SpecTcl Filter Output

Shane Watters

Graduate Research Assistant
watterss@frib.msu.edu

Version 3, 02/09/2024

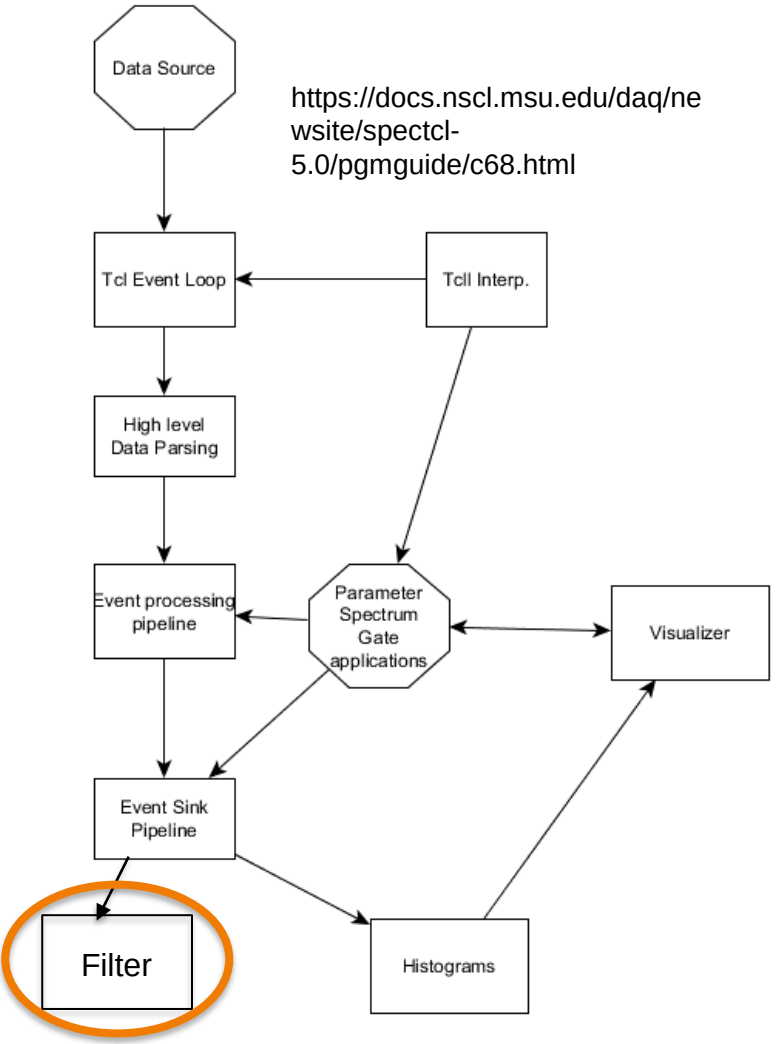
MICHIGAN STATE UNIVERSITY

U.S. DEPARTMENT OF ENERGY | Office of Science

Outline

1. SpecTcl’s processing model
2. SpecTcl’s filtering utility and filtered event files
3. Changes to SpecTcl’s filtering utility
4. Creating filters
5. Uses of filtered data

