AM205 HW5 Writeup

Jiawen Tong

December 1, 2017

Problem 1

(a) Given the Rosenbrock's function, we can calculate its gradient and use the minus of it as the direction of line search. By iteratively moving toward $-\nabla f(x)$ by an appropriate step, we will approach the minimum (stationary point). For each iteration, **the step size of line search is determined by applying `scipy.optimize.minimize`** to $g(\eta) = f(x - \eta \nabla f(x))$.

$$f(x) = f(x, y) = 100(y - x^2)^2 + (1 - x)^2$$

$$-\nabla f(x) = (-f_x, -f_y) = (400x(y - x^2) + 2(1 - x), -200(y - x^2))$$

Using **steepest descent**, the iterations required for the three starting points are:

- **(-1,1)** number of iterations = 2000
- **(0,1)** number of iterations = 2000
- **(2,1)** number of iterations = 2000

See the contour plot of Rosenbrock function and the optimization path with steepest descent in Figure 1.

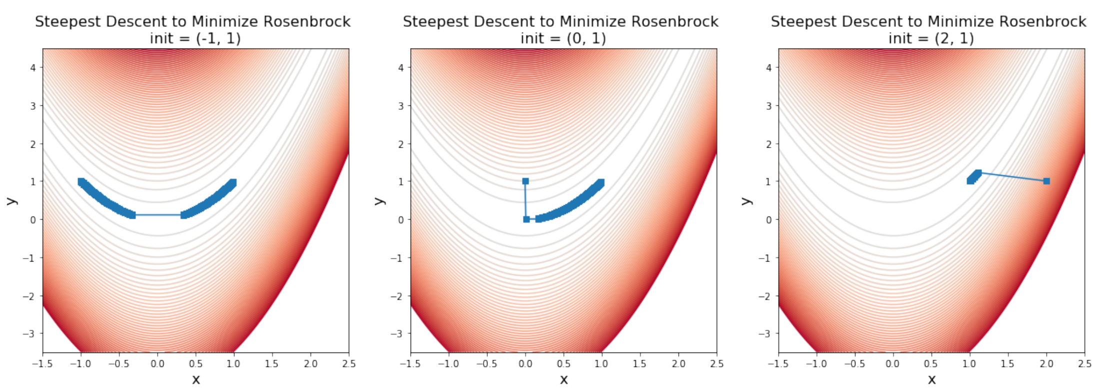

Figure 1: Problem 1: (a) steepest descent

(b) Using **Newtons's method** to find the minimum, we actually find the roots of the Jacobian of f(x). We need to calculate the Jacobian (J, the same as $\nabla f(x)$) and Hessian (H) of f(x). The roots are found by starting from an appropriate initial guess and iteratively update this guess by solving Δx for $H\Delta x = -\nabla f(x)$.

Using **Newtons's method**, the iterations required for the three starting points are:

- **(-1,1)** number of iterations = 3
- **(0,1)** number of iterations = 6
- **(2,1)** number of iterations = 6

See the contour plot of Rosenbrock function and the optimization with Newton's method in Figure 2.

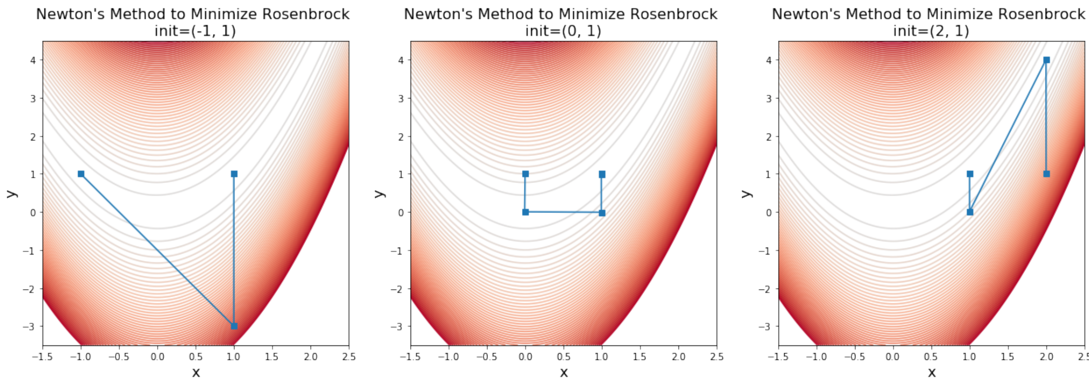

Figure 2: Problem 1: (b) Newton's method

(c) Finding the minimum using **BFGS** is similar to Newtons's method. Instead of iteratively updating the initial guess by solving Δx for $H\Delta x = -\nabla f(x)$, we solve Δx for $B\Delta x = -\nabla f(x)$ where B is an approximation of the Hessian. My implementation is based on the algorithm on page 14 of `lecture 19` notes. Specifically, set the $B_0 = I_2$ and in each iteration, do:

$$solve \quad B_k \Delta x = -\nabla f(x)$$
$$x_{k+1} = x_k + \Delta x_k$$
$$y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$$
$$\Delta B_k = \frac{y_k y_k^T}{y_k^T \Delta x_k} - \frac{B_k \Delta x_k \Delta x_k^T B_k}{\Delta x_k^T B_k \Delta x_k}$$
$$B_{k+1} = B_k + \Delta B_k$$

