Notes on the `GlueX FSRoot` Format

Ryan Mitchell

July 26, 2024

Abstract

This document describes the `GlueX FSRoot` format.

Contents

1 Overview **1**

2 Final States **2**

3 Event Information **2**

4 Particle Four-Momenta and Vertices **3**

5 Track Information **4**

6 Shower Information **4**

7 Vee Information **5**

8 MC Truth Information **5**

9 Final State Numbering **6**

1 Overview

The `GlueX FSRoot` format is a flat `TTree` format. All variables are `double`. Multiple combinations within an event are listed as separate `TTree` entries, just like entries from distinct events. The format is designed to be compatible with the `FSRoot` package[1].

[1]The `FSRoot` package, along with documentation, can be found here: https://github.com/remitche66/FSRoot

2 Final States

The `FSRoot` format can hold information from any final state composed of a combination of Λ (decaying to $p\pi^-$), $\overline{\Lambda}$ (decaying to $\overline{p}\pi^+$), e^+, e^-, μ^+, μ^-, p, $\bar{p}$, η (decaying to $\gamma\gamma$), γ, K^+, K^-, K^0_S (decaying to $\pi^+\pi^-$), π^+, π^-, and π^0 (decaying to $\gamma\gamma$).

Final state particles are listed in trees in the following order:

```
Lambda ALambda e+ e- mu+ mu- p+ p- eta gamma K+ K- Ks pi+ pi- pi0
```

For example, in the process $\gamma p \to \pi^+\pi^- J/\psi p; J/\psi \to \mu^+\mu^-$, the μ^+ is particle 1, the μ^- is particle 2, the p is particle 3, the π^+ is particle 4, and the π^- is particle 5. Or as another example, in the process $\gamma p \to \gamma\chi_{c1} p; \chi_{c1} \to \eta\pi^0\pi^0$, the p is particle 1, the η is particle 2, the γ is particle 3, one π^0 is particle 4, and the other π^0 is particle 5. In cases like this where there are identical particles, no ordering is assumed.

3 Event Information

The "Event Information" variables contain information about the event as a whole:

```
Run:          run number
Event:        event number
Chi2:         chi2 of the kinematic fit
Chi2DOF:      chi2/dof of the kinematic fit
RFTime:       RF time determined from the kinematic fit
RFDeltaT:     difference between RF time and beam photon time

EnUnusedSh:       total energy of unused showers
NumUnusedTracks:  number of unused tracks
NumNeutralHypos:  number of neutral hypotheses
NumBeam:          number of beam hypotheses from tagger
NumCombos:        number of combos

ProdVx:       production vertex parameters
ProdVy:         "
ProdVz:         "
ProdVt:         "
PxPB:         kinematically fit beam parameters
PyPB:           "
PzPB:           "
EnPB:           "
VxPB:           "
VyPB:           "
VzPB:           "
```

```
RPxPB:        measured beam parameters
RPyPB:          "
RPzPB:          "
REnPB:          "
RVxPB:          "
RVyPB:          "
RVzPB:          "
MCPxPB:       thrown beam parameters (MC only)
MCPyPB:         "
MCPzPB:         "
MCEnPB:         "
MCEnergy:       "
MCVxPB:         "
MCVyPB:         "
MCVzPB:         "
```

4 Particle Four-Momenta and Vertices

The "Particle Four-Momenta" variables contain the four-momentum for each particle in the final state:

```
(prefix)PxP(n):  x momentum of particle (n)
(prefix)PyP(n):  y momentum of particle (n)
(prefix)PzP(n):  z momentum of particle (n)
(prefix)EnP(n):  energy of particle (n)
```

The vertices for all non-decaying final state particles (including $\pi^{\pm}$, γ, etc., but not including decaying particles like K_S or π^0) are also kept:

```
(prefix)VxP(n):  x vertex of particle (n)
(prefix)VyP(n):  y vertex of particle (n)
(prefix)VzP(n):  z vertex of particle (n)
```

Different types of four-momenta and vertices are distinguished using prefixes. Raw (i.e. measured) values have a prefix `R`; final (i.e. kinematically fit) values have no prefix; MC truth information have a prefix `MC`.

Different particles are differentiated using the postfix `P(n)`, where `(n)` is the number of the particle in the ordered list. Four-momenta and vertices for secondaries originating from particle `(n)`, such as the two γ's from a π^0, are recorded using `P(n)a` and `P(n)b`, where the ordering follows the same conventions as above, or, in the case of identical daughter particles, no ordering is assumed. As two examples: in the process $\gamma p \to \pi^+\pi^- J/\psi p; J/\psi \to \mu^+\mu^-$, the raw energy of the π^+ is given by `REnP4`; and in the

process $\gamma p \to \pi^+\pi^- J/\psi p; J/\psi \to \pi^+\pi^-\pi^0$, the y-momentum of a photon from the π^0 decay, after the kinematic fit, is given by `PyP6b`.