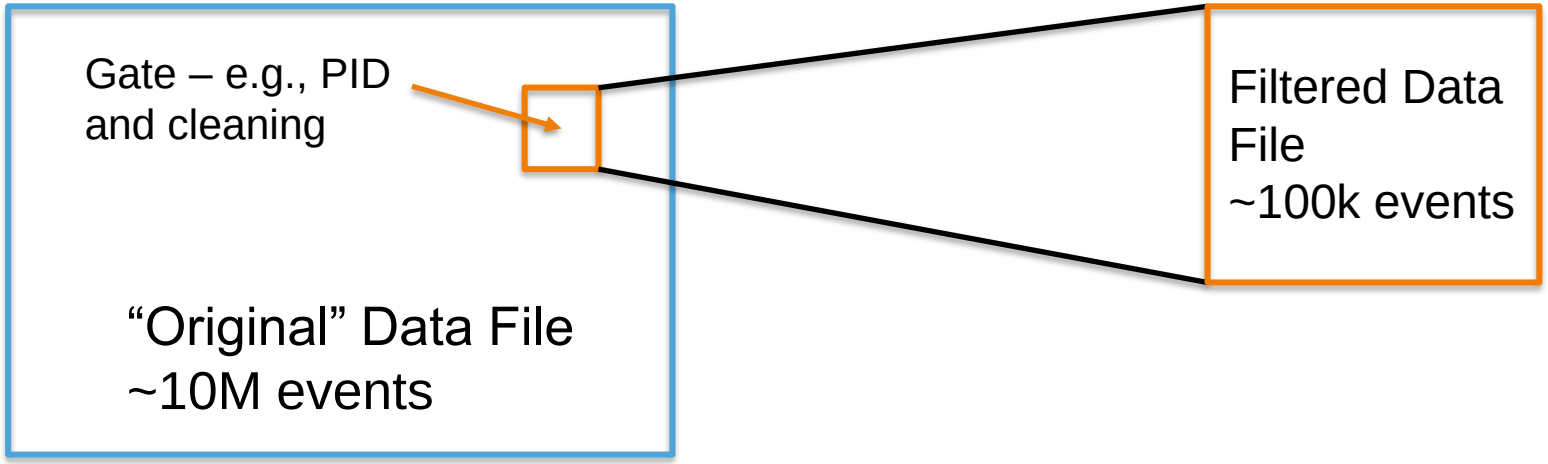
SpecTcl Processing Model



- SpecTcl's processing model is summarized below:
  - The **Tcl Event Loop** processes events that are taken from the Data Source (either Online or a .evt file) and dispatches them to the appropriate handlers
  - The **High Level Data Parsing** unpacks blocks of data into event units. These units are then passed one-by-one to the Event Processing Pipeline
  - The **Event Processing Pipeline** transforms the raw events into meaningful parameters. This step can consist of an arbitrary number of stages
  - Once the data has been passed through the Event Processing Pipeline, it is passed to the **Event Sink**. This is where Histograms and Filters live
  - From there, the **Histograms** interface with the Visualizer (Xamine or SpecTk), which concurrently has access to the Parameters
  - Describing the **Filters** is the focus of this presentation

Filtered Event Files

- The primary intention of the filter is to reduce large data files to more manageable sizes.
- However, this can be modified to extract specific information from SpecTcl – the purpose of this presentation.
- A filtered file will extract a subset of the ‘original’ file based on an event’s satisfaction of a user-defined gate.



“Out of the Box” SpecTcl Filters

- SpecTcl comes pre-built with a filter utility.
- This filter utility generates filtered files that SpecTcl is able to read (extension .flt).
- In order to use these filtered files, a Filtered Event Processing Pipeline must be established.
  - This requires a filter buffer decoder, as the filtered events cannot be parsed by the original data unpackers.
  - Without a separate processing pipeline, SpecTcl will be unable to reread the filtered events.
- The purpose of the CSVFilterOutputStage is to modify the Filter Output in the Event Sink to allow SpecTcl to generate (human) readable data files.
- Beginning with ARIS SpecTcl v.25, the necessary code has been added to generate readable event files.
  - A working Filtered Event Processing Pipeline has not yet been implemented.

