CARMA

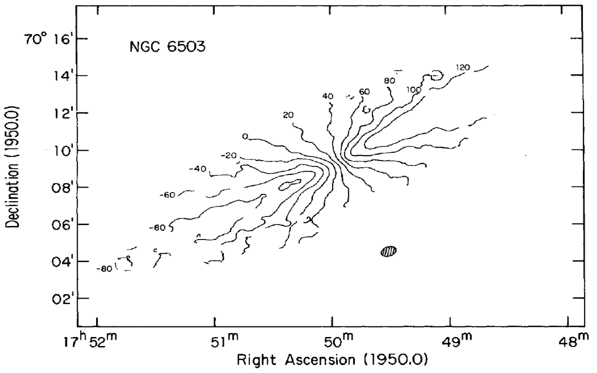
Combined Array for Research in Millimeter-Wave Astronomy

Creating and Using Velocity Fields in MIRIAD

Peter Teuben and LMA (Lisa, Misty, Nurur, ...)

68

February 3, 2011



Change Record

REVISION	DATE	AUTHOR	SECTIONS/PAGES AFFECTED
	REMARKS		
50	24-jun-2008		
	Draft for 2008 Summerschool		

Abstract

This memo discusses a number methods to create good quality bias and noise free velocity fields of disk galaxies, and some of the software to analyze these velocity fields. It uses the MIRIAD, ds9 and NEMO packages, but also recommends GIPSY and KARMA for qualitative inspection.

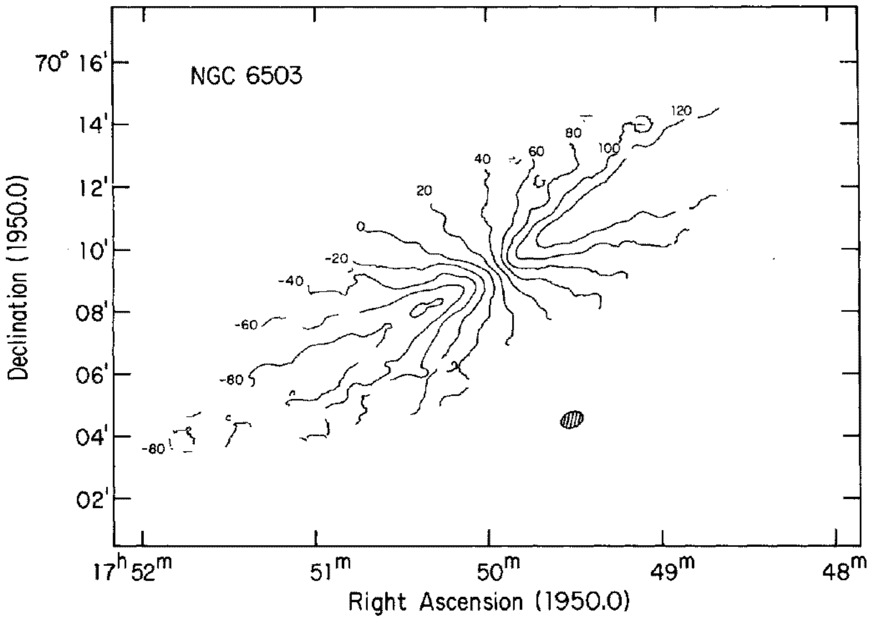


Illustration 1: Velocity field of NGC 6503 (van Moorsel & Wells, 1985)

1. Introduction

One of the most heavily used derived products from a RA-DEC-VEL datacube are the first 3 moment maps along the velocity axis. The 0th moment is simply the total emission, the 1st moment is usually a good representation for the radial velocity along the line of sight, and the 2nd moment the velocity dispersion of the gas (along the line of sight). Each of these have biases that depend on the convolving beam, the signal to noise (which can depend on the location in the map) and the geometry of the object (e.g. for highly inclined disk galaxies), and even the distribution of the gas across the beam.

To simplify some of the examples below, there is a task in MIRIAD to generate a model data cube with the velocity field of a rotating disk, which we will use in the examples

```
imgen    out=den0 object=disk spar=1,0,0,200,100,30
velmodel in=den0 pa=30 incline=60 radius=50,100 out=vel0
         vrot=200,200 options=sample
velimage in=den0,vel0 nchan=60 start=-300 step=10 out=cube0 sigma=20
```

The simplest way, using **moment,**

```
moment in=cube0 out=vel0 mom=1 clip=0.2
```

will often have biases, even with a clip level applied. A clip level is always useful, where you would only use values larger than the clip value (say 1.5 or 2 times the RMS in the cube). Many discussions in the literature, going back to Bosma (1978), and Begeman (1987).

