Selection and analysis of user-defined subvolumes in radio-astronomy software

Calvin Brizzi
Department of Computer Science, UCT
calvin.brizzi@gmail.com

ABSTRACT

Currently available tools for radio astronomy, such as DS9 and Karma, have limited or non existent functionality for selecting three dimensional subvolumes of data cubes for analysis. Tools designed for other disciplines which use similarly structured data, such as for three dimensional medical imagery, implement various selection methods which give users vital insights into the data. However, to date astronomy tools have not adopted these features.

We introduce a software prototype that allows the user to rapidly define subvolumes, specifically Elliptical Cylinders, automatically calculate common statistics and export the data to a plaintext format for portability.

Expert testing demonstrated that the features would improve existing options, with Elliptical Cylinder selection in particular would provide the biggest gains, with automated analysis and plaintext export complementing it, while still being useful in their own right.

Implementing these features in current tools would reduce the amount of repetitive tedious work astronomers currently perform allowing them to be more productive and efficient.

CCS Concepts

•**Applied computing** → **Astronomy;** •**Human-centered computing** → *Human computer interaction (HCI);* Visualization;

Keywords

Astronomy Software, Astronomy Tools, 3D Selection

1. INTRODUCTION

HI Radio Astronomy data is encoded as a three dimensional array of floats with each point being the flux (amount of energy transferred from a point) at a given right ascension, declination (which define a location on the celestial sphere) and frequency.[11] Analysis of these cubes can give us information on the overall gas content, distance and internal velocity of remote galaxies [7], but contain large amounts of noise due to various sources of interference, such as the temperature of the receiver itself[16]. This noise makes it harder to identify galaxies and is challenging to filter out, hindering exploration of the data.

While methods for automatically detecting galaxies in data cubes have improved over the years, they are still very susceptible to low signal-to-noise ratios[17], forcing astronomers to resort to direct human inspection. Once a potential galaxy is discovered, much of the analysis that is done on the candidate could be automated, as it is for the most part simple algebra such as the computation of the mean, standard deviation and the generation of simple graphs.[6]

Most current tools such as DS9 and Karma limit selection to a 2D square (sometimes rectangle) and analysis to average flux in that square, or at best an overview of how the flux varies at a given point in the sky across various frequencies.[3][8] None consider the wide variety of shapes celestial objects come in, thus including large amounts of noise in the analysis, nor do they leverage the third dimension in the data.

Time and resource constraints convinced us to develop a throwaway prototype. The process for the development of an effective throwaway prototype has nine distinct steps: elicit initial requirements, model the requirements, identify constraints, prioritize initial requirements, design, evaluate design, specifications, interactive prototyping and requirement validation[1]. Due to limited opportunities to interact with domain experts, we had to modify the process slightly, as discussed in Methodologies below.

The prototype was evaluated by staff of the astronomy department at the University of Cape Town according to the "expert evaluation" paradigm, in which users familiar with the problem domain are asked to test and provide feedback.

The aim was to justify the implementation of similar methods in current tools by obtaining a qualitative estimate of its usefulness. Three experts were given a brief introduction to the package and were then instructed to complete a series of tasks designed to get each user to interact with every part of the prototype. Finally each subject answered a series of questions about what aspects of the prototype were most effective, what improvements could be implemented and how the elliptical selection compared to tools they used daily.

During implementation of the prototype we focused our attention on developing the desired features rather than per-

UCT '16 Cape Town, South Africa
© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

fecting the interface. By implementing faster and more accurate tools for astronomers we aim to enable more efficient and effective data analysis.

2. BACKGROUND

2.1 Astronomy Tools for Visualization

Below we present a brief description of some tools that are widely used with a focus on what they have to offer in terms of analysis of subregions.

2.1.1 CyberSKA Viewer

Kiddle et al.[9] developed CyberSKA as an online tool that provides the infrastructure and functionality to meet the needs of data-intensive endevours such as the Square Kilometer Array.
Users interact with data through an online portal to avoid local storage and processing of the data. While it does allow for collaboration and sharing of specific views, it it limited to 2D interactions and even then only rectangles can be selected.

