Coming to ARIS SpecTcl v.30

Oleg B. Tarasov, Elaine Kwan

18 April 2025

Research presentation

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics and used resources of the Facility for Rare Isotope Beams (FRIB) Operations, which is a DOE Office of Science User Facility under Award Number DE-SC0023633, and by the US National Science Foundation under Grants No. PHY-20-12040 and 23-10078 “Windows on the Universe: Open Quantum Systems in Atomic Nuclei at FRIB”.

Content

Main purpose is to push aside “debug and heavy” source and settings, easy switch between user and debug settings

- #define directives for “light” source
- pid_values.csv re-organization
- PID fine tuning
- Tof variables and parameters (cpp,cvs) re-organization
- Z-calculation based on dE, dZ - variable
- Ztke revision
- Channel Assign and packets update
- Others

- List of modifications

Directives

It’s recommended to comment the next definitions for the operation version

CAris_pid.h

```
#define _AnalysisZTKE      // Z-calculation based on dE-TKE-Brho without ToF use (Ztke or Ztimeless)
#define _deltaZ            // Fast check of Z-stabilty. Do not use it for gates!
#define _E_CORRECT         // dEnergy correction for dispersion. No matrix and Brho use
#define _AnalysisQint      // A/q and Brho recalculation based on q-integer value
```

CAris_tof.h

```
#define _useDeltaAnalysis  // delta ToF between 34 and 45 ppacs (e.g. db34.ppac0.xy-db45.ppac0.xy) ,
                           // and between db5 detectors (ppac-pin-sci)
```

pid_values.csv re-organization

pid_values.csv → pid_values.csv
tke_sci_values.csv

first symbol in line comments this line

Global tof offset.
It's very flexible to use with tof_average

Flag to use PID fine tuning variables

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	run_start	run_stop	Z_method	tof_mehod	brho_method	Z_slope2	Z_slope	Z_offset	Z_log_beta	tkebox	tof_offset	fine	L34_correct	L45_correct	Q_offset	exp #	notes	
2	#3186	4186	0	96	45	0	7.35	-0.32	1	5	-1.2	0	0	0	0	all		tof96
3	3186	4186	0	94	1	0	7.35	-0.32	1	5	-2.5	0	0	0	0	all		tof94
4	#3186	4186	1	96	45	0	7.22	0	1	5	-1.2	0	0	0	0	all		tof96
5	#3186	4186	1	94	45	0	7.22	0	1	5	-2.5	0	0	0	0	all		tof94
6	#----------																	
7	#3219	4186	0	94	345	0	4.19	0.45	1	5	-1.9	1	0	0	-0.5	70Ni-isomer		tof94
8	#3241	3241	1	94	45	0	7.22	0	1	5	-2.5	0	0	0	0	70Ni-isomer		tof96
9	#3241	3241	1	96	45	0	7.22	0	1	5	-1.3	0	0	0	0.43	70Ni-isomer		tof96
10	#3242	4218	1	94	45	0	7.29	0	1	5	-0.8	0	0	0	0.43	71Cr		db3sci
11	#3219	3240	1	94	45	0	7.29	0	1	5	-0.8	0	0	0	0.43	71Cr		
12	#3189	3218	1	94	45	0	7.29	0	1	5	-0.8	0	0	0	0.43	71Cr		db3sci
13	#3241	3241	0	94	45	0	4.19	1.45	1	5	-1.7	1	0	0	-0.5	70Ni-isomer		tof96
14	#3242	4218	0	94	45	0	4.19	0.05	1	5	-1.2	1	0	0	-0.5	71Cr		db3sci
15	#3219	3240	0	94	45	0	4.19	0.05	1	5	-1.2	1	0	0	-0.5	71Cr		
16	#3189	3218	0	94	45	0	4.19	0.05	1	5	-1.2	1	0	0	-0.5	71Cr		db3sci
17	#3189	3218	1	94	45	-0.1554	6.5667	0	1.2	5	-1.2	1	0	0	-0.5	71Cr		db3sci
18	#3189	4186	0	913	345	0	4.19	0.05	1	5	-1	1	0	0	-0.5	71Cr		db3sci
19	#3188	3188	0	913	34	0	4.19	-0.35	1	5	-1.35	1	0	0	-0.4	78Zn		db3sci
20	#3188	3188	1	94	45	0	7.29	0	1	5	-0.8	0	0	0	0.43	78Zn		like ne

