Physics 465: Computing Project 4

September 7, 2011

Investigating the properties of coherent states of the Simple Harmonic Oscillator (SHO).

This week's project is to investigate the behavior of special superpositions of energy eigenstates of the simple harmonic oscillator called *coherent states*. The setup will be similar to last time, except that the eigenstates in the superposition are SHO eigenstates, rather then infinite square well eigenstates.

What is a *Coherent State*?

A *coherent* state is a superposition of energy eigenstates that behaves (as nearly as quantum mechanically possible) like an ideal classical system. If you check Griffiths derivation of a "minimum uncertaintly wave packet" (Eq. 3.67, which we'll get to in detail in chapter 3) you can rearrange it to look like this (where Griffiths' parameter a is replaced by $m\omega$):

$$(i\hat{p} + m\omega x)\psi = (i\langle p\rangle + m\omega\langle x\rangle)\psi \tag{1}$$

Which, aside from the overall factor $A = 1/\sqrt{2m\hbar\omega}$, is the equation to find eigenstates of the lowering operator $\hat{a}_-$! (note that $\langle x\rangle$ and $\langle p\rangle$ are just the expectation values of x and p, so the factor in parentheses on the right is just a scaler, the eigenvalue.) In other words... packets of minimum uncertainty can be found by looking for eigenstates of the lowering operator $\hat{a}_-$. So.. let's do it. Assume that ψ_α is an eigenstate of the lowering operator $\hat{a}_-$ with eigenvalue α:

$$\hat{a}_-\psi_\alpha = \alpha\psi_\alpha \tag{2}$$

but also assume that it's a superposition of energy eigenstates like so:

$$\psi_\alpha = \sum_{n=0}^{\infty} c_n \psi_n \tag{3}$$

If we substitute Eq. 3 into Eq. 2 we get:

$$\begin{aligned}
\hat{a}_- \psi_\alpha &= \hat{a}_- \textstyle\sum_{n=0}^{\infty} c_n \psi_n && (4)\\
\alpha \psi_\alpha &= \textstyle\sum_{n=0}^{\infty} c_n \sqrt{n} \psi_{n-1} && (5)\\
\alpha \sum_{n=0}^{\infty} c_n \psi_n &= \textstyle\sum_{n=0}^{\infty} c_n \sqrt{n} \psi_{n-1} && (6)\\
\alpha \left(c_0 \psi_0 + c_1 \psi_1 + c_2 \psi_2 + \cdots \right) &= c_1 \sqrt{1} \psi_0 + c_2 \sqrt{2} \psi_1 + c_3 \sqrt{3} \psi_3 + \cdots && (7)
\end{aligned}$$

which, remembering that the ψ_n are all orthogonal means that:

$$\begin{aligned}
c_1 &= \alpha c_0 / \sqrt{1} && (8)\\
c_2 &= \alpha c_1 / \sqrt{2} = \alpha^2 c_0 / \sqrt{2 \cdot 1} && (9)\\
c_3 &= \alpha c_2 / \sqrt{3} = \alpha^3 c_0 / \sqrt{3 \cdot 2 \cdot 1} && (10)\\
\vdots &= \vdots && (11)\\
c_n &= \alpha c_{n-1} / \sqrt{n-1} = \alpha^n c_0 / \sqrt{n!} && (12)
\end{aligned}$$

So.. the result is

$$c_n = \alpha^n c_0 / \sqrt{n!} \tag{13}$$

we can put that back into Eq. 3 to get:

$$\psi_\alpha = c_0 \sum_{n=0}^{\infty} \frac{\alpha^n}{\sqrt{n!}} \psi_n \tag{14}$$

This is called a *coherent state*. In order for it to be properly normalized c_0 must be $e^{-\alpha^2/2}$. After it's normalized it's easy to show that the expectation value of n is $|\alpha|^2$, and that the probability distribution of n is a poisson distribution.

So... what are we supposed to do?

Your mission is to produce two deliverables:

1) A 3D representation of the complex wavefunction over a time interval corresponding to two full cycles of the coherent state. At the beginning your 3D representation should look something like Fig. 1.