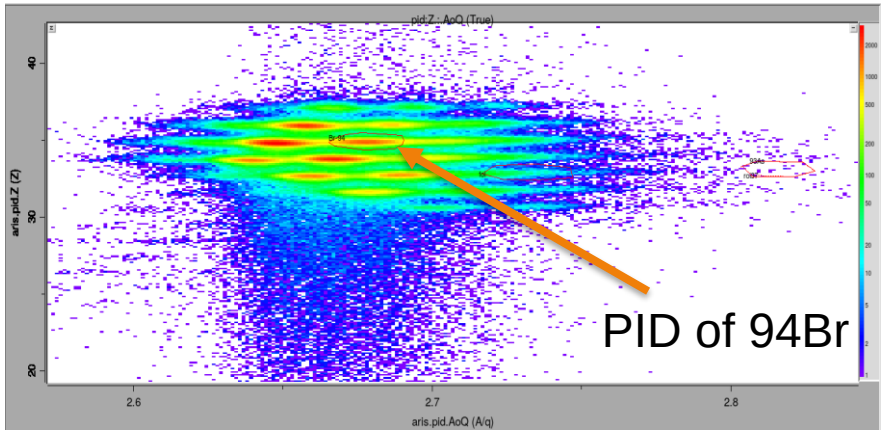
CSVFilterOutputStage

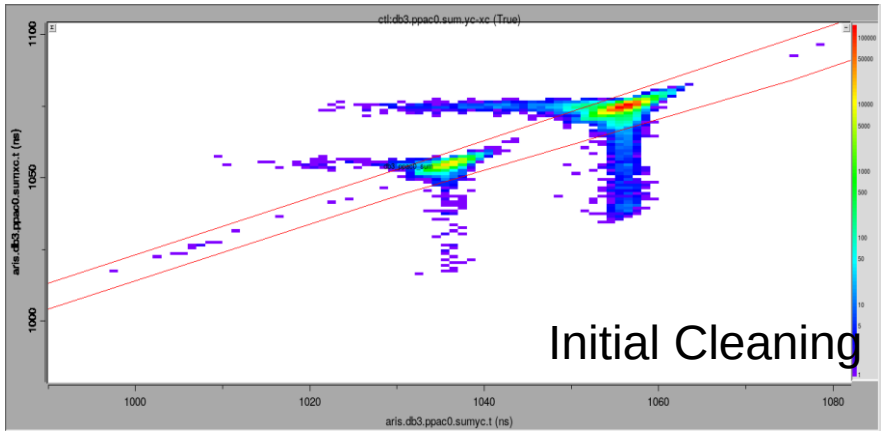
- The code that defines how SpecTcl should handle the data as it is being processed by the filter.
- It is a simple code that creates an arbitrary number of columns (equal to the number of parameters that need to be extracted) and places the events in rows.

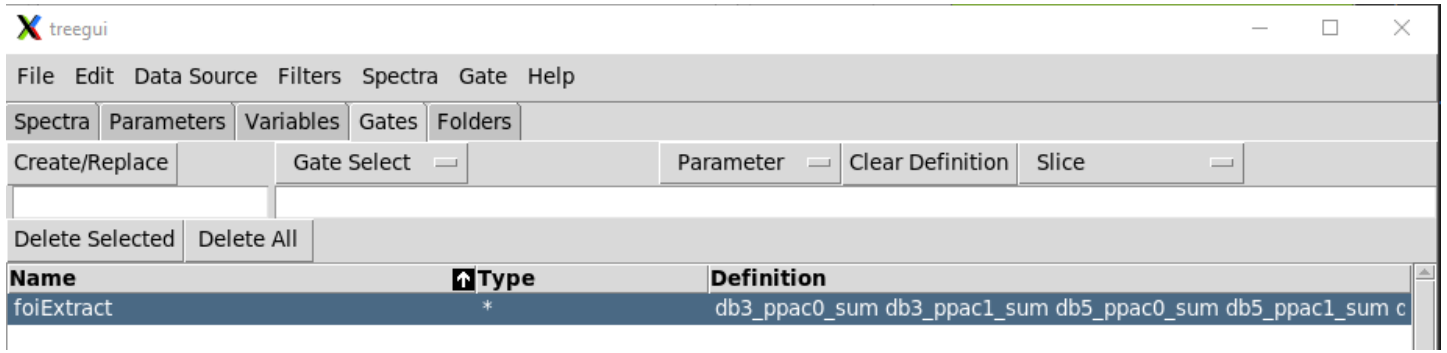
```
//Each event is described by parameter names and parameter IDs. If the piece of data is not the last one in the line,
//it's appended with ','; if it is, it's \n (new line).
void
CSVFilterOutputStage:: DescribeEvent(std::vector<std::string> parameterNames,
             std::vector<UInt_t>      parameterIds)
{
  for (int i = 0; i < parameterNames.size(); i++) {
    char delim = ',';                       // Delimeter for all but end.
    if ((i+1) == parameterNames.size()) {
      delim = '\n';                                 // Last one has a newline following it.
    }
    m_file << parameterNames[i] << delim;
  }

  m_params = parameterIds;
}
```

