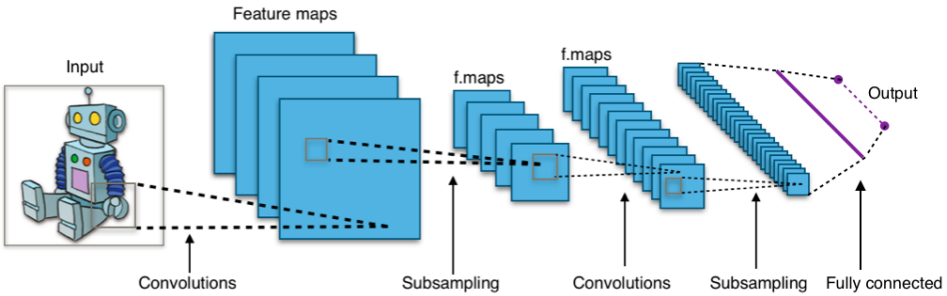


Image from Wikipedia

Big Data by Nick Youngson CC BY-SA 3.0 Alpha Stock Images

SKA SQUARE KILOMETRE ARRAY

LSST Large Synoptic Survey Telescope

Data science

Heidelberg Institute for Theoretical Studies

HITS

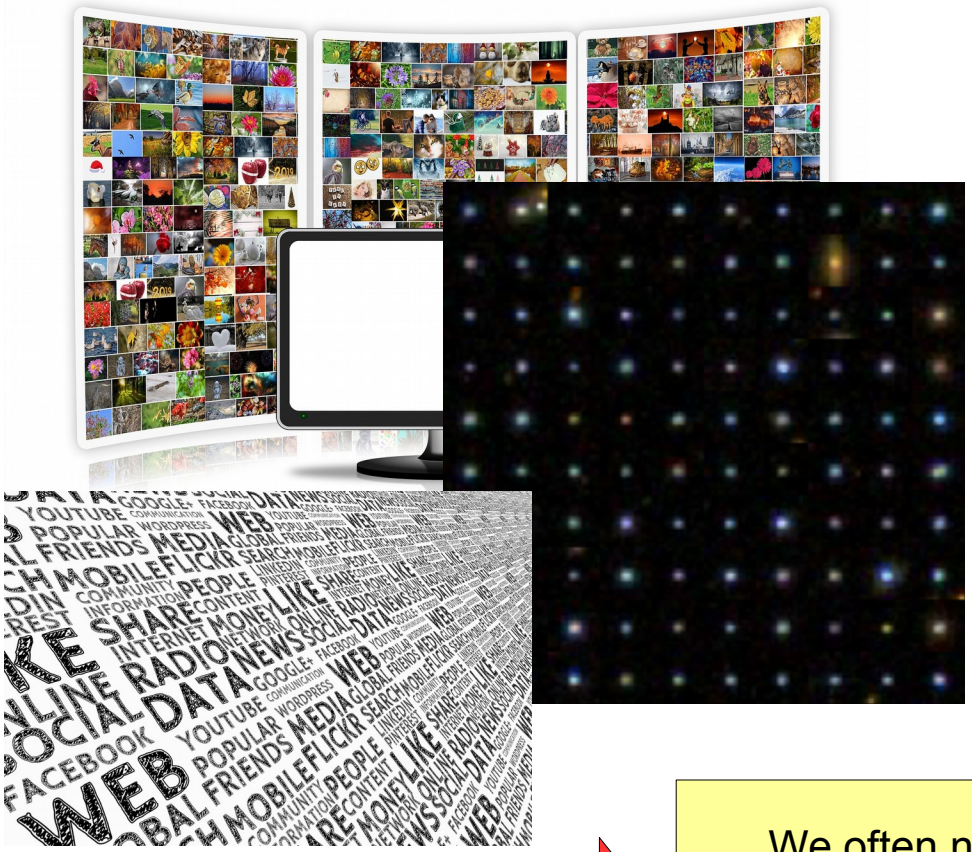
"Big data is like teenage sex: everyone talks about it, nobody really knows how to do it, everyone thinks everyone else is doing it, so everyone claims they are doing it..." [Dan Ariely]

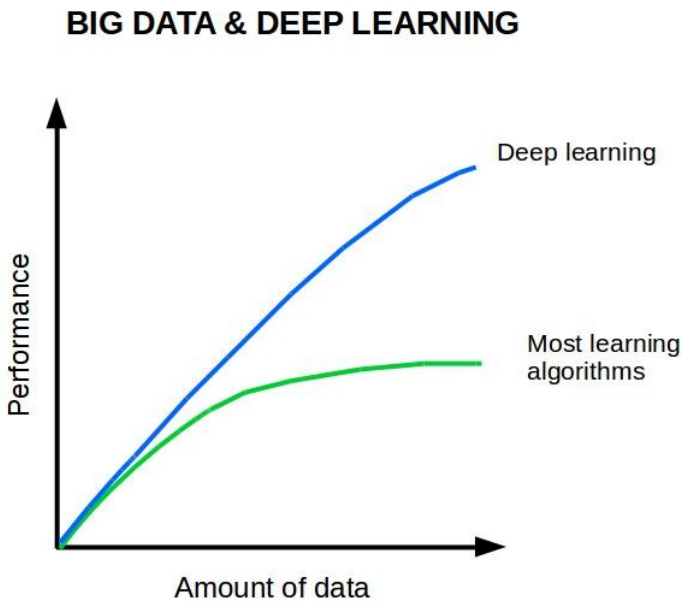
...so be careful with it!

We are hungry for data!

Heidelberg Institute for Theoretical Studies

HITS





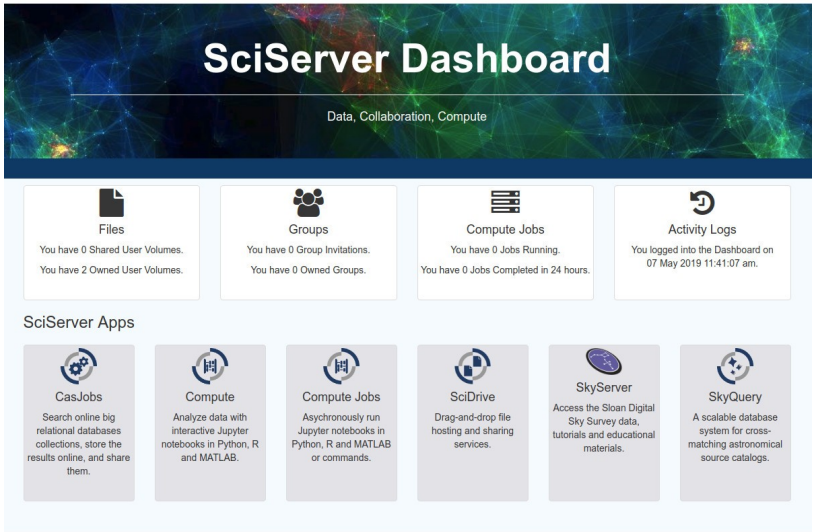
We often need to download huge amounts of data

