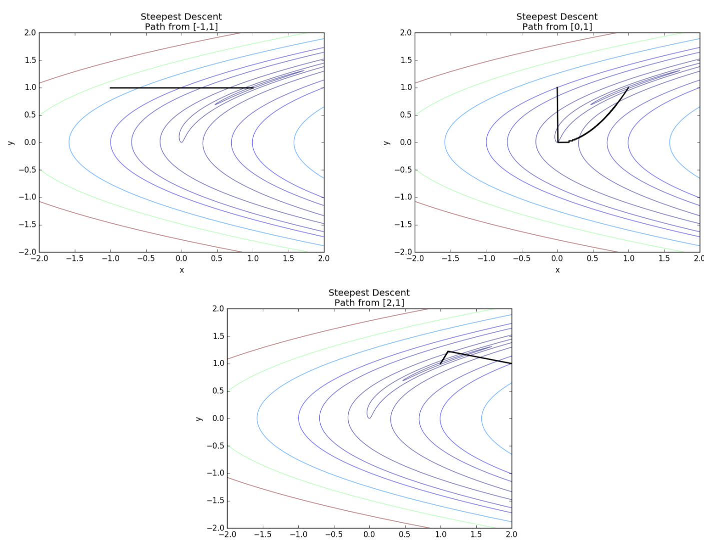
APMTH 205 HW 5
Louis Baum
December 1, 2016

Problem 1

a)

For this section I looked at minimizing Rosenbrock's function using steepest descent. I found that the biggest issue I ran into was that my resulting path and thus final answer for a given number of iterations was highly dependant on the initial guess for the minimum of the line search algorithm. I ended up going with a 'guess' based on the length of the step I thought it should make. To do this I normalized the Gradient and set my step to be equal to 1/(iteration number) this gave me a 'guess' for the η. This means that the further along we are in the iterative process the shorter we expect the step size to be. While qualitatively this seemed to perform better (ie a path that doesn't double back on itself of miss obveous minima) I am sure it is incredibly situational.

Here are the plots:

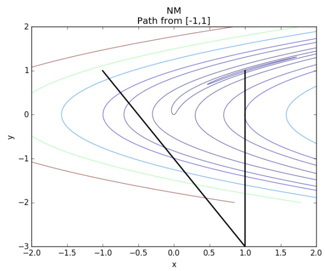


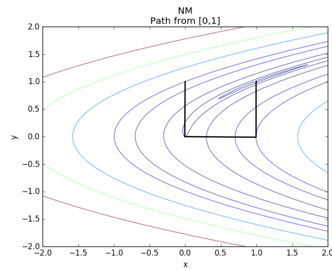
Iterations:

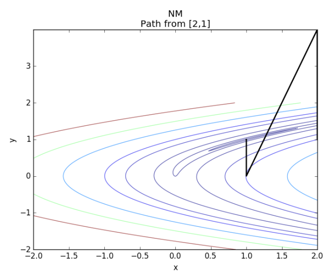
$[-1, 1] = 2$ $\quad$ $[0, 1] = 1729$ $\quad$ $[2, 1]$= 854

b)

This was substantially easier. Here are the plots:







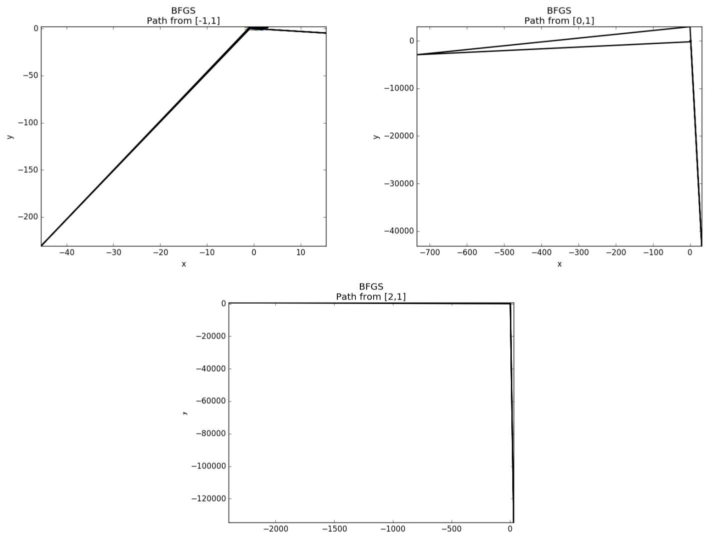
Iterations:

$[-1, 1] = 3 \qquad [0, 1] = 6 \qquad [2, 1] = 6$

c)

The paths that the minimization algorithm take are extremely erratic for the BGFS method. I only plot the contours of the Rosenbrock function over (-2,2) in both x and y. The paths for this method extend far ourside of that range.