2.1.2 CASA

The Common Astronomy Software Applications (CASA) package allows users to process radio astronomy data. It is built as a collection of C++ tools wrapped by a Python interface. This allows users to write scripts to perform custom tasks[10].
It runs on consumer hardware and is still actively developed. Its graphical interface only allows for selection and analysis of 2D rectangular regions.

2.1.3 SAOImage DS9

SAOImage DS9 is a powerful and feature rich data visualization application. It is still in development with new features being added often[8].
It has features for automatically detecting iso-value contours (closed lines where all point have similar values) in 2D but this does not extend into three dimensions.

2.1.4 Karma

While no longer maintained, Karma is still widely used and has some powerful features that enable users to overlay iso-value contours, examine the position-velocity diagram for a 2D slice through a 3D galaxy etc.[3]. While the contour overlay would be useful for accurate selection, Karma has neither tools for selecting 3D volumes nor analyzing user-defined areas.

2.2 Visualization Tools in other fields

Medical imaging tools often produces data formats very similar to those used in astronomy[4], where the data represents density in 3D space rather than energy at a certain frequency for a specific point in the sky, so we will look at some of the more widely used tools.

2.2.1 3D Slicer

3D Slicer is an open source program for visualization of medical data. It has a focus on usability for non-programmers[12].
While it has many user submitted modules none of them add analysis capabilities, as they focus on the creation and visualization of surfaces of equal values, useful to visualize different tissue within the data.
Work has been done on adapting it for use with astronomical data but still focuses on visualization rather than analysis[12].

2.2.2 CAVASS

CAVASS is newer than 3D Slicer and specifically designed to offer better analysis tools[5].
It has some very interesting analysis tools not present in other packages. For example, CAVASS can generate an iso-surface and then calculate the volume encosed by it.
While these tools would need to be adapted to astronomical needs, they are a good starting point.

2.3 Principles

Shneiderman describes seven tasks that, in a well-designed system, should be easy and intuitive for a user to complete: overview, zoom, filter, details-on-demand, relate, history, extract[14].
When working with radio astronomy data all these tasks are important but our aim is to develop a design that improves user experience in the filtering, details-on-demand and extraction tasks. In this paper we refer to the filtering task as selection (the user "filters out" the uninteresting area from the data cube), the details-on-demand task as analysis (the program calculates statistics relating to the selected area) and the extraction as plain-text export.

While the throwaway nature of the prototype meant that the focus was on implementing the features identified as important, we found Shneiderman's "Eight Golden Rules of Interface Design" [15] to be concise and useful. They are:

Consistency the design should be consistent both internally (similar tasks require similar actions) and with other software, so selection mechanisms in particular should rely on the actions users usually take to select items (dragging out a rectangle for example)

Shortcuts expert users should be able to complete tasks faster with well-defined shortcuts so we should include those in a discoverable way alongside our interface

Feedback the system give the user proportional feedback for their actions

Closure when a user completes a task, they should be aware of it, and be ready to begin the next task, we should have a way to save the analysis and clear the interface, ready for the next selection

Error handling the system should not allow the user to make an error. When this is unavoidable, errors should be easy to handle

Undo actions should be easily reversed

Control users should initiate actions rather than respond to system events

Reduce recall the interface should be designed so that the user can recognize how to complete a certain task from the interface itself, rather than remember how to complete a task or consult documentation

3. METHODOLOGY

3.1 Requirements Gathering

Due to limited availability of the domain experts, we combined the first six steps of the throwaway prototype process into a single brainstorming session.

Initially various issues with current software were raised (elicit initial requirements), through discussion about how these issues related to each other and what was achievable in the time given (model requirements and identify constraints). Three main pain points were highlighted (prioritize requirements): tools neglect to take into account the three-dimensionality of the data or when they do take it into account, they do not give flexibility in selection shape, users are forced to do many simple but tedious calculations manually and when they need to move information obtained in one tool to a different tool for further analysis, the information is not in a portable format. Finally, using the interface of existing tools as a starting point, a design that would feel familiar was agreed upon (design and evaluation) and the desired behaviors of the prototype were identified (specification).