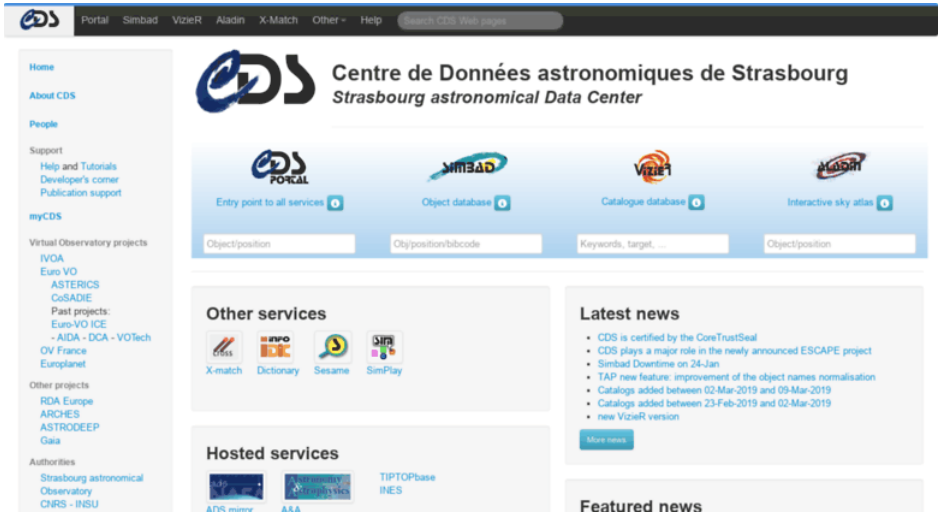
Catalogs

Heidelberg Institute for Theoretical Studies

HITS

Working with catalogs is a simple task:





Problems start with images and spectra!

Some “simple” tasks...

Heidelberg Institute for Theoretical Studies

HITS

1. Given the coordinates, download 28x28 pixel^2 images for all the quasars in SDSS.

2. Download some hundreds of thousands of images from FIRST/UKIDSS.

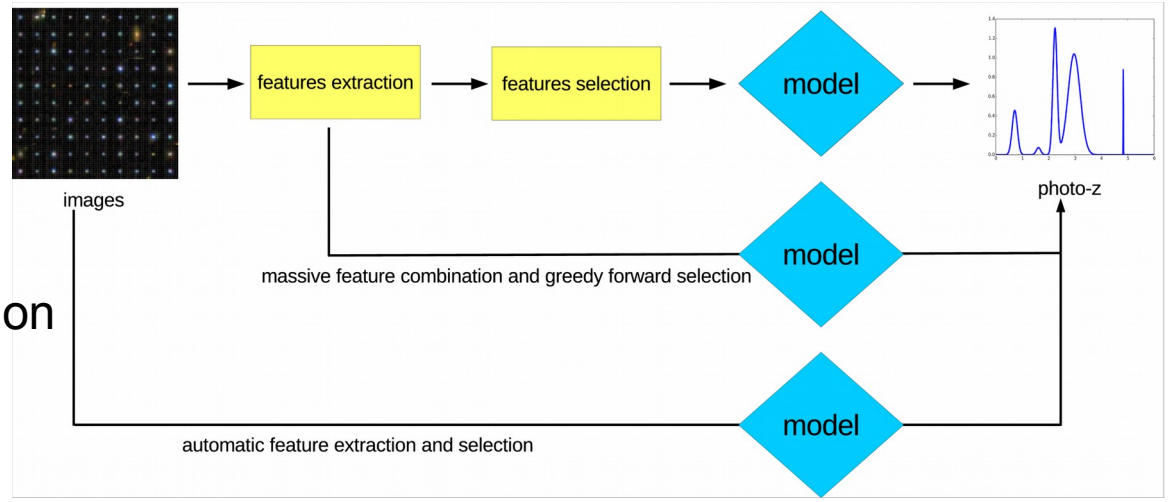
3. Download all the HARPS spectra from ESO archive.

Obtaining data products can be a not easy task

Task 1

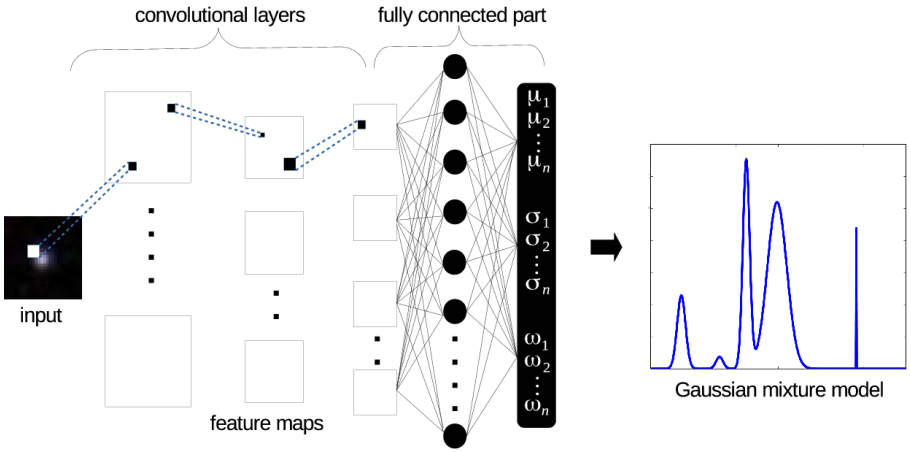
Heidelberg Institute for Theoretical Studies

HITS



Scientific problem:
Photometric redshift estimation

Deep Convolutional Mixture Density Network (DCMDN)



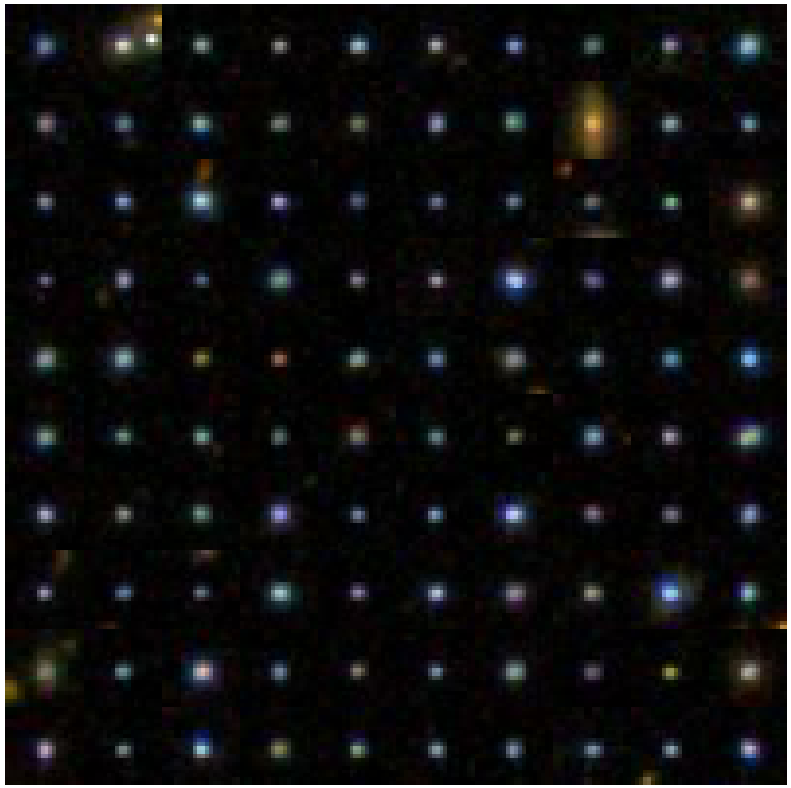
Task 1

Heidelberg Institute for Theoretical Studies

HITS

- 185,000 quasar from SDSS DR7/DR9
- 200,000 galaxies from SDSS DR9
- 200,000 stars from SDSS DR9 (redshift set to 0)

