Virtual Observatory

Jaroslav Vazny

Department of Theoretical Physics and Astrophysics

Masarykova Univerzita

A thesis submitted for the degree of

Master

Yet to be decided

I would like to dedicate this thesis to my loving parents ...

Acknowledgements

And I would like to acknowledge ...

Abstract

This is where you write your abstract ...

Contents

List of Figures

Introduction

From the dawn of existence astronomy has always been starved for data, but in the last few decades the situation has changed and now we are facing the data deluge of biblical proportions. The data are not just increasing in size but in complexity and dimensionality. Ball and Schade [2010] Astroinformatics is the new field of science which has emerged from this technology driven progress. Virtual Observatory, Machine Learning, Data Mining, Grid Computing are just few examples of new tools available to scientist.

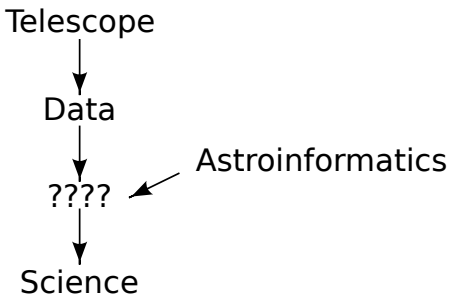

Figure 1: Astroinformatics in the context of astronomy Ball and Schade [2010]

Of course astronomers are not alone and particle physics, biology and other sciences are also in the vanguard of the data intensive science. This is great opportunity for interdisciplinary collaboration.

This work deals with the problem of semi-automatic procedures for finding Be stars Porter and Rivinius [2003] candidates in the astronomy surveys. More than straight forward process it's trail and error approach probing new possibilities with rather interesting that useful results.

The aim of this work is to be introductory to the technologies of Virtual Observatory and Data Mining and for this reason it is intended to have following properties:

- Main Chapters starts with diagram to ease orietation,
- is full of examples,

- is non-linear in nature,
- is meant to be compact and consistent,
- is far from complete.

Chapter one is an introduction to the technologies related to Virtual Observatory. The motivation behind the concept is given without paying too much attention to historical details. Main principles and protocols are discussed and explained. Important aspect are demonstrated on numerous examples. Chapter two is an introduction to Machine Learning and Data Mining in the context of astrophysics. Only methods used in practical part of this work are described in detail: Decision Trees and Support Vector Machines. Examples of several classifications are demonstrated. Third chapter introduces problematic of Be stars. Chapter Four is practical application of previously described technologies and methods. Training data of confirmed Be stars from Ondrejov are correlated with others catalogues to obtain color indexes and spectra. Results are processed by Data Mining algorithms using several libraries and tools. In the last chapter achieved results are critically discussed.

Many scripts were written to achive individual goals. In the text there are numerous commented snippets of codes. Their purpose is to demonstrate the concept and they are thefore short and without auxiliary technicalities such are error handling etc. They are mostly Python and shell scripts. Any interested person can obtain the full source codes (including thesis itself) from GIT repository at `git://github.com/astar/diplomaWork.git`

Name	Input	Output	Description
analyse.py	Spectrum	Spectrum	Check the
getSpectraList.py	List of objects	SSA Compliant list	
getSpectra.py	SSA Compliant list	Links to spectra	
getMax.py	Spectrum	List of extracted values	
reduce.py	Spectrum	Reduced spectrum	
pf.py	Spectrum	Figure of the spectrum	
dm.sh	Trainnig and testing sets	List of classified objects	

Table 1: Scripts developed within the scope of the thesis.

Activities related to this work went beyond this text. Wiki pages[1] were created to present the results and discuss related topic with supervisor as well as with others scientist around the world. Source codes were maintained by GIT version system allowing easy sharing. All software used and produced are open source.

[1] `http://physics.muni.cz/~vazny/wiki/index.php/Diploma_work`

Chapter 1

Virtual Observatory (VO)

1.1 Data avalanche: Opportunity or disaster?

There are two important trends in current astronomy surveys:

- Size: The cumulative compressed data holdings of the ESO archive will reach 1 PetaByte by 2012 Hanisch and Quinn [2010]. Projects like Large Synoptic Survey Telescope (LSST) will produce about 30 TB per night, leading to a total database over the ten years of operations of 60 PB for the raw data Becla et al. [2006].
- Complexity: Modern surveys will cover the sky in different wave-bands, from gamma- and X-rays, optical, infrared, through to radio. The ability to cross correlate these observations together may lead to the new understanding of physical phenomenas. Hanisch and Quinn [2010]

Such amount of data is not possible to transfer over the network. Data resources are heterogeneous, distributed and decentralized in nature.

There is an interesting analogy with the problem (and the solution) which had scientist during LEP project at CERN. Their problem was too many documents in different formats. Tim Berners-Lee[1] designed set of protocols (URIs, HTTP and HTML) which allowed link and share documentsBerners-Lee and Cailliau [1990]. This was recognized as generally useful and World Wide Web was born. An important role plays the World Wide Web Consortium (W3C) in developing Web standards[2].

1.2 International Virtual Observatory Alliance (IVOA)

[1] Sir Timothy John "Tim" Berners-Lee. British engineer and computer scientist and MIT professor credited with inventing the World Wide Web.

[2] Prior to its creation, incompatible versions of HTML were offered by different vendors, increasing the potential for inconsistency between web pages.

What is necessary is sets of standards and protocols to deal with heterogeneous distributed data and the authority which encourages their implementation. Such authority is the International Virtual Observatory Alliance (IVOA). It comprises 19 VO programs from Argentina, Armenia, Australia, Brazil, Canada, China, Europe, France, Germany, Hungary, India, Italy, Japan, Russia, Spain, the United Kingdom, and the United States and inter-governmental organizations (ESA and ESO)Hanisch and Quinn [2010].

Figure 1.1: IVOA members

Standards and specifications produced by IVOA can be obtained at http://www.ivoa.net/.

1.3 Architecture

The Architecture is depicted on the figure 1.2.The level of abstraction goes from top to bottom. Starting with interfaces, used by people or applications to discover resources. Next level is the service layer implemented by standard protocols, followed by the hardware level where actual data are stored. This onion like structure hide the complexity of the lower layer and provide data and meta-data to the higher layer. This concept is similar to TCP/IP [1] protocol.

[1]TCP/IP (Transmission Control Protocol/Internet Protocol). The basic communication language or protocol of the Internet.