Trajectory reconstruction: Fine tuning

https://lise.frib.msu.edu/work/SpecTcl/ReconstructionFineTuning.pdf

Content : A/q tuning

- Trajectory reconstruction
- A/q tuning
- X/A & X/D
- pid_fine_values.csv
- A/q fine tuning

Making Matrices for ARIS SpecTcl from savesets

Prepared by Shane Watters (07/26/2024)
Updated by Oleg B. Tarasov (04/02/2025)

https://lise.frib.msu.edu/work/SpecTcl/Matrices_for_ARIS_SpecTcl.pdf

ToF parameters re-organization

tof_offsets_seg35.csv → tof_offsets_seg35_ppac.csv (4 x 10)
tof_offsets_seg35_sci.csv (2 x 10)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	run_start	run_stop	p0xy-pin	p0xy-si0	p0xy-si1	p0xy-sci	p0xy-scin	p0xy-scis	p0xy-p0xy	p0xy-p1xy	p0xy-p0a	p0xy-p1a	p1xy-pin	p1xy-si0	p1xy-si1	p1xy-sci	p1xy-scin	p1xy-scis	p1xy-p0xy	p1xy-p1xy	p1xy-p0a	p1xy-p1a	p0a-pin
2	3091	6000	609.65	0	0	0	528.1	518.07	110.8	0	119.32	0	614.27	0	0	0	532.71	522.68	0	112.6	0	124.08	579.21
3	3053	3090	609.65	0	0	0	528.1	518.07	110.8	0	119.32	0	614.27	0	0	0	532.71	522.68	0	112.6	0	124.08	579.21
4	3040	3052	609.65	0	0	0	528.1	518.07	110.11	0	119.32	0	614.27	0	0	0	532.71	522.68	0	113.26	0	124.08	579.21
5	3039	3039	609.65	0	0	0	528.1	518.07	110.11	0	119.32	0	614.27	0	0	0	532.71	522.68	0	113.26	0	124.08	579.21

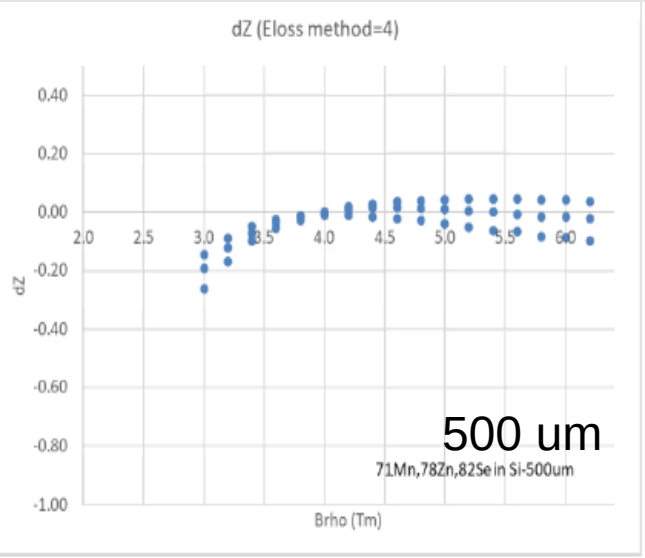
```
// DB3 sciN - db5

    case 20:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5pin;
    case 21:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5si0;
    case 22:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5si1;
    case 23:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5sci;
    case 24:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5scin;
    case 25:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5scis;
    case 26:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5p0_xy;
    case 27:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5p1_xy;
    case 28:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5p0_a;
    case 29:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scin.to_db5p1_a;

// DB3 sciS - db5

    case 30:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5pin;
    case 31:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5si0;
    case 32:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5si1;
    case 33:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5sci;
    case 34:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5scin;
    case 35:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5scis;
    case 36:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5p0_xy;
    case 37:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5p1_xy;
    case 38:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5p0_a;
    case 39:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3scis.to_db5p1_a;
```