Need for 28x28 pixel² images in *ugriz* filters

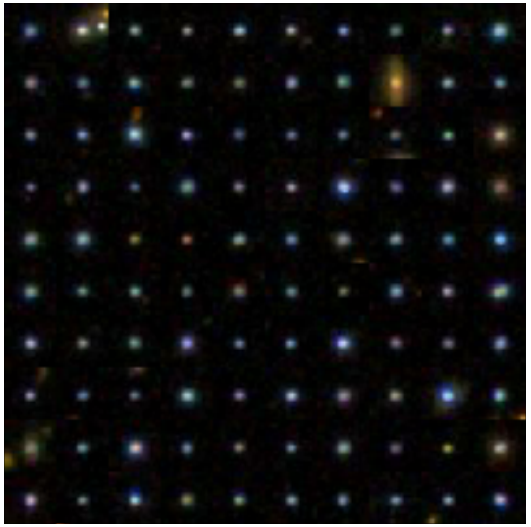


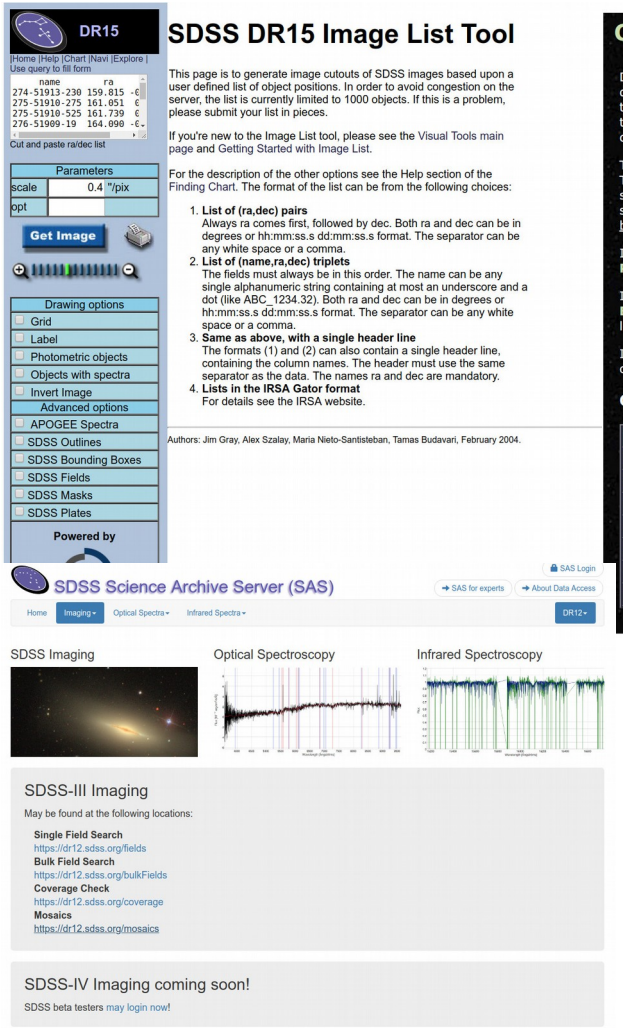
Task 1

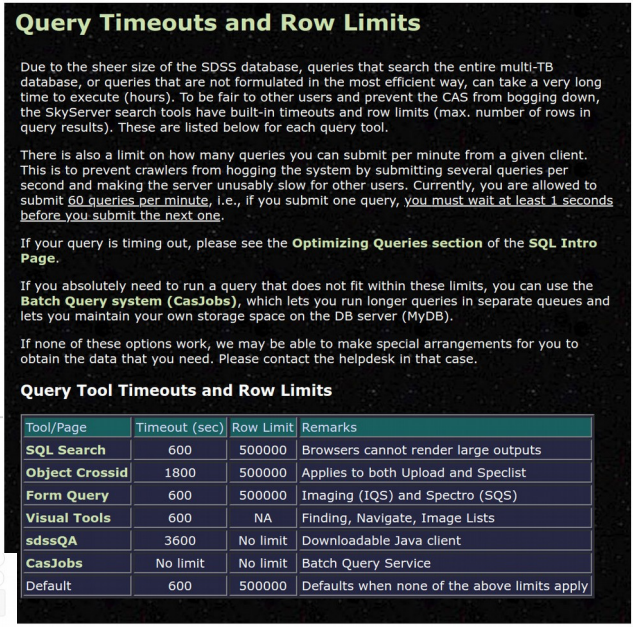
Heidelberg Institute for Theoretical Studies

HITS

Given the coordinates, download 28x28 pixel^2 images for all the quasars in SDSS:







HiPS: Hierarchical Progressive Survey

Heidelberg Institute for Theoretical Studies

HITS

Based on HEALPix sky tesselation → mapping of survey data at various spatial resolutions into a collection of HEALPix tiles

Make your HiPS in 10 steps

By P.Fernique (last update April 2018 - additions in green)

- 10 steps
- Tips
- Booster
- Manual

... or how to build a Hierarchical Progressive Survey from your image (or cube) collection.

Your HiPS step by step

1. **Download the latest version of Aladin/Hipsgen code**. Do not hesitate to use the last beta version:
   → AladinBeta.jar
   Note: You can also use Hipsgen.jar package, but it is more useful to use directly Aladin for generating & checking the HiPS with the same tool (and concretely, it is the same code).
2. **Set together the collection of the original images or cubes in a local directory** (sub-directory hierarchy is allowed, symbolic links can be useful, gzipped ou rice compressed images are allowed)
   → we assume "`Data`" directory
   Note: There is no limit (image size and/or number of images) apart your disk.
3. ***Optional:* Load one image in Aladin** to be sure that its astrometrical calibration is correctly interpreted and to determine the HiPS generation parameters:
   → `java -Xmx2g -jar AladinBeta.jar Data/OneImage.fits`
   - If your images are packaged in Multi-Extension FITS, memorize the index of the good HDU or possibly all HDUs for multi-CCD packaging (cf. *hdu* parameter)
   - Adjust the contrast and memorize the pixel min & pixel max display value (*Ctrl+M*: left and right values over the pixel histogram) (cf. *hips_pixel_cut* parameter)
   - Check if the borders of your images are ok, or if you have to remove a margin, and/or if the borders of image have not been observed -typically filled up with 0 (cf. *blank* parameter), or if your observations are circular (cf. *shape* parameter).
   - Look in the FITS header (*Alt+H*) which additionnal keywords could be kept as meta data information for progenitor facility (see below), for instance: *DATE-OBS, INSTRUM, EXP_TIME*, ... (cf. *fitskey* parameter)
4. ***Optional:* Create a "pilot" HiPS on few images** for a quick test:
   - Launch the Pilot HiPS creation on the first 100 images by this script command:
     →
     `java -Xmx16g -jar AladinBeta.jar -hipsgen in=Data out=PilotHiPS creator_did=HiPSID pilot=100`
   - Check the resulting HiPS thanks to Aladin:
     → `java -jar AladinBeta.jar PilotHiPS`
   - If required, reiterate the HiPS computation by inserting/modifying these Hipsgen parameters: *blank, border, fov, skyval, hips_pixel_cut, fading, mixing, partitioning...* - see the manual below.
     Do not forget to insert "`-e`" option for removing the previous HiPS (or remove it manually).
5. **Launch the HiPS creation on all original data.**
   Take care that this process can be long and very computer consuming (RAM and CPU). You can adjust the RAM by the java *-XmxNNg* parameter, and the number of threads by the *maxthread* Hipsgen parameter. The final HiPS size can be estimated to 1.3x of the original size of data.
   → `java -Xmx16g -jar AladinBeta.jar -hipsgen maxthread=20 in=Data ...`
6. **Check the result** thanks to Aladin Desktop and/or via Aladin Lite
   → Aladin Desktop: `java -jar AladinBeta.jar DataHiPS`
   → Aladin Lite: in your browser, load the directory `DataHiPS`
7. **Complete the description of your HiPS** by editing the file `DataHiPS/properties` (read the inside comments). Do not forget to provide a `creator_did` identifier for your HiPS. The syntax follows the IVOA IVOID recommendation (*ivo://XXX/...*). The first word MUST be the acronym of your institute (CDS, IAS, ESAC, STScI, ...) and your HiPS - or atleast your institute - should be declared in the IVOA VO registry (see point 10).
8. **Distribute your HiPS** by serving or moving your *DataHiPS* directory thanks to a basic httpd server (apache...).
   No data base software or CGI are required, just a HTTP access to the HiPS directory.
9. ***Optional:* Add your HiPS in Aladin and/or other HiPS aware clients**.
   For that, you need to declare your "HiPS server" and become by this way a member of the "HiPS network".
   - Become a official HiPS server by just generating and publishing your *hips list*. The HiPS list is basically the concatenation of all the properties files of your HiPS. You can do that manually, or via a CGI script like this one (cf HiPS IVOA doc - section 5, page 22)

