1 Random Sampling from Distribution Functions

Simple demonstration: sampling from a normal distribution.

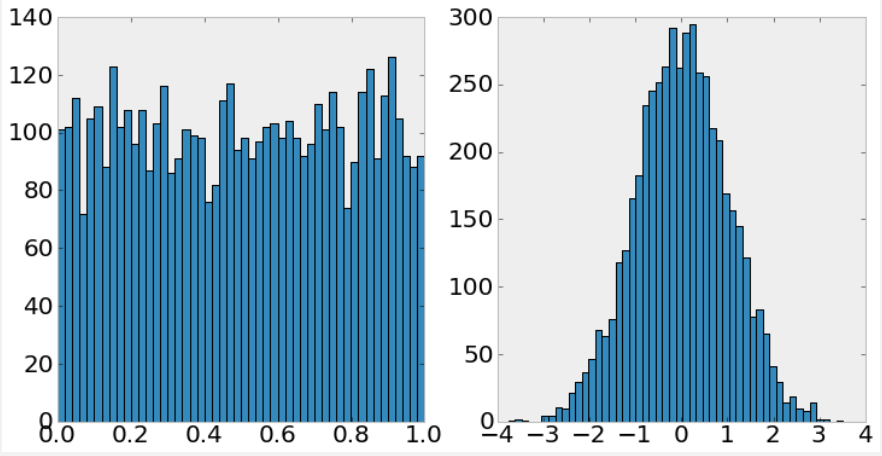
Goal: Given numbers sampled from a uniform distribution, instead get numbers sampled from a different distribution.

Starting with the simplest case: we already have a function that generates numbers sampled from a uniform distribution and a normal distribution. What do those look like?

```
# setup stuff. skipped in slideshow
import itertools
import scipy.stats
rcParams['patch.facecolor'] = "348ABD"#, 7A68A6, 467821, CF4457, 188487, E24A33"
rcParams['axes.color_cycle'] = "348ABD, A60628, 7A68A6, 467821, CF4457, 188487, E24A33".
    split(", ")
rcParams['axes.facecolor'] = "eeeeee" # axes background color
rcParams['axes.edgecolor'] = "bcbcbc" # axes edge color
rcParams['axes.titlesize'] = "x-large" # fontsize of the axes title
rcParams['axes.labelsize'] = "large" # fontsize of the x any y labels
rcParams['lines.linewidth'] = 2
rcParams['font.size'] = 20
```

```
/Users/adam/virtual-python/lib/python2.7/site-packages/scikits/__init__.py:1: UserWarning: Module argpar
  __import__('pkg_resources').declare_namespace(__name__)
```

```
ru = np.random.rand(5000) # uniformly distributed data
rn = np.random.randn(5000) # normally distributed data
close(1); figure(1,figsize=(12,6)); clf();
subplot(121); h=hist(ru,bins=50)
subplot(122); h=hist(rn,bins=50)
```



Since we already have a tool capable of generating a normal distribution, it makes the most sense to provide our first test.

First, let's define the distribution properties, though. "Probability Distributions" are defined by a "Probability Distribution Function" and can also be characterized by a "Cumulative Distribution

Function".

A Probability Distribution Function, or PDF, tells you for each number, how likely it is that number will appear. The "D" in PDF can also mean "Density", because there is some "probability per dx" or "probability density" for continuous distributions.

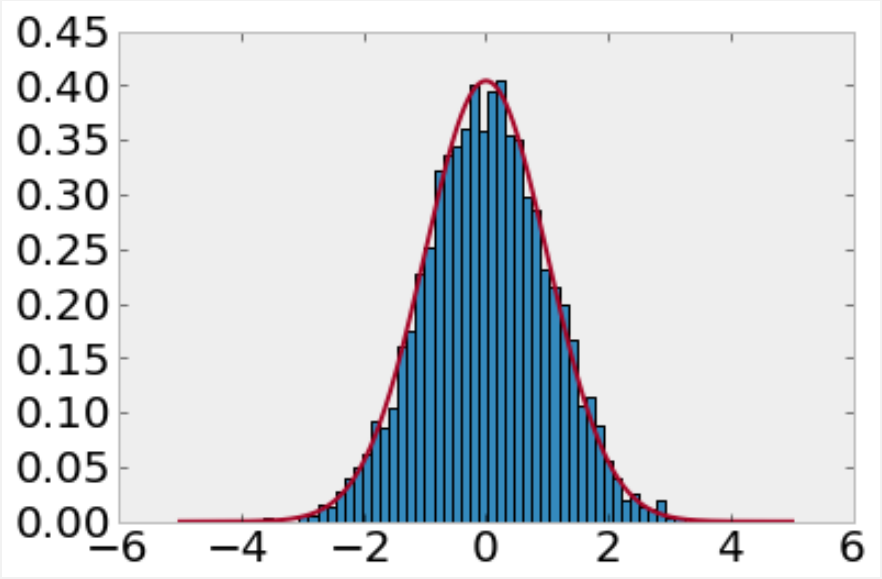
A Cumulative Distribution Function says instead, for each number, how likely it is that number *or a smaller number* will appear.

```
# Normal distribution:
def p_normal(x, sigma=1, x0=0):
    """ Probability Density Function of the Normal distribution with width `sigma` and
        center `x0` """
    return np.exp(-(x-x0)**2/(2.*sigma**2))

import scipy.special # need a special function for this
def c_normal(x, sigma=1, x0=0):
    """ Cumulative Distribution Function of the Normal distribution with width `sigma`
        and center `x0 """
    return 1-0.5*(1-scipy.special.erf((x-x0)/((2.*sigma**2)**0.5)))
```

Let's look at what the PDF shows us first:

```
x = linspace(-5,5,1000)
clf(); h,l,p=hist(rn,bins=50,normed=True); plot(x,p_normal(x)*h.max())
```

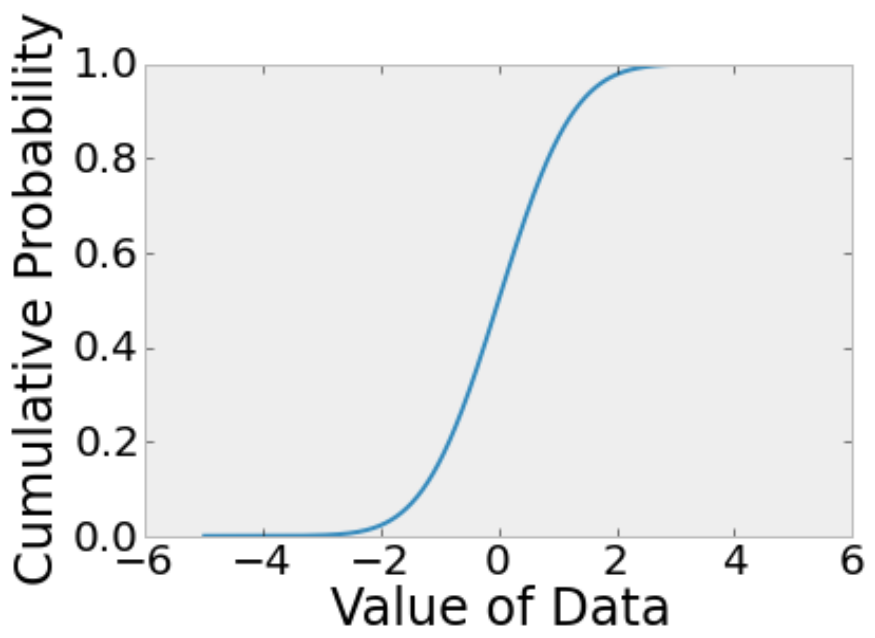
```
[<matplotlib.lines.Line2D at 0x109e21150>]
```



The PDF shows you approximately the bin heights (how many points fell into that range) for our normal distribution.

The CDF looks different:

```
# The Cumulative Distribution Function
x = linspace(-5,5,1000)
clf(); plot(x,c_normal(x))
xlabel("Value of Data"); yl=ylabel("Cumulative Probability")
```



We can use the CDF to "invert" the distribution. i.e., given a *probability* value, determine the number that corresponds to it. Let's explore this idea further.

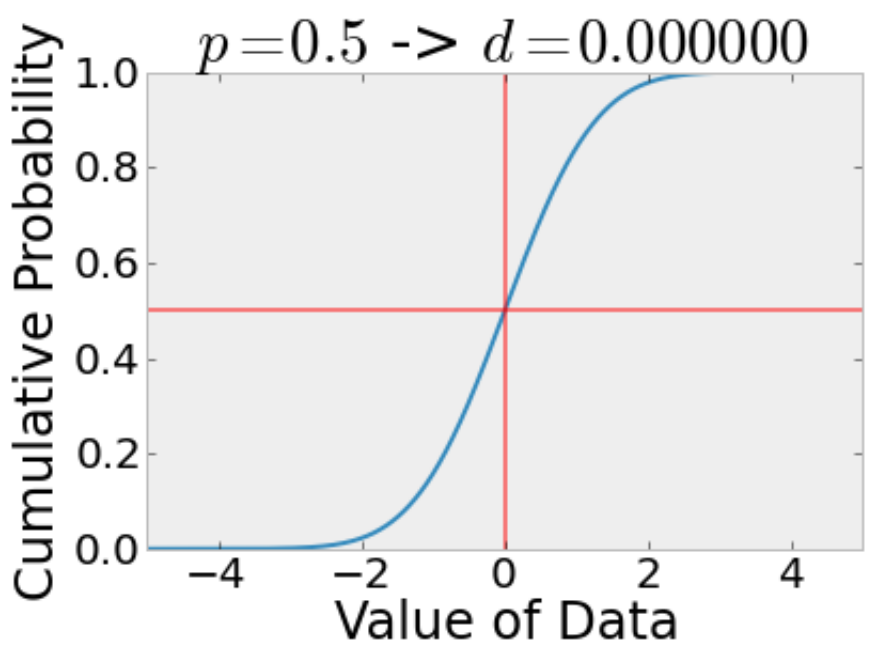
Probabilities tell you how likely an event is to occur. For a coin flip, the probability of heads or tails is always 0.5, so there is an equal chance of getting either flip.

Zero probability, $p = 0$, means the event can never happen. $p = 1$ means it will definitely happen. Therefore, everything that can happen must have some probability value between 0 and 1. Don't worry too much about the logic of this, but know that the range of the CDF and PDF are always $[0, 1)$.