Here are the figures. please note the axis.



Iterations:

$[-1,1] = 123$ $\quad$ $[0,1] = 38$ $\quad$ $[2,1]$= 45

Overall the number of iteration for each method makes sense. We would expect Newtons method to perform the best followed by BFGS followed by Steepest Descent.

Problem 2

a)

$$r(b) = V - T$$

$$V = \int_0^R \mu\Big(\sqrt{(\frac{dx}{ds})^2 + (\frac{dy}{ds})^2} - 1\Big)^2 ds \qquad T = \int_0^R \rho y^2 w^2 ds$$

$$r(b) = \int_0^R \Big(\mu\Big(\sqrt{(\frac{dx}{ds})^2 + (\frac{dy}{ds})^2} - 1\Big)^2 - \rho y^2 w^2\Big) ds$$

To take the gradient we are taking the derivative with respect to c_k and d_k to a total of 40 terms.
since c_k and d_k are coefficients independent of s we can move the derivative inside of the integral.
We then have

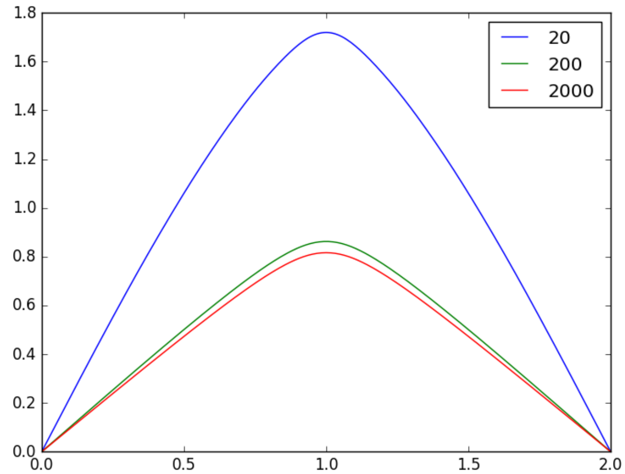
$$\frac{\partial r}{\partial c_k} = \int_0^R 2\frac{dx}{ds}\frac{k\pi Cos[\frac{k\pi s}{R}]}{R}\frac{\mu\Big(\sqrt{(\frac{dx}{ds})^2 + (\frac{dy}{ds})^2} - 1\Big)}{\sqrt{(\frac{dx}{ds})^2 + (\frac{dy}{ds})^2}} ds$$

$$\frac{\partial r}{\partial d_k} = \int_0^R 2\frac{dy}{ds}\frac{k\pi Cos[\frac{k\pi s}{R}]}{R}\frac{\mu\Big(\sqrt{(\frac{dx}{ds})^2 + (\frac{dy}{ds})^2} - 1\Big)}{\sqrt{(\frac{dx}{ds})^2 + (\frac{dy}{ds})^2}} - 2w^2 Sin[\frac{k\pi s}{R}] y ds$$

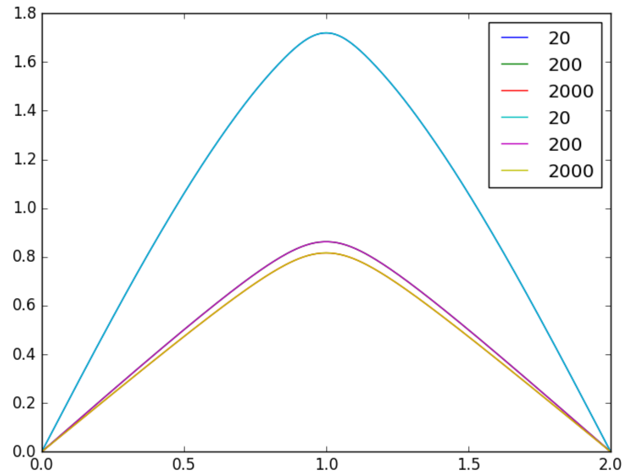
b)

Initially I only wrote a script that would return the value of r. I combined that with *scipy.optimize.minimize* and that allowed me to find the three curves that corresponded to minimizing the action for the three values of μ. However this lead to problems when attempting to find the "second" mode in part c.