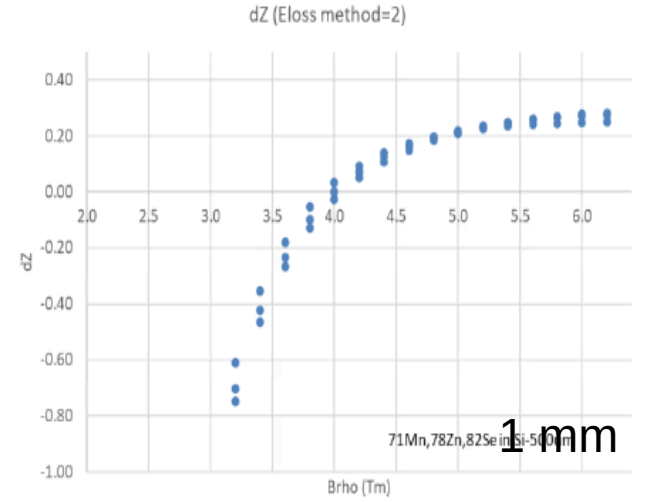
```
// DB3 ppac0 xy - db5

    case 40:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5pin;
    case 41:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5si0;
    case 42:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5si1;
    case 43:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5sci;
    case 44:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5scin;
    case 45:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5scis;
    case 46:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5p0_xy;
    case 47:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5p1_xy;
    case 48:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5p0_a;
    case 49:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_xy.to_db5p1_a;

// DB3 ppac1 xy - db5

    case 50:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5pin;
    case 51:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5si0;
    case 52:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5si1;
    case 53:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5sci;
    case 54:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5scin;
    case 55:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5scis;
    case 56:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5p0_xy;
    case 57:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5p1_xy;
    case 58:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5p0_a;
    case 59:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_xy.to_db5p1_a;
```

```
// DB3 p0_a - db5

    case 60:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5pin;      break;
    case 61:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5si0;      break;
    case 62:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5si1;      break;
    case 63:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5sci;      break;
    case 64:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5scin;     break;
    case 65:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5scis;     break;
    case 66:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5p0_xy;    break;
    case 67:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5p1_xy;    break;
    case 68:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5p0_a;     break;
    case 69:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p0_a.to_db5p1_a;     break;

// DB3 p1_a - db5

    case 70:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5pin;      break;
    case 71:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5si0;      break;
    case 72:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5si1;      break;
    case 73:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5sci;      break;
    case 74:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5scin;     break;
    case 75:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5scis;     break;
    case 76:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5p0_xy;    break;
    case 77:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5p1_xy;    break;
    case 78:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5p0_a;     break;
    case 79:  seg=dbseg_35;  p[0] = &aris.tof.tdc.DB3p1_a.to_db5p1_a;     break;
```

Z-calibration

- v.29 Z-calibration has been based on e-raw signals instead e-calculated assuming zero-offset in order to avoid dependence from dE-calibration
- During last experiments Z-misidentification was observed in several cases after moving far by Brho calibration point
- Z-identification analysis has been performed in this context https://lise.frib.msu.edu/work/SpecTcl/Zcalibr_2025.pdf
- The analysis shown Z-identification is stable (±0.25) for Z=25-34 for Brho=3.5-6.0 Tm with Zigler energy loss model, and for Brho=3.0-6.0 Tm with ATIMA models in the case of 500 um Si-detector. The calibration point was at 4 Tm.
- with 1 mm Si-detector Brho ranges are decreased: Brho=3.7-4.5 Tm with Zigler, Brho 3.6-6.0 Tm with ATIMA. It was concluded that thick dE-detector provides better resolution, but requires second order and log_beta calibrations
- The large positive offset has been reported in dE-calibrations. After that Z-calibration was set on e-calculated values to take into account these offsets. As result: Z-stability for all runs from 3187 up to 3250 (Brho45 ranges 3.3691-4.8114 Tm) reported to be stable. See the next slide.





v.30: After modifying the dE calibration, the Z-calibration must be rechecked for consistency