Creating SpecTcl Filters

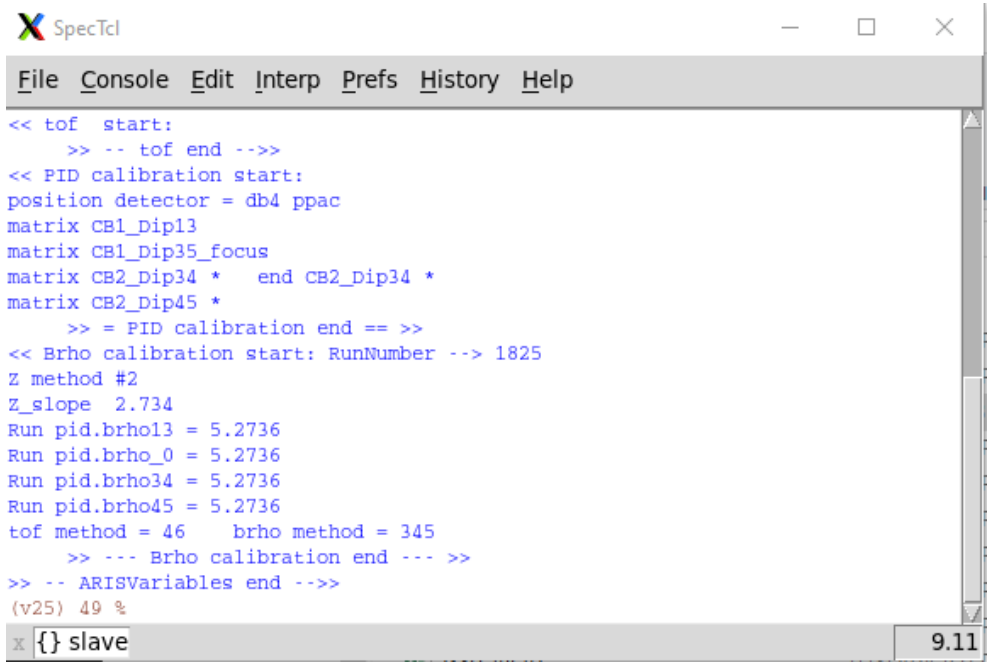
1. In SpecTcl, begin by reading in the data you wish to work with.
2. Apply any cleaning and calibrations to the data before continuing.
3. Create a gate(s) which will define which events are extracted.
   - The gate can be an AND gate, meaning the events must pass multiple requirements before being extracted
   - Normally, I name the gate-to-pass ‘foiExtract’







Creating SpecTcl Filters (Cont.)



4. Inside the SpecTcl console, type* the following series of commands (or run TCL batch file with them):

- filter –new [filter name] [gate name] {parameters}
  - » "filter –new 94Br_v1 foiExtract {**aris.db3.x aris.db3.a** …}"
  - » Note that the parameter names should match those in treegui.

- filter –file [file name] [filter name]
  - » "filter –file 94Br_v1.csv 94Br_v1"

- filter –format [filter name] csv
  - » "filter –format 94Br_v1 csv"

- filter –enable(disable) [filter name]

Filter Output

- Once the filter has been created and enabled, reopen the data file. As part of the Event Sink, the data points that satisfy the gate requirements will be extracted into a csv format.
- Below is an example of the filter output:

  - Note that the order in which you specify the parameters is the order in which they appear. The csv files are organized by row. Event 1 is the first row, with the associated parameters in the following columns. If an event is missing a parameter, then it will appear empty.
  - In this example, the original data file had 996,037 events. The filtered file (PID and first-pass cleaning) has only 82,329 events.