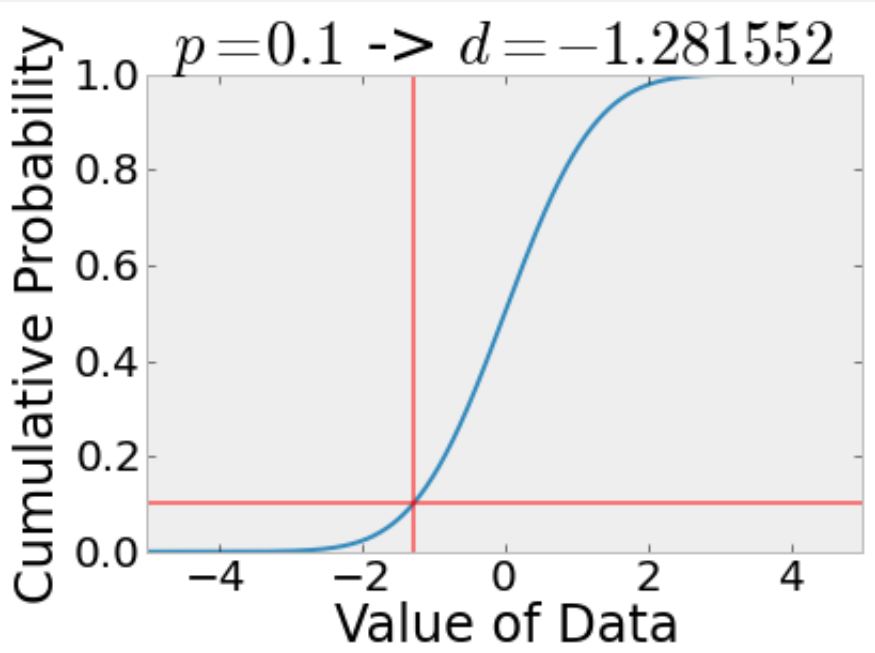
So if we pick different values of the probability p, we want to get back out the values of the data that correspond to that likelihood.

Let's see what happens for a few values. Let's try $p = 0.5$, $p = 0.1$, and $p = 0.95$. We'll plot their probabilities on top of the CDF and see what values they correspond to.

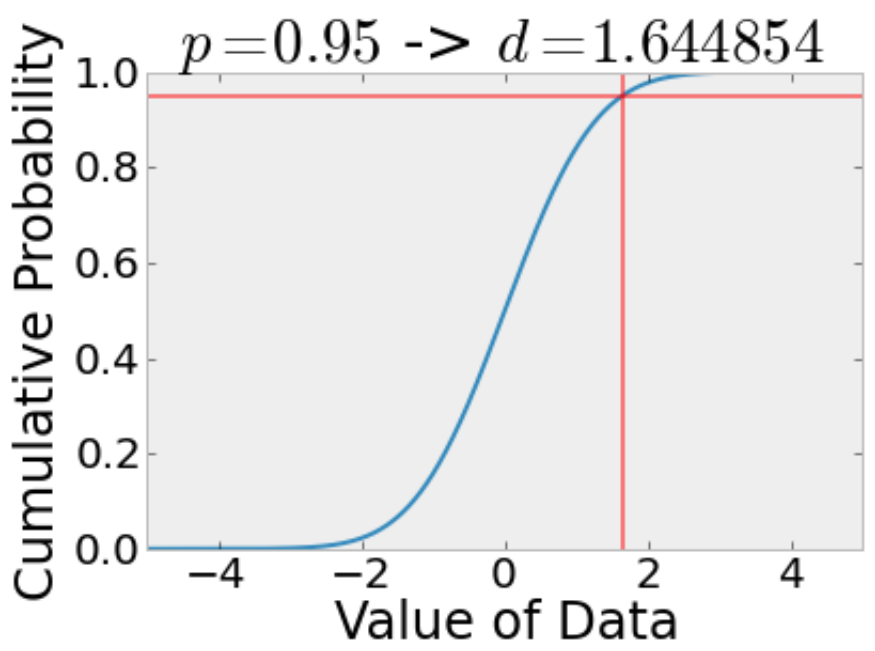
```
clf(); plot(x,c_normal(x))
xlabel("Value of Data"); yl=ylabel("Cumulative Probability"); title("$p=0.5$ -> $d=%f$"
    % scipy.stats.norm.ppf(0.5))
plot(x,x*0+0.5,'r',alpha=0.5); plot([scipy.stats.norm.ppf(0.5)]*2,gca().get_ylim(),'r',
    alpha=0.5)
a=axis([-5,5,0,1])
```




```
clf(); plot(x,c_normal(x))
xlabel("Value of Data"); yl=ylabel("Cumulative Probability"); title("$p=0.1$ -> $d=%f$"
    % scipy.stats.norm.ppf(0.1))
p=plot(x,x*0+0.1,'r',alpha=0.5); plot([scipy.stats.norm.ppf(0.1)]*2,gca().get_ylim(),'r'
    ,alpha=0.5)
a=axis([-5,5,0,1])
```

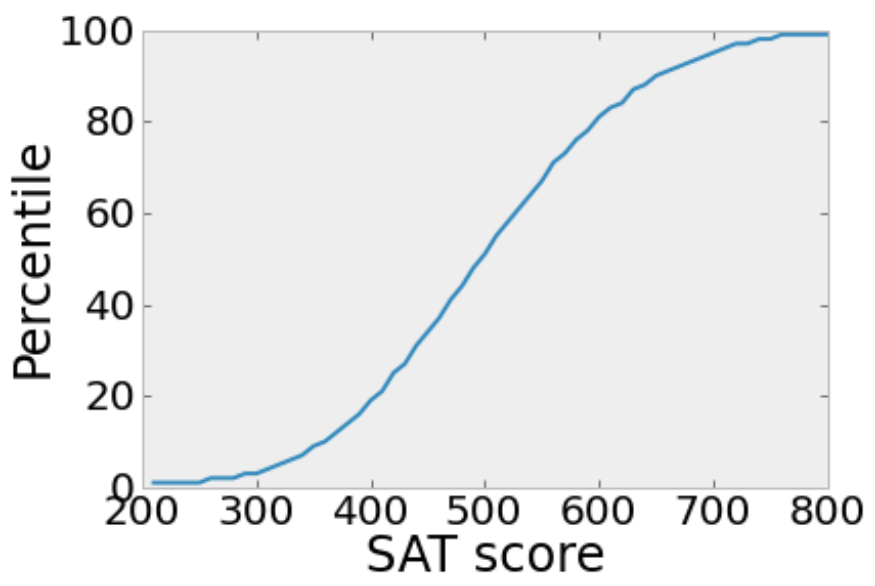


```
clf(); plot(x,c_normal(x))
xlabel("Value of Data"); yl=ylabel("Cumulative Probability"); title("$p=0.95$ -> $d=%f$"
     % scipy.stats.norm.ppf(0.95))
p=plot(x,x*0+0.95,'r',alpha=0.5); plot([scipy.stats.norm.ppf(0.95)]*2,gca().get_ylim(),'
    r',alpha=0.5)
a=axis([-5,5,0,1])
```



The inverse function, which takes a *probability* and returns the value of the data is called a "quantile function" or a "percent point function". You may be familiar with this concept from, e.g., SAT data. The "percentile" associated with an SAT score is from a cumulative distribution function. It tells you what fraction of scores are below a given score.

```
scores,mathp,writep,verbp = np.loadtxt('sat_data.txt').T
clf(); xlabel("SAT score"); ylabel("Percentile")
p=plot(scores,mathp)
```



The `ppf` is the inverse of the `cdf` function and can't always be computed analytically. Sometimes you have to approximate it. The normal distribution is actually one of these cases. Luckily, most languages implement very nice approximations for the inverse CDF. But if you want to approximate an inverse-CDF yourself, you can follow this process:

1. Compute the CDF on a fine grid `x`. This means you'll have some array `cdf` with values from 0-1.
2. Find the nearest value in that array to your probability value. In python, you can use `a=np.searchsorted(p)`. In IDL, the easiest thing to do is `x=min(abs(cdf-p),a)`
3. Use your initial grid `x` to find the value corresponding to the input probability: `x[a]`

We'll just do one simple example here, using the SAT data:

```
# our x-array is "scores"
p = 0.85
scores,mathp = sort(scores),sort(mathp) # both must be in increasing order
a = np.searchsorted(mathp,p*100)
print "For probability p=%f, the index is %i, and the score is %f" % (p,a,scores[a])
```

```
For probability p=0.850000, the index is 42, and the score is 630.000000
```

We'll do a more involved example with the Normal distribution now. We defined `c_normal` above.

```
# make our "x-array" go from -3-sigma to 3-sigma
x = linspace(-3,3,1000)
# compute the CDF
cdf = c_normal(x)
# Hunt for our p-value
p = 0.85
a = np.searchsorted(cdf,p)
```

```
print "For probability p=%f, the index is %i, and the score is %f" % (p,a,x[a])
```

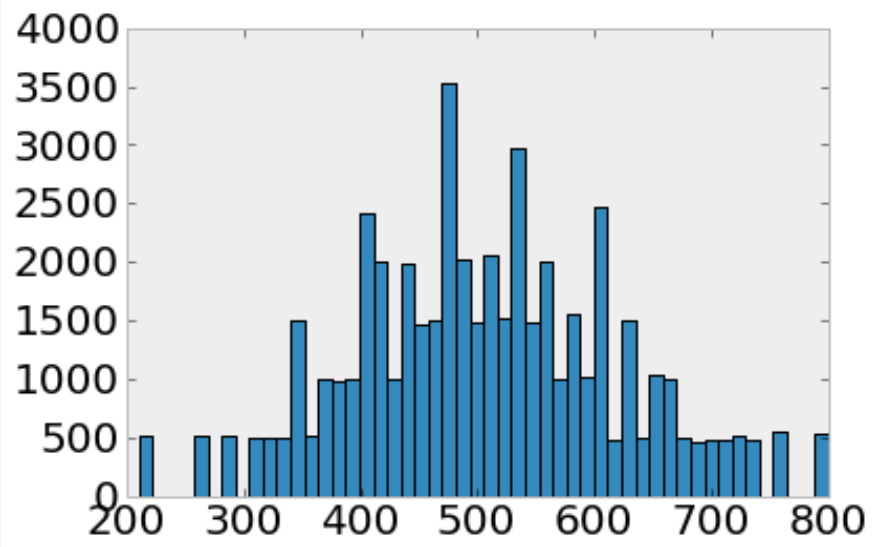
```
For probability p=0.850000, the index is 673, and the score is 1.042042
```

We now know the process for sampling from a distribution, but really what we want is a way to turn random numbers between 0 and 1 into numbers sampled from a different distribution. We'll define a function to do this.

```
def ppf_sat(probability, cdf=sort(mathp), scores=sort(scores)):
    """ Given a cumulative distribution function that goes from 0-100 and a probability
        from 0-1,
    return the associated score."""
    a = np.searchsorted(cdf,probability*100)
    if a == len(cdf):
        a-=1
    return scores[a]
```

Now we can randomly sample scores from the distribution. We select random numbers with values $[0, 1)$ and put them into the `ppf_sat` function.

```
random_values = np.random.rand(50000) # 100 random numbers
random_scores = [ppf_sat(r) for r in random_values]
# Show them in a histogram:
clf(); h,l,p=hist(random_scores,bins=50)
```



So this is a distribution showing the scores for 50,000 students taking the SAT using the data we were given about the percentiles. It's actually a fairly terrible-looking distribution, most likely because the data we were given was not finely-spaced enough. When writing your own inverse-CDF, or ppf, functions, it's better to use an analytic version or a very good approximation, not the junky one we used for SAT scores.