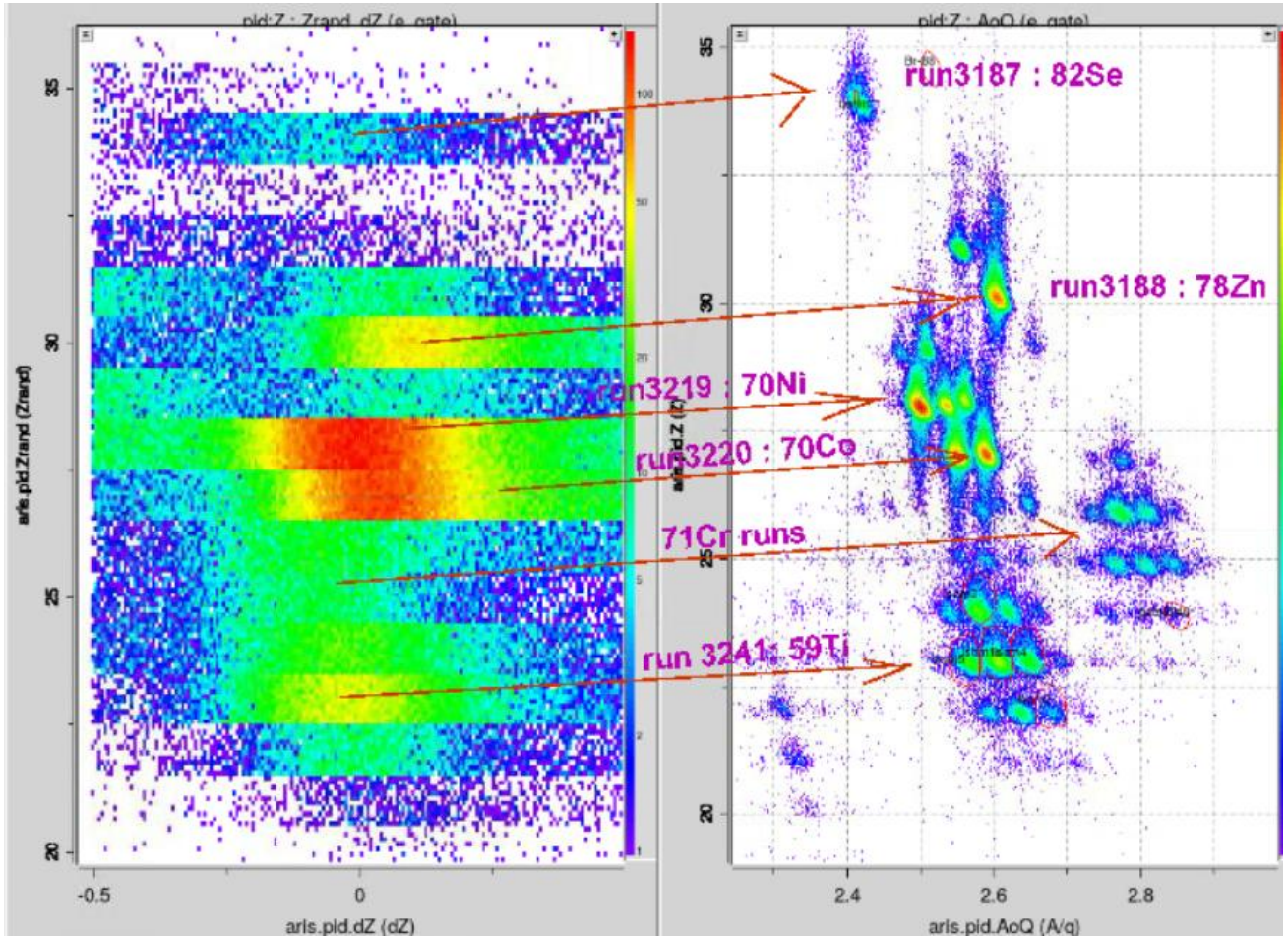
dZ-variable

left plot demonstrates the Z-stability for all runs from 3187 up to 3250 (Brho45 ranges 3.3691-4.8114 Tm)

Fast observation of Z-stability

```
#ifdef _deltaZ
        CTreeParameter Zrand;
        CTreeParameter dZ;
#endif
```

```
#ifdef _deltaZ
if(flag_Z)
{
        int Zint = Z + 0.5;
        dZ = Z - Zint;
        Zrand = Zint + EqualDistr(-0.5, 0.5);
}
else
{
        dZ = 0; dZ.setInvalid();
        Zrand = 0; Zrand.setInvalid();
}
#endif
```



Z-calibration: large positive offset in energy calibration

e1_slope=	0.3033	PIN	
e2_slope=	0.2797	stack0	
e3_slope=	0.4902	stack1	
e4_slope=	0.5408	stack2	
e5_slope=	0.7139	stack3	
e6_slope=	1.0236	stack4	
e7_slope=	0.8650	stack5	
e8_slope=	0.6334	stack6	
			mode_invers
e1_offset=	58.3	PIN	192.18
e2_offset=	76.1	stack0	272.02
e3_offset=	84.9	stack1	173.27
e4_offset=	92.1	stack2	170.26
e5_offset=		stack3	0.00
e6_offset=		stack4	0.00
e7_offset=	0.0	stack5	0.00
e8_offset=	0.0	stack6	0.00
offset_sum=	311.3906		
	all		
chi_dE1_PIN=	0.077		
chi_dE2_stack0=	1.038		
chi_dE3_stack1=	0.960		
chi_dE4_stack2=	0.962		
chi_dE5_stack3=	46.513		
chi_dE6_stack4=	44.295		
chi_dE7_stack5=	18.118		
chi_dE8_stack6=	48.740		
chi_sum=	171.857		