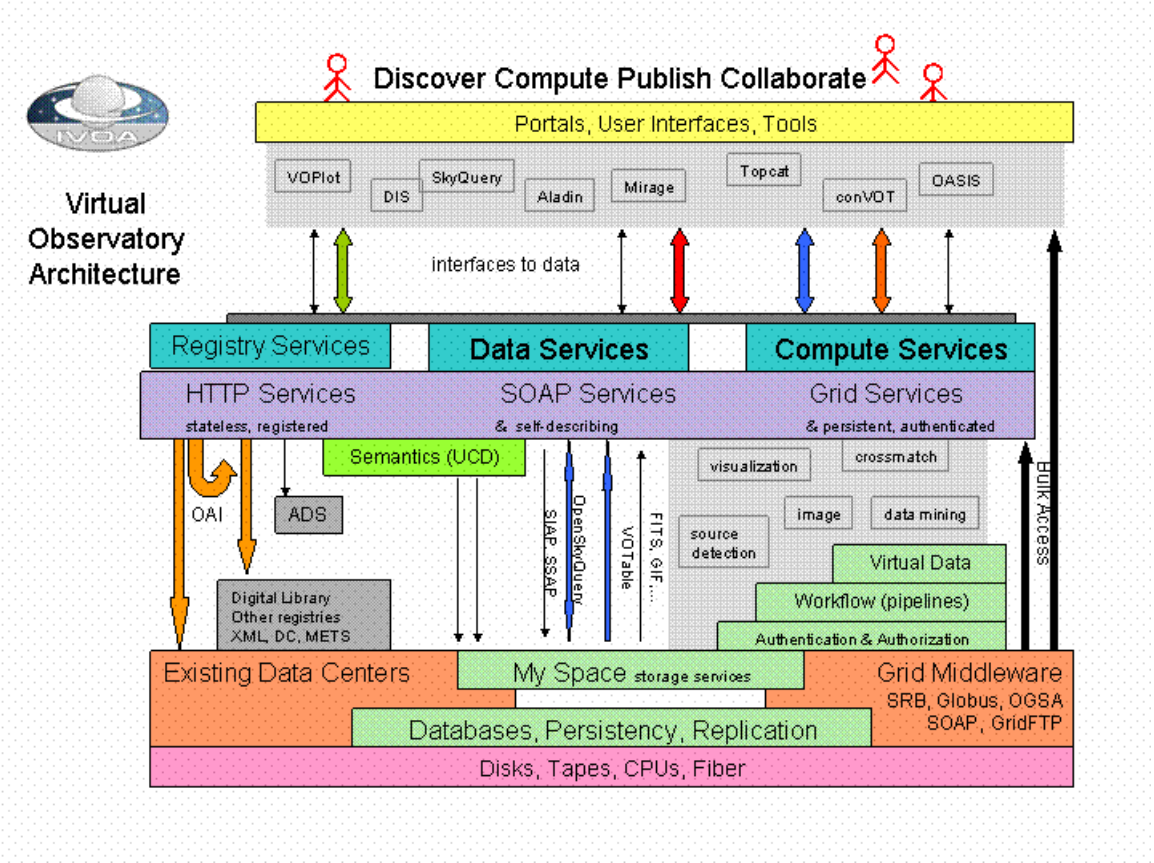

Figure 1.2: VO Architecture

The essence of VO architecture is service orientation. Each service is autonomous with well defined defined boundaries. Very important aspect of VO implementation is the adoption of formats and protocols used in astronomy (FITS) and computers science (XML [1] , Web service [2] SOAP [3]) for many years. In other words VO does not try to reinvent the wheel but it's stands on the shoulders of giants.

1.4 VOResources

A resource is a general term referring to a VO element that can be described in terms of who curates or maintains it and which can be given a name and a unique identifier. Just about anything can be a resource: it can be an abstract idea, such as sky coverage or an instrumental setup, or it can be fairly concrete, like an organization or a data collection. Benson et al. [2009]

UML[4] diagram of the resource in on the figure 1.3. Next paragraph is an attempt to explain this diagram to non-programmers. Full arrow means generalization, Resource can be a generalization of organization, data collection, application or service. Single arrow means association. Organization can be linked (associated) together with other organization (multiplicity is represented by number 1, 0..). The same is true for data collection. Organization is a generalization of and/or provider which can own zero to N services. Diamond means aggregation Publisher can have any resources.

[1]Extensible Markup Language (XML) is a set of rules for encoding documents in machine-readable form.

[2]method of communication between two electronic devices over a network.

[3]Simple Object Access Protocol, is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks.

[4]Unified Modeling Language. Standardized general-purpose modeling language in the field of object-oriented software engineering.

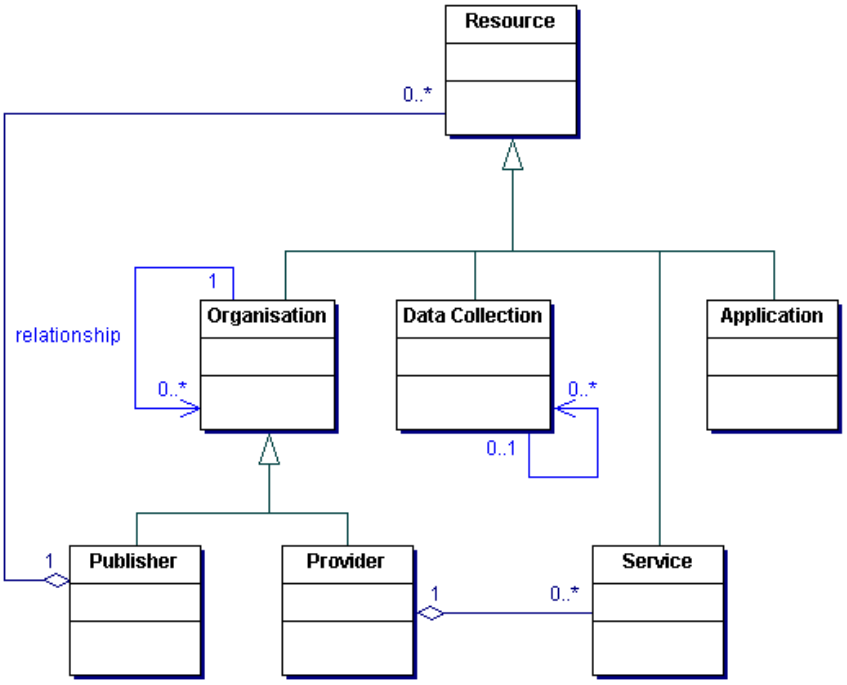

Figure 1.3: UML diagram of VOResource

Following example uses program stilts[1] to query registry with parameter shortName equal to 'AIASCR'[2]. This return VOTable containing informations about the resource.

```
1 stilts regquery query="shortName like 'AIASCR'"
2 regurl=http://registry.euro-vo.org/services/RegistrySearch
3 ofmt=votable-tabledata > resourceExample.vot
```

Rows 1–4 define XML nad VOTable schema with adequate locations (xmlns[3]) followed by informations about the actual resource. The listing is abbreviated.

```
1  <?xml version='1.0'?>
2  <VOTABLE version="1.1"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns="http://www.ivoa.net/xml/VOTable/v1.1">
5  .
6  <DATA>
7  <TABLEDATA>
8    <TR>
9      <TD>ivo://asu.cas.cz</TD>
10     <TD>AIASCR</TD>
11     <TD>Astronomical Institute of the Academy of Sciences of the Czech Republic
           Naming Authority</TD>
12     <TD>http://stelweb.asu.cas.cz/web/index/index-en.php</TD>
13     <TD>Petr Skoda &lt;skoda@sunstel.asu.cas.cz&gt;</TD>
```

1.5 Data Access Protocols

Protocols are very important part of Virtual Observatory. Their understanding is key to comprehend the concepts behind VO. They allow to discover resource and obtain desirable data. All of them are based on existing web standards and are designed to be simple and therefor easy to implement on existing astronomical archives. The main idea is simple and universal: HTTP GET request with parameter is sent to the resource and the structural document (VOTable) is sent back.