Instead of taking moments, a parameterized fit, e.g a gaussian fit (**gaufit**) can also provide excellent unbiased values for the velocity field. However, if the emission across a beam has strong velocity gradients and/or is not uniform, biases will still be introduced. In these extreme cases more appropriate modeling is needed, some of which are mentioned below.

A simple example to create a velocity field from our model cube is as follows:

```
gaufit in=cube0 ...
```

but a more in depth discussion can be found elsewhere (...)

By taking moments and reducing the information to a single number, an operation such as fitting a rotation curve becomes a simple non-linear fit (e.g. NEMO programs **rotcur**, **rotcurshape**). A completely alternative approach is starting from a full 6D (position and velocity) model, project grid and smooth it, and compare this directly to the observations (e.g. NEMO's **snapgrid** and **snapgridsmooth**). An example of this approach can be found in the GIPSY package as the tool `TiRiFiC` . See also arXiv:astro-ph/0703207v1

2. Automated Methods: smooth cube masking

Smooth the cube in position (**smooth**, **convol**) and velocity (**hanning**), or use invert to make a smoother cube, but be sure to get the same gridding. Use **histo** to find the (now lower) noise in the map.

```
smooth in=cube1 out=cube2 fwhm=10
hanning in=cube2 out=cube3 width=5
histo in=cube3 | grep Rms
```

Make as mask where the signal is greater than say 3 times the RMS, lets say 0.561,

```
maths exp=cube1 mask=cube3.gt.0.561 out=cube1sm
```

After this traditional moments can be used to obtain an improved velocity field:

```
moment in=cube1sm out=1sm.mom1 mom=1
```

See also the **`moment.csh`** script, discussed at more length below.

3. Automated Methods: rotcurmask

To remove some bias around a rotation curve, use **rotcurmask** to create a mask around the “model” expected signal and cut N-sigma around this expected signal.

```
moment in=cube3 out=cube3.mom1 mom=1 clip=0.561
moment in=cube3 out=cube3.mom2 mom=2 clip=0.561
rotcurmask in=cube1 out=cube1rm rotmod=cube3.mom1 minsigma=8
```

Note sigma could be given as a constant, or sigma(x,y) from a mom=2 map from **moment**.

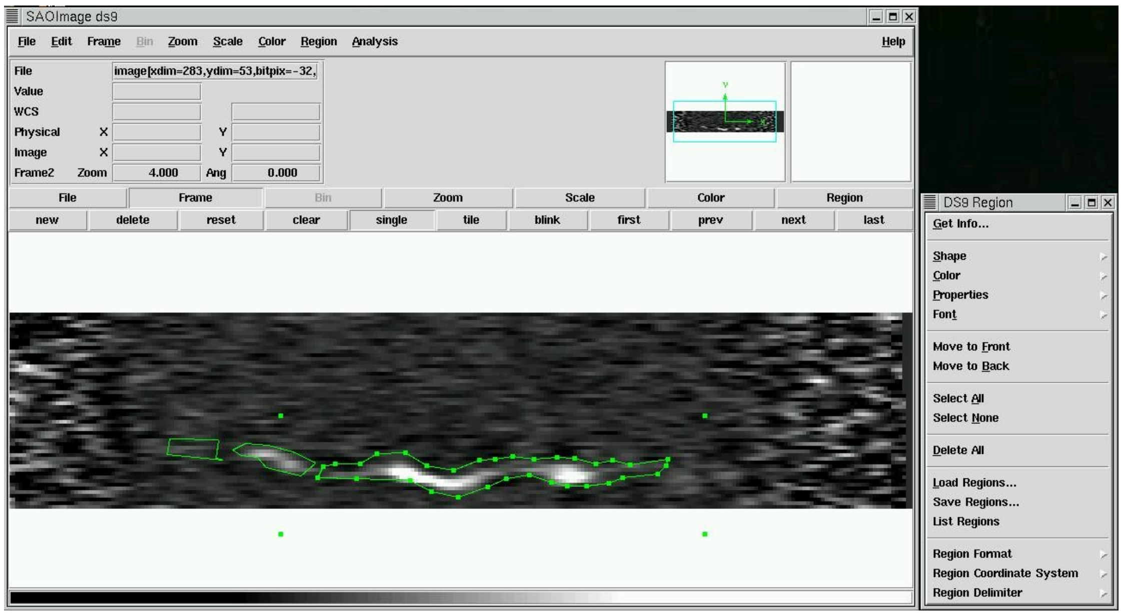
4. Interactive methods

Cubes can be interactively masked in RA-VEL, DEC-VEL or RA-DEC planes using **ds9**, though for anything but the latter you will have to use **reorder** first.