Distance between Z-lines in the 71Cr case is just 30 MeV

In nuclear physics, a **PIN diode detector** used for **energy loss (ΔE) measurement** typically outputs a signal proportional to the ionization caused by the passing ion. When you observe a **large positive offset in energy calibration** (i.e., the baseline of the spectrum is shifted up), it may indicate one or more of the following issues:

1. Electronic Offset / DC Bias
•The most common cause: the preamplifier or shaping amplifier has an **offset voltage** that hasn't been subtracted.
•This can be due to **misconfigured electronics** or **insufficient pole-zero cancellation**.

2. Incorrect Pedestal Subtraction
•In digital signal processing (like with ADCs), the **pedestal** (baseline ADC value with no signal) needs to be subtracted. If it's not done correctly, the energy values will appear artificially high.

3. Leakage Current or Dark Current
•**PIN diodes** may have **leakage current**, especially if **not cooled** or if the diode is **damaged**. This adds to the baseline and can shift all measured energies up.

4. Calibration Error
•If the **calibration point (e.g., from alpha sources or known particles)** is off (e.g., only high-energy points were used), the extrapolated calibration line might overshoot low-energy values, resulting in a positive offset.

5. Pile-up Effects or Noise
•Pile-up of multiple signals may result in artificially higher signals being interpreted as higher energy.

Diagnostic Steps:
•Check **baseline without beam**: what is the offset?
•Verify **electronic zero** with a pulser.
•Compare with a **known source (e.g., 241Am alpha particles)**.
•Recheck **gain and offset settings** in the electronics.
•Measure **leakage current** on the detector bias line.

Ztke revision

Based on:

$$dE \approx \frac{AZ^2}{E},$$

$$E = k(Bx)^2 \frac{q^2}{A},$$

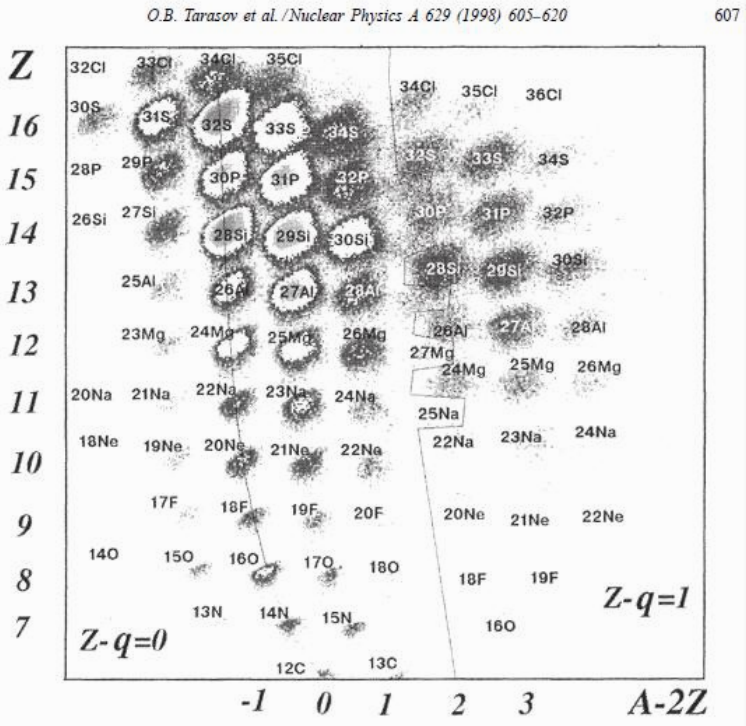
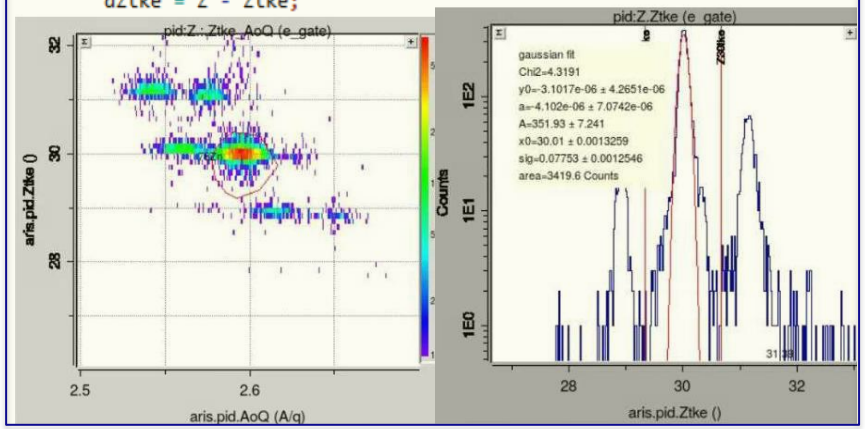