hipspy

Report abuse

Repositories 2 | People 3 | Projects 0

Find a repository… | Type: All | Language: All

hips
Python library to handle HiPS
python | science | astronomy | image-viewer
Python ★ 7 ⑂ 6 1 issue needs help Updated on Feb 6

hips-extra
hips extras (datasets, etc)
Jupyter Notebook ★ 3 ⑂ 1 Updated on Oct 27, 2017

Top languages
Python Jupyter Notebook

People 3
adl1995 Adeel Ahmad
cdeil Christoph Deil
tboch Thomas Boch

© 2019 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

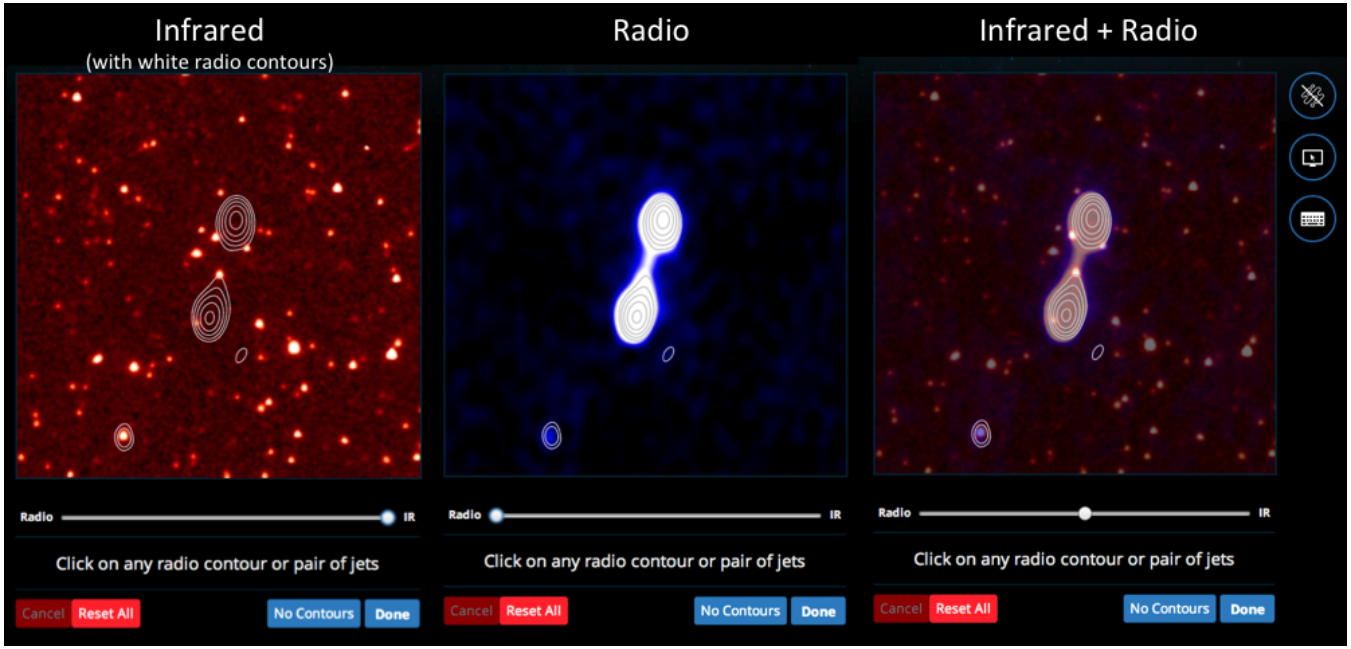
Task 2

Heidelberg Institute for Theoretical Studies

HITS

Classifying radio galaxies with the help of tens of thousands of citizen scientists

GALAXY ZOO RADIO



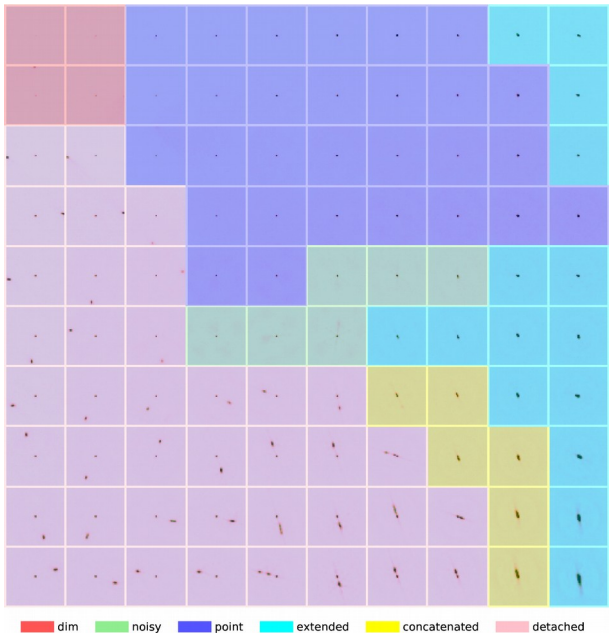
Task 2

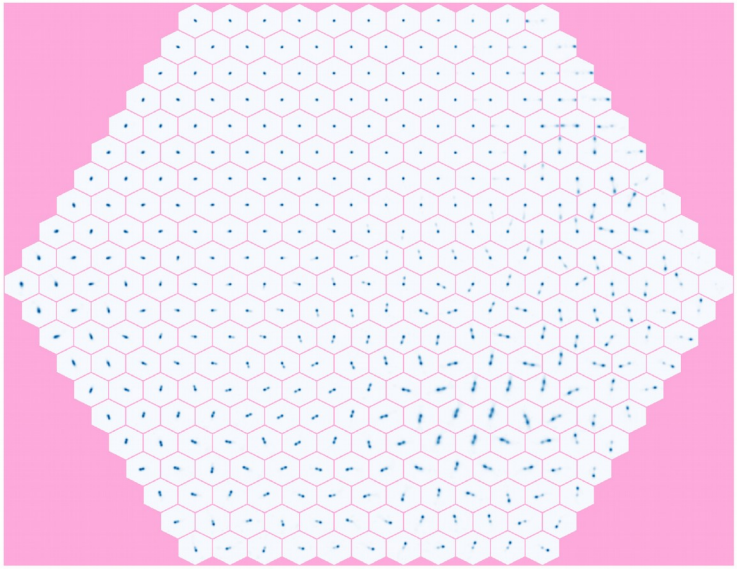
Heidelberg Institute for Theoretical Studies

