TAMS Analyzer 3.00 Release Notes

TA3 introduces a new layout for the workbench, a new project document structure, and critically, a client-server model for multi-researcher projects (note: if you are a solo researcher, you can still work as usual). MySQL is the first server to be supported. This client server system also has a special mode for "students" which allows a whole class/computer lab to download a project to work on.

Things to note:

1. TA3 creates a project directory in which all files are kept and managed. This should have been done in 2.00 but I'm a slow learner. This primarily affects new projects. Users familiar with XCode will recognize the operating procedure here: pick a location for your project and give it a name. Then pick create.

2. Existing projects are not automatically converted to TA3 directory tree structure, though otherwise TA3 remains compatible with earlier TA projects. I discuss below how to convert your existing files to TA3 file structure which is necessary if you want to share your data through the server.

3. The future synch information is kept in a separate file in the same directory as the project: This means you should not have more than one project in a directory (or the synch information will overwrite itself).

Changes/New features:

1. save as in documents now adds a new file to projects
2. project prompts users for file name for results and documents (it puts them in the right folder)
3. New tabbed project gui. Needed for future functionality!
4. Project file now has tools for code creation and code set creation built right in the project window, though you can still use the traditional panels
5. Project now has a part of the search panel for creating hot code lists
6. File list now permits multiple selection
7. Copy without line numbers
8. Hot code list creation from the workbench
9. Ability to create hot code sets by doing set operations on the current code list (find the intersection or create the union of different code sets
10. Zoom window now stores the zoom factor for document windows
11. Zoom now works in results
12. Conversion function provided
13. Select near feature in results. Provides a hierarchical view of records related to other records.
14. Client-server now working with MySQL as the server
15. Group by code set in results
16. Create a hot code list of all codes not in any of the defined code sets

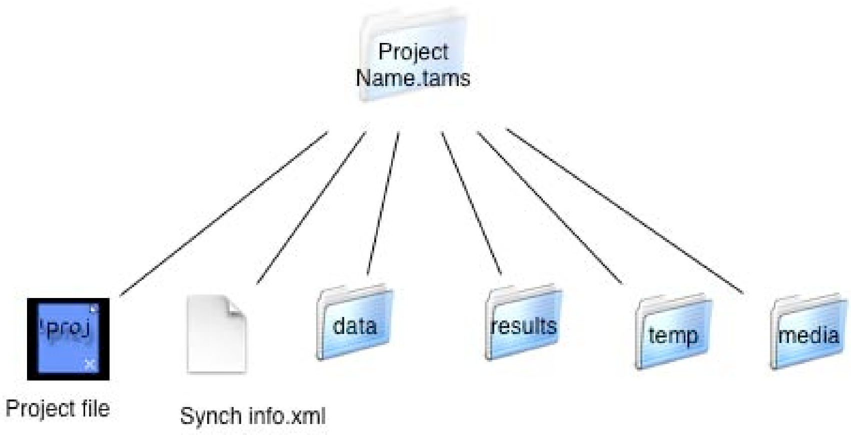
Documentation

Contents

For documentation on the client-server functions see "Using TA3 with Multiple Researchers" in the Docs folder after reading the following. It is too involved to detail here.

1. The new project directory structure:

TAMS 3 takes responsibility for placing files in and relocating files to a specific directory structure. When a file is added to a project it is actually copied to the tams project data (or results) directory. The directory structure looks like the following:



In this release old files are not converted automatically (though see below). Furthermore, some new features will not work (remembering last selected tab, for instance) if you are not using the xml project structure. If your project is of the older type (icon doesn't have an x in the lower right hand corner, extension is .tprj rather than .xtprj) you should do a "save as" to convert the project to the new file format.

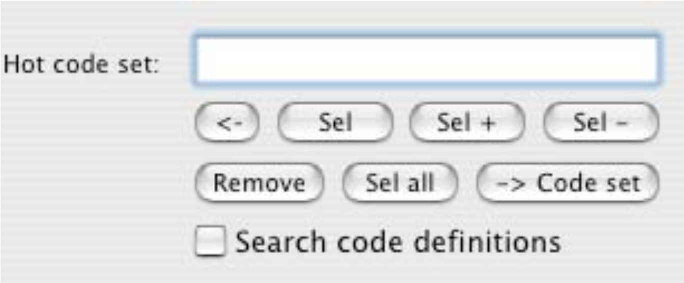
What having TAMS take this responsibility for data files means is that if you click "new," you are immediately prompted for a file name. A "blank" file is created with that name and placed in the "data" subdirectory (these directories are created automatically for older projects and files created with this version are put in the appropriate folders. However, unless you run the conversion function, old files will remain in their previous locations).

Search results are more complex. When you press the search button you will be immediately prompted for a result file name. You have the option of making a search result temporary, which means it will be trashed next time you load the project. This is because results files are often exploratory and not intended to survive from ones tams session to the next. **To expedite this, TA3 will automatically create a temporary result file with a unique name if you just leave the name empty and hit return when you are prompted (don't even bother checking the temp box).**

Note that the whole temp directory is trashed and recreated each time that the project is opened.

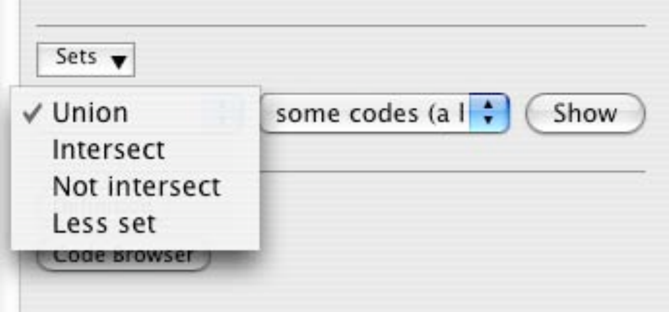
2. Code extensions

In earlier versions of TA, hot code lists could only be created through the document new code field. Basically you could search for codes (and optionally definitions) for words by entering a phrase into the new code field and hitting return. More subtle manipulations could then be done through choices on the Coding->Hot code sets menu.



This functionality is now supported in the workbench under the Search tab. You have the ability to craft subsets of codes by selecting more and fewer items, removing sets that meet criteria, etc. There are buttons to facilitate this or the menu items under the Coding->Hot code sets menu. **Additionally, the dictionary can now be included or not by checking/unchecking a box on the workbench or through a toggle menu item under Coding->Hot code sets.**

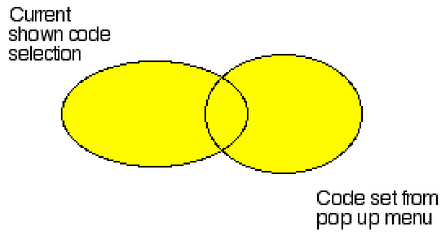
TA3 adds additional functionality for code sets by allowing you to do set operations on whole code sets. For instance, you could have a code set of codes related to "positive student interactions" and a code sets of "students interactions with teachers." You can now pull up the intersection of those two code sets. This is not manipulating data at all, merely the code sets so that you can search for codes that meet very narrow criteria.



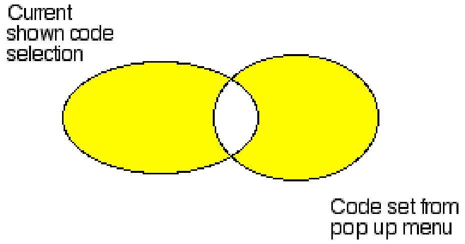
Note that all of the set operations listed above work on the current code set and do the set operation with the set selected by the second pop up menu.

Venn diagrams illustrating the results of the code set operations:

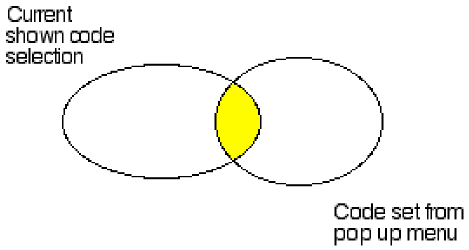
Union:



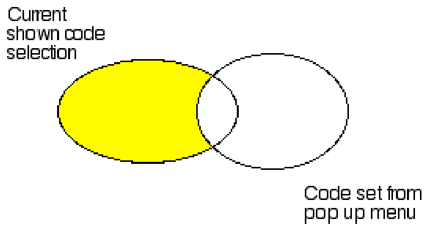
Not intersect:



Intersection:



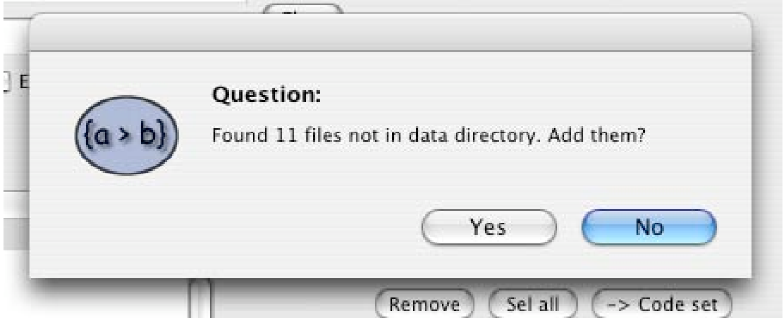
Less set:



3. Converting to TAMS Analyzer 3 format

To facilitate moving to the new directory structure TAMS has a new conversion menu item to help create the directory structure and move the files into the proper folders. To convert from earlier (2.x) versions of TAMS do the following

1. Save the project in a new empty folder as an XML TAMS Project file. The program will say something about saving as absolute paths, just click “Ok”
2. Pick *Project->Convert->Convert to ver 3 directory structure*
3. You will then be walked through a number of steps. First creates the directories, then it tries copying the files into the directories (assuming files are not already there)

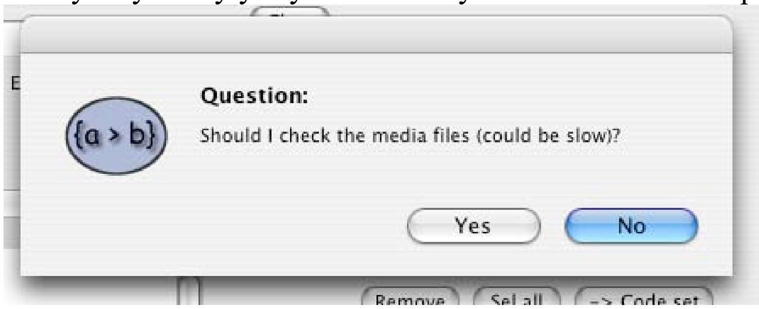


Question:

Found 11 files not in data directory. Add them?

Answering Yes here will copy (not move) the missing files to the ./data directory

a. You may be prompted if it can’t find a file whether you want to remove the “dead record” from the project file. If you say NO you can locate the file and drop it in the /data directory. If you say yes you can always find the file and import it later.

b. 

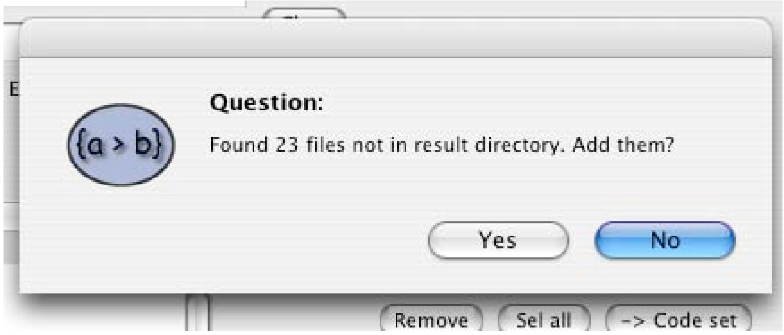
Question:

Should I check the media files (could be slow)?

Answering yes will go through each data file looking for the !mediafile metatag. If that file exists in that spot it will copy it into the ./media directory of your project. If you have moved your media files, it will not find them, however. You can still use your media file by dragging them to the ./media directory yourself. The program always checks there for your media first (even before the directory specified in the !mediafile tag!). If you answer no, no attempt will be made to suss out the media files locations. If you have mp3’s or other media attached to your documents you probably will want to move them to the media directory.

c. The program then report any files it could not find on the hard disk. There’s nothing to do here but click “ok.”

d. The process then repeats itself with result files. It starts by asking if you want to move result files to the ./results folder:

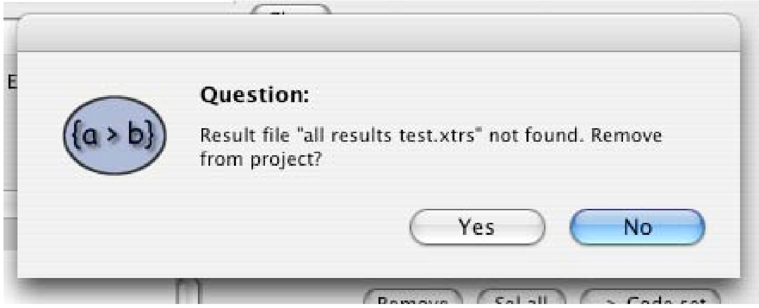


Question:

Found 23 files not in result directory. Add them?

Click yes to have these results files copied to the ./results folder.

e. The program then asks if you want to remove lingering, crufty, useless data that it is holding onto about various files that it still has data structures for. This is a good time to clean up the project file by saying yes to questions like:


Question:

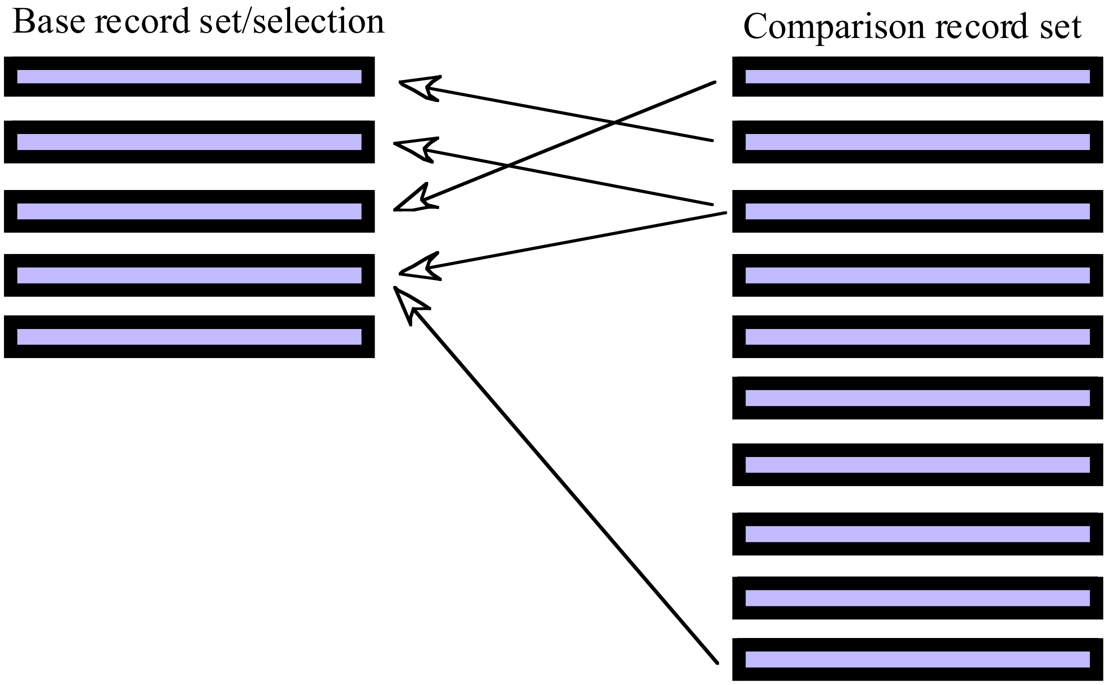
Result file "all results test.xtrs" not found. Remove from project?

f. Voilà, you are done. You now have a TAMS Analyzer 3 project. One thing you might want to do is to add this new project to your work menu and get rid of any old TA 2.XX projects from the work menu you have converted.

4. Select near

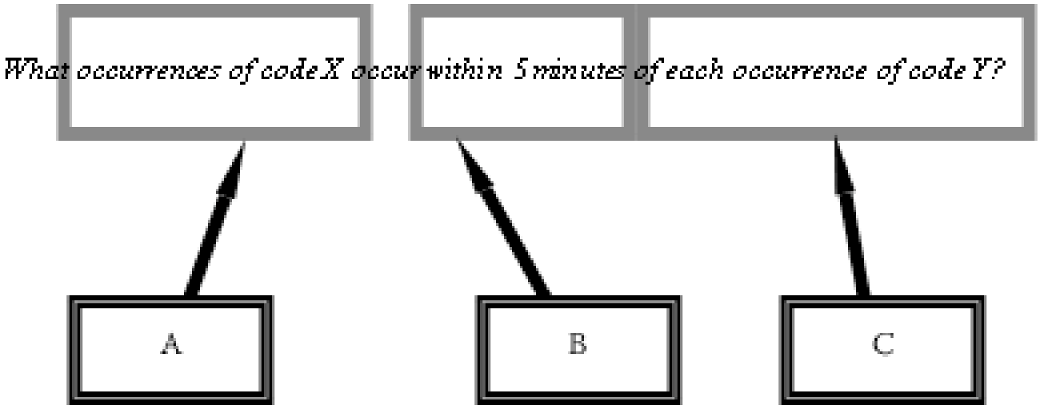
Select near provides a hierarchical view of the data that groups records related to other records in an outline structure. *Select near* allows users to answer questions such as "What occurrences of code X occur within 5 minutes of each occurrence of code Y?" "What occurrences of code set X are within 4 sections of the current selection?"

In both of these model questions there are two sets: a base set which is defined by the current record selection, and a comparison set. In the first question Y is the base set and X is the comparison set. A set image of this relationship is suggested by the following:



It is clear here that each base can have many comparison records assigned to it, and that each comparison record may be assigned to multiple base records.

To understand this complex new function I want to work through the 1st example above. Here it is diagramed with its different parts:

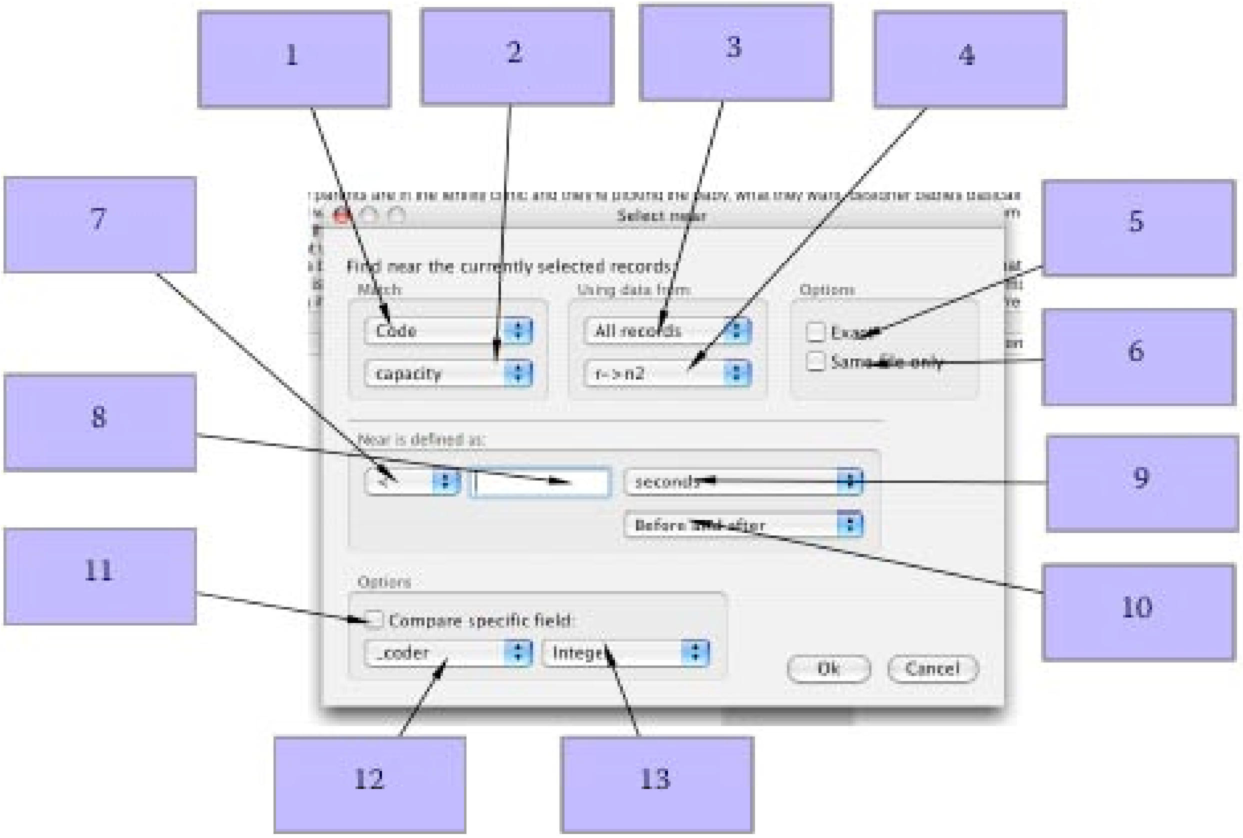

What occurrences of code X occur within 5 minutes of each occurrence of code Y?
A
B
C

Sample Query

1. To turn this into a TAMS operation, first select the records that you want to take as the base of the *Select near* operation. This is determined by the part of the question I've marked as "C," the "each occurrence of code Y." To do this select the _code column and do a simple selection for code Y.

   Note: any selection will work, it doesn't have to be the _code column. It can be as complex a selection of records as you like.

2. Pick "Select near" from the results menu. It will provide you with this sheet



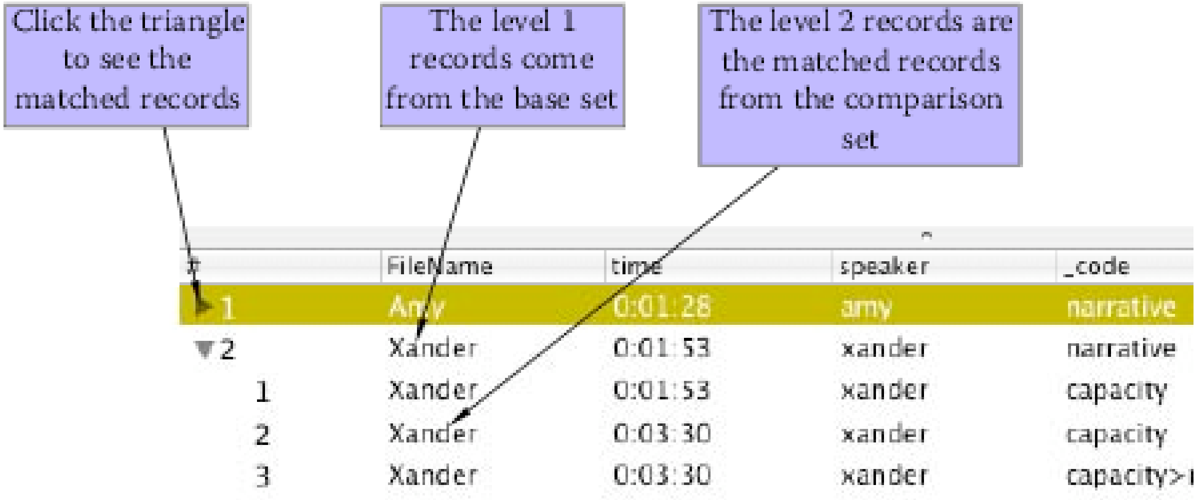
Index to the parts of *Select near* dialogue

A. Since we are looking for code X pick "code" in the "Match" box menu 1and then pick X from the lower menu (Menu 2). This matches the A part of the diagrammed query above..
B. Presumably we want the comparison set to be with all of the records in the result file. If so, pick "All records" from menu 3. Alternatively you can use the current selection or a particular results set. In the latter case pick the results set from menu 4.
C. If we want only to match X select check box 5; if we want anything in the X family leave it unchecked.
D. If we want the matching records only to come from the same source/data file as the base record select box 6. If box 6 is unchecked comparison records from any file will be matched to the base set.
E. According to our Query we want comparison records that are less than 5 minutes from their matching base records (see part B of the query). So for G we'll pick "<" (within is similar to less than) from menu 7. Fill in "5" in field 8. Pick minutes from menu 9. If we want those 5 minutes to be before or after the time in the base record pick "Before and after" from menu 10, otherwise select before (the base record time) or after as is appropriate.

Note that TA will pick the field to examine based on your choice in menu 9. For time units it will use the time field indicated in your program preferences, for characters it will use

_begin_loc, for line numbers it will use _line_number (if you are scanning for line numbers and have used them in your source files. To over ride these decisions use items 11, 12, and 13. Check box 11, pick the field to search from menu 12, and indicate the type of data included in it from 13. You will still need to pick a comparison operator from menu 7 and fill in a value into field 8.

3. At this point pick OK and the results will appear in an outline format:



Click the triangle to see the matched records

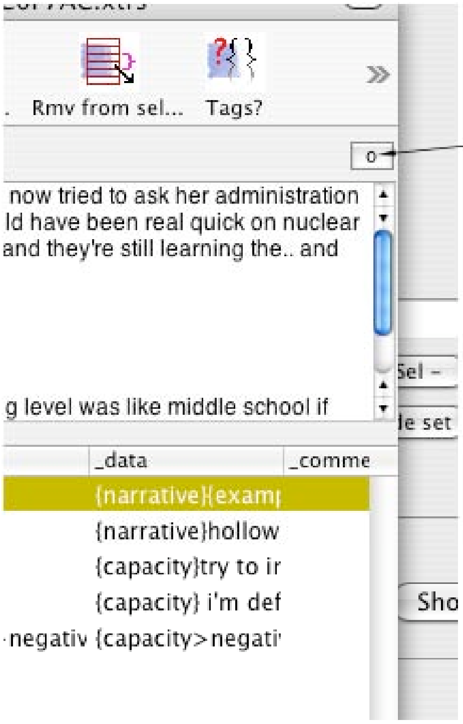
The level 1 records come from the base set

The level 2 records are the matched records from the comparison set

#	FileName	time	speaker	_code
▶ 1	Amy	0:01:28	amy	narrative
▼2	Xander	0:01:53	xander	narrative
1	Xander	0:01:53	xander	capacity
2	Xander	0:03:30	xander	capacity
3	Xander	0:03:30	xander	capacity>

Results of Select Near

4. At this point you will want to get back to the "table view" of your results. TA3 will not allow you to do any thing other than view results and export them from the "outline view." There are three different ways to toggle back and forth between table and outline views..
   A. Use the *Results->Switch data view* menu item
   B. Use the quick toggle button on the info bar. It will switch from O (outline) and T (table) to indicate which view you are currently in:



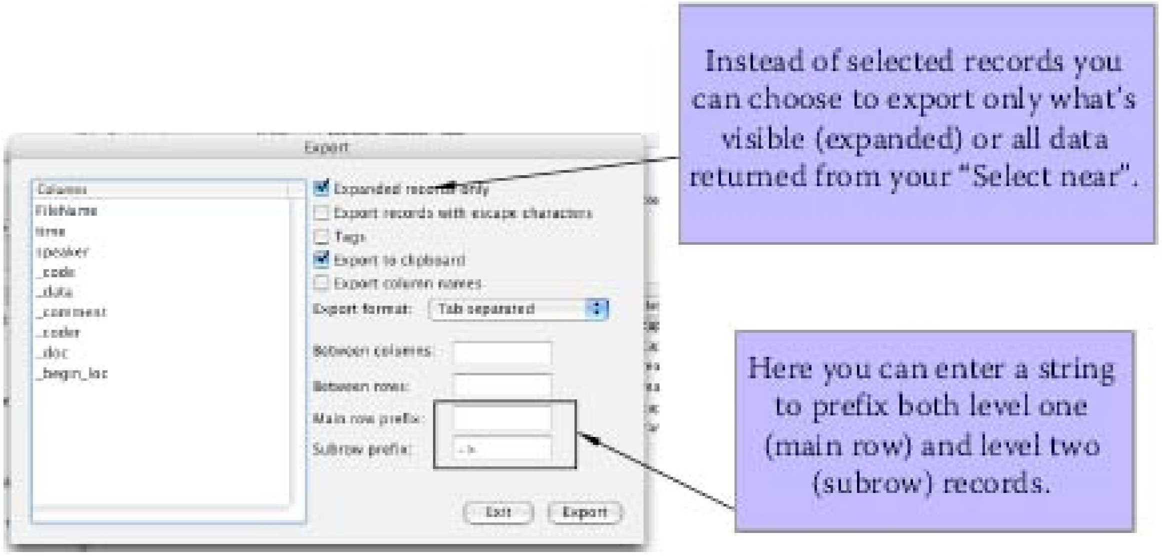
Data view toggle button. It also indicates which view you are currently seeing: o=outline, t=table.

C. You can add an optional button to your toolbar which will toggle the two views:

5. Exporting “Select near” records

Select near results are not saved when the result file is saved. The only way to save these results is to export them with the *File->Export data* menu item.

First make sure that your results window is showing the results of your select near search. In other words, the result window must be in outline view. Pick *File Export data* and you will get an export data pane with some additional fields and altered text from the one for the table view:



Columns
FileName
time
speaker
_code
_data
_comment
_coder
_doc
_begin_loc

☑ Expanded records only
☐ Export records with escape characters
☐ Tags
☑ Export to clipboard
☐ Export column names

Export format: Tab separated

Between columns:

Between rows:

Main row prefix:

Subrow prefix: -->

Instead of selected records you can choose to export only what's visible (expanded) or all data returned from your "Select near".

Here you can enter a string to prefix both level one (main row) and level two (subrow) records.

Modified Export Data dialog

The dialog functions exactly as the export function did before: select the columns you want exported on the left; you can rearrange the order (drag and drop) to meet your needs, and you have a variety of formatting options, both preset and user defined.

Remember that the fields can include escape characters. So if you want an extra space before each level 1 headline fill in “Main row prefix” with “\n” (new line character). “\t” means tab.

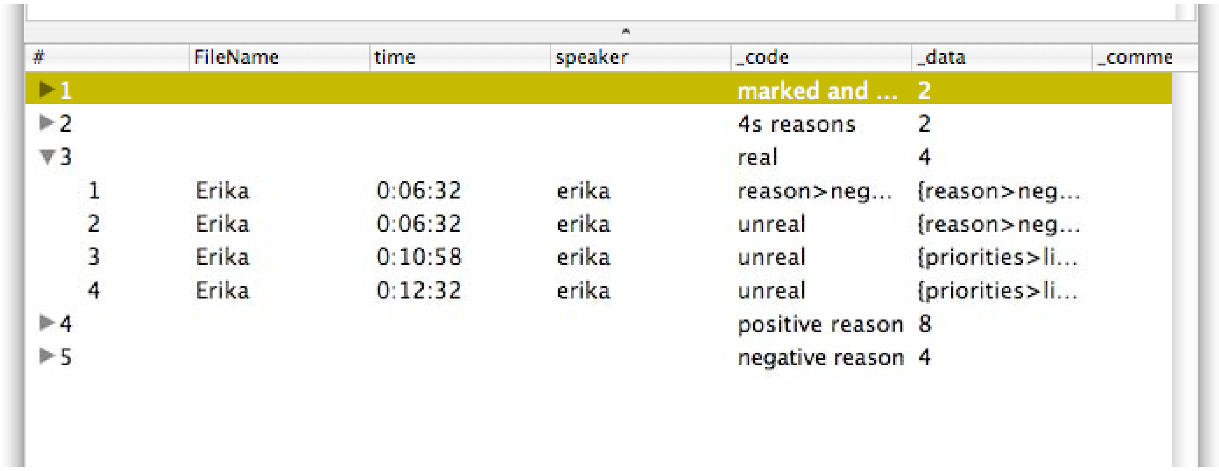
The first check box is modified to allow you to only export what is expanded or visible or to do export all of the results. By expanded, I mean “are the sub rows showing?”

When you hit export you will either be prompted for a file name or your data will be waiting on the clipboard.

Click exit to leave the export dialogue.

6. Grouping and exporting code sets

TA3 also adds additional power to the code set concept. Marked records can now be used to generate a code set using the “Results->Turn marked to code set” menu item. More important for many is the ability to create a code set report including counts of records. Given a shown selection, TA3 will create a hierarchical display showing the code set and count at the top level and all pertaining records underneath. Any given record will appear under as many code sets as is applicable. To generate this just make the selection of the records you are interested in analyzing and pick “Results->Group by code set”. This is what you will see:



#	FileName	time	speaker	_code	_data	_comme
▶ 1				marked and ...	2	
▶ 2				4s reasons	2	
▼ 3				real	4	
1	Erika	0:06:32	erika	reason>neg...	{reason>neg...	
2	Erika	0:06:32	erika	unreal	{reason>neg...	
3	Erika	0:10:58	erika	unreal	{priorities>li...	
4	Erika	0:12:32	erika	unreal	{priorities>li...	
▶ 4				positive reason	8	
▶ 5				negative reason	4	

Results of group by code set

You can see that the code set name is held under the "_code" column and the count of records is held under "_data". In the above example you can see that 4 records had codes in the "real" code set from the shown selection.

Export works as described above in section 5. TA3 tries to be smart about the rows and always prints out the code set name and count using the prefixes and separators you indicate. The subrecords are formatted using your choice of columns and delimiters.

Note, clicking on a code set (level 1) row does not change the display. Clicking on a level 2 row will show the data associated with the selected row.

Note, if you would like an exact match of the codes in the records and the codes in the code sets select the Sort options->Case sensitive flag.

Happy coding. Joyous analysis.

TAMS Analyzer 3.1 Release Notes

TA 3.1 adds powerful new reports including counts and graphs of variables (context codes) and code sets (Reports->Graph code sets, chosen with a result window in front), of code sets relations to codes and to each other (Same menu item chosen from with the work bench window in front), and a code definition and code set definition report (Reports->Code set definitions, work bench needs to be in front). Document searches have been updated to have the same interface and naming process as work bench searches. Default font can now be set for document and result windows. Blanks field entries can be either thrown out or counted as “zero” in select, select near, and auto sets. Improved diagnostics… AND MORE!!!