During this process it emerged that, while galaxies tend to take on a variety of shapes in the sky, an ellipse can be used to approximate a large subset of those that appear in typical HI data cubes and so a cylindrical ellipse was chosen as the 3D selection volume to implement in the prototype.

3.2 Development approach

Attempts at extending existing software proved more time-consuming than helpful, so after a review of languages and libraries available, we identified Astropy, a modular Python library with many powerful built-in features for astronomy[13], and PyQt5, a cross-platform user interface framework, as the foundation for our prototype. Python enabled us to develop rapidly and gave us access to a large collection of libraries that we could rapidly combine to implement powerful features.

Python also facilitated an Agile approach to software development[2]: we defined sprints that focused on a single feature until we had a viable implementation that could be used and improved based on internal feedback.

The short cycles allowed for a sense of progress and achievement as each feature was implemented and gave flexibility to adapt as problems arose. Four main cycles were completed: initial setup and visualization, elliptical selection, channel range definition and finally analysis and export of data.

3.3 Prototype implementation

Many of the final features were provided by libraries already present in Python and it was simply a question of combining the libraries and providing the correct data to get powerful functionality rapidly. Keeping in mind Shneiderman's "Eight Golden Rules of Interface Design", features that were shared with other tools behaved consistently with those tools (scroll wheel to zoom, mouse drag to pan), we provided shortcuts to common action ('e' to switch to elliptical selection), ensured that the user always received feedback and closure when completing an action, gave the user complete control over the system by keeping it responsive at all times and attempted to make the interface intuitive in order to reduce recall. Python itself handles errors gracefully, with the system proving stable during user testing.

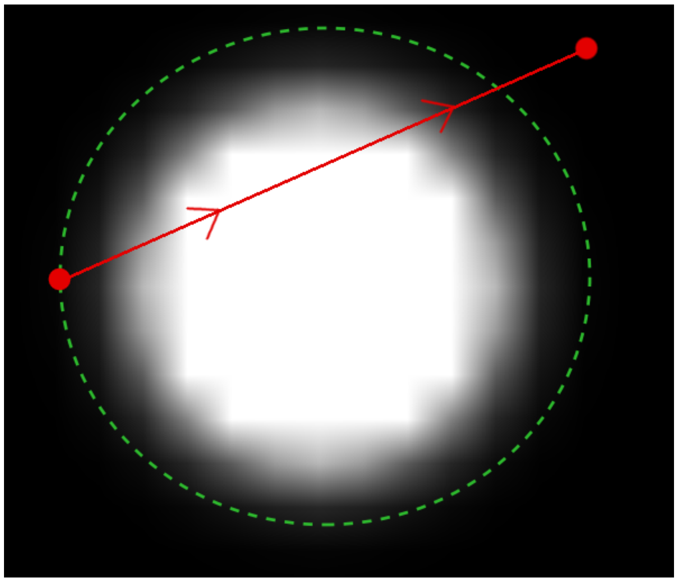


Figure 1: Demonstration of selection actions, the red overlay demonstrated the mouse movement: the leftmost dot is the initial mouse click location, the user drags the mouse along the red line and releases at second dot. The green ellipse is redrawn continuously as the user drags the mouse, showing the final selection.

3.3.1 Elliptical Cylinder selection

The implementation of the Elliptical Cylinder selection required more than simply combining existing libraries. First, an appropriate method of defining the desired ellipse was required. An initial implementation that mimicked the way users usually define rectangles was attempted: the user selects the top-leftmost point of an imaginary rectangle and then drags to the desired bottom right point. As the user defines this imaginary rectangle, an ellipse with axes of length equal to the sides of the rectangle inscribed in the rectangle is displayed on screen. This method was scrapped rapidly as it proved hard to select a desired region: it required the user to begin the selection at a point external to the desired selection and it was hard for the user to judge if their selection would align with the area of interest until they had completed the selection actions.
An alternative design was adopted: the user first uses their mouse to define an ellipse by clicking on the leftmost extreme of the desired major axis and dragging the mouse to the right a distance equal to the desired length of the major axis and up half the desired length of the minor axis. While this process may appear complicated on paper, it provides accurate and intuitive selection(Figure 1).
Once the selection is defined, the program generates a 2D array of booleans that has width and height equivalent to the axes of the ellipse and simple geometry is used to determine if each point on the array would be inside an ellipse inscribed in the array or not. This creates a mask that, once the range of channels is known, can be combined with a subvolume of the original data cube in order to calculate the relevant statistics and generate the desired graphs while excluding the noise around the elliptical cylinder (Figure 2).