1	aris.db3.x	aris.db3.a	aris.db3.y	aris.db3.b	aris.db5.x	aris.db5.a	aris.db5.y	aris.db5.b	aris.pid.b	aris.db4.p	aris.db4.p	aris.db1.x	aris.db1.a	aris.db1.y	aris.db1.b
2	-0.17472	-4.22252	-1.32453	4.87663	-7.8732	-28.4301	-4.60181	-8.55637	0.532887	-10.6338	0.883345				
3	-6.25682	-5.56811	-0.77411	-0.04032	2.15927	5.32452	-3.07981	-7.37264	0.533664	0.297423	-0.03216				
4	12.065	29.6164	-0.25058	2.28246	-10.3988	-20.3391	-0.92337	-3.77772	0.532734	-20.9446	0.280654				
5	-2.28518	-7.47839	1.14501	-4.93327	-7.14201	-26.8926	0.973176	-2.85531	0.532251	-7.23521	-1.39242				
6	3.88073	4.03614	-1.34157	2.1904	-12.6891	-34.394	-2.95618	-4.66775	0.532063	-4.74386	0.50555				
7	-14.5125	-12.2183	2.37236	-8.11937	9.5435	27.9925	3.42415	0.355952	0.538363	8.69135	-3.20231				
8	6.26257	18.7143	-4.29713	10.5488	-5.3343	-9.20091	-7.7518	-11.2545	0.533628	-15.4384	3.36501				
9	-5.59505	-5.76575	-0.05218	2.26197	-1.53379	-5.30523	-2.2528	-5.59688	0.532066	-3.59779	-0.68854				
10	3.78417	13.0588	0.961121	-12.4221	-6.43681	-11.6185	6.10648	4.68669	0.533657	-6.81208	-3.93809				
11	0.482991	8.52205	-3.29862	3.5743	-2.67859	-3.69	-5.00183	-9.09985	0.533576	-7.08897	2.5122				
12	5.10082	16.1063	-3.57175	-1.9076	-5.99466	-11.1452	-3.0294	-6.5451	0.53336	-13.2905	1.63639				
13	-6.46504	-13.5128	-2.85624	4.34721	-1.69918	-11.9315	-4.593	-7.14337	0.532817	1.52004	1.79707				
14	-3.08611	-0.77025	3.00465	-10.6509	-1.30932	-3.3966	4.83353	1.77177	0.534489	-3.99236	-4.55974				

DB1 PPACs were not in

Filtered Data Uses

- Example of regression
- LISE Ray Reader for 2D/3D plots

Filter Features

- It is important to note that SpecTcl will have the filter enabled until SpecTcl is closed or it is disabled. This is a feature that allows you to run multiple data files through the filter quickly.
- The filter definitions are stored in the def file.
  - To delete filters, type “filter –delete [filter name]” in the command line
  - To see what filters you currently have, type “filter –list”
- Any number of gates can be used to define the filter. If one wishes to read out all the data in a run, then the ‘True’ gate may be used in the filter definition.

Final Comments

- Using the SpecTcl filter feature to extract event-by-event information has proven useful to run regressions on the data to extract transfer matrices.
- The CSVFilterOutputStage code created the formatting to allow SpecTcl to create csv files.
  - Using this file as a template, one could direct SpecTcl to create the output in any number of formats or extensions.
- A working Filtered Event Processor has not yet been implemented, but would greatly reduce the analysis time of large files.

Plotting data with LISE Ray Reader

- https://lise.nscl.msu.edu/download/download.html
  "LISE Ray Reader" by I.Richardson
  (https://lise.nscl.msu.edu/download/Windows/other/LISE_RayReader/Ray-Reader-BETA-1.0.exe)

