PHYS 3317 Fall 2012
Homework 3 Quantum Motion in Phase Space and Electrostatic Potential from Electron Cloud in Atom

Some additional references: Tannor Chapter 5 deals with the Wigner distribution function. Atomic spectra basics are covered in many books and you should have seen it before. One topic to pay attention to here is getting familiar with the spectroscopic notation, e.g. $n\,^{2S+1}L_J$ with n, S, L, and J being principal, spin, orbital, and total angular momentum quantum numbers. L is replaced with letters SPDFGH... for $L = 0, 1, \ldots$. You can use some mnemonic to remember the sequence – Wikipedia provides a funny one – "**S**ober **P**hysicists **D**on't **F**ind **G**iraffes **H**iding **I**n **K**itchens **L**ike **M**ine".

1. Marginals of Wigner distribution and its time evolution

There are many texts dealing with Wigner Distribution Function (WDF) and its use for signal processing, optics, or quantum mechanics (where it all began!). For a succinct review of WDF and its properties in the most relevant to us context refer to Phys. Rev. ST AB 15, 050703 (2012), Section II. You should be able to follow most of it except perhaps when it comes to mixed quantum states (subsection II.B). Prove the following properties of the Wigner distribution.

(a) Property 2 in the paper about marginals, namely:

$$\int_{-\infty}^{+\infty} W(x,p)dp = |\psi(x)|^2.$$

Follow the hint provided at the bottom of the page after this property. The other expressions given in this property are proven in much the same way.

(b) Property 5 on time evolution of the WDF, $W = W(x,p;t)$, or the following expression:

$$\frac{\partial W}{\partial t} + \frac{p}{m}\frac{\partial W}{\partial x} + \frac{i}{\hbar}\left[V\left(x + \frac{i\hbar}{2}\frac{\partial}{\partial p}\right) - V\left(x - \frac{i\hbar}{2}\frac{\partial}{\partial p}\right)\right]W = 0.$$

As stated in the paper, the proof is by using time-dependent Schrödinger equation as done in Ref. 16. Go to this reference, Sov. Phys. Usp. 26, 311 (1983), Part 4(a) and follow the derivation steps he provides filling in explicitly all the details. Your destination is formula (4.4a). This paper has more interesting things to say about the Wigner distribution (not all of his *subjective* statements one would necessarily have to agree with), and is worth reading if you are interested in the subject.

2. Quantum tunneling movie in phase space

We can talk all we want about wonderful properties of the Wigner distribution (and they are both *many* and *wonderful*!). But, as the saying goes "better to see once than to hear 100 times". In this exercise you will produce a movie of a Gaussian wave packet tunneling through a potential barrier. In fact, you will use the exact same state and the potential that you already "animated" in the previous homework.

I did the more difficult part and provided a MATLAB function that computes the Wigner distribution from a wavefunction of position. Download this function (`wig2.m`) from *Blackboard→Simulations*. It takes as input arguments a vector of positions in nm and the wavefunction and returns as output arguments the WDF and the corresponding momentum array (the units of $\times 10^{-31}$ kg nm/fs, so that if you divide the momentum by `emass` as found in the original `schrod.m` you will obtain velocity in nm/fs). Now bring the quantum motion to life by doing the following.

(a) Plot both the tunneling and free propagating versions of the initial wave packet. Use `subplot()` command to create two subplots. When it comes to displaying the Wigner function, use `imagesc()` command, followed by `axis xy;` (the first command plots color-scaled array, and the second one prevents MATLAB from flipping y-axis in a manner common when displaying images; refer to the documentation for specifics). Before plotting the Wigner distribution, check that $\iint W(x,p)dxdp = 1$, which in the case of finite sampling reduces to `sum(sum(W*dx*dp))`, where `dx` and `dp` are step sizes in position and momentum respectively. (Hint: a useful trick when you need to sum multiple dimensions of a higher dimensional array without invoking multiple sums is to write `sum(W(:))`. `W(:)` squashes the array into one long vector so that only one sum is required to add all the elements.)