New features

- Added complex reports so that you can count matches of variables (table columns) and code sets; this can be generated as .dot or .html
- Added report for the relationship of codes to codesets
- Added definition report for code sets and hot code list
- Added font preferences for doc and results in project
- Added "blank = 0" check-box for select, select near, and auto sets
- Document search now works same as workbench search
- Same folder mode now obviated, deprecated and demolished by relative file mode and the new folder structure

Bug fixes

- Fixed outline view so you can use up and down arrows
- Fixed isParentOf function which may affect different options
- Fixed !if so that it works with !name (remember that the universal for !name is FileName)
- Fixed prompts for regex search on workbench
- Improved check pair diagnostic and
- Tweaked the way file names are handled so that if a path exists the file name is returned even if the file is a fiction at that path location
- Changed permission to save project to isReallyOpen from hasFile
- Tags are colored for new files.
- Improved hunting for files for relative and same folder project types

TAMS Analyzer 3.20a3 Release Notes

New features:

- On the fly table columns in results **(see documentation, 3)** that can be selected, counted, sorted, graphed, and used to mark up source documents with universal and context codes.
- New table report that allows you to compare text and counts side by side for different variables organized by code or code set **(see documentation, 1)**
- Ability to mark and unmark records in outline views
- Ability to convert marked to a code set from outline view
- Ability to copy ruler styles from document to document
- AND and OR buttons on the search tabs
- Tool bar buttons in results windows for compare data, group code sets and graph code sets

Bug fixes

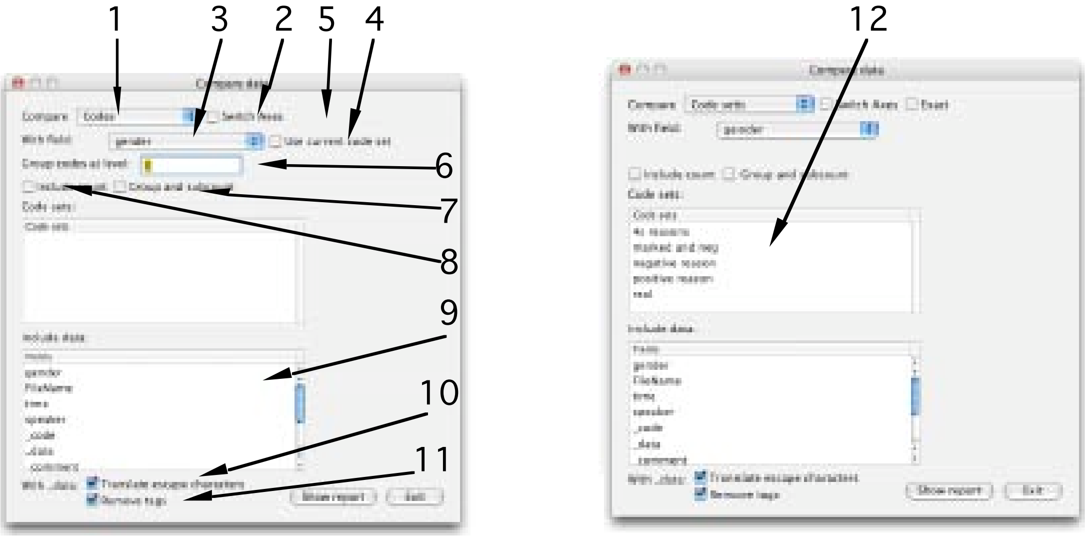
- More careful pruning of files when a project is opened. If a file can't be found the user is prompted as to whether they want to remove it from the project
- Double clicking code in search windows now works
- Export codes and defs now creates a proper project file in the data directory; definitions are trimmed
- More saving using “Export result file formats” in the program preferences search tab, including data export and the new html based reports. This should help foreign language users **(see documentation, 2)**
- Graphviz documents now hard coded to be utf-8 format to support at least other non-english, roman charactersets.

Documentation

1. Reports->Data comparison table

This is an HTML report generated from a result window that opens up in your default browser. It presents either counts, like the data summary, or the actual text of various columns of the result window in a table structure. The column heads are either the code sets defined in the project or the codes of the data currently selected. The rows are the values from one of the columns representing either a context or universal variable. What is filled in in the table are the values from the fields selected from the table, as well as the count. The idea here is that you can directly compare what different people said that you coded X. Unlike the results window which shows you one piece of data at a time, the data comparison table puts comparable data next to each other for your purusal. Note: it is not doing the comparison, you are; it’s just giving you a format that allows you to do so. Also, like dot graphs, an ethos of less is more (revealing) holds here as well. Generating a 200 by 40 table of all your codes matched against all your interviews will be a mess to navigate, at best. So here’s how it works:

The data, as always in TA, comes from the current selection: When you pick Reports->Data comparison table you get this dialogue, which looks slightly differently depending on whether you have selected “Codes” or “Code sets” from menu #1



Data Comparison Table Dialogue

If you pick "Codes" from menu #1, items 4 and 6 will show, but the code set list (table 12) will be empty, since; if you pick "Code sets", items 4 and 6 will hide, and all of your code sets will appear in table 12, as seen on the right screen shot.

The default table will have the codes or code sets across the top. The rows will represent the values from the column you choose from menu 3. If you want to reverse the columns and rows, click on switch #2 "Switch axes", which will put the codes (or code sets) down the table, and the values of the field selected on menu 3 across the table.

For a code data comparison table, the default table will use only the codes from the currently selected rows. If you check box 4, the current code list (from your work bench) is used as the basis. Only rows in the current selection that have a code included in the current code list will appear in the table. Use the exact switch (item 5) to determine whether the codes of the current selection have to exactly match the current code set (a>b will not match a) or not (a>b does match a, it's in its family). **Note: the exact switch is only visible if item 4 is checked!**

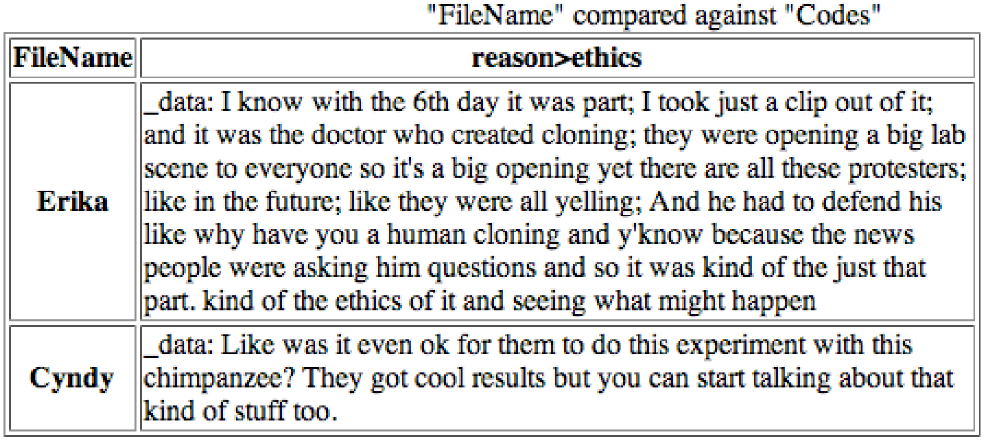
For a code data comparison table you can group codes together by setting a code level. This is just the same as with autosets and other tables. If you put "1" into item 6 you a>b and a will be treated as the same. If "0" is entered all levels are matched.

For both types of tables (codes and code sets) you can use switch 8 to include a count with your data. You can also provide subcounts of the cluster of fields chosen in table 9, by checking box 7. Note that this does not give separate counts for the data selected in table 9, but of the fields in table 9 treated as a single entity.

The data you include in your table is selected from item 9. You can drag to rearrange and multiply select the information you want to compare in your table. If you are including your "_data", which typically you will, you can have the program remove or include tags as well as convert escape characters (\n for a new line, \t for a tab) using items 10 and 11.

For code set data comparison tables you will be provided a list of project code sets in item 12. You must select the code sets you want included in your table from this list. The codes from records from the current selection in your results window are compared to the code sets either exactly or not depending on the state of item 5. Otherwise the choices are similar. You must choose a universal or context value from menu 3, you have the option to have counts printed in the table, you pick the data you want to compare from table 8, and items 9 and 10 determine how _data information will be formatted.

The report you get simply allows you to see the value of a code (or code set) across files:

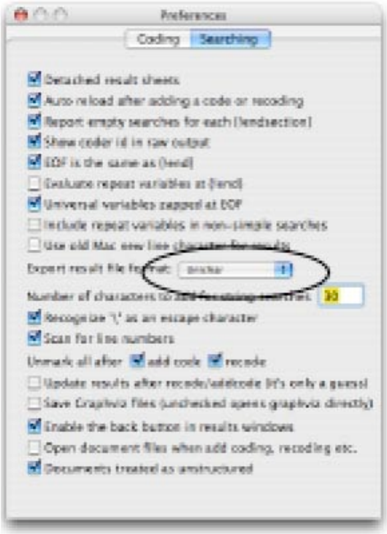


"FileName" compared against "Codes"

FileName	reason>ethics
Erika	_data: I know with the 6th day it was part; I took just a clip out of it; and it was the doctor who created cloning; they were opening a big lab scene to everyone so it's a big opening yet there are all these protesters; like in the future; like they were all yelling; And he had to defend his like why have you a human cloning and y'know because the news people were asking him questions and so it was kind of the just that part. kind of the ethics of it and seeing what might happen
Cyndy	_data: Like was it even ok for them to do this experiment with this chimpanzee? They got cool results but you can start talking about that kind of stuff too.

Screen shot of a data comparison table

2. Saving files and using reports with non-English character sets.
This release extends the ability to save files using different character encodings. This is very important for users that are working in non-English, or non-ASCII character sets. Export data (when you save data to a file, not to the clipboard) and the html reports generated by "Graph code sets" and the "Data comparison table" use the information in the program preference panel under the "Search" tab:

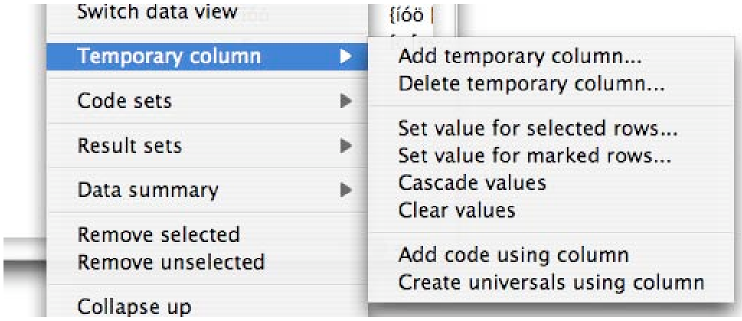


☑ Detached result sheets
☑ Auto reload after adding a code or recoding
☑ Report empty searches for each [endsection]
☑ Show coder id in raw output
☑ EOF is the same as [end]
□ Evaluate repeat variables at [end]
☑ Universal variables zapped at EOF
□ Include repeat variables in non-simple searches
□ Use old Mac new line character for results
Export result file format: [illegible]
Number of characters to add for missing searches: 30
☑ Recognize '\' as an escape character
☑ Scan for line numbers
Unmark all after ☑ add code ☑ recode
□ Update results after recode/addcode (it's only a guess)
□ Save Graphviz files (unchecked opens graphviz directly)
☑ Enable the back button in results windows
□ Open document files when add coding, recoding etc.
☑ Documents treated as unstructured

The character set option in program preferences

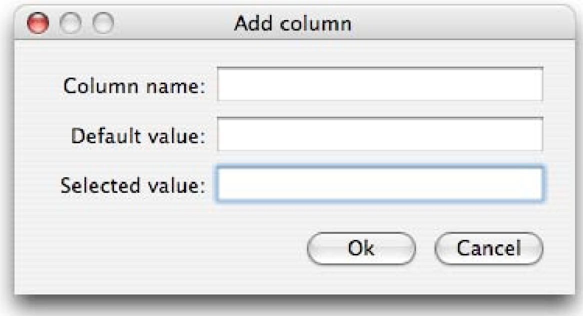
Note that this does not affect graphviz documents and reports. Those are now generated using utf-8 format, which will at least support some additional, non-English, character sets, but cannot support many (any?) non-European character sets.

3. On-the-fly, temporary data columns
Now tams results windows allow you to create new columns on the fly. These don't change the source files (initially), but they allow you to group and select data and create analysis tables using emergent categories. The data in these columns can be created through setting values directly (double clicking on a cell), setting values for the selected records (or marked records), and by cascading values so that blank cells take their contents from the ones above them.



The "Temporary column" submenu of the Results menu

While temporary columns and their contents are saved if the result file is saved (as a non-temporary result file), they are destroyed if you refresh or do anything thing that refreshes results windows (check your preferences!). For most of the functions listed in the submenu, you must select the column by clicking its header. When you add a column you fill in this dialogue:



The add column dialogue

The column name is the only required field. The program fills in the default value for every record. Then it goes through the selected records and fills in the value in the "Selected value" field. It does this even if both are blank.

DANGER

The temporary columns can also be used to code at both the !universal and !context levels your source documents. Playing with these features can potentially scramble your data files, are likely to have unintended consequences, and if used more than once—will cancel earlier uses of these features in unintended ways. But if you want to play with fire here are some implementation notes.

You can set a repeat value using the !setcontext. What tams will do if you pick “Add code using column” will be to insert “{!setcontext X=“Y”}, where X is the column title, and Y is the value in X for that row. If you have 3 or 4 rows of a result window with nested codes marking the same data, you can imagine the sort of mess this can leave in a file.

Somewhat, but only somewhat, less dangerous is the “Create universals using column” feature. This will first check that only one value in the selected rows applies to any file and will inform you if more than one applies (without doing anything). If it finds that one value maps to each file (though the same value can apply to many files), then it will go through each file represented in the selection and insert at each one’s start a {!universal X=“Y”} metatag where X is the column head and Y is the value for that file.

Both of these features also alter the init file of the project. “Add code using column” inserts a {!repeat X} where X is the column title at the start of the init file. “Create universals using column” inserts a {!universal X=””} at the start.

Use at your own risk!!! REMEMBER YOU CANNOT UNDO THIS!

So to recap: Add columns, that’s safe. Just be careful if you use the two menu options which tinker with your source files

TAMS Analyzer 3.21a6 Release Notes

New Features:

- automatic generation of !if based on on-the-fly table columns (See documentation)
- Data comparison table improvements
    - ability to group without count
    - ability to set the vertical and horizontal justification for different table elements
    - ability to sort each axis (or not)
    - ability to make a table from any two data columns rather than just codes and code sets

Bug Fixes

- close button eliminated in new dialogues
- various menus are now constructed more carefully by checking the universals and repeats values
- closed file handling is improved
- fill selected and fill marked now operational

Documentation:

You can now automatically generate an !if table based on a temporary data column. An !if table is simply a series of !if statements which set the value of a variable based on the value of a different variable. TAMS only looks at the selected rows when setting up the !if table. To create an {!if X="Y" => A="B"} statement TAMS will use the name of the temporary column for A, and the values in that column as B. When you pick "Results->Temporary column->Generate !if table" it will prompt you for the column to use as the source for X and Y. Pick the column from the pop up menu and then the name of the column will be X and the matched values will serve as "Y". **You must have a designated init file for this to work.** Also, you will not be allowed to create the table if there are conflicting assignments, i.e., if in your selection for a given (X,"Y") set of pairings there are both (A, "B") and (A, "C").

TAMS Analyzer 3.22b9 Release Notes

New Features:

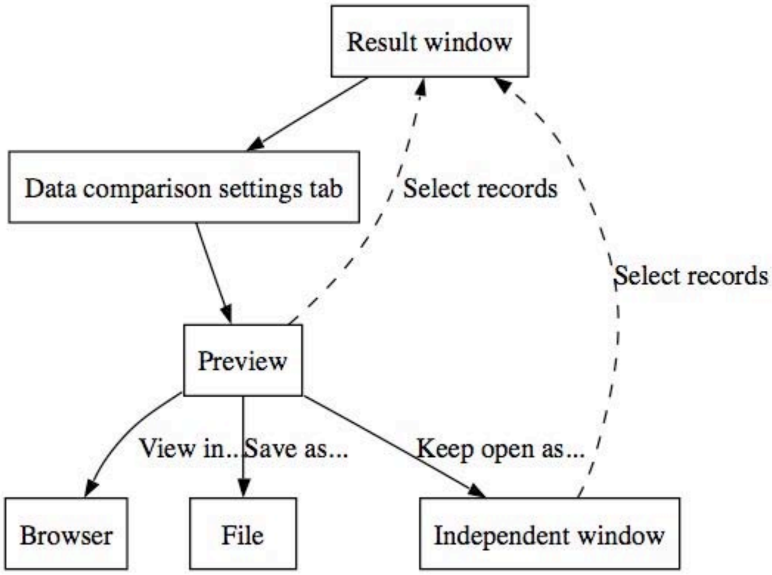
1. Data comparison tables now are rendered in tams with an option to load the tables in your browser (requires that webkit is installed; for panther/tiger users this isn't a problem. Jaguar users will need to run Safari to install webkit on their machines). See Documentation.
2. Data comparison tables now calculate totals if count or "Group and subcount" are selected.
3. New ways to work with temporary data columns
    a. Convert selected records to lower, upper, or title case.
    b. Copy values from another column to a temporary column
    c. Set value now can append the given value to the temporary data column
    d. Use regular expressions to transform temporary data columns
4. Data summary table dialog now stays open until the "Exit" button is clicked. This allows the user to generate multiple tables before exiting the dialogue.
5. Better white space stripping in various operations that alter code definitions

Documentation

In this version of TA, data comparison tables are linked back to the data. You can either have TA pick out the item from the current selection or select it (so only those items are showing). The idea here is that you can use the TAMS web windows to explore your data but then pop the data open in your browser to print it (looking at linked text for a long time can drive you crazy!)

So I want feedback on this. How does it work for you? Is it too complicated? How would you like it all to behave? How might it be laid out better?

Just so that you see the new logic, here is my little dot graph of how it all works:



Logic of running data comparisons

So the way to work the new data comparison dialogue is to set the parameters for a table, then hit show report. This switches the window to the preview mode, which lets you see your table (and use it: you can click on any cell to highlight or select that data). Now you can go back to the “Settings” tab to modify the table until it represents your data as you wish. At this point you can spin off windows for the tables you like from the preview tab by clicking “Open in new window”. This step is necessary since the preview will change as you keep making data comparisons, so either spin it off or you’ll need to regenerate the report. For more control, over printing, over font size, etc., you can open the table in your browser. Note that both in the preview tab, and in windows spun off from the tab, you can choose to highlight, select, or add to current selection, the data you click on.

One hint: If you are selecting rather than just highlighting records, and you then return to highlighting, the records you want to see will probably not be in the current selection. Use the back button of the browser to get back to the selection state you need.

TAMS Analyzer 3.23b1 Release Notes

New Features:

1. Word count across all documents in search list (See documentation pt. 1)
2. Word count on select fields of result windows (See documentation pt. 2)

Bug Fix:

1. Hot code list on workbench fixed so that selecting code list (or other item) doesn't trigger a hot code search

Documentation

1. Project word counts

This release adds the ability to count words (defined initially as a series of letters) across all files in the current search list. From the workbench pick Reports->Word count. You will then be asked if you want to "sort by words" (i.e., alphabetically). If you click "yes" the count will be presented in alphabetical order, otherwise the list will be in frequency order with the most commonly used words listed first. The report will break down word use by document and with totals and grand totals.

What is a word?

By default a word is just letters. This leads to some problems: the word "don't" for instance will be considered 2 words. To add additional characters to what defines a word pick "Project->Word count options…". Simply type in the additional characters you want to be considered when defining words. To add numbers, for instance, and the apostrophe type in "0123456789'" (in any order). Click "Ok". Now run your word count…

Saving a word count

The word count is rendered as an HTML table. If you wish to save the table, click the save button at the bottom of the table, this will prompt you for a file name. The table will be saved as a tab delimited (rows terminating in a new line character) file, which is great for Excel and other databases and spreadsheets. If you wish to save a word count report as html, click the "Open in browser" button and use your default browser's "Save as" to create the file for the table.

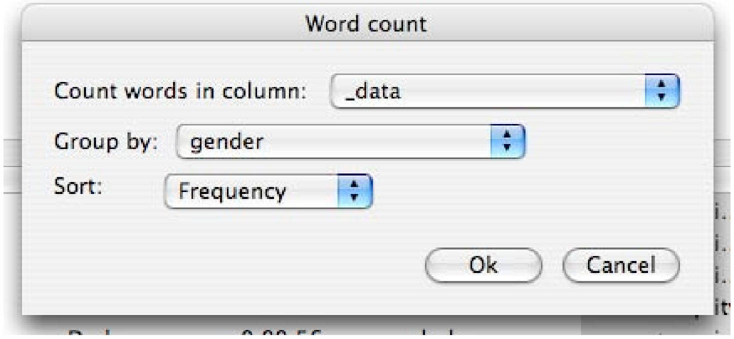
Note, word count examines the file without tags. If you wish to know how many tags/codes you have, you will need to use the Code count report.

2. Result window word counts

A more complicated type of word count can be generated from result windows. Here you can count codes not just by file, but grouped by the values of any table column. There are some serious problems that can arise with this, since simple searches can return the same text multiple times (i.e., if a passage is coded twice, it will have 2 rows in a result window, and the words will be counted twice in the word count). So you need to know what you're counting and why. Generally you will want to select a very small subset of your data, for instance chosing one code (that should prevent the problem just mentioned) and examine the words people are using for that single code.

Note that result windows use the “Project->Word count options” to determine what counts as a word.

With a result window in front, picking “Report->Word count” results in this dialogue:



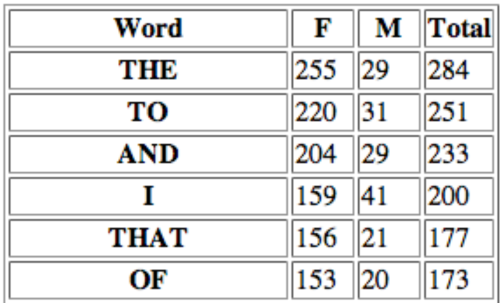
Count words in column: _data

Group by: gender

Sort: Frequency

Results word count document

The first menu (“Count words in column”) determines what words to count. The second menu (“Group by”) groups the counts by the values in that column. Picking gender in the “Group” will break the counts by gender:



Word	F	M	Total
THE	255	29	284
TO	220	31	251
AND	204	29	233
I	159	41	200
THAT	156	21	177
OF	153	20	173

Word count by gender

The last menu item indicates to TAMS whether the data should list the words in order of frequency (count) or alphabetically.

Again, clicking “Save” will save as a tab delimited file.

TAMS Analyzer 3.23b2 Release Notes

This release adds one small, but powerful feature to data comparison tables and fixes two formatting problems with the tables. There is now an “empty” check that will appear for a couple of search types in the “With field” box. This will return values of the field picked in the “With field” menu and display them even if they have 0 matches with the field picked in the “Compare” box. This way you can identify immediately who did not have code X, Y, or Z. in their file, for instance. There are only two types of searches where this is available. If you are using “Codes” in the “Compare box” and you check the “Use current code set” box, then you can check the “empty box”. In addition if you pick code sets, you can see which values of the “With field” field do not have codes in the selected code set.

New Feature

- Empty flag now implemented for code and code set searches

Bug Fixes

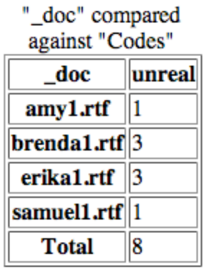
- Corner “grand” total column does not inappropriately get created. Previously, it would show even if there was no vertical count column.
- Division bar not printed if count is 0

Documentation

Empty flag in Data Comparison Reports

Previous versions of TA in the Data comparison report, only reported counts greater than 0. In this example I am looking at how many of my interviewees (n = 7) used code unreal.

1. I create a hot code set from my desktop for “unreal,”
2. I create a simple, unlimited search (i.e., I just click the search button on the workbench without using a search string)
3. I pick “Reports->Data comparison table”
4. For the “Compare” box, I pick codes
5. I check the “Use current code set” box
6. I check the exact box
7. For the “With field” box I pick “_doc” from the pop up menu
8. From the Data elements box I check “Include count” (leave everything else as is)
9. Click generate report and I see something like this in the preview pane:

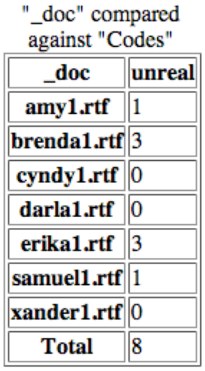


"_doc" compared against "Codes"

_doc	unreal
amy1.rtf	1
brenda1.rtf	3
erika1.rtf	3
samuel1.rtf	1
Total	8

Results from not using the empty check box

As you can see four of my interviewees used unreal. But what if I had enough other interviewees that I would want to see who didn’t use unreal. In other words, those with a count of 0 don’t show up. To remedy this just press the settings tab and click the “empty” switch in the “with field” box. Click generate report and now I see:



"_doc" compared against "Codes"

_doc	unreal
amy1.rtf	1
brenda1.rtf	3
cyndy1.rtf	0
darla1.rtf	0
erika1.rtf	3
samuel1.rtf	1
xander1.rtf	0
Total	8

Same report with “empty” checked

Now I see explicitly that Cyndy, Darla and Xander did not use unreal.

The same logic also works with code sets. To see who does not have a code in a particular code set check the empty box.

TAMS Analyzer 3.30b1 Release Notes

New features

1. Ability to make virtual "end"s either end's or endsections.
2. Ability to create contextual variables that reset at endsections, ends, end of files (eof) or never.
3. New code and add code now have menu items and key equivalens (Document window)

Bug fixes

1. Result file resize bug: often the browser panel would disappear or strech in bizarre ways if a window was resized
2. Generate code list now is operational again
3. Scan for line number partial fix
4. Fix for remove files for multiple selections

Documentation

To understand the changes implemented in this release it is important to understand how TAMS handles context and universal variables

How context and universal variables work (up to this version)

When TAMS reads through the init file and the files in your search list it creates an account for each declaration of a universal or context variable it finds. As it scans the document, if it's a structured file (i.e., it uses !last, !first, !end, !endsection, or !struct) it keeps track of what each context/universal variable's value is, it also creates a list of all the coded passages it finds. Here's the important point: it doesn't match up the coded passages to the context and universal variables until it hits an !end or an !endsection (or a virtual one placed there by a !last or !first tag). At that point it matches the values of the context and universal variables with the coded passages and creates one row in the result window for each coded passage. If it is an unstructured document every time it hits a coded passage (the end tag of one) it matches the values of the universals and repeats with the coded passage and creates a record right there and then in the results window.

Now before this release (3.3), if in scanning a structured document the program hit an {!end} tag, the program would clear away all the values of the context variables after creating the record in the results window. If it hit an !endsection tag, it would create the row in the result window *but not clear the context variables*. The values would just cascade through to the end of the document. Universal values cascade through regardless of !end and !endsections.

Unstructured documents simply treat every coded passage as if it has an !endsection before it, letting values cascade to the end of the document.

What's new? Controlling virtual ends

The !last and !first metatags, up to this version, stuck virtual !endsections through the document. Declaring {!first X} meant that every occurrence of X was treated as having a !endsection in front of it; {!last X} meant that the *next context code* that appeared after X occurs was treated as having an !endsection in front of it.

Starting with version 3.3, researchers can decide on the fly whether !last and !first create virtual !endsections or virtual !ends (the difference being that context codes are blanked out after !ends.

To set whether !last and !first create !ends or !endsections use the {!virtualend} and {!virtualendsection} tags in your init file. The program defaults to {!virtualendsection} if you do not indicate which you prefer (which was the former behavior of the program).

This is particularly powerful when mixed with the new ability to create variables with specific “horizons”.

Unified variable space

In previous versions TAMS kept separate “accountings” of context variables and universal variables. There were two sets of books: one for universals the other for contexts. That is why !if statements and !map statements had to connect similar variables (context to context; universal to universal); you were not allowed to map a context variable onto a universal variable or vice versa.

In this version variables are all accounted for in the same log book, which now enables that sort of mixing. More important, the variables can actually switch their type. Basically what TAMS tracks are two properties of a variable with a given name (say X). First it tracks the value of the variable (e.g., whether a variable named "sex" is currently set to “m” or “f”) and the horizon of the variable. The horizon indicates what triggers a variable to clear its value. Previous versions of TAMS would only clear context variables at an {!end} tag, and a universal at the end of the file (eof). This version allows you to clear the value at an endsection, an end, an eof (end of file), or never. These points represent the horizon of the current variable, the point at which its value is blanked out (in technical terms, changed to an empty string). So in the new way of thinking, context variables are simply variables whose horizon is “end” and universal variables are ones whose horizon is “eof”. Now there are two other possibilities. You can have variables that get their value wiped at !endsection or never get their value scrubbed at all!

To manage this, TAMS 3.3 introduces a new metatag: !var (for variable). This metatag lets you set a combination of the name, value and horizon of a variable, separated by commas:

{!var name="City", horizon="endsection", value="NYC"}

Note that name, horizon, and value can happen in any order. Also only name is required. {!var name="City"} is the same as {!var name="City", value="", horizon="end}, if a variable "City" has not been created (through !var, !context, !universal, etc.) earlier.

If a variable "City" has already been created then only the listed properties (value and/or horizon) are changed. In other words if a variable City is already declared {!var name="City", horizon="eof") only changes the horizon, not the value.

So !var combines the ability to declare variables, set their value, and set their horizon at the same time.

A single !var metatag can declare several variables by separating them by a semicolon: {!var name="City"; name="Gender", horizon="never"}

If this were placed at the top of the init file it would create two variables: "City" with an initial value of "" (nothing) and a horizon of "end" (that's the default) and a second variable "Gender" which never is wiped out and also has a blank ("") value.

So what happens to legacy calls like:

1. {!context gender, city}
2. {!name "My doc"}
3. {!universal type="Interview"}

??? TAMS 3.3 just saves them in the same accounting system, treating them as if they were created with the following:

1. {!var name="gender", horizon="end"; name="city", horizon="end"}
2. {!var name="FileName", value="My doc", horizon="eof"}
3. {!var name="type", value="Interview", horizon="eof"}

An example

Someone asked me recently to help with the following problem. Given an interview like:

```
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```

They first wanted to structure it using !last, with time and name as context variables::

```
{!context time, name}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```

This basically stuck a virtual **!endsection** in front of every {time}, except the first one (since no variable named "name" was used before then). To TAMS the data now looks like this, because of the !last statement :

```
{!context time, name}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{!endsection}{time}00:01:25{/time} {name}Sally{/name} First time, you?
{!endsection}{time}00:01:30{/time} {name}Bob{/name} Every day!
```

Now the researcher wanted to add some cultural information about Bob, so he declared a new variable "culture" and used **!if** to assign it to Bob. Note this is what is in the document window, not what TAMS sees (I've removed the virtual **!endsection**s)

```
{!context time, name, culture}{!if name="Bob"=>culture="Canadian"}
{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```

The problem he had was that when he searched, Sally turned into a Canadian as well, since !endsection does not clear the value of !context variables, and no other value had been assigned to Sally as her "culture".

One solution would be to make "time" a universal code, though universals are designed to be constant over the scope of a whole document. Another solution might be to hardcode !end's before each value of {time} (not too bad with regular expressions). However, it uglifies the data. With TA3.3 he could also make !last create virtual !ends rather than !endsections, which would accomplish the same thing without the uglification.

So what our researcher would type would be:

```
{!context time, name, culture}
{!if name="Bob"=>culture="Canadian"}
{!virtualend}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```

!virtualend tells TAMS to create implied **!end** statements rather than **!endsection**s with **!last**. And what TAMS would see would be this if the virtual **!end**s were to be made explicit.

```
{!context time, name, culture}{!if name="Bob"=>culture="Canadian"}
{!virtualend}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{!end}{time}00:01:25{/time} {name}Sally{/name} First time, you?
{!end}{time}00:01:30{/time} {name}Bob{/name} Every day!
```

In this case, this would work because every speaker also has a time code. In my interviews this wouldn't work because I don't enter a time code for each person. I may have several turns of conversation after a given time code. In other words, I want time codes to trickle down (which they wont because !end clears context codes, which includes "time"), but I want culture codes to be wiped. So the better approach might be to let !last be !endsection, as before, but to indicate that culture's horizon is an !endsection, and time's horizon should be the end of the document (or even !end, in this case, since there are none!):

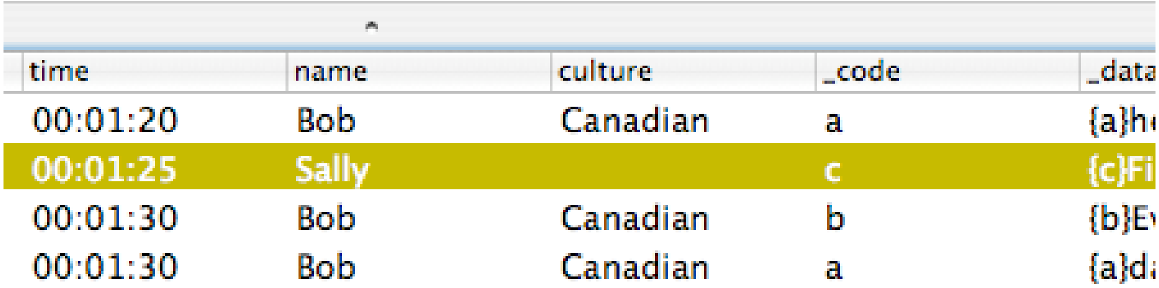
```
{!context time, name, culture}
{!var name="time", horizon="eof"; name="culture", horizon="endsection"}
{!if name="Bob"=>culture="Canadian"}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```

Now "time"s will carry forward, speaker to speaker til the end of the document, while "culture" will be wiped clean for each speaker! Note that it is redundant to have "time" and "culture" declared as context variables here, since !var will create those variables if they don't exist. So I could have just done:

```
{!context name}
{!var name="time", horizon="eof"; name="culture", horizon="endsection"}
{!if name="Bob"=>culture="Canadian"}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```

I could also have rolled name right into the !var statement:

```
{!var name="name"; name="time", horizon="eof"; name="culture",
horizon="endsection"}
{!if name="Bob"=>culture="Canadian"}{!last name}
{time}00:01:20{/time} {name}Bob{/name} Come here often?
{time}00:01:25{/time} {name}Sally{/name} First time, you?
{time}00:01:30{/time} {name}Bob{/name} Every day!
```



time	name	culture	_code	_data
00:01:20	Bob	Canadian	a	{a}h
00:01:25	Sally		c	{c}Fi
00:01:30	Bob	Canadian	b	{b}E
00:01:30	Bob	Canadian	a	{a}d

Figure 1: Sally is not Canadian, and all is right with the data

TAMS Analyzer 3.31 Release Notes

New features

1. Co-occurrence report

Bug fixes

1. Word count function restored
2. rtfd support restored

Documentation

Co-occurrence Table (on Report menu)

This release adds a report that is something akin to the quantitative idea of correlation. It allows you to track what variables are related each other. To conceptualize what this table tells you, you have to understand that three independent things are tracked:

1. a watched variable: this is the variable you want to compare
2. a group variable: this defines what constitutes a co-occurrence only if this variable shares a value with the watch value will it be tracked
3. a count variable: this defines what data is monitored when the group variable is shared.