1.5.1 Cone Search Protocol

Cone Search was the first standard protocol of Virtual Observatory. It enables to retrieve records from an astronomical catalog. The input is the query which describes sky position and the radius on the sky. The output is a list of objects whose positions lie in the defined

[1] STIL Tool Set. Set of command-line tools based on STIL, the Starlink Tables Infrastructure Library.
[2] Astronomical Institute of the Academy of Sciences of the Czech Republic
[3] XML namespaces. Provide uniquely named elements and attributes in an XML document.

vicinity. The output is formatted as a VOTable. Service compliant with Cone Search Protocol is called Cone Search Service. Only the request and response is specified not the implementation or data storage.

The requirements are:

1. Respond to a HTTP GET request represented by a URL

```
http://<server-address>/<path>?[<extra-GET-arg>&[...]]
```

   The constrains are expressed as a list of ampersand-delimited GET arguments. For example:

```
http://simbad.u-strasbg.fr/simbad-conesearch.pl?RA=24.5&DEC=-57.2&SR=0.1
```

   Where RA is right-ascension, DEC declination and SR the radius of the cone in the ICRS coordinate system in decimal degrees.

2. Return an XML document in the VOTable format.

   There are several requirements on the contents of the table:

   - UCD fields "ID_MAIN", "POS_EQ_RA_MAIN", "POS_EQ_DEC_MAIN" must be present.
   - Return VOTable with single PARAM element name="Error" in the case of error.

Cone Search is implemented in many software packages. Besides standard VO tools like TOPCAT or STILTS also in MUNIPACK and many others. Following example show simple query to SIMBAD**?** catalog using method *urlopen* of Python library *urllib2*.

```
import urllib2
response = urllib2.urlopen('http://simbad.u-strasbg.fr/simbad-conesearch.pl?
    RA=24.5&DEC=-57&SR=0.1')
print response.read()
```

The same result can be obtained using program like wget[1] or Web browser.

1.5.2 Simple Image Access Protocol

The key idea behind the SIA Protocol is to allow users and programs to retrieve images created by an image service on-the-fly. From technical point of view it is designed in a similar way as Cone Search Protocol (see 1.5.1), specifically as name-value HTTP GET requests and the VOTable XML format output. The user specify ideal image (position and

[1] program for non-interactive download of files from the Web

the size) he wants to receive and the image service produces a list of images it can return in VOTable format. The user then could issue getImage request to retrieve desirable images.

There are following requirements for compliance. To be a SIA service To be a SIA service, it must support:

- Image Query web method,
- Image Retrieval (getImage) web method.

Furthermore the image service should be registered to allow locate optimal service. There are several types of image services:

- Image Cutout Service.

  Provides rectangular regions of large images.

- Image Mosaicing Service.

  Size, scale and projection could be specified.

- Atlas Image Archive

  Pre-computed atlas of images.

- Pointed Image Archive.

  Images are not part of a sky survey but rather focused on specific source

To get a list of images query has to send via HTTP GET method. The first part is base URL. The second part are parameters specifying image properties such as position (POS), size (SIZE), etc.

```
1 http://<server-address>/<path>?[<extra GET arg>&[...]]
```

Example of using SIA protocol to obtain image from SDSS.

```
1 http://skyview.gsfc.nasa.gov/cgi-bin/vo/sia.pl?SURVEY=SDSS&POS
     =18.87667,-0.86083&SIZE=1
2 http://hubblesite.org/cgi-bin/sia/hst_pr_sia.pl?POS=83.6,22.0&SIZE=1.0
```

There is more complex example using Astrogrid framework to show how to discover SIA service and obtain an image. First registry method searchSiap is used to find SIA service for SDSS, this is then used in SiapSearch method to obtain result in VOTable format.

```
In [1]: from astrogrid import Registry, ConeSearch
In [2]: list = reg.searchSiap('SDSS')
In [3]: print [p['id'] for p in list]
-------> print([p['id'] for p in list])
['ivo://nasa.heasarc/skyview/sdss']

In [4]: siap = SiapSearch('ivo://nasa.heasarc/skyview/sdss')
In [5]: result = siap.execute(18.8, -0.8, 1.0)
```

1.5.3 Simple Spectra Access Protocol

SSA Protocol allow to discover and obtain 1-D spectra from VO Service. It shares many similarities to the previously discussed SIA Protocol.

defines a uniform interface to remotely discover and access simple 1-D spectra.
similar to that of the older Simple Image Access (SIA)
The process to obtain a spectrum compose of following steps:

- Query the resource registry.
- Data discovery to selected service to get aviable resources in VOTable format.
- Download selected spectra using URL.

The spectra could be on of the following types:

- Pre-computed
- Computed on the fly

To be a SSA-compliant, the service must provide:

1. HTTP GET interface, returning the query response encoded as a VOTable document, with at least parameters POS, SIZE, TIME, BAND, and FORMAT.
2. GetData method returning data in at least one of the SSA-compliant data formats (VOTable, FITS)
3. FORMAT=METADATA metadata query feature

Following example show how to discover resources with SSA capability using STILTS program.

```
stilts regquery query="shortName like 'ESO' capability/@standardID =
'ivo://ivoa.net/std/SSA'" ocmd="keepcols 'ShortName accessUrl'"
ofmt=ascii
```

With information of service URL, one can specify a query to obtain a list with aviable spectra in VOTable format. This can be used in Web browser or via programs such *wget* or *curl*.

```
http://archive.eso.org/apps/ssaserver/EsoProxySsap?REQUEST=queryData&POS
   =83.63,22&SIZE=1
```

1.6 Data Formats

Astronomy has always been, by its nature, on vanguard of image producing and processing. This is especially true for the era of digitalization. The situation with data formats in astronomy is unique. There are just few very good standards with variety of implementation in many programming languages. Virtual Observatory takes advantage of this heritage and implement these formats in sensible way into its own standards.

1.6.1 VOTable

Motivation

VOTable is flexible storage and exchange format fundamentally interconnected with Virtual Observatory. It has features for big-data and Grid computing. Data can be stored in different ways in dependence on the charatecter and size. Small tables can be stored in pure XML [1], while large-scale data can be referenced with the URL [2] syntax protocol://location. It combine web standards (it is based on XML) and astronomy tradition in storing data (it is FITS compatible). Expiration and authentication are also supported.