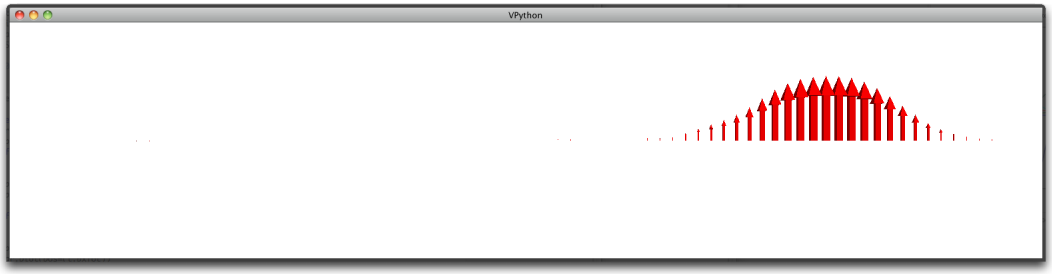

Figure 1: The initial wavefunction, 80 arrows, a=15.0

Later on your display should look something like this Fig. 2.

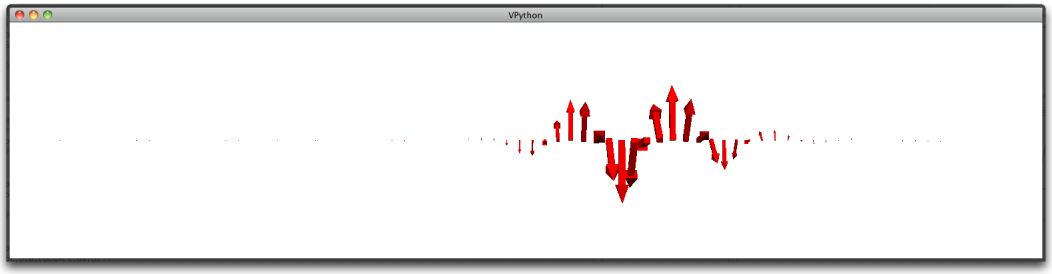

Figure 2: The wavefunction later, 80 arrows, a=15.0

Please include at least one screen capture of your wavefunction and include it in your report.

2) A graph of the expectation value of x as a function of time. With my setup I get a graph that looks like Fig. 3.

(Note: If you set ω=1, then your time units will essentially be equal to the phase of corresponding classical harmonic oscillator. You can always choose other time units, but this will be simplest.)

The questions that need to be answered as part of the report require you to reflect on this graph, so it's important to get it right. If you graph doesn't look pretty close to mine, please ask!

What follows is some sample code from the beginning of my solution program. It starts out a lot like computing project 3, except that the stationary states are not sin functions

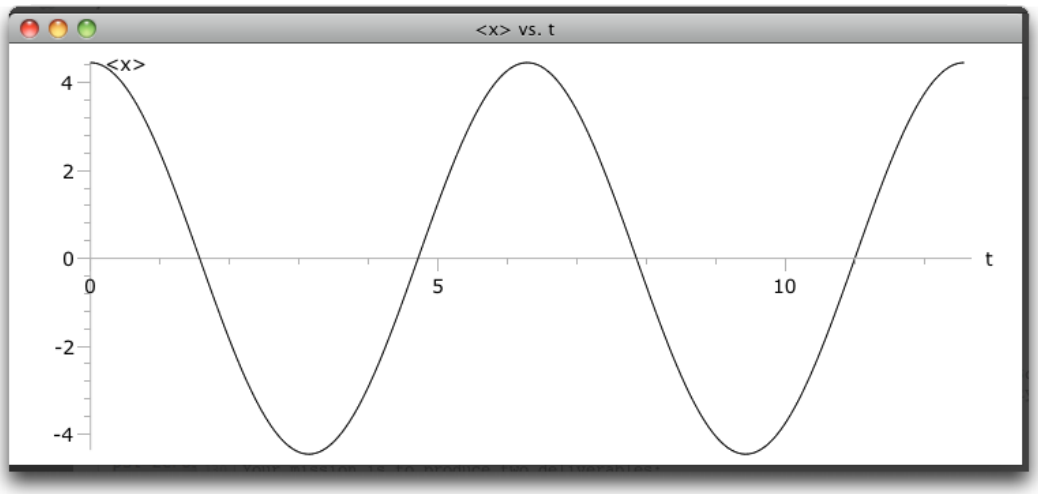

Figure 3: The value of $\langle x \rangle$ (as a function of time)