Fig. 1. Identification matrix $(A-2Z, Z)$ of the products of the $^{32}S(14.5\ MeV/A)+C$ reaction obtained at a magnetic field $B = 0.7975$ T. The left solid line passes through the completely stripped nuclei with zero isotopic spin. The right curve discriminates between regions of nuclei with charge $q = Z$ and $q = Z-1$.

CTreeVariable dEpower_Ztke;

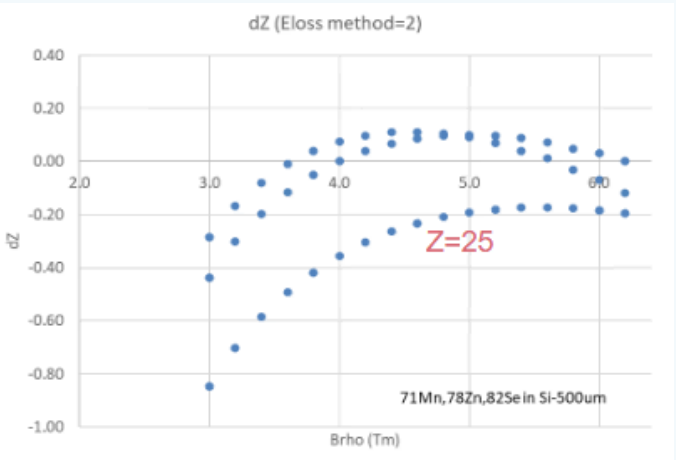
```
#ifdef  _AnalysisZTKE

    if (TKE.isValid() && brho.isValid() && e_reduced > 0)
    {
    double v = pow(e_reduced,dEpower_Ztke)* TKE / brho;
    Ztke = Ztke_slope2*v + Ztke_slope*sqrt(v) + Ztke_offset;
    Ptke = lightSpeed * Ztke * brho;
    Atke = (Ptke*Ptke/TKE - TKE) / amu / 2.;
    if(Ztke < 0.1) Ztke = 0.5;
    AoZtke = Atke / Ztke;
    }
//---------------------------
    if(Q.isValid() && Ztke.isValid())
    {
    ZmQtke = Ztke - Q;
    AoQtke = Atke / Q;
    }

//---------------------------
    if(Z.isValid() && Ztke.isVali
    {
    dZtke = Z - Ztke;
```

Fantastic Z-resolution 0.077!



- Z-identification analysis https://lise.frib.msu.edu/work/SpecTcl/Zcalibr_2025.pdf →
- Z-tke:
  - excellent resolution for local region. It can be used for cleaning and time shift verification.
  - it requires second order and dE_power calibrations.

Eloss_methoc	Result	dE_power	Result @ 0.5
0	0.240	0.670	0.867
1	0.232	0.672	0.882
2	0.177	0.697	0.937
4	0.186	0.694	0.926



Channel Assign and packets update

Upcoming changes to the channel/module mapping:

- Vectors be used will hold the name of the signals and modules
  - Number of signals can easily scale with number of channels in an event of upgrades to modules
  - The location in the vector stack is the slot number (for module) or channel number (signal)
  - i.e in module slot 10, bank1[1] = db1 ppac0 l signal = 2nd channel in the module
- Two banks are defined to simply the counting and will reflect physically what each signal means in the ribbon cable to the modules
- Same names to be used in an event of a reconfiguration
- Simplifying if/else cases for the configuration for easier readability/addition
- Run numbers within the function allows for backwards compatibility

```
// future upgrade to clean up mapping below
int channel = ch& 0x3F-1;
std::vector<std::string> TDC1_config = mapTDC1(runno);
//cout<<std::hex<<ch<< "=ch "<<std::dec<<channel<<endl;
//if(channel<32)
  {
    // cout<<TDC1_config[channel]<<endl;
    if(TDC1_config[channel] == "ARIS_DB1_PPAC0_L_CH") map_ppacT(aris.db[1].ppac[0].l, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC0_R_CH") map_ppacT(aris.db[1].ppac[0].r, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC0_U_CH") map_ppacT(aris.db[1].ppac[0].u, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC0_D_CH") map_ppacT(aris.db[1].ppac[0].d, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC0_A_CH") map_ppacT(aris.db[1].ppac[0].a, data);

    else if(TDC1_config[channel] == "ARIS_DB1_PPAC1_L_CH") map_ppacT(aris.db[1].ppac[1].l, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC1_R_CH") map_ppacT(aris.db[1].ppac[1].r, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC1_U_CH") map_ppacT(aris.db[1].ppac[1].u, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC1_D_CH") map_ppacT(aris.db[1].ppac[1].d, data);
    else if(TDC1_config[channel] == "ARIS_DB1_PPAC1_A_CH") map_ppacT(aris.db[1].ppac[1].a, data);

    else if(TDC1_config[channel] == "ARIS_DB3_PPAC0_L_CH") map_ppacT(aris.db[3].ppac[0].l, data);
    else if(TDC1_config[channel] == "ARIS_DB3_PPAC0_R_CH") map_ppacT(aris.db[3].ppac[0].r, data);
    else if(TDC1_config[channel] == "ARIS_DB3_PPAC0_U_CH") map_ppacT(aris.db[3].ppac[0].u, data);
    else if(TDC1_config[channel] == "ARIS_DB3_PPAC0_D_CH") map_ppacT(aris.db[3].ppac[0].d, data);
    else if(TDC1_config[channel] == "ARIS_DB3_PPAC0_A_CH") map_ppacT(aris.db[3].ppac[0].a, data);
```

Call to map channel number to detector

Mapping value to correct channel

```
std::vector<std::string> mapmodules(int Run) {
  std::vector<std::string> modulelist, left, right;
  //need to verify mapping of old config
  //vme starts with 1
  //ARR06
  if(Run<1000)
    {
      left = {"TDC1","","","","",
              "","","","",""};
      right = {"","","","","",
               "","ADC1","","","",
               ""};
    }
  else
    {
      left = {"","","","","",
             "","","","","TDC1"};
      if(Run<1586)
        right = {"TDC2","TDC3","","","",
                 "QDC1","", "ADC1","","ADC2",
                     ""};
      else
        right = {"","TDC2","","TDC3","",
                  "TDC4","","QDC1","","ADC1",
                  "ADC2"};
    }
    modulelist = left;
    modulelist.insert(modulelist.end(), right.begin(), right.end());
    return modulelist;
}
  //WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
//mapping channels
std::vector<std::string> mapTDC1(int Run) {
  std::vector<std::string> TDC, bank1, bank2;

  if(Run<=2932)
    {
    }
  else if(Run<=3178)
    {
    }
  else if(Run(<=3180)
  else
    {
      bank1 = {"", "ARIS_DB1_PPAC0_L_CH", "ARIS_DB1_PPAC1_L_CH","ARIS_DB1_PPAC0_R_CH",
               "ARIS_DB1_PPAC1_R_CH","","ARIS_DB1_PPAC0_U_CH","",
               "ARIS_DB1_PPAC1_U_CH","ARIS_DB1_PPAC0_D_CH","ARIS_DB1_PPAC1_D_CH","",
               "ARIS_DB1_PPAC0_A_CH","","ARIS_DB1_PPAC1_A_CH",""};
      bank2 = {
               "ARIS_DB3_PPAC0_U_CH","ARIS_DB3_PPAC1_L_CH","ARIS_DB3_PPAC1_L_CH","ARIS_DB3_PPAC1_U_CH",
               "ARIS_DB3_PPAC0_D_CH", "ARIS_DB3_PPAC1_R_CH","ARIS_DB4_PPAC0_L_CH","ARIS_DB3_PPAC0_A_CH",
               "ARIS_DB4_PPAC0_A_CH","ARIS_DB3_PPAC1_A_CH","ARIS_DB3_PPAC0_R_CH","ARIS_DB4_PPAC0_D_CH",
               "ARIS_DB4_PPAC0_R_CH","","ARIS_DB3_PPAC1_D_CH","ARIS_DB4_PPAC0_U_CH"};
```