Example of using this report.

1. Given a set of group interviews where I have coded (probably using the !if statement) such things as sex, culture, age range, etc. I could ask what cultures were involved in the groups for each member of the database (especially useful if one person is in more than one group). In this case my watch variable would be the whatever my interviewee column (i.e., the column that names the person), my count column would be culture, and my group column would probably be _doc or FileName (assuming 1 document = 1 conversation).
2. Given a summary document describing who participated in a series of interactions like this:

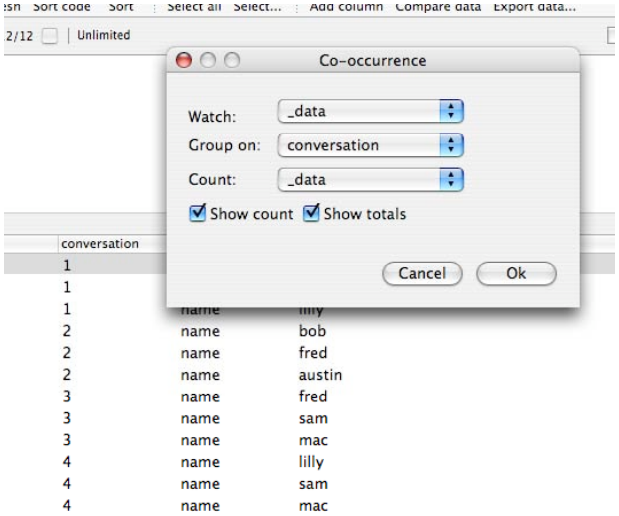
```
{!name "file1.rtf"}
{!context conversation}

{conversation}1{/conversation}
{name}bob{/name}
{name}sam{/name}
{name}lilly{/name}

{conversation}2{/conversation}
{name}bob{/name}
{name}fred{/name}
{name}austin{/name}

{conversation}3{/conversation}
{name}fred{/name}
{name}sam{/name}
{name}mac{/name}

{conversation}4{/conversation}
{name}lilly{/name}
{name}sam{/name}
```

```
{name}mac{/name}
```

I can analyze the interaction among people by

a. Doing a search with raw turned off
b. Running the co-occurrence report with
    i. Watch = _data
    ii. Group = conversation
    iii. Count = _data

So the dialogue will look like:

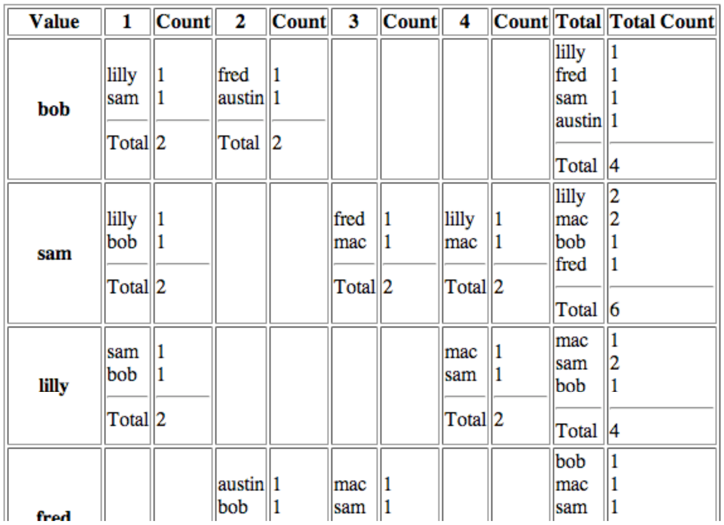


esn Sort code Sort Select all Select... Add column Compare data Export data...
2/12 Unlimited
Co-occurrence
Cancel
Ok

Watch: _data
Group on: conversation
Count: _data
☑ Show count ☑ Show totals

```
conversation
1
1
1          name        lilly
2          name        bob
2          name        fred
2          name        austin
3          name        fred
3          name        sam
3          name        mac
4          name        lilly
4          name        sam
4          name        mac
```

Running the report will indicate which people interacted with whom, both by group and overall with different sorts of subtotals depending on which option boxes we checked:



Value	1	Count	2	Count	3	Count	4	Count	Total	Total Count
bob	lilly sam Total	1 1 2	fred austin Total	1 1 2					lilly fred sam austin Total	1 1 1 1 4
sam	lilly bob Total	1 1 2			fred mac Total	1 1 2	lilly mac Total	1 1 2	lilly mac bob fred Total	2 2 1 1 6
lilly	sam bob Total	1 1 2					mac sam Total	1 1 2	mac sam bob Total	1 2 1 4
fred			austin bob	1 1	mac sam	1 1			bob mac sam	1 1 1

If the show count box is checked the number of instances of each counted variable are indicated in an adjacent count box, with an additional "Show totals" box checked a total count is presented below, as in the above screen shot. If "Show totals" is checked but not the count, then only the number of different instances is indicated, i.e., with the redundancies removed. This provides a very different "count" of the interactions.

Note: The group variable assumes that each group is defined by a unique name. You may have to play with temporary table columns to achieve this.

TAMS Analyzer 3.31 Release Notes

3.31 Build 4:

Bug fixes:

1. Fixed Applescript set-up so it works with VPedal, a USB foot pedal (www.vpedal.com).
2. Kludged a fix for “Play media” from a results window. There is now a play/stop button. Hopefully Apple will fix things so that sheets will enable NSMovieViews. The media player also works with VPedal (has the same keyboard mappings as the document media player).

TAMS Analyzer 3.31b5 Release Notes

This release provides updated source code that compiles to universal binaries. It also has a compiler flag defined in tams.h (DISSERTATOR) which if #defined compiles to a single user version. Single user releases are identified by su in the disk image name. Multiuser releases are identified by mu in the disk image. Also projects say “Single user version” on the info tab. The buttons that control multiuser sharing of files are rendered invisible: “Synch,” “Manage users”, “Check in” and “Check out”.

Note as of this release date Graphviz has not been released as a universal binary, so those reports that use Graphviz.app may or may not work.

TAMS Analyzer 3.31b8 Release Notes

This is a bug fix release. Repairs include:

1. Check for Pairs from the project has been fixed so that it handles end of files properly (See notes below)
2. With temporary columns copying the _doc column works correctly
3. Save as in results file now works correctly and doesn't crash the machine or lose the renamed file
4. Save as in document file now saves the file correctly the write format. This caused a lot of problems since saveing a file with tags lost most of the tags (they were being interpreted as rich text format commands which also use "{".
5. _doc column now collapses up and down properly

Notes:
Check for pairs has one known bug, which seems to be related to Apple's cocoa system. If the last character of file is a } rather than a return, the brace is invisible to tams, for some reason. This means that TAMS will complain of an unmatched pair. Simply add a return at the end to correct the complaint.

TAMS Analyzer 3.32b4 Release Notes

This release includes the following changes:

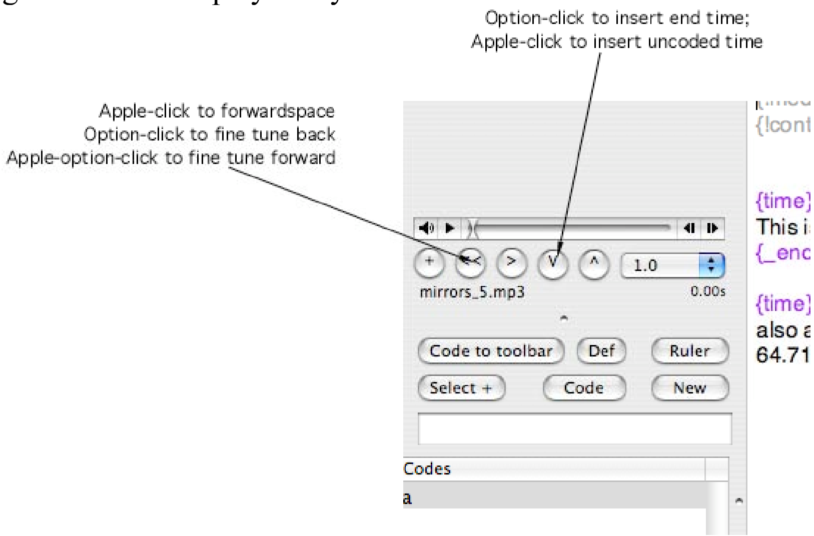
- Code level option for data elements of the Data Comparison table
- Group but don't count duplicates for data elements of the Data Comparison table
- Option to fold some results window dialogues (see program preference) for use on small screens with low resolution
- General interface improvements (especially in Data Comparison Table)
- Bug fixes for tamsadmin log in for multiple users
- Bug fixes for saving the variable space and temporary columns in results files
- Bug fix for setting dirty flag for temporary column operations

TAMS Analyzer 3.33b5 release notes

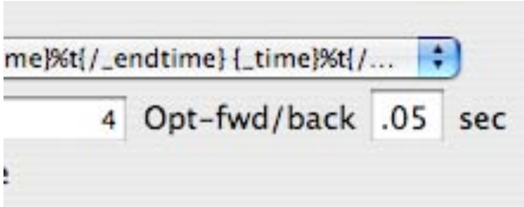
TA 3.33 includes some fine tuning of features in previous releases. These include

- The ability to set end times as well as start times
- The ability to fine tune the position of the scrub bar.

These features all have their details set in the program preference panel. End times are triggered by option-clicking the "v" media player key:


Option-click to insert end time;
Apple-click to insert uncoded time
Apple-click to forwardspace
Option-click to fine tune back
Apple-option-click to fine tune forward
{!cont
{time}
This i
{_enc
{time}
also a
64.71
+
<<
>
V
^
1.0
mirrors_5.mp3
0.00s
Code to toolbar
Def
Ruler
Select +
Code
New
Codes
a

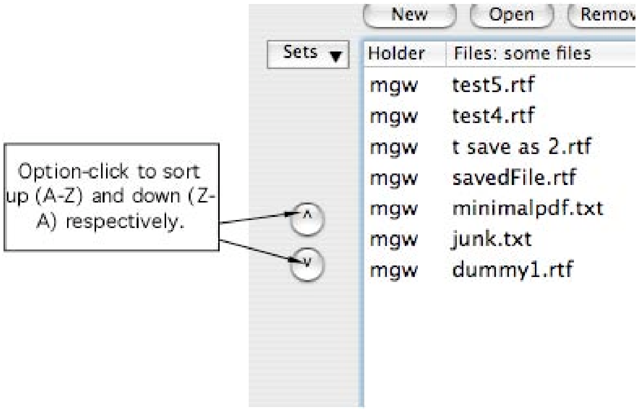
The amount of fine tuning of the player is set in the program preferences:


me}%t{/_endtime} {_time}%t{/...

4 Opt-fwd/back .05 sec

To move backwards option-click the << key. To move forward option-apple click the same (<<) key, also to forward space the bar just apple click the << button.

- The ability to sort the search and file lists on the project window. Just option-click the shuffle up and down buttons.



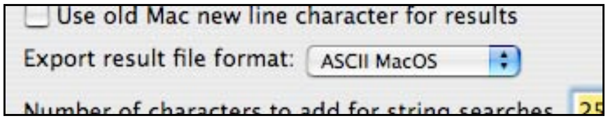
Holder	Files: some files
mgw	test5.rtf
mgw	test4.rtf
mgw	t save as 2.rtf
mgw	savedFile.rtf
mgw	minimalpdf.txt
mgw	junk.txt
mgw	dummy1.rtf

TAMS Analyzer 3.34 Release Notes

TA 3.34:

New Features:

1. text encoding other than ascii now supported (e.g., utf8) for source documents by selecting the file type from the preferences menu:

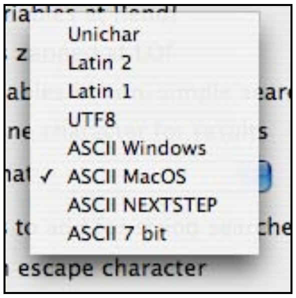


□ Use old Mac new line character for results

Export result file format: ASCII MacOS

Number of characters to add for string searches 25

Note, that the same encoding scheme across all text files in the project. Supported encodings:



Important Fix:

1. The interface has changed so that support for Jaguar has been restored. Basically, my use of setHidden: in the project window and data comparison sandbox has been tweedled so that it becomes a setEnabled: for jaguar users.

TAMS Analyzer 3.34b6 Release Notes

This is a bug fix release. It corrects problems introduced in the release of version 3.33, which added option click modifications to the media player buttons. These modifications, unfortunately, overrode the keyboard shortcuts for the basic operations (Backspace, etc.). These are now restored.

TAMS Analyzer 3.34b7 Release notes

This is a bug fix release. This release solves the following problems:

1. A bug is fixed which accidentally enabled the “New Code” menu item from the coding menu when the workbench is selected. Choosing this item would lock up the program and corrupt the project file. See note below.

2. A number of bugs were fixed that prevented the last code in the code list to be deleted. Also there were problems with using the code browser to delete codes and then switching to the workbench.

Note on the corruption of project files. This is a note to myself as much as others that run into this problem. The project file is corrupted in the following way. Adding the new code saves two arrays of dictionaries with fake items that ultimately prevent it from being reopened. To correct the problem open the project file in a text editor (you may have to temporarily change the extenstion to xml or txt). Find the list of codes. Scroll previous to this and find just prior two arrays of dictionaries. These will have one more object in them than exist in the list of codes. Delete the last entry in both arrays and save.

TAMS Analyzer 3.35b5 Release Notes

TA 3.35 introduces a number of changes to improve useability: a better project opener and a search button on the file tab in the project window; it also now calculates the location and length of data discounting the space taken up in the file by metatags and tags. This is used in select near now for character searches (find records less than 50 **characters** from the current selection. It also means that users can see the length of their data strings (not counting tags) in the result window. This has only been tested on simple searches.

Changes:

1. New Project/Work selector. Working projects are now presented on a table rather than menu
2. *_end_loc* now shown in results table
3. A series of columns: _bare_loc, _bare_end, and _bare_len are now included in results, these keep track of the location and length of the data string indexed and measured as if the file had ntags and metatags.
4. Select near now uses _bare variables rather than _begin loc and _end loc.
5. Search button has been added tthe file tab of projects. This uses the values indicated on the file tab.
6. {!blockcontext} metatag. Once found the program treats all context codes applied tpassages of text as regular codes.
7. File tab of project window now uses square buttons
8. The type of select near drops specific reference to _begin_loc; in fact it uses _bare_loc.

TAMS Analyzer 3.40b14 Release Notes

- New features
    1. Remove work project button on new project browser
    2. Data comparison has a single column option which basically automatically creates a temporary column and groups all the data you want in that column**
    3. Improvements to count column (didn't like the old look); hope it works
    4. Temporary columns now can be created that calculate length, average and count based on regular expressions: see "Analyze" menu option in the Results->Temporary Column submenu**
    5. Temporary columns now can be filled with the row #
    6. Strip tags function for temporary columns
    7. Temporary columns can be created from existing columns in one operation (Add copy of current column); previously users had to create a new column in one operation, and then copy the old column in a second
    8. Group by column**
    9. Toolbar buttons added for Copy to temporary column, Add copy of current column, Regex transform of temporary column, Analyze temporary column, and Group column.

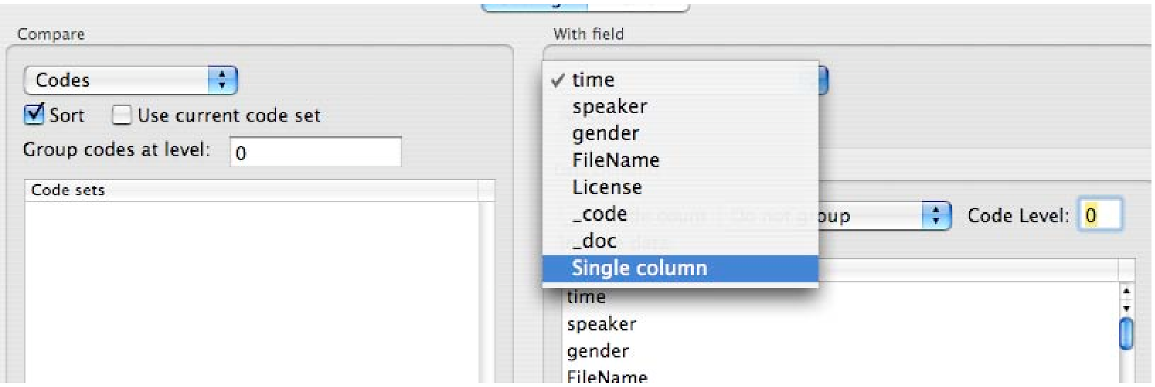
- Bug fixes
    1. Small results browser present in all builds
    2. Small results browser has updates to (I hope) the same as regular results
    3. Work menu and new project browser synchronize
    4. !if tags created from temporary columns now have spaces removed so that they are proper (more or less) variable names
    5. Major fixes in export of outlined data (group by column or code set)
    6. Code definitions not restored when flipping tabs
    7. Fixes to Delete code in project window.

**Details described in the following documentation

Documentation

1. Data comparison in a single column or row

In the Data Comparison table, it is now possible to have the program group all the results in a single column rather than spread them across the values provided by a specific column in the results table. To do so pick "Single column" for the With field:


Compare

Codes

☑ Sort ☐ Use current code set

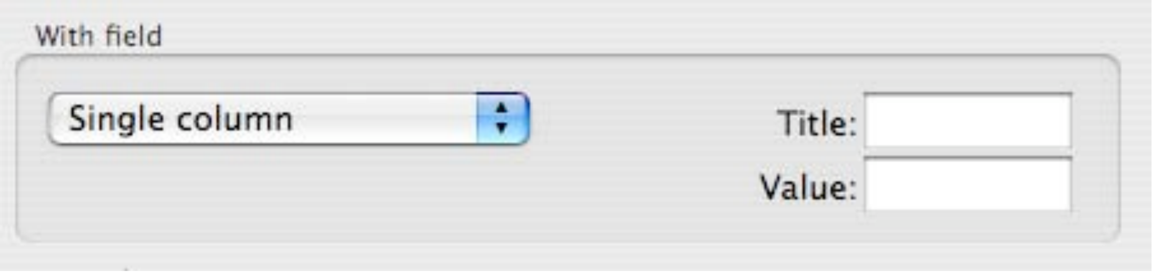
Group codes at level: 0

Code sets

With field

Code Level: 0

This will provide you with fields as follows:


With field

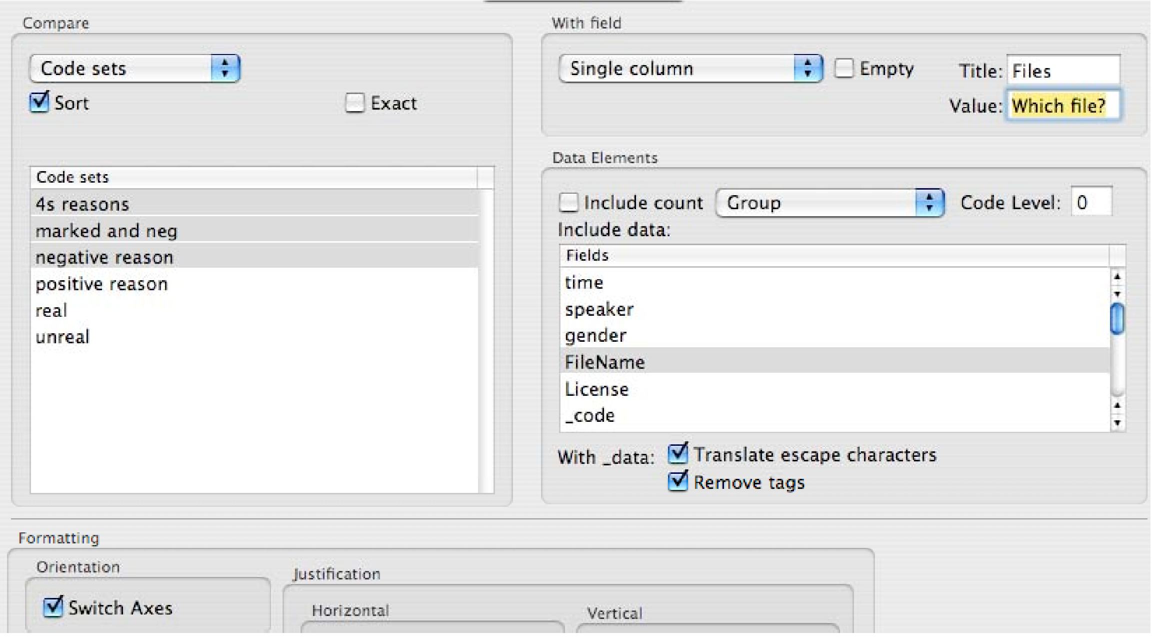
Single column

Title:

Value:

Fill in a title for this comparison and a value that will be used as a column header. **Important note:** Click “Switch axes” to present the data in a column. The ntatural direction will be a single row.

Example:


Compare

Code sets

☑ Sort ☐ Exact

Code sets
4s reasons
marked and neg
negative reason
positive reason
real
unreal

With field

Single column ☐ Empty

Title: Files

Value: Which file?

Data Elements

☐ Include count Group Code Level: 0

Include data:

Fields
time
speaker
gender
FileName
License
_code

With _data: ☑ Translate escape characters

☑ Remove tags

Formatting

Orientation

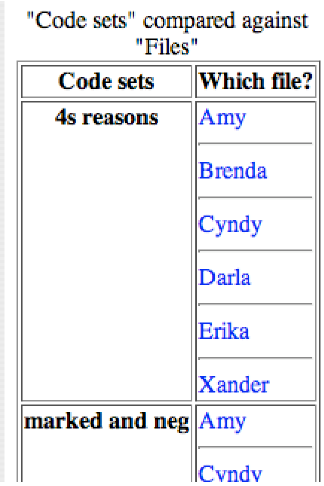
☑ Switch Axes

Justification

Horizontal

Vertical

Results in this table (partial view):



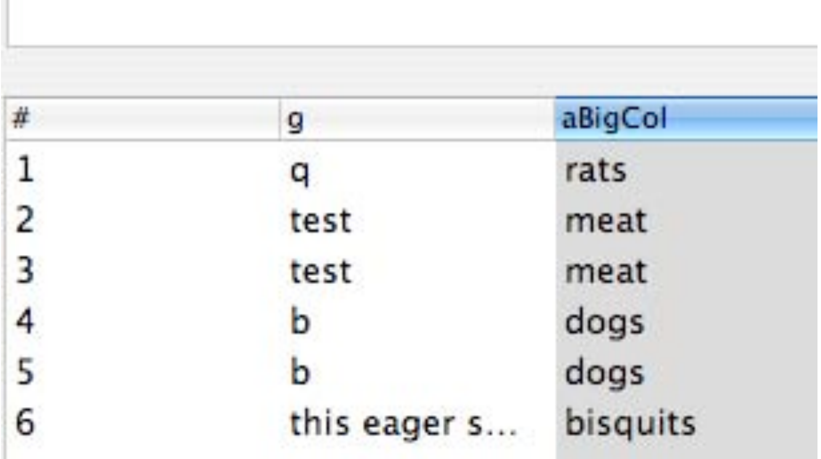
"Code sets" compared against "Files"

Code sets	Which file?
4s reasons	Amy Brenda Cyndy Darla Erika Xander
marked and neg	Amy Cyndy

2. Group column

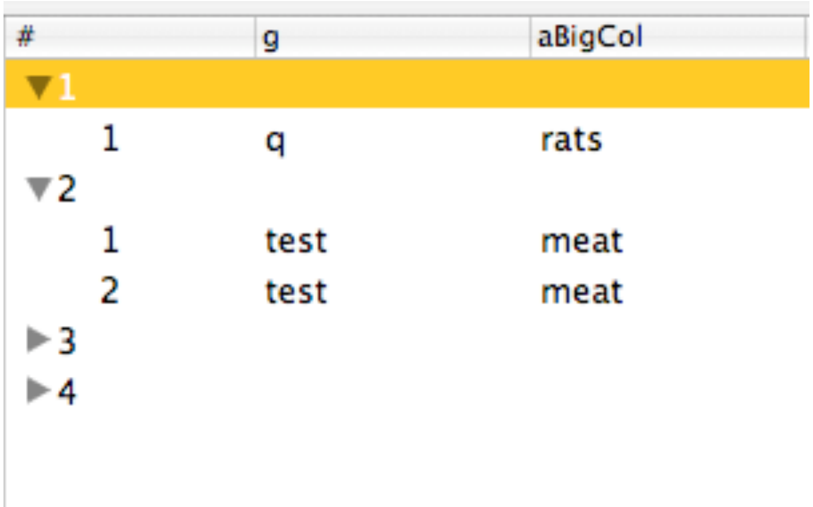
Selecting a column and then picking “Results->Group column” provides a hierarchical view of your data, as the name suggests, grouped by the values in that column. This also provides for single sub-level hierarchical reporting using “File->Export data”

Thus, this data:



#	g	aBigCol
1	q	rats
2	test	meat
3	test	meat
4	b	dogs
5	b	dogs
6	this eager s...	bisquits

Grouped by “aBigCol” (indicated by the fact that it is that column which is selected) would provide an outline view like this (I’ve opened the first two outline-triangles):



#	g	aBigCol
▼1		
1	q	rats
▼2		
1	test	meat
2	test	meat
▶3		
▶4		

The _code column reveals the value grouped on (not shown above) and the _data column provides a count of the records found for each value.

Remember that export data provides a wide variety of options on exporting outlines, when the outline view is showing (lower case “o” in the upper left corner of the results window.

Remember to click the “o” to return to table view (the o will turn into a “t”).

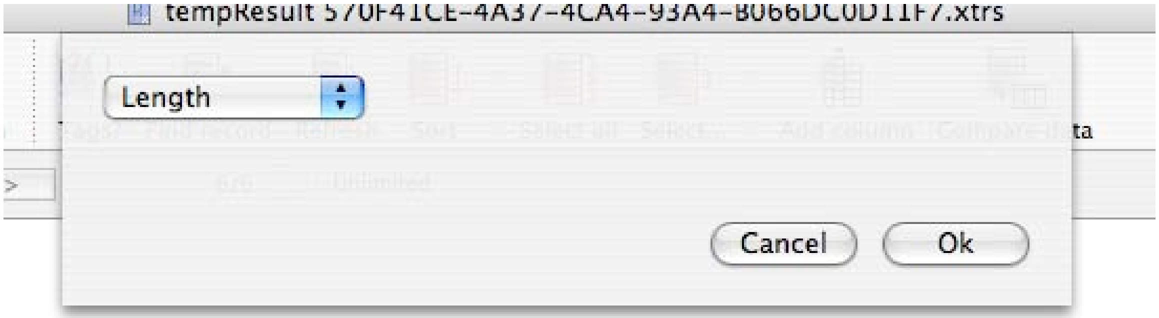
3. Analysis of temporary columns

Selecting a temporary column and picking “Results->Temporary column->Analyze selected rows...” provides a number of numerical analyses of the data in the selected temporary column.

Make sure you are working on a copy of the column as the contents will be replaced by the results of the analyses described below. This is made easier in this version with the new “Results->Temporary column->Add copy of the current column...” command, which asks for a column header and then generates a new temporary column with a copy of the currently selected column.

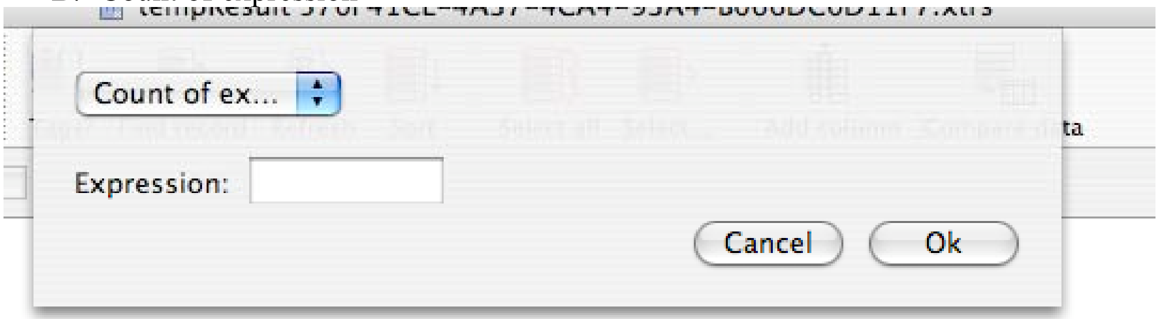
Three analyses are available to you

A. Length:



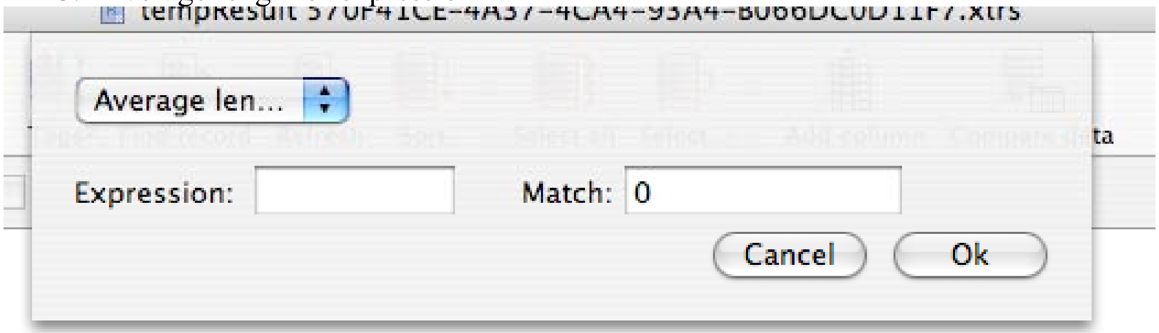
This simply replaces the current column with a measure of the length of the text in each row in the currently selected column.

B. Count of expression



Using regular expressions (filled into the Expression box), this choice will replace the current column with the number of matches of that expression in each row in the current column. This is useful for getting a word count (Expression = “[a-zA-Z]+”) for instance.

C. Average length of expression



Match: 0

Here the program replaces the column contents with a calculation of the average length of patterns. Users can indicate if they wish to examine the whole expression found (Match=0) or a substring of that expression (Match > 0). Refer to a regular expression tutorial to learn about substrings. To find the average word length we would fill in the expression “[A-Za-z]+” and keep Match at 0.

TAMS Analyzer 3.41b4 Release Notes

Leopard compatibility!!!

- New features
  1. Enumerate contents of a temporary column both breaking (resetting) on a new value or continuing to sequence subsequent matching entries.**
  2. Rename temporary columns
  3. The separator for copy column to temporary column now interprets escape characters \t, \n and \r
  4. Remove tag from temporary column now actually available on the Temporary column menu

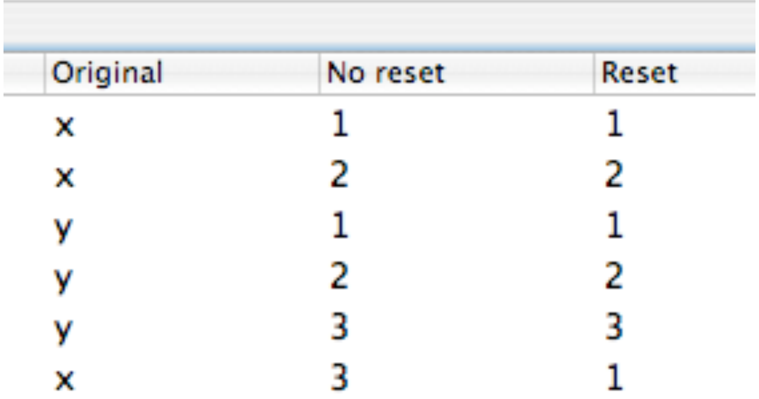
- Bug fixes
  1. Fixed bug for create new file with blank name
  2. _doc can now be used to generate a temporary column in Add column and copy.
  3. Links in data comparison now work with Leopard, Safari 3, and 10.4.11 (which installs safari 3).

**Details described in the following documentation

Documentation

1. Enumeration of values

TAMS Analyzer now can take a temporary column and replace it with an enumeration of its values. Consider this example:



Original	No reset	Reset
x	1	1
x	2	2
y	1	1
y	2	2
y	3	3
x	3	1

Originally, this consisted of three identical columns. Notice that there is an extra “X” in the last row, that’s orphaned from the two X’s at the top. TAMS offers two ways to count this data. The first way keeps counting when it finds matching values, whether those values are contiguous with others or not. That’s the “No reset” column above. The final X is numbered “3” since it is the 3rd X that TAMS finds in the column.