It uses **ds9** to plot plane by plane, and for each slice you will interactively create regions (using *Region->Shape->Polygon*) and save (using *Region->Save Regions...*) them. The shell script **cubemask** and **saomask** are using to launch **ds9** for all the slices, and masking in all the region files into the miriad cube.

Note that although quite laborious, changes from plane to plane are often small and can be done efficiently by re-using the previous polygon(s) and re-morphing them right-clicking on a vertex point when near enough to the point (the cursor shape will change when you are close enough). New vertex points can be added in the middle of a vertex by right-clicking on the vertex, again notice when the cursor shape changes between adding and moving the whole selected polygon. If during this action the BACKSPACE key is hit, the point is removed. Be careful, you could also loose the polygon this way, though there is also Control-Z to undo this.

Here is a screendump of ds9 in action:


File image[xdim=283,ydim=53,bitpix=-32,
Frame2 Zoom 4.000 Ang 0.000

DS9 Masking

The **ds9** image tool is a very effective program to view MIRIAD images (or image planes from a cube). Here is a description how to create region files that can later be used to mask these regions for further analysis (any program that can use non-bounding box region masking, of which **moment** happens to not be a member!).

First launch **ds9:**

```
ds9 &
```

A few shortcuts are useful here. With the left mouse pull down *Region* menu from the top line and you will see a menu pop up. Click on the dashed line at the top of this menu (this is called a teardown menu), it will tear down the menu and put it to the side, because you're going to need this menu a lot!

First make sure the region output format into something we understand. Currently under *Region -> File Format* we probably see the following options:

ds9/funtool

ciao

saotng

saoimage <------ *this is the one we want*

iraf/pros

If you ever had a region file in the wrong format, here's a recipe to change them via ds9's xpa commands:

```
xpaset -p ds9 regions delete all
xpaset -p ds9 regions format ds9
cat file1.reg | xpaset ds9 regions
xpaset -p ds9 regions format saoimage
xpaset -p ds9 regions save file2.reg
```

Also maybe we'll add some tagname in the object field. You need to add this with *Edit -> Preferences -> Information Panel -> Object* *NOTE: this is from old ds9, now different.*

The script that we will use first is called '**cubemask**' which will allow you to loop over planes (usually planes will be constant in RA or DEC, since it ioften easier to identify emission in DEC-VEL or RA-VEL images, as opposed to RA-DEC images, but there is nothing to stop you from doing that.

First we create two derived cubes, always choosing velocity along the first axis of the new cube because of limitations in the **moment** progam to use axis=2:

```
reorder in=m51cube out=cube1 mode=312        (vel-ra-dec)
reorder in=m51cube out=cube2 mode=321        (vel-dec-ra)
```

We now run a helper script to launch planes into ds9, though you can equally well do this with a single **mirds9** using the *Data Cube* slider (remembe to save regions in *saoimage* format):

```
cubemask cube1 region1  110 120
```

or

```
mkdir cube1/region1 ;  mirds9 cube1
```

This would create a new subdirectory '`region1`' inside of the MIRIAD dataset '`cube1`', within which region files (small ascii text files that describes regions as produced with ds9), one for each plane. they will be called `saoregion.N`, where N is the plane number. In this example only planes 110 through 120 would be viewed. In the case of cube1 each plane represents a different DEC.

```
saomask cube1 region1
```

This would then recreate a mask, based on all the regions described in the `region1` directory of `cube1`. This script can easily take few minutes if you have a lot of regions to be masked into the cube. Now that the mask has been set, a moment analysis will automatically use only points where the mask was set.

```
moment in=cube1 out=cube1.mom0 mom=0 axis=1
moment in=cube1 out=cube1.mom1 mom=1 axis=1
mirds9 cube1.mom0
mirds9 cube1.mom1
```

The same analysis can be done on (the VEL-DEC-RA) `cube2`, with one minor modification. After the axis=1 moments of this VEL-DEC-RA cube, we wind up with a DEC-RA map, and reorder needs to be used once more to get a proper RA-DEC map.

```
cubemask cube2 region2
saomask cube2 region2
moment in=cube2 out=cube2.tmp.mom0 mom=0 axis=1
moment in=cube2 out=cube2.tmp.mom1 mom=1 axis=1
reorder in=cube2.tmp.mom0 out=cube2.mom0 mode=21
reorder in=cube2.tmp.mom1 out=cube2.mom1 mode=21
```

Your final map will seem to have some streaks of emission with rather choppy edges and places where you did end the polygons along the RA or DEC direction.