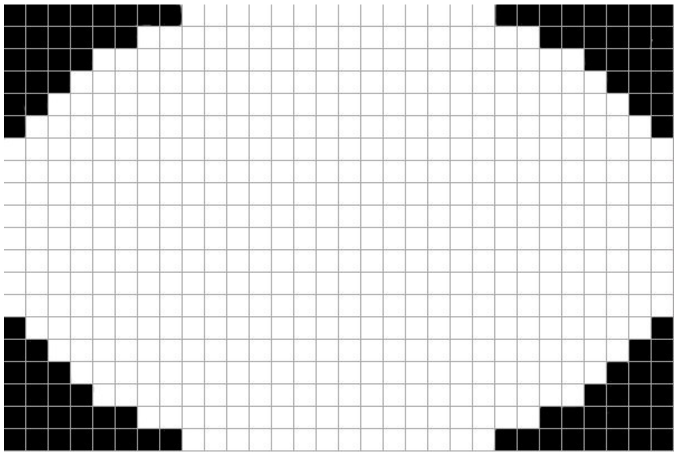


Figure 2: Example mask created during elliptical selection. The black squares map to values that are ignored during the calculation of statistics.

4. RESULTS AND DISCUSSION

4.1 System Overview

The interface has three main areas (Figure 3): a view of the data currently loaded, a set of controls to set the range of channels the user wants the Cylinder to span and finally the stats and graph. The main functions implemented are: Elliptical Cylinder selection, analysis of selected areas and export to plaintext. The prototype also featured a cuboid selection option, but it was not part of the testing.

4.1.1 Interface

The view of the data cube included all the main features of current tools. It allows the user to step through the cube, zoom on areas of interest and pan across the data. The intensity of the signal at each point is represented by a monotone gradient that, while not ideal from a visualization standpoint, was sufficient for our needs.

The channel controls, labeled "Z controls", were designed to give the user multiple options to set ranges: the start and end variables could be set by manually inserting values, stepping through those values with increment and decrement buttons or setting the values to match the slice in the current views with a single click.

The statistics shown were those identified during requirements gathering as potentially useful: minimum, maximum, mean, total and standard variation. Finally the graphs chosen was one that would allow users to rapidly identify specific properties of the galaxy: the shape of the total or mean intensity per slice graphs give domain experts valuable information about the rotational speed of the galaxy.

4.1.2 Features

Elliptical Cylinder selection is completed in two steps: defining the base and then the range. The ellipse, along with the range defined with the "Z controls" is used to extract a the smallest three dimensional array that contains the area of interest. A 2D mask is then created that marks elements outside of the ellipse as invalid, so that they are ignored when calculating statistics and generating graphs.

The graph generated can be saved at any time to a plain text file for use in other tools.

4.2 Expert Evaluation

The prototype was positively received by the domain experts: all subjects agreed that elliptical cylinder selection was superior to current options available, both in terms of reducing the amount of noise introduced in selection and the ability to gain insight into multiple channels of data at the same time.

4.2.1 Defining the Cylinder base

The definition of the elliptical cross-section of the cylinder differed from the techniques widely used in other graphical tools (which usually fit an ellipse inside a user defined rectangle) but the panel agreed that it allowed for accurate selection once the user was comfortable with the technique. One subject noted that a tutorial or clear explanation of the steps would be particularly useful for first time users.