The second way that TAMS enumerates restarts the count every time the value in the column shifts. In this case, the final X is counted as “1” since the immediately previous value is a “Y”. Contiguous values are enumerated starting with each time a patch of such contiguous values are encountered by TAMS as it works its way down the list.

Both enumeration options are found on the “Results->Temporary column” submenu.

TAMS Analyzer 3.42b6 release notes

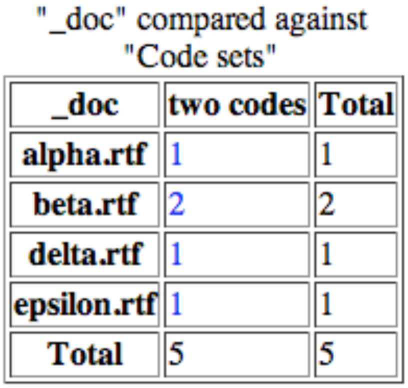
New Features

- Item count in data comparison chart
- Sub code temporary column
- Fast access buttons for data and results files on Files tab
- Updated regex engine
- Updated mysql software (Multiuser version only)
- Small interface improvements
- Builds with help (manual and tutorial) built into the program

Notes:

1. Item count in data comparison chart.

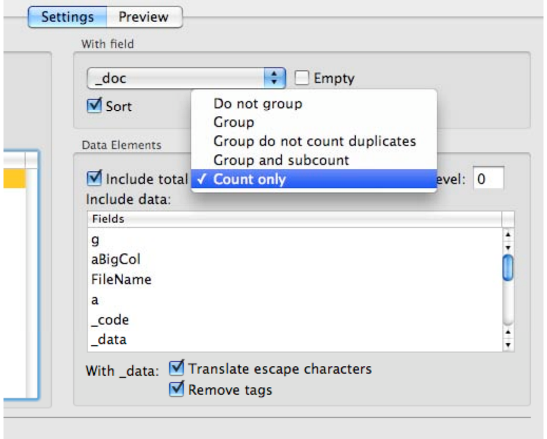
Some time ago TAMS lost the ability to generate raw counts like this in data comparison reports:



"_doc" compared against "Code sets"

_doc	two codes	Total
alpha.rtf	1	1
beta.rtf	2	2
delta.rtf	1	1
epsilon.rtf	1	1
Total	5	5

This version restores this capability. To put counts in your report pick “Count only” from pull down menu in the data elements box:



_doc □ Empty

☑ Sort

☑ Include total ✓ Count only evel: 0

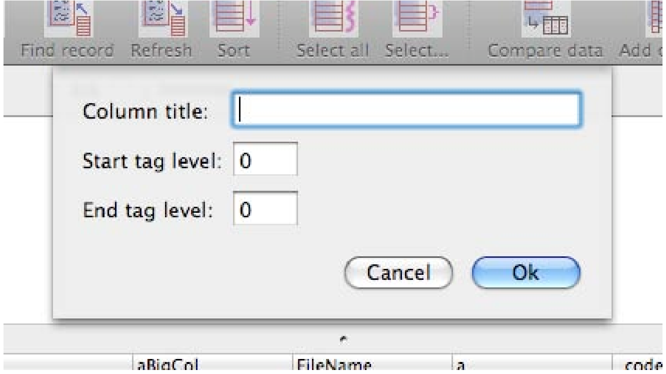
Include data: g, aBigCol, FileName, a, _code, _data

With _data: ☑ Translate escape characters

☑ Remove tags

2. Subcode temporary column

I have added the ability to create a column of “subcodes.” Subcodes simply means a part of a code. Given code a>b>c subcodes might include a>b or b>c or a or b or c. Previously this could be done with tricky regex coding, I’ve just made it easy. Pick “Temporary column->Add column with subcode...” That will give you this dialogue:



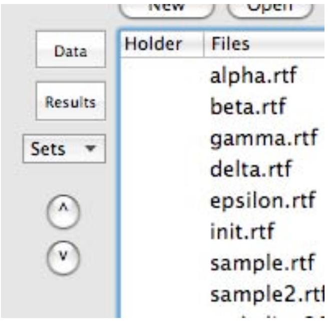
Column title:
Start tag level: 0
End tag level: 0

Fill in a unique name for the column. Then put in the start and end levels you want: for a>b>c putting in a start tag level of 1 and end tag level of 2 would give you a>b. Putting in 1 and 1 respectively would give you a. Putting in 3 and 9999 (or some other large number) will give you c for this case but for a>b>c>d would give you c>d.

You can also use negative numbers to put in the end of codes. so putting in -1 -1 would give you the lowest level of code for each row. -1 and -2 would give the last 2 for each row.

3. Fast access buttons for data and results files.

The biggest issue I have to deal with supporting TAMS is people opening their results files from the desktop rather than from within TAMS. Right now getting to the results files is hidden in a little pull down menu on the Files tab. To make things more transparent I’ve added two buttons which give you instant access to your Results and data files:



Holder	Files
	alpha.rtf
	beta.rtf
	gamma.rtf
	delta.rtf
	epsilon.rtf
	init.rtf
	sample.rtf
	sample2.rtf

TAMS Analyzer 3.43b3 Release Notes

This release adds powerful control over organizing and formatting dates. It also fixes two bugs: data comparison tables sometimes didn't print the both total columns when "include total" was checked; saved result windows didn't rebuild the named code set menu right away. Finally it includes a normal search/replace for temporary columns. The old regular expression transformation is still there, just click the regex check box, other wise the menu item formerly called Results->Temporary column->Transform column—and now called Results->Temporary column->Find/Replace/Transform column—will do a straight forward find/replace through the temporary column that's selected. Note, that even in regular find and replace you can use \n and \r for return characteres and \t for tabs. Finally, I've worked on updating the documentation. I've also broken up the user guide into separate documents for the purposes of the help function. There are two new sections in the updated documentation: one on temporary columns, the other on data comparison tables.

New features:

- In result windows, each column can have its own date/time format for sorting and grouping purposes.
- Date formats can be set automatically through the {!dateformat} metatag.
- Dates can be transformed and grouped by day, month, and year.
- Find/Replace in temporary data columns

Bug fixes

- Named sets menu wasn't updated when a saved result file was loaded
- Data comparison tables didn't print out both row and column totals when "include totals" was checked.
- Improvements to sorting by date/time
- For date format dialog, the field for specifying the date format is hidden unless other is specified from the menu.

Other

- Updated User guide and Tutorials

Documentation

1. Setting the date format

Researchers can now set a date format for a specific variable. This is done through a metatag {!dateformat} which identifies a variable (e.g., a context variable), and the string format for the date. For setting string formats see

http://developer.apple.com/documentation/Cocoa/Conceptual/DataFormatting/Articles/dfDateFormatterSyntax.html

Typical use will be setting date formats after declaring the variable:

```
{!context publicationDate}
{!dateformat publicationDate="%m/%d/%Y"}
```

Note that these must be straight quotes. This format says that the date is formatted as mm/dd/yyyy. You are telling TA what dates look like for the purposes of sorting the column associated with the context variable publicationDate and for grouping which will be explained below.
Alternatively, researchers can set the format associated with a date in the results window. If no column is selected, picking "Results->Sort options->Date format..." will set the default format for dates (very useful). If a column is selected, the date format will apply to that column and any columns created through "Results->Temporary column->Add copy of current column..." If you do not specify a format, the column will be sorted using the default date value.

2. Grouping and transforming dates

First, **this only applies to temporary columns. The changes are made directly to the column that is selected.**

Transforming these dates is primarily a way to view your data by date/year/month range for Data comparison reports. It also can be a way to transform date formats (changing mm/dd/yyyy to yyyy-mm-dd, for instance)

Select a column containing date data. As of this release this will group and transform dates by year, day, and month. At some point in the future, similar support for time may be added.

Make sure that you have specified the current format for the date strings using "Results->Sort options->Date format..." (see note 1 above).

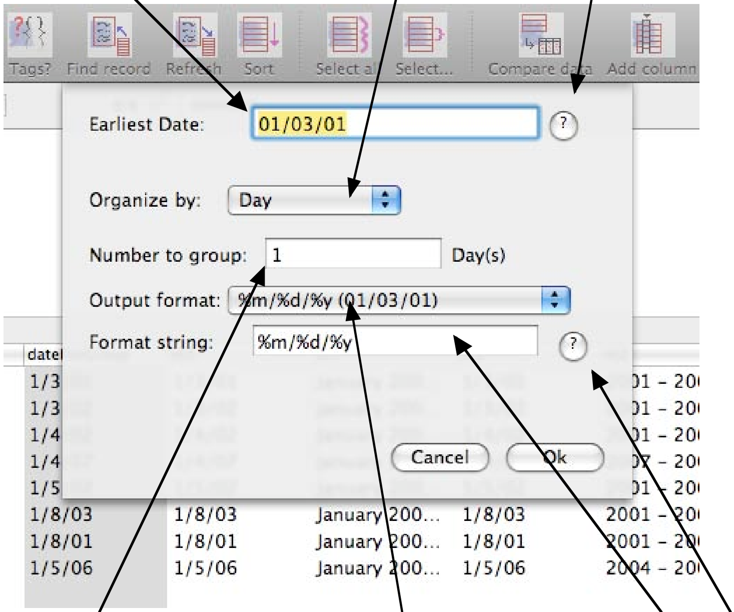
Pick "Results->Temporary column->Group dates..."

If you have specified your date format correctly, you should see the following dialog:

1. This will show the earliest date in the selected column. You can alter this date to control how date groups are defined (e.g., forcing the groups to start on a sunday or a later date)

2. The ? button here will show the day of the week represented by this earliest date. Useful for adjusting for ranges of dates

3. You can transform this column into one that just specifies month, year, or ranges of days, months or years. Pick the scale you want from here. It will change the contents of the output format menu.



Earliest Date: 01/03/01

Organize by: Day

Number to group: 1 Day(s)

Output format: %m/%d/%y (01/03/01)

Format string: %m/%d/%y

date
1/8/03 1/8/03 January 200… 1/8/03 2001 - 20
1/8/01 1/8/01 January 200… 1/8/01 2001 - 20
1/5/06 1/5/06 January 200… 1/5/06 2004 - 20

4. Indicate what range you want to group things within. If you want to group by week, you will pick Days from menu 3, and indicate "7" in this field. If you want to group by quarter, you will pick month from menu 3 and 3 (3 months in a quarter) here. If you want to group by decade, pick years from menu 3 and fill in 10 here.

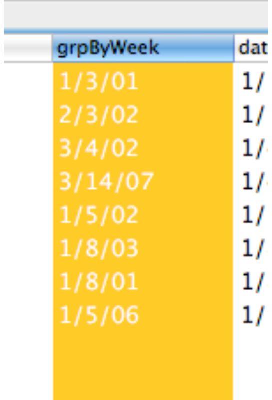
5. Pick how you want dates formatted after they have been grouped/transformed. This will also set a new date format for the column. Notice that the menu shows both the format string and an example of how it will look.

6. Menu 5 just puts the format string into this field. You can then modify it, if you like. Clicking the ? button will show you how your modifications will look. This field is then used for the formatting of the modified dates, and as the date format for this column.

Examples:

1. Grouping data by week

A. Make a temporary column of your dates called grpByWeek; make sure the date format is set for this column. E.g. your initial data may look like this:

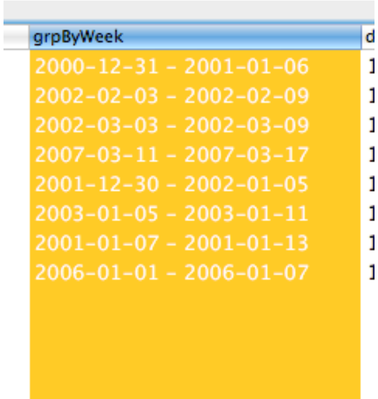


grpByWeek	dat
1/3/01	1/
2/3/02	1/
3/4/02	1/
3/14/07	1/
1/5/02	1/
1/8/03	1/
1/8/01	1/
1/5/06	1/

B. Pick "Results->Temporary column->Group dates"

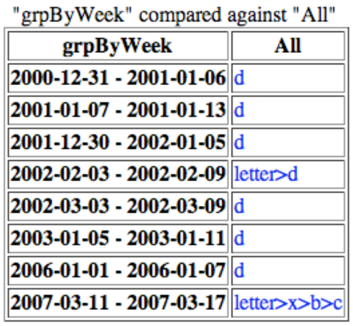
C. In the dialogue, adjust the start date to a Sunday (use the ? button to check the day of the week. Pick Days from menu 3 and fill in 7 for "Number to group."

D. Pick, just as an example, "%Y-%m-%d" from menu 5, and click "Ok." The data should now look like:



grpByWeek	d
2000-12-31 - 2001-01-06	1
2002-02-03 - 2002-02-09	1
2002-03-03 - 2002-03-09	1
2007-03-11 - 2007-03-17	1
2001-12-30 - 2002-01-05	1
2003-01-05 - 2003-01-11	1
2001-01-07 - 2001-01-13	1
2006-01-01 - 2006-01-07	1

Now we can see what codes are used by week through a data comparison table:



"grpByWeek" compared against "All"

grpByWeek	All
2000-12-31 - 2001-01-06	d
2001-01-07 - 2001-01-13	d
2001-12-30 - 2002-01-05	d
2002-02-03 - 2002-02-09	letter>d
2002-03-03 - 2002-03-09	d
2003-01-05 - 2003-01-11	d
2006-01-01 - 2006-01-07	d
2007-03-11 - 2007-03-17	letter>x>b>c

Similar procedures could be used for grouping by quarter or by decade or other range of dates.

2. Transforming dates

To change dates rather than group them make sure you have set the date format for your data as it now stands, pick "Results->Temporary column->Group dates..." and then simply

A. Pick day from menu 3

B. Set the "Number to group" field to "1"

C. Pick the new format for your dates from menu 5.

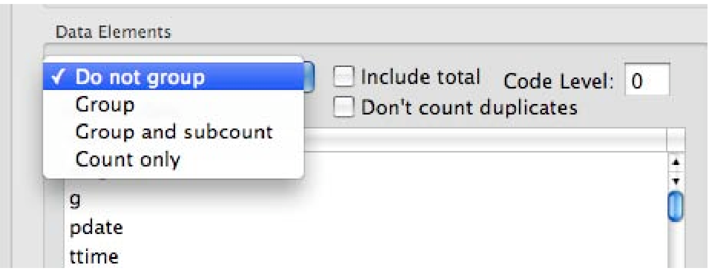
TAMS Analyzer 3.44b4 Release Notes

This release adds a variety of real usability improvements to data comparison tables. These include:

1. You now have the ability to set your own titles/captions for a data comparison table
2. Subtotals and totals are now hot linked to the data
3. The default selection technique for hot linking in reports is now a new mode called “smart highlight/selection” which works as follows: if you click on an item linked to a single row of data, the program will try to highlight it in the current selection; in all other cases it will select the related records.
4. “Do not count duplicates” is a check option/item, so it can be used with all types of data comparison tables including “count only” See documentation below
5. Clicking on a hot linked item with “Do not count duplicates” checked will select all the relevant records: it’s as though you had run your report with “Group” picked from the Data Elements menu.
6. The comparison data dialogue will optionally (on by default) restore the current selection, which means you can click on different data items, and then generate another report. This is zeroe’d out if you pull up data comparison report from the Report menu or the icon for data comparison reports. The selection is retained if you use the Window menu or click on the data comparison dialogue.
7. Data comparison table button is now part of the Results window’s default tool bar button set. Note that this will not effect current users, only new users. Current users will need to go to “Configure tool bar” on the Windows menu to change their tool bars.
8. Small interface tweaks to the Data Comparison Dialogue

Documentation

Do not count duplicates check box

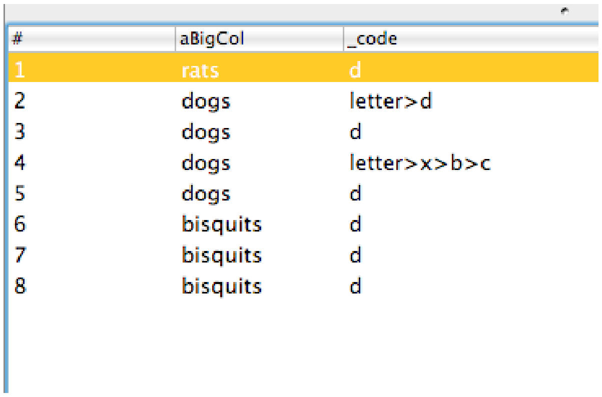


☐ Include total
☐ Don't count duplicates
Code Level: 0

On the data comparison dialogue, in the Data elements box, the old menu option “Group do not count duplicates” has been moved to a check box to the right of the menu. Include total has also been moved to the right.

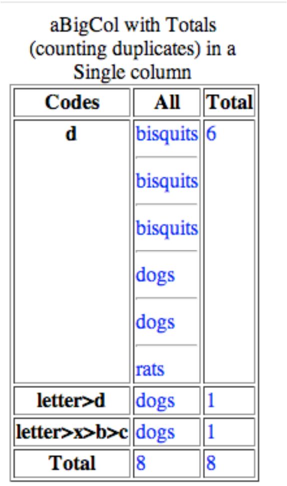
Checking “Don’t count duplicates” primarily effects the totals if “Include total” is checked. For “Do not group”, “Group”, and “Group and subcount” menu items, that is the only thing effected by the “Don’t count duplicates” option. If you don’t “include total” column/rows, not counting duplicates does nothing (visible).

To clarify the effect consider the following phony data:



#	aBigCol	_code
1	rats	d
2	dogs	letter>d
3	dogs	d
4	dogs	letter>x>b>c
5	dogs	d
6	bisquits	d
7	bisquits	d
8	bisquits	d

If you were to use the data comparison table to look at the contents of aBigCol grouped by code You would get the following (I'm comparing _code to a single column), including total but leaving the "Do not count duplicates" box unchecked:

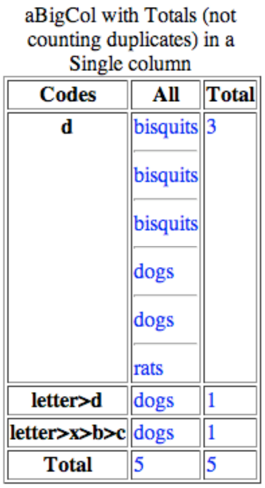


aBigCol with Totals (counting duplicates) in a Single column

Codes	All	Total
d	bisquits bisquits bisquits dogs dogs rats	6
letter>d	dogs	1
letter>x>b>c	dogs	1
Total	8	8

So for code "d" there are 6 different rows, and we see the values of each, and the Total is 6: the number of rows that d has regardless of the value of aBigCol.

Now this is the same report with the "Do not count duplicates" checked:

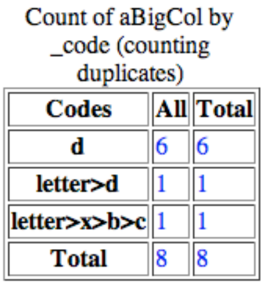


aBigCol with Totals (not counting duplicates) in a Single column

Codes	All	Total
d	bisquits bisquits bisquits dogs dogs rats	3
letter>d	dogs	1
letter>x>b>c	dogs	1
Total	5	5

Notice that now the for code “d” is 3. That’s because there are 3 *different* values for aBigCol in the row (bisquits, dogs, rats: 3 different values). Clicking the 3 in the total column will still pull up all 6 records, but it only counts each different value one time.

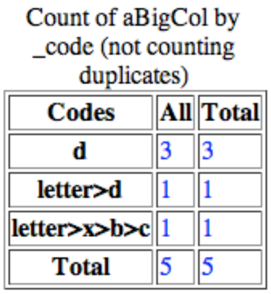
The one menu option that is really effected by the new check box is “Count only” since it will apply this logic to each intersection of the data comparison table (and then total those in the Total column). In other words, picking “Count only” from the menu and *not* selecting “Do not count duplicates” produces the following in our example:



Count of aBigCol by _code (counting duplicates)

Codes	All	Total
d	6	6
letter>d	1	1
letter>x>b>c	1	1
Total	8	8

But picking “Do not count duplicates” ignores the redunancy in values (“dog” will only be counted once:



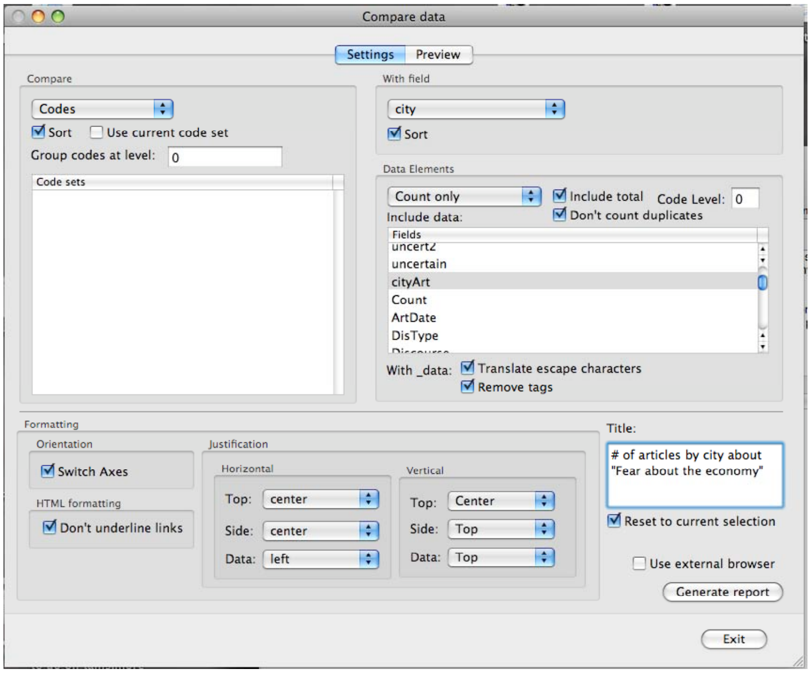
Count of aBigCol by _code (not counting duplicates)

Codes	All	Total
d	3	3
letter>d	1	1
letter>x>b>c	1	1
Total	5	5

Real world example

In looking at the discourse of the anthrax attacks, the data is all grouped in articles. Often I’m not interested in how many occurrences there are of a code like fear>about>economy, for instance, but how many articles talk about fear>about>economy. One article may talk about it only one time, or dozens of times, I don't care, since the article is a more organic unit than the actual coded instance. Previously I could accomplish this by selecting the rows that contained "fear>about>economy", sorting them by article, and then collapsing up or down to get rid of the redundancy, and then doing a data comparison table. Now all I need to do is select for "fear>about>economy" and do a data comparison table (_code by "Single column" or my preference is to see it by city, and article selected in the data elements field table) and make sure "Do not count duplicates" is checked.

The Data comparison options I would use:



Compare data

Settings | Preview

Compare: Codes
☑ Sort ☐ Use current code set
Group codes at level: 0
Code sets

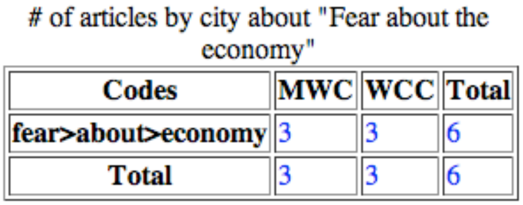
With field: city
☑ Sort

Data Elements: Count only
☑ Include total
Code Level: 0
☑ Don't count duplicates
Include data:
Fields: uncert2, uncertain, cityArt, Count, ArtDate, DisType, Discourse
With _data: ☑ Translate escape characters
☑ Remove tags

Formatting
Orientation: ☑ Switch Axes
HTML formatting: ☑ Don't underline links
Justification
Horizontal — Top: center; Side: center; Data: left
Vertical — Top: Center; Side: Top; Data: Top

Title: # of articles by city about "Fear about the economy"
☑ Reset to current selection
☐ Use external browser

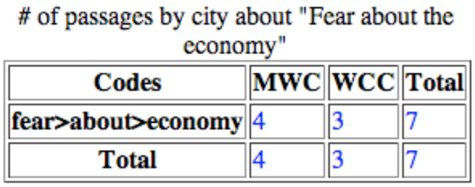
You can see I've highlighted codes in the "Compare box", "city" in the "With field" box. I'm counting "cityArt" (Article number with city name attached—the column which uniquely identifies the article each coded passage came from) without duplicates:



# of articles by city about "Fear about the economy"

Codes	MWC	WCC	Total
fear>about>economy	3	3	6
Total	3	3	6

Voila! There are 6 such articles; 3 from each city. Clicking on any of these numbers will show me all the relevant rows. How many raw rows are there? I'd uncheck the "Do not count duplicates" and find there are 7:



# of passages by city about "Fear about the economy"

Codes	MWC	WCC	Total
fear>about>economy	4	3	7
Total	4	3	7

TAMS Analyzer 3.44b7 Release notes

This is a bug fix release. Twiddling on Apple’s part made it impossible to import text files; this required counter twiddling on my part. The code definition/code set definition dialogue is now interactive: you don’t need a refresh button. Other small changes asked for by users have also been incorporated in this release.

TAMS Analyzer 3.44b10 Release Notes

This version of TA includes the following minor changes

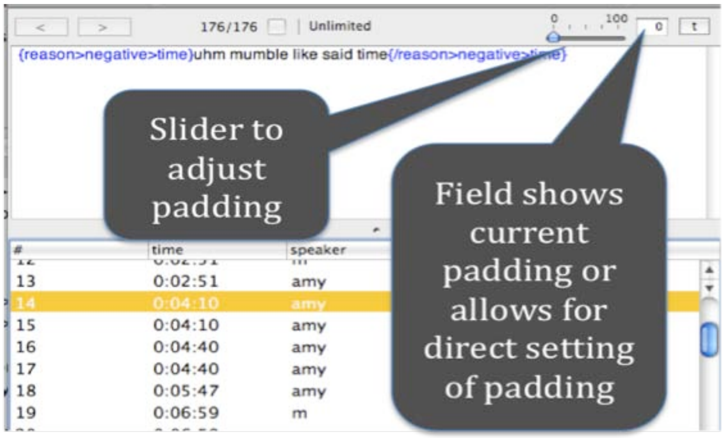
- It restores the ability to rebuild the code list by having the program search through the Search List files and add all codes it finds.
- It extends Data Comparison tables to non-simple searches. All results can now use the Data Comparison table, but only simple searches can let you use code sets as one of the table's dimensions. (note that you can not use code sets in these other searches because the _code field doesn't contain a real code, typically it contains the search string).

TAMS Analyzer 3.45b3 Release Notes

New Feature:
Control amount of “context” (padding) surrounding the text you are browsing. (Warning: on slower machines this could affect performance)

Context slider: To see more of your source document in the results browser adjust a slider or fill in the a textfield to precisely adjust the padding around the passage in question:

At the top of the results window is now a slider and connected text field



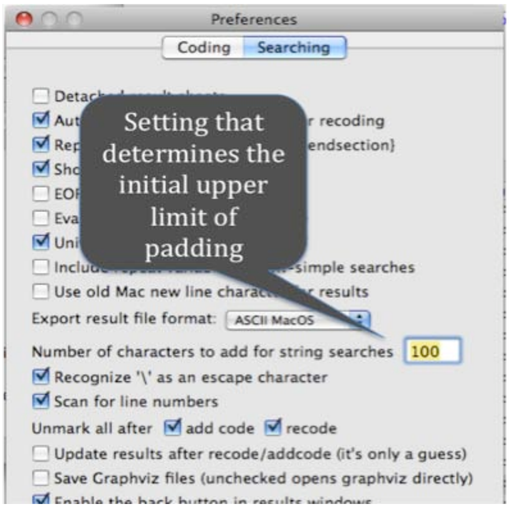
<
>
176/176
Unlimited
0
100
0
t
Slider to adjust padding
Field shows current padding or allows for direct setting of padding

{reason>negative>time}uhm mumble like said time{/reason>negative>time}

#	time	speaker
13	0:02:51	amy
14	0:04:10	amy
15	0:04:10	amy
16	0:04:40	amy
17	0:04:40	amy
18	0:05:47	amy
19	0:06:59	m

Figure 1: Zero Padding

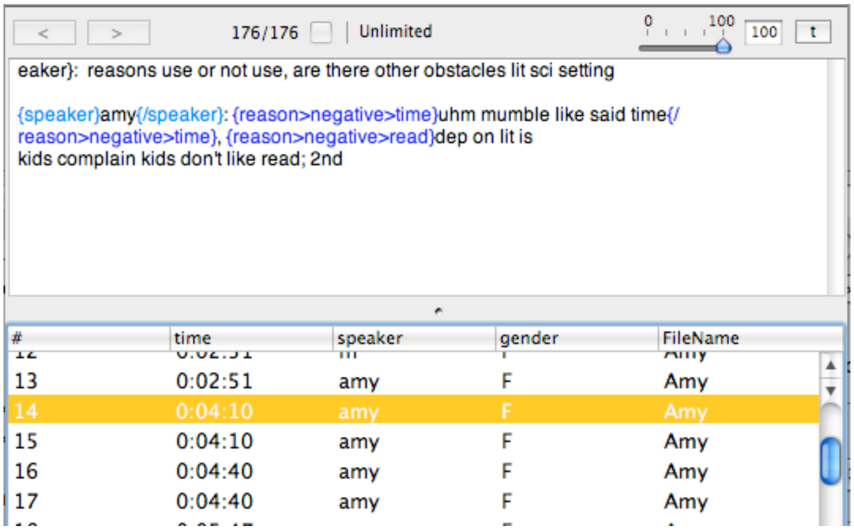
Note that the upper limit is initially set by your program preferences:



□ Detac
☑ Aut r recoding
☑ Rep endsection}
☑ Sho
□ EOF
□ Eva
☑ Uni
□ Inclu -simple searches
□ Use old Mac new line charac r results
Export result file format: ASCII MacOS
Number of characters to add for string searches 100
☑ Recognize '\' as an escape character
☑ Scan for line numbers
Unmark all after ☑ add code ☑ recode
□ Update results after recode/addcode (it's only a guess)
□ Save Graphviz files (unchecked opens graphviz directly)
☑ Enable the back button in results windows

Figure 2: Preference setting for upper limit of padding

Moving the slider will include more of your document in the browser window; here is same data viewed with 50 characters (added to each side) padding:



eaker}: reasons use or not use, are there other obstacles lit sci setting

{speaker}amy{/speaker}: {reason>negative>time}uhm mumble like said time{/
reason>negative>time}, {reason>negative>read}dep on lit is
kids complain kids don't like read; 2nd

#	time	speaker	gender	FileName
12	0:02:51	m	F	Amy
13	0:02:51	amy	F	Amy
14	0:04:10	amy	F	Amy
15	0:04:10	amy	F	Amy
16	0:04:40	amy	F	Amy
17	0:04:40	amy	F	Amy

Figure 3: Same data, padding = 100

By using the text field next to the slider you can directly adjust the padding. You can enter values larger than maximum value of the slider (and the slider will readjust its maximum), or by entering a negative number you can directly set the maximum (e.g. entering -300 will set a maximum padding of 300). None of this will change the program default. However, if you go to your preferences and change the default it will affect the padding on all open results windows.

TAMS Analyzer 3.51b9 Release Notes

Version 3.51
New Features

- ability to set codes for a section of a document through a metatag (See documentation below)
- ability to drag codes and place their name in a document (to support first item)
- menu item to duplicate #2
- Ability to rename files (See documentation below)
- Updated code set tab (See documentation below)
- New file set tab which matches the code set tab

Bug Fixes

- Fixed serious bug for !last
- Fixed bug that caused crash when changing named search lists
- Fixed bug that caused crash when grouping _doc column
- Fixed bug that required users to hit the "last hot code set" button twice to pop the last code set off the stack.

Version 3.50
This release adds the ability to create links to other documents. This is done using the already existing bookmark system. Bookmarks can now act like "anchors" in html. You put a bookmark where you want to jump to. Then in another document (or the same document) you put a "goto" metatag. Bookmarks used in this way are called references.

New Features

- Ability to insert and jump to references (See documentation below)

Bug fixes

- Safeguards added to prevent importing files multiple times
- Improved handling of end tags in the padding controls
- Updated documentation

Documentation

1. References

In qualitative research it is often helpful to refer to other documents. In creating memos, for instance you may want to go to a specific passage to see concretely what you are talking about. TA uses the bookmark feature to implement a system for referring to other places in your database. The basic scheme involves putting a bookmark at the site of interest, and then putting a !goto metatag where you want to refer to that bookmark. A bookmark in this context I will also call a "reference."

Bookmarks have a format like the following:

```
	{!bookmark My first bookmark}
```

"My first bookmark" is the title or name of the bookmark. You would put a tag (with the appropriate name) somewhere in your document that you want to refer to. Make sure your bookmark names are unique, or TAMS will refer you only to the first bookmark with that name.

In your memo, you will want to add a link to that bookmark. The link is a metatag with the following structure

```
	{!goto file="myfile.rtf" bookmark="My first bookmark"}
```

The file has to have the value of the full and unique name of the file containing the bookmark, the bookmark has to have the value of the unique name of the bookmark in that file.

To use the link, simply click anywhere in the !goto metatag, and pick "Coding->References->Go to reference". TA will open the file and move you to the bookmark.

Rather than typing in the !goto yourself, it's easier to have TA do it for you. Go to the bookmark you are interested in and click inside of it. Then pick "Coding->References->Remember reference". TA will put together a !goto for you and put it on a special clipboard. Then go to where you want the link, and pick "Coding->References->Insert reference link" and an appropriate !goto will be inserted at the cursor position.

All three reference functions are available as toolbar buttons. The "goto reference" is particularly useful for jumping with a click to the !bookmark location. To permanently add these functions to your tool bar you will need to use the !button metatag followed by the following, in some order, cmd::goToReference, cmd::rememberReference, or cmd::insertReferenceLink in your init file, or at the top of document file.

2. Section coding

TA 3.51b4 introduces a new way to code sections of text. This is useful for the following situations:

1. Tagging memos. Often in a memo, you want to just provide a list of tags that can act as codes for the entire memo.
2. Coding highly structured data. Often projects don't involve coding portions of a text but entire objects. Think of creating cards with items you are coding. Each card represents one something, and you want to code the cards, not the parts of the something on the cards. Again, you are basically tagging the cards.