Using **BFGS**, the iterations required for the three starting points are:

- **(-1,1)** number of iterations = 124
- **(0,1)** number of iterations = 38
- **(2,1)** number of iterations = 45

See the contour plot of Rosenbrock function and the optimization with BFGS in Figure 3.

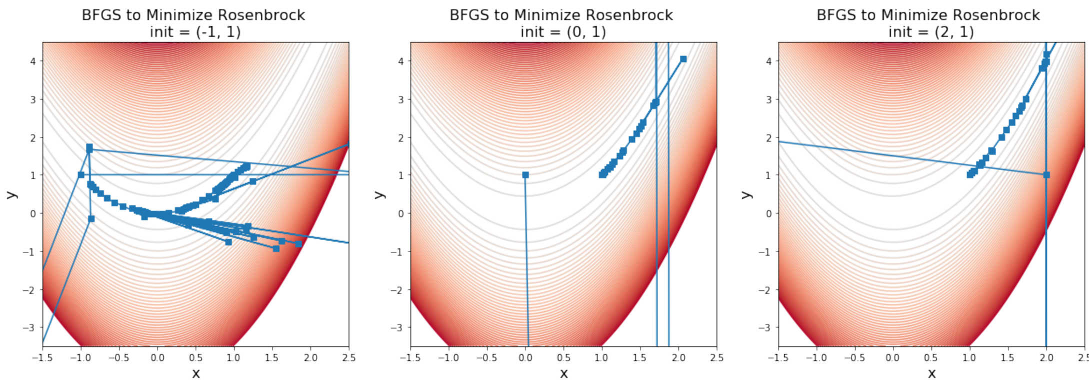

Figure 3: Problem 1: (c) BFGS

Problem 2

(a)

$$\mathcal{L} = T + \lambda(I - R)$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = I - R = \int_0^L \sqrt{1 + (\frac{dy}{dx})^2} dx - R$$

$$\nabla_b \mathcal{L} = (\frac{\partial \mathcal{L}}{\partial b_1}, \frac{\partial \mathcal{L}}{\partial b_2}, \ldots, \frac{\partial \mathcal{L}}{\partial b_{20}})$$

For k = 1, 2, ..., 20:

$$\frac{\partial \mathcal{L}}{\partial b_k} = \rho\omega^2 \int_0^L (2ysin\frac{\pi kx}{L}\sqrt{1 + (\frac{dy}{dx})^2} + y^2\frac{\pi k}{L}cos\frac{\pi kx}{L}\frac{\frac{dy}{dx}}{\sqrt{1 + (\frac{dy}{dx})^2}})dx + \lambda \int_0^L (\frac{\pi k}{L}cos\frac{\pi kx}{L}\frac{\frac{dy}{dx}}{\sqrt{1 + (\frac{dy}{dx})^2}})dx$$

(b)

$$y(x) = \Sigma_{k=1}^{20} b_k sin\frac{\pi kx}{L}$$

$$\frac{dy}{dx} = \frac{\pi}{L}[\Sigma_{k=1}^{20} kb_k cos\frac{\pi kx}{L}]$$

By composite trapezoid rule (n=250, h=L/n),

$$Q = \int_0^L f(x)dx \sim h[0.5f(x_0) + 0.5f(x_n) + f(x_1) + \ldots + f(x_{n-1})]$$

, we can express the $\nabla_b \mathcal{L}$ and $\frac{\partial \mathcal{L}}{\partial \lambda}$ as a collection of 21 functions. Plug in the 21 functions and the initial guess of $b_1, \ldots, b_{20}, \lambda$ as the parameters for the python's routine `scipy.optimize.fsolve`, we will get the roots of $\nabla \mathcal{L}$.

Under $R = 3$, $\omega = L = \rho = 1$ and the initial guess of $b_1 = 1.3$ with all other components being zero, the optimized values of b and λ are in Figure 4. The optimized solution for $y(x)$ is plotted in Figure 5.