I then implemented my gradient function gradr(b) which allowed me to find the zeros corresponding not only to mininima but also critical(ish) points. I coupled this with $scipy.optimize.fsolve$. This yielded the form shown below. I was able to determine that my gradient function was in agreement with my r function by employing the *scipy.optimize.check_grad*



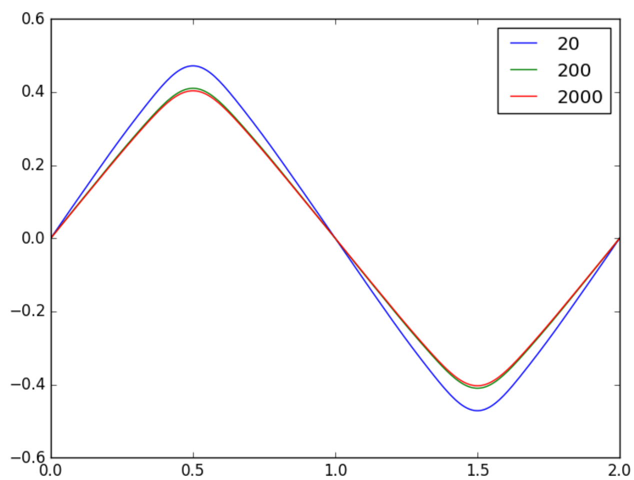
Jump rope with d1=1 as my initial condition
legend indicates value of μ



Comparison of results from scipy.optimize.minimize and ".".fsolve show excellent agreement

c)

I then used the initial condition d2 = .5 and was able to see the "second" mode of the jump rope. I have seen this before when using a very long rope and swinging it quickly. In this mode there are two anti-nodes where you can have potential jumpers.
It is also possible to get even higher order modes.



Second Mode jump rope

Problem 3

a)

We first plug in the second order finite differences approximation for the second derivative.

$$-\frac{\partial^2 \Psi}{dx^2} + v(x)\Psi = E\Psi$$

$$\frac{-\Psi(x+h) + 2\Psi(x) - \Psi(x-h)}{h^2} + v(x)\Psi(x) = E\Psi(x)$$

We can put this in terms of a matrix equation as follows:

$$M\vec{\Psi} = IE_\lambda\vec{\Psi}$$

where $\vec{\Psi}$ is a vector with the value of Psi at each grid point (including ghost nodes), M the Hamiltonian Matrix and E is a particular eigenenergy.
We can then construct the bulk of matrix M with the values of v(x) and h.
We must now consider boundary conditions - Ψ(x) = 0 at x= -12 and 12. To accomplish this I used two ghost nodes at -12-h and 12+h.
We solve

$$\frac{-\Psi(x+h) + 2\Psi(x) - \Psi(x-h)}{h^2} + v(x)\Psi(x) = 0$$

$$\Psi(x) = (\frac{2}{h^2} + v(x)) * \frac{\Psi(x+h) + \Psi(x-h)}{h^2}$$

setting $\Psi(x) = 0$

$$\Psi(x+h) = -\Psi(x-h)$$

at the ghost nodes.
We plug this into our expression and get

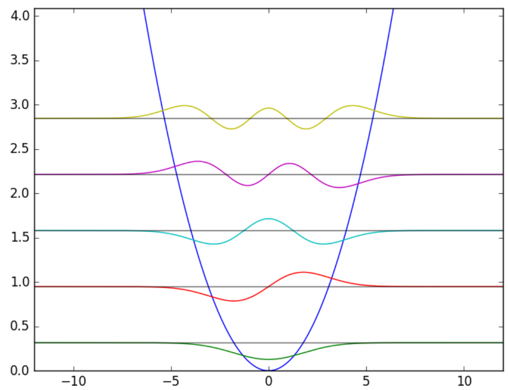
$$\frac{2\Psi(x)}{h^2} + v(x)\Psi(x) = E\Psi(x)$$

to use to construct the first and last row of M.
We then use *numpy.linalg.eig*() to obtain the eigenenergies and eigenfunctions

