Plans for KFParticle refactoring

Sergey Gorbunov

KFParticle Meeting 11 Oct 2023

KfParticle package

KFParticle for CBM & ALICE

- https://github.com/cbmsw/KFParticle
- STAR version to be merged

We need one official repository with the up-to-date version

KfParticle rewrite

- Get rid of C-style defines for different cases
  - minimize the amount of cases
  - replace defines with template arguments

```
#ifdef __ROOT__ //for the STAR experiment
#define HomogeneousField
#endif

#ifdef HLTCA_STANDALONE
#include "RootTypesDef.h"
#else
#include "TObject.h"
#endif
```

```
  //* Method to access ALICE field
#ifdef HomogeneousField
  static float GetFieldAlice();
#endif

 private:
#ifdef HomogeneousField
  static float fgBz;  ///< Bz compomen
#endif
#ifdef NonhomogeneousField
  /** \brief Approximation of the magn
   ** Each component (Bx, By, Bz) is a
   **/
  float fieldRegion[10];
#endif
```

KfParticle rewrite

- Unify the math
  - rotate CS with respect to the magnetic field?

```
float KFParticleBase::GetDStoPointCBM( const float xyz[3], float dsdr[6] ) const
{
  /** Returns dS = l/p parameter, where \n
```

```
float KFParticleBase::GetDStoPointB( const float* B, const float xyz[3], float dsdr[6] ) const
{
  /** Returns dS = l/p parameter, where \n
```

```
float KFParticleBase::GetDStoPointBy( float By, const float xyz[3], float dsdr[6] ) const
{
  /** Returns dS = l/p parameter, where \n
```

```
float KFParticleBase::GetDStoPointBz( float B, const float xyz[3], float dsdr[6], const float* param) const
{
  /** Returns dS = l/p parameter, where \n
```

KfParticle rewrite

- Cleanup the interfaces, make sure everything work

```
//* Set mass constraint

void SetNonlinearMassConstraint( float Mass );
void SetMassConstraint( float Mass, float SigmaMass = 0 );
```

- A special case when the mass is set (E from M, not M from E)

- Same math at all opening angles (may be it is already the same)

KfParticle rewrite

- Coding conventions - from ALICE O2?

- Split data and operations
  - KfParticle.Transport..() -> KfManager.Transport(KfParticle, ..)
  - we can do more sophisticated stuff when we are not limited by the KfParticle data structure. KfManager can have some setup knowledge, store extra information for linearisation etc.
- Unify the creation of particles

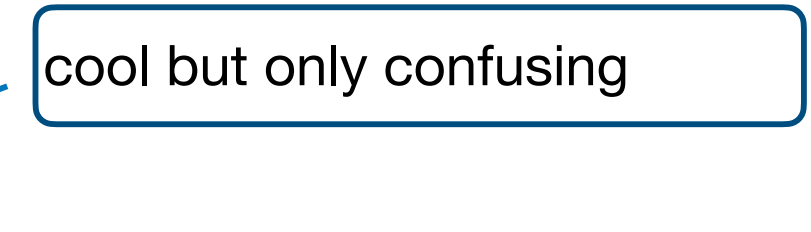
```
KFParticle D0;
KFParticle D0+=Pion;
KFParticle D0+=Kaon;
```

```
KFParticle D0(Pion, Kaon);
```

KfParticle rewrite

- Unify the creation of particles

```
KFParticle D0;
KFParticle D0+=Pion;
KFParticle D0+=Kaon;
KFParticle D0-=Pion;
KFParticle D0+=Pion;
```



KfParticle rewrite

- Unit Test first (Valentina’s talk)
- resources for refactoring: Valentina & Sergey