Two of the experts praised the use of an ellipse as a base for the cylinder, confirming that when dealing with real life data galaxies and other volumes of interest can be approximated well, though one of them noted that letting users define rotated ellipses would be necessary to improve accuracy.

Finally two of the experts suggested implementing an alternative method for defining the ellipse by selecting the extremes of the major and minor axes as they felt this would further increase their accuracy, allow them to define rotated ellipses and solve some of the issues with how slow the current selection is (though the latter is more of an implementation issue).

4.2.2 Specifying the channel range

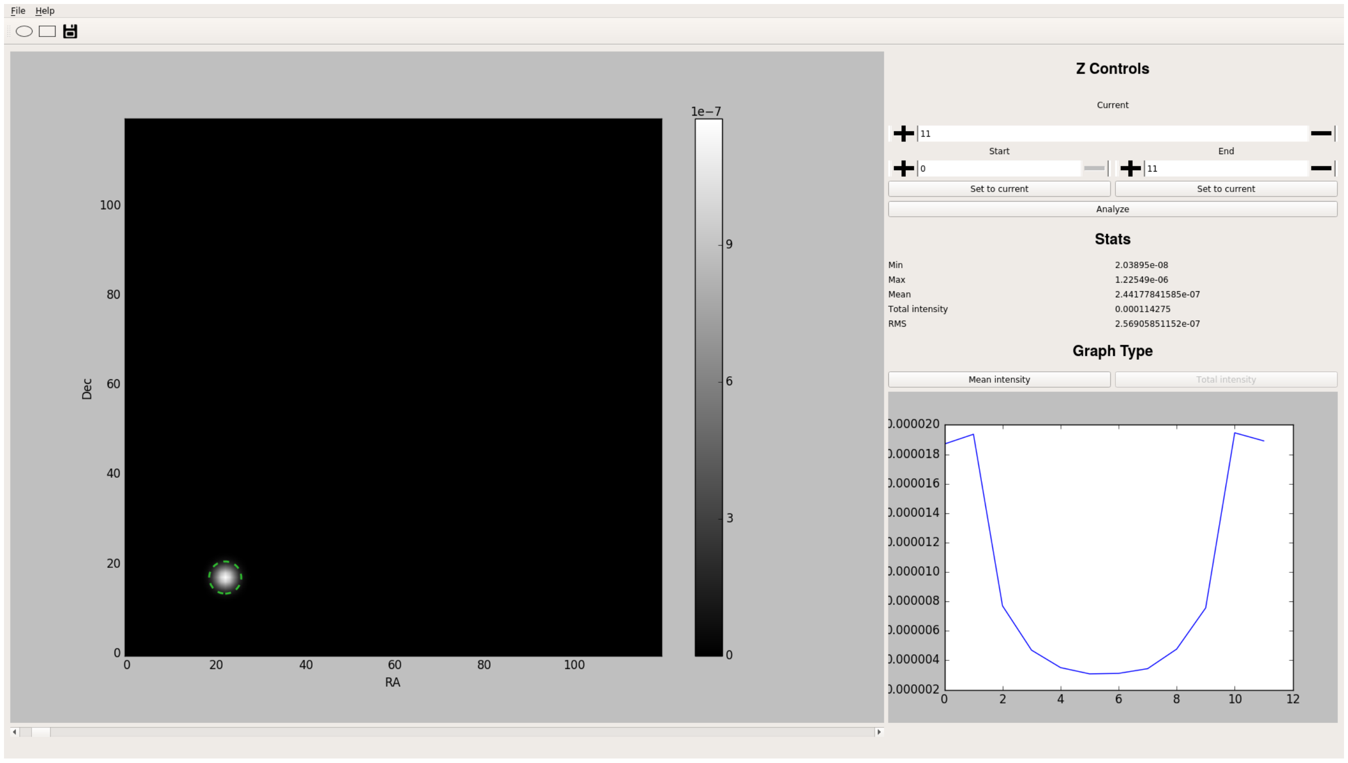
Users were excited by the ability to define a channel range over which to do analysis, mainly due to the lack of the option in current tools which prevents them from gaining useful insights into the data. One of the experts explained that they usually only performed analysis on three dimensional volumes once they were already relatively sure that the volume contained interesting information rather than while exploring the data.