(b) MATLAB has many color maps, which can be applied to an image (or a real valued 2D array) using `colormap()` command. However, none of the color maps provided are well suited for the dual-signed functions such as the WDF. Download the function I provide at *Blackboard→Simulations* called `bluewhitered.m` (which itself was provided by one of MATLAB users at Mathworks file exchange server). This color map changes from blue-to-white-to-red for negative-to-zero-to-positive function values. If you plot $\pi\hbar W(x,p)$, we expect this quantity to be bound between -1 and 1. To specify color axis ranges and the specific color map use the following commands somewhere after `imagesc()`: `caxis([-1 1]); colormap(bluewhitered(256));`

(c) We know that marginals or projections of the Wigner distribution are very useful being probability densities to find the particle with a given position or momentum. Add $\int W(x,p)dp = |\psi(x)|^2$ to the bottom and $\int W(x,p)dx = |\tilde{\psi}(p)|^2$ to the left of each subplot. If need be scale the projections by multiplying

them by some constant so that they are clearly visible on each plot (see Fig. 1).

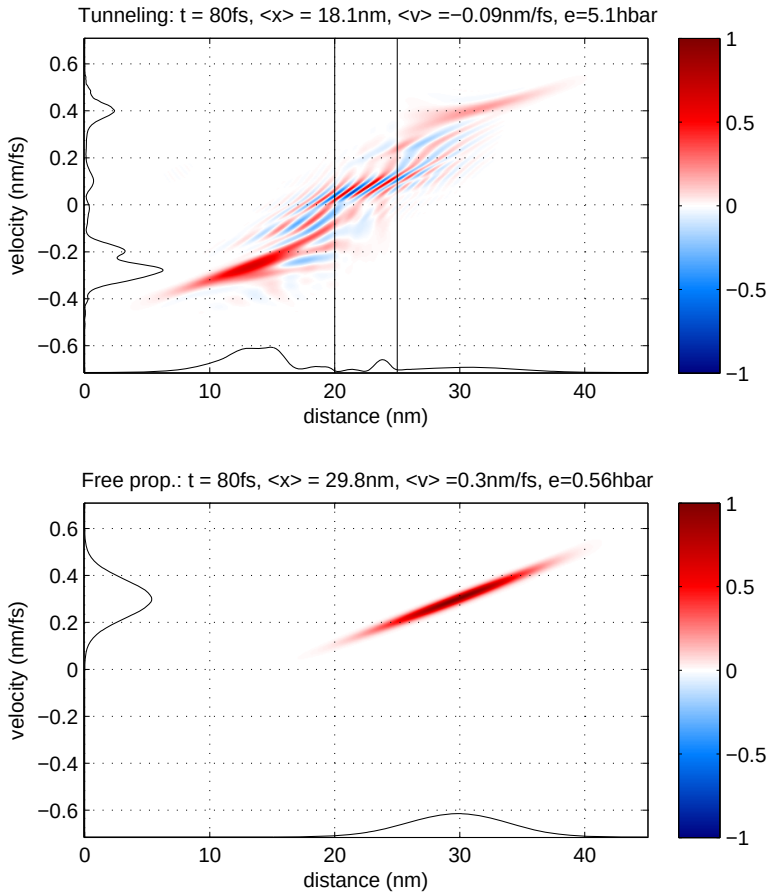

Figure 1: Final Wigner distribution, $\pi\hbar W$, at $t = 80\,\text{fs}$ for tunneling (top) and free propagating (bottom) wave packets along with its projections. The location of the barrier is also shown.

(d) Now provide some additional information about the Wigner distribution plots (using `title()` command) found inside your `for` loop as you are changing the time from 0 to 80 fs. Using Wigner-Weyl representation of momentum and position operators, compute $\langle x \rangle$ and $\langle v \rangle = \langle p \rangle / m_e$. Your Wigner-Weyl representations of momentum and position operators must have the same size as the Wigner distribution array. Use MATLAB's command `[pp,xx] = ndgrid(p,x);` which will return arrays `pp` and `xx` out of vectors `p` and `x` representing grids of momentum and position respectively. These quantities `pp` and `xx` are the Wigner-Weyl functions (arrays) that you need.