Structure

Following example of VOTable was created from SDSS FITS file used in this work. First there is an information about XML and VOTable versions and references to corresponding XML Schema [3]. ¡TABLE¿ tag encapsulating tabular data. ¡FIELD¿ tag describe

[1] Extensible Markup Language. W3C standard. Set of rules for encoding documents in machine-readable form

[2] Uniform Resource Locator. Uniform Resource Identifier (URI) that specifies where an identified resource is available and the mechanism for retrieving it.

[3] Define the legal building blocks of an XML document. Note: XML schema can be described by XML Schema, Document Type Definition (DTD) or RELAX NG

identifinaction (ID), type and precision of columns. ¡DATA¿ tag contains data (here) in TABLEDATA format (other types are FITS and BINARY)

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Produced with vo.table version 0.6
     http://www.stsci.edu/trac/ssb/astrolib
     Author: Michael Droettboom <support@stsci.edu> -->
<VOTABLE version="1.0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="http://www.ivoa.net/xml/VOTable/v1.0"
 xmlns="http://www.ivoa.net/xml/VOTable/v1.0">
 <RESOURCE type="results" >
  <TABLE >
   <FIELD ID="col0" name="wave" datatype="float" unit=""
   precision="F9"/>
  <DATA>
    <TABLEDATA>
     <TR>
      <TD>4012.50757</TD>
     </TR>
 </TABLEDATA>
   </DATA>
  </TABLE>
 </RESOURCE>
</VOTABLE>
```

Examples

All example created using ATpy[1]

Following example shows transformation FITS into VOTable.

```
In [1]: import atpy
In [2]: tbl = atpy.Table('spSpec-53401-2052-458.fit',hdu=1)
Auto-detected input type: fits
In [3]: tbl.write('votableExample.xml')
Auto-detected input type: vo
```

[1]High-level Python package providing a way to manipulate tables of astronomical data in a uniform way.

1.6.2 FITS

Motivation

"An archival format must be utterly portable and self-describing, on the assumption that, apart from the transcription device, neither the software nor the hardware that wrote the data will be available when the data are read." Council [1995]

FITS (Flexible Image Transport System) was originally created for data exchange between WSRT [1] and the VLA [2] Schlesinger [1997]. It is now used as a file format to store, transmit, and manipulate scientific data and it is (thanks to it's revolutionary design) de facto standard in astronomy.

Structure

One file can contain several HDUs (Header Data Units).The first part of each HDU is the header, composed of ASCII card images containing keyword=value statements that describe the size, format, and structure of the data that follow.

- Primary header and data unit (HDU).
- Conforming Extensions (optional).
- Other special records (optional, restricted).

Standards and documents related to FITS are maintained by IAUFWG [3] and aviable at http://fits.gsfc.nasa.gov.

Examples

There are many libraries for working with FITS files. The official list is aviable at http://fits.gsfc.nasa.gov/fits_libraries.html. PyFITS, library for Python programming language was used for following examples. PyFITS is a development project of the Science Software Branch at the Space Telescope Science Institute http://www.stsci.edu/resources/software_hardware/pyfits.

Reading FITS headers.

```
1 In [1]: import pyfits
2 In [2]: hdulist = pyfits.open('spSpec-53237-1886-248.fit')
3 In [3]: hdulist.info()
4 Filename: spSpec-53237-1886-248.fit
5 No.    Name        Type      Cards   Dimensions   Format
```

[1] Westerbork Synthesis Radio Telescope
[2] Very Large Array
[3] International Astronomical Union FITS

```
0    PRIMARY    PrimaryHDU    213  (3874, 5)    float32
1               BinTableHDU    54  6R x 23C     [1E, 1E, ...
2               BinTableHDU    54  44R x 23C    [1E, 1E, ...
3               BinTableHDU    18  1R x 5C      [1E, 1E, ...
```

Printing primary HDU.

```
In [4]: print hdulist[0].header
-------> print(hdulist[0].header)
DATE-OBS= '2004-08-20'      / 1st row - TAI date
TAIHMS  = '10:36:18.11'     / 1st row - TAI time (HH:MM:SS.SS) (TAI-UT = appr
TAI-BEG =      4599713999.00 / Exposure Start Time
TAI-END =      4599717089.00 / Exposure End Time
MJD     =              53237 / MJD of observation
MJDLIST = '53237 '          /
VERSION = 'v3_140_0'        / version of IOP
TELESCOP= 'SDSS 2.5-M'      / Sloan Digital Sky Survey
```

Updating FITS file.

```
  In [1]: prihdr = hdulist[0].header
  In [2]: prihdr.update('observer', 'Astar')
  In [3]: prihdr.add_history('I updated this file 3/27/11')
```

1.7 Tools & Libraries

There are many programs and libraries allowing user to interact with VO services. Such application is called VO Enabled. Thanks to openness and standardisation anyone can develop his own application or enable existing[1] application to interact with VO Sevices . Libraries are also aviable for many programming languages enabling advanced users to interact with VO from scripts and programs. Such diversity is healthy and propably the only possible way to ensure natural evolution of Virtual Observatory. This chapter describes some of the libraries and applications ued during this work. Low level tools and libraries are stressed as opposite to standard introductory texts of Virtual Observatory where the emphasis is on "user friendly" GUI applications.

1.7.1 Command line tools

Astro Runtime

middleware that makes it simple to call Virtual Observatory services from any programming language

[1]For example Astroweka or Mirage.

1.7.2 GUI Appliactions

Chapter 2

Data Mining

Virtual Observatory may be seen as data infrastructure. It enables astronomers to get data more easily in a uniform way. But there is another and even bigger problem now. How to deal with such amount of data? Can we change the problem to opportunity? Can we discover new phenomena, new types of objects or exploit natural groups in the data? Data Mining and related techniques are created exactly for such purposes. Used correctly, it can be powerful approach, promising scientific advance. On the other side this field is very complex with dozens of different methods and algorithms. This forms needs and opportunity for interdisciplinary cooperation with Data Mining experts. This can be very beneficially for both fields, providing astronomers with interesting methods for data analysis and computer scientist with large ammout of quality data.

2.1 Supervised Methods

These methods are also known as predictive**?**. They rely on training set with known target property. This set must be representative. The selected method is trained on that set and the result is then used on data for which the target property is not known. Among supervised method are classification, regression, anomaly detection and others.

2.1.1 Decision Tree (DT)

Is an example of supervised classification. Based on final number of data $(x^{(1)}, \ldots, x^{(p)})$ with known class $C_1, \ldots, C_m$ classifier is created, i.e. image f classifying any $x \in \mathcal{X}, f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ is a set of possible input vectors and $\mathcal{Y}$ is a set which values represents classes $C_1, \ldots, C_m$ (for example $\mathcal{Y} = 1, \ldots, m$). The model is consctructed based on traning set as a tree structure, where leaves represent classifications and branches conjunctions of features that lead to those classifications. The main advatages of DT are:

- Simple to understand and interpret.

- Able to handle both numerical and categorical data.
- Uses a white box model.
- Perform well with large data in a short time.

In pseudocode, the general algorithm for building decision trees is **?**:

1. Check for base cases
2. For each attribute a
    - Find the normalized information gain from splitting on a
3. Let "a best" be the attribute with the highest normalized information gain
4. Create a decision node that splits on "a best"
5. Recur on the sublists obtained by splitting on "a best", and add those nodes as children of node

Furtheremore algoritms C4.5 is described for several reasons: Its code is aviable and free implementeations exist (J48 in weka), is de-facto standard in classsifiction using DT, is used in practical part of this work. The key question of DT algoritm is how to choose attribute for splitting the tree. C4.5 Uses meassures based on information entropy:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i), \tag{2.1}$$

where $p(x_i)$ is probability of occurrence of class i and n is the number of classes.

After the tree is created it is optimized by pruning, which prevent over-fitting.

2.1.1.1 Cross-validation

The quality of the training set is crucial to good results. The amount of data for testing is always limited. In general, one cannot be sure whether a sample is representative. If for example certain group is missing, one could not expect a classifier learned from such data to perform well on the examples s of that class. One of the technique used here is cross-validation.

The data is divided into fixed number of partitions and each in turn is used for testing and the reminder is used for training. Finally, the number of partitions error estimates are averaged to yield an overall error. The standard is to used 10-fold cross-validation. This number is a result of tests on numerous data sets **?**

2.1.1.2 Example: Classifying Galaxies Stars and QSO

There is an example of classifying Galaxies Stars and QSO based on photometric properties using Decision Tree algorithm J48 (C4.5 in Weka). The data are from SDSS (Sloan Digital Sky Survey) DR7. 298 Objects were used (100 Stars, 99 Galaxies, 99 QSO). SDSS Filters u,g,r,i were used as parameters. Data were obtained using SQL query from SDSS CAS.

