Interoperability in data mining and visualization (and why was AstroGrid so hard?)

Clive Page
NeSC
2005 December 13

AstroGrid 5 years old next week

- The meeting at which the AstroGrid Project was kicked off took place in Edinburgh in December 2000 – almost exactly 5 years ago.
- So – what has been achieved in 5 years?
- Software released, finally, in 2005. Not much sign of widespread use yet.
- Of course AstroGrid is now a 6-year project and has only been funded for 4.5 years of that – so only around 3/4 complete.

Original Aims

- A working data grid for UK databases
- High throughput data mining facilities for interrogating those databases
- A uniform archive query and data mining software interface
- The ability to browse simultaneously multiple datasets
- A set of tools for integrated on-line analysis of extracted data
- A set of tools for on-line database analysis and exploration
- A facility for users to upload code to run their own algorithms on the data mining machines
- An exploration of techniques for open-ended resource discovery.

So – why the limited progress?

Two main reasons

- AstroGrid tried to tackle problems which were intrinsically hard – and which astronomers have not solved even for local datasets, let alone over the wide area network.

- Too much emphasis on The Grid, XML, and other trendy and bleeding edge stuff from computer science.

AstroGrid has tackled hard problems

- Outstanding problems include
  - How to define metadata of universal applicability
  - How to tackle the diversity of data formats
  - How to cross match source catalogues
  - How to store and manipulate sky footprint information
  - How to do data mining and visualisation

- We really don't know how to solve these even on a local machine, let alone over the wide-area network.

Metadata problem

- Can’t retrieve and combine data from remote systems without having standarised data descriptions. For tabular datasets this means:
  - Data type
    - Not much of a problem, even DBMS can do this.
  - Semantics - UCD (universal content descriptor)
    - UCDs were starting to be used, then UCD1 invented.
  - Physical units
    - No standard yet, except ad-hoc ones in some FITS communities
  - Whether/where error information is present
    - Almost no standards yet – but Starlink NDF solved this a decade earlier.
  - Handling of non-standard values (nulls, upper-limits, etc)
    - Very little uniformity yet, let alone standardisation.

Data Formats Problem (1)

- Astronomers really were fortunate to have an agreed format, FITS, which nearly all applications supported (the situation in most other branches of science is much worse).
- Then the VO projects invented VOTable – I suspect more because FITS was not an XML-based format than because of really could not do the job.
- VOTable has 3 forms – the most commonly used is around takes about 5 times as much space as a FITS file.
- A few applications support VOTable, but a very small proportion, compared to those which support FITS.
- Fortunately TOPCAT can convert between the two.

Data Formats Problem (2)

- But: hardly any non-astronomical applications understand FITS (do any understand VOTable?)
- If you want to ingest data into a DBMS, or use a statistics or visualisation package, lowest common format is CSV.
- CSV (character-separated value)
  - Not a standard at all, and very variable rules in practice.
    - E.g. do strings have to be enclosed in quotes?
  - No way of specifying data types
  - Column names, may be on line 1, or may not.
  - Physical units, UCDs, never supported.
  - All other metadata – no chance.

