Query driven visualization of large scale multi-dimensional astronomical catalogs

Buddelmeijer, Hugo

Chapter 1

Introduction: Large Astronomical Data Sets

Abstract: Astronomical data sets are growing to enormous sizes. Modern surveys provide petabytes of image data and catalogs with billions of sources, each with hundreds of associated parameters. This creates the need for new mechanisms of data mining: data storage, processing, selection, disclosure, visualization and interaction.

Astronomy is a data driven discipline, where raw data is sacred. Data archives turn into information systems, where data and processing is shared. A key element in this is storing information about how each data product is derived, creating a full data lineage of the entire processing chain. Astronomy is a very open field, where data is public by default, which gives astronomy a head start in this trend that is also seen in other sciences.

The use of these information systems requires intensive interoperation between data storage, processing and visualization. By utilizing data lineage this integration will be profound, without integrating everything in one system. One of the things this will lead to is a more declarative way of interaction with the data. Scientists will be able to specify the data they require on a higher level than before: the information system will decide how to fulfill the user's request in an optimal way. This allows for query driven visualization that scales to billions of sources while maintaining interactivity.

A field that will benefit from this approach that has our particular interest is the evolution of galaxies. The open questions require large scale multi-dimensional datasets to be answered. The interaction of galaxies with their environment is important, but not well understood. What properties of galaxies are most influenced by the environment? What are the relative roles of different interaction mechanisms? Where and at what timescales do these mechanisms occur? How should the environment be quantized at all? Exploring the parameter space, algorithm choice and sample selections to answer these questions is a driver in our research.

1.1 Data Explosion in Astronomy

Current optical catalogs like the Sloan Digital Sky Survey contain almost a billion sources with hundreds of parameters per source. In the future, astronomers will be engaged in even larger and deeper surveys to study the evolution of galaxies and cosmology to even greater detail. For example, OmegaCAM will provide a 1500 degree optical survey of galaxies (KIDS). This optical data will be combined with infrared data from the VISTA telescope (VIKING). Euclid plans to cover 20 000 square degree and will detect up to 10^{10} galaxies (Laureijs, 2009).

1.1.1 Growth of Data Volume and Analysis Capacity

The exponential growth of raw data in astronomy is equaled by the growth in processing power, but this is offset by other factors. Bandwidth increases at a slower pace than the data, the current trend is to bring processing to the data instead of the more traditional other way around. Information that will lead us to new insights is buried deeper in the raw data, requiring more processing to retrieve it. The human brain, the last step in data analysis, can only interpret a limited amount of information. It is not insightful to create multi-dimensional scatter plots of thousands of objects. More processing power and smarter techniques are required to compress the data to meet these human limits.

1.1.2 Analysis of Large Scale Multi-Dimensional Data

While this data explosion affects all parts of astronomy, the focus in this thesis is on source catalogs. The data growth of catalog data manifests itself in two ways: the number of objects grows and the number of quantified properties of these objects increases as well.

Analysis: Needles or Haystacks

The growth of detected sources leads to scalability problems, which can be represented in two extremes. On one hand it becomes more difficult to find the 'the needles in the haystack', the few objects that have the specific properties that a scientist wants to study, e.g. evolving galaxies or stars with exoplanets. On the other hand, studying 'the haystack' itself is harder as well, e.g. studying the large scale structure of the universe.

Analysis: Parameter Complexity

Besides cataloging more and more sources, we also quantify more properties of these sources. Dealing with the growth of parameters poses a problem that might be even harder to solve than the scaling of the number of sources.

For example, it is common in astronomy to combine data from different telescopes and surveys: La Barbera et al. (2008) study the fundamental plane with optical SDSS and infrared UKIDSS data. This increases the dimensionality of the data rapidly,

such as in color space. Color is defined as the difference between two magnitudes that correspond to different spectral bands. With five bands (e.g. SDSS) there are ten different colors, with two times five bands (e.g. SDSS and UKIDSS combined) there are 45 different possible colors. Exploring data with such increased complexity poses a challenge.

1.1.3 Collaboration and Sharing

In the last couple of decades, astronomical research is more and more performed by groups of scientists. It has become harder to share data for collaboration, because growth in bandwidth does not follow the growth in the volume of the data. The current trend is to store data on central servers from which the data is distributed only if necessary.

The coordination of these groups can be tight or loose: it is even possible for collaborators to have never met in person. Furthermore, due to the internet, data is more easily shared between scientists that do not have a formal collaboration at all, e.g. through the Virtual Observatory. For sharing data in this manner, it is essential that not only the data itself is shared, but also information about what the data represents and how it is created.

1.1.4 Interactive Exploration and Visualization

Besides effects on storing and processing the data, larger data sets also cause problems for visualizing and exploring the data interactively. Most visualization software does not have the scalability required for displaying millions of objects. A close interoperation between visualization tools and the processing and storage systems is necessary to tackle the scalability issues.

1.2 Galaxies as a Multi-Dimensional Problem

The evolution of galaxies is a field where large scale, multi-dimensional data sets are needed. There are about 200 billion galaxies in the entire universe, about the same number as stars in the Galaxy. It is expected that we will be able to observe 1 to 2% of the galaxies, which sets the scale for future data sets. Studying the differences between so many galaxies and understanding the cause of these differences is a daunting task. We are beginning to understand the general trends in galaxy evolution, but the relative importance of various effects are still unknown (section 1.2.2).