```
SELECT TOP 100 u-g,g-r,r-i,s.specClass
FROM PhotoPrimary p join SpecPhotoAll s on p.objid=s.objid
WHERE s.specClass in (1)
AND u between 18 and 19
UNION all
SELECT top 100 u-g,g-r,r-i,s.specClass
FROM PhotoPrimary p join SpecPhotoAll s on p.objid=s.objid
WHERE s.specClass in (2)
AND u between 18 and 19
UNION all
SELECT top 100 u-g,g-r,r-i,s.specClass
FROM PhotoPrimary p join SpecPhotoAll s on p.objid=s.objid
WHERE s.specClass in (3)
AND u between 18 and 19
```

The following listing shows the result of classification. The classifier was able to distinguish 95% of the processed objects.

Filter	Wavelength [$\AA$],
Ultraviolet (u)	3543,
Green (g)	4770,
Red (r)	6231,
Near Infrared (i)	7625,
Infrared (z)	9134,

Table 2.1: SDSS Filters

```
Correctly Classified Instances          96               95.0495 %
Incorrectly Classified Instances         5                4.9505 %
Kappa statistic                          0.9257
Mean absolute error                      0.0669
Root mean squared error                  0.1778
Relative absolute error                 15.0587 %
Root relative squared error             37.6973 %
Total Number of Instances              101
```

The big advantage of Decision Trees over black box algorithms (such as Neural Network) is that one could understand the classification process. The decision tree generated for this example is following:

```
  ug <= 0.663668
|   gr <= -0.191208: 1 (7.0)
|   gr > -0.191208: 3 (104.0/5.0)
ug > 0.663668
|   ri <= 0.285854: 1 (88.0/5.0)
|   ri > 0.285854
|   |   ri <= 0.314657
|   |   |   gr <= 0.692108: 2 (6.0)
|   |   |   gr > 0.692108: 1 (3.0)
|   |   ri > 0.314657: 2 (90.0/2.0)
```

Useful tool for understanding how classifier was successful on individual classes is the confusion matrix. Columns shows how the object was classified and the row what is his actual class. In this example QSO were classified correctly in 100% cases. Distinction between stars and galaxies are a bit worse and the algorithm classify 2 galaxies incorrectly as stars and two stars were confused with galaxies. On stars was incorrectly classified as QSO.

```
  s  g  q   <-- classified as
 30  2  1 |  s
  2 33  0 |  g
  0  0 33 |  q
```

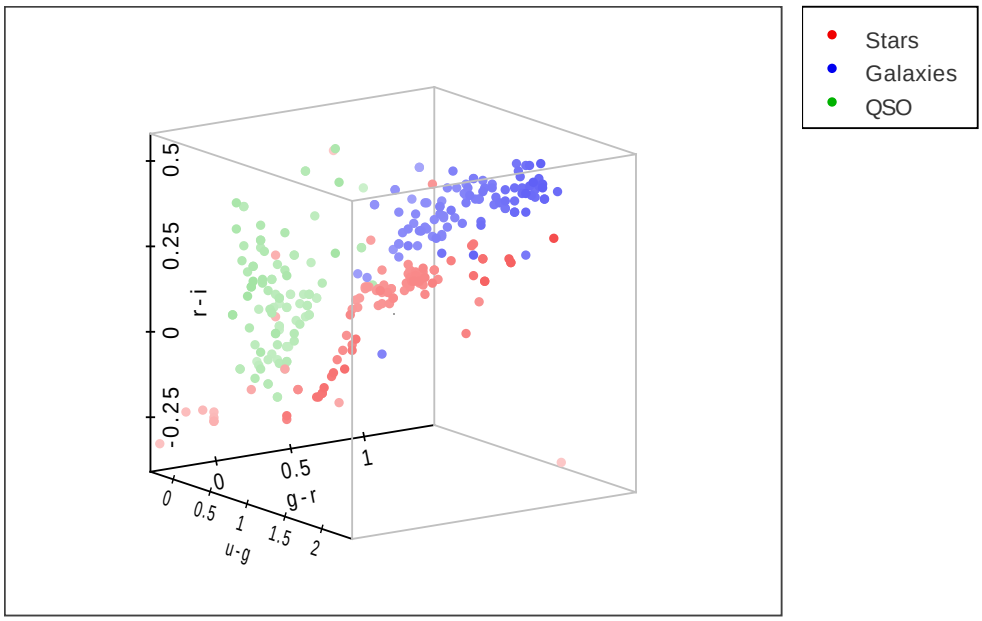

Figure 2.1: Color Diagram of the problem. It shows that individual objects classes occupies different region in the digram.

2.1.2 Support Vector Machine (SVM)

2.2 Unsupervised Methods

2.3 Existing Projects

2.3.1 Weka

2.3.2 SVM lib

2.3.3 DAME

Chapter 3

Be candidates

Astronomical objects used in this work to demonstrate some of the discussed technologies and methods were Be stars. The target was to develop a process of finding new candidates in the available data. Several approaches were considered and two of them are discussed in the rest of this text. First one utilizes photometric properties of Be stars, second uses spectra characteristics.

3.1 Be stars

"Classical Be stars are B-type stars close to the main sequence that exhibit line emission over the photometric spectrum. The excess is attributed to a circumstellar gaseous component that is commonly accepted to be in the form of an equatorial disk." Porter and Rivinius [2003].

3.2 Photometric Data Mining

The question I have tried to answer in this chapter was: Is it possible to find Be stars candidates based on photometric properties only? To answer this question I needed training set of confirmed Be stars, set of non Be stars (spectral type B was considered) and some Data Mining algorithm to perform classification.

3.2.1 Data preprocessing

I was provided by a list of confirmed Be stars from Academy of Science Ondejov. This list consist of 625 manually chosen objects. Data were correlated with Hipparcos **?** catalog to obtain RA, DEC and then with 2MASS**?** catalog to obtain J,H,K Colors using method of multi-cone search in Virtual Observatory. The second set was acquired from Hipparcos catalog using following SQL query:

```
Select *
From maincat as m, hipva1 as h
Where (m.HIP=h.HIP )
And h.SpType Like 'B%'
```

The result was cross-correlated with 2MASS catalog to obtain the same colors as for the confirmed Be stars. Color digram of this two sets are on the figure 3.1

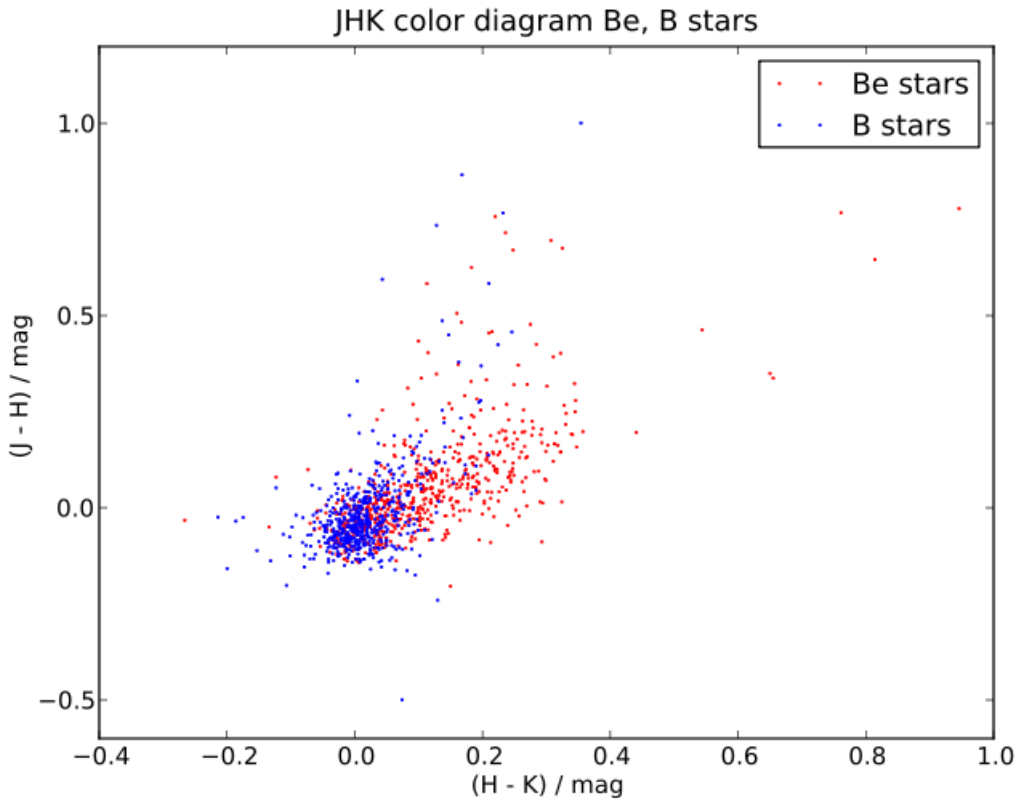

Figure 3.1: Color diagram of confirmed Be stars Vs B stars

The uncertainties were computed for each object using propagation of error. These errors and depicted on the figure 3.2. Although the uncertainties are significant certain trends are presented.

$$\delta_{(j-h)} = \sqrt{\left(\frac{\partial(j-h)}{\partial j}\right)^2 \delta_j^2 + \left(\frac{\partial(j-h)}{\partial h}\right)^2 \delta_h^2}$$

$$\frac{\partial(j-h)}{\partial j} = 1, \frac{\partial(j-h)}{\partial h} = -1$$

$$\delta_{(j-h)} = \sqrt{\delta_j^2 + \delta_h^2}$$

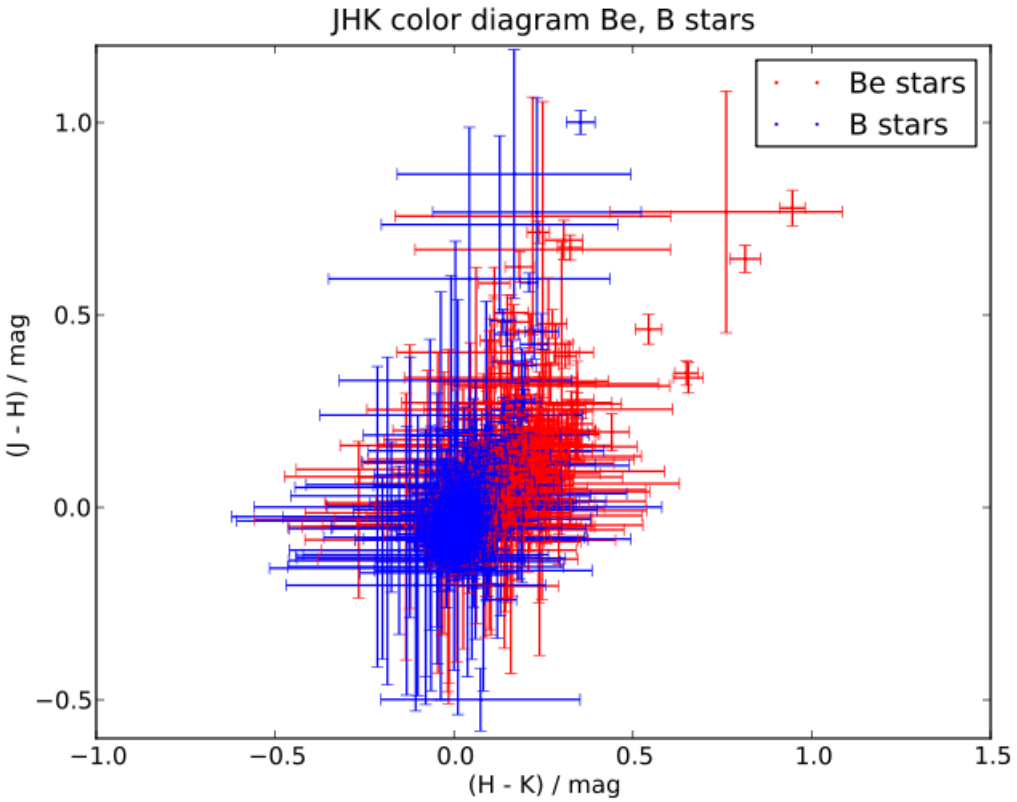

Figure 3.2: Color diagram of confirmed Be stars Vs B stars with errors

3.2.2 Classification

Data were transformed from original VOTable obtained from Virtual Observatory tools to arff[1] format used in Weka Data Mining system. Algorithm C4.5 (J48) was used to perform actual classification with following result:

```
Correctly Classified Instances     769              73.0989 %
Incorrectly Classified Instances   283              26.9011 %
Kappa statistic                      0.4496
Mean absolute error                  0.3843
Root mean squared error              0.4383
Relative absolute error             79.4985 %
Root relative squared error         89.1648 %
Total Number of Instances         1052
```

As seen on the first row 73 % from 1052 objects were classified correctly. More details can be obtained from confusion matrix below.