Be especially careful if you see sharp transitions, it may be that you selected your regions not well enough. It is very easy to re-edit your regions , just

```
cubemask cube1 region1 121 121
```

would bring up plane 121 and load any regions that you may have saved. Edit and save them again.

To now combine the two cubes you would reorder the 2nd cube and mask its mask into that of the 1st cube using immask:

```
reorder in=cube2 out=cube2r mode=132
immask in=cube1 region="mask(cube2r)" logic=or
```

after which you can repeat the moment analysis from before. If you instead prefer a more conservative approach in merging the two masks, instead use:

```
immask in=cube1 region="mask(cube2r)" logic=and
```

some details on the interactive parts:

load a plane in **ds9** , e.g. via a loop in a cube that calls **mirds9**

- Click on Region ..
- Click on Polygon ..

Somewhere around/near your area of interest, drag with the left mouse a shape that will look like a square. if you click inside the square with the left mouse, you will see the corners being highlighted with little squares, as well as another "virtual square" around the visible square. The cursor shape, when inside the square, will be a big +, meaning the object has been selected (e.g. if you don't like it, you can then remove it from the list using the BACKSPACE key. In this mode you can also move the shape to another location in the image .

If you move the cursor near one of the inner squares, the shape of the cursor changes from + to [], and you can drag that point to another location, therefor changing the shape of the square. Moving the cursor near one of the 4 unmarked corners will allow you to change the shape of your new object.

If you move the mouse near one of the 4 edges of your object, you will see yet another change of shape of the cursor, now denoting that when you press the left mouse button, a new vertex (?) is added to the polygon at that location, and you can then immediately move that vertex to another location, therefore changing the shape of you object.

If you mouse is near one of the existing corners, and the shape indicates you could move that point to a new location, you can also delete that point from the list of vertices by hitting the BACKSPACE key.

Continue adding ...

<<we'll add some screendumps here>>

The middle mouse still allows you to move the selected point to the middle of the display window, whereas the right mouse allows you to change the color lookup table contrast (unfortunately the shape will disappear as the lookup table is changed on the fly).

Another hint:

set the colormap to greyscale, since the polygon has a green color (you can change that though)

sometimes shapes (default in green) get in the way of seeing where the emission is. There are two solutions to this:

- with the right mouse button depressed, change the color table a little bit. this has the side effect of disappearing the shapes, so you can sort of re-evaluate if they are blotted the right way

- step back (and maybe forth) a channel to sort of blink between images. With the 3D slider this is very easy to accomplish now.

- bring up another image, preferable in **karma**, so you can see how the emission is changing and blot correspondingly

when you load the new map, reload the previous mask, as from plane to plane the differences are often small, and you only need to adjust a few points, or slightly shift the object around a little bit.

Occasionally use "List Regions" to make sure you don't have any hidden regions. Sometimes you can wind up loading the same region twice, or create small one pixel regions by accident.

you can safely just double check on a certain plane

```
        cubemask cube1 region1  125 125
```

which would load plane 125 from cube1. If a region would be present (cube1/region1/saoregion.125) it would be copied to the default region name (cube1/region1/ds9.reg) for optional editing. After viewing `ds9.reg` would be copied back onto `saoregion.125`, which means if you don't edit i t, nothing will be changed.

Warning: make sure your regions don't run into any edge, as **saomask** (the **immask** program specifically) will not be able to handle this. In particular, the center of a pixel is an integer number, and it is easy to be near the edge and any pixel coordinate outside the pixel center will be taken as outside the image in MIRIAD.

So, cursor shapes are:

"arrow" - you cannot do a thing, you need to select an object (left click inside it) or create a new object with the left mouse by dragging it

"plus" - you can move the shape

box + arrows on diagonal sides - you can resize the shape, these normally only work on the 4 outside corners that are not on the vertices.

two spiral arrows - you can now rotate the object. This is activated if you hold the SHIFT key near one of the four corner points surrounding the object.

