PHYS 357 Pset 9. Due 11:59 PM Thursday Nov. 21

1. Townsend 6.4.

2. Balancing a pencil. What's the longest can you balance a pencil on its tip? The Heisenberg uncertainty principle means I can't start with both $x = 0$ and $p = 0$. I'm after a ballpark answer here, so you can e.g. assume all the mass in the pencil is at the end away from the tip. You can use 10g and 20cm for the pencil weight/length. NB - a pencil balanced on its tip looks very much like a pendulum, but with a sign change in the equation of motion. *The usual motion of a pendulum is theta(t)=sin(omega t) where omega=sqrt(L/g), but when we have the pendulum at the top omega becomes imaginary and we get theta(t)=c1 exp(omega t) +c2 exp(-omega t). The pencil has fallen over when theta=pi/2, which is close enough to 1 that we can use either. If the pencil starts at distance x0 from the top, then theta0=x0/L. If we start at theta0, at t0 we have theta0 = c1 exp(0) + c2 exp(0), so c1+c2=theta0. theta' = c1 omega exp(+)-c2 omega exp(-), so theta'=omega(c1-c2). Since we started at rest, this equals zero, and we get c1=theta0, c2=0. that leaves theta(t)=theta0 exp(omega t). When the pencil has fallen, we have 1=theta0 exp(omega t), omega t=-log(theta0), t=-log(theta0)/omega = -log(x0/l)/omega.*

   If we start with some inital velocity v0 but x=0, then we get c1+c2=0, omega(c1-c2)=v0/l, then c1=-c2=v0/2l omega. The time to fall becomes 1 = v0/2l omega exp(omega t), or omega t=log(v0/2l omega). v0=p0/m, so t=log(p0/2ml omega)/omega. However, we know that x0p0 is about hbar/2, so let's replace p0 with hbar/2x0. That leaves t=log(hbar/4mlx0 omega)/omega. You could do a fancy minimization, but a lazy thing here is to set the times for the two ways of falling equal. This works because if I squeeze down on x0, then p0 goes up, and that time to fall decreases and vice versa, so the longest overall time to fall is about when the two time are equal. That leaves us with log(x0/l)=log(hbar/4mlx0 omega) or x0=sqrt(hbar/4m omega). plug this in, and we get t=-log(sqrt(hbar/4m omega)/l)/omega. m=0.01,l=0.2, so omega=7, and plugging in the rest of the values gives about 5 seconds to fall.

3. Townsend 6.6

4. Townsend 6.12

5. Repeat Townsend 6.4, but this time do it on a computer. We can approximate a free particle by having a long stretch of space with zero potential. Find the eigenvalues/eigenvectors of this free space (this is fastest using scipy.linalg.eigh_tridiagonal, but you could certainly use numpy.linalg.eigh if you wanted), and describe a Gaussian well away from the boundary region as the sum of these eigenmodes. Make a movie showing the evolution of the Gaussian as it spreads out. Does your time for the width to double agree with your calculation from 6.4? Now make the same movie, but using a boxcar initial wave function ($\Psi(x) = 1$ for $0 < x < 1$, and zero otherwise).

6. Bonus - repeat the previous problem, but now use the Fourier transforms built into numpy (numpy.fft.rfft and irfft will be easiest to use). The discrete Fourier transform is defined to be

$$\sum_{0}^{N-1} f(x) \exp(-2\pi i k x/N)$$

for integer x, k, and $0 \leq x, k < N$. You'll need to take care with numerical factors and tune the spacings of your points, but you'll see that you can handle these free-space questions much, much faster with FFTs than with direct matrix inversions.