To accomplish this two new metatags are added to TA's vocabulary. The simple version is !setcode. The format is

```
{!setcode code1, code2, code3}
```

It will code the entire section starting at the point of the tag with code1, code2, and code3, which should **NOT** be a context code. These will be data codes applied to the sections. For example, a memo file might look like the following:

```
{!context date}
{!last date}

{date}1/12/2009{/date}
{!setcode a, b, c}

This is the text of my first memo.

{date}2/1/2009{/date}
{!setcode b, d}

This is the text of my second memo.
```

This is basically the same as

```
{!context date}
{!last date}

{date}1/12/2009{/date}
{a}{b}{c}

This is the text of my first memo.
{/a}{/b}{/c}
{date}2/1/2009{/date}
{b}{d}

This is the text of my second memo.
{/b}{/d}
```

Obviously, the first is easier to read, and serves as a simple way to apply a list of tags to your memos, and include them in your results windows.

!setcode takes its horizon (the point at which the codes expire) from the same horizon as !last and !inner. By default it is !endsections, but it can be changed through the !virtualend or !virtualendsection. There is no "coder" added to these codes.

To allow more control, there is a second tag !setcodeinfo which takes the form:

```
{!setcodeinfo codes="code1, code2, code3"
  coder="myinitals" horizon="end or endsection"}
```

Here you can control both the coder's initials (which will allow you to have multiple people tag items) and set the end point for the tags (either end or endsection, not both!)

There are limits to coding this way. First, addcode, recode, and all the other functions in the Results->Recode menu will not work. Also, setting codes in tags opens up all sorts of possibilities for mayhem, especially nested codes. There's no way of having TA pre-check for nesting in this case, but you will get lots of error messages if the codes and horizons land up conflicting! There's also the very dangerous possibility of using a context code in one of the tags which will lead to at best strange behavior and at worst crashes!!! I've tried to anticipate problems as best I can, but my experience is that there's much I simply can't.

To assist you in adding codes to your !setcode tag, you can now drag and drop codes from the code list in document windows and the name of the code will be inserted in the text. There is also a menu item which does this: Coding->Insert code name, which will take whatever code is selected and insert it at the cursor. This way you can use your code list as the source of your tags.

3. Renaming files

Several people have asked for a rename function, and this release has at least a first attempt at one. Because TA micromanages things even this simple function can get tricky. But the operation is simple enough: select one file from your file list and pick Project->Rename file.

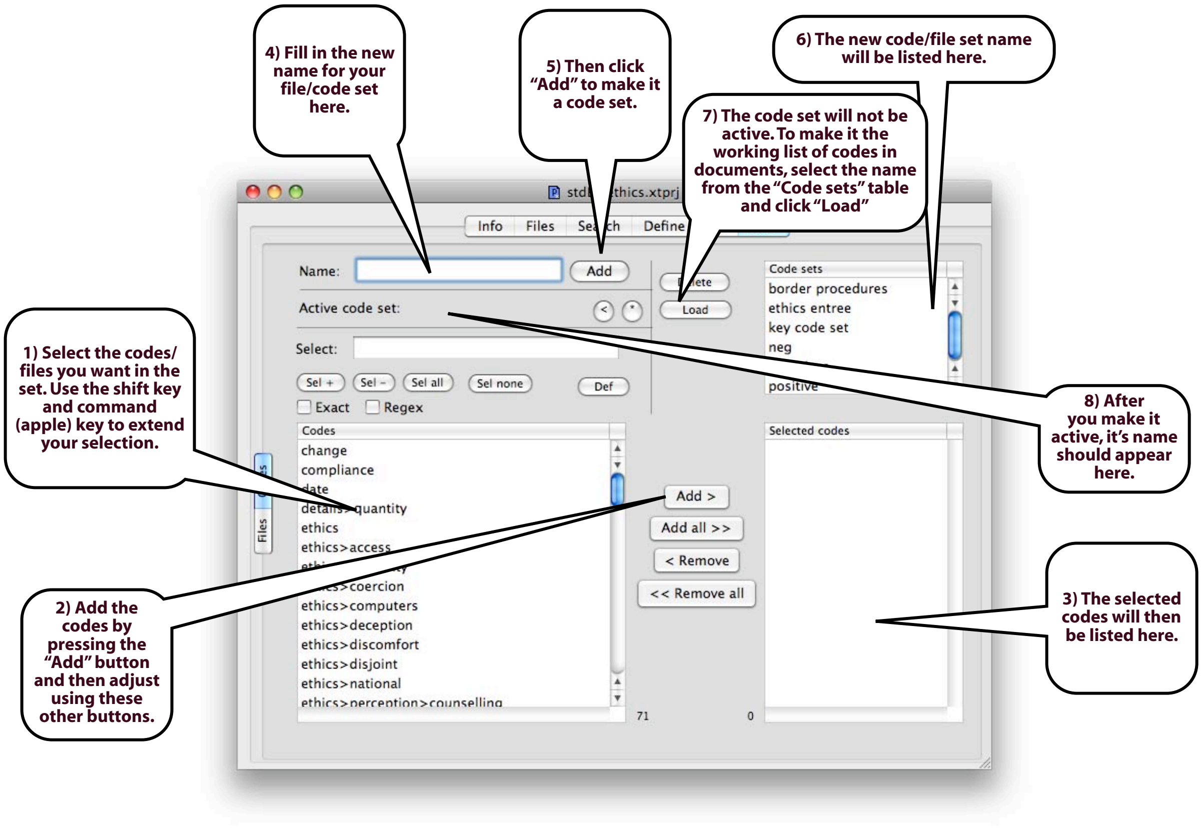
There are limits to what changes when you rename a file, and renaming isn't really recommended. Here are a small number of things to consider:

1. You can't rename any open files; TA will give you a warning
2. Renaming will not change any tags (like !name) in the file; so your data will be out of synch in that sense
3. In results windows, the _doc column will change, but FileName (which is created from the !name tag) will not.
4. In results windows, if you group the _doc column, the level 1 headings wont change when you rename the file, but the level 2 _doc column will properly update.

Don't say I didn't warn you. TA creates a pretty entangled web. But you asked for it; you got it!

5. *Code set tab*

The tab that used to be called "Code Sets" is now just called "Sets" and allows you to adjust both file and code sets. To pick whether you want to work with code or file sets, click the tab on the left side of the window (marked files or codes). The operations are very similar to using the file tab to put together a search list. The basic operation involves following the steps shown in this diagram.



1) Select the codes/ files you want in the set. Use the shift key and command (apple) key to extend your selection.

2) Add the codes by pressing the "Add" button and then adjust using these other buttons.

3) The selected codes will then be listed here.

4) Fill in the new name for your file/code set here.

5) Then click "Add" to make it a code set.

6) The new code/file set name will be listed here.

7) The code set will not be active. To make it the working list of codes in documents, select the name from the "Code sets" table and click "Load"

8) After you make it active, it's name should appear here.

Being able to load the code set directly from this pane is one new feature of the pane. There won't be much evidence of it in this window (just a change in the name by the "Active code set" sign; blank means all codes are showing), but document windows will only show the given code set OR the Files tab will only show the selected file set.

In addition to just picking the names with the mouse, you can use the select field and buttons, just like before.

To rename a code/file set double-click the name from the Code sets table and directly change the name.

Next to the name of the active code/file set there are two buttons. The button labeled "<" will move back to the last active code set. If the code set name is blank, it means all codes are active (i.e., your document window will show all your codes; the file tab will show all the files). To directly load all the codes click the * button.

Changes to 3.52b15

New features

- Code sets and file sets can now auto-update through a preference item (See Documentation Below)
- Hot list button for define codes
- Import !setcode (and !setcodeinfo) into the code dictionary with automatic hot set creation and editing of definitions
- Paragraph and Sentence searches (See Documentation Below
- Advanced Boolean options in paragraph and sentence search
- Exact case match in paragraph and sentence search
- Ability to modify characters used as the criteria for sentence ends.

Interface changes

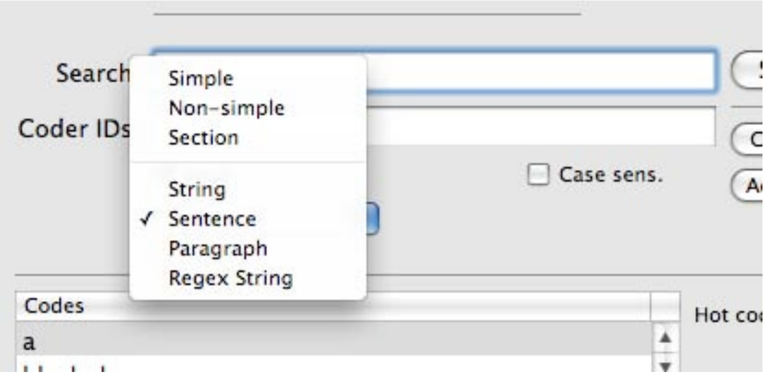
- Warning for delete code
- Interface made parallel for both the little and big code definition tabs and dialogues
- Rename file now has key equivalent
- Rename button added to file and code sets
- Deleting a single file now has a warning dialogue
- Def button removed from file sets subtab
- Def button better placed in code sets subtab

Bug fixes

- Bug fix for code history, though for single user doesn't do anything
- Bug fix in project.hotcodelist routine, caused fatal crash if the code set menu on code tab was clicked
- Document window scroll problem fixed (right scroll bar would gradually vanish off of window)
- Bug fix for !media when media file isn't there
- Bug fix for delete code not removing code from codesets
- For multiuser build, checks holder before renaming

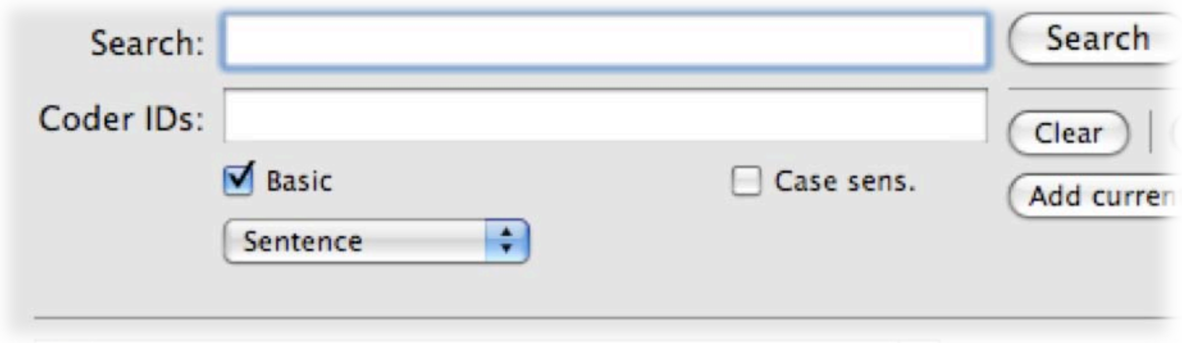
Documentation

1. Sentence and Paragraph searches.



The new search menu

Two new searches are added to the search menus (the one on the project window and the one on the document window): Sentence and Paragraph. Notice that there is now a divider separating code searches from character (aka text) searches.



Options on a sentence or paragraph search

Sentence and paragraph searching: In basic mode (i.e., with the Basic box checked) you can enter a word or expression and TA will return all sentences or paragraphs that contain that expression. Before this version users would have to construct complex regular expression searches to accomplish this task. This expression can include spaces, punctuation, etc. In fact it will need to match exactly what you typed (though there is a case sensitivity option); if you type a single space between words, it will not match two spaces.

In the second mode (Basic box unchecked) treats what you type in as a boolean expression, meaning you can say find all sentences with slime but not hairball. Actually what you would type in is “

```
    slime+!hairball
```

“And” is the plus sign (+); “or” is the comma (,) and not is the exclamation mark (!). Unless you use quotes, TAMS looks for individual words. Spaces are ignored (unlike in the basic search). In other words if you are using the non-basic sentence/paragraph search mode (basic box is unchecked) you will get an error typing in

```
      nice day
```

because the space between nice and day throws off TAMS. Instead, put quotes around the phrase (either single or double, but both ends have to match) if it includes spaces or punctuation. So you would search for

```
      'nice day'
```

with those single quotes actually there!

So to find sentences with either gday, "good day" or "g'day" you enter the following search string and use a sentence search:

```
      gday, 'good day', "g'day"
```

Here double quotes around g'day are necessary, otherwise TAMS will treat the ' as a terminating quote. I used single quotes for the middle words, but I could have used doubles around "good day" just as well. For all sentences (or paragraphs) containing the words Paul and Simon; you would look for

```
      Paul+Simon
```

Spaces are ignored, so you could also have typed

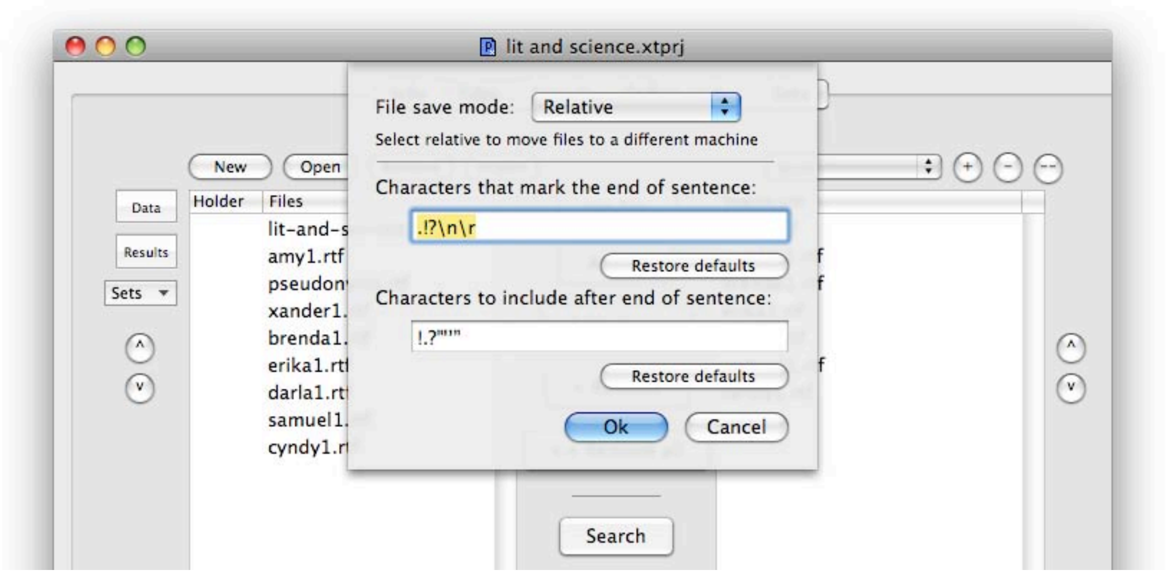
```
      Paul +              Simon
```

But only if the "Basic" box is unchecked! Unlike the Boolean connectors for non-simple code searches, you can use parentheses to group. The order of operations is parentheses, NOT, AND, OR. Parentheses force precedence.

Leaving the search string blank will return every sentence or paragraph in the files in the search list. This can be slow. In one large project of mine filled with news articles it took 3 ½ minutes to complete. If you leave it blank (no spaces, nothing), then it doesn't matter if the Basic switch is checked.

Once TAMS returns all of the sentences or paragraphs matching the criteria you have specified, you can mark and "Add code" these sentences (paragraphs). This is a pretty effective way of autocoding key themes in a document if those themes can be tied to specific word usage and paragraph/sentence boundaries.

Note, paragraphs are easy to locate and returned to you; sentences are very hard. Quotes after periods, unterminated sentences, etc., make sentence detection more or less a nightmare. I hope to improve my algorithm over time. I'm afraid the current version is English biased at the moment. Should work with Spanish too, but I don't know about other language's closed quotes. You can add to the list of characters that TAMS looks for as a sentence ender, and the list of characters that it then scans for that should be included in the sentence. These are set and stored in the project preferences (Project->Preferences menu). You'll get a dialogue like this:



File save mode: Relative

Select relative to move files to a different machine

Characters that mark the end of sentence:

.!?\n\r

Characters to include after end of sentence:

!.?"'"

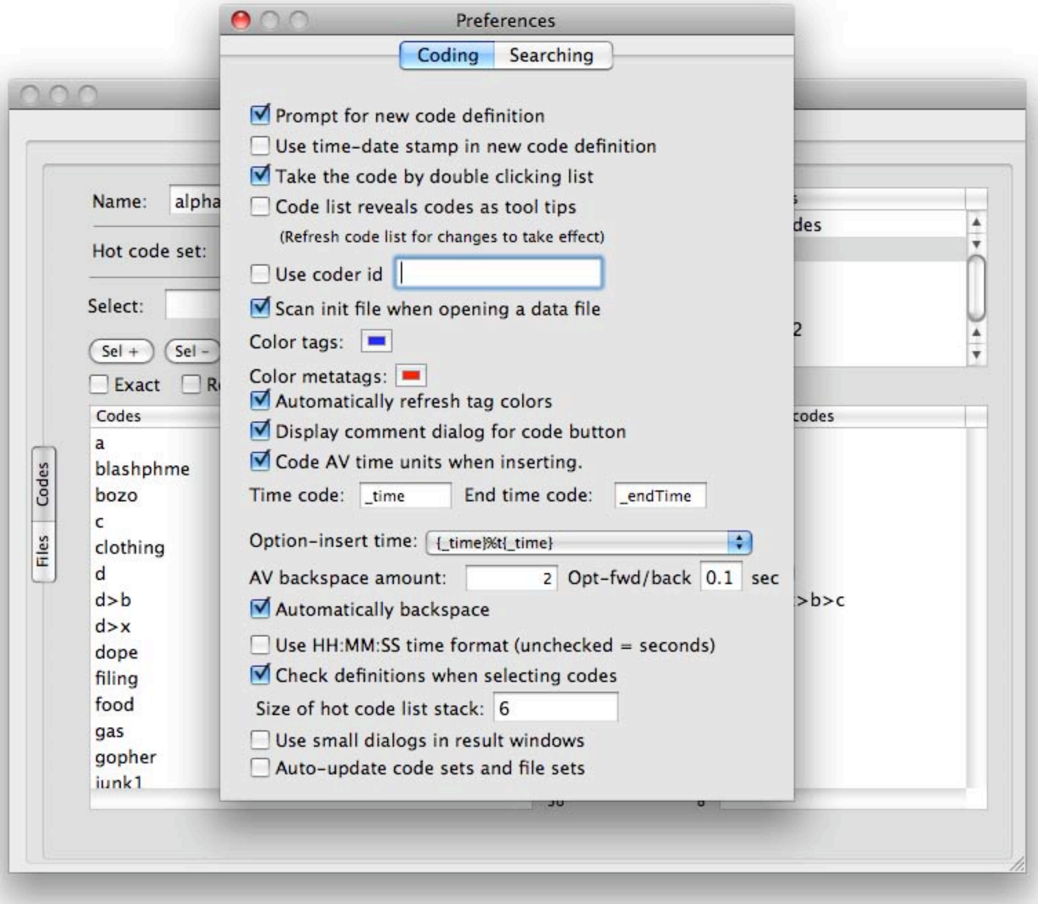
Project Preferences Menu

The default values are provided, though you probably want to leave these alone. Instead, move to the end of the text field and add additional characters for TAMS to look for. The second field includes things like ! since if someone types "Jump!!!" you want the first exclamation mark to indicate the end of the sentence (that's why ! is in the first box), but you also want all additional !'s to be included (that's why ! is also in the second box). So the sentence in the result window will be "Jump!!!" and not just "Jump!"

Good luck. Let me know how it works out!

2. Auto-updating Code Sets

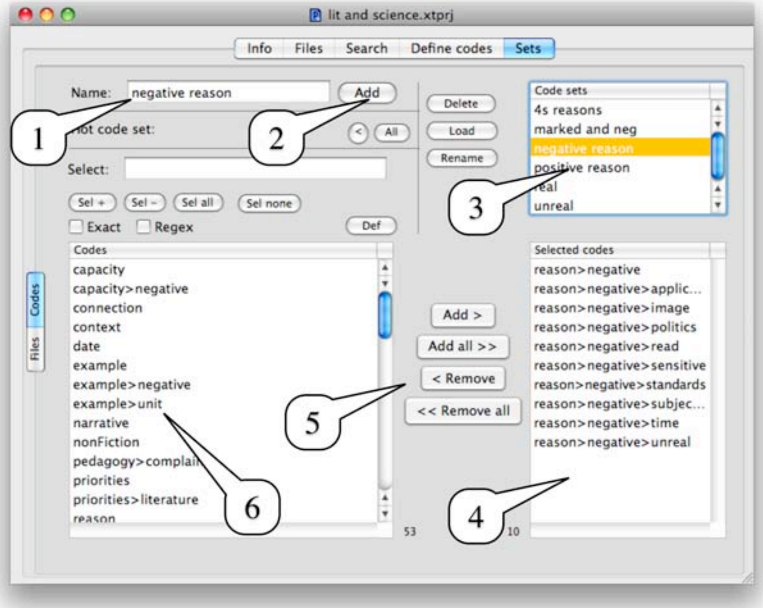
There's a new option that works better with some people's workflow (albeit not mine). People have asked to have code sets auto-update when they're fiddling around, adding and subtracting codes on the workbench. Here's the way it works. First you need to check the "Auto-update code sets and file sets" box on the program preferences.



Preferences

☑ Prompt for new code definition
□ Use time-date stamp in new code definition
☑ Take the code by double clicking list
□ Code list reveals codes as tool tips
(Refresh code list for changes to take effect)
□ Use coder id
☑ Scan init file when opening a data file
Color tags: 🟦
Color metatags: 🟥
☑ Automatically refresh tag colors
☑ Display comment dialog for code button
☑ Code AV time units when inserting.
Time code: _time End time code: _endTime
Option-insert time: {_time}%t{_time}
AV backspace amount: 2 Opt-fwd/back 0.1 sec
☑ Automatically backspace
□ Use HH:MM:SS time format (unchecked = seconds)
☑ Check definitions when selecting codes
Size of hot code list stack: 6
□ Use small dialogs in result windows
□ Auto-update code sets and file sets

Last Option Enables Auto-update



lit and science.xtprj

Code Set Tab

Without auto-update, you would move codes from table #6 over to table #4 using the buttons (#5). Then you would fill in a name into #1 and hit #2. Only when #2 is hit is the code set updated. With the auto-update checked, however, whatever code set is listed in #1 will be automatically updated as the codes are added and deleted from table #4. You don't hit the add button (#2).

I'm more conservative. I want to get the code set right in #4 and then hit the add button.

Typical work process would start, with auto-update, with clicking the code set you want to update in table #3. That will fill in the name into #1, and load the codes into #4. You then add and subtract as you like, the code set is changed each time! You need to realize you may be inadvertently altering your code sets!

If no name is present in #1, you will get an error message. You should then fill in a name and click add; otherwise the set changes will not be registered.

TAMS Analyzer 3.53b4 Release Notes

New Features

1. Full Boolean searches for simple, non-simple and section searches.
2. Open files by double clicking search list (just like the files list)
3. Gather citations (with toolbar button)
4. Code Count report now counts codes in setcontext, setcode, setrepeat, universal and var metatags (only counted if value is set). NOTE: co-coding is not changed and will not report on setcode, etc.

Bug fixes

1. Window boundaries better placed relative to tabview in project window
2. Infinite loop bug for !setcode fixed
3. Several memory leaks plugged in the tams character engine (the parser at the heart of searching).
4. Memory leaks plugged in the sentence/paragraph Boolean parser
5. Find record now works with outline view created by grouping column in results window; double clicking a row (or single clicking the row followed by clicking the Find Record button) in the outline will take you to the associated document and location.
6. Updated documentation; a-v how to and tool bar how to updated and added to on line help and pdf help. Qual. Anal. for Beginners updated.

Documentation

Full Boolean Searches For Codes

Now users can use AND, OR, NOT and parentheses (to force priority) for any of the code searches. The syntax follows the traditional tams syntax: ! = NOT , = OR + = AND.

So to find all codes other than mycode search for !mycode; to search for sections with both thiscode and thatcode search for thiscode+thatcode. You can still use modifiers like “>” which searches for a subcode, or “*” which does a substring search or “ ‘ “ which does an exact search.

Since **simple searches** apply your search criteria to each code it finds it makes no sense to ask for thiscode+thatcode in a simple search (you probably want a **non-simple search** which asks about stretches of text rather than about each individual code). There are some AND searches that make sense however for simple searches. For example you might want to find all codes and their text that have “>some” and “>level” as parts of their code. Or you might want to search for codes that have “>some” (meaning “some” appears as some level of the code) and the same code has a substring “sub” you would then search for

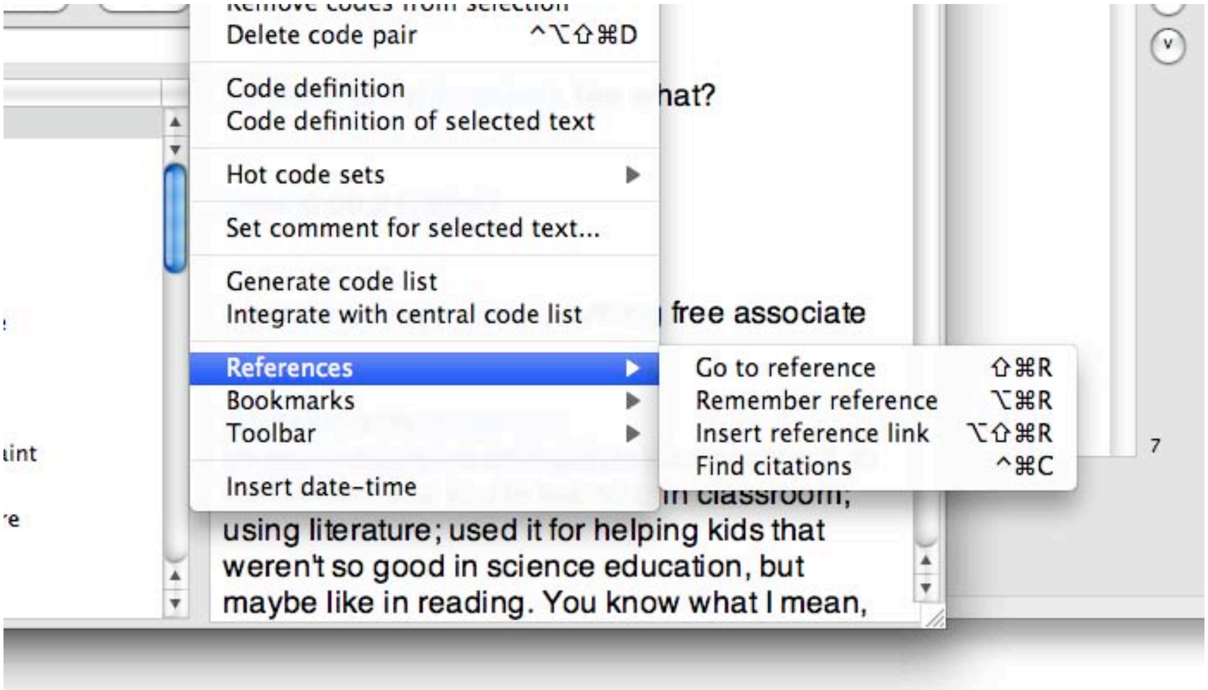
>some+*sub

Only passages coded with things like submarine>problems>some will be returned. The code has to meet the criteria of each part of the AND search. Like I said, it’s pretty unusual to use AND in a **simple search**, and most of the time the researcher really wants a **non-simple** search (passages of text coded with thiscode and thatcode). TAMS will warn you that AND (+) in a simple search is probably wrong, but if you are certain you can click "YES" to proceed.

Gather citations

This adds the final feature for the reference functions used for memoing. Earlier releases added the ability to link to bookmarks and files by using the !goto metatag. This release allows you to find all the citations to a given document (which documents have "!goto's" pointing to this file or bookmark).

Open the document for which you wish to find citations and pick Coding->References->Find citations

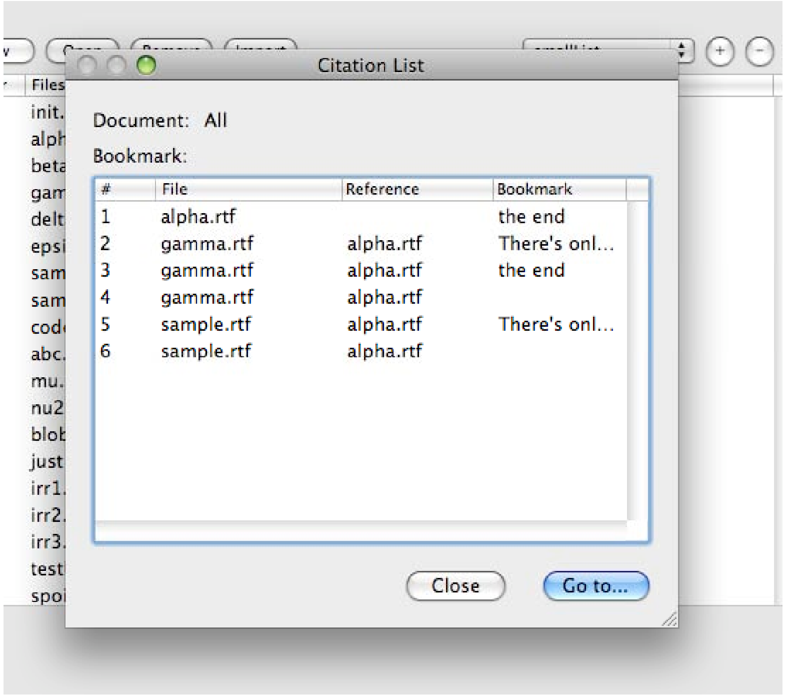


in classroom; using literature; used it for helping kids that weren't so good in science education, but maybe like in reading. You know what I mean,

Fig. 1: The new "Find citations" command

If you want to find citations to a specific bookmark, simply click inside of the bookmark and pick the same "Coding->References->Find citations" menu command!

What you will get is a window with a list of documents referencing the document/bookmark of interest:



Citation List

Document: All

Bookmark:

#	File	Reference	Bookmark
1	alpha.rtf		the end
2	gamma.rtf	alpha.rtf	There's onl...
3	gamma.rtf	alpha.rtf	the end
4	gamma.rtf	alpha.rtf	
5	sample.rtf	alpha.rtf	There's onl...
6	sample.rtf	alpha.rtf	

Fig. 2: Citation list

At the top is the document and bookmark (if appropriate) that you want to find citations for. If a document has more than one citation (i.e., !goto tag) to the current document/bookmark , it's name will be listed multiple times (see Fig. 2 where gamma and sample are listed multiple times because each has multiple !gotos pointing at alpha). Double clicking a row (or clicking once followed by the "Go to…" button) will open the listed document and move you to the !goto metatag that cites the current bookmark/document. Cancel dismisses the window.

Using the Find Citation command allows you to find any memos that refer to the current document. If executed from the workbench, the list contains all references in the project (see fig. 2)

The find Citation command is also available a tool bar button. To add it permanently to your toolbar use the !button command. The key word for find citation is cmd::citations.:

```
{!button cmd::citations}
```

Placed in your init file it will add a Find citations button to each of your windows:

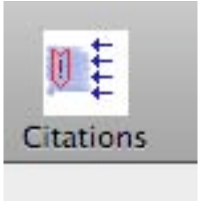


Fig. 3: The Find Citations Button.

TAMS Analyzer 3.53b9 Release Notes

This release patches a number of small annoying bugs that have plagued TAMS for a while. None of these bugs were fatal.

Bug Fixes

1. The back button for hot codes on both the SEARCH and SETS tabs now works if a (named) code set is loaded as the first hot code set.
2. Insert time code button on document windows now works correctly, which means it functions the same as the menu command. Time codes will now be coded if you use the button without modifiers. With the apple/command button it inserts the time without code. With the option it can do a variety of things depending on program preferences.
3. Small results window (set in the options) now is updated with padding controls and small fixes to the compare data report.

Interface changes

1. If you use AND (+) in a simple search you are now given the option of running it as an unlimited search rather than stopping the search.
2. Column selections in result windows are now maintained between actions
3. Select field to browse now uses the current selected column rather than a dialogue box. The icon has been changed and renamed to “Browse column.”
4. Quicktime stuff has been refactored into two classes that can be easily re-written for GNUstep: MWAVMediaFile and MWAVMediaView.

Release Notes for TAMS Analyzer 3.60b6

This release adds the ability to tag both data and results files and provides an easy search box in the files tabs for selecting files based on tags or name (or both). Tags can also be assigned while working on individual files.