An important element in these relations is the environment of the galaxies. The properties of galaxies are very different in low-density voids than in the high-density centers of galaxy clusters. The intermediate densities between the field and the cluster is a most interesting region in this parameter space, because galaxies interact the most with their environment at these densities. There are several proposed mechanisms that drive the evolution of galaxies that could explain some of the perceived trends

and relations, but evidence for these is often anecdotal. The relative roles of the mechanisms is therefore unclear.

1.2.1 Galaxy Relations

Even before the 'extragalactic nebulae' were confirmed to be galaxies, they were already classified by Hubble in two types: ellipticals and spirals. Hubble called these respectively *early types* and *late types*, however, these labels do not refer to an implied temporal evolution (Hubble, 1927). This denomination is still often used today.

In color space, this bimodality of galaxies is even stronger (Strateva et al., 2001; Ball et al., 2008): red galaxies are in general ellipticals and blue galaxies are spirals. This bimodality has been extended and correlates with a number of other parameters of galaxies (table 1.1). The *Morphology-Density Relation* (Dressler, 1980; Whitmore et al., 1993) states that the fraction of elliptical galaxies to spiral galaxies correlates with the local galaxy number density: the higher the density the more early-type galaxies. This relation is even stronger for high luminosity galaxies (Ball et al., 2008), and also correlates with mass and concentration (Baldry et al., 2006). There are more properties that correlate. For example, at fixed mass, nuclear activity depends strongly on density, and star formation —a property which is relatively hard to quantify— correlates extremely well with density (Kauffmann et al., 2004).

Attribute	**Early Types**	**Late Types**
Morphology	Elliptical	Spiral
Color	Red	Blue
Environment	High density	Low density
Structure	Concentrated	Extended
Mass	High Mass	Low Mass
Population	Old Stars	Young Stars

Table 1.1: Bimodality of Galaxies

The bimodality described in table 1.1 is simplistic. For example, there are also red spirals, and blue ellipticals (Bamford et al., 2008). The correlations also differ in strength: Van der Wel (2008) shows that morphology correlates mainly with environment, but structure mainly with mass. In particular, galaxies that do not fall neatly into these categories can give rise to new insights which cannot be found by studying the trends alone. These are often evolving galaxies such as the flaming blue galaxies of Braglia et al. (2007).

Although general relations between properties of galaxies are known, it is not clear what combinations of parameters best represent primary relations and which parameters only correlate due to secondary relations. Astronomical data sets have grown to contain hundreds of parameters, from which the important ones have to be extracted. Data analysis software should be able to deal with these very large number of parameters in our attempts to study the relations between them.

1.2.2 Galaxy Evolution

Galaxies are not static objects, in their lifetime of several billion years they evolve. Their stellar (and interstellar) composition changes, as well as non-local properties such as their shape. The prevailing idea, in a very simplified form, is that galaxies form as late types, fall into galaxy clusters and by interaction with their environment transform into early types. During this evolution, the properties listed in table 1.1 change from values in the right (late type) column to values in the left (early type) column. These changes are not immediate, they happen on different timescales and at different epochs in the evolution of the galaxy. The opposite direction is also considered to be a possibility. Left alone, an elliptical galaxy might relax into a disk galaxy with an elliptical shaped bulge in its center.

The evolution of a galaxy can happen on short timescales with respect to the total lifetime of the galaxy. This means only a small fraction of the galaxies are currently undergoing such a transition. The study of these different evolutionary phases of galaxies led to the proposal of several mechanisms that can make a galaxy evolve. These mechanisms include passive evolution, strangulation (Larson et al., 1980), ram pressure stripping (Gunn and Gott, 1972) and minor and major mergers.

In general, these mechanisms occur at different epochs of the galaxy evolution and in different environments (Moran et al., 2007). An open question is what the relative roles of these mechanisms are. Because they can happen on a short timescale and can cause very abrupt changes in the galaxy (e.g., star formation bursts or sudden cessation of star formation), galaxies that are actually involved in such a mechanisms are rare and therefore studied mostly anecdotal.

To perform a better, quantitative analysis of which role each mechanism plays in order to derive a representative picture of how galaxies evolve, large data sets are necessary.

1.2.3 Nature or Artifact: the Influence of Algorithms

The parameters used in studying the evolution of galaxies can have a complex derivation. Often, a scientist has his or her preferred method of calculating a specific parameter. The methods we choose to quantify the properties of the galaxies can have an influence on the results. This is especially true for parameters that can not be measured directly from image data, such as the mass of an object.

The environment of a galaxy is a property that is hard to quantify. There are different methods to estimate it, each with different pros and cons and free parameters to set. The properties in table (1.1) correlate with the environment of galaxies. Evolution of galaxies occurs mostly in the boundaries of clusters, which are regions of intermediate density. The density of these regions is the hardest to quantify correctly. The effect of different methods to quantify the environment on the relations between galaxies is not well understood yet.

To study the effects of algorithm choice, the system processing the data should be flexible and transparent in the choice of method. Furthermore, it is beneficial if exploration of different methods is possible on a high level, such as from within visu-

alization software. Due to sharing, scalability and interaction requirements, systems designed for large datasets can have properties that are well suited for these goals.

1.2.4 Effects of Sample Selection

Just like in any other field of science, the proper choice of samples is crucial in astronomical research. First of all, there is always a selection effect inherently in the data, due to the limits of instrument used to collect it. Furthermore, explicit sample selection is performed at different stages in astronomical analysis: when accessing the survey data, before and after calculation of parameters and while visualizing the data.

In a field where experiments are mostly impossible, astronomers have become well aware of selection effects and the influence they can have. As an example relating to galaxy evolution, the relation between the fraction of early type galaxies and density depends on whether a luminosity selected sample or a mass selected sample is used (Holden et al., 2007; Van der Wel et al., 2007).