They also believed that allowing the analysis during exploration would possibly enable users to discover sources that they might have missed with currently available tools and verify sources that were identified with automated systems.

The experts also found that having multiple ways of stepping through the channels in the cube (scrollbar, step buttons, manual input) and specifying the range of channels extremely useful and a feature they would like to see in other tools. They did note that the increment and decrement buttons were arranged in a non-intuitive manner, but this would be relatively easy to rectify. One of the experts particularly enjoyed the ability to set the start or end channel with a single click while exploring the data, noting that tools often required the user to copy over information from other parts of the interface or repeat a series of steps to set each variable.

4.2.3 Analysis of selected sub-volumes

Both the statistics and graphs were received positively by the panel: they commented that they often spend several minutes to extract even one of the results that the prototype was able to give them near-instantly. They further men-



Current: 11

Start: 0

End: 11

Stats

Min	2.03895e-08
Max	1.22549e-06
Mean	2.44177841585e-07
Total intensity	0.000114275
RMS	2.56905851152e-07

Figure 3: Overview of the final prototype, with the main view of the data on the left and controls for the range of channels, statistics and graph on the right.

tioned that none of the tools they were aware of was able to automatically generate graphs based on three dimensional volumes and the manual techniques they used would often introduce a lot of noise into the data due to limitations of defining areas of interest.

One of the experts mentioned that having statistics of the data cube as a whole would also be of interest to users, as often having an idea of how the numbers related to the global statistics is more useful than absolute values.

Two of the users also mentioned that other graphs would be valuable, specifying that a rotational velocity field graph of a galaxy would be extremely valuable and can be difficult to obtain with current tools while being relatively easy to implement.

The ability to export the graphs to plain text for analysis or recreation in other tools was particularly well received: the experts lamented that it was not an option currently available in other tools and that it would allow them to avoid extracting the data manually or repeating tasks in the future. Several improvements were suggested for this feature, though they were mostly implementation details: allowing the user to specify the destination to save to, allowing the user to specify what information to include and placing the save icon closer to the graph as its current location created ambiguity as to what is being saved.

4.2.4 *Overall impressions*

Overall the feedback was largely positive, with the panel agreeing that the tools presented in the prototype would be valuable to their research and would allow them to complete tasks with more ease than the current tools available.

When asked what part of the prototype they found ineffective, the experts mentioned some user experience issues, such as elliptical selection not being clearly explained to new users, and some interface issues, such as the "save" button being quite far from the data that was being written to file and the channels being specified by their position in the data cube rather than the actual frequency they represent.

Finally one of the experts noted that due to how various properties differ between data cubes (noise levels for example) the user needs to have more control over the colour scale of the visualization: being able to define both what colours to use and whether a linear or logarithmic scale would be more appropriate at the time, as this allows for quicker source discovery.

4.2.5 *Discussion*

The favourable reception of the prototype was mainly due to it being developed as a response to shortcomings in current tools raised during the requirements gathering from domain experts and the continued communication during the development cycle.

The throwaway prototype proved effective: with limited time and resources (about two weeks of a single developer's time and two hours of meetings with domain experts) we were able to collect valuable data on features that experts believe would improve current tools. While it is true that this approach can only really be implemented in situation where the new features do not depend heavily on existing features (to avoid wasted time reimplementing existing tools), we feel that it has many applications. Combined with the ever-increasing number of libraries that are available in Python, the throwaway prototype should be the first step in developing many new features.

5. CONCLUSIONS

The Elliptical Cylinder selection presented in this paper is accurate, fast and allows users to rapidly define subvolumes with high signal to noise ratios. The analysis performed by the prototype, while being relatively simple maths, gives experts rapid insight into the data and makes possibly error prone manual calculations unnecessary. The option to export information to plain text was also well received as users often have a pipeline of tools they process their data with and are frustrated by the lack of an easy way to move their data through the pipeline.