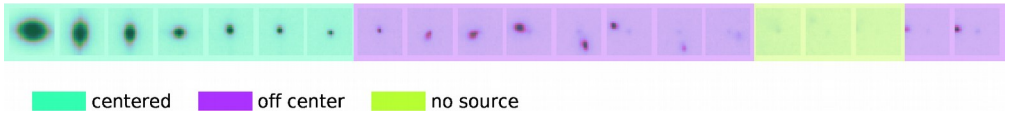
HITS

Morphological classification from image data using PINK (Parallelized rotation and flipping INvariant Kohonen maps)

FIRST







UKIDSS

Task 2

Heidelberg Institute for Theoretical Studies

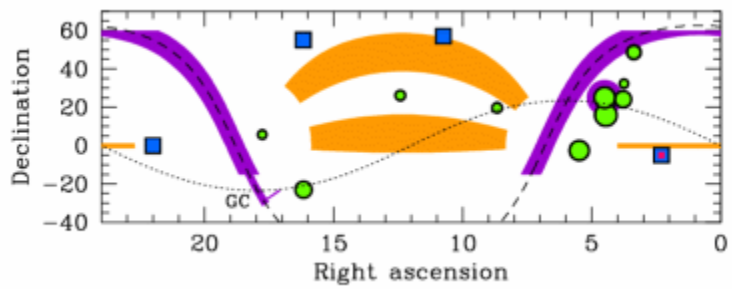
HITS

Download some hundreds of thousands of images from FIRST/UKIDSS: No HiPS!

UKIDSS:

UKIDSS is a set of five surveys. The areas and 5-sigma depths are as follows:

1.	Large Area Survey (LAS)	4000 sq. degs	K=18.4	extraGalactic
2.	Galactic Plane Survey (GPS)	1800 sq. degs	K=19.0	Galactic
3.	Galactic Clusters Survey (GCS)	1400 sq. degs	K=18.7	Galactic
4.	Deep Extragalactic Survey (DXS)	35 sq. degs	K=21.0	extraGalactic
5.	Ultra Deep Survey (UDS)	0.77 sq. degs	K=23.0	extraGalactic



Task 2

Heidelberg Institute for Theoretical Studies

HITS

Main problems:

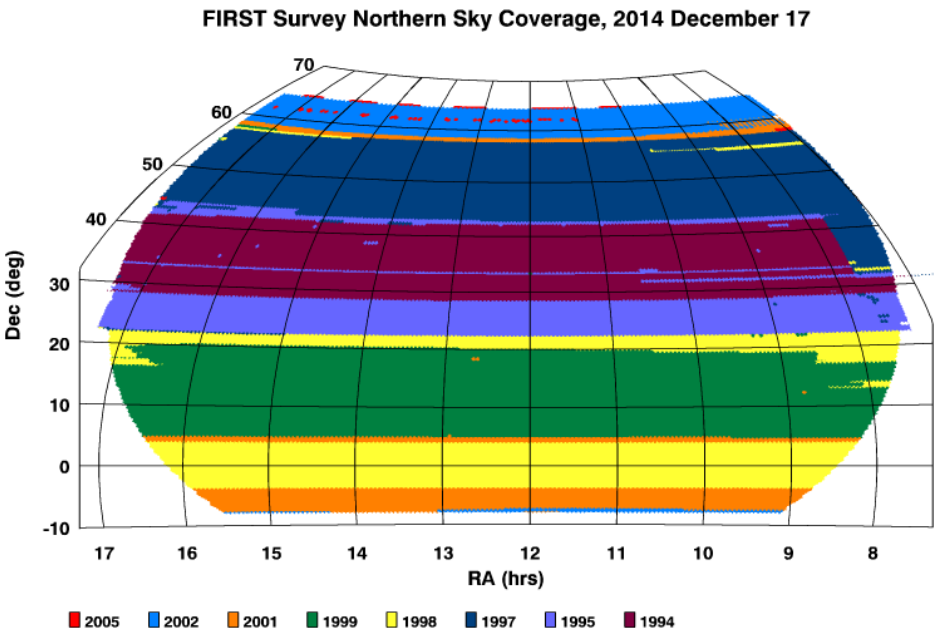
- Online service
  - Only extracts data from tiles which contain the coordinate (no way to combine tiles)
- Sensitivity
  - Lack of stacked images (Y, J, H, K)
- astroquery.ukidss available in astropy, but VERY VERY slow download (requiring 6-8 weeks) and still affected by previous problems
  - When using more clients → DOS attack
- Solution?
  - Download entire survey and implement our own methods for accessing the data
  - Used KDTree to find tiles and coadded images

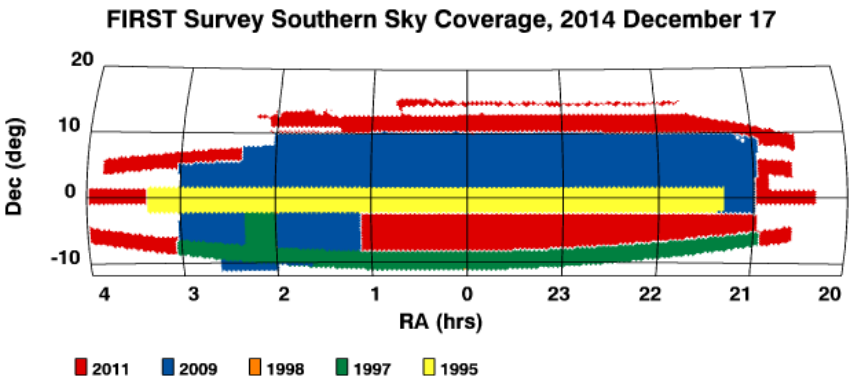
Task 2

Heidelberg Institute for Theoretical Studies

HITS

FIRST:





Main problem: No publicly accessible method to download huge amount of data

Asking friends and colleagues for data and support can be helpful, but it is not what VO is about