Catalogs with billions of sources make managing these selections more difficult but also more important. For example, it can be on the one hand too expensive (in bandwidth or processing power) to have a selection that is broader than needed, but on the other hand having to redo calculations because the selection was too narrow is wasteful as well. A flexible sample selection mechanism is essential to for a system managing such large datasets. This is especially the case in the data exploratory phase where selections are often performed interactively.

1.3 Techniques for Handling Large Scale Data

In the previous section we pointed out that astronomical research requires large scale multi-dimensional data sets. We identified several requirements for the software dealing with such data. The way data is handled should

- be scalable to billions of sources with hundreds of parameters,
- facilitate sharing of the data between scientists,
- enable interoperation of software dealing with the data,
- enable trend finding and outlier detection,
- enable exploration of methods and process parameters and
- have transparent and interactive sample selection.

An information system with persistent data with data lineage can meet most of the listed requirements. It is useful to divide working with data into four parts:

- Storage of Data Products (section 1.3.1)
- Bookkeeping and Data Lineage (1.3.2)

- Processing (1.3.3)
- Visualization and Analysis (1.3.4)

The advent of large data volumes has set new requirements on interoperation between these components, since the separation between them is more profound. This separation is not only conceptual: due to the growth of data these parts are often physically disconnected as well. Bulk data is stored on redundant data servers and processed on computing clusters and databases are used to store catalog data and information about the data. The actual visualization and analysis will be performed with the client machine of the scientist.

1.3.1 Astronomical Data Products

Storing large astronomical data sets is not trivial. To achieve scalability and facilitate sharing, the data storage is often separated from the processing and visualization. This allows the system responsible for data storage to be designed for scalability and reliability.

To benefit from this scalability, abstract interfaces are implemented in order to quickly find and disseminate necessary data. In modern systems such as the Virtual Observatory and Astro-WISE, it has become less important for the end user to know about the details of how the data is stored. The data access is abstracted and operations on the data can be performed without knowing exactly how the data is handled internally.

Astronomical Data: Image Data

Images of the sky are perhaps the most important astronomical data products. Although the research in this thesis is focused on catalog data, images are still important: most of the catalog data we use is ultimately derived from images and astronomical research often contains a phase in which a scientist wants to see images of the studied objects. There are several ways in which image data is commonly stored:

- **FITS files**: The de facto standard format for astronomical images is the Flexible Image Transfer System (FITS)[1]. Almost every application that processes or displays astronomical images can handle FITS images.

  Besides pixel data, image FITS files usually contain essential information such as the celestial coordinate used (Calabretta and Greisen, 2002), time of observation (if applicable) and other information in the headers. There is no theoretical limit on the size of FITS images; image viewers such as Aladin[2] can handle images up to 50000 by 50000 pixels. FITS files can also contain catalog data at the same time as image data.

[1] http://fits.gsfc.nasa.gov/
[2] http://aladin.u-strasbg.fr/

- **Data Server**: Large volumes of image data are often stored on data servers which can deliver (subsets of) images on request. The exchange of images is usually done in the form of FITS files. Examples of such services are the SDSS CAS[3], the Astro-WISE image server[4] and any of the Virtual Observatory repositories that support the Simple Image Access Protocol[5].

  It is often more efficient to 'bring the processing to the data' than to transmit large volumes of data. The result of this is that scientists store less data locally, and more on central servers. Another benefit of a data server is that it is easier to share data if it is centrally located. Information about the image on the data server can be stored separately in a database. For example, in Astro-WISE everything beyond pixels is stored in the database (section 1.4).

- **Database** In a sense, pixel data is tabular data and can be stored in specialized databases and indexed. This enables operations that require images, such as source finding, to be formulated in a declarative instead of imperative way.

  An example of this approach is FastBit[6], although this is primarily focused on physical simulations instead of astronomical images. A pixel database combined with query-based visualization, such as DEX[7], can create an environment for fast exploration of image data.

Astronomical Data: Catalog Data

Catalogs of sources are the second most important type of data in astronomy. Usually this tabular data is derived directly or indirectly from image data. There are several mechanisms in use to facilitate the storage, retrieval, selection and sharing of catalog data:

- **FITS tables**: *The* standard for sharing astronomical catalog data is the FITS binary table. FITS tables (as default) store data in binary format, either column or row oriented.

  The binary approach saves space, preserves precision and shortens load time compared with ASCII formats. Accessing subsets of the data is fast since software requiring a specific cell can calculate exactly where the cell is located within the file.

  Besides raw data, FITS files can also store information about the columns, and the table itself in its headers. FITS files can store multiple tables and also images and spectra at the same time.

  With the advent of the Virtual Observatory, there is a FITS table variant called 'FITS-plus'. This uses the image part of the FITS file to store a VOTable header, so the FITS file can be used as a VOTable file as well.

[3] `http://cas.sdss.org/dr7/en/tools/chart/chart.asp`
[4] `http://imageview.astro-wise.org/`
[5] `http://www.ivoa.net/Documents/SIA`
[6] `https://sdm.lbl.gov/fastbit/`
[7] `http://vis.lbl.gov/Research/Dex/`

When dealing with large data sets, size and speed are important. If data is required to be shared or stored through files, FITS files are usually used.

- **VOTables**: Another emerging standard in astronomy is the VOTable, an XML based table file[8]. A VOTable is more flexible than a FITS file: columns can be addressed semantically through Unified Content Descriptors (UCDs), VOTables are more suitable for streaming, etc. More and more astronomical applications (such as Topcat and VisIVO) are able to read VOTable files but support is not as widespread as it is for FITS files.