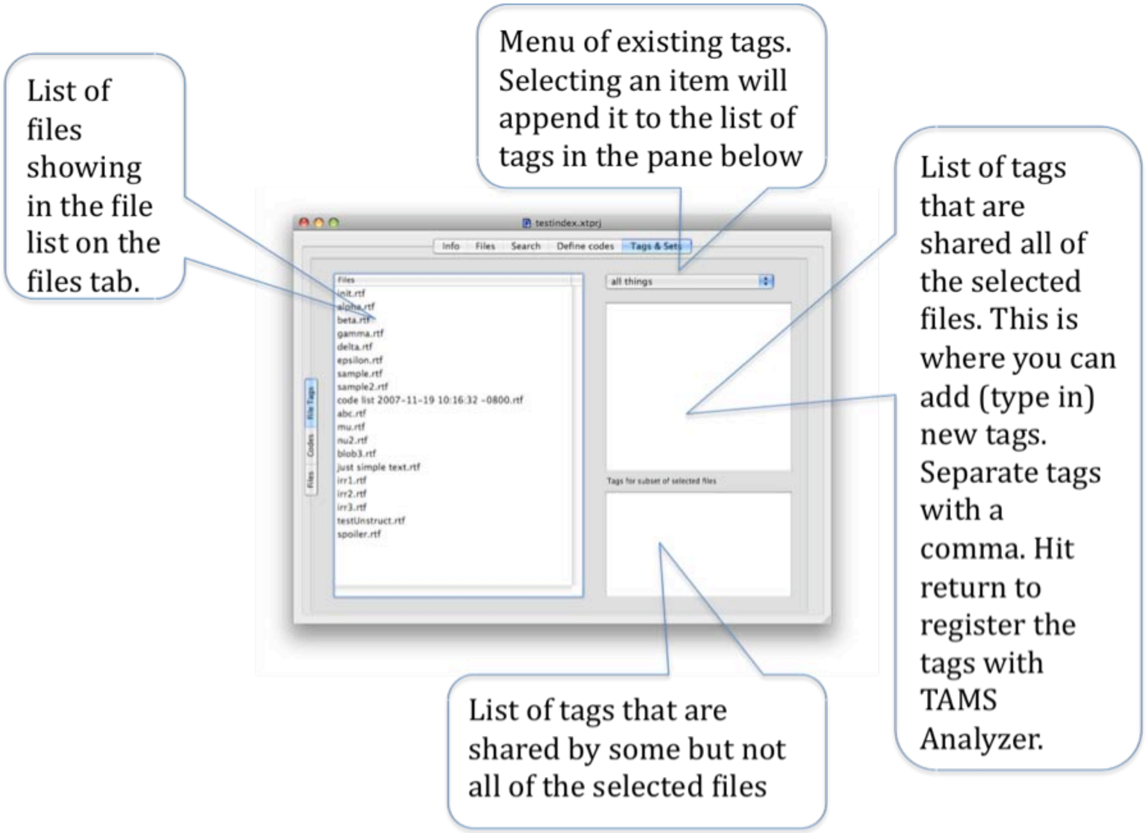
There have also been some improvements to menu items to make their purpose clearer.

Documentation

Tagging files is an alternative to file sets. File sets and tags are parallel structures for accomplishing very similar tasks: grouping and finding files. Tags provide a slightly cleaner interface and also a certain amount of flexibility in that they can be assigned while working in a document or en masse from the project workbench. Searches can be done with Boolean operators on tags, which is another feature not currently available for file sets.

1. Assigning tags from the workbench

The best way to assign tags for existing files is to use the “File Sets” tab in the workbench. First click the “Tags and Sets” tab, and then the “File Tags” tab on the left side of the pane:



Files
init.rtf
alpha.rtf
beta.rtf
gamma.rtf
delta.rtf
epsilon.rtf
sample.rtf
sample2.rtf
code list 2007-11-19 10:16:32 -0800.rtf
abc.rtf
mu.rtf
nu2.rtf
blob3.rtf
just simple text.rtf
irr1.rtf
irr2.rtf
irr3.rtf
testUnstruct.rtf
spoiler.rtf

List of files showing in the file list on the files tab.

Menu of existing tags. Selecting an item will append it to the list of tags in the pane below

List of tags that are shared all of the selected files. This is where you can add (type in) new tags. Separate tags with a comma. Hit return to register the tags with TAMS Analyzer.

List of tags that are shared by some but not all of the selected files

Figure 1. The File Tags Tab

You can assign a tag to multiple files by selecting the files that tag applies to from the left hand list and typing in different appropriate tags (separated by commas) in the box right beneath the menu. You can also append tags that have already been used by picking them from the menu above. The pane in the lower right hand corner is simply for information (you cannot edit it). It indicates which tags apply only to some rather than all of the selected files on the list on the left.

Normally tags are single words, but they can have spaces. When you search for a tag with a space in the name, the tag must be surrounded by double quotes (").

2. Assigning tags from a document you are working on

While you are working you can assign tags to the document by picking File->Tags... from the menu bar.

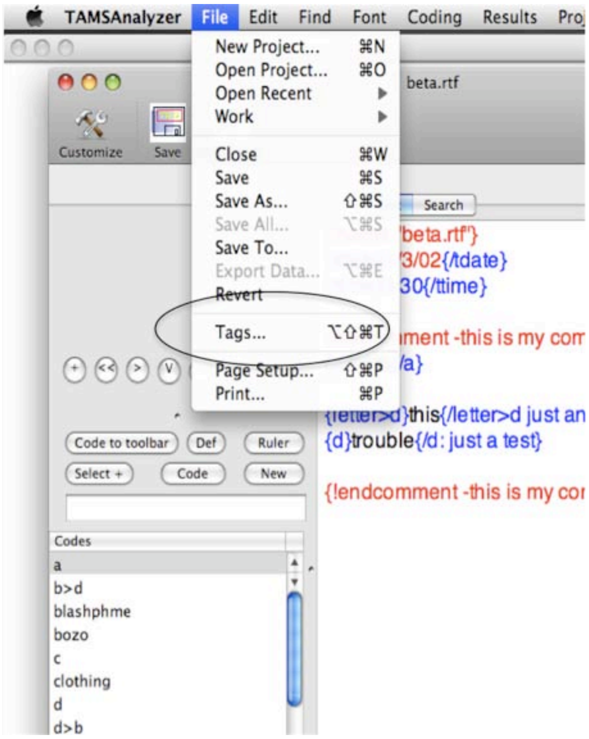
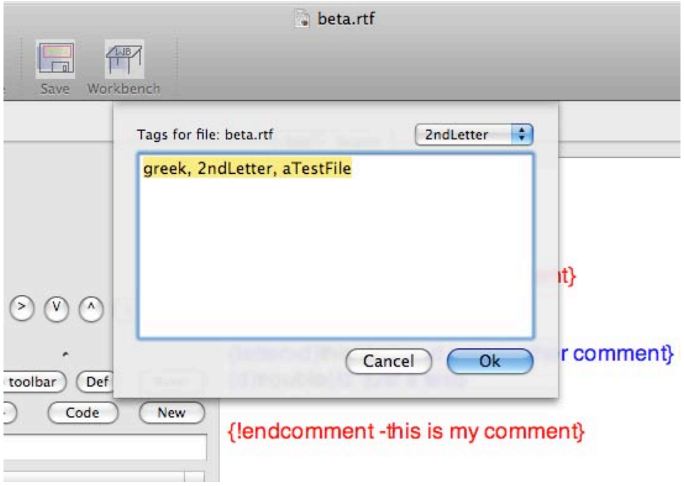


Figure 2. The Tags command on the File Menu

This brings up a box that will let you see and edit (including add) the tags assigned to a file. Again, enter tags separated by commas:



Tags for file: beta.rtf

greek, 2ndLetter, aTestFile

{!endcomment -this is my comment}

Figure 3. The in-document tag editor

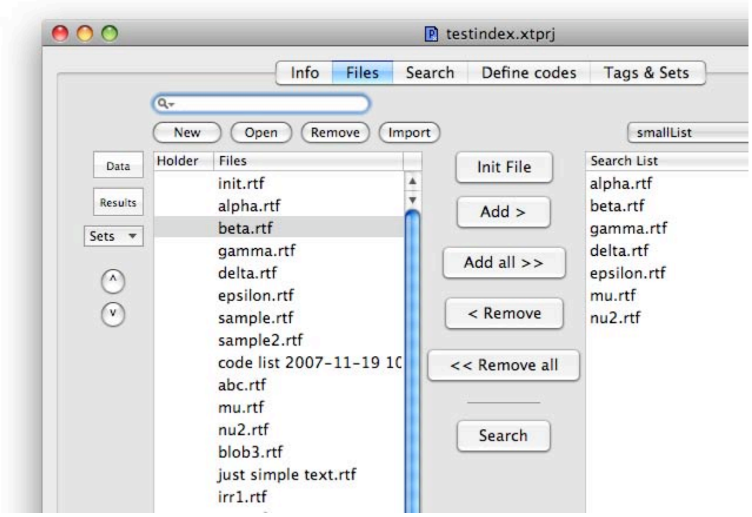
There is also a small pull down menu that will let you append existing tags to the editor.

3. Adding tags to a single file from the workbench file tab.

The same editor described in part 2 can be used from the file tab in the workbench. Bring the workbench to the front, pick the file tab, and select a single file. Then pick File->Tabs… from the menu bar. Note: if you are on any other tab of the work bench or have more than 1 file selected from the file list (that's the left hand list of files), you will get an error message.

4. Using (finding) tagged files.

Once you have tagged your files you can search for them using the search field above the file list on the workbench:



Holder	Files
	init.rtf
	alpha.rtf
	beta.rtf
	gamma.rtf
	delta.rtf
	epsilon.rtf
	sample.rtf
	sample2.rtf
	code list 2007-11-19 1C
	abc.rtf
	mu.rtf
	nu2.rtf
	blob3.rtf
	just simple text.rtf
	irr1.rtf

Search List
alpha.rtf
beta.rtf
gamma.rtf
delta.rtf
epsilon.rtf
mu.rtf
nu2.rtf

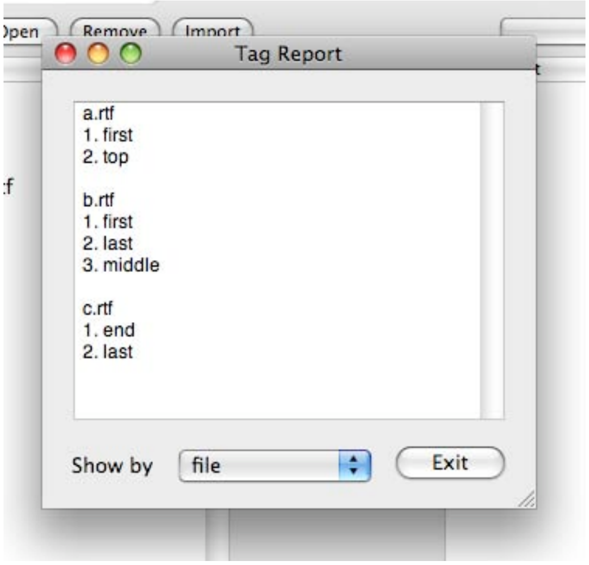
Figure 4. Search by using the search field (with the magnifying glass) above the file list.

You need to enter the full tag name, and the search is case sensitive. MEMO is not equal to memo. You can use an asterisk (*) before a partial tag name. So you could enter "*mo" to get files tagged memo, moment, demo, and amor. You can also use Boolean operators + for logical AND and comma (,) for logical OR. You can also use parentheses to force the order of evaluation. So to find all files labeled both memo and urgent type in "memo+urgent" (you could also use "*mo+*urg" if you were unsure of the tags). Similarly if you wanted to find interviews and memos both, type "interview, memo".

The searches will only be done of the files showing in file list, so you will need to click the data (or results) buttons to see all of your files before doing subsequent searches.

5. Getting a report of your tagged files

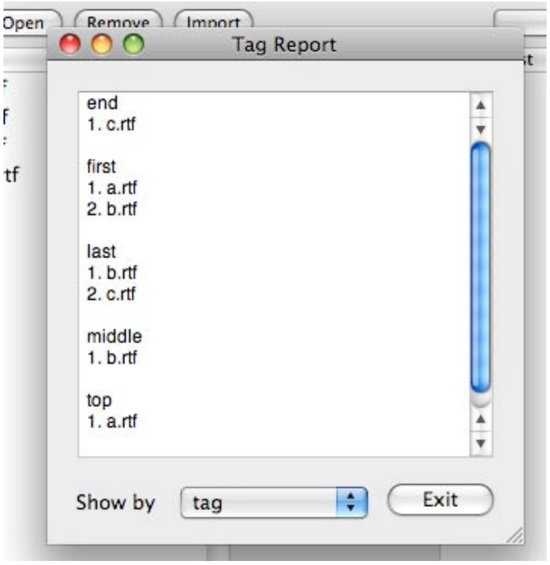
TAMS Analyzer has a new report which simply lists your tags by file or files by tag. This is accessed through the Report->File tags menu option (your workbench must be in front). In the window that comes up there is a menu which allows you to pick whether you want to see your tags listed by each file or your files listed by each tag. To print: copy and paste to a word processor or TextEdit.app.



```
a.rtf
1. first
2. top

b.rtf
1. first
2. last
3. middle

c.rtf
1. end
2. last
```

Show by: file

Figure 5. Tag report of tags by file



```
end
1. c.rtf

first
1. a.rtf
2. b.rtf

last
1. b.rtf
2. c.rtf

middle
1. b.rtf

top
1. a.rtf
```

Show by: tag

Figure 6. Tag report of files by tag

6. Searching for files

In addition to searching for tags, TAMS Analyzer will now let you search your file names directly using the same search box as for tags. Pull down the small menu next to the magnifying glass on the search box and pick "File names". Now the search will try to match the string in the search box with the filenames below (until you indicate otherwise by picking the tags option).

These are partial file name searches. Looking for “emo” will return everything with “emo” in it. All of the following would be returned: “emotion.rtf” “memos.rtf” “Memos.rtf” “Nemo.rtf”, but not MEMOS.rtf (the search is case sensitive). To do an exact search prefix the search term with a single quote ('). So to find memos.rtf exactly search for 'memos.rtf. It would not include “new memos.rtf” or “memos.rtfd”. Searches can include logical AND (+) and OR (,). So to look for file names that have memo and September in them search for “September+memo”.

TAMS Analyzer 3.61b7 Release Notes

This release fixes a few problems with tagging files. It also adds a tag menu to the workbench Files tab. In addition to a number of small wording changes in the interface, this version also automatically saves project, document, and results positions. Result window positions are only saved if the files are not temporary.

Release Notes for TAMS Analyzer 3.61b8

This is a bug fix release. It patches a problem that prevented the creation of new projects.

TAMS Analyzer 3.61b16 Release Notes

This version updates a number of features. File and Code sets in particular are given a couple of feature boosts.

New features:

1. Time codes in seconds can now be sloppily selected when jumping to a time code. Previously the time had to be selected exactly (no additional characters, not even the tags that surround the time, could be included).

2. Find paired tags now accepts sloppier selecting. Users no longer need to click inside the tag, but can select the whole tag when searching for a paired tag.

3. Import definition and codes from code file now properly interprets escape characters.

4. Double clicking a code or file in the sets panel (code and file tabs) now indicates which code sets include that code by selecting them in the code set list. (see Documentation below)

5. Clicking multiple file and code sets (previously not allowed) selects all the codes from those file sets allowing you to create new file sets by combining others. Use the command and shift keys to extend selections.

6. In the sets and tags pane, there is now a count for the number of code sets and file sets.

Change in behaviors:

1. Creating or modifying a File set no longer automatically “loads” that file set.

2. Buttons on the search tab now combined into a menu

Bug fixes

1. Interface repaired for file tag tab so labels and fields resize correctly

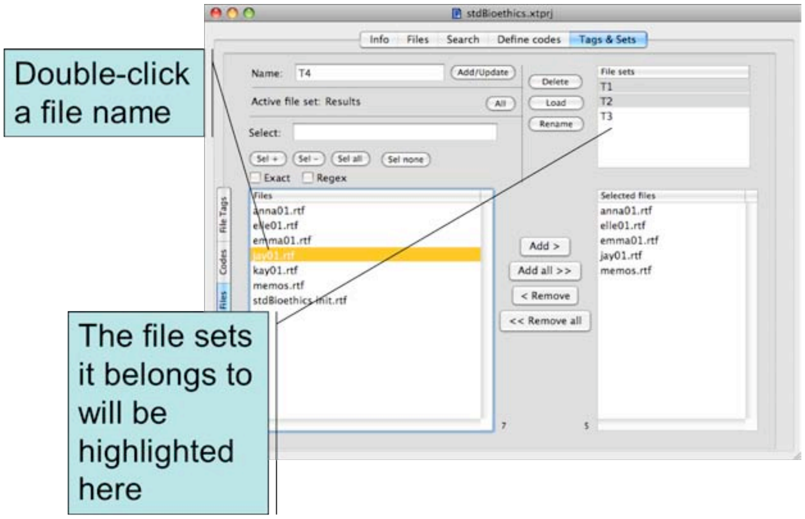
2. Buttons on file tab changed so they are Panther compatible

3. File sets menu and select more (and other functions) work with each other; previously using the file set menu did not register the file set so that select more (and maybe other functions) did not work correctly

Documentation

Finding the file sets or code sets that contain a particular code.

In previous version this could be done through generating a graph of the code sets, but often you don’t want anything as formal as a graph, you just want to see which code sets contain a particular code. Now this can easily be done on the sets tab of the work bench:



Name: T4

Active file set: Results

Select:

File sets
- T1
- T2
- T3

Files
- anna01.rtf
- elle01.rtf
- emma01.rtf
- jay01.rtf
- kay01.rtf
- memos.rtf
- stdBioethics init.rtf

Selected files
- anna01.rtf
- elle01.rtf
- emma01.rtf
- jay01.rtf
- memos.rtf

Double-click a file name

The file sets it belongs to will be highlighted here

Figure 1: How to find out which sets hold a file

The same procedure will work with codes on the Codes tab of the Tags & Sets tab.

Release Notes for TAMS Analyzer 3.70b5

This release is a major number bump for a reason. There are new features, major bug fixes, logical enhancement, and then some "behind the scenes" activity—all described below:

New Features:

1. Value sets: create arbitrary groups of values (in one table column) (See documentation below, section I)

2. Select results based on value sets

3. Edit value sets

4. Create file sets based on current selection in results (See documentation below, section II)

5. Filter results by file set

6. Ability to intersect (in addition to combine [union]) sets through selecting multiple set names in the tags and sets tab. (See documentation below, section III)

7. Multiuser TA: ability to put project files in the "trash." This will delete those files as users synch with the database (See documentation below, section IV)

8. Multiuser TA: Principle Investigators can empty the trash, or untrash files; all users can view files.

Interface Changes

1. Menu keyboard shortcuts for next and previous on the project menu have been changed to remove redundancy (those key shortcuts were used everywhere)

2. Multiple code sets can now be selected on the codes set subtab on the "tags and sets" tab.

3. Warnings added for addcode, recode and the other recode menu options to warn users that they need to "mark" the records they want to change

Bug fixes

1. Project window no longer accepts Apply code

2. Bug fixes for multiuser tams to facilitate connectivity in tams.

3. Bug fix to allow the user to enter larger file sizes for multiuser tams upload.

Code base changes

1. Multiuser and Single user are now two different targets in the same XCode project

2. XCode 3.2.1 is now required to maintain the code (still using GCC 4.0)

Documentation

I. Value Sets

Value sets are simply a collection of values in a particular table column which can then be used to filter your results. Code sets have long acted as a value set for the _code column. Value sets simply generalizes this idea to any text column of your results window.

A. Creating Value Sets

Create a selection of rows that has a set of values you would like to preserve. For instance if you have a “Interviewee” context variable, you could create a value set of “Interviewees that use Code X” by
1. Selecting code X
2. Clicking on the “Interviewee column” (clicking it’s header)
3. Picking Results->Value sets->Create value set…

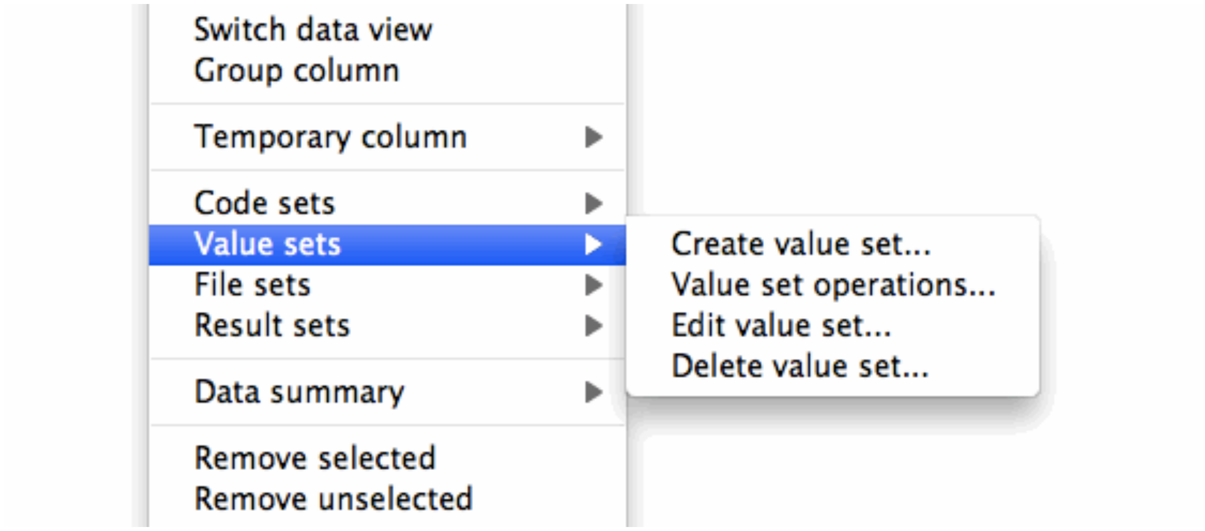
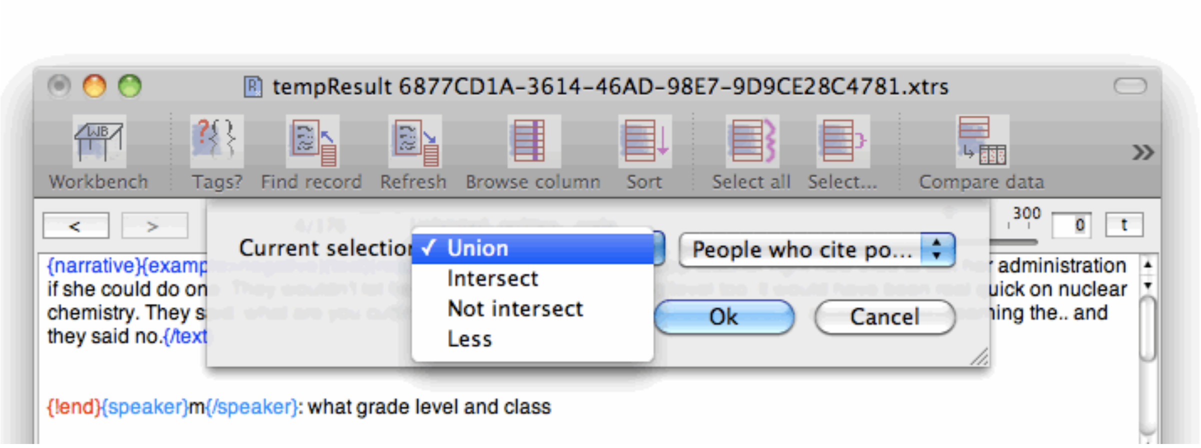


Figure 1. Value Set Menu

You will be prompted for a name. You can se spaces and other characters: there are no specific limits on what a code set name can contain, but they may be truncated in certain panes.

B. Using Value Sets
TAMS provides a variety of set operations to allow you to filter your results by value set. These operations all operate on the current selection in your results window. To filter (either additional records or fewer records) pick Results->Value sets->Value set operations… Figure 2. shows the choices you will have (if you have created value sets)



Current selection ✓ Union People who cite po...

{narrative}{examp
if she could do on
chemistry. They s
they said no.{/text

administration
uick on nuclear
ning the.. and

{!end}{speaker}m{/speaker}: what grade level and class

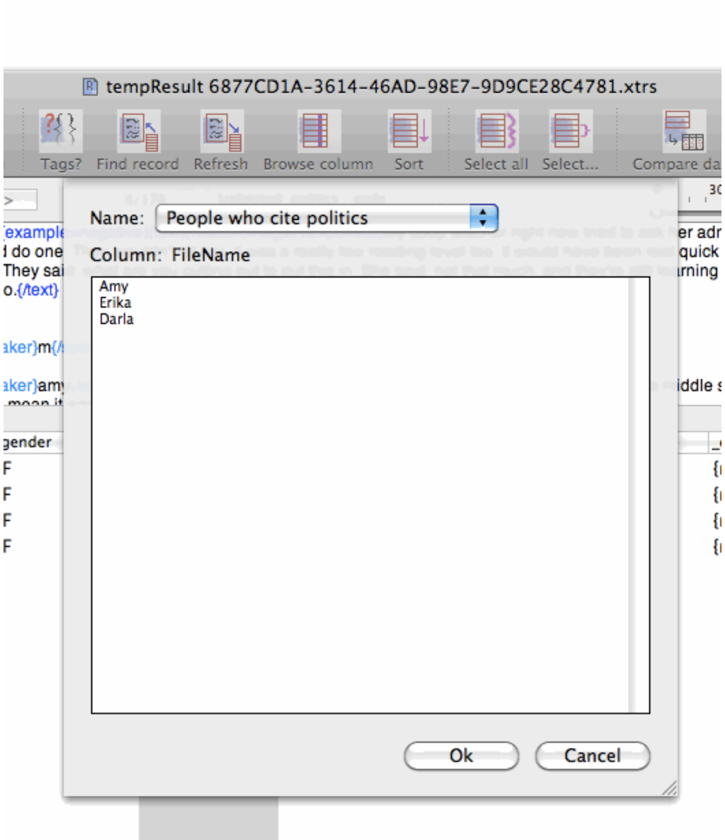
Figure 2. Value set operations

Here you pick the type of set operation you want applied (with the current selection) and the name of the value set to interact with:

- Union: Combines the given set with all instances (rows) that satisfy the vale set.
- Intersect: Reduces the current selection to only those that satisfy the value set you choose.
- Not intersect: Combines the current selection with those that satisfy the value set you choose, but removes those from the current selection that would exist at the intersection
- Less: Simply subtracts out records from the current selection that satisfy the value set you choose

C. Editing value sets

You can hand edit the values collected in a particular value set by bringing up the value set editor: Results->Value sets->Edit value set… The pane shown in figure 3 shows the editor and the set selector menu:



Name: People who cite politics

Column: FileName

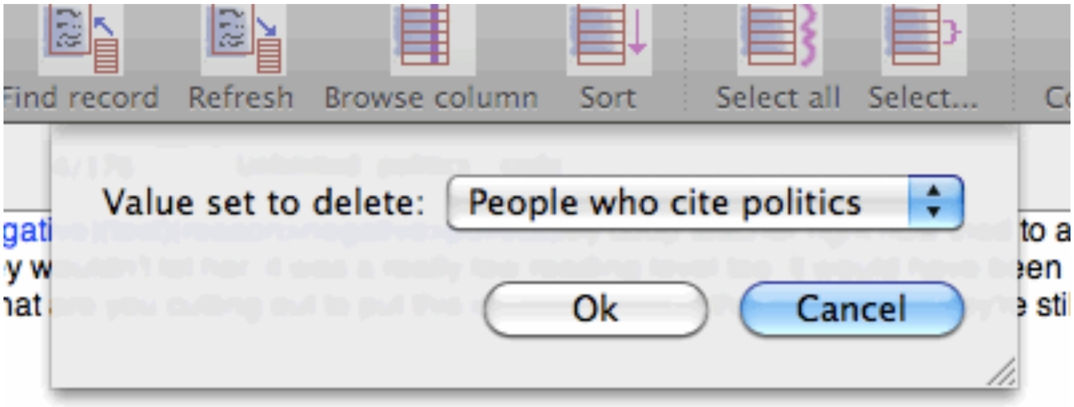
Amy
Erika
Darla

Figure 3. Value Set Editor

Pick the value set from the menu, and a text edit appears. You can copy the values into another program to make a chart and use these values for reporting purposes or edit them, just putting a return between each value. Returns and tabs internal to each value will appear as escape characters (\n, \t).

D. Deleting value sets

To drop a value set simply pick Results->Value sets->Delete value set… Just pick the value set to delete from the menu. (See Figure 4.)



Value set to delete: People who cite politics

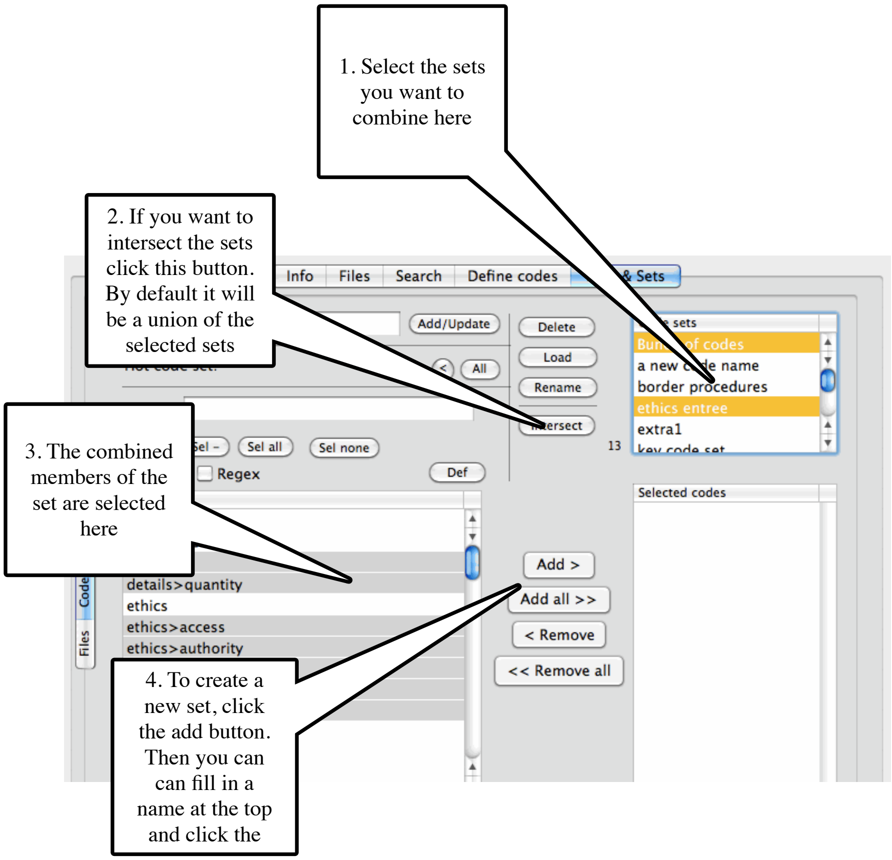
Figure 4. Deleting Value Sets

II. File set operations in the results window

Until this release file sets were basically just to facilitate project management. They had no analytic function. Results windows had no access to the file sets, unlike code sets. With this release, results windows can both be filtered by file sets and file sets can be created from the documents included in any particular selection. This means you can create a file set that is “Files that contain code X,” for instance. Any selection can be represented as a file set. In essence, file sets are just value sets of the _doc column. Unlike value sets which are stored in the results window (they are not globally available at this point), file sets are stored in the project file/window and thus can be accessed everywhere. Deleting file sets and editing them is still handled in the project window’s tags and sets tab. Finally the set operations you can use are identical with those described in section I under figure 2.

III. Combining (union and intersect) code and file sets in the Tags & Sets tab

In theory forming a union of two code sets was available in the last release. In practice a bug kept this from being available for code sets (but not for file sets). To combine sets simply select them in the upper right table of the code sets manager (I’ll just address code sets, but this all applies to file sets too) and the result will be indicated in the Codes table in the lower left corner. This figure 5 walks through the steps of combining sets.



1. Select the sets you want to combine here

2. If you want to intersect the sets click this button. By default it will be a union of the selected sets

3. The combined members of the set are selected here

4. To create a new set, click the add button. Then you can can fill in a name at the top and click the

Figure 5. Combining Sets

Example of a problem that needs these new features:

Final thoughts about the features described in sections I-III above. These features were developed because one TA user asked how to select records that met exactly the following conditions: All codes X and Y from files that must contain both X and Y. While there were lots of ways to visually get this information: either with a select near or with a data comparison table, for instance; these solutions involved visually looking through a lot of noise to identify the answers. The new features allow, with a bit of work, getting exactly these results. One way might be run an unlimited search and selecting X and creating a file set of "Files that contain X" and repeat for "Files that contain Y". Now using the workbench create a file set of the intersection of these two file sets following the steps described in section III ("Files that contain X & Y"). Now you can make a search list of "Files that contain X & Y" and look for X and Y.

Alternatively you could do an unlimited search, select code X, and make a value set of the _doc column called "files that contain X"; similarly make a value set for "files that contain Y". Now select all records that have X and Y (use select and then select additional), now filter these results by intersecting this selection with the two value sets you created.

These are awkward, but they do make available specific selections that previously were not possible.

IV. Garbage management in Multiuser TA

In previous versions of TA there was no means to really remove a file from a project. Using the traditional "Remove" button would temporarily dispose of the file, but synching with the project database would restore the file. Administrators would have to go in and delete the file after everyone had thrown it out.

With this release TA helps manage that process. PI's (principle investigators) can select files on the files tab of the workbench and push the new trash button. The files must be checked in for this to work. This will change the holder of the file to "__trash" and remove the file from the PI's project. As other people synch, this file will be removed from their projects as well. No new files with the name of the trashed file can be added or imported, however. That is, they can't be added until a PI empties the trash. The PI will need to use email (or some other means) to make sure that everyone has synched or manually removed the file. Once this is confirmed they should press "show trash" on the files tab and click the empty trash button. This will free that file name for others to reuse. The PI can also empty the trash of specific items by selecting them in the "Show trash" pane. In addition, the PIs can undelete (restore to the library) items in the trash by selecting them and pressing the "Undelete" button. These buttons will not be visible for those who are not PIs.

See Appendix 4 of *Using TA3 with Multiple Researchers* for more details.

TAMS Analyzer 3.71b4

This release expands the power of value sets (introduced in 3.70). Now value sets can be saved globally so all result windows can take advantage of particular value sets. In addition, value sets and file sets can be used in "autosets" (macros that automate the selection process in results windows).

LOTS of CRITICAL (yes, I'm shouting) bug fixes as well, especially related to section searches. This became visible only if there were codes that spanned sections.