I confirm that my program works by duplicating the results for $\frac{x^2}{10}$

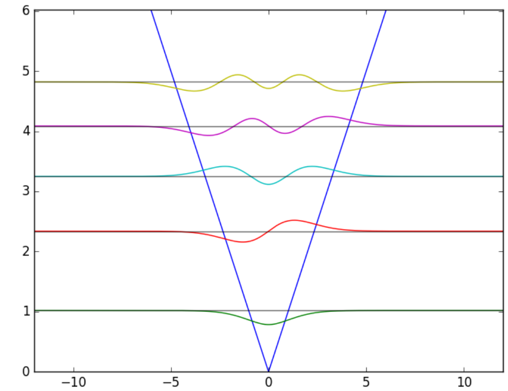
Please note that for these plots the amplitude of the wavefunctions has been exaggerated for clarity.

Test Potential [v(x) $= \frac{x^2}{10}$



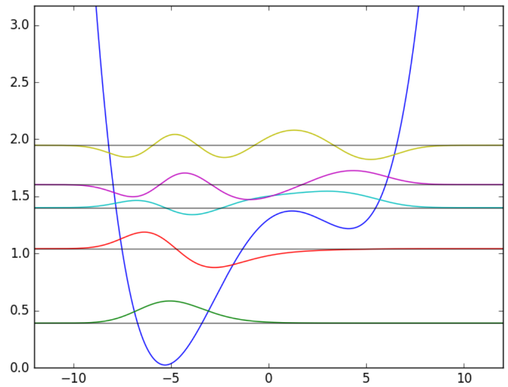
E_5	2.84601
E_4	2.21357
E_3	1.58112
E_2	0.94867
E_1	0.31622

i) v_i(x) $= |x|$



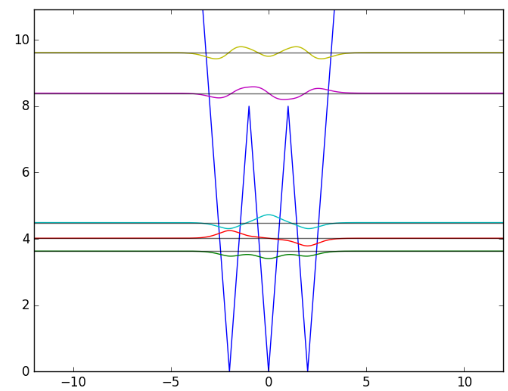
E_5	4.82004
E_4	4.08791
E_3	3.24817
E_2	2.33809
E_1	1.01878

ii) $v_{ii}(\mathrm{x}) = 12(\frac{x}{10})^4 - \frac{x^2}{18} + \frac{x}{8} + \frac{13}{10}$



E_5	1.94743
E_4	1.60332
E_3	1.40186
E_2	1.04243
E_1	0.39068

iii) $v_{iii}(\mathrm{x}) = 8|||x| - 1| - 1|$



E_5	9.61648
E_4	8.39430
E_3	4.49088
E_2	4.01853
E_1	3.63108

b)

I evaluated these integrals using *scipy.integrate.simps* which employs the composite Simpsons rule

For $v_i(\mathrm{x}) = |x|$

Energy	Probability [0,6]
E_5	0.49688
E_4	0.49961
E_3	0.49998
E_2	0.50000
E_1	0.50000

For v_i(x) = $|x|$

Energy	Probability [0,6]
E_5	0.53251
E_4	0.39990
E_3	0.78731
E_2	0.03036
E_1	0.00032

For v_{iii}(x) = $8|||x| - 1| - 1|$

Energy	Probability [0,6]
E_5	0.50000
E_4	0.50000
E_3	0.50000
E_2	0.50000
E_1	0.50000

These probabilities do make sense when you compare then to the wavefunctions.