5 Track Information

Track information is written out for every reconstructed track that is part of a final state. The postfix `P(n)` follows the same convention as for the four-momenta (section 4).

```
TkChi2P(n):   chi2 of the track fit
TkNDFP(n):    number of degrees of freedom for the track fit
```

If PID information is requested, the following variables are added to the tree:

```
TkTOFBetaP(n):   beta from the TOF (using kinematic fit if available)
TkTOFChi2P(n):   chi2 from the TOF (using kinematic fit if available)
TkDEDXCDCP(n):   DEDX from the CDC
TkDEDXFDCP(n):   DEDX from the FDC
TkDEDXChi2P(n):  chi2 for the DEDX measurement
TkDEDXNDFP(n):   NDF for the DEDX measurement
```

Information from the DIRC can be requested using the argument -dirc 1 [it is not included by default]. The following variables will be added to the tree:

```
TkLpiDIRCP(n): log-likelihood from DIRC for a pion hypothesis.
TkLpDIRCP(n): log-likelihood from DIRC for a proton hypothesis.
TkLkDIRCP(n): log-likelihood from DIRC for a kaon hypothesis.
TkLeleDIRCP(n): log-likelihood from DIRC for an electron hypothesis.
TkNumPhotonsDIRCP(n): number of Cherenkov photons measured in the DIRC.
TkXDIRCP(n): x-position in the DIRC.
TkYDIRCP(n): y-position in the DIRC.
```

6 Shower Information

Shower information is written out for every reconstructed shower that is part of a final state. The postfix `P(n)` follows the same convention as for the four-momenta (section 4).

```
ShQualityP(n):   a shower quality score
```

The argument "-gAddUnusedNeutrals N" can be used to write out four-momentum information for N unused showers:

```
PxPUN(n):  x-momentum of the nth unused shower
PyPUN(n):  y-momentum of the nth unused shower
PzPUN(n):  z-momentum of the nth unused shower
EnPUN(n):  energy of the nth unused shower
```

7 Vee Information

This information is recorded for each $K^0_S \to \pi^+\pi^-$, $\Lambda \to p\pi^-$ and $\overline{\Lambda} \to \overline{p}\pi^+$ decay:

```
VeeLSigmaP(n):   the separation between the primary and
                  secondary vertex (L) over its error (sigma)
VeeLP(n):        the flight length (L)
```

8 MC Truth Information

The following variables can be used to help keep track of truth information.

```
MCDecayCode1:      the true code1, described in the
                     "final state numbering" section
MCDecayCode2:      the true code2, described in the
                     "final state numbering" section
MCExtras:          1000 * number of neutrinos +
                    100 * number of K_L +
                     10 * number of neutrons +
                      1 * number of antineutrons
MCSignal:          1 if the reconstructed final state
                     matches the generated final state;
                   0 otherwise
```

The `MCDecayParticle` variables are an ordered list of the PDG ID numbers of the particles coming from the initial reaction. For example, for $\gamma p \to \pi^+\pi^- p$, `MCDecayParticle1` is -211 (for the π^-); `MCDecayParticle2` is 211 (for the π^+); `MCDecayParticle3` is 2212 (for the p); and all the others are zero. These variables are meant to help distinguish between reactions with different resonances but the same final state (for example $\gamma p \to \rho p$ vs. $\gamma p \to \pi^+\pi^- p$), but many resonances are not currently included in the analysis trees.

```
MCDecayParticle1:  the PDG ID of the 1st particle
MCDecayParticle2:  the PDG ID of the 2nd particle
MCDecayParticle3:  the PDG ID of the 3rd particle
MCDecayParticle4:  the PDG ID of the 4th particle
MCDecayParticle5:  the PDG ID of the 5th particle
MCDecayParticle6:  the PDG ID of the 6th particle
```

9 Final State Numbering

Final states are designated using two integers, "code1" and "code2". The digits of each integer are used to specify the number of different particle types in the final state:

```
code1 = abcdefg
        a = number of gamma
        b = number of K+
        c = number of K-
        d = number of Ks  ( --> pi+ pi- )
        e = number of pi+
        f = number of pi-
        g = number of pi0 ( --> gamma gamma )

code2 = hijklmnop
        h = number of Lambda (--> p+ pi-)
        i = number of ALambda (--> p- pi+)
        j = number of e+
        k = number of e-
        l = number of mu+
        m = number of mu-
        n = number of p+
        o = number of p-
        p = number of eta  ( --> gamma gamma )
```

These integers are sometimes combined into a single string of the form:

```
"code2_code1"
```

Here are a few examples:

```
"0_111":      pi+ pi- pi0
"0_1000002":  gamma pi0 pi0
"1_220000":   eta K+ K+ K- K-
"11000_110":  mu+ mu- pi+ pi-
```