New features
1. Value sets can now be made available globally (in all results).
2. Value sets and file sets can be used (through the add history or as an added step) in autosets (see documentation below)
3. Raw section searches now prefix results with any codes that span more than one section (just like simple searches)
4. Code set step in autosets now implements "not intersect"
5. Select (intersect) is now the default value for file and value sets

Interface changes
1. Hint regarding tags added to tags and sets tab
2. The wording on the select file sets and value sets matches the language on the autosets and code sets panels (see documentation below)
3. _code field of section searches now shows the limit string

Bug fixes
1. Major memory leak in section searches repaired
2. Create file set bug fixed (couldn't create file sets from the tags and sets tab)
3. Show trash button now hidden on single user tabs
4. Vestigial methods created for single user tams so the nib doesn't complain when loaded that it can't find multiuser related methods.
5. Several other bug patches related to string comparisons deep in the functioning of TA

Documentation

1. The equivalence of set operations

In this release I have changed the name of set operations on value and file set operations panels to match those of code sets. There is no difference in functionality only in names so I thought a table would help. I will refer to this set diagram in the "Results" column:

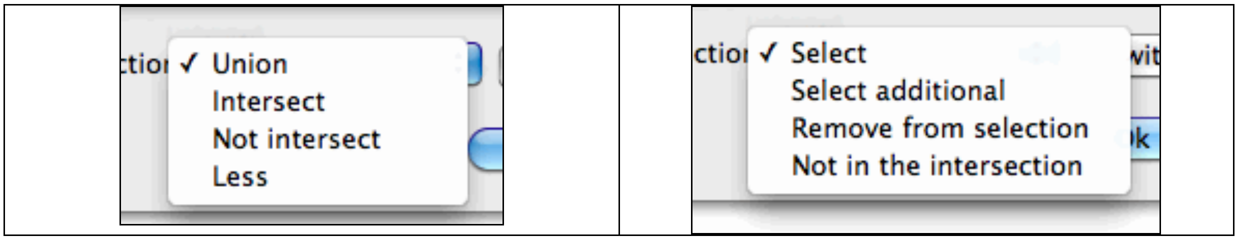


Figure 1. Old and new set operations

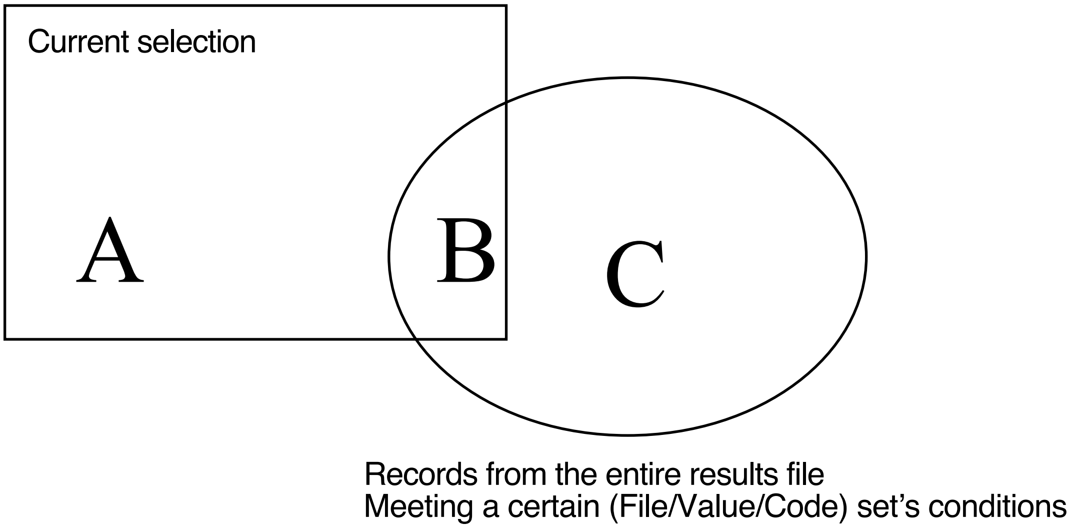
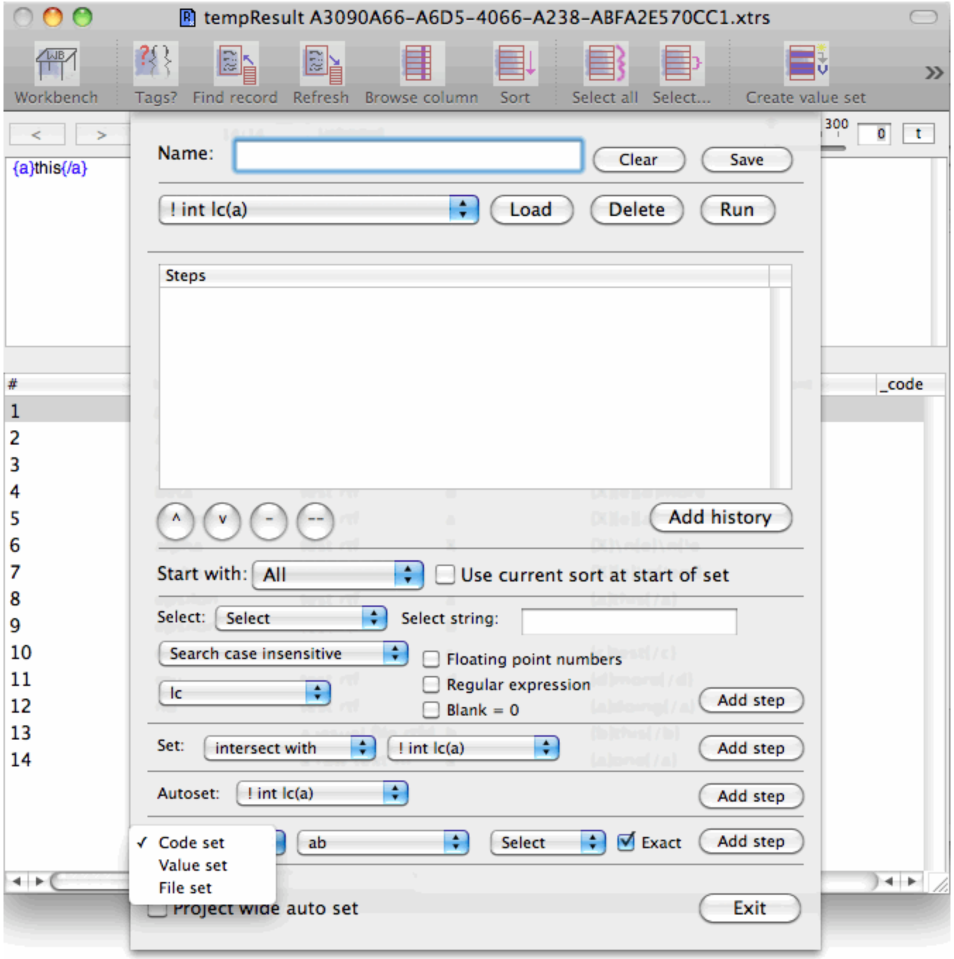

Current selection
A
B
C
Records from the entire results file
Meeting a certain (File/Value/Code) set's conditions

Figure 2. Set diagram to explain set operations…

Old name	**New name**	**Results**
Intersect	Select	B
Union	Select additional	A, B, C
Less	Remove from selection	A
Not intersect	Not in the intersection	A, C

Table 1. Set operation equivalencies

2. Value sets and file sets in Auto set operations



Name:

! int lc(a)

Start with: All □ Use current sort at start of set

Select: Select Select string:

Search case insensitive □ Floating point numbers

lc □ Regular expression

□ Blank = 0

Set: intersect with ! int lc(a)

Autoset: ! int lc(a)

Code set ab Select ☑ Exact

□ Project wide auto set

Figure 3. The autoset panel with set choices showing

In the old autoset panel (Results->Result sets->Create autoset…), one of step choices was code sets. This has been replaced with a pop up menu where you can decide if you want to add a code set, value set, and file set selection. Once you pick the type of set, the possible sets of that type will be offered on the adjacent menu (in figure 3 showing a code set called “ab”). Finally you can pick the type of set interaction you want in this step:

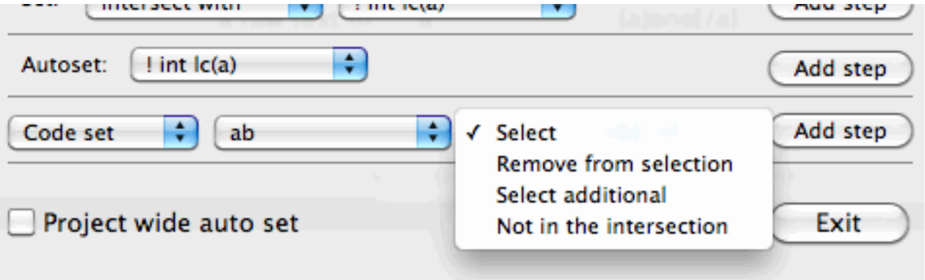


Figure 4. Set interactions

These now match those on the set operations menus. See this documentation, part 1.

TAMS Analyzer 3.72b6 Release Notes

This version of TAMS continues to expand on the analytic roles of value and file sets by integrating them into data comparison tables.

New Features

1. Ability to use value sets and file sets in data comparison tables (See documentation)
2. Ability to use code sets on both axes of the data comparison tables
3. Regular expressions have been updated to PCRE 8.02

Interface Changes

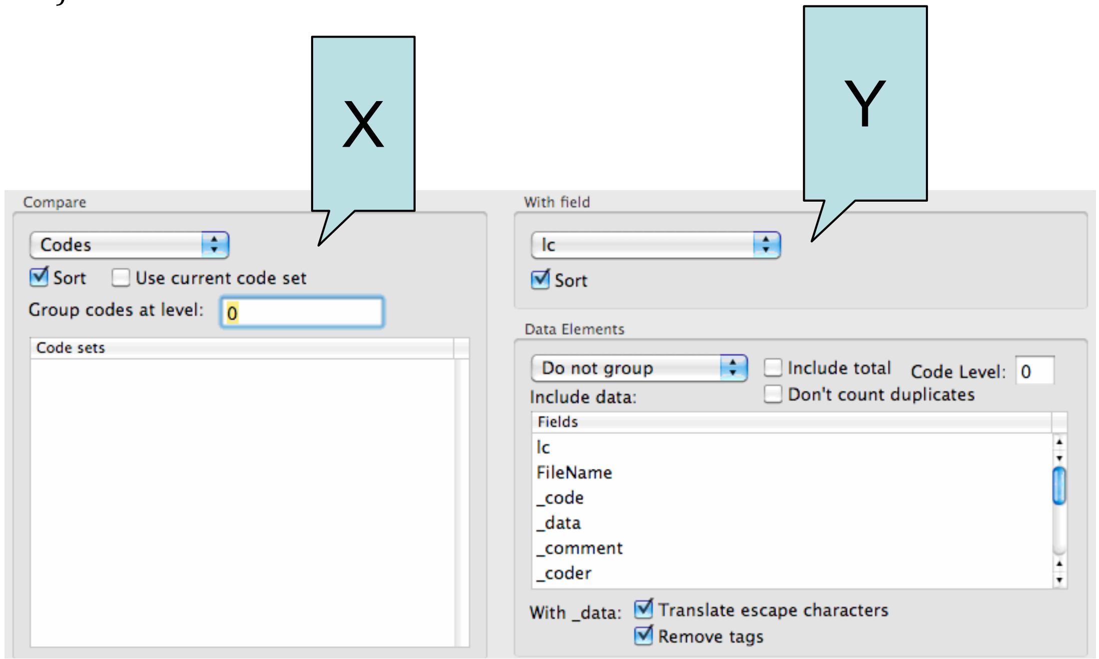
1. Data comparison tables now work with setting X and Y axes rather than “Compare” and “With field”. The interface is a little trickier but much more flexible

Bug Fixes

1. Serious bug fix that affected accurate display of data; for some reason this came out using files saved on Truecrypt volumes, and not at other times. I suspect this is a snow leopard issue (or a Truecrypt on snow leopard issue)
2. Small dialogues have now been integrated into the same nib file as regular dialogues for results. This will mean people using the small nib files will find that tams is much more feature complete.

Documentation:

Figure 1 shows how data comparison dialogue looked in prior releases. Note that the area called “Compare” controlled the X axis and “With field” controlled the Y axis (see figure 2)



Codes
☑ Sort ☐ Use current code set
Group codes at level: 0

lc
☑ Sort

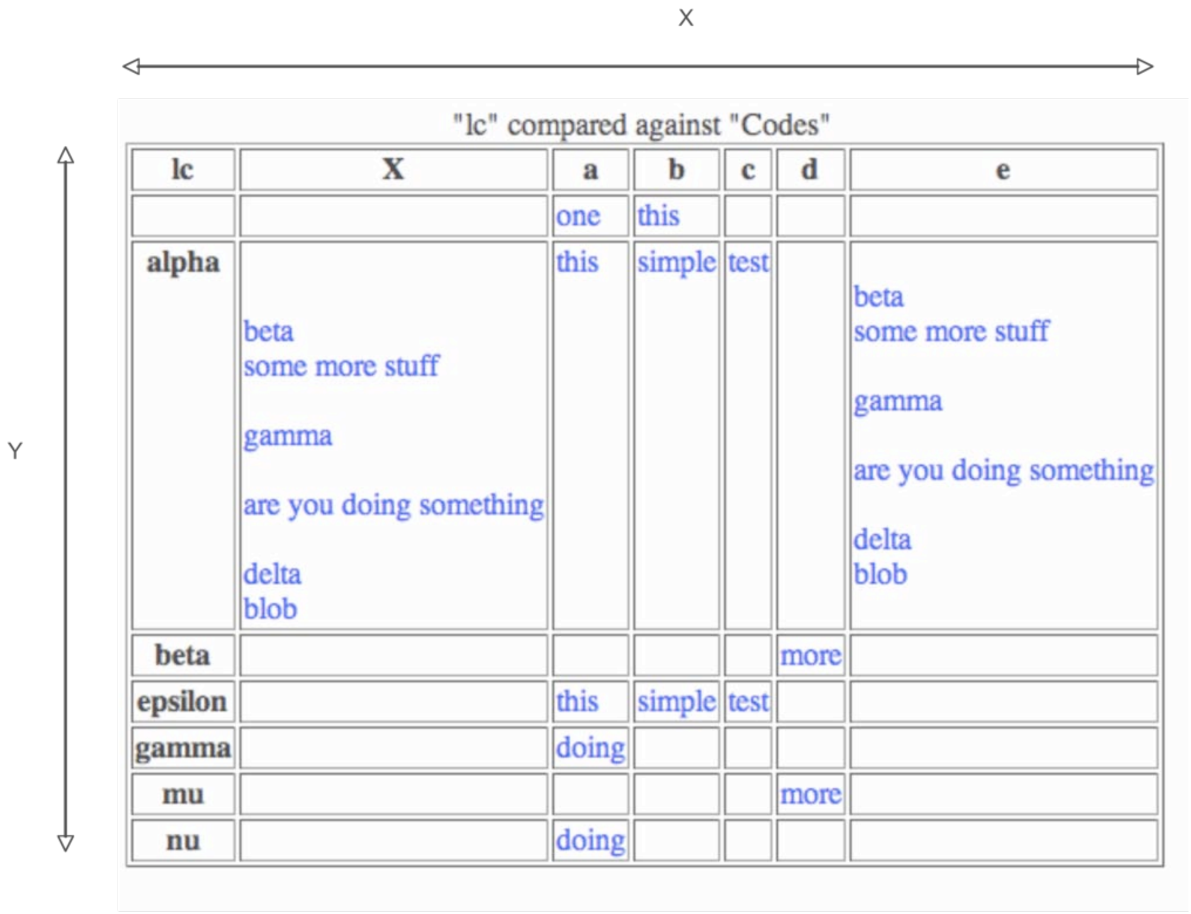
Do not group ☐ Include total Code Level: 0
Include data: ☐ Don't count duplicates

lc
FileName
_code
_data
_comment
_coder

With _data: ☑ Translate escape characters
☑ Remove tags

Figure 1: Old data comparison table pane with “Compare” field as X and “With field” as Y

So X and Y control horizontal and vertical layout of the table:



"lc" compared against "Codes"

lc	X	a	b	c	d	e
		one	this			
alpha	beta some more stuff gamma are you doing something delta blob	this	simple	test		beta some more stuff gamma are you doing something delta blob
beta					more	
epsilon		this	simple	test		
gamma		doing				
mu					more	
nu		doing				

Figure 2: X and Y in a data comparison table

In the above example the horizontal direction maps the data against the codes in the results table (X, a, b, c, d, and e are all codes) and the vertical dimension maps out the values of a particular column: "lc" (alpha, beta, epsilon, etc., are values in the lc column). The data elements in the rest of the table in this case are the vales of _data, the actual text being coded. So I can see, for instance that the word "more" was coded as "d" twice, once with a context of lc = beta and once with a context of lc = mu.

In the new data comparison table X and Y are controlled by the same panel by toggling an X and Y menu, rather than separate panels:

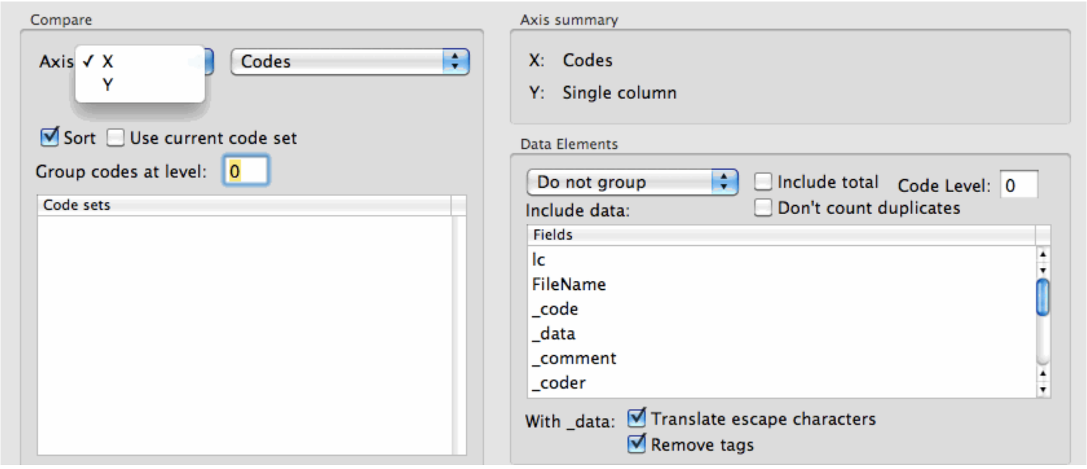


Figure 3: New data comparison table pane with the X/Y Axis menu

When X is chosen on the axis menu, you are setting the horizontal options, when Y is chosen, you are setting the vertical options. Some general information is shown in the spot where the old "With field" information was selected, now labeled as "Axis summary."

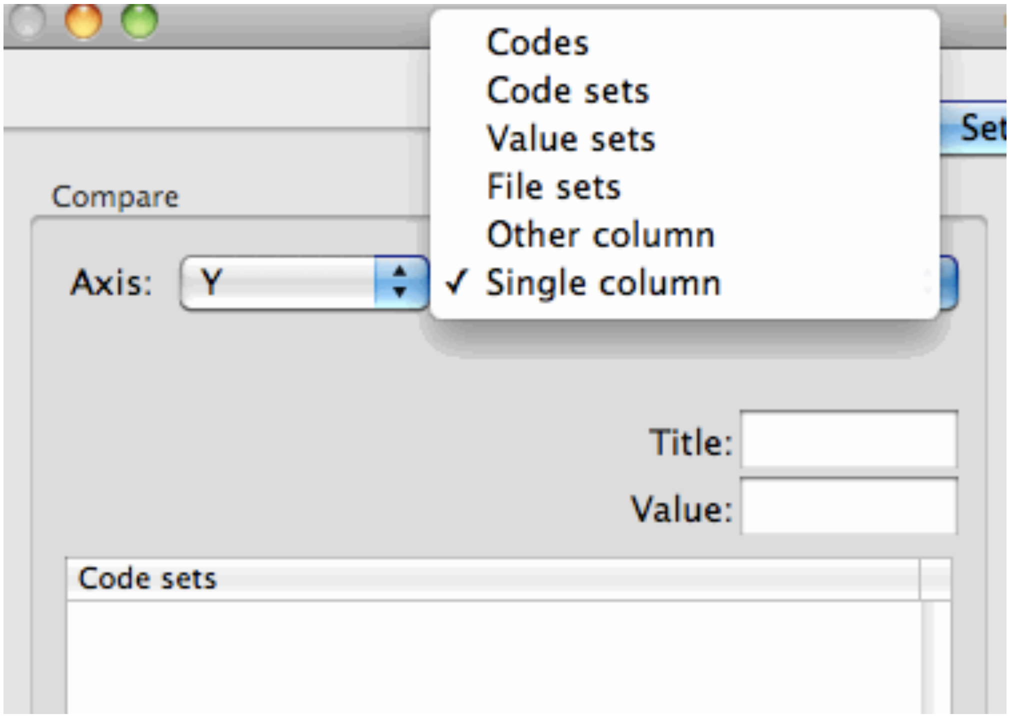


Figure 4: Y field options

The options for choosing the variables for the "with field" (Y) now are exactly the same (with one additional option) as with the X field. You can pick Codes, Code sets, Value sets, File sets, or "Other column" as the type of data to display. The names of the sets are listed below in the table (Column titled "code sets" in Figures 3 and 4 above, but the title of the table will change depending on the type of set you are picking). From this table you can select the sets you want (each will appear as a separate column or row), or none (which TA takes as meaning **all**). Use the command (apple) key and shift key to pick multiple sets from the table. . The Y field also will have a “Single column” option where all the values appear in one long list.

Picking “Other column” will reveal an additional menu of the fields (columns) from your table. You can pick one of those to compare context variables or the _doc column against other context variables, codes, sets, or columns.

To summarize, what’s new in this version is that the X field can have Value Sets and File Sets and that the Y field can be Code Sets, Value Sets, or File Sets.

Different options will appear as appropriate for the different options picked from the type of data you are trying to display. The Y axis has a few more options than the X axis, such as the ability to present data in a single column and to list column rows for codes even if no code appears in the results (the whole code list from the project window is used to create the rows on the table) – this is the “empty” option.

So now you can look at your data in terms of file sets vs. value sets or one code set vs. another code set, etc. Note that while this is complex, the options have not substantially changed from earlier releases other than the ability to chart all types of sets on both axes.

From a users point of view this is a pretty dramatic change in how these tables are created, but there is no lost functionality, and in fact, there is a lot gained in being able to chart data according to the sets you create.

TAMS Analyzer 3.72b15 Release Notes

This is a bug fix and feature smoothing release. Some of the bugs are important, but none program crashing or so serious as to prevent analysis of data.

Interface changes

- Holder column will now *not* resize when window resizes
- Axis information in Data Comparison Tables (DCT) is a little more informative
- Results window will restore table view (vs. outline view) when quitting the DCT panel
- DCT now selects _data as the default for table data elements

Bug fixes

- Critical bug fix for single user version. Now users can remove files from the project
- Important bug fix for single column DCTs which had no actual data to show
- Resizing of both the small and large panel versions of DCTs now works. Before it left table elements drifting all over each other
- Zeros (0) in counts now obey the formatting rules in terms of vertical and horizontal justification

Other

- DCT documentation brought up to date in both the user manual and tamsZine #2.

TAMS Analyzer 3.73b11 Release Notes

This release involves some major improvements and expansion of options on TAMS reports. There are two totally new reports (a graph of co-coded passages and a graph and table co-occurring codes[1], the latter is a variant on the existing co-occurrence table). In addition, where appropriate, graphs now can represent how many instances are through the thickness of the edge.

New Features
1. Co-coding report now allows for control of the contents of both axes
2. Co-coding graph
3. Co-occurrence graph
4. Simpler co-occurrence table
5. Graphs can now visualize the number of instances through the thickness of the edges (lines connecting notes).
6. Mark All Rows now works with outline view; only marks expanded rows
7. Collapse up and collapse down now expand/collapse rows in outline view of results windows

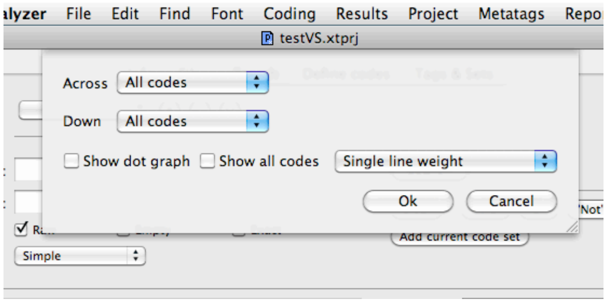
Interface changes
1. Smart sort no longer de-selects the column that is selected
2. Co-coding report now provides options for what codes will be included in the counts
3. Reports menu is now expanded. Many items have clearer titles. Reports for the project window are now separated from reports for the results window. (This may make some buttons for reports stop working. Users should remove and reinstall the button from their toolbars if they have problems. If problems persist, dispose of the program preference file).

Bug fixes
1. Export data now interprets escape characters (e.g. tabs [\t])
2. Select near for character distance now looks at both begin and end to determine nearness.

Documentation:

1. Co-coding Report

The co-coding report, run from the workbench, previously used the hot code list to determine what codes appeared on the results table. The new report presents a dialogue; see figure 1. This dialogue allows you to choose the codes across and down.


alyzer File Edit Find Font Coding Results Project Metatags Repo
testVS.xtprj
Ok
Cancel
Add current code set
Simple

Across: All codes

Down: All codes

☐ Show dot graph ☐ Show all codes Single line weight

Figure 1. Co-Coding Report Panel

[1] Co-coded refers to passages that have two or more codes applied to them. Co-occurring refers to passages that share some context value. Typically they are two passages (overlapping or not) that are in a particular section of a document.

Each direction has options for **all codes**, **the current hot code set**, or **named code sets**. In addition, users can choose to view the co-coding as a graph in which each code is displayed in a node (oval) and the count for number of times these codes co-occur appears along the edge; see figure 2.

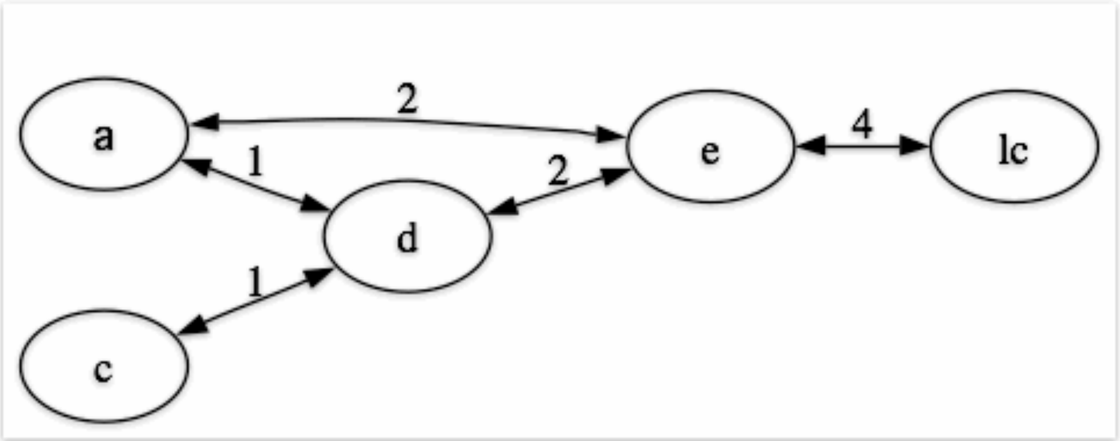


Figure 2. Graph of co-coding in a project. [2]

In figure 2, we can see that codes *a* and *e* coded overlapping segments of text twice. Of course these codes may have been used in lots of non-overlapping segments, and many more codes might be out there that are never involved in overlapping segments. The latter can be made part of the graph by checking the "show all codes" box. See figure 3.

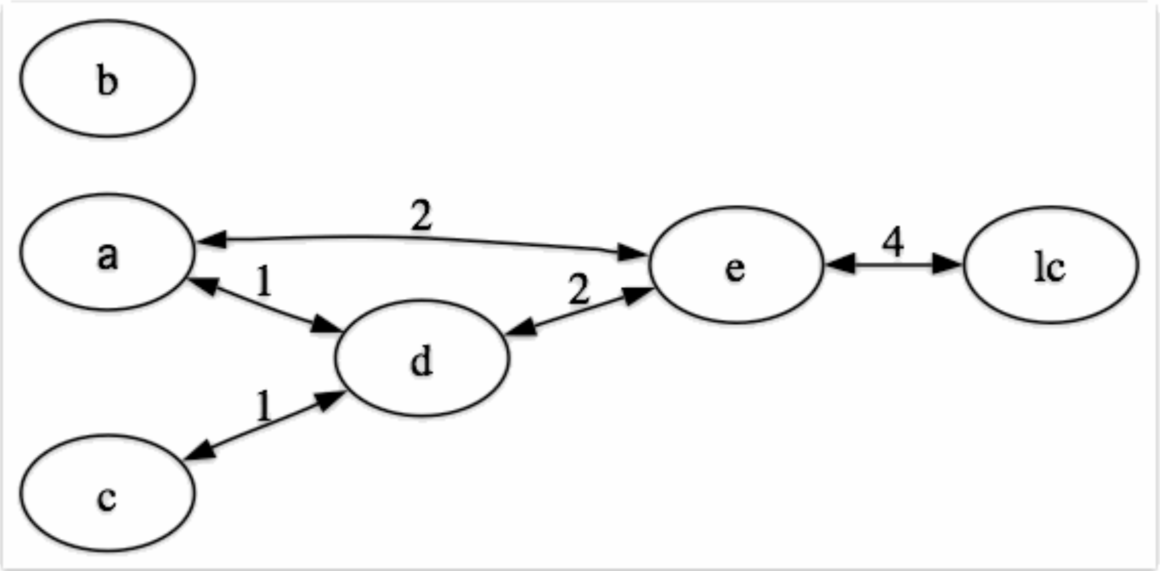


Figure 3. Co-coding graph with "All codes showing" checked

[2] I have used Graphviz.app to change the "Rank direction" using the "graph" tab to switch the direction of the graphs shown here from vertical to horizontal.

In this second case we can see that a code *b* never co-codes with any other code.

Finally we can add visual support for the number of times that items co-code by picking that we want "variable line weight" for the edges of the graph. What this means is that edges indicating "2" (i.e., that indicate that the codes they connect co-coded twice) will be twice as thick as ones labeled as "1". The resulting graph looks like figure 4.

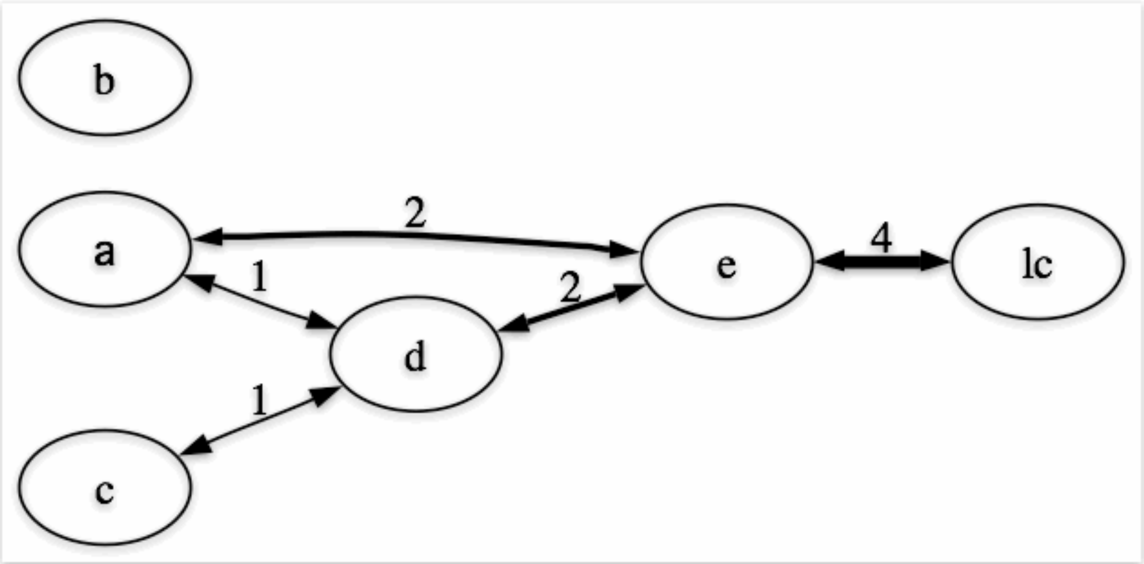


Figure 4. Graph of co-coding with variable edge weight

In figure 4, the line connecting "e" and "lc" is 4 times as thick as that connecting "a" and "d". This use of the visual to reinforce the label can really help with separating noise and signal in many circumstances.

There are three options on the variable edge menu, which appears on most dialogues for most graphing reports. These options are shown in figure 5.

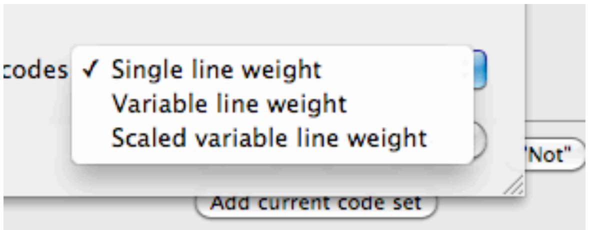
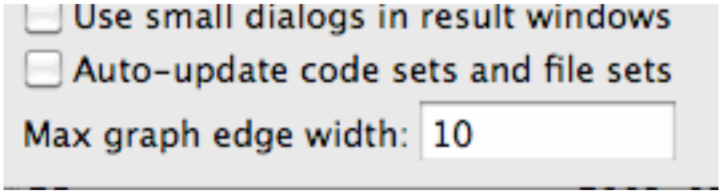


Figure 5. Variable edge menu options

The first option, "Single line weight" draws all edges as one pixel lines. Variable line weights matches the width of the edge with count of the edge. If the edge has a label with a count of 100, the line will be 100 pixels wide. This can be a problem obviously. So the third option provides a way to scale the edges to a particular width. This width is set by the user in the program preferences. See figure 6.