but are rather eigenstates of the SHO hamiltonian. As we're learning in section 2.3.2, those solutions can be expressed in terms of the scaled distance $\xi = x/\sqrt{\hbar/m\omega}$ as:

$$\psi_n(\xi) = \left(\frac{1}{\pi}\right)^{\frac{1}{4}} \frac{1}{\sqrt{2^n n!}} H_n(\xi) e^{-\xi^2/2} \tag{15}$$

I've set my units up so that x is measured in scaled units (i.e., $\xi = x$) and time is measured in units that make $\omega = 1$. You can start your program with my code, if you like, to save time.

```
from visual import *
from visual.graph import *
from safcn import SetArrowFromCN  # defined in a file safcn.py

gd = gdisplay(title="<x> vs. t", xtitle="t", ytitle="<x>",
                foreground=color.black, background=color.white)
gr = gcurve(color=color.black)
scene.background=color.white

NA=80                           # how many arrows?
a=15.0                          # range of x is -a/2 to a/2 in units
                                # of $\sqrt{\hbar/m\omega}$
x = linspace(-a/2, a/2, NA)     # NA locations from -a/2 to a/2

NHs=20
hs=zeros((NHs,NA),double)    # the hermite polynomials, an NHs x NA array
```

```
coefs=zeros(NHs,double)      # the coherent state coefficients, an NHs x 1 array
psis=zeros((NHs,NA), double) # the stationary states, an NHs x NA array

alpha=sqrt(10)               # <n> = 10.0

hs[0]=0*x + 1.0              # zeroth Hermite Polynomial, H0
hs[1]=2*x                    # first Hermite Polynomial, H1

#
# Compute the first NHs Hermite Polynomials,
# use recurrence relation to get the rest of the Hn(x)
#
# (see e.g., http://en.wikipedia.org/wiki/Hermite_polynomials#Recursion_relation)
#

for n in range(1,NHs-1):
    hs[n+1]=2*x*hs[n] - 2*n*hs[n-1]

#
# Use the coherent state coefficient relation to get the c[n]s.
# avoid overflow by computing them in a loop. Don't worry about
# the overall factor of c[0] since we'll renormalize our discrete
# psi array later anyway.
#

coefs[0]=1.0
for i in range(1,NHs):
    coefs[i]=coefs[i-1]*alpha/sqrt(i)

#
# Get the stationary states using the hs array and compute the
# normalization factor in a loop to avoid overflow
#

normFactor = 1.0/pi**0.25
psis[0]=exp(-x**2/2)
for i in range(1,NHs):
    normFactor = normFactor/sqrt(2.0*i)
    psis[i]=normFactor*hs[i]*exp(-x**2/2)

#
# Now do the sum to compute the initial wavefunction
#
```

```
psi=zeros(len(x),complex)
for m in range(NHs):
    psi += coefs[m]*psis[m]

#
# Normalize!
#

psi=psi/sqrt((abs(psi)**2).sum())

#
# build the arrows. Scale them on the screen by a factor
# of 3 so they look nice.
#

alist = []
for i in range(NA):
    alist.append(arrow(pos=(x[i],0,0), color=color.red))
    SetArrowFromCN(3*psi[i],alist[i])

t = 0.0
dt = 2*pi/1000.0

#
# After this.. create the "time loop" that animates the wavefunction
# and computes the values for <x> and graphs them.
#
```

Questions

Please answer these questions at the end of your report.

1) Normalize Eq. 14 to show that c_0 must be $e^{-\alpha^2/2}$.

2) Use Eq. 14 to show that the expectation value of n is $|\alpha|^2$.

3) Find an approximate relationship between the value of the parameter α and the amplitude of the variation in $\langle x \rangle$ over time. Run the program with several values of α to check that your relationship is at least approximately correct.