  The data in a VOTable can be stored in several formats such as in ASCII or as a FITS file. This FITS file can either be inline (embedded in the VOTable file), or external (referenced from the VOTable file). For large data sets, the binary formats are preferred.

  VOTables are the standard for interoperation within Virtual Observatory, e.g. through the Simple Application Messaging Protocol (SAMP) (section 5.2). When interoperating software allow VOTables to be used, they should be preferred over FITS files because they are more flexible.

- **Database**:

  Large catalogs are usually stored in databases. These maintain statistics about the data that makes it faster to find and access required data. Examples of astronomical databases are the SDSS CAS (Catalog Archive Server), the AstroWISE database and other Virtual Observatory repositories.

  A standard language to access databases is SQL (Structured Query Language), which is designed for managing data in relational database management systems. SQL is a declarative language, that is, it expresses the logic of a computation without describing its control flow. The structure of SQL was originally designed by Codd (1970) and has been standardized by ANSI and ISO.

Astronomical Data: Other Data Products

In this thesis we limit ourselves to images and catalogs. The same scalability problems arise and similar solutions are devised for other types of astronomical data products (spectra, simulations). An example of this is the SQL based access of the Millennium Simulation data (Springel et al., 2005).

1.3.2 Bookkeeping and Data Lineage

One of the important contributions of modern information systems such as AstroWISE is the increased level of bookkeeping they perform. Data is stored as *persistent objects* with *full data lineage*, i.e. with all the information required to process them. This allows data to be *pulled* (section 1.3.3), which allows implicit data sharing.

[8] `http://www.ivoa.net/Documents/VOTable/`

Bookkeeping: Persistent Data Objects

Data that is stored persistently is kept over sessions and can be accessed from different workstations or even by different users. An object oriented approach to this, is to see every science product as an instantiation of a class in the computer science reading of the term. Important properties of such an object are stored persistently in a database, and an existing object is found by querying the database on these persistent properties. The bulk data of such a science product can be stored on a dataserver (e.g. images) or in a database as well (e.g. catalogs).

Bookkeeping: Data Lineage

A persistent object has data lineage if its stored properties contain all the information required to create the data product it represents. In this thesis we use the term *Process Target* to refer to these objects and the data of such an object is derived from other objects by *processing* it. Processing an object requires three pieces of information:

- **code**: The software required to create the data. This is part of the class definition of the Process Target.
- **progenitors**: Other Process Targets from which the data should be derived.
- **process parameters**: Parameters that are used to influence the creation of the data.

The data lineage creates links between all science products, from the final result all the way back to the raw data, this is called *backward chaining*. With perfect data lineage, it is possible to create the data of an object at any time by processing the object. Therefore, only the raw data and the data lineage *has* to be stored, and in some circumstances it can be unwanted to store processing results. The availability of the data lineage makes it possible to only create data that is actually requested, leading to the *pulling data* paradigm (section 1.3.3).

Bookkeeping: Sharing

Persistent data objects can be used in different sessions, even by different users. Full data lineage makes it easier to share data with collaborators: prospective users can determine whether the data suits their scientific needs by inspecting its derivation. Furthermore, the data lineage allows the objects to be found through data pulling, sharing data implicitly (section 1.3.3).

If data is shared with other users, it is important that once the object is stored, the data it represents should not change anymore. If another user than the creator of an object is relying on the data it contains, the original creator should not be able to change this data because that could jeopardize the use by the other scientist.

1.3.3 Processing

Raw astronomical data needs to be processed before being useful for scientific analysis. From a raw image to a science-ready image can take dozens of steps and requires intermediate data objects such as regridded and coadded frames. Source catalogs with mostly photometric parameters are created after the images are processed. These catalogs are subsequently used in the calculation of derived parameters. Processing can usually be parallelized and can therefore be performed on distributed computing clusters for large data sets.

Processing: Pulling Data and Reprocessing

In astronomy, data is traditionally pushed towards a data release. That is, all the data from a specific period is reduced with one version of the software and is all made available at the same time. This is impractical because there is no 'one size fits all' approach to data reduction. Researchers with different scientific goals have different requirements of the data.

The *pulling data* paradigm turns this around. Only data that is actually required is reduced, with settings specified by the scientist requesting the data. This ensures that processing power is only spent on data that is actually needed, that optimal versions of the code and parameters are used, and allows concurrent versions of the same data to be used side by side, for different scientific goals. Furthermore, sharing of data is implicit because the information system will reuse existing data products if possible.

Full data lineage makes reprocessing of data easy because every bit of data that a persistent object represents can be recreated. Recreating data is useful if better raw data is available or when a scientist wants to use different processing parameters.

Processing: Image Processing

Data servers of push-based information systems usually allow access to the raw and reduced data, but not always to intermediate data. It happens often that a scientist feels the need to redo some of the processing, e.g. La Barbera et al. (2008) redo the entire photometry for their SDSS and UKIDSS samples to be confident that the photometry is done in a comparable way.

When reprocessing images, it is often not necessary to reprocess the entire image. When selecting a couple of thousands of sources from a catalog the size of SDSS, most frames only contain one source or none at all. Reprocessing the data on a subimage level can increase the processing speed by orders of magnitude (Mwebaze et al., 2010).

Processing: Catalog Processing

Source catalogs with photometric parameters are derived directly from images. These are further used to calculate new parameters and to compose samples. This can be just as involved as image processing, and therefore needs the same infrastructure of persistent objects and full data lineage.