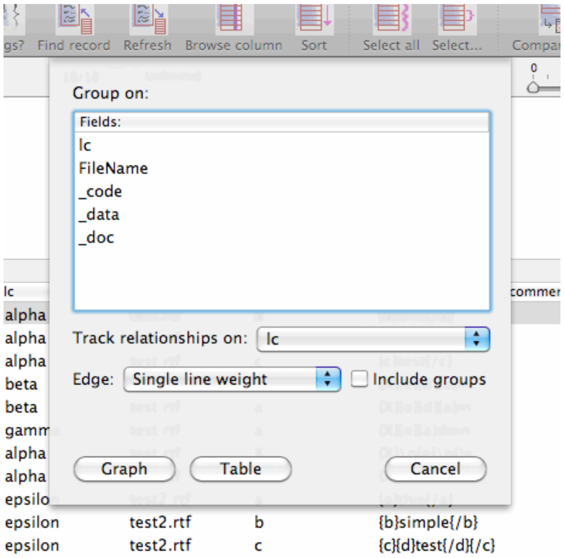


□ Use small dialogs in result windows
□ Auto-update code sets and file sets
Max graph edge width: 10

Figure 6. Option to set the maximum width of an edge in the program preferences.

2. Co-occurrence graph and chart

TAMS has had a fairly complicated chart with some residual usability for tracking co-occurrence, i.e., which codes appear in the same section of the data. This version introduces a simplified table and graph representation of the same idea. It retains the earlier report, and the new report appears under the Reports menu as "Graph/Chart co-occurences". When selected from a result window the panel shown in figure 7, appears.



Group on:

Fields:
lc
FileName
_code
_data
_doc

Track relationships on: lc

Edge: Single line weight □ Include groups

Graph Table Cancel

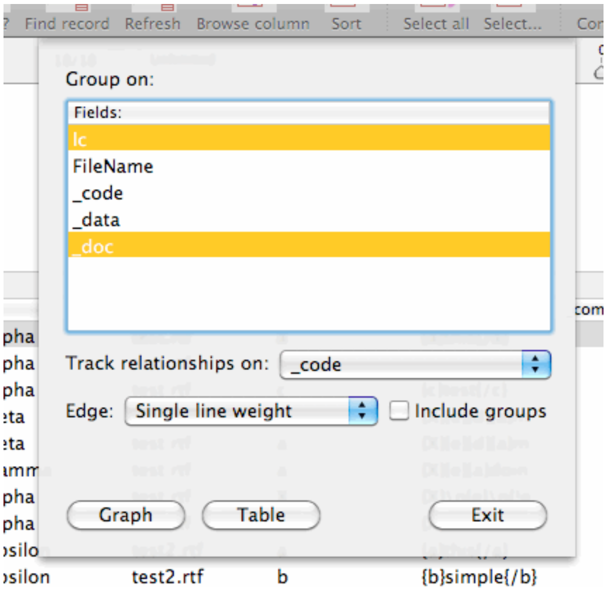
epsilon	test2.rtf	b	{b}simple{/b}
epsilon	test2.rtf	c	{c}{d}test{/d}{/c}

Figure 7. Graph/Chart co-occurrences panel

The user first picks the fields that define a section. More than one field can be chosen using the shift and apple/command keys. Note that if you want sections in different documents to be treated as different sections, select _doc or FileName (if you are using that field) in addition to other fields to make clear the different section distinctions.

The user then picks the field from the "Track relationships on" menu in which to examine co-occurrences. Usually this will be the _codes field—we want to see which codes co-occur in sections defined by the values of *lc* and _*doc*, for instance. But it does not have to be _codes, it could be any context or temporary value (or _doc, if that makes sense; you could group by code, for instance, and see (i.e., track) which docs share codes).

So using my example I will select _*doc* and *lc* to "group on." I will "track" _codes. Figure 8 shows the modified dialogue.



Group on:

Fields:
lc
FileName
_code
_data
_doc

Track relationships on: _code

Edge: Single line weight

☐ Include groups

psilon test2.rtf b {b}simple{/b}

Figure 8. Selections for the tracking of co-occurrences of _code, where _doc and lc co-define a section of the source documents.

Pressing the "Graph" key provides the output from Graphviz.app shown in figure 9.

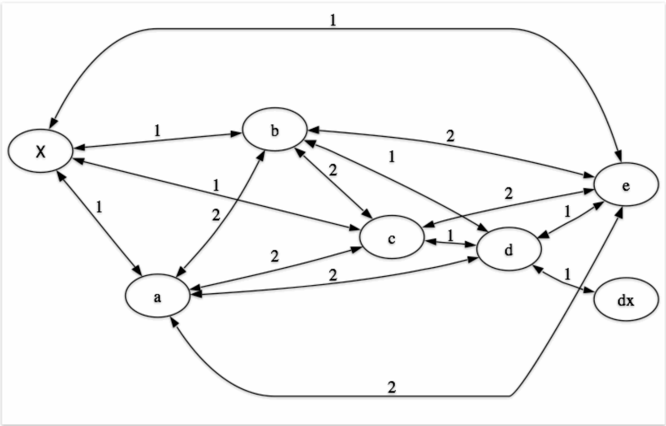


Figure 9. Graph of the co-occurrences of codes in the results

So, *a* and *d* co-occur in two sections; *d* and *e* in only one section. But which sections? We can check the "include groups" box to show on the edges the defining values of the groups for each of the co-occurrences. This results in the graph shown in figure 10.

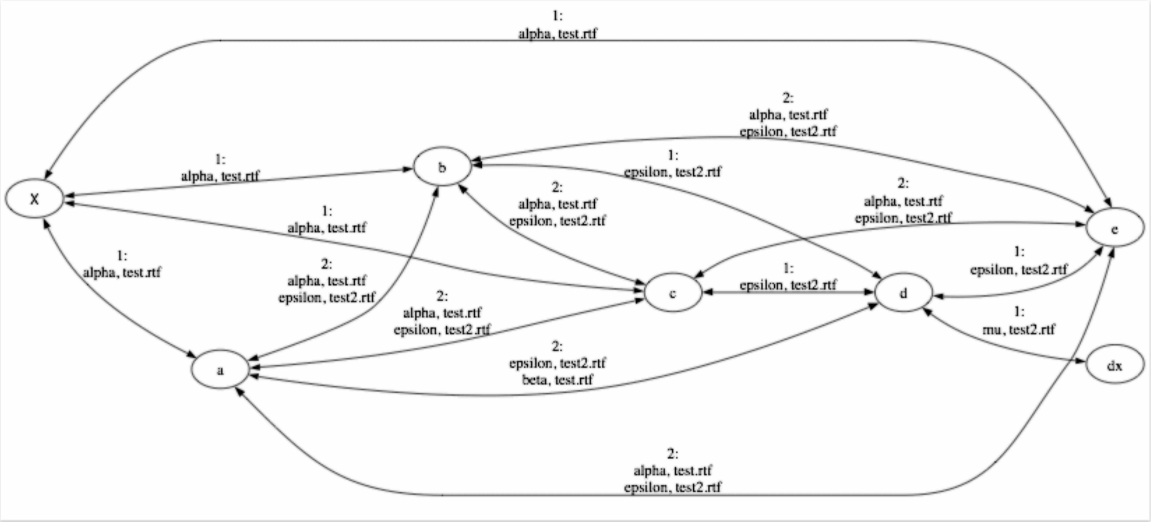
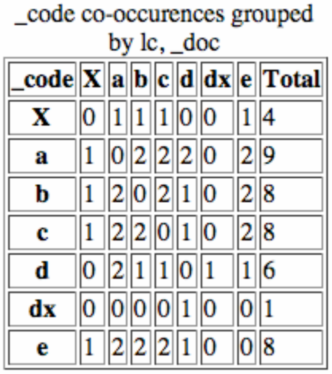

1: alpha, test.rtf
2: alpha, test.rtf epsilon, test2.rtf
1: alpha, test.rtf
b
1: epsilon, test2.rtf
X
2: alpha, test.rtf epsilon, test2.rtf
2: alpha, test.rtf epsilon, test2.rtf
1: alpha, test.rtf
e
1: alpha, test.rtf
1: epsilon, test2.rtf
2: alpha, test.rtf epsilon, test2.rtf
1: epsilon, test2.rtf
c
d
2: alpha, test.rtf epsilon, test2.rtf
1: mu, test2.rtf
2: epsilon, test2.rtf beta, test.rtf
a
dx
2: alpha, test.rtf epsilon, test2.rtf

Figure 10. Graph of co-occurrences with groups included

Because we chose two fields to define the groups, the values of both are listed, separated by commas. While hard to read in figure 10, one could find out that the single co-occurrence of *d* and *e*, for instance happened when context variable *lc* had a value of "epsilon" in file test2.rtf. The edge connecting *a* and *d* has two values listed: "beta" (a value of *lc*) in test.rtf and "epsilon" in test2.rtf. Codes *a* and *d* appear in both sections, overlapping or not.

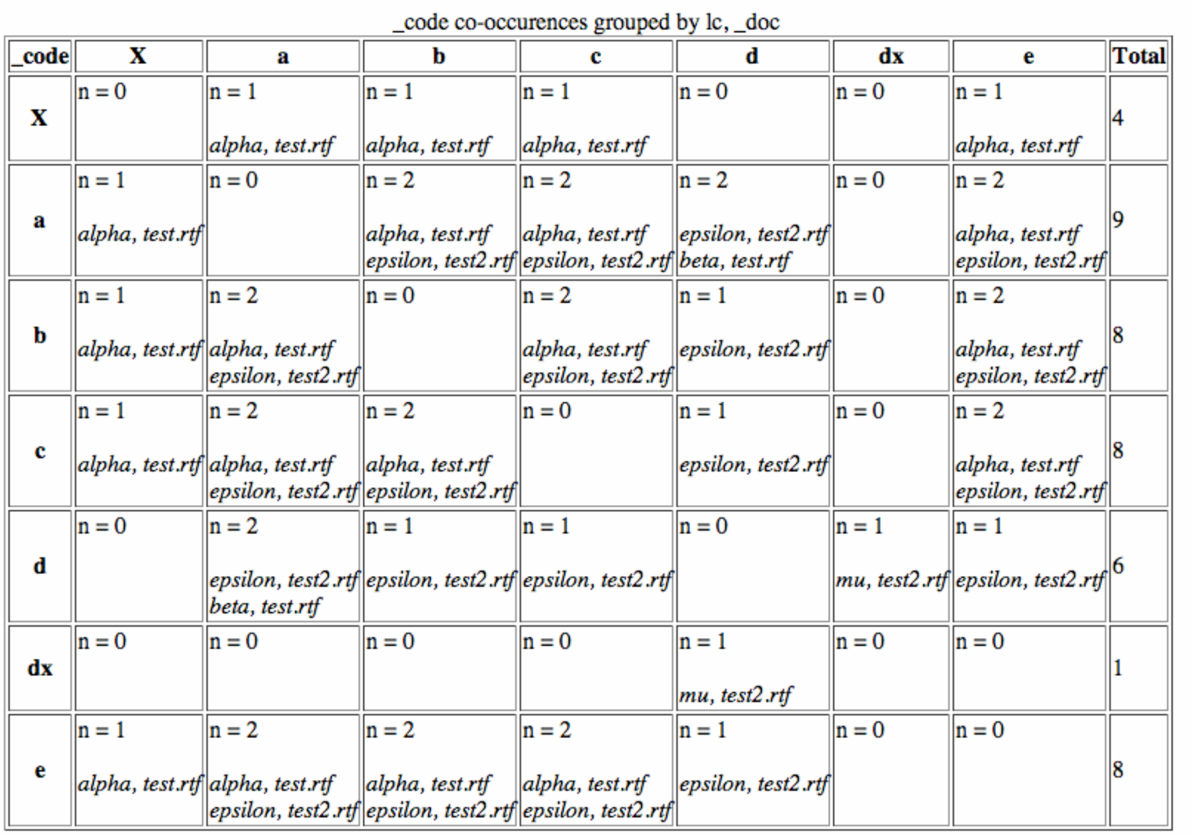
This same information can be displayed in a chart form as well. The charts are html and can be copy and pasted into recent versions of word. Figure 11, shows this same information as a chart without checking "include groups" and figure 12, shows with that box checked.



_code co-occurences grouped by lc, _doc

_code	X	a	b	c	d	dx	e	Total
X	0	1	1	1	0	0	1	4
a	1	0	2	2	2	0	2	9
b	1	2	0	2	1	0	2	8
c	1	2	2	0	1	0	2	8
d	0	2	1	1	0	1	1	6
dx	0	0	0	0	1	0	0	1
e	1	2	2	2	1	0	0	8

Figure 11. Co-occurrence chart without checking "include groups"



_code co-occurences grouped by lc, _doc

_code	X	a	b	c	d	dx	e	Total
X	n = 0	n = 1 *alpha, test.rtf*	n = 1 *alpha, test.rtf*	n = 1 *alpha, test.rtf*	n = 0	n = 0	n = 1 *alpha, test.rtf*	4
a	n = 1 *alpha, test.rtf*	n = 0	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 2 *epsilon, test2.rtf* *beta, test.rtf*	n = 0	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	9
b	n = 1 *alpha, test.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 0	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 1 *epsilon, test2.rtf*	n = 0	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	8
c	n = 1 *alpha, test.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 0	n = 1 *epsilon, test2.rtf*	n = 0	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	8
d	n = 0	n = 2 *epsilon, test2.rtf* *beta, test.rtf*	n = 1 *epsilon, test2.rtf*	n = 1 *epsilon, test2.rtf*	n = 0	n = 1 *mu, test2.rtf*	n = 1 *epsilon, test2.rtf*	6
dx	n = 0	n = 0	n = 0	n = 0	n = 1 *mu, test2.rtf*	n = 0	n = 0	1
e	n = 1 *alpha, test.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 2 *alpha, test.rtf* *epsilon, test2.rtf*	n = 1 *epsilon, test2.rtf*	n = 0	n = 0	8

Figure 12. Same chart with “include groups” checked

Note that the graphs shown in figures 9 and 10 can be rendered with variable width lines. See the discussion in section 1 of this documentation.

Important Caveat: **Co-occurrences use results tables to identify the sections that codes or other values occur in. This is a problem for coded passages that cross sections. Typically only the first section a coded passage crosses is represented (there are options for making it the last section in the program preferences and through metacodes), however at present TA does not count ALL sections as containing the coded passage, so co-occurrence analysis will only count the coded passage in the first section, not all sections it crosses. If you have codes that cross lots of sections, know that this will not be reflected in the results.**

TAMS Analyzer 3.74b7 Release Notes

Lots of improvements to existing reports, a couple of new reports, improvements to select near…

Changes

New Features:

1. Hierarchical results such as select near and group on column now have a graph and table report for viewing relationships (see documentation below)
2. Select Near matches string values (doesn't use the less/more than) (see documentation below)
3. New "code count by document" report generated from workbench (see documentation below)
4. Former code count now prompts for sort order of answer
5. New result window buttons for mark all/unmark all
6. Data Comparison Tables (DCT) options can now be saved; they are saved in the workbench so that they are available project wide. (see documentation below)
7. DCT title accepts variables for x axis, y axis, date and time (%%x, %%y, %%d, and %%t respectively) (see documentation below)

Bug fixes

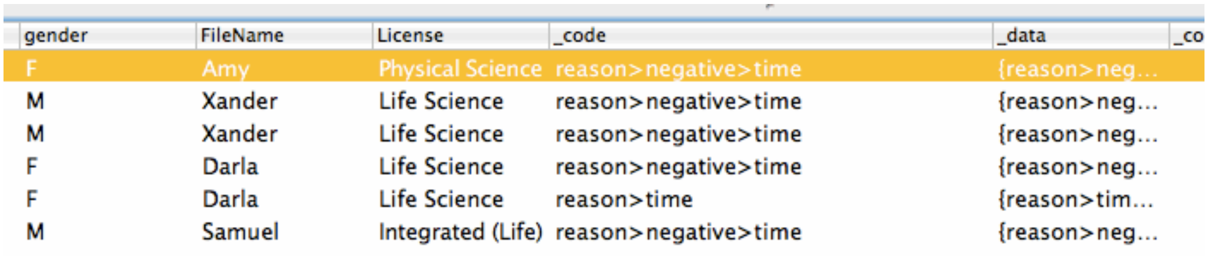
1. DCT crash that occurred if single column results were given a column name the same as a table column. This is now no longer allowed.
2. Fix to graph code sets: showed file sets rather than code sets in the dialogue

Documentation

1. Hierarchical result report

Previously there was no way to get a report from the outline view of data. The outline view is generated by commands like group column and select near. Where the DCT compares columns within a given record, the outline view tends to show relationships between records. Prior to this release users would have to export results to get representations of their data. This release presents both a graph and table view of hierarchical data.

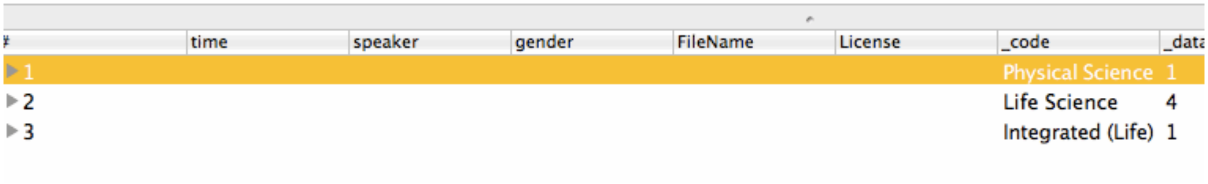
Note that the report cannot be run from the table view of your results window. You must generate an outline! We'll take the following data, group it and use it to walk through the features of the report panel:



gender	FileName	License	_code	_data	_co
F	Amy	Physical Science	reason>negative>time	{reason>neg…	
M	Xander	Life Science	reason>negative>time	{reason>neg…	
M	Xander	Life Science	reason>negative>time	{reason>neg…	
F	Darla	Life Science	reason>negative>time	{reason>neg…	
F	Darla	Life Science	reason>time	{reason>tim…	
M	Samuel	Integrated (Life)	reason>negative>time	{reason>neg…	

Figure 1. Sample Data

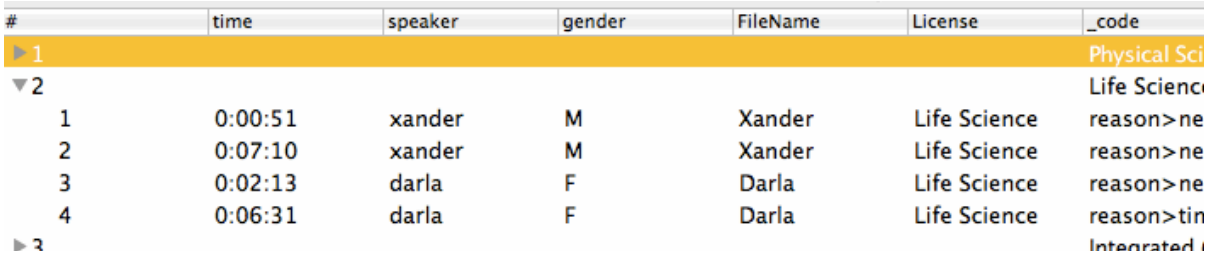
While trivial, this small data set will allow me to show (1) how to generate an outline view (this is an old feature of TA), create a simple graph of this, and a chart. I will focus on the relationship between the code and the File name, which represents an interviewees name, and their teaching license area (coded here as License). To create my outline I'm going to select the license column and pick Results->Group column. That results in this:



#	time	speaker	gender	FileName	License	_code	_data
▶ 1						Physical Science	1
▶ 2						Life Science	4
▶ 3						Integrated (Life)	1

Figure 2. Outline of the data in figure 1

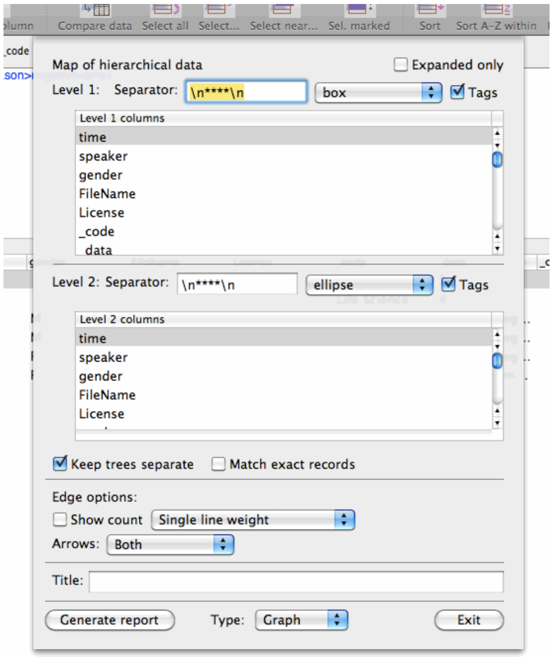
Each row holds all of the records that meet have a particular value of the grouped column. The value is shown in the _code column (and a count in the _data column). Clicking on the small triangle at the far left of the column will reveal the relevant records, see figure 3.



#	time	speaker	gender	FileName	License	_code
▶ 1						Physical Sci
▼ 2						Life Scienc
1	0:00:51	xander	M	Xander	Life Science	reason>ne
2	0:07:10	xander	M	Xander	Life Science	reason>ne
3	0:02:13	darla	F	Darla	Life Science	reason>ne
4	0:06:31	darla	F	Darla	Life Science	reason>tin
▶ 3						Integrated (

Figure 3. The outline expanded

To create a chart or graph of this data, pick Reports->Graph/Chart hierarchical data. The panel shown in figure 4 will appear.



Map of hierarchical data ☐ Expanded only

Level 1: Separator: \n****\n box ☑ Tags

Level 1 columns
- time
- speaker
- gender
- FileName
- License
- _code
- data

Level 2: Separator: \n****\n ellipse ☑ Tags

Level 2 columns
- time
- speaker
- gender
- FileName
- License

☑ Keep trees separate ☐ Match exact records

Edge options:

☐ Show count Single line weight

Arrows: Both

Title:

Generate report Type: Graph Exit

Figure 4. The Chart/Graph Hierarchical Data Pane

Level 1 refers to the first rows of the outline, the ones with the triangles. Level 2 refers to the rows that are "underneath" each first level row. You can for each level pick the fields you want to disclose (for the example above for Level 1 we would want _code and/or _data, since when you group a column they are the only fields used), you can specify how these fields are separated, the shape of the node in the graph, and you can indicate if you want tags (i.e., codes with braces around them; mostly relevant if you are using your _data field).

Picking _code for "level 1" and "FileName" for "level 2" but otherwise leaving the choices as indicated above, I would get the graph from Graphviz.app shown in figure 5.

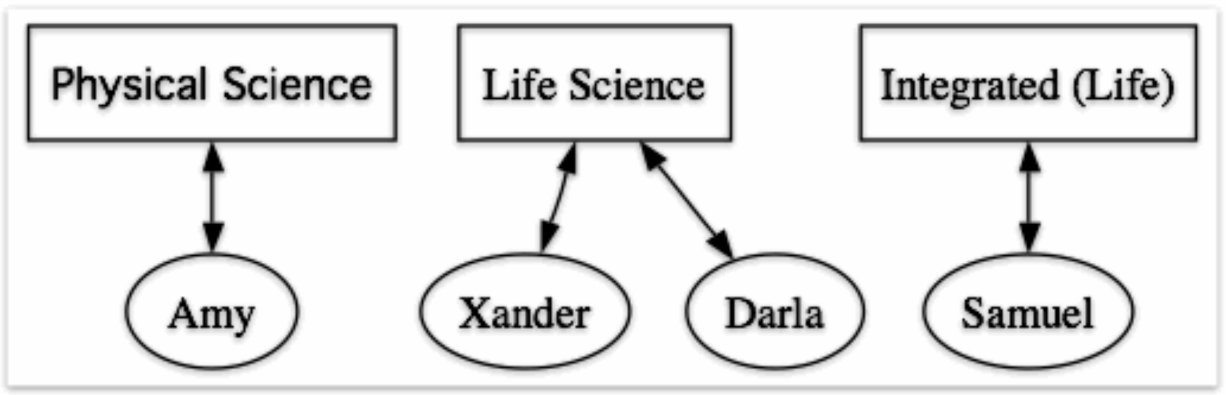
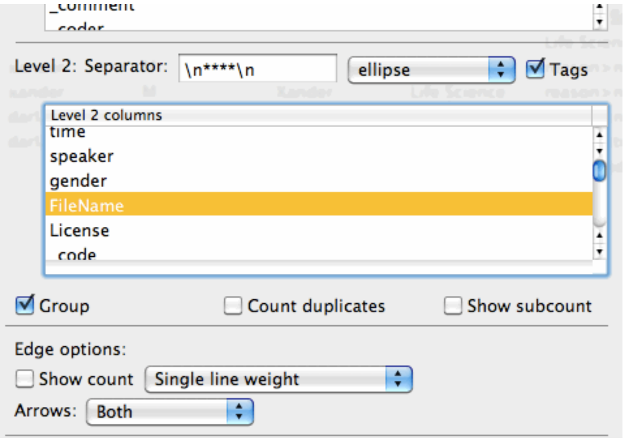


Figure 5. Graph of hierarchical data

The boxes show the level 1 values, the ovals the level 2 values. In this example none of the elements at level 1 or level 2 are identical (there is no Amy in Life Science, for instance). I could show the crossing of these trees if values are not identical, by unchecking the "keep trees separate." I can add a further constraint on this mix and matching of level 1s and level 2s by insisting that not only must the values be the same but they must actually refer to the exact same rows—in group column that is not possible, but it is a possibility if you use select near to create your outline.

Finally, you have a variety of line and edge options including, showing count, revealing count through line thickness, and the direction of the edges (arrows).

To produce a table, pick table from the menu at the bottom of the panel. Different options appear under the level 2 fields (see figure 6).



_comment
coder

Level 2: Separator: \n****\n ellipse ☑ Tags

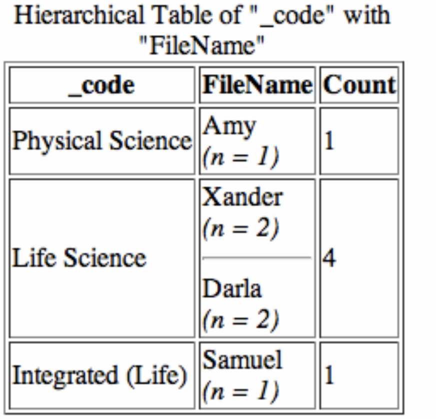
Level 2 columns
time
speaker
gender
FileName
License
code

☑ Group □ Count duplicates □ Show subcount

Edge options:
□ Show count Single line weight
Arrows: Both

Figure 6. Table options

Here you can choose to group values and count duplicates or not (as well as show subcounts). Note that the show count option under edges is a working switch for tables—it will provide totals. In the table shown in figure 7, I have checked all 4 options: group, count duplicates, show subcount, and show count.



Hierarchical Table of "_code" with "FileName"

_code	**FileName**	**Count**
Physical Science	Amy *(n = 1)*	1
Life Science	Xander *(n = 2)* Darla *(n = 2)*	4
Integrated (Life)	Samuel *(n = 1)*	1

Figure 7. Table of hierarchical data view

Column 1 represents the level 1s, the second column shows the related values of level 2 (along with sub counts), and the totals are shown in the last column (activated by the "show count" switch in the "Edges options").

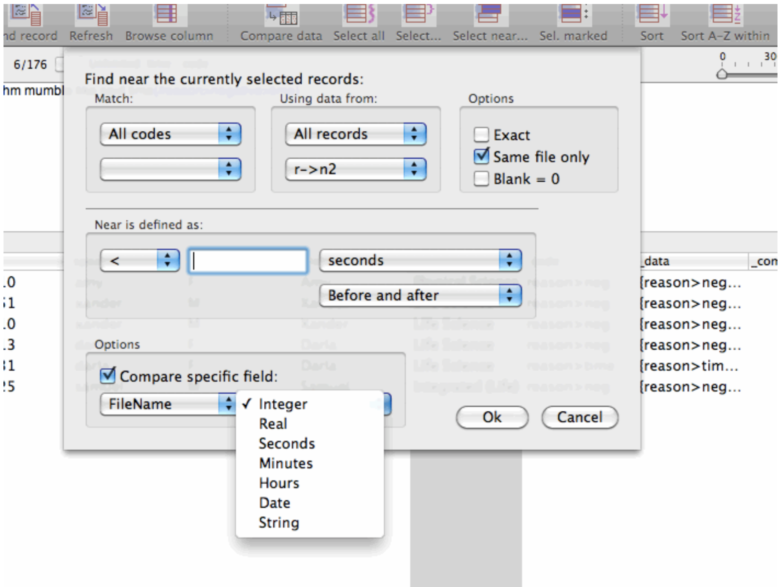
You can also set the title of the chart yourself by filling in the "Title" field (see bottom of figure 4). This title can use the variable fields described in section 4 below. The variable %%x will be substituted with the title of column 1 (level 1) and %%y will be substituted with the title of column 2 (level 2). %%d and %%t will be replaced with the date and time respectively.

These two representations allow you to view and analyze the relationships between records.

2. Using select near with string values

TAMS Analyzer's results windows are full-featured flat file data bases with complex sorting, selecting and reporting capabilities. With such databases, the relationship between rows is often difficult to obtain. Co-occurrence reports represent one way of getting at this information. More subtlety can be squeezed out through a new feature of select near. Here is a quick review of how this feature works. With select near you start with a base set (the current selection), and then match a comparison set with it (all the data, or a subset). This creates an outline view in which the level ones are the rows from the base set. In previous releases the criteria for matching the comparison set with the base set of rows were basically numeric: find rows within 30 seconds of the base set, find rows less than 200 characters from the records in the base set, etc.

This release adds one option that is of a different flavor. You can now match rows in the comparison set with the base set if they share a value in a specified field. You will pick the base and comparison sets exactly the same way. The box marked "Near is defined as" is ignored entirely, as--at least in this release (see figure 8). Then you pick the field that must contain the same value. Note that the strings that define the mapping of the comparison set to the base set must be exact. Check the box marked "Compare specific field." Then pick the field to compare from the menu just below the check box. Finally from the second menu below the box pick "String" (shown in the pull down menu in figure 8).



Find near the currently selected records:

Match: All codes

Using data from: All records; r->n2

Options: □ Exact ☑ Same file only □ Blank = 0

Near is defined as: < seconds; Before and after

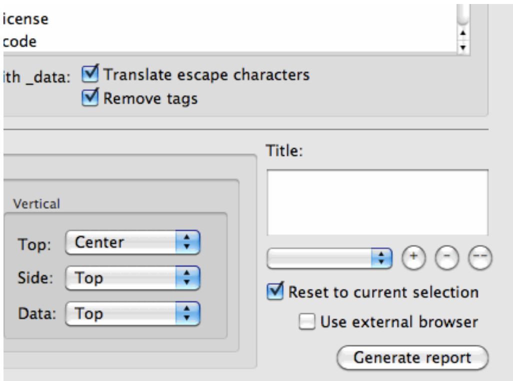
Options: ☑ Compare specific field: FileName

Figure 8. Revised Select Near panel

In the panel shown in figure 8, comparison set rows will be matched to base set rows if their value for Filename is the same.

3. Saving Data Comparison Table options

Now the options in a Data Comparison Table can be saved. The various choices are stored in the project file, not in the results file, so that they are available project wide. The table has to be given a unique title (see figure 9). Then they are saved using the same system as search lists: + means save this DCT, - means remove the DCT selected in the pull down menu, -- means delete all saved DCTs.



icense
code
ith _data: ☑ Translate escape characters
☑ Remove tags

Vertical
Top: Center
Side: Top
Data: Top

Title:

☑ Reset to current selection
□ Use external browser

Figure 9. The corner of the DCT panel with
the storage and management options for saving DCTs

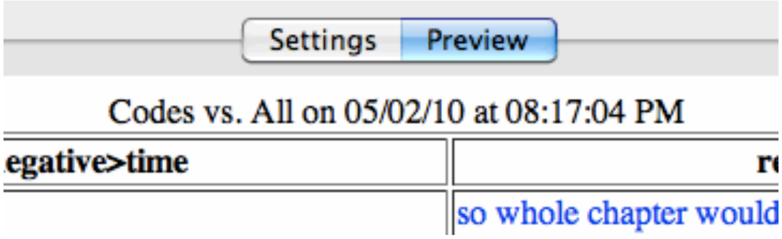
To load a saved DCT simply choose it from the pop up menu beneath the title.

Note that all that is saved are the choices for the X and Y axes, the fields and options for the data elements, the formatting for the cells, and whether the axes are switched. The current data, its sort order and selections, and creation of temporary columns are not saved and will have to be recreated before running the DCT dialogue and picking a saved DCT. You're not storing the data for the report, only the variable choices.

Also note that the current axes (not shown in figure 9), "Don't underline links" switch, the "use external browser," and "reset to current selection" options are **not** saved.

4. Variables in the title field of DCTs

The title of DCTs specified in the box shown in figure 9, now supports four variables which are expanded when the table is created. These include variables for the name of the X axis (%%x), the Y axis (%%y), as well as the current date (%%d) and time (%%t). The date format is specified by the format option specified by Results->Sort options->Date format. Note that you must select a row and not a column of your results window when specifying the date format (the former creates the default date format, the latter the specific format for the selected column). Figure 10 shows an example of a table produced with the title "%%x vs. %%y on %%d at %%t."



Codes vs. All on 05/02/10 at 08:17:04 PM

egative>time	r
	so whole chapter would

Figure 10. Title of a table with variables expanded

5. Code count by file

There's a new code count by file that's executed from the workbench. It's very simple and doesn't need any walk through, you just choose how you want the data sorted. Here is the only important thing to note: the program uses the search list to select the files, and the hot code set to select the code. You can use these two lists to limit the inputs to your counts.

TAMS Analyzer 3.72b10 Release Notes

This release fixes one bug that prevented Tiger users (OSX 10.4) from accessing the Data Comparison Report pane. The multiuser release includes a fix that allows rtfd files to be shared.

Interface changes

1. Graph/Chart Sets To Data Report now uses a TAMS window to display the report rather than opening the report in a browser (e.g. Safari).

Bug fixes

1. Multiuser tams can now share rtfd files
2. Tiger users can now access the Data Comparison Table