```
 B   Be   <-- classified as
304 126 |  B
157 465 |  Be
```

304 of B and 456 of Be stars were classified correctly but 126 of B and 157 of Be stars were classified incorrectly. In virtue of these results one should be sceptical if the distinction based only on photometric properties is significant enough to find relevant new candidates of Be stars. For this reason more sophisticated (and much more complicated) approach using spectra analysis was tested.

3.3 Spectral Data Mining

3.3.1 Testing Data

As testing sample the project SEGUE of SDSS were selected. This contains 178315 spectra in DR7. Following SQL query was used to generate the list of URL links for individual FITS files. These files were then download to local sever using wget command.

```
SELECT objid,dbo.fGetUrlFitsSpectrum(s.specObjID)
INTO mydb.segue_1
FROM SpecPhotoAll s, platex p
WHERE s.specObjID is not null
AND s.plateid = p.plateid
AND p.programname LIKE 'segue%'
```

[1] Attribute-Relation File Format. Developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato.

```
AND specClass = 1
```

3.3.2 Training Data

The spectra from Ondejov Observatory were used as training sample. Files were downloaded using SSA protocol. The SSA server is not publically aviable, therefore SSH tunneling was used. Two scripts for this process were created. First to construct the list of SSA compliant adresses, the second to analyse acquired response in VOTable format. Then the spectra were downloaded using wget command. The fuction for constructing the links based on list of the RA, DEC which were obtained from Hipparcos catalog using the specification of IDs from Ondejov's index.

```
def createQuery(data):
    """ From raw data construct ra, dec """
    """ Convert to degrees """
    for line in data:
        ra = ac.AngularCoordinate(line[0:10]).degrees # convert ra to degrees
        dec = ac.AngularCoordinate(line[-13:-1]).degrees # convert dec to
            degrees
        ra = line[0]
        dec = line[1]
        ssaTemp = 'http://tvoserver/coude/coude.cgi?c=ssac&n=coude_ssa&REQUEST=
            queryData&POS=<ra>,<dec>&SIZE=1'
        ssaTemp = ssaTemp.replace('<ra>',"%0.3f" % ra)
        ssaTemp = ssaTemp.replace('<dec>',"%0.3f" % dec)
        ssa.append(ssaTemp)
    return ssa
```

The script generate the following output. The same process were used later for obtaining th sample of non Be stars.

```
http://tvoserver/coude/..._ssa&REQUEST=queryData&POS=83.113,-65.582&SIZE=60
http://tvoserver/coude/..._ssa&REQUEST=queryData&POS=162.537,148.333&SIZE=60
http://tvoserver/coude/..._ssa&REQUEST=queryData&POS=19.907,-73.502&SIZE=60
```

3.3.2.1 Spectra Reduction

Because spectra from SDSS and Ondejov Observatory had different resolution, reduction was needed. First the parameter CD1_1 (Coordinate increment per pixel) had to be obtained form FITS file.

```
In [1]: hdu = pf.open('sdss_test.fit')
In [2]: hdu[0].header['CD1_1']
```

```
Out[2]: 0.0001 # SDSS spectrum
Out[3]: 0.2567 # Onejov spectrum
```

Spectra in SDSS are stored in logarithmic scale thus the value is computed as $10^{CD1\_1} = 1.0002302850208247$. The ratio is then $CD1\_1_{SDSS}/CD1\_1_{OND} = 3.8964580808433253$. Based on this computation 4 pixels of Ondejov's spectra were reduced into one. There is the critical part of the reduction program:

```
def convolution(f, g):
   """ Convolve two functions"""
   fg = np.convolve(g,f,'same')
   return fg
def reduce(x,y,bin):
   """ Reduce bin pixel into 1"""
   size = x.size/bin
   l = 0
   xx = x[:x.size-1:bin]
   yy = list()
   for i in range(0,size):
       s = 0
       for j in range(0,bin):
           s = s + y[l]
           l+=1
       yy.append(s/bin)
   return xx, yy
```

Prior to binning pixels convolution with gaussian function was performed on the spectra. Convolution is defined:

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau \tag{3.1}$$

Here it was used in it's dicrete form

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m]\, g[n-m] \tag{3.2}$$

The result is on the figure

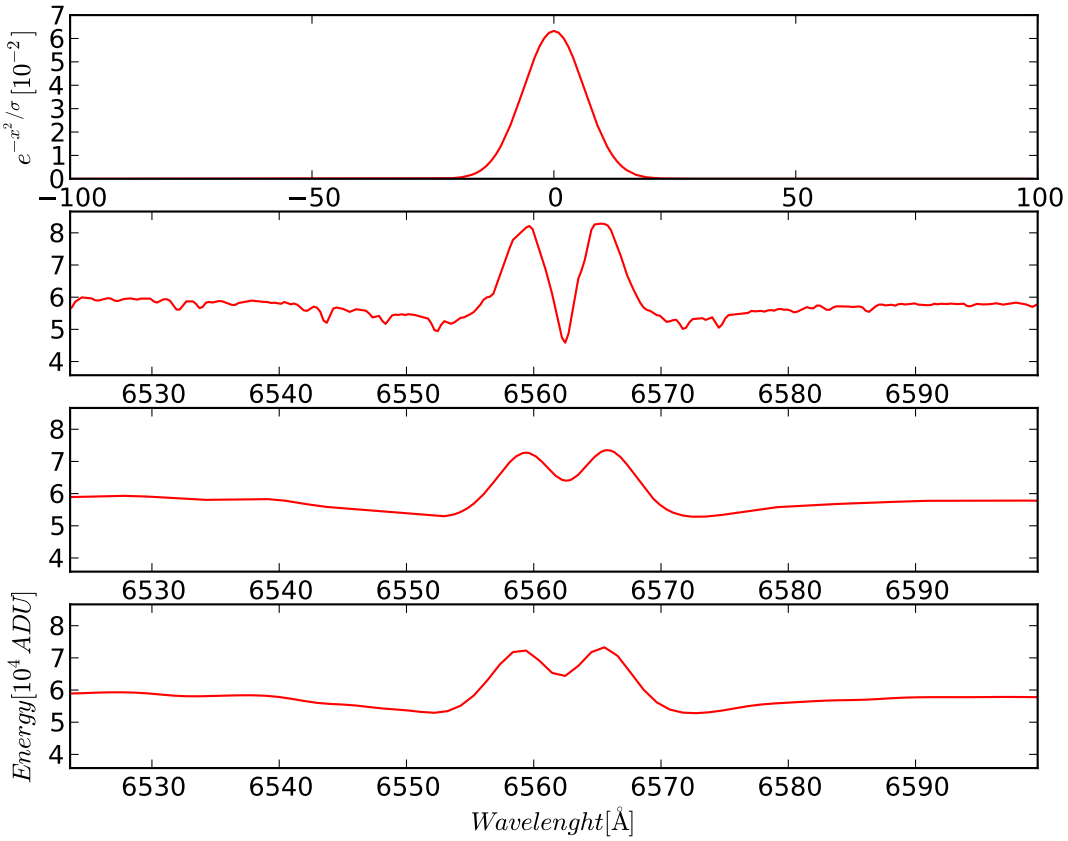

Figure 3.3: Reduction of Ondejov's spectra of the Be star 4 Hercules. The top figure shows gaussian function used for convolution with the structrum, followed by the original spectrum then there is a spectrum after convolution with the gausian function. The last is the final spectrum after reduction.

3.3.3 Spectra Lines Characteristics

As parameters for Data Mining process characteristics value of Hα line were extracted from the spectra. Many possible characteristics from fitting functions through Wavelets Coefficients and Eigen Values were discussed with experts. On the end the most simple approach was used taking the maximum value in the region of 50$\mathring{A}$.

3.3.3.1 Normalization

Spectra from SDSS are normalized but the spectra from Ondejov are not. Therefore the spectra were devided by it's continuum fit function. This process ensures the compatibility when comparing the values for different spectra.

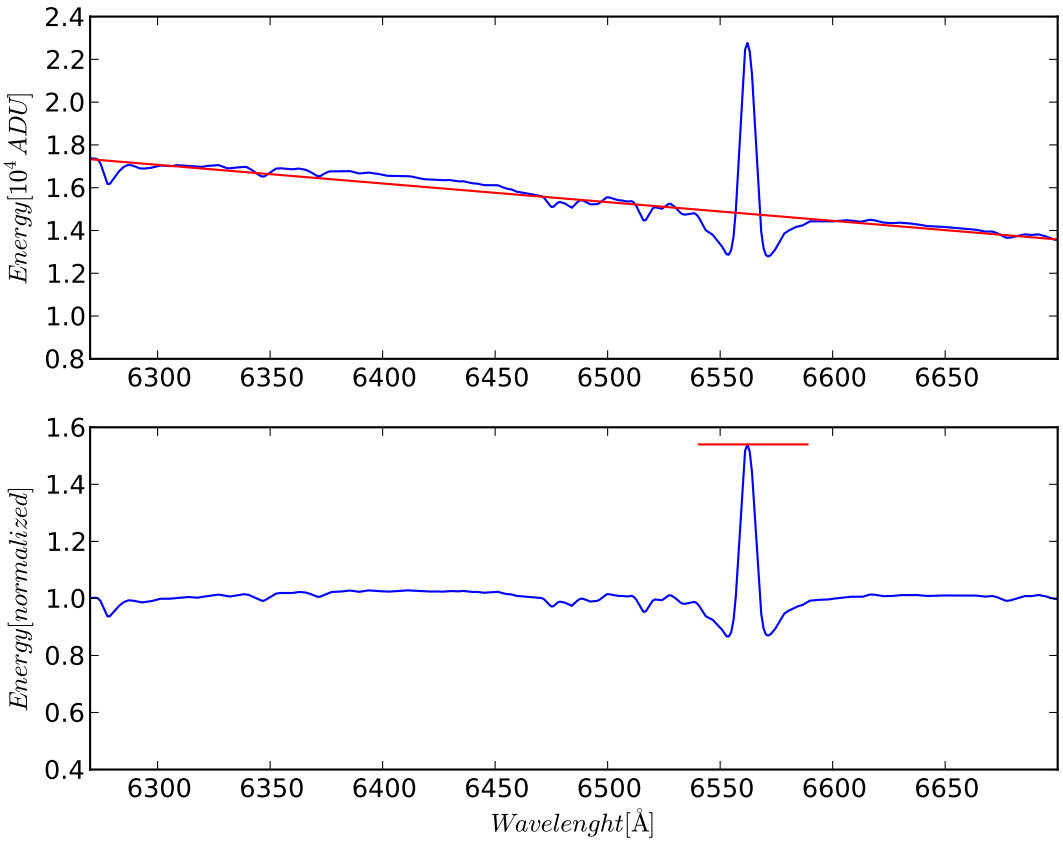

Figure 3.4: Normalized spectrum of β Cygni B (Albireo B). The top figure depicts the continuum fit. The bottom figure shows the region (width of the red line) used for extraction. The position of the line correspond to the characteristic value.

The script was written to normalize the spectrum and extract the line characteristic value. This program also plots the results of the process as it is shown on previous picture. The function used to extract the line characteristic value is below.

```
def getMax(x,y,line,range):
    """ Return maximum value of range in the spectrum"""
    xrange = x[(x < line + range) & (x > line - range)]
    yrange = y[(x < line + range) & (x > line - range)] - 1
    maximum = yrange.max()
    minimum = yrange.min()
    if abs(maximum) > abs(minimum):
        extrem = ( maximum + 1)
        sgn = np.sign(maximum)
    else:
        extrem = (minimum + 1)
        sgn = np.sign(minimum)
    return xrange, extrem, sgn
```

References

N.M. Ball and D. Schade. ASTROINFORMATICS IN CANADA. 2010. vi, 1

J. Becla, A. Hanushevsky, S. Nikolaev, G. Abdulla, A. Szalay, M. Nieto-Santisteban, A. Thakar, and J. Gray. Designing a multi-petabyte database for LSST. *Arxiv preprint cs/0604112*, 2006. 3

K. Benson, R. Plante, E. Auden, et al. IVOA Registry Interfaces. *IVOA Working Draft*, 2009. 6

T. Berners-Lee and R. Cailliau. WorldWideWeb: Proposal for a HyperText project. *European Particle Physics Laboratory (CERN)*, 1990. 3

National Research Council. Preserving Scientic Data on our Physical Universe. 1995. 14

RJ Hanisch and PJ Quinn. The international virtual observatory. *Retrieved from http://www. ivoa. net/pub/info/TheIVOA. pdf on*, 24, 2010. 3, 4

J.M. Porter and T. Rivinius. Classical Be Stars. *Publications of the Astronomical Society of the Pacific*, 115(812):1153–1170, 2003. ISSN 0004-6280. 1, 23

B. Schlesinger. A Users Guide for the Flexible Image Transport System. 1997. 14