2 Stellar Distributions

Moving on, what we really want to do is create a realistic-looking star cluster.

We need to sample from a few distributions:

1. The spatial distribution of stars. Where are they?
2. The mass distribution of the stars.
3. The velocity distribution of stars. How fast are they moving? If they're moving too fast, the cluster will "explode", if too slow, it will "collapse"

The first thing we'll do is define a spatial distribution function. I explicitly go through the derivation on the next few slides, but you don't really need to know it - I just had to do the derivation to get the functions we want.

Define $p(r)$, assuming $\alpha > 1$:

$$p(r) = Cr_0^{-\alpha} \text{ if } r < r_0$$

$$p(r) = Cr^{-\alpha} \text{ if } r > r_0$$

By definition, the integral of the PDF is 1, so we can use this to compute the constant for normalization:

$$\int_0^\infty p(r)dr = 1$$

$$\int_0^{r_0} Cr_0^{-\alpha}dr + \int_{r_0}^\infty Cr^{-\alpha}dr = 1$$

$$Cr_0^{1-\alpha} + \left[\frac{1}{1-\alpha}Cr^{1-\alpha}\right]_{r_0}^\infty = Cr_0^{1-\alpha} - \frac{Cr_0^{1-\alpha}}{1-\alpha} = C\frac{(-\alpha)r_0^{1-\alpha}}{1-\alpha} = 1$$

$$C = \left(1 - \frac{1}{\alpha}\right) r_0^{\alpha-1}$$

These slides can be skimmed over; no need to spend time on them.

Derive the Cumulative Distribution Function (CDF) by integrating the PDF:

$$c(r) = \int_{-\infty}^r p(x)dx$$

$$c(r) = \int_0^r Cr_0^{-\alpha}dx = Cr_0^{-\alpha}r = (1 - \frac{1}{\alpha})r_0^{-1}r = \frac{(\alpha-1)r}{\alpha r_0} \text{ if } r < r_0$$

$$c(r) = \int_0^{r_0} Cr_0^{-\alpha}dx + \int_{r_0}^r Cx^{-\alpha}dx = (1 - \frac{1}{\alpha}) + \left[C/(1-\alpha)x^{1-\alpha}\right]_{r_0}^r$$

$$= (1 - \frac{1}{\alpha}) + \frac{-1}{\alpha}r_0^{\alpha-1}(r^{1-\alpha} - r_0^{1-\alpha})$$

$$= (1 - \frac{1}{\alpha}) + \frac{1}{\alpha} + \frac{-1}{\alpha}((r/r_0)^{1-\alpha})$$

$$= 1 - \frac{1}{\alpha}(r/r_0)^{1-\alpha} \text{ if } r > r_0$$

Solve $c(r)$ for r to get $q(p)$:

$$q(p) = r = \frac{c(r) * r_0 * \alpha}{\alpha - 1} \text{ if } r < r_0$$

$$q(p) = r = (-\alpha * (c(r) - 1))^{1/(1-\alpha)} r_0$$

$\alpha = 2$ case:

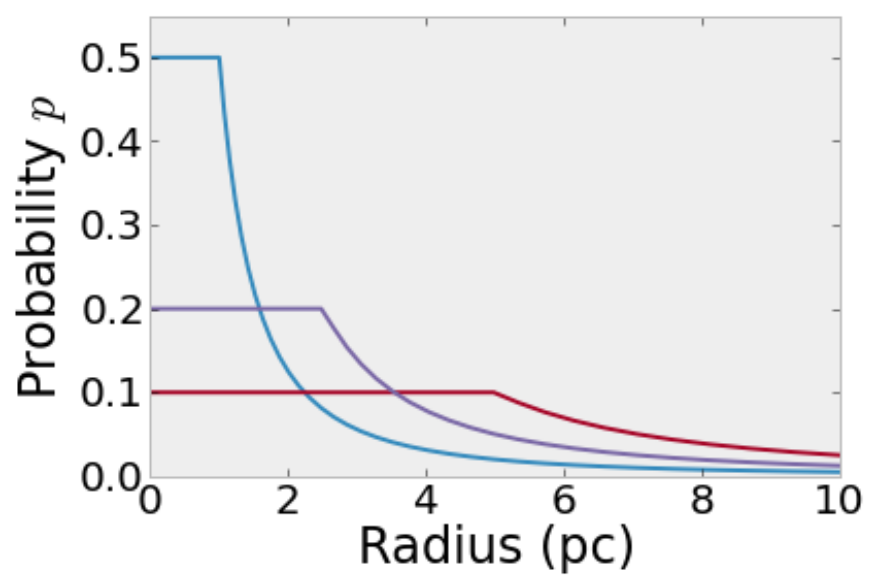
$$q(p) = p * r_0 * 2$$

$$q(p) = (-2 * (p - 1))^{-1} * r_0 = \frac{r_0}{2(1-p)}$$

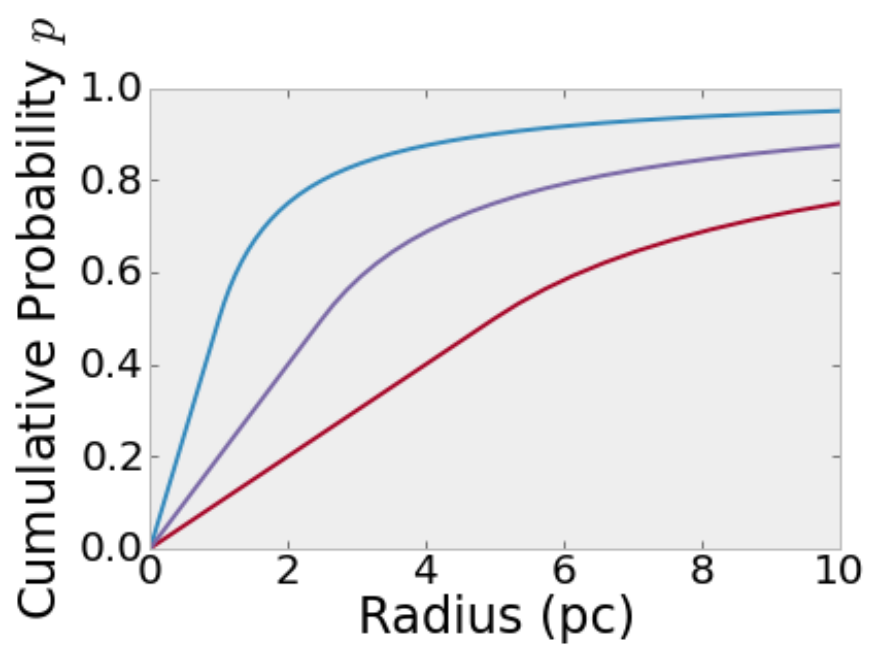
```
# Probability distribution functions describing a cut-off power law
def p_cluster(r, r0=1., alpha=2):
    C = r0**(alpha-1)*(1-1./alpha)
    prob = C/r0**alpha * (r<=r0) + C/r**alpha * (r>r0)
    return prob

def c_cluster(r, r0=1., alpha=2):
    p1 = (alpha-1.) * r / (alpha*r0) * (r<=r0)
    p2 = (1. - 1./alpha * (r/r0)**(1-alpha)) * (r>r0)
    return p1 + p2

def q_cluster(p, r0=1.,alpha=2):
    r1 = (-alpha*(p-1))**(1/(1.-alpha)) * r0
    r2 = p * r0 * alpha / (alpha-1.)
    return r1*(r1>r0)+r2*(r2<=r0)
```

Some plots of the distribution function. First, the PDF:

```
r = logspace(-2,1,100)
alpha=2
clf(); xlabel("Radius (pc)"); ylabel("Probability $p$")
plot(r,p_cluster(r,alpha=alpha))
plot(r,p_cluster(r,r0=5,alpha=alpha))
plot(r,p_cluster(r,r0=2.5,alpha=alpha))
a=axis([0,10,0,0.55])
```



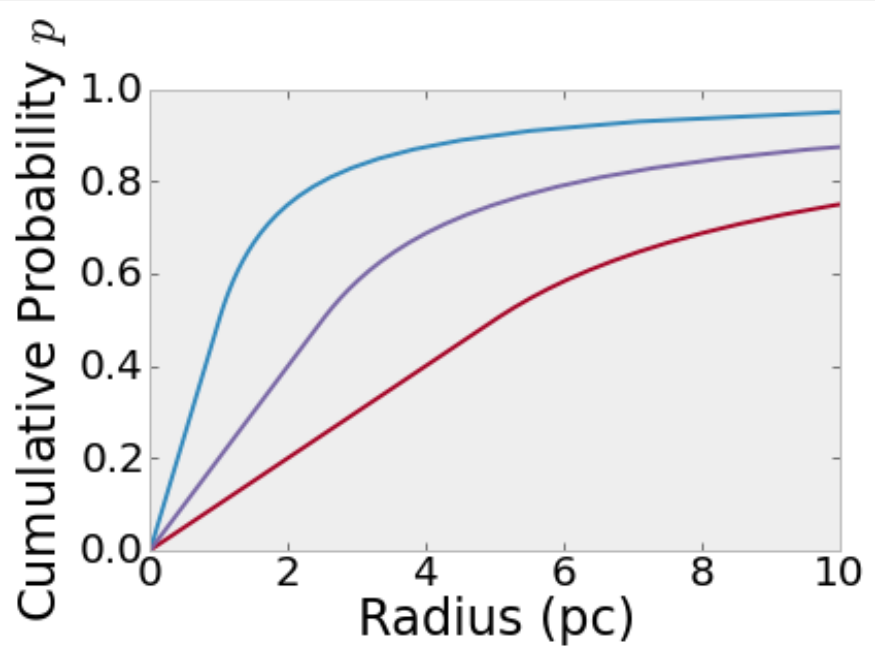

```
clf();xlabel("Radius (pc)"); ylabel("Cumulative Probability $p$")
plot(r,c_cluster(r,alpha=alpha))
plot(r,c_cluster(r,r0=5,alpha=alpha))
plot(r,c_cluster(r,r0=2.5,alpha=alpha))
xlim=gca().get_xlim()
ylim=gca().get_ylim()
```