"dot in box" - you are on top of a point, and you can either move the point by dragging it

BACKSPACE - if an object is selected, the BACKSPACE will almost always remove this object

Note that in all these operations there is no “undo” operation in **ds9!**

- now do *Region | Save Regions*

make sure the region is saved in a file called 'ds9.reg' in your region directory

–

- TODOS:
- document how to steal the mask of another (smooth) cube
- how to boolean masks of two cubes , this is useful if you have cubemasks from an VEL-RA-DEC and VEL-DEC-RA cube. **done**

Combining masks I:

It is also easy to combine masks from the different masking techniques (e.g. rotcurmask, saomask):

```
rotcurmask in=cube0 out=cube0rcm ...
saomask cube1
immask in=cube1 region="mask(cube0rcm)" logic=or
```

where you can play with the logic.

Note this will overwrite the mask that existed in `cube1,` although you can always regenerate this with **`cubemask`**.

Combining masks II: recleaning with saomask regions

Another approach (see also BIMASONG's **moment.csh** script in $MIR/examples/bimasong) is to mask the dirty cube (as it comes out of **invert**) with regions from saomask. For this we need to apply these regions to the dirty cube and clean this cube again.

```
rotcurmask in=map out=map.rcmsk rotmod=mom1 mom2=mom2
maths exp=map out=maskedmap mask=map.rcmsk
```

you can then view the `maskedmap` and it will contain 0s outside the range that **rotcurmask** deemed reasonable to find emission.

There is something nasty about viewing what has been masked, see the **mask=** description in **`maths`**. Here is the example how to view a masked cube:

```
maths exp="cube*0+1" out=cube.tmp
maths exp=cube mask=cube.tmp out=cube1
```

Now `cube1` will contain both positive and negative signal, but only where you had masked (e.g. using **cubemask/saomask** or **rotcurmask**).

It is also possible to use **imsub** to split the cube into sub-cubes with different velocity ranges, and apply different styles of masking to each sub-cube, and then combine them again with **imcat**. Apart from warning the reader that (s)he better understand why this is necessary, this is also beyond the scope of this paper!

Now to the re-cleaning part. The mask (in the proper RA-DEC-VEL order) is copied to the dirty cube, and recleaned:

```
mossdi2 map=map1 beam=beam1 out=map.cc ...
mospsf beam=beam1 out=mospsf ...
imfit in=mospsf object=beam
set maj=`imfit in=mospsf object=beam|grep 'Beam Major'|..`
set min=
set pa=
restor map= beam= model=map.cc fwhm=$maj,$min pa=$pa out=map.cm
moment in=map.cm out=masked.mom0 mom=0
```

BIMASONG's moment.csh

The **moment.csh** script from the BIMASONG project has recently been polished up for the STINGS project.

A note on NEMO and MIRIAD images!

For historic reasons some of the software is available in NEMO only. Image files in NEMO are slightly different from those in MIRIAD. The scripts **mirds9** and **nds9** launch their respective images into **ds9**. The old-style NEMO image files have a fixed reference pixel at (0,0) and cannot be easily registered in ds9 together with MIRIAD or FITS files. This problem will disappear over time as different programs will be changed to use the new-style NEMO image file format.

NEMO new style images will have a refpix defined:

```
tsf my_image | grep refpix
double Xrefpix 0.00000
double Yrefpix 0.00000
double Zrefpix 0.00000
```

MIRIAD programs:

moment	take moment along an image cube axis [ccdmom]
velfit	fit rotation curve to a disk [rotcur]
velmodel	create model velocity field [ccdvel]
velimage	create cube from velocity field
rotcurmask	create mask cube from rotation curve
fits	convert MIRIAD image to/from fits
mirds9	script to display MIRIAD images, also FITS files, into ds9

