Analysis Tools

Kinematics Library

Eddy Offermann

CEBAF 12000 Jefferson avenue, Newport News VA 23606

QUICK START

In this document the FORTRAN library, KINEMA, is described containing routines to calculate kinematical quantities in scattering reactions. In Table 1 an overview of the routines is given. In order to use the library with f77, add to your Makefile on the HP-UX machines the following lines:

```
HALLA_LIB = /adaq1/users/offerman/lib
FFLAGS    = -C -K +ppu
LDFLAGS   = +U77 -Wl,-L$(HALLA_LIB)
LIBS      = -lkinema
```

Define also the shell variable `HALLA_SRC` by adding to your `.login` file (for the `csh`) the line:

```
HALLA_SRC = /usr/user9/offerman/src
```

There exists also a double precision version of this library. One can link to it by replacing `-lkinema` by `-lkinema_db` in the `Makefile`.

Transformation of 3/4-vectors	
beam_to_transport	rotate3v
collimator	rotate3v_axis
ctrans	rotate4v
newang	spherang
pgang	transport_to_beam
rotate	
4-vector manipulations	
boost	extang4v
conetest	pmag
conserve	put4v
contract	scale4v
Path length through target	
circular_target	in_target
cylindrical_target	target_exit
flat_target	
Energy loss in material	
deltam	eloss
drange	radlength
Kinematic variables	
findbeta	decaylab
find_q	omega
decaycm	masses
Random number generator	
hisran	rangauss
random_gen	ranloren

Table 1: Overview of routines in the library KINEMA .
Routines are grouped according to subject.

In the following sections each individual routine is described. A star in front of a variable in the argument list indicates that it is input while a star after the argument is assigned to the routine's output.

CHAPTER 1

Transformation of 3/ 4-vectors

```
subroutine beam_to_transport(*idet,*rpoint,*pvect,*pcentral,
                                                    txy*,ray*)
```

```
integer idet
real rpoint(3),pvect(3),pcentral,txy(2),ray(6)
```

description:

This routine converts a particle vertex going through the point `rpoint` and having direction `pvect` from the beam coordinate system to the transport coordinate system coupled to the spectrometer under consideration. The transformation is defined by a call to the alternate entry (`define_beam_to_transport`) and a call to this entry point should preceed the first call to this routine..

input:

`idet`		an integer indicating which spectrometer the particle is entering. The transformation for each spectrometer is set up by a call to `define_beam_to_transport`. `idet`<32
`rpoint`	cm	the reaction point (x,y,z)
`pvect`		the particle vector momentum vector given in the beam coordinate system
`pcentral`		the momentum of the central spectrometer ray in the same units as `pvect`.

output:

`txy`	m	the *true* transport coordinate x and y of the reaction point. This differs from the values put into the `ray` vector below. The values put into the `ray` vector are the x and y values of the vector projected onto the $z_{transport} = 0$ plane
`ray`	m	the transport coordinates of the particle; these are $(x,\tan(\theta),y,\tan(\phi),z,\Delta p/p)$ with (x,y,z) in meters, $\tan(\theta,\phi)$ unitless, and $\Delta p/p$ unitless

```
entry define_beam_to_transport(*idet,*imode,*th0,*ph0)

integer idet,imode
real th0,ph0
```

description:

This entry initializes the transformation matrix for spectrometer `idet` as used in the subroutine `define_beam_to_transport`.

input:

`idet`		an integer indicating which spectrometer the particle is entering; `idet`<32
`imode`		switch indicating the orientation of the spectrometer bend plane. Three modes are possible: +1 = "up" bend; e.g. OOPS, –1="down"; e.g. ELSSY, 0="in" bend, e.g. CLAS. Formaly the three modes are defined as: Suppose the beam coordinate system is defined by z along the incoming beam and y pointing upward (gravity). If the $x_{transport}$ axis is in the y-z plane of the beam system, we have `imode` = –1 or +1. (+1 for $y_{beam} \vee x_{transport}$>90˚ else if the $x_{transport}$ axis lies in the plane defined by z_{beam} and $z_{transport}$ `imode` = 0.
`th0,ph0`	deg	the spherical angles θ and φ of the central ray in the beam coordinate system (as used for `rpoint` and `pvect` in `beam_to_transport`)

output:

none

```
subroutine collimator(*idet,*discol,*r0,*theta,*phi,hit*)
```

```
integer idet
real discol,r0(3),theta,phi
logical hit
```

description:

This routine is used to decide whether or not a particle trajectory passes through a spectrometer collimator. Before using this routine the collimators have to be defined by a call to the entry `define_collimator`. It sets up collimators as rectangular planes in space of zero extent. Finite extent may be enforced by using two calls with different distances to the collimator plane. The orientation of the collimators may be *up-down* (along y_{hall}) or radial (symmetric about $z_{transport}$ of the spectrometer). The method used here can readily be extended to define other multi-sided collimators such as hexagons, octagons, ...etc.

input:

`idet`		an integer indicating which spectrometer the particle is entering; `idet`<32
`discol`		The distance from the hall coordinate system origin to the center of the collimator (given in the same units as the reaction point coordinates `r0`). This is included as an argument here instead of in the alternate entry (`define_collimator`) so that one can give the collimator a finite spacial extent by calling this routine twice with everything the same except the collimator distance (the width of the collimator is then the difference of the distances).
`r0`		the reaction point (origin of particle trajectory) in cartesian coordinates in the same units as the collimator distance `discol` given in the hall system.
`theta,phi`	deg	the spherical angles of the particle trajectory in the hall system

output:

`hit`	`.true.` if the particle characterized by (`r0,theta,phi`) passes through the collimator characterized in the entry by (`th0,ph0,discol,delta1,delta2`).

```
entry define_collimator(*idet,*ishape,*imode,*th0,*ph0,*delta1,*delta2)

integer idet,ishape,imode
real th0,ph0,delta1,delta2
```

description:

Entry point where the collimator of spectrometer `idet` ican be defined

input:

`idet`		an integer indicating which spectrometer the particle is entering; `idet`<32
`ishape`		0: circular collimator model 1: rectangular model
`imode`		this integer determines the orientation of the collimator. 0: the collimators are all oriented symmetrically about $z_{transport}$ with `delta1` indicating the radial 1/2 angular opening. 1: the collimators are all oriented "up-down" with `delta2` indicating the 1/2 angular opening in the direction of y_{hall} ("up") and `delta1` indicating the "in-plane" angular opening.
`th0,ph0`	deg	the spherical angles of the colimator center in the hall systemy
`delta1`	mrad	the 1/2 angular opening of the collimator in one direction (the "sideways/inplane" or "radial" direction depending upon `imode`)
`delta2`	mrad	the 1/2 angular opening of the collimator in the other direction

output:

none

```
subroutine ctrans(*imode,*th_geo*,*ph_geo*,*th_sph*,*ph_sph*,ierr*)

integer imode,ierr

real th_geo,ph_geo,th_sph,ph_sph
```

description:

This routine changes the angles θ and φ, defining the direction of a vector in 3d between two definitions: the spherical angle system and the geometrical angle system. The switch imode defines the direction the transformation is being done.

input:

`imode`		switch deciding which conversion to make and which angles are input and which are output. Valid values: 0,1,2,3 0 : take a geometrical angle th_geo(-180,180) and change it to the spherical definition with th_sph in (0,180) and ph_sph=0 (-180<th_geo<0) or 180 (0<th_geo<180) 1 : regular conversion of (th_geo,ph_geo) into (th_sph,ph_sph) 2 : regular conversion of (th_sph,ph_sph) into (th_geo,ph_geo) 3 : conversion of (th_sph,ph_sph) into (th_geo,ph_geo) but put th_geo in (-180,180)
`th_geo, ph_geo`	deg	geometrical angles: $0<\theta<360,-90<\phi<+90$
`th_sph, ph_sph`	deg	spherical angles: $0<\theta<180,0<\phi<+360$

output:

`ierr`	0:	no errors encountered
	1:	error(s) encountered

```
subroutine newang(*isystem,*th_1,*ph_1,*th_2,*ph_2,th_new*,ph_new*)

integer isystem
real th_1,ph_1,th_2,ph_2,th_new,ph_new
```

description:

The spherical angles (`th_2,ph_2`), describing the direction of a vector, are transformed to another coordinate system. Depending on the switch `isystem` the orientation of the z-axis of the new system is the same as in the old system or given by the pair (`th_1,ph_2`). The direction of the vector with respect to this new z-axis is given by (`th_new,ph_new`).

This routine can be applied when a particle track is described in the hall system by (`th_2,ph_2`) and the momentum transfer direction by (`th_1,ph_1`). By selecting `isystem=2` the track direction is given in spherical angles with respect to q.

input:

`isystem`		1 : new coordinate system is the same as the old one. The new angles are spherical wrt this system.
		2 : orientation of new coordinate system is given by (`th_1,ph_1`). The new angles are spherical wrt this system.
		3 : new coordinate system is the same as the old one. The new angles are geograhical wrt this system.
		4 : orientation of new coordinate system is given by (`th_1,ph_1`). The new angles are geograhical wrt this system.
`th_1, ph_1`	deg	spherical angles defining orientation of the z-axis of the new coordinate system for `isystem`=2,4
`th_2, ph_2`	deg	spherical angles of the particle track to be transformed

output:

`th_new, ph_new`	deg	new angles of particle track

```
subroutine pgang(pd*,*thd,*phd,*pp,*thp,*php,thg*,phg*)
```

```
real pd,thd,phd,pp,thp,php,thg,phg
```

description:

Consider the decay: $\Delta^+ \rightarrow p+\gamma$, where for instance the delta has been created through electron scattering. The delta has momentum q equal to the momentum transfer by the electron. If the size and direction of the momentum of the outgoing proton (pp,thp,php) is known, the direction of the gamma (thg,phg) and the momentum of the delta (pd) can be calculated.

c (m1,p1,th1,ph1)-->(m2,p2,th2,ph2;m3,p3,th3,ph3)

c send the subroutine the following:

c m1,p1,th1,ph1-->m2,th2,ph2;m3

c and it will complete the kinematics, returning

c p2,p3,th3,ph3

c note ranges on angles:

c input: 0<th2<180 ; 0<ph2<180

c output: 0<th3<180 ; 0<ph3<180

input:

thd	deg
phd	deg
pp	MeV
thp	deg
php	deg

output:

pd	MeV
thg	deg
phg	deg

comment:

CHAPTER 2

4-vector manipulations

CHAPTER 3

Path Length through Target

CHAPTER 4

Energy Loss in Material

CHAPTER 5

Kinematic Variables

CHAPTER 6

Random Number Generator