Just to prove that the inverse function, the `ppf` or `quantile` function, works, we'll plot probability on the y-axis again, even though it's the independent variable in this set of plots:

```
clf();xlabel("Radius (pc)"); ylabel("Cumulative Probability $p$")
p=linspace(0.001,0.99)
plot(q_cluster(p,alpha=alpha),p)
plot(q_cluster(p,r0=5,alpha=alpha),p)
plot(q_cluster(p,r0=2.5,alpha=alpha),p)
gca().set_xlim(*xlim)
```

```
(0.0, 10.0)
```

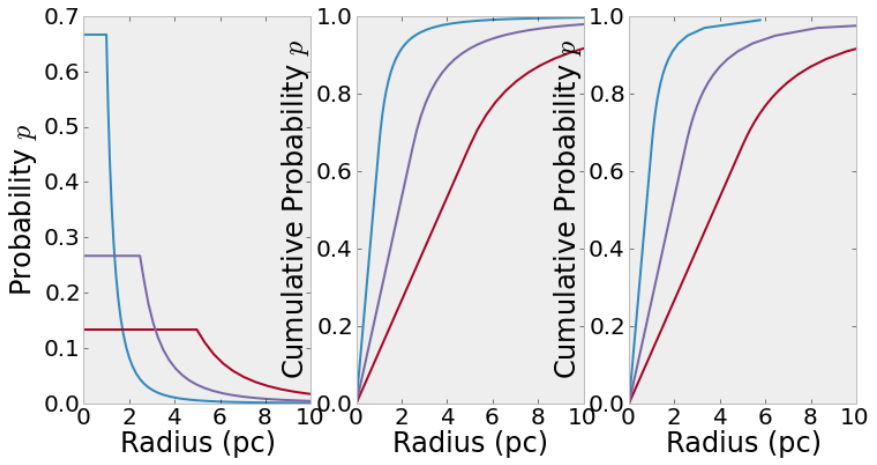


Now we'll do the same thing with $\alpha = 3$ instead of $\alpha = 2$. That means a "steeper" profile, or more centrally concentrated.

```
r = logspace(-2,1,100)
alpha=3
figure(figsize=[12,6])
clf(); subplot(131); xlabel("Radius (pc)"); ylabel("Probability $p$")
subplots_adjust(hspace=0.5)
plot(r,p_cluster(r,alpha=alpha))
plot(r,p_cluster(r,r0=5,alpha=alpha))
plot(r,p_cluster(r,r0=2.5,alpha=alpha))
subplot(132); xlabel("Radius (pc)"); ylabel("Cumulative Probability $p$")
plot(r,c_cluster(r,alpha=alpha))
plot(r,c_cluster(r,r0=5,alpha=alpha))
plot(r,c_cluster(r,r0=2.5,alpha=alpha))
xlim=gca().get_xlim()
subplot(133); xlabel("Radius (pc)"); ylabel("Cumulative Probability $p$")
p=linspace(0.001,0.99)
plot(q_cluster(p,alpha=alpha),p)
plot(q_cluster(p,r0=5,alpha=alpha),p)
plot(q_cluster(p,r0=2.5,alpha=alpha),p)
```

```
gca().set_xlim(*xlim)
```

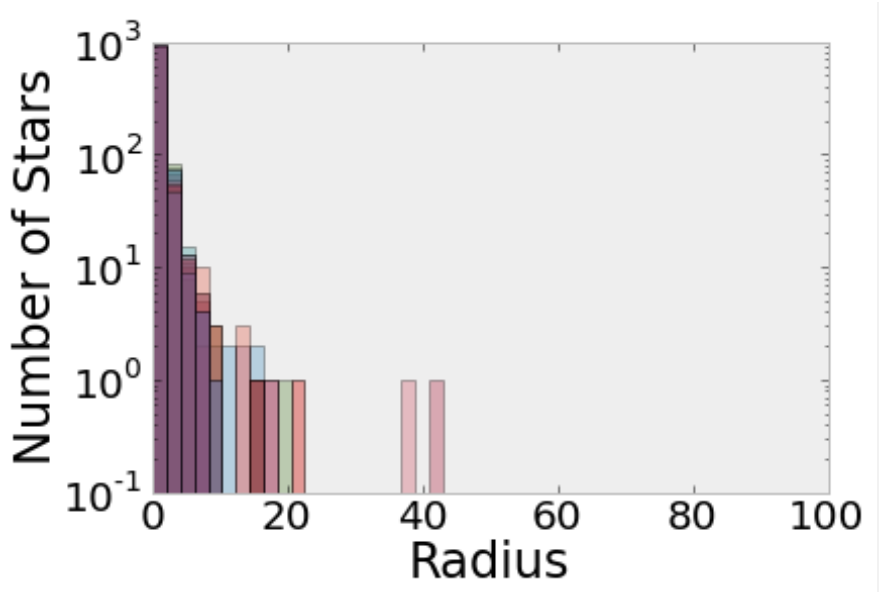
```
(0.0, 10.0)
```



Now some random sampling to show what the cluster histograms look like.

```
figure(1); clf()
figure(2); clf()
for ii in xrange(10):
    r = np.random.rand(1000)
    figure(1); xlabel("Radius"); ylabel("Number of Stars")
    hist(q_cluster(r,alpha=3),bins=linspace(0,100),log=True,alpha=0.3)
    print "rmax: ",r.max()," q(rmax): ",q_cluster(r.max())," c(q(rmax)): ",c_cluster(
        q_cluster(r.max()))
```

```
rmax:  0.998624442079  q(rmax):  363.488874015  c(q(rmax)):  0.998624442079
rmax:  0.9998050198  q(rmax):  2564.36295164  c(q(rmax)):  0.9998050198
rmax:
0.998031675911  q(rmax):  254.023208305  c(q(rmax)):  0.998031675911
rmax:  0.999128956923  q(rmax):  574.024423638  c(q(rmax)):  0.999128956923
rmax:
0.999762237776  q(rmax):  2102.94129417  c(q(rmax)):  0.999762237776
rmax:  0.998691448685  q(rmax):  382.101943059  c(q(rmax)):  0.998691448685
rmax:
0.9992105474  q(rmax):  633.350248004  c(q(rmax)):  0.9992105474
rmax:  0.996302024336  q(rmax):  135.209110455  c(q(rmax)):  0.996302024336
rmax:
0.998554086535  q(rmax):  345.802160381  c(q(rmax)):  0.998554086535
rmax:  0.998896437745  q(rmax):  453.07820005  c(q(rmax)):  0.998896437745
```




```
<matplotlib.figure.Figure at 0x10b670cd0>
```

```
def rand_xyz(r):
    """This function takes a scalar random variable and turns it into a 3D random vector
        """
    randx,randy,randz = rand(r.size)*2-1,rand(r.size)*2-1,rand(r.size)*2-1
    norm = (randx**2+randy**2+randz**2)**0.5
    randx = randx/norm * r
    randy = randy/norm * r
    randz = randz/norm * r
    return randx,randy,randz
```

To keep the cluster bound, we need to do some coordinate conversions.

$$x = r\cos(\phi)\sin(\theta)$$
$$y = r\sin(\phi)\sin(\theta)$$
$$z = r\cos(\theta)$$

$$r = (x^2 + y^2 + z^2)^{1/2}$$
$$\theta = \arccos(z/r)$$
$$\phi = \arccos\left(\frac{x}{r\sin(\theta)}\right)$$

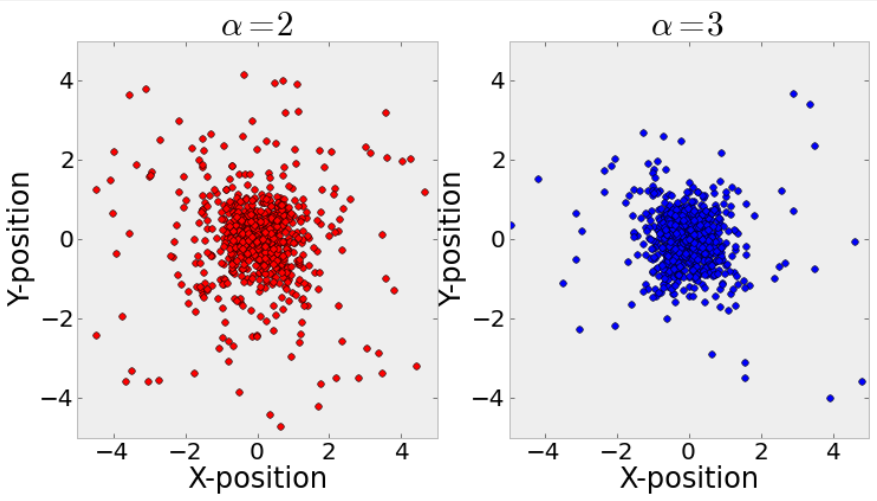
Use `rand_xyz` to turn our random radial locations into clusters

```
cluster2_r = q_cluster(rand(1000),alpha=2)
cluster3_r = q_cluster(rand(1000),alpha=3)
```

```
cluster2x,cluster2y,cluster2z = rand_xyz(cluster2_r)
cluster3x,cluster3y,cluster3z = rand_xyz(cluster3_r)
```

This is what the clusters look like with $\alpha = 2$ and $\alpha = 3$.

```
f=figure(figsize=[12,6]); clf(); s=subplot(121)
p=plot(cluster2x,cluster2y,marker='.',markersize=10,color=(1,0,0,0.2),linestyle='none')
t=title("$\\alpha=2$"); xl=xlabel("X-position"); yl=ylabel("Y-position")
a=axis([-5,5,-5,5])
s=subplot(122)
p=plot(cluster3x,cluster3y,marker='.',markersize=10,color=(0,0,1,0.2),linestyle='none')
t=title("$\\alpha=3$")
a=axis([-5,5,-5,5]); xl=xlabel("X-position"); yl=ylabel("Y-position")
```



Making a cluster.

1. Determine the locations of the stars [CHECK]
2. Determine the masses of the stars (use simplified log-normal IMF)
3. Determine the velocities of the stars via energy in the system

```
# deprecated
def star_velocity(enclosed_mass, radius, starmass=1.):
    " Virialized: v = v_esc/2 "
    energy = enclosed_mass * 6.67e-8 / radius
    velocity = energy**0.5
    return velocity
```

Initial mass functions are a bit awkward to deal with; I'm not going to set up realistic IMFs but just give a simple log-normal, which is "kinda" realistic.