NEMO programs:

snapgrid	grid particles and take moments to a map
snapgridsmooth	grid and smooth particles to a cube
snapmap	map particles for smooth interpolated maps
snappat	convert snapshot to a regular P-A-T cube
ccdmom	take moment along a cube [moment]
ccdfits	convert NEMO image to fits [fits]
fitsccd	convert fits to NEMO image [fits]
ccdvel	create model velocity field
rotcur	fit tilted ring model velocity field [velfit]
rotcurshape	fit parameterized rotation curves to velocity field
rotcurtab	print rotation curves based on rotcur shapes
rotcurves	plot rotation curves based on potentials
velfit	fit rotation curve to a disk (same as MIRIAD counterpart)
velmap	velocity field type map manipulations
runvelfitss07	fit non-circular motions to disk galaxies (SS07 method)
pvtrace	rotation curve from PV envelope tracing
tabplot	plot tables
perorb	compute periodic orbits in potentials
otos	convert orbits to snapshot
nds9	script to display NEMO images, also does FITS files, into ds9

KARMA for MIRIAD users:

There are two programs in KARMA probably the most useful to use for visual inspection and to get an idea of what's in the data: **kview** and **kpvslice**. In more recent version **kvis** is recommended instead of the now deprecated **kview**.

kview offers orthogonals views (slices) through the cube, and interactive spectra. **kpvslice** is more general and you can put arbitrary slits across an image and get pos-vel images out of the cube, and do this interactively. Both should be used to inspect the data before you endeavor into something like **cubemask**. The GIPSY program **sliceview** is another example of viewing data cubes.

There is a good manual for *Karma* available, but here's a quick starting guide for both:

kview:

Although deprecated, this is still the quickest way to view a data cube in orthagonal slices.

-click on *Set1*, SELECT your cube, and wait a few seconds until the cube is loaded. The first plane will be loaded.

-To change the colortable/contrast:

- click on *Intensity*, select *Pseudocolor* (8bit)

- select e.g. *Rainbow* from the right column

- you can move the ball in the left panel around if you wish, but there is another useful histogram based metho d to change the contrast, so:

- close this pseudoCmapwinpopup window

- click in intensity, select "Iscale for dataset 1", you should see a histogram pop up

- click with the left mouse button a little left to the peak of noise level (or whereever you like :-). This level will be you lower cutoff.

- click with the right mouse button towards the right part where you can see signal sticking out of the noise (or whereever). This level will be your upper cutoff

- with the middle mouse you can now pick up the little ball

in the middle and drag it around and see the result in the display window.

- you can leave this window up as you may want to change it in subsequent viewing

- To make a movie:

- click on view, this will popup another window (View Control)

- click on movie in this View Control window, it pops up Animation Contr ol

- click on start movie. it will probably go too fast, so add some delay using the Frame Interval (ms) controls

- click on stop movie. With the middle button you can drag the channel bar in the black window

- To look at spectra

- select from the View Control window a "Profile Mode" like Line. If you move the cursor in the kview window, you should see spectra being animated as you move around.

- you can also select a "Profile Mode: Box Sum". This one is a little tricky again, as you need the drag a box with the Middle Button (!). If you use the (perhaps more intuitive) left button, it will zoom the image window instead. If so, use "Zoom" and select "Unzoom" to get back to the original display.

- To add some labels: (most windows don't come with much window dressup)

  - Click on Overlays, select "Axis labels"
  - Toggle "Display Axis Labels" in the right way, you should see results right away
  - close the window

kpvslice:

- two windows will initially pop up: kpvslice, in which an image will have to be loaded (either derived from the cube, or indepantly), and a Slice Window, which you will see generator from actions you do in the kpvslice window.

- click on "Cubes" and select the cube you want to work with

- if you have an Image (e.g. a moment 0 map), click on Images, and load the image in that way. Otherwise, if you just quickly want to derive it from the cube, click on "Image Mode: loaded image" and select

  "0^{th} moment". A new window pops up, just click on "Apply parameters" and a new image will be loaded now. You can now also close the Moment Generator image/window

- With the middle button drag a line (slice) across the image and in the other (slice) window you will now see a position-velocity map show up.

- with the right mouse button you can pickup the line in the middle and drag it to any position in the image, and again you will see the Pos-Vel diagram change correspondingly.

- also with the right mouse button, you can rotate the line around its center by pick it up and either end of the line

- the Pos-Vel diagram sometimes has the wrong aspect ratio, to fix this select "Zoom Policy" from the "Zoom" pulldown in the Slice Window, and simply deselect the "Fix Aspect" toggle button. You can now resize (using normal X windows controls) this window to suit your needs, I prefer to make the pos-vel beam about circular.

kvis:

There are many more details in these two programs, but there is already a good manual on the *Karma* home page that covers everything in far more detail.

See also http://www.atnf.csiro.au/karma/.

Acknowledgements:

References