Derived parameters, such as galaxy mass and the quantification of the environment of galaxies, can often be calculated with several different methods (section 1.2.3) and it is insightful to be able to compare those. Full data lineage requires only one scientist to calculate and store a parameter with a particular method. This parameter can then be used by colleagues without having to redo the calculations or create duplications of data. Data pulling mechanisms can be used to achieve scalability: The calculations can be defined on the largest sample they are applicable to, but only the subsets that are actually requested have to be processed.

Astronomical catalogs range from surveys with billions of sources to scatter plots with a few parameters of a handful of objects. Sample selection happens on all levels in between and the mechanism for selecting data should scale between these two extremes. Furthermore, it is important for large catalogs to minimize the amount of data stored. Many operations on catalogs can be performed on the fly, such as most sample selections and simple parameter calculations, as long as the data lineage unambiguously defines how to do so. The results of such operations only need to be stored if this is required for performance.

1.3.4 Analysis and Visualization

Large data volumes make it difficult to explore data interactively. A close interaction between the visualization, data server, database and computation facility can reduce these problems.

Analysis and Visualization: Backward Chaining

Astronomers have a close relation with their data, and often want to 'go back' to the raw data or to intermediate data products. An information system with data lineage makes this easy, even from within visualization software.

One reason for this desire when visualizing source catalogs, is that each data point represents a property of a physical object. When an interesting object is found through visualization, such as an outlier, then the scientist will want to know exactly which source this was and how its parameters were calculated. Another reason to go back to the images is to study noise properties.

To some extent this differentiates astronomy from other sciences. When studying climate, weather stations are distributed in a way that allows statistical conclusions to be drawn from the data. This forces a large amount of randomness in their locations which ensures that collected (and visualized) data does not correspond to specific physical objects of any particular interest. With sociology and medical studies, data lineage is explicitly broken with anonymization, because it is undesired to be able to connect an individual to a particular data point. In psychology, test scores often have no intrinsic meaning: they have only relative value, unlike, say, the stellar mass of a galaxy.

That said, also in astronomy there are cases where an individual data point on its own is meaningless, such as in large scale structure simulations or studies of the cosmic microwave background.

Analysis and Visualization: Trends, Clusters and Outliers

Enormous catalogs cannot be visualized as sets of individual points: it is not possible to create a scatter plot with a million sources. However, other techniques can be used, such as contour plots, which are more suitable to visualize relations between groups of objects than to show information about individual sources.

Interactive or semi-automatic visualization can be used to detect trends or clusterings in the data. When coupled to a processing system, this can be scaled to millions of sources and dozens of parameters (Ferdosi et al., 2010).

Furthermore, visualization can also be used to study the outliers of trends and clusterings. These can be scientifically more interesting than the trends, and might require large data volumes to find (section 1.2.2).

Analysis and Visualization: Data Selection

The composition of a sample is an important factor in scientific analysis. Proper understanding of the selection effects in the sample is essential to draw conclusions. In an information system with full data lineage, any selection criterion explicitly imposed by a scientist is stored. Therefore, it is known at all times which selection criteria were used for every data set.

Sample selections can be performed at several stages during research. It is not trivial to determine the best place in the pipeline to do this: The earlier the less unnecessary data is carried around, the later the less potentially valuable data is thrown away. The benefit of both can be achieved by storing the selection criteria as data lineage. The selection can be specified and stored late in the pipeline, but moved backwards during processing.

The selection criteria can be exported to analysis or visualization programs which in turn can be used to modify them or to specify new criteria. This makes it possible to explore different sample selections directly from within the visualization, without requiring the visualization to have specialized knowledge of the way the selections are performed.

It is useful to specify and store such a selection criterion in a conceptual way. That is, not directly tied to the particular implementation that underlies the selection mechanism. This allows the information system to determine where to perform the selection, e.g. within a database, or on a local machine, etc.

Analysis and Visualization: Method Exploration

Properties of astronomical sources, e.g. of galaxies, can be differentiated into parameters that are derived directly from the observations, such as photometries, and parameters that are derived indirectly, such as mass. The focus in this thesis is on the latter, which can often be calculated with various methods.

A closer interoperation between the visualization and the processing of the data can help with exploring the effects of different methods of calculating derived parameters. Full data lineage allows the design of abstraction layers that make it possible

for the user of the visualization package to determine how the calculations are done, and even manipulate these calculations, without leaving the visualization software.

The processing of the data has to be designed to allow easy access and modification from the visualization package. It cannot be expected that the visualization software knows how to perform any of the processing steps. This can be achieved in a data pulling environment with full data lineage, see section (1.5.2).

Analysis and Visualization: Query Driven Visualization

Query driven visualization is a methodology to explore large data sets by limiting processing and visualization to the subsets of the data deemed "interesting" as defined by the user (Stockinger et al., 2006). Data pulling is an excellent opportunity for query driven visualization: data is not pushed from the information system to the visualization, but pulled by the visualization from the information system. Query driven visualization combined with data lineage offers a solution to the need of exploring selection effects and method exploration in large scale multi-dimensional datasets.

In non-query driven systems, the scientist first has to select data from the data storage, save it, and load it into the visualization program. Besides this being inefficient manual labor, there are conceptual drawbacks to this as well. Once the data leaves the information system, there is no data lineage any more, so it is difficult to go from the visualization back to the original data.

Furthermore, large data sets should be stored in a database which is specialized in searching and selecting data. A closer interaction between the visualization and the information system allows the visualization tool to benefit from the expertise that the database has.

An information system with persistent data could also store what data is visualized, effectively creating persistent visualizations that can be shared.