Lognormal means taking Gaussian random variables, and taking 10 to the power of those. So if our normal random variable is x, the lognormal will be 10^x

```
x_randn = np.random.randn(1000)
star_masses = 10**x_randn * 2e30 # in kilograms
star_radii = cluster3_r * 3.08e16 # in meters
```

Now we'll make functions to determine the stars' speeds and then turn those speeds into velocities. In order to make sure our cluster is bound but not collapsing, we want the stars' total kinetic energy (the energy of their motion) to be approximately the same as the gravitational energy. This means we're using the Virial theorem.

$$2K \sim U$$

$$2\frac{1}{2}mv^2 \sim \frac{GMm}{R}$$

In this case, m is the mass of a star, M is the enclosed mass of the cluster out to radius R, and v is the velocity of the star.

```
def star_speeds(star_masses,star_radii):
    sortarr = np.argsort(star_radii)
    inverse_sortarr = np.argsort(sortarr)
    star_speeds = np.zeros(star_masses.size)

    cumul_mass = np.cumsum(np.sort(star_masses))
    enclosed_mass = np.concatenate([[0],cumul_mass[1:]])
    star_speeds = (6.67e-11*enclosed_mass/star_radii[sortarr])**0.5
    return star_speeds[inverse_sortarr]

def star_velocities(speeds):
    return rand_xyz(speeds)
    nstars = speeds.size
    phi = np.random.rand(nstars) * 2 * pi
    theta = np.random.rand(nstars) * pi
    vx = cos(phi) * sin(theta) * speeds
    vy = sin(phi) * sin(theta) * speeds
    vz = cos(theta) * speeds

    return vx,vy,vz
```

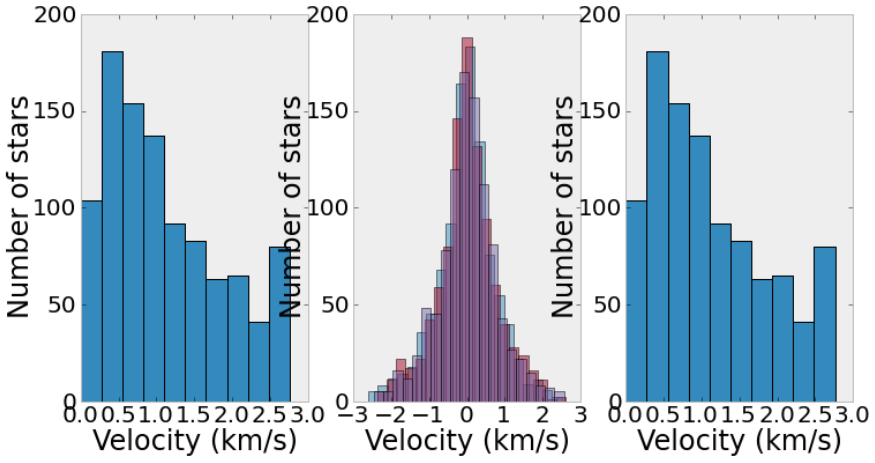
```
speeds = star_speeds(star_masses,star_radii)
velocities = star_velocities(speeds)
```

Plot some histograms of the star speeds and velocities - these are sanity checks.

```
figure(figsize=[12,6]); clf(); subplot(131)
hist(speeds/1e3)
xlabel("Velocity (km/s)"); ylabel("Number of stars")
subplot(132)
hist(velocities[0]/1e3,alpha=0.5,bins=20)
hist(velocities[1]/1e3,alpha=0.5,bins=20)
hist(velocities[2]/1e3,alpha=0.5,bins=20)
```

```
xlabel("Velocity (km/s)"); ylabel("Number of stars")
subplot(133)
hist(np.sum(np.array(velocities)**2,axis=0)**0.5/1e3)
xlabel("Velocity (km/s)"); ylabel("Number of stars")
```

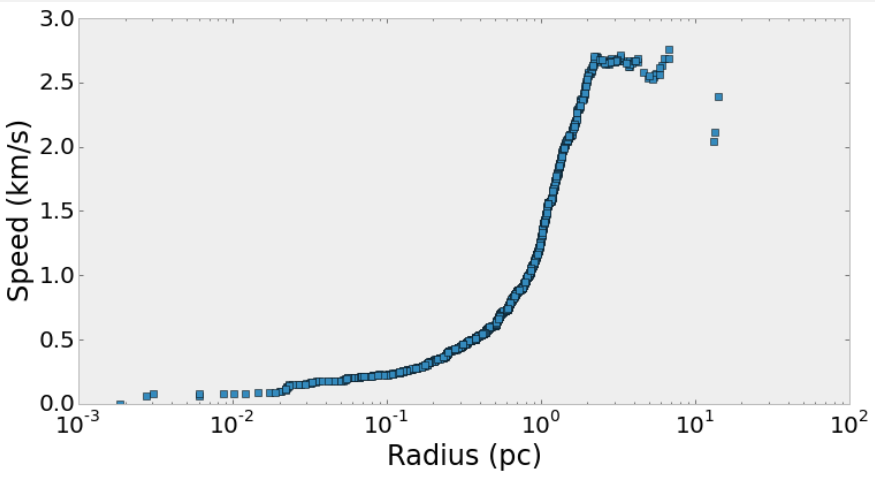
```
<matplotlib.text.Text at 0x10c900a50>
```



Another cool thing to plot is speed vs. radius: Stars move faster at larger radii, then drop.

```
figure(figsize=[12,6]); clf()
semilogx(star_radii/3e16,speeds/1e3,'s')
xlabel("Radius (pc)"); ylabel("Speed (km/s)")
```

```
<matplotlib.text.Text at 0x10b642f10>
```



At this point, we have all the tools needed to set up our initial conditions for the N-body simulation.

So we will! Using Jordan's solver.

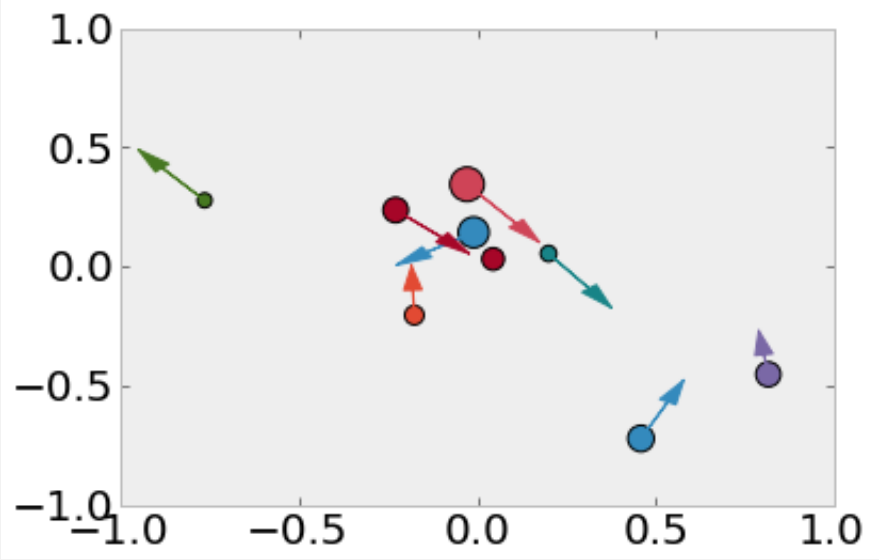
```
import nbody
import nbody.InitialConditions as IC
```

```
nstars = 10
pc = 3.08e16 # m
msun = 2e30 # kg
x_randn = np.random.randn(nstars) / 5.
star_masses = 10**x_randn * msun # in kilograms
cluster3_r = q_cluster(rand(nstars),alpha=3)
cluster3x,cluster3y,cluster3z = np.array(rand_xyz(cluster3_r)) * pc # meters
star_radii = cluster3_r * pc # in meters
speeds = star_speeds(star_masses,star_radii) # m/s
velocities = star_velocities(speeds)
```

This plot shows the star masses by the size of the star symbol, and their velocity direction and speed by arrows.

```
# star vector field
clf()
ax=gca()
colors = itertools.cycle(rcParams['axes.color_cycle'])
for x,y,z,c,m,vx,vy,vz in zip(cluster3x,cluster3y,cluster3z,colors,star_masses,*
    velocities):
    ax.arrow(x/pc,y/pc,vx/1e3,vy/1e3, head_width=0.05, head_length=0.1, fc=c, ec=c)
    scatter(x/pc,y/pc,c=c,s=100*m/msun,marker='o')
zoomscale = 1
axis([-zoomscale,zoomscale]*2)
```

```
[-1, 1, -1, 1]
```



```
thin=1
start = IC(star_masses.size/thin,dims=3)
start.set_masses(star_masses[::thin]/sqrt(6.67e-11))
start.set_positions(pos=2*np.array([cluster3x,cluster3y,cluster3z]).T[::thin,:])
start.set_velocities(vel=np.array(velocities).T[::thin,:])
```