CSV in practice

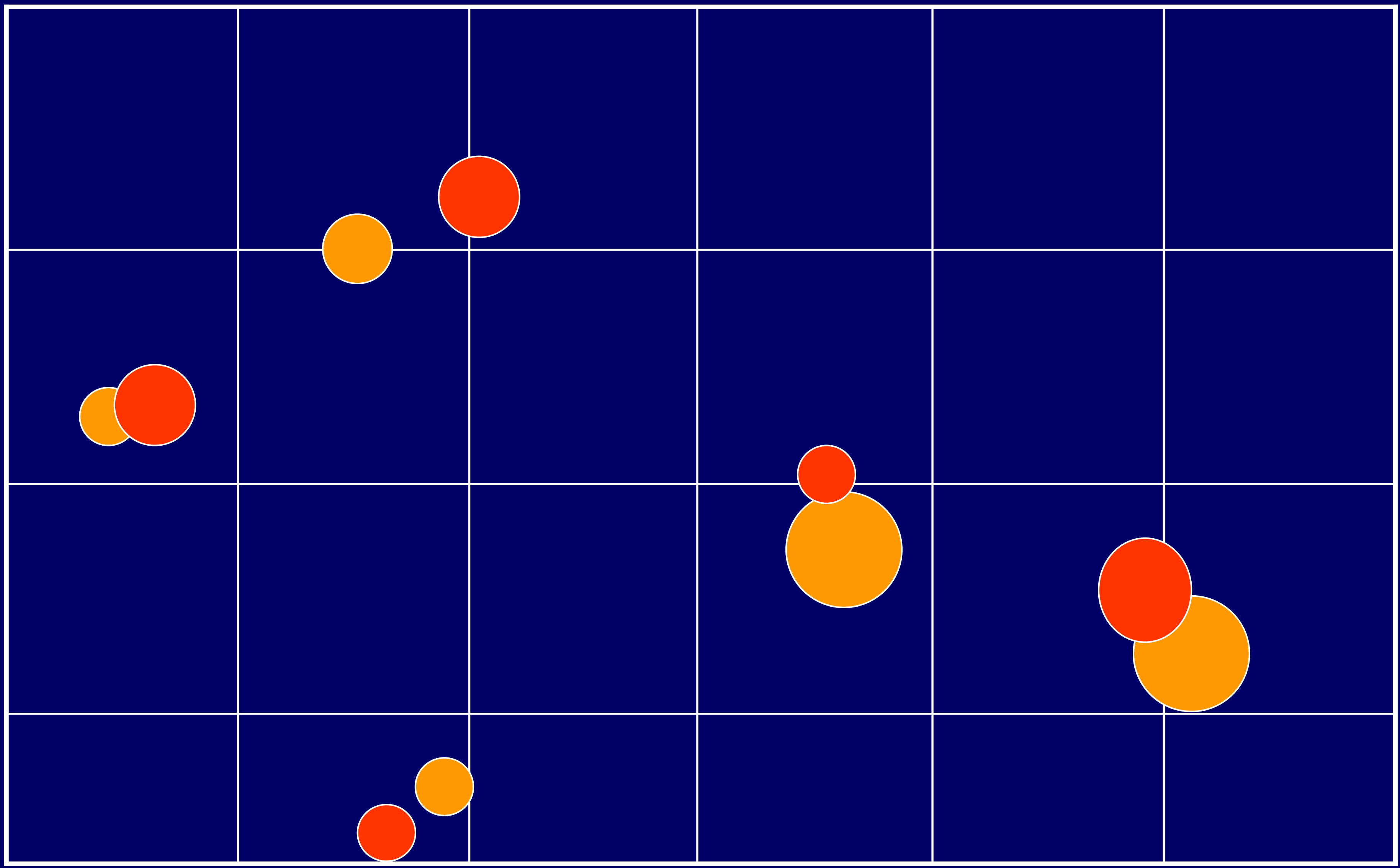
	Column names	Default field separator	Null values
Postgresql MySQL	(provided in CREATE statement)	Tab	\N
TOPCAT	Optional first line	Comma	Two successive commas
R	First line	Space	NA

Cross-matching Problem

- Valuable scientific information often results from combining results from two wavebands or two epochs. When applied to source lists.
  - What one wants to do is to cross-match them to find for each source in one list the counterpart(s) in the other.

- Straight forward in principle, surprisingly complicated in practice.

Schematic of cross-match problem



Cross match requirements

- Match on basis of overlap of error regions
  - May be circles, ellipses, or even more complex
  - Size may be specified as “N-sigma”or by likelihood, e.g. 90% contour.
- Ideally get exactly one counterpart for each source but often get none or more than one.
  - Choose best match, or include all?
  - Include unmatched cases (LEFT OUTER JOIN)?
- Which columns to copy to output – include distance between matching sources?

Variety of cross match algorithms

- Databases with 2-d indexing such as R-tree can handle spatial join (e.g. Postgresql, MySQL).
- For DBMS without 2-d index (e.g. SQL Server) can use
  - Zone method
  - Pixel-based matching (HTM, HEALPix, Igloo, etc)
- Sort/sweep algorithm efficient for large catalogues implemented by CSIRO group.
- **All** of these depend on having both datasets resident in the same DBMS – extending to distributed DBMS is an unsolved problem – latency is a killer.