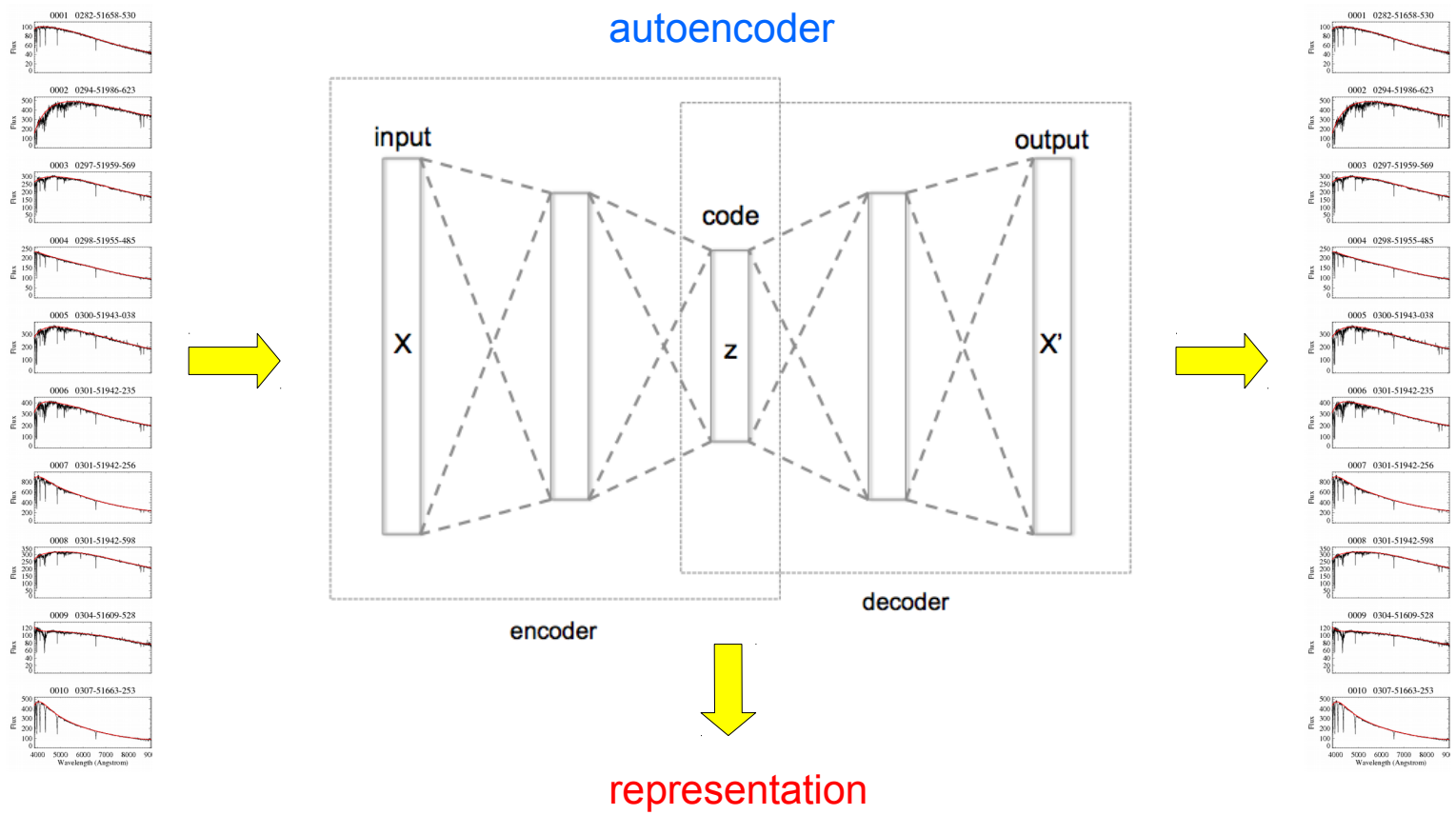
Task 3

Heidelberg Institute for Theoretical Studies

HITS

ESCAPE project – Development of toy example for:

Dimensionality reduction and classification on spectra



Task 3

Heidelberg Institute for Theoretical Studies

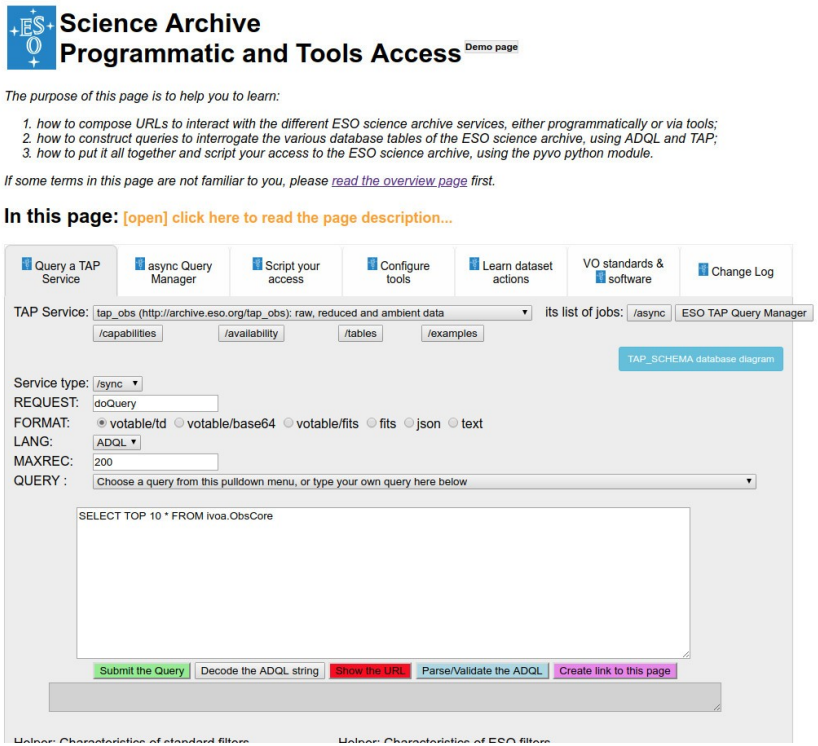
HITS

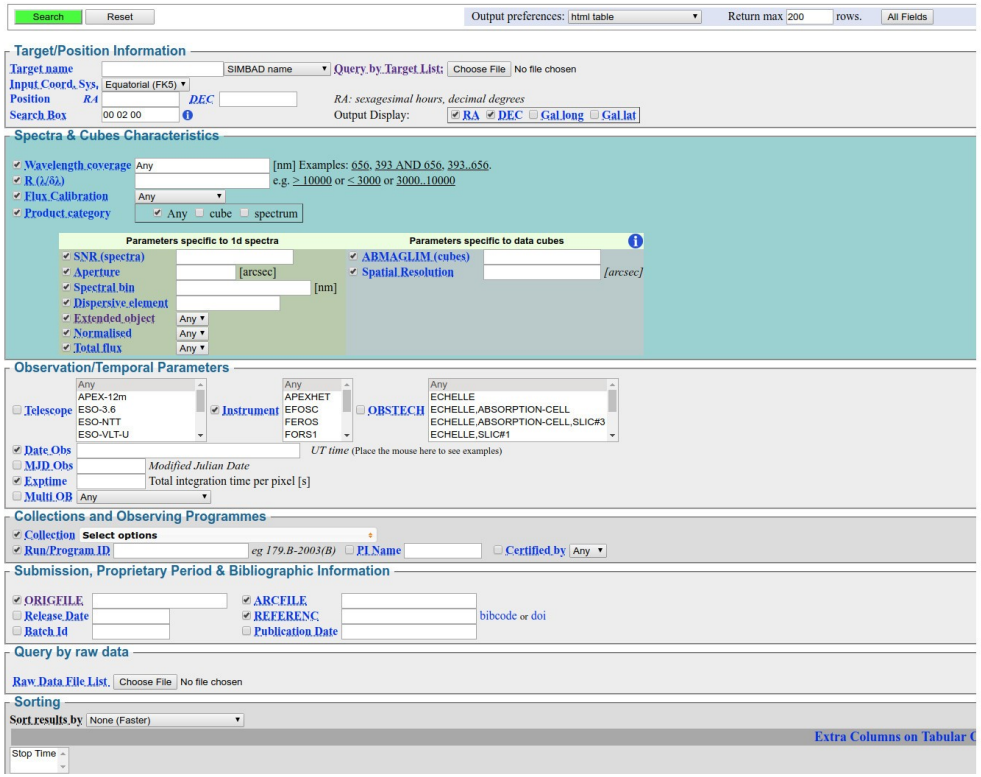
Download all the HARPS spectra from ESO archive:

Two options

TAP service

Request system



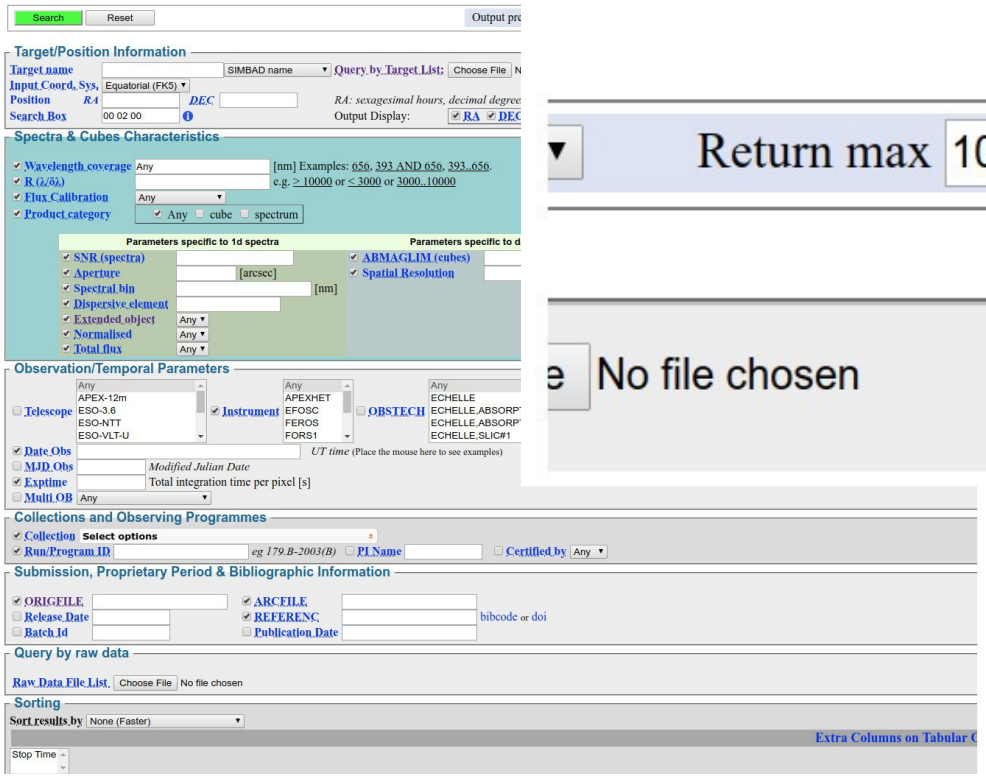


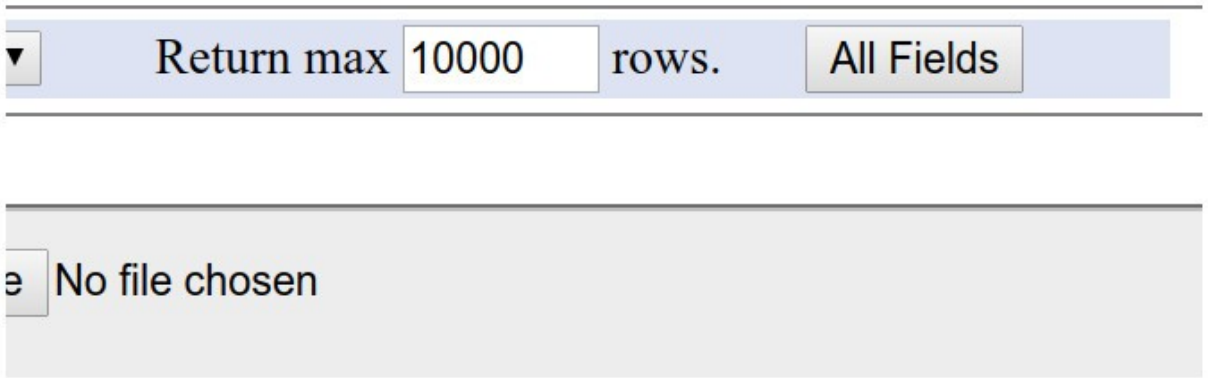
Problems found:

Heidelberg Institute for Theoretical Studies

HITS

Request system





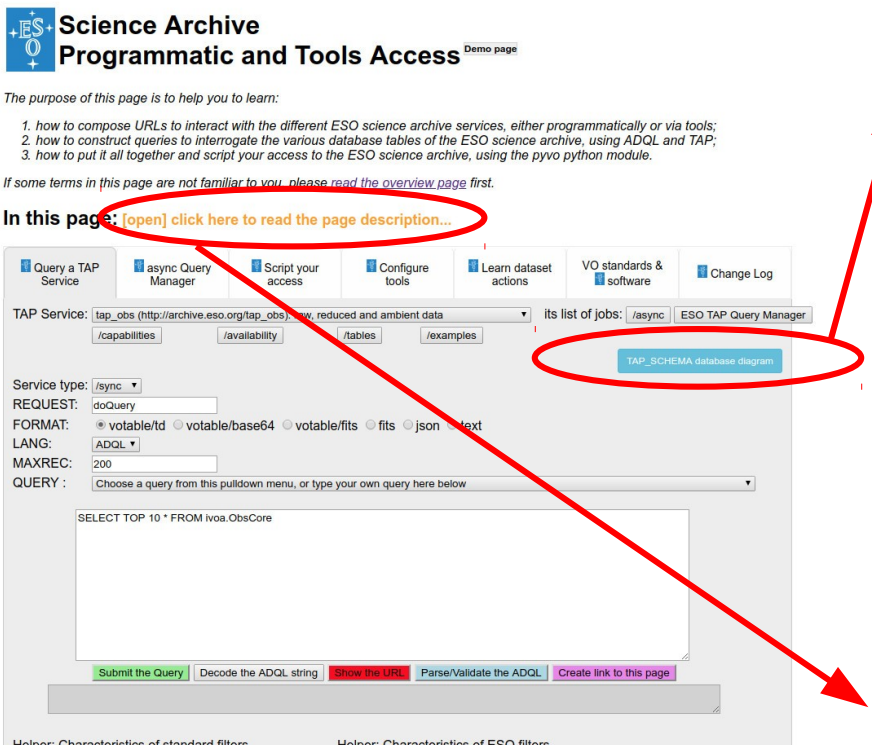
Limit in the amount of data:
not enough for deep learning!