Set up for the N-body solver. Next step will be to run it for a few years...

```
G = nbody.SolveGravity.Gravity(start)
```

```
G(2*365*86400,86400)
```

```
n-body:   0% |                                                  | ETA:  --:--:--
n-body:   1% |                                                  | ETA:  0:00:04
n-body:   2% |-                                                 | ETA:  0:00:04
n-body:   3% |\                                                 | ETA:  0:00:04
n-body:   4% |||                                                | ETA:  0:00:04
n-body:   5% |//                                                | ETA:  0:00:03
n-body:   6% |---                                               | ETA:  0:00:03
n-body:   7% |\\\                                               | ETA:  0:00:03
n-body:   8% |||||                                             | ETA:  0:00:03
n-body:   9% |////                                              | ETA:  0:00:03
n-body:  10% |-----                                             | ETA:  0:00:03
n-body:  12% |\\\\\                                             | ETA:  0:00:03
n-body:  13% ||||||||                                          | ETA:  0:00:03
n-body:  14% |//////                                            | ETA:  0:00:03
n-body:  15% |-------                                           | ETA:  0:00:03
n-body:  16% |\\\\\\\\                                          | ETA:  0:00:03
n-body:  17% |||||||||                                         | ETA:  0:00:03
n-body:  18% |/////////                                         | ETA:  0:00:03
n-body:  19% |----------                                        | ETA:  0:00:03
n-body:  20% |\\\\\\\\\\                                        | ETA:  0:00:03
n-body:  21% ||||||||||                                         | ETA:  0:00:03
n-body:  23% |//////////                                        | ETA:  0:00:03
n-body:  24% |-----------                                       | ETA:  0:00:02
n-body:  25% |\\\\\\\\\\\\                                      | ETA:  0:00:02
n-body:  26% ||||||||||||                                       | ETA:  0:00:02
n-body:  27% |////////////                                      | ETA:  0:00:02
n-body:  28% |------------                                      | ETA:  0:00:02
n-body:  29% |\\\\\\\\\\\\\\                                    | ETA:  0:00:02
n-body:  30% ||||||||||||||                                     | ETA:  0:00:02
n-body:  31% |//////////////                                    | ETA:  0:00:02
n-body:  32% |--------------                                    | ETA:  0:00:02
n-body:  33% |\\\\\\\\\\\\\\\\                                  | ETA:  0:00:02
n-body:  35% ||||||||||||||||                                   | ETA:  0:00:02
n-body:  36% |////////////////                                  | ETA:  0:00:02
```

```
n-body:  37% |------------------                                  | ETA:  0:00:02
n-body:  38% |\\\\\\\\\\\\\\\\\\                                  | ETA:  0:00:02
n-body:  39% ||||||||||||||||||||                                  | ETA:  0:00:02
n-body:  40% |//////////////////                                  | ETA:  0:00:02
n-body:  41% |--------------------                                | ETA:  0:00:02
n-body:  42% |\\\\\\\\\\\\\\\\\\\\                                | ETA:  0:00:02
n-body:  43% ||||||||||||||||||||||                                | ETA:  0:00:02
n-body:  44% |//////////////////////                              | ETA:  0:00:02
n-body:  46% |----------------------                              | ETA:  0:00:02
n-body:  47% |\\\\\\\\\\\\\\\\\\\\\\                              | ETA:  0:00:02
n-body:  48% ||||||||||||||||||||||||                              | ETA:  0:00:02
n-body:  49% |////////////////////////                            | ETA:  0:00:01
n-body:  50% |------------------------                            | ETA:  0:00:01
n-body:  51% |\\\\\\\\\\\\\\\\\\\\\\\\                            | ETA:  0:00:01
n-body:  52% ||||||||||||||||||||||||||                            | ETA:  0:00:01
n-body:  53% |//////////////////////////                          | ETA:  0:00:01
n-body:  54% |--------------------------                          | ETA:  0:00:01
n-body:  55% |\\\\\\\\\\\\\\\\\\\\\\\\\\                          | ETA:  0:00:01
n-body:  56% |||||||||||||||||||||||||||                          | ETA:  0:00:01
n-body:  58% |////////////////////////////                        | ETA:  0:00:01
n-body:  59% |----------------------------                        | ETA:  0:00:01
n-body:  60% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\                        | ETA:  0:00:01
n-body:  61% ||||||||||||||||||||||||||||                        | ETA:  0:00:01
n-body:  62% |//////////////////////////////                      | ETA:  0:00:01
n-body:  63% |------------------------------                      | ETA:  0:00:01
n-body:  64% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                      | ETA:  0:00:01
n-body:  65% ||||||||||||||||||||||||||||||                      | ETA:  0:00:01
n-body:  66% |////////////////////////////////                    | ETA:  0:00:01
n-body:  67% |--------------------------------                    | ETA:  0:00:01
n-body:  69% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                    | ETA:  0:00:01
n-body:  70% ||||||||||||||||||||||||||||||||                    | ETA:  0:00:01
n-body:  71% |//////////////////////////////////                  | ETA:  0:00:01
n-body:  72% |----------------------------------                  | ETA:  0:00:01
n-body:  73% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                  | ETA:  0:00:01
n-body:  74% ||||||||||||||||||||||||||||||||||                  | ETA:  0:00:01
n-body:  75% |////////////////////////////////////                | ETA:  0:00:00
n-body:  76% |------------------------------------                | ETA:  0:00:00
n-body:  77% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                | ETA:  0:00:00
n-body:  78% ||||||||||||||||||||||||||||||||||||                | ETA:  0:00:00
n-body:  80% |////////////////////////////////////                | ETA:  0:00:00
n-body:  81% |------------------------------------                | ETA:  0:00:00
n-body:  82% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\            | ETA:  0:00:00
n-body:  83% ||||||||||||||||||||||||||||||||||||||||            | ETA:  0:00:00
n-body:  84% |////////////////////////////////////////            | ETA:  0:00:00
n-body:  85% |----------------------------------------            | ETA:  0:00:00
```

```
n-body:  86% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\       | ETA:  0:00:00
n-body:  87% |||||||||||||||||||||||||||||||||||||||||||||       | ETA:  0:00:00
n-body:  88% |////////////////////////////////////////////      | ETA:  0:00:00
n-body:  89% |--------------------------------------------      | ETA:  0:00:00
n-body:  90% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\     | ETA:  0:00:00
n-body:  92% |||||||||||||||||||||||||||||||||||||||||||||||    | ETA:  0:00:00
n-body:  93% |//////////////////////////////////////////////    | ETA:  0:00:00
n-body:  94% |-----------------------------------------------   | ETA:  0:00:00
n-body:  95% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\   | ETA:  0:00:00
n-body:  96% |||||||||||||||||||||||||||||||||||||||||||||||||  | ETA:  0:00:00
n-body:  97% |////////////////////////////////////////////////  | ETA:  0:00:00
n-body:  98% |------------------------------------------------- | ETA:  0:00:00
n-body:  99% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\ | ETA:  0:00:00
n-body: 100% |||||||||||||||||||||||||||||||||||||||||||||||||||| Time: 0:00:04
```

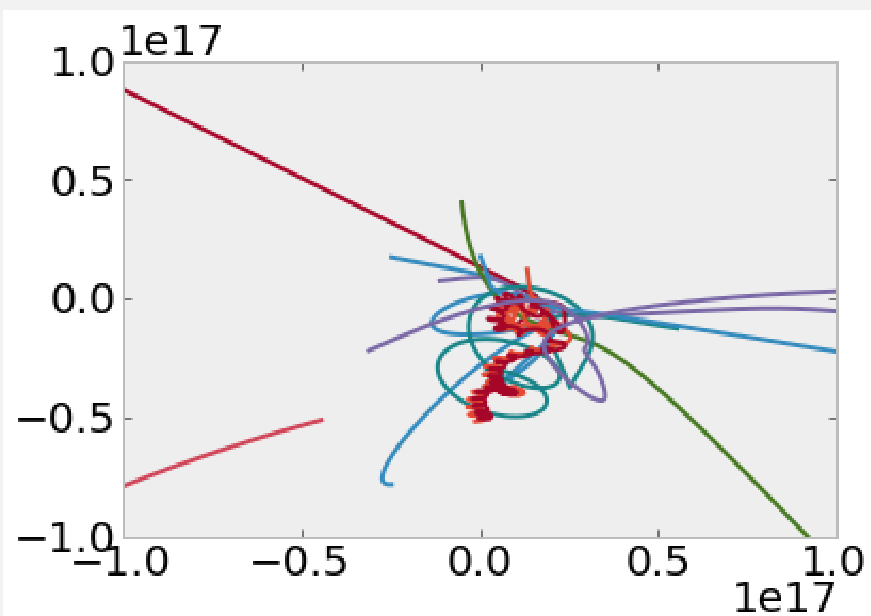
```
G.particle_history.shape
```

```
(731, 10, 6)
```

Plot the orbits of the particles. This can be as useful as animation.

```
clf()
for ii in xrange(star_masses.size/thin):
    plot(G.particle_history[:,ii,0],G.particle_history[:,ii,1])
xr = 10e16
axis([-xr,xr,-xr,xr])
```

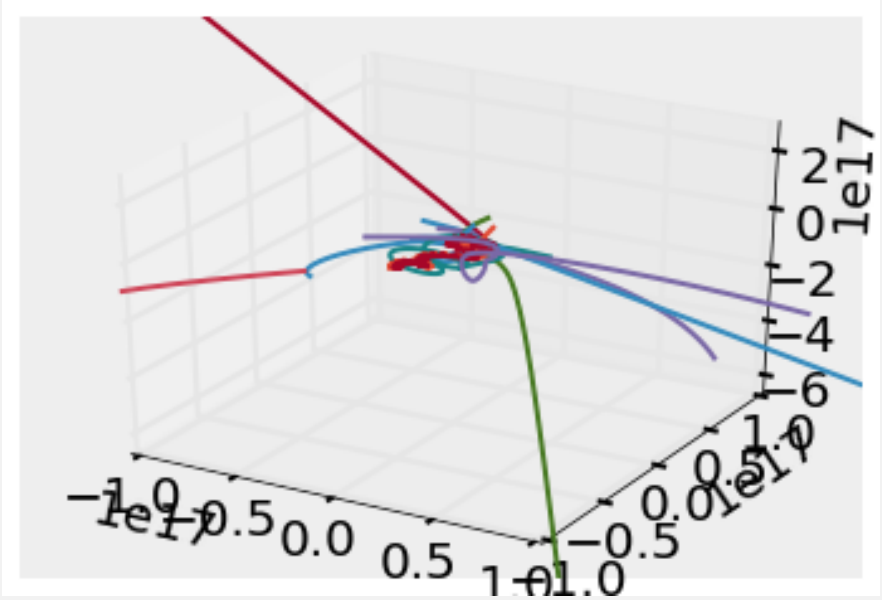
```
[-1e+17, 1e+17, -1e+17, 1e+17]
```



Python can do 3D plots. Sometimes they're cool, sometimes boring.

```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
clf()
ax = fig.add_subplot(111, projection='3d')
for ii in xrange(star_masses.size/thin):
    ax.plot(G.particle_history[:,ii,0],G.particle_history[:,ii,1],G.particle_history[:,ii
        ,2])
ax.axis([-xr,xr,-xr,xr])
```

```
[-1e+17, 1e+17, -1e+17, 1e+17]
```



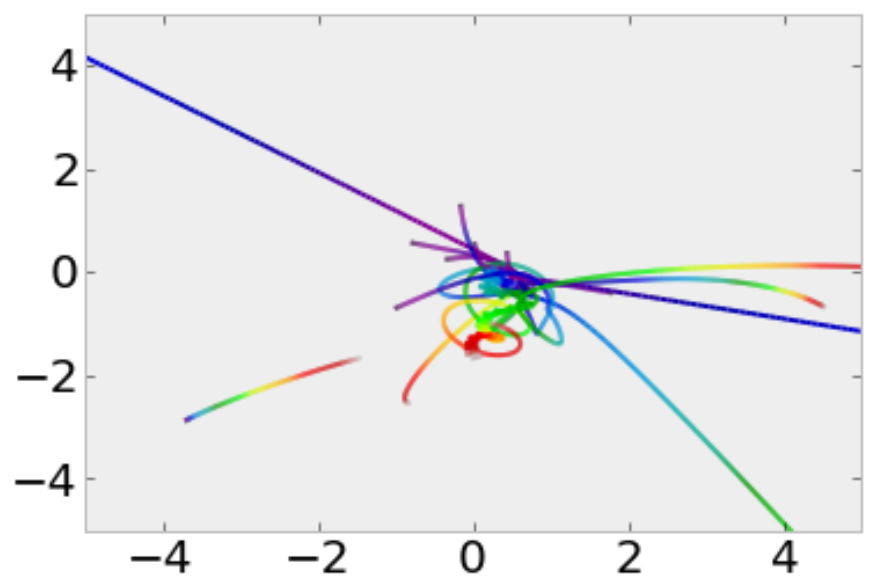

```
def plot_colorshift(x,y,z,cmap=matplotlib.cm.spectral, **kwargs):
    """ plot an x,y array colored by z """
    points = np.array([x.squeeze(), y.squeeze()]).T.reshape(-1, 1, 2)
    segments = np.concatenate([points[:-1], points[1:]], axis=1)
    lc = mpl.collections.LineCollection(segments, cmap=cmap, **kwargs)
    lc.set_array(z.squeeze())
    gca().add_collection(lc)
    return lc
```