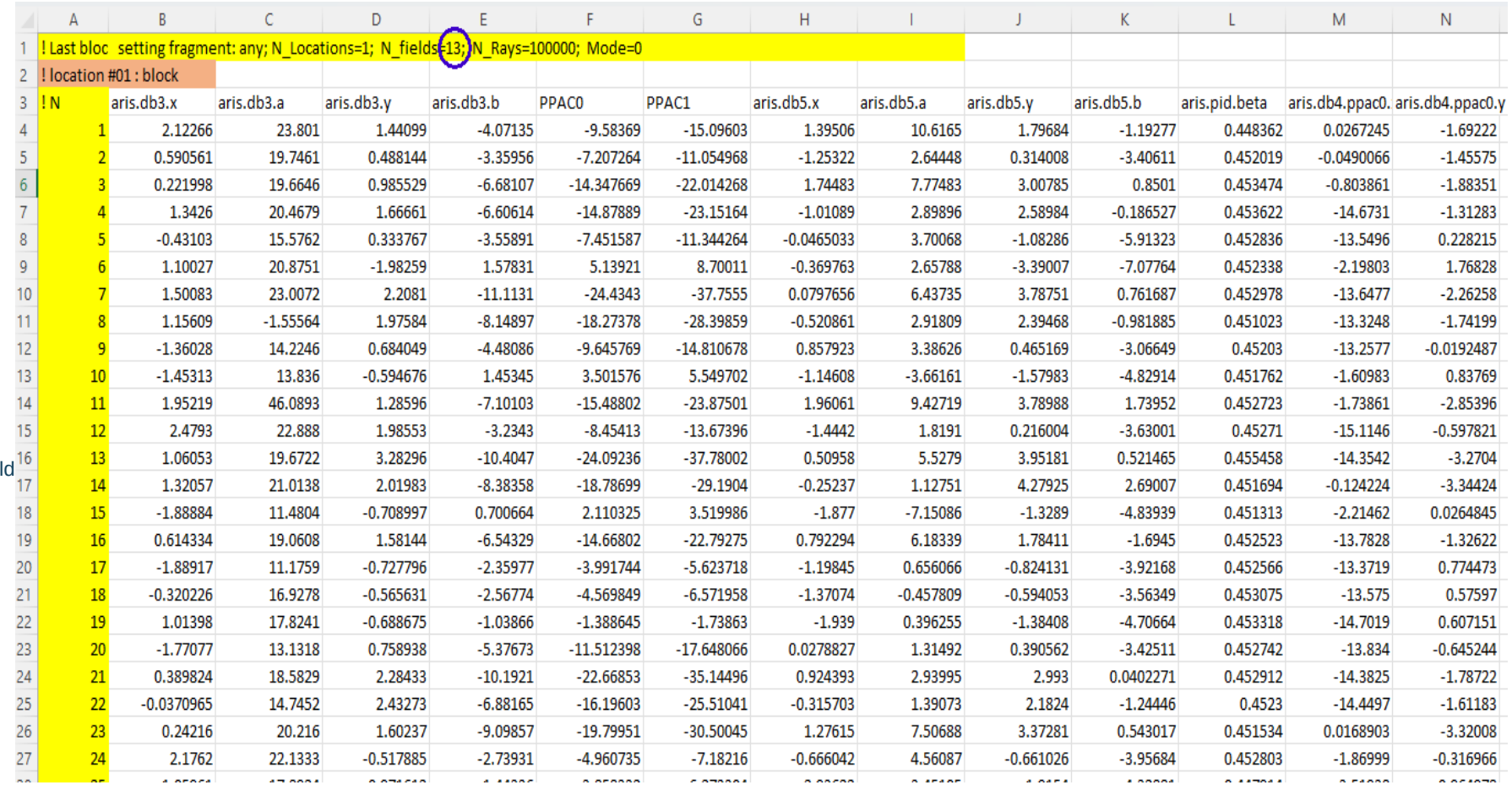
- It's beta-version. Serious improvement are still required, including the installation procedure

- LISE Ray Reader's documentation: https://lise.nscl.msu.edu/doc/ray-reader-documentation/

- Your data file (separated by comma) should be modified to "LISE ray reader" format (see next slide)

Input file for LISE Ray Reader

- One column and two rows (marked by yellow and red colors) should be inserted in your comma-separated data file
- Modify the number of fields corresponding to your format
(see blue open circle)
- Text file example