1.3.5 Interoperability

Interaction between different software components plays a central role in information systems handling large astronomical data sets. Large data sets require software handling the data to be segregated, often even physically. For example, the visualization and interaction of the data will occur on the display the scientist is working with, while the processing will have been performed on a dedicated computation facility. To make up for the boundaries that arise by this separation, a close interoperation between the different pieces of software is required.

Large monolithic programs that try do everything are passé. The future is a more UNIX like approach: a large set of small tools that each perform a small task, but do a very good job at it. To be convenient to use, the different tools will need to be able to communicate, interact and share with each other. The term 'tool' should be interpreted broadly, a plugin to a specific software package would classify as well.

Data pulling mechanisms can make this interoperation very powerful, without requiring the different tools to have intimate knowledge about how the other components work internally. We discuss some important aspects of interoperation, with the

focus on scalability and visualization.

Interoperation: Exchange of Data

The most common interoperation between programs is to share data between them. Traditionally the user of the software has to save the data in one program and load it in another. Besides that such a manual process only works well for small data sets, it puts limits on the information about the data that is shared between applications. For example, without data lineage it is not implicitly clear what the data represents and how it is created, the scientist should keep track of this manually.

A more flexible approach is to add automation in sending data from one piece of software to another (and back, if required). The benefit of this is that the programs themselves can do all the bookkeeping, so it becomes easier for the user to exchange data. As an example, the Simple Application Messaging Protocol (SAMP) has several messages to send data between SAMP capable clients (section 5.2). This approach still has limitations, mainly because it is a pushing approach. Every step in the flow of the data has to be performed in order by interaction of the scientist and this requires the user to have access to all relevant systems.

This is turned around by the concept of data pulling. Instead of pushing the data from one program to another, the data is requested between the applications. The data lineage of a data object should be easily retrieved. There is a multitude of benefits in this approach.

The data should be requested at the last stage in the data flow, that is, where it will actually be used. By doing so, the collaboration of programs dealing with the data knows what the end result should be. The underlying information system can utilize this information to perform the required operations in an optimal fashion. With a 'data pushing' approach, this is not possible.

Perhaps surprisingly, sharing data by pushing becomes easier in a data pulling environment. Instead of sending the entire data set, only the information about what data should be pulled, has to be sent. The receiving end can then request the data itself. With persistent data objects, any selection of data can be stored with a unique identifier, sharing data becomes sharing this identifier.

Interoperation: Exchange of Actions

The second most important reason for interoperation between software is to influence each other's behavior. There are two directions for programs to interact for this reason.

A program can ask another program to continue working with the data and perform a certain procedure. This program in turn can ask the next program in line to perform an action, etc., creating a pipeline that pushes not only data forward, but also processing.

The other direction is a pulling approach where a program influences the software before it. The traditional way of implementing this is through feedback loops that require programs to have intimate knowledge about how to manipulate the behavior

of the other software. Full data lineage offers unique possibilities for more powerful mechanisms: required changes in the processing can implicitly be communicated through data pulling.

For example, selection criteria should be stored as the data lineage. Such a criterion can be changed from within the visualization software by pulling a new sample. Due to the data lineage, this change can automatically be communicated back to the software that performed the selection. This communication can even be performed without the scientist or the visualization application having to know which software is actually performing the selection.

Interoperation: Communication Method and Abstraction

To allow a wide range of applications to collaborate with each other, it is important that the used communication protocol abstracts away the internal aspects of the programs involved. The more abstraction is included in the communication, the easier it is to add new tools to an existing software suite or swap one component for another, but the more difficult it is to make the interoperation thorough.

A low level of abstraction can be seen in programs that connect directly through hooks in each other's code. An example is the interaction between the software environment for statistical computing *R* and the visualization software *GGobi*. R has a plugin called *rggobi* which allows an R user to visualize data with GGobi, and vice-versa, a GGobi user can retrieve statistical information from the visualized data. The benefit of such an approach is that the interaction can be very thorough, the programs usually have direct access to each other's memory.

A more general way to interoperate is through a standard protocol. An example is databases, where the client and server agree on a language such as SQL to request data. The International Virtual Observatory Alliance created a variant of SQL called ADQL (Astronomical Data Query Language)[9]. This allows software (such as the VO Desktop and Topcat) to access data on remote registries.

A standard protocol itself is not enough. The example of SQL can be used to show that such a protocol often only abstracts the inner workings of the programs to limited extend. While SQL abstracts how the database should go about retrieving any selected data, it requires the query formulation to be very explicit in where the data has to come from.

An information system with full data lineage and a pulling data environment allows for a greater level of abstraction, while retaining close integration. With data pulling, the discovery of existing data and the creation of new data is already automated to a large extent. Therefore, data can be requested even if it does not yet exists.

1.4 The Astro-WISE Environment

Astro-WISE is an is an environment that is designed with the properties we need to tackle the problems that arise when handling large scale multi-dimensional data sets.

[9] `http://www.ivoa.net/Documents/cover/ADQL-20081030.html`

Astro-WISE is created by the Astro-WISE Consortium, coordinated by OmegaCEN-NOVA. OmegaCEN is an astronomical data center for storage and a data processing system for OmegaCAM optical wide field imaging surveys (such as KIDS). In recent years functionality has been added which made it a universal tool for astronomical data processing, archiving and scientific analysis. It is designed to accommodate raw data for thousands of nights of observations with more than a petabyte of data storage. It utilizes federated databases and data servers, and parallel compute clusters to manage these vast amounts of data.