(e) One final quantity to compute and display on the plots is the generalized uncertainty in position and momentum (known as *emittance* in beam physics):

$$\epsilon = \sqrt{\langle x^2 \rangle \langle p^2 \rangle - \langle xp \rangle^2}.$$

This quantity should remain invariant when the wave packet is free propagating or subject to linear forces. Display this quantity in units of $\hbar$ (again, it cannot be smaller than 0.5).

Your "fully dressed" movie frames should look something like Fig. 1.

There are several interesting things to note. First that the Gaussian wave packet has WDF which is everywhere positive (and that's the *only* wave packet which has this property). Second, notice that the regions with negative WDF emerge when the wave packet begins to interact with the barrier. These negative regions are responsible for the "fringes" in the probability density (or the projections). Third, the phase space picture and its projections clearly show e.g. that the transmitted wave packet has a higher momentum due to the barrier acting as a filter while the reflected wave packet contains the lower energy part of the initial electron.

You should also observe that the free propagating wave packet becomes "squeezed" as it propagates with time while conserving the generalized uncertainty ϵ (a small deviation that you will encounter at the end is due to the fact that the potential box has impenetrable boundaries reflecting back small parts of the wave packet). If you increase time well beyond 80 fs you will see a complicated phase space pattern due to the infinite force from the boundary "breaking" or reflecting back the wave packet. Give it a try!

3. Electrostatic potential of electron cloud

This problem is meant to explain a formula that you'll be using in your next homework assignment where we will compute energy levels of multi-electron atoms (a very real and non-trivial application of quantum mechanics). An electron in atom "feels" not only its positively charged nucleus but also the repulsive potential due to the "cloud" from all the other electrons. If we adopt the approximation of the electron cloud being spherically symmetrically distributed, we can efficiently evaluate the electrostatic potential due to this distribution. And recall that in order to solve Schrödinger equation we need to know the potential at *all points* along the radial coordinate.

The electrostatic potential due to a spherically symmetric charge distribution $\rho(r)$ is:

$$\phi(r) = \frac{1}{4\pi\epsilon_0} \int d^3\mathbf{r}' \frac{\rho(r')}{|\mathbf{r} - \mathbf{r}'|}.$$

This is a 3D integral, which is numerically quite expensive. Your job is to prove the

following expression, which reduces this 3D integral to the sum of two 1D integrals:

$$\epsilon_0 \cdot \phi(r) = \frac{1}{r} \int_0^r r'^2 \rho(r') dr' + \int_r^\infty r' \rho(r') dr'.$$

(Hint: one approach is to compute the electric field at each r using Gauss' law. Writing $E(r) = Q(r)/4\pi\epsilon_0 r^2$, and expressing $Q(r)$ in terms of ρ, the potential can be obtained by writing down the integral $\int E(r)dr$ and by integrating by parts.)

An additional remark: when it comes to evaluating integrals such as $\int_0^r r'^2 \rho(r') dr'$ with the 'running' upper integration limit (and remember we will require the knowledge of this integral at *many* values of r), MATLAB provides a useful function called `cumsum()` or cumulative sum. E.g. `cumsum([1 2 3 4 5])` returns `1 3 6 10 15` by successively summing up the vector elements. This can be used to numerically compute the integral above with the running upper integration limit and obtain the integral's multiple values in a single line of code. Furthermore, if we rewrite the expression above as

$$\epsilon_0 \cdot \phi(r) = \frac{1}{r} \int_0^r r'^2 \rho(r') dr' - \int_0^r r' \rho(r') dr' + \int_0^\infty r' \rho(r') dr',$$

we can use `cumsum()` for the first two terms, and the normal `sum()` for the 3rd term. No `for` loops are needed, we can build the entire array of ϕ vs. r with a single line of code.