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	! Last bloc setting fragment: any; N_Locations=1; N_fields=13; N_Rays=100000; Mode=0													
2	! location #01 : block													
3	! N	aris.db3.x	aris.db3.a	aris.db3.y	aris.db3.b	PPAC0	PPAC1	aris.db5.x	aris.db5.a	aris.db5.y	aris.db5.b	aris.pid.beta	aris.db4.ppac0.	aris.db4.ppac0.y
4	1	2.12266	23.801	1.44099	-4.07135	-9.58369	-15.09603	1.39506	10.6165	1.79684	-1.19277	0.448362	0.0267245	-1.69222
5	2	0.590561	19.7461	0.488144	-3.35956	-7.207264	-11.054968	-1.25322	2.64448	0.314008	-3.40611	0.452019	-0.0490066	-1.45575
6	3	0.221998	19.6646	0.985529	-6.68107	-14.347669	-22.014268	1.74483	7.77483	3.00785	0.8501	0.453474	-0.803861	-1.88351
7	4	1.3426	20.4679	1.66661	-6.60614	-14.87889	-23.15164	-1.01089	2.89896	2.58984	-0.186527	0.453622	-14.6731	-1.31283
8	5	-0.43103	15.5762	0.333767	-3.55891	-7.451587	-11.344264	-0.0465033	3.70068	-1.08286	-5.91323	0.452836	-13.5496	0.228215
9	6	1.10027	20.8751	-1.98259	1.57831	5.13921	8.70011	-0.369763	2.65788	-3.39007	-7.07764	0.452338	-2.19803	1.76828
10	7	1.50083	23.0072	2.2081	-11.1131	-24.4343	-37.7555	0.0797656	6.43735	3.78751	0.761687	0.452978	-13.6477	-2.26258
11	8	1.15609	-1.55564	1.97584	-8.14897	-18.27378	-28.39859	-0.520861	2.91809	2.39468	-0.981885	0.451023	-13.3248	-1.74199
12	9	-1.36028	14.2246	0.684049	-4.48086	-9.645769	-14.810678	0.857923	3.38626	0.465169	-3.06649	0.45203	-13.2577	-0.0192487
13	10	-1.45313	13.836	-0.594676	1.45345	3.501576	5.549702	-1.14608	-3.66161	-1.57983	-4.82914	0.451762	-1.60983	0.83769
14	11	1.95219	46.0893	1.28596	-7.10103	-15.48802	-23.87501	1.96061	9.42719	3.78988	1.73952	0.452723	-1.73861	-2.85396
15	12	2.4793	22.888	1.98553	-3.2343	-8.45413	-13.67396	-1.4442	1.8191	0.216004	-3.63001	0.45271	-15.1146	-0.597821
16	13	1.06053	19.6722	3.28296	-10.4047	-24.09236	-37.78002	0.50958	5.5279	3.95181	0.521465	0.455458	-14.3542	-3.2704
17	14	1.32057	21.0138	2.01983	-8.38358	-18.78699	-29.1904	-0.25237	1.12751	4.27925	2.69007	0.451694	-0.124224	-3.34424
18	15	-1.88884	11.4804	-0.708997	0.700664	2.110325	3.519986	-1.877	-7.15086	-1.3289	-4.83939	0.451313	-2.21462	0.0264845
19	16	0.614334	19.0608	1.58144	-6.54329	-14.66802	-22.79275	0.792294	6.18339	1.78411	-1.6945	0.452523	-13.7828	-1.32622
20	17	-1.88917	11.1759	-0.727796	-2.35977	-3.991744	-5.623718	-1.19845	0.656066	-0.824131	-3.92168	0.452566	-13.3719	0.774473
21	18	-0.320226	16.9278	-0.565631	-2.56774	-4.569849	-6.571958	-1.37074	-0.457809	-0.594053	-3.56349	0.453075	-13.575	0.57597
22	19	1.01398	17.8241	-0.688675	-1.03866	-1.388645	-1.73863	-1.939	0.396255	-1.38408	-4.70664	0.453318	-14.7019	0.607151
23	20	-1.77077	13.1318	0.758938	-5.37673	-11.512398	-17.648066	0.0278827	1.31492	0.390562	-3.42511	0.452742	-13.834	-0.645244
24	21	0.389824	18.5829	2.28433	-10.1921	-22.66853	-35.14496	0.924393	2.93995	2.993	0.0402271	0.452912	-14.3825	-1.78722
25	22	-0.0370965	14.7452	2.43273	-6.88165	-16.19603	-25.51041	-0.315703	1.39073	2.1824	-1.24446	0.4523	-14.4497	-1.61183
26	23	0.24216	20.216	1.60237	-9.09857	-19.79951	-30.50045	1.27615	7.50688	3.37281	0.543017	0.451534	0.0168903	-3.32008
27	24	2.1762	22.1333	-0.517885	-2.73931	-4.960735	-7.18216	-0.666042	4.56087	-0.661026	-3.95684	0.452803	-1.86999	-0.316966

"! Last block ""any""", setting fragment: any; N_Locations=1; N_field
! location #01 : block,,,,,
! N,aris.db3.x,aris.db3.a,aris.db3.y,aris.db3.b,PPAC0
1,2.12266,23.801,1.44099,-4.07135,-9.58369
2,0.590561,19.7461,0.488144,-3.35956,-7.207264
3,0.221998,19.6646,0.985529,-6.68107,-14.347669
4,1.3426,20.4679,1.66661,-6.60614,-14.87889
5,-0.43103,15.5762,0.333767,-3.55891,-7.451587

Plotting data with LISE Ray Reader: results