In this plot, color corresponds to time, so you can see which stars fell in to the cluster and which ones ran away.

```
clf()
pc=3.08e16
for ii in xrange(nstars):
    lc = plot_colorshift(G.particle_history[:,ii,0]/pc, G.particle_history[:,ii,1]/pc,
        arange(G.particle_history.shape[0]),
        norm=mpl.colors.Normalize(vmin=0,vmax=G.particle_history.shape[0]-1))
zoomscale = 5
```

```
axis([-zoomscale,zoomscale]*2)
```

```
[-5, 5, -5, 5]
```



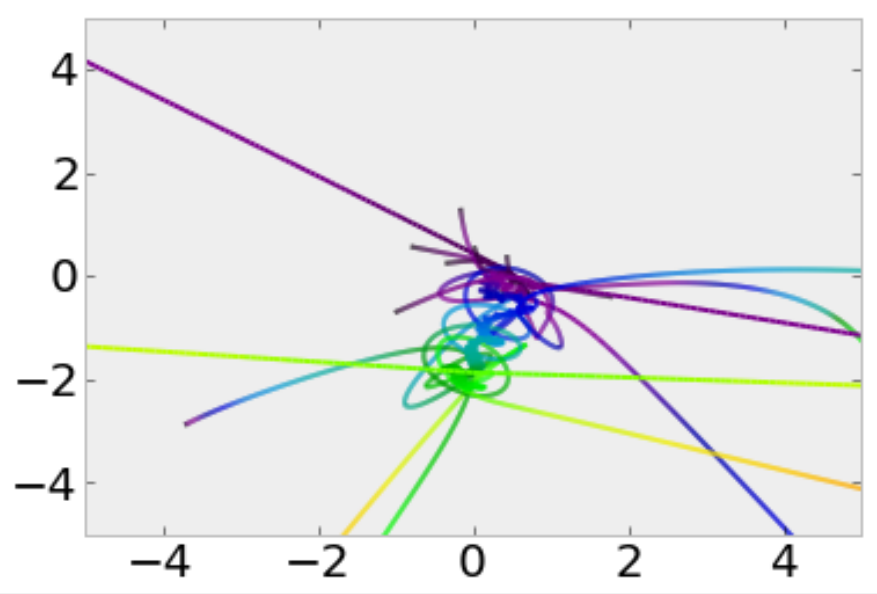
Here's another cool looking, longer-running simulation

```
G(5*365*86400,86400)
```

```
n-body:   0% |                                                  | ETA:  --:--:--
n-body:   1% |                                                  | ETA:  0:00:15
n-body:   2% |-                                                 | ETA:  0:00:13
n-body:   3% |\                                                 | ETA:  0:00:12
n-body:   4% |||                                                | ETA:  0:00:11
n-body:   5% |//                                                | ETA:  0:00:10
n-body:   6% |---                                               | ETA:  0:00:10
n-body:   7% |\\\                                               | ETA:  0:00:09
n-body:   8% |||||                                             | ETA:  0:00:09
n-body:   9% |////                                              | ETA:  0:00:09
n-body:  10% |-----                                             | ETA:  0:00:09
n-body:  11% |\\\\\                                             | ETA:  0:00:08
n-body:  12% ||||||||                                          | ETA:  0:00:08
n-body:  13% |//////                                            | ETA:  0:00:08
n-body:  14% |-------                                           | ETA:  0:00:08
n-body:  15% |\\\\\\\                                           | ETA:  0:00:08
n-body:  16% |||||||||                                         | ETA:  0:00:08
n-body:  17% |/////////                                         | ETA:  0:00:08
n-body:  18% |---------                                         | ETA:  0:00:07
n-body:  19% |\\\\\\\\\                                         | ETA:  0:00:07
```

```
n-body:  20% ||||||||||                                          | ETA:  0:00:07
n-body:  21% |//////////                                          | ETA:  0:00:07
n-body:  22% |----------                                          | ETA:  0:00:07
n-body:  23% |\\\\\\\\\\                                          | ETA:  0:00:07
n-body:  24% ||||||||||||                                        | ETA:  0:00:07
n-body:  26% |////////////                                        | ETA:  0:00:07
n-body:  27% |------------                                        | ETA:  0:00:06
n-body:  28% |\\\\\\\\\\\\                                        | ETA:  0:00:06
n-body:  29% ||||||||||||||                                      | ETA:  0:00:06
n-body:  30% |//////////////                                      | ETA:  0:00:06
n-body:  31% |--------------                                      | ETA:  0:00:06
n-body:  32% |\\\\\\\\\\\\\\                                      | ETA:  0:00:06
n-body:  33% ||||||||||||||||                                    | ETA:  0:00:06
n-body:  34% |////////////////                                    | ETA:  0:00:06
n-body:  35% |----------------                                    | ETA:  0:00:06
n-body:  36% |\\\\\\\\\\\\\\\\                                    | ETA:  0:00:05
n-body:  37% ||||||||||||||||||                                  | ETA:  0:00:05
n-body:  38% |//////////////////                                  | ETA:  0:00:05
n-body:  39% |------------------                                  | ETA:  0:00:05
n-body:  40% |\\\\\\\\\\\\\\\\\\                                  | ETA:  0:00:05
n-body:  41% ||||||||||||||||||||                                | ETA:  0:00:05
n-body:  42% |////////////////////                                | ETA:  0:00:05
n-body:  43% |--------------------                                | ETA:  0:00:05
n-body:  44% |\\\\\\\\\\\\\\\\\\\\                                | ETA:  0:00:05
n-body:  45% ||||||||||||||||||||||                              | ETA:  0:00:05
n-body:  46% |//////////////////////                              | ETA:  0:00:05
n-body:  47% |----------------------                              | ETA:  0:00:04
n-body:  48% |\\\\\\\\\\\\\\\\\\\\\\                              | ETA:  0:00:04
n-body:  49% ||||||||||||||||||||||||                            | ETA:  0:00:04
n-body:  51% |////////////////////////                            | ETA:  0:00:04
n-body:  52% |------------------------                            | ETA:  0:00:04
n-body:  53% |\\\\\\\\\\\\\\\\\\\\\\\\                            | ETA:  0:00:04
n-body:  54% ||||||||||||||||||||||||||                          | ETA:  0:00:04
n-body:  55% |//////////////////////////                          | ETA:  0:00:04
n-body:  56% |--------------------------                          | ETA:  0:00:04
n-body:  57% |\\\\\\\\\\\\\\\\\\\\\\\\\\                          | ETA:  0:00:03
n-body:  58% ||||||||||||||||||||||||||||                        | ETA:  0:00:03
n-body:  59% |////////////////////////////                        | ETA:  0:00:03
n-body:  60% |----------------------------                        | ETA:  0:00:03
n-body:  61% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\                        | ETA:  0:00:03
n-body:  62% |||||||||||||||||||||||||||||                      | ETA:  0:00:03
n-body:  63% |///////////////////////////////                     | ETA:  0:00:03
```

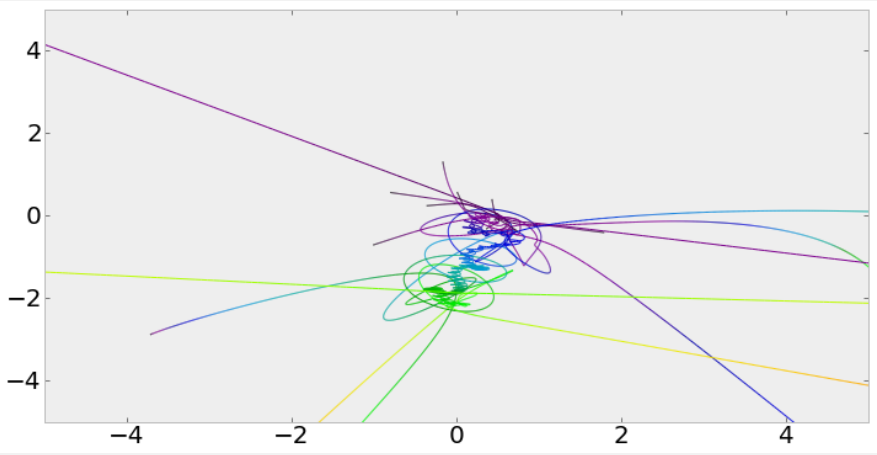
```
n-body:  64% |-------------------------------                  | ETA:  0:00:03
n-body:  65% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                 | ETA:  0:00:03

n-body:  66% ||||||||||||||||||||||||||||||||||                 | ETA:  0:00:03
n-body:  67% |/////////////////////////////////                | ETA:  0:00:03

n-body:  68% |---------------------------------                | ETA:  0:00:02
n-body:  69% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\               | ETA:  0:00:02

n-body:  70% ||||||||||||||||||||||||||||||||||||               | ETA:  0:00:02
n-body:  71% |///////////////////////////////////              | ETA:  0:00:02

n-body:  72% |-----------------------------------              | ETA:  0:00:02
n-body:  73% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\             | ETA:  0:00:02

n-body:  74% ||||||||||||||||||||||||||||||||||||||             | ETA:  0:00:02
n-body:  76% |/////////////////////////////////////            | ETA:  0:00:02

n-body:  77% |-------------------------------------            | ETA:  0:00:02
n-body:  78% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\           | ETA:  0:00:02

n-body:  79% ||||||||||||||||||||||||||||||||||||||||           | ETA:  0:00:01
n-body:  80% |///////////////////////////////////////          | ETA:  0:00:01

n-body:  81% |---------------------------------------          | ETA:  0:00:01
n-body:  82% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\         | ETA:  0:00:01

n-body:  83% ||||||||||||||||||||||||||||||||||||||||||         | ETA:  0:00:01
n-body:  84% |/////////////////////////////////////////        | ETA:  0:00:01

n-body:  85% |-----------------------------------------        | ETA:  0:00:01
n-body:  86% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\       | ETA:  0:00:01

n-body:  87% ||||||||||||||||||||||||||||||||||||||||||||       | ETA:  0:00:01
n-body:  88% |///////////////////////////////////////////      | ETA:  0:00:01

n-body:  89% |-------------------------------------------      | ETA:  0:00:00
n-body:  90% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\     | ETA:  0:00:00

n-body:  91% ||||||||||||||||||||||||||||||||||||||||||||||     | ETA:  0:00:00
n-body:  92% |/////////////////////////////////////////////    | ETA:  0:00:00

n-body:  93% |---------------------------------------------    | ETA:  0:00:00
n-body:  94% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\   | ETA:  0:00:00

n-body:  95% ||||||||||||||||||||||||||||||||||||||||||||||||   | ETA:  0:00:00
n-body:  96% |///////////////////////////////////////////////  | ETA:  0:00:00

n-body:  97% |-----------------------------------------------  | ETA:  0:00:00
n-body:  98% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\ | ETA:  0:00:00

n-body:  99% ||||||||||||||||||||||||||||||||||||||||||||||||| | ETA:  0:00:00
n-body: 100% ||||||||||||||||||||||||||||||||||||||||||||||||||| Time: 0:00:09

[-5, 5, -5, 5]
```