1.4.1 Astro-WISE nodes

Astro-WISE is a federated system. Nodes are operational in the Netherlands (Groningen and Leiden), Germany (Bonn and Munich), and Italy (Naples). Data created in one location can be used in any of the others. An Astro-WISE node can be structured in 5 interconnected components:

Astro-WISE Component: Data Server

Pixel data is stored as FITS images on the data server. The data servers of the separate nodes are interconnected. A user requesting a file that is not stored on his own data server will get the file from one of the others.

Astro-WISE Component: Database

Everything beyond pixel data is stored in the database. For example, the database contains the persistent properties of all objects in the system. A typical session of Astro-WISE will start with querying the database for requested objects. All Astro-WISE nodes synchronize their databases.

Astro-WISE Component: `awe`-prompt

The `awe`-prompt is the main end user interface of Astro-WISE. It is a Python prompt that is modified to allow easy querying of database objects, ingesting and manipulating data, running tasks on the distributed processing unit, performing scientific analysis, etc.

Astro-WISE Component: Distributed Processing Unit

The Distributed Processing Unit (DPU) is the linux cluster that allows parallel processing of data.

Astro-WISE Component: Web Services

Besides the `awe`-prompt, there are various other data disclosure and processing mechanisms, such as the web based *dbviewer* and *Target Processor* or the Virtual Observatory registry.

1.4.2 Astro-WISE key features

There are several key aspects of Astro-WISE that make it especially useful for handling and visualizing large multi-dimensional data sets. The important ones are summarized.

Feature: Persistent Database Objects

In Astro-WISE all information is stored as a persistent database objects (section 1.3.2). Objects representing astronomical data products are called *Process Targets*. This object oriented approach allows the classes that correspond to the persistent objects to inherit from other classes, e.g. for images there is a BaseFrame class from which more complicated image classes such as the ReducedScienceFrame are derived.

The use of persistent objects allows the use of Astro-WISE on any workstation. In principle no files have to be stored locally on the workstation, everything is kept on the server. During processing, e.g. when running SExtractor, the image has to be retrieved to the machine performing the processing. This can be the user's workstation, but often this will be the distributed computing unit.

Properties of objects that are stored in the database are called *persistent properties*. The one property that every database object has is the `object_id` which is a machine-readable identifier of the object. The main use of the `object_id` is for other objects to denote their dependencies. Subclasses will define their own persistent properties in addition to the ones of their parents.

Most objects can be disclosed through various means, e.g through the `awe`-prompt or through the web services (dbviewer and imageserver). The web services are useful for sharing data between programs: To share an image or catalog, only an URL pointing to these services has to be transmitted between the applications.

Feature: Reprocessing

A Process Target can be processed at any time, because Astro-WISE adheres to the principle of full data lineage (1.3.2). This will create a new object, which will exist side by side with the old object. This is required because other data objects, perhaps created by other scientists, might depend on the old object. We list some reasons to reprocess data:

- The data that has been used to create the object has been superseded by better data. This is usually noticed when one of the dependencies of the object has been invalidated.
- The code to create the object has been improved. This does not happen often.
- The user wants to change the process parameters, e.g. modifying an aperture size when measuring the photometry of sources. When exploring data, the scientist might discover that he or she performed steps in a sub-optimal way and might want to redo them.

Feature: Target Processing or Data Pulling

Data is selected and processed in Astro-WISE through data pulling: scientists specify their required data set in a declarative way and the information system will determine what other Process Targets are necessary to deliver the required data. That is, a set of chained Process Targets is created, with as end node a Process Target that represents the requested data. The information system will reuse existing Process Targets if possible and will instantiate new Process Targets if necessary. This process is called *Target Processing* and is the heart of Astro-WISE.

Feature: `awe`-prompt and 5 Line Scripts

The `awe`-prompt is a Python version modified to use persistent objects and it is the main user interface of Astro-WISE. All database objects correspond to a Python class in the `awe`-prompt. Most other software to interface with Astro-WISE, such as the Target Processor and SAMP interaction, are built in Python as well.

Using the `awe`-prompt, a user can query for database objects using the persistent properties. Found objects can then be used as instantiated Python classes. A leading idea in the `awe`-prompt is that common operations should be possible with scripts of only 5 lines of code.

Feature: Data Sharing

Full data lineage improves the usefulness of sharing data, which is encouraged in Astro-WISE. By inspecting the data lineage of existing data objects it is possible for scientists to check whether data they require is already available and meets their demands, or whether they should (re)create it themselves.

The `privileges` property of data objects determines which Astro-WISE users have access to the data object. The default mode of Astro-WISE is to share newly created objects with all other users. Each data object in Astro-WISE has one of five privileges as listed in Table 1.2.

`privileges` level	data is shared with
1: MYDB	only the creator
2: PROJECT	every member of the project to which the data object belongs
3: ASTRO-WISE	all Astro-WISE users
4: WORLD	Astro-WISE users and persons without an Astro-WISE account
5: VO	data is also available through the Virtual Observatory

Table 1.2: Table describing the different privilege levels in Astro-WISE.

Although there are ways to remove data that is not used, this is discouraged. Instead of deleting data, data objects can be flagged to indicate that they should not

be used anymore, e.g. for the creation of future objects. The most common case when this is done, is when a better version of the object has been created. In such a case, any data object that depends on the outdated data can be recreated automatically using the latest version of the dependency.

Feature: Scalability

The Astro-WISE database and file system are designed to hold very large data sets. The data server is designed to hold over 10 petabytes of data such as images and the database is able to handle about a petabyte.