While each one of these features are effective independently, the combination of easy definition of low-noise volumes, rapid automated analysis of said volumes and easy export of the data generated are complimentary and would preempt a lot of the manual and error prone calculations that consume domain experts' time.

Having demonstrated the effectiveness of Elliptical Cylinder selection future work should analyze the strengths and weaknesses of other selection techniques, using for example Bézier curves or simply allowing the user to draw the area of interest free-hand in order to optimize accuracy and speed. The inclusion of other statistics and graphs would also be beneficial, with a velocity field graph being the most requested during user interviews.

Finally, implementing these features as modules for existing packages while keeping in mind the changes suggested during expert testing would provide massive value to users with relatively small amounts of effort, automating a lot of use-cases.

6. ACKNOWLEDGMENTS

- Supervisor
- People who gave us money
- People who helped from Astro
- the academy
- team mates

7. REFERENCES

[1] S. J. Andriole. Fast, cheap requirements prototype, or else! *IEEE Software*, 11(2):85–87, 1994.

[2] K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, et al. Manifesto for agile software development. 2001.

[3] R. Gooch. Space and the spaceball. In *Astronomical Data Analysis Software and Systems IV*, volume 77, pages 144–147, 1995.

[4] P. F. Gorder. Medical software has astronomers seeing stars. *Computing in Science and Engineering*, 10(4):4–9, 2008.

[5] G. Grevera, J. Udupa, D. Odhner, Y. Zhuge, A. Souza, T. Iwanaga, and S. Mishra. CAVASS: A Computer-Assisted Visualization and Analysis Software System. *Journal of Digital Imaging*, 20(1):101–118, 2007.

[6] A. H. Hassan, C. J. Fluke, D. G. Barnes, and V. A. Kilborn. Tera-scale astronomical data analysis and visualization. *Monthly Notices of the Royal Astronomical Society*, 429(3):2442–2455, 2013.

[7] M. P. Haynes, R. Giovanelli, T. Herter, N. P. Vogt, W. Freudling, M. Maia, J. Salzer, and G. Wegner. 21 cm h1 line spectra of galaxies in nearby clusters. *The Astronomical Journal*, 113:1197–1211, 1997.

[8] W. Joye. New features of saoimage ds9. In *Astronomical Data Analysis Software and Systems XV*, volume 351, pages 489–492, 2006.

[9] C. Kiddle, A. R. Taylor, J. Cordes, O. Eymere, V. Kaspi, D. Pigat, E. Rosolowsky, I. Stairs, and A. G. Willis. CyberSKA: An On-line Collaborative Portal for Data-intensive Radio Astronomy. In *Proceedings of the 2011 ACM Workshop on Gateway Computing Environments*, GCE '11, pages 65–72, New York, NY, USA, 2011. ACM.

[10] J. McMullin, B. Waters, D. Schiebel, W. Young, and K. Golap. CASA architecture and applications. volume 376, pages 127–130, 2007.

[11] J. D. Mink. Astronomical data formats: What we have and how we got here. *Astronomy and Computing*, 12:128–132, 2015.

[12] S. Pieper, M. Halle, and R. Kikinis. 3d Slicer. In *IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2004*, pages 632–635, 2004.

[13] T. P. Robitaille, E. J. Tollerud, P. Greenfield, M. Droettboom, E. Bray, T. Aldcroft, M. Davis, A. Ginsburg, A. M. Price-Whelan, W. E. Kerzendorf, et al. Astropy: A community python package for astronomy. *Astronomy & Astrophysics*, 558:A33–A42, 2013.

[14] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

[15] B. Shneiderman and C. Plaisant. Designing the user interface: Strategies for effective human-computer interaction. *ACM SIGBIO Newsletter*, 9(1):6, 1987.

[16] A. R. Thompson, J. M. Moran, and G. W. Swenson Jr. *Interferometry and synthesis in radio astronomy*. John Wiley & Sons, 2008.

[17] S. Westerlund, C. Harris, and T. Westmeier. Assessing the Accuracy of Radio Astronomy Source-Finding Algorithms. *Publications of the Astronomical Society of Australia*, 29(3):301–308, 2012.