Problems found:

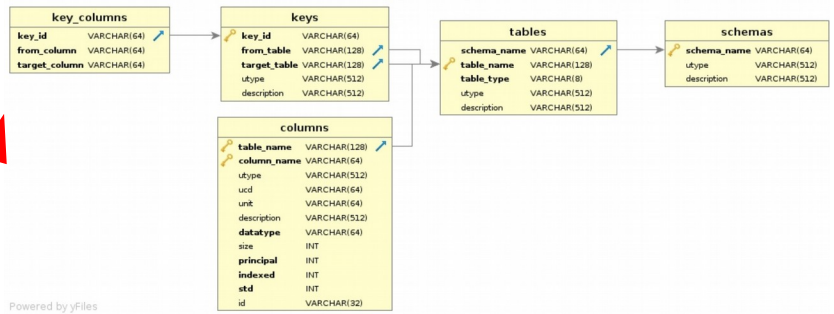
Heidelberg Institute for Theoretical Studies

HITS

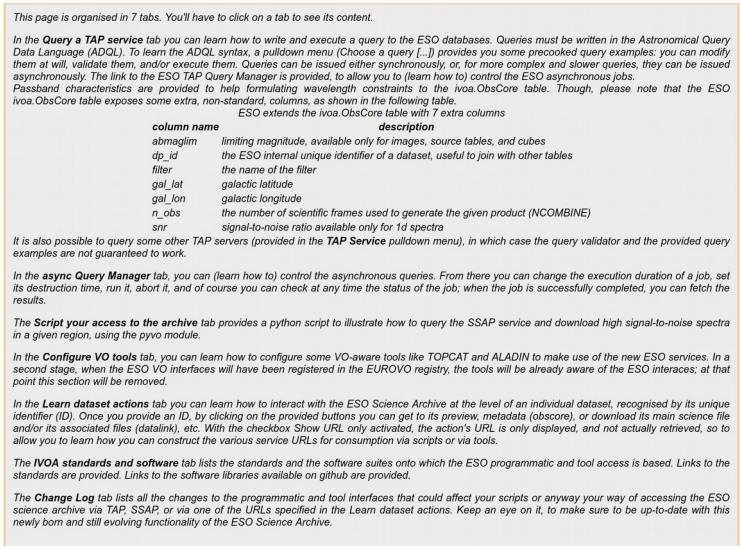
TAP service



- Absence of a clear schema browser



- Poor and unclear documentation



This page is organised in 7 tabs. You'll have to click on a tab to see its content.

*In the **Query a TAP service** tab you can learn how to write and execute a query to the ESO databases. Queries must be written in the Astronomical Query Data Language (ADQL). To learn the ADQL syntax, a pulldown menu (Choose a query [...]) provides you some precooked query examples: you can modify them at will, validate them, and/or execute them. Queries can be issued either synchronously, or, for more complex and slower queries, they can be issued asynchronously. The link to the ESO TAP Query Manager is provided, to allow you to (learn how to) control the ESO asynchronous jobs.*
Passband characteristics are provided to help formulating wavelength constraints to the ivoa.ObsCore table. Though, please note that the ESO ivoa.ObsCore table exposes some extra, non-standard, columns, as shown in the following table.

ESO extends the ivoa.ObsCore table with 7 extra columns

column name	description
abmaglim	limiting magnitude, available only for images, source tables, and cubes
dp_id	the ESO internal unique identifier of a dataset, useful to join with other tables
filter	the name of the filter
gal_lat	galactic latitude
gal_lon	galactic longitude
n_obs	the number of scientific frames used to generate the given product (NCOMBINE)
snr	signal-to-noise ratio available only for 1d spectra

*It is also possible to query some other TAP servers (provided in the **TAP Service** pulldown menu), in which case the query validator and the provided query examples are not guaranteed to work.*

*In the **async Query Manager** tab, you can (learn how to) control the asynchronous queries. From there you can change the execution duration of a job, set its destruction time, run it, abort it, and of course you can check at any time the status of the job; when the job is successfully completed, you can fetch the results.*

*The **Script your access to the archive** tab provides a python script to illustrate how to query the SSAP service and download high signal-to-noise spectra in a given region, using the pyvo module.*

*In the **Configure VO tools** tab, you can learn how to configure some VO-aware tools like TOPCAT and ALADIN to make use of the new ESO services. In a second stage, when the ESO VO interfaces will have been registered in the EUROVO registry, the tools will be already aware of the ESO interaces; at that point this section will be removed.*

*In the **Learn dataset actions** tab you can learn how to interact with the ESO Science Archive at the level of an individual dataset, recognised by its unique identifier (ID). Once you provide an ID, by clicking on the provided buttons you can get to its preview, metadata (obscore), or download its main science file and/or its associated files (datalink), etc. With the checkbox Show URL only activated, the action's URL is only displayed, and not actually retrieved, so to allow you to learn how you can construct the various service URLs for consumption via scripts or via tools.*

*The **IVOA standards and software** tab lists the standards and the software suites onto which the ESO programmatic and tool access is based. Links to the standards are provided. Links to the software libraries available on github are provided.*

*The **Change Log** tab lists all the changes to the programmatic and tool interfaces that could affect your scripts or anyway your way of accessing the ESO science archive via TAP, SSAP, or via one of the URLs specified in the Learn dataset actions. Keep an eye on it, to make sure to be up-to-date with this newly born and still evolving functionality of the ESO Science Archive.*

Solutions?

Heidelberg Institute for Theoretical Studies

HITS

- Python script available on request (thanks Alberto Micol!), but frequent crashes experienced → still investigating the problem

- Download much slower with respect to request system

Asking friends and colleagues for data and support can be helpful, but it is not what VO is about

Bringing code to the data

Heidelberg Institute for Theoretical Studies

HITS

Uploading my code and work on server side could solve many issues…

```
In [ ]: class MDN(nn.Module):
            def __init__(self, n_hidden, n_gaussians):
                super(MDN, self).__init__()
                self.z_h = nn.Sequential(
                    nn.Linear(1, n_hidden),
                    nn.Tanh()
                )
                self.z_pi = nn.Linear(n_hidden, n_gaussians)
                self.z_sigma = nn.Linear(n_hidden, n_gaussians)
                self.z_mu = nn.Linear(n_hidden, n_gaussians)

            def forward(self, x):
                z_h = self.z_h(x)
                pi = nn.functional.softmax(self.z_pi(z_h)#, -1)
                sigma = torch.exp(self.z_sigma(z_h))
                mu = self.z_mu(z_h)
                return pi, sigma, mu

In [ ]: oneDivSqrtTwoPI = 1.0 / np.sqrt(2.0*np.pi) # normalization factor for Gaussians
        def gaussian_distribution(y, mu, sigma):
            # make el_mu - y broadcast to size of mu, divide by sigma
            result = (y.expand_as(mu) - mu) * torch.reciprocal(sigma)
            result = -0.5 * (result * result)
            return (torch.exp(result) * torch.reciprocal(sigma)) * oneDivSqrtTwoPI

        def mdn_loss_fn(pi, sigma, mu, y):
            result = gaussian_distribution(y, mu, sigma) * pi
            result = torch.sum(result, dim=1)
            result = -torch.log(result)
            return torch.mean(result)

In [ ]: network = MDN(n_hidden=20, n_gaussians=3)

In [ ]: optimizer = torch.optim.Adam(network.parameters())

In [ ]: def train_mdn():
            for epoch in range(3000):
                pi_variable, sigma_variable, mu_variable = network(x_variable)
                loss = mdn_loss_fn(pi_variable, sigma_variable, mu_variable, y_variable)
                optimizer.zero_grad()
                loss.backward()
                optimizer.step()

                if epoch % 500 == 0:
                    print(epoch, loss.data[0])

        train_mdn()
```

Who is going to provide and pay for resources?

Conclusions

Heidelberg Institute for Theoretical Studies

HITS

- Download of data products, in particular images and spectra, can easily become a nightmare
- Apparently simple tasks are not simple at all
- Bringing code to the data could be a solution
- A big effort is required from all of us to improve the situation

We are constantly speaking of LSST, SKA, the data explosion in astronomy, but...

Seriously, what are we going to do with them?

Final remark

Heidelberg Institute for Theoretical Studies

HITS

No data service or provider has been harmed during this talk!