1.5 Summary and Prospect

Many astronomical questions, e.g. about the evolution of galaxies, require large multi-dimensional catalogs to be answered. Astronomical catalogs are growing to sizes of billions of sources with hundreds of parameters and new methods are required to explore and analyze these data sets.

For scalability, the data handling should be split up in four parts: data storage, handling of data lineage, data processing and visualization. There are unprecedented possibilities for interoperation between these components by emphasizing the role of data lineage and data pulling. For storage and reduction of image data, this well developed and successfully in use. Data lineage and data pulling for catalog data, the analysis of sources and parameters, is less well established.

Extending data lineage to catalog data opens up possibilities for data pulling in the analysis domain, such as through query driven visualization. Furthermore, with persistent objects, data lineage effectively forces a well structured organization of the data and how it is created. Therefore the creation of data can be abstracted to allow intensive interoperation between different pieces of software.

1.5.1 Data Pulling in the Analysis Domain

Data pulling is well developed for image data, for example in Astro-WISE. In chapter 2 we explore how data lineage and data pulling can be extended to the analysis domain. Our focus is on catalog data, that is tabular data where each row corresponds to a source and each column to physical property of the sources. One of our goals is to allow exploration of calculation methods and selection criteria through data pulling.

The challenge is to ensure scalability all the way from surveys with billions of sources and hundreds of parameters to individual scatter plots while at the same time enable flexible exploration of calculations and selections, for example through query driven visualization. These goals can be achieved, because with full data lineage it is more important to store how to derive the data, than to store the processing results itself. Therefore, we realize that not all data products that have been defined, also have to be stored, or even created. This is useful when the data can be calculated ‘on the fly’ or when it is never requested in its entirety.

The operations to create new catalogs from existing objects should be predefined, because this allows the information system to assess several aspects of the catalogs without actually having to process them. These operations should be designed as elementary as possible to maximize the benefits of the data lineage. For example we developed a mathematical model to determine the logical relationship between selections of sources without actually having to know the exact composition of these selections.

A major benefit of this approach is that complex and expensive operations can be defined on a very large dataset without computational drawbacks, even if in the end only a small subset of the data is requested. This allows for better sharing and reuse of data, preventing duplication of data storage and calculations. Furthermore, it paves the way for powerful query driven visualization mechanisms.

Astro-WISE is one of the first astronomical information systems that strives to have complete data lineage. While our research is primarily targeted at Astro-WISE (chapter 3), the results should be applicable for information systems in general.

1.5.2 Query Driven Visualization with Data Lineage

Query driven visualization (QDV) can be seen as the natural continuation of data pulling, because it limits the creation of data products to the data sets required for the visualization. In chapter 5 we explore the possibilities that arise when performing QDV with an information system with data pulling and we demonstrate how this leads to increased interoperation with a high level of abstraction.

Data pulling allows existing data to be found and new data to be created from within the visualization software, without requiring that the visualization package has explicit knowledge about the inner workings of the information system. The main idea is that the requesting of data becomes entirely declarative. This does not mean that the user has no control over how the data is created (such as how process parameters are set), quite the contrary. With a proper abstraction layer, the stored data lineage—which contain such details—can be exported to the visualization from where it can be modified. Of course the person requesting the data should have scientific understanding about the details of the calculations in order to be able to adapt it in a beneficial way.

Finally, the data lineage itself, and thus the process to create the requested data, can be visualized without requiring the visualization software to understand the actual processing steps themselves. Within this visualization, mechanisms to modify the processing steps can be included.

We research what properties an abstraction layer should have to achieve these goals and design new messages for the Simple Application Messaging Protocol (SAMP). The required server code is implemented in Astro-WISE and several prototype client applications are developed.

1.5.3 Comparison of Density Estimation Methods for Astronomical Datasets

The main driver of our technical work is being able to answer the questions about galaxy evolution posed in section 1.2. In particular we are interested in the relationships that galaxies have with their environment. There are different methods to estimate the environment of galaxies and it is important to understand the differences between these methods. Using an early prototype of our mechanisms, we study the effects of algorithm choice on detecting these relationships. We compare the k-th nearest neighbor, two kernel based methods and the Delaunay tessellation field estimator.

The performance of the estimators is determined with various error measures on particle and field densities of artificial and simulated data. The effects of the different estimators on quantifying relations, such as the fraction of red galaxies as function of environment and detecting the bimodality of galaxies, is studied with SDSS data.

1.5.4 Galaxy Evolution

We combine SDSS data and morphologies from Galaxy Zoo to study galaxies at the edges of clusters. The density of galaxies plays an important role in separating the populations of galaxies, especially the selection of sources that is used for the density defining population (DDP). We use our query driven visualization system with full data lineage and different density estimators to show that the edges of clusters can be defined as regions with a high density that are dominated by blue galaxies.

These regions are shown to contain a larger fraction of red spiral galaxies and blue elliptical galaxies than is expected from the morphology density relation. The structure of galaxies, quantified by their concentration index, is less affected by the composition of the environment. This indicates that any morphological transition in these regions is largely independent from the change in color and concentration. Furthermore we detect an opposite effect for red galaxies in low-density regions: these are more likely to be spirals in the presence of other red galaxies.

1.5.5 Thesis Layout

In chapter 2 we explore how data pulling can be extended to the analysis domain. An implementation of our mechanisms within the Astro-WISE framework is described in chapter 3. The Astro-WISE implementation is used to study the effect of density quantization methods on astronomical results in chapter 4. In chapter 5 we present query driven visualization as a continuation of data pulling. In chapter 6 our methods are used on SDSS and Galaxy Zoo data to study the evolution of galaxies.