Others

29.6 calc beta for ARIS timing calibration

```
//WWWWWWWWWWWWWWWWWWWWWWW
class ARIStiming
{
friend class ARIStimingCalib;
friend class ARIStimingTDC;

public:

    CTreeParameter t;
    CTreeParameter length;
    CTreeParameter beta;

.......
    void Calibrate();
    void calcBeta();
};
//WWWWWWWWWWWWWWWWWWWWWWW
void ARIStimingTDC::Calibrate()
{
........

    calcBeta();
}
//WWWWWWWWWWWWWWWWWWWWWWW
void ARIStiming::calcBeta()
// can be called from CAris_pid_LenToF with updated length
{
    bool flag=false;

    if(t.isValid() && length.isValid())
     if( t > 0 && length > 0)
        {
        beta = length / t / lightSpeed;
        if (beta > 0. &&  beta < 1.)   flag = true;
        }
        if(!flag)
    {       beta=0.;        beta.setInvalid();    }
}
```

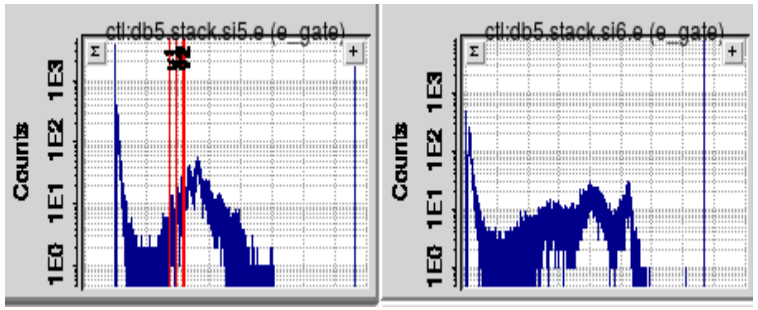
29.10 db3-db4 and db4-db5 special insert for length corrections (CAris_pid_LenToF.cpp)

```
//=========================================================  special inserts to check dL   begin
ARIStiming *p0 = &aris.tof.tdc.DB34p0_xy.to_db4p0_xy;
if(p0->t.isValid() && p0->length.isValid() && p0->t > 10)
 if(aris.Dip34.deltaL.isValid() && aris.pid.length34_correct_use!=0)
    {
    p0->length += aris.Dip34.deltaL * aris.pid.length34_correct_use;
    p0->calcBeta();
    }
p0 =&aris.tof.tdc.DB4p0_xy.to_db5p0_xy;
if(p0->t.isValid() && p0->length.isValid() && p0->t > 10)
 if(aris.Dip45.deltaL.isValid() && aris.pid.length45_correct_use!=0)
    {
    p0->length += aris.Dip45.deltaL * aris.pid.length45_correct_use;
    p0->calcBeta();
    }
//=========================================================  special inserts to check dL   end
```

29.4 upper threshold for PINs (temporary)

```
void ARISpin::Calibrate()
{
    bool flag = false;

    if(e>0.)
      e+= + EqualDistr(0,1);

    if (e>e_threshold && e < 7600) // Oleg inserted temporary v.29.4
      {
            ……
```

Eliminate these events for TKE



List of modifications

// 29.26 revision of tof.csv and tof.tcl
// 29.25 update Aq-fine tuning parameters (cpp-source, csv)
// 29.24 new parameters dZ and Zrand
// 29.23 Z-calculation based on dE instead e_raw
// 29.22 new variable tke_offset
// 29.21 new ToF values 04/09/25
// 29.20 Channel Assign and packets by Elaine 04/09/25
// 29.19 Channel Assign and packets by Elaine 04/08/25
// 29.18 A/q - vertical dispersion @ db5// 29.17 New variable dEpower_Ztke for Ztke calculations
// 29.16 Channel Assign and packets by Elaine
// 29.15 new dZtke parameter --> Z - Ztke, additional modifications with Ztke
// 29.14 new ZmQtke parameter --> Ztke - Q
// 29.13 purging all user #define blocks in pid.h and tof.h
// 29.12 brho from Qint
// 29.11 deltaTOF moved to #define
// 29.10 db3-db4 and db4-db5 special insert for length corrections
// 29.9 brho_method == 345 --> AoQ *= (1.+ AoQ_dP_factor*(aris.Dip34.deltaP+aris.Dip45.deltaP)/200.);
// 29.8 ARIS_PID class revision
// 29.7 ARIS_PID class : useFineTuning
// 29.6 calc beta for ARIStiming calibration
// 29.5 using db4.ppac[0].y for A/q correction
// 29.4 upper threshold for PINs
// 29.3 new timing signals and t_offset for global timing
// 29.2 new AoQ corrections for aris.db[3].b and aris.db[5].b;
// 29.1 optics cpp-file was changed.