Other cross-match requirements

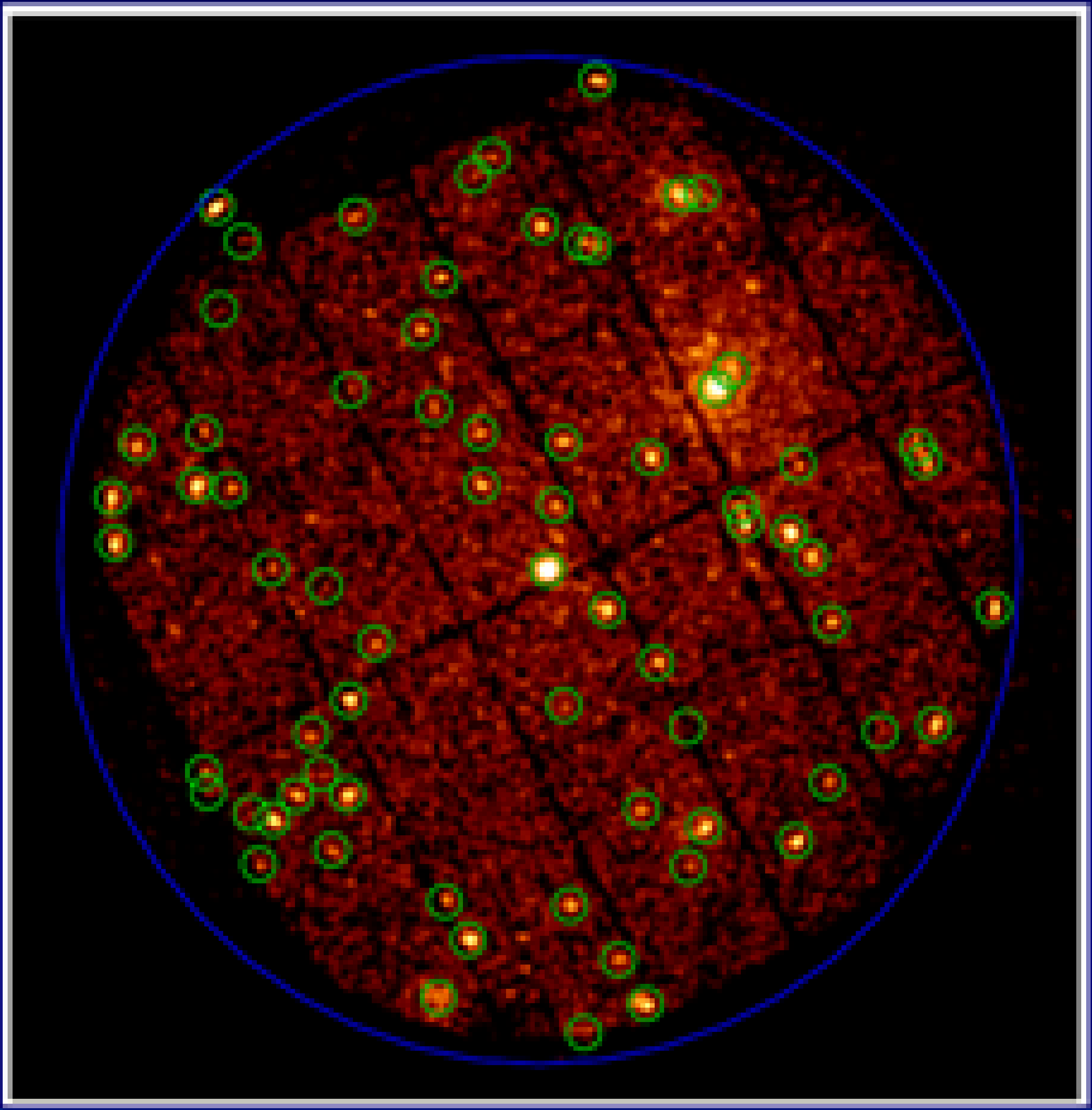
- Where there is no unique match, need to base match on other parameters such as flux, spectrum, distance/redshift etc.
- May need to know the density of sources in the field before the likelihood that a positional coincidence corresponds to a real match.
  - Computing source densities is non-trivial.

- I don't know of any application which supports all of these options as present, even for locally resident catalogues.

A Resource Discovery Problem – sky footprints

- Current plans for VO Registry can find resources such as sky survey results.
- But most telescopes and space observatories have only performed a sequence of discrete pointings (e.g. HST, XMM, Chandra, Integral, etc.)
- To find data available in given part of sky need to store the sky area covered by each observation of each observatory.
  - Cover sky with grid of pixels and store as bitmap?
    - 1 arcmin → 18.5 Mbytes.
  - Store each pointing as sequence of HTM or HEALPix indices?
    - XMM-Newton pointings at 1 arcmin → 11 Mbytes.
- Is there a better way – almost certainly, but not yet researched enough.

XMM catalogue detections in one field



XMM EPIC
T_{exp} =15 ksec

XMM cat srcs

Mike Watson
Leicester University

Finding duplicate detections

- Some fields overlap – so get duplicate detections.
- Resolving these surprisingly difficult
  - RDBMS designed to handle sets with absolutely no duplication.
  - So no built-in software to handle duplicates.
  - Best DBMS method is to start with a spatial self-join to identify duplicates, then weed or merge rows later.
  - Can be done in Postgres with the assistance of some procedural code - which Postgres allows in its user-defined functions (= stored procedures).

Finding anomalies

- Important to check for oddities for two reasons
  - Generally the result of instrumental imperfections, or software bugs, or just source confusion in crowded fields.
    - These need to be identified to remove bad entries from the final catalogue.
  - May be genuine scientific discoveries
    - Need to be studied further and published.

Functionality needed

- Select extrema, e.g. values over Nσ above/below the mean
- Plot histograms to inspect shape, examine tails
- Plot X vs Y for many pairs of columns
- In many cases, e.g. fluxes, need to take logarithms first
- When anomalous entry is found – examine all the other properties of this source (all 300 of them) comparing to what is expected.

Software used

- RDBMS – Postgres
- Table handlers – FTOOLS and TOPCAT
- Statistics package – R
- General purpose package – IDL.
- Various graphics packages (IDL, Grace, GnuPlot, etc).

- Both TOPCAT and R can in principle access tabular data from a DBMS.
  - Have to jump through hoops to get this working.
- Otherwise – only common format is CSV.

Conclusion: where to go next?

- Is XML the solution? If so VOTable may be a start.
- Metadata – is UCD1 the solution – if so need to campaign for widespread implementation.
- Cross-matching: probably DBMS with spatial indexing is the best general-purpose solution. But how to do this over the wide area net is an unsolved problem.