The same simulation, now zoomed out.

```
figure(figsize=[12,6]); clf()
pc=3.08e16
for ii in xrange(nstars):
    lc = plot_colorshift(G.particle_history[:,ii,0]/pc, G.particle_history[:,ii,1]/pc,
        arange(G.particle_history.shape[0]),
        norm=mpl.colors.Normalize(vmin=0,vmax=G.particle_history.shape[0]-1),
        linewidth=1)
zoomscale = 5
axis([-zoomscale,zoomscale]*2)
```

```
[-5, 5, -5, 5]
```




```
G(10*365*86400,86400)
```

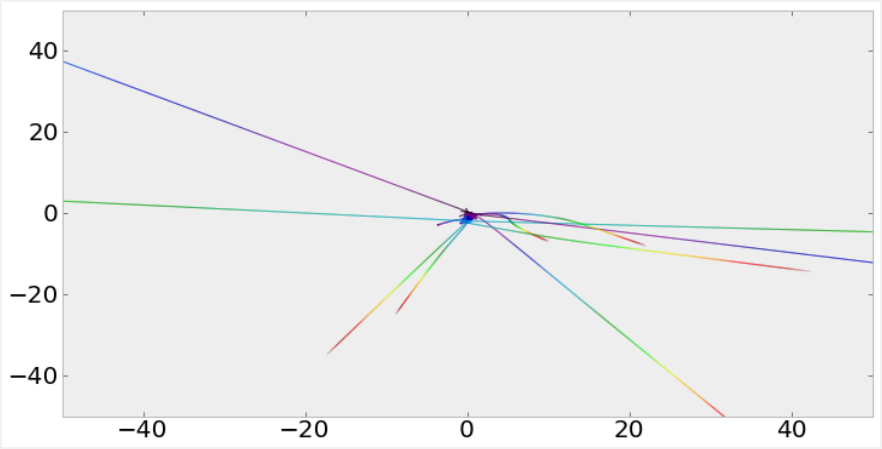
```
n-body:   0% |                                                      | ETA:  --:--:--
n-body:   1% |                                                       | ETA:  0:00:24
n-body:   2% |                                                       | ETA:  0:00:21
n-body:   3% |\                                                      | ETA:  0:00:20
n-body:   4% ||                                                       | ETA:  0:00:20
n-body:   5% |//                                                     | ETA:  0:00:20
n-body:   6% |--                                                     | ETA:  0:00:20
n-body:   7% |\\\                                                    | ETA:  0:00:19
n-body:   8% ||||                                                     | ETA:  0:00:19
n-body:   9% |////                                                   | ETA:  0:00:18
n-body:  10% |----                                                   | ETA:  0:00:18
n-body:  11% |\\\\\                                                  | ETA:  0:00:19
n-body:  12% ||||||                                                   | ETA:  0:00:18
n-body:  13% |//////                                                 | ETA:  0:00:18
n-body:  14% |------                                                 | ETA:  0:00:18
n-body:  15% |\\\\\\\                                                | ETA:  0:00:17
n-body:  16% ||||||||                                                 | ETA:  0:00:17
n-body:  17% |////////                                               | ETA:  0:00:17
n-body:  18% |--------                                               | ETA:  0:00:16
n-body:  19% |\\\\\\\\\                                              | ETA:  0:00:16
n-body:  20% |||||||||                                              | ETA:  0:00:16
n-body:  21% |/////////                                              | ETA:  0:00:16
n-body:  22% |----------                                             | ETA:  0:00:16
n-body:  23% |\\\\\\\\\\                                             | ETA:  0:00:15
n-body:  24% ||||||||||                                             | ETA:  0:00:15
n-body:  25% |//////////                                             | ETA:  0:00:15
n-body:  26% |----------                                             | ETA:  0:00:15
n-body:  27% |\\\\\\\\\\\                                            | ETA:  0:00:14
n-body:  28% |||||||||||                                            | ETA:  0:00:14
n-body:  29% |///////////                                            | ETA:  0:00:14
n-body:  30% |----------                                             | ETA:  0:00:14
n-body:  31% |\\\\\\\\\\\\                                           | ETA:  0:00:14
n-body:  32% ||||||||||||                                           | ETA:  0:00:13
n-body:  33% |////////////                                           | ETA:  0:00:13
n-body:  34% |----------                                             | ETA:  0:00:13
n-body:  35% |\\\\\\\\\\\\\                                          | ETA:  0:00:13
n-body:  36% |||||||||||||                                          | ETA:  0:00:12
n-body:  37% |/////////////                                          | ETA:  0:00:12
n-body:  38% |----------                                             | ETA:  0:00:12
n-body:  39% |\\\\\\\\\\\\\\                                         | ETA:  0:00:12
n-body:  40% ||||||||||||||                                         | ETA:  0:00:12
n-body:  41% |//////////////                                         | ETA:  0:00:11
```

```
n-body:  42% |--------------------                                  | ETA:  0:00:11
n-body:  43% |\\\\\\\\\\\\\\\\\\\\                                  | ETA:  0:00:11
n-body:  44% ||||||||||||||||||||||                                | ETA:  0:00:11
n-body:  45% |//////////////////////                                | ETA:  0:00:10
n-body:  46% |----------------------                                | ETA:  0:00:10
n-body:  47% |\\\\\\\\\\\\\\\\\\\\\\\\                              | ETA:  0:00:10
n-body:  48% ||||||||||||||||||||||||                              | ETA:  0:00:10
n-body:  49% |//////////////////////////                            | ETA:  0:00:10
n-body:  50% |------------------------                              | ETA:  0:00:09
n-body:  51% |\\\\\\\\\\\\\\\\\\\\\\\\\\                            | ETA:  0:00:09
n-body:  52% ||||||||||||||||||||||||||                            | ETA:  0:00:09
n-body:  53% |///////////////////////////                           | ETA:  0:00:09
n-body:  54% |--------------------------                            | ETA:  0:00:08
n-body:  55% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\                          | ETA:  0:00:08
n-body:  56% ||||||||||||||||||||||||||||                          | ETA:  0:00:08
n-body:  57% |////////////////////////////                          | ETA:  0:00:08
n-body:  58% |--------------------------                            | ETA:  0:00:08
n-body:  59% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                        | ETA:  0:00:08
n-body:  60% ||||||||||||||||||||||||||||||                        | ETA:  0:00:07
n-body:  61% |//////////////////////////////                        | ETA:  0:00:07
n-body:  62% |------------------------------                        | ETA:  0:00:07
n-body:  63% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                      | ETA:  0:00:07
n-body:  64% ||||||||||||||||||||||||||||||||                      | ETA:  0:00:07
n-body:  65% |////////////////////////////////                      | ETA:  0:00:06
n-body:  66% |--------------------------------                      | ETA:  0:00:06
n-body:  67% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                    | ETA:  0:00:06
n-body:  68% ||||||||||||||||||||||||||||||||||                    | ETA:  0:00:06
n-body:  69% |//////////////////////////////////                    | ETA:  0:00:06
n-body:  70% |----------------------------------                    | ETA:  0:00:05
n-body:  71% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\                  | ETA:  0:00:05
n-body:  72% ||||||||||||||||||||||||||||||||||||                  | ETA:  0:00:05
n-body:  74% |////////////////////////////////////                  | ETA:  0:00:05
n-body:  75% |------------------------------------                  | ETA:  0:00:05
n-body:  76% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\              | ETA:  0:00:04
n-body:  77% ||||||||||||||||||||||||||||||||||||||              | ETA:  0:00:04
n-body:  78% |//////////////////////////////////////              | ETA:  0:00:04
n-body:  79% |------------------------------------                  | ETA:  0:00:04
n-body:  80% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\          | ETA:  0:00:04
n-body:  81% ||||||||||||||||||||||||||||||||||||||||          | ETA:  0:00:03
n-body:  82% |////////////////////////////////////////          | ETA:  0:00:03
n-body:  83% |------------------------------------                  | ETA:  0:00:03
n-body:  84% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\  | ETA:  0:00:03
```

```
n-body:  85% |||||||||||||||||||||||||||||||||||||||||||||||||        | ETA:  0:00:02
n-body:  86% |/////////////////////////////////////////////        | ETA:  0:00:02
n-body:  87% |---------------------------------------------        | ETA:  0:00:02
n-body:  88% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\      | ETA:  0:00:02
n-body:  89% |||||||||||||||||||||||||||||||||||||||||||||||||||||||      | ETA:  0:00:02
n-body:  90% |/////////////////////////////////////////////////     | ETA:  0:00:01
n-body:  91% |-------------------------------------------------     | ETA:  0:00:01
n-body:  92% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\    | ETA:  0:00:01
n-body:  93% |||||||||||||||||||||||||||||||||||||||||||||||||||||||    | ETA:  0:00:01
n-body:  94% |/////////////////////////////////////////////////////   | ETA:  0:00:01
n-body:  95% |-----------------------------------------------------   | ETA:  0:00:00
n-body:  96% |\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\  | ETA:  0:00:00
n-body:  97% ||||||||||||||||||||||||||||||||||||||||||||||||||||||||  | ETA:  0:00:00
n-body:  98% |/////////////////////////////////////////////////////// | ETA:  0:00:00
n-body:  99% |-------------------------------------------------------- | ETA:  0:00:00
n-body: 100% |||||||||||||||||||||||||||||||||||||||||||||||||||||||||| Time: 0:00:20
```

```
figure(figsize=[12,6]); clf()
pc=3.08e16
for ii in xrange(nstars):
    lc = plot_colorshift(G.particle_history[:,ii,0]/pc, G.particle_history[:,ii,1]/pc,
        arange(G.particle_history.shape[0]),
        norm=mpl.colors.Normalize(vmin=0,vmax=G.particle_history.shape[0]-1),
        linewidth=1)
zoomscale = 50
axis([-zoomscale,zoomscale]*2)
```

```
[-50, 50, -50, 50]
```



Lab time. Tutorial will get you working on some of the functions we did in lecture, but using IDL.