(c)

```
5 b_lam1
```

```
array([ 1.44289102e+00, -8.10521914e-12,  1.00094880e-01,
       -3.08717722e-12,  7.50006855e-03,  1.23656531e-12,
        5.62211237e-04,  7.53751475e-13,  4.21439112e-05,
       -1.04484445e-13,  3.15914916e-06,  4.53378613e-13,
        2.36812546e-07,  3.72462709e-13,  1.77512092e-08,
       -2.75615094e-13,  1.32826918e-09,  7.50383908e-14,
        9.29831429e-11,  3.06261741e-13, -2.20982903e+00])
```

Figure 4: Problem 2: (b) values of b (first 20) and λ (the last one)

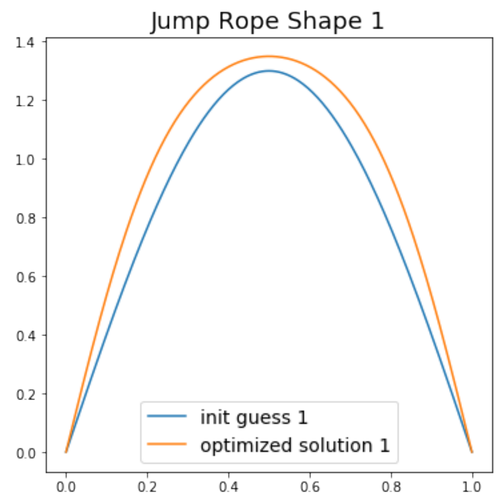

Figure 5: Problem 2: (b) initial and optimized solution for $y(x)$

Similar to part (b), under $R = 3$, $\omega = L = \rho = 1$ and the initial guess of $b_2 = 0.7$ with all other components being zero, the optimized values of b and λ are in Figure 6. The optimized solution for $y(x)$ is plotted in Figure 7.

```
5 b_lam2
```

```
array([-6.78326864e-12,  7.21445497e-01,  7.53019760e-12,
        1.48672174e-12, -3.62036242e-13,  5.00473784e-02,
        1.60638303e-12,  8.82583142e-13,  1.00921372e-12,
        3.74980724e-03,  7.71210472e-13,  4.79102307e-13,
        2.32528926e-13,  2.80369427e-04,  5.29007697e-13,
        2.60338184e-13,  1.35530816e-14,  1.94822263e-05,
        1.58498140e-13,  1.03973326e-13, -5.52457258e-01])
```

Figure 6: Problem 2: (c) values of b (first 20) and λ (the last one)

Problem 3

(a) I used **central difference approximation** $\frac{\partial^2 \Psi(x_j)}{\partial x^2} \sim \frac{U_{j+1} - 2U_j + U_{j-1}}{\Delta x^2}$ to discretize the Schrodinger equation. The truncation error of this discretization is

$$T = \frac{\Psi(x_{j+1}) - 2\Psi(x_j) + \Psi(x_{j-1})}{\Delta x^2} - \frac{\partial^2 \Psi(x_j)}{\partial x^2}$$

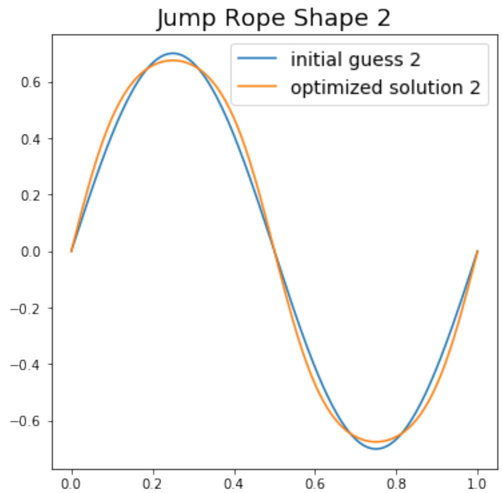

Figure 7: Problem 2: (c) initial and optimized solution for $y(x)$

By Taylor series,

$$\Psi(x_{j+1}) = \Psi(x_j) + \Delta x \frac{\partial \Psi(x_j)}{\partial x} + \frac{\Delta x^2}{2}\frac{\partial^2 \Psi(x_j)}{\partial x^2} + \frac{\Delta x^3}{6}\frac{\partial^3 \Psi(x_j)}{\partial x^3} + O(\Delta x^4)$$

$$\Psi(x_{j-1}) = \Psi(x_j) - \Delta x \frac{\partial \Psi(x_j)}{\partial x} + \frac{\Delta x^2}{2}\frac{\partial^2 \Psi(x_j)}{\partial x^2} - \frac{\Delta x^3}{6}\frac{\partial^3 \Psi(x_j)}{\partial x^3} + O(\Delta x^4)$$

Hense, $T = O(\Delta x^2)$. This proves that **the central difference approximation is second-order accurate**.

Apply central difference approximation to Schrodinger equation $-\frac{\partial^2 \Psi}{\partial x^2} + v(x)\Psi(x) = E\Psi(x)$, we get

$$-\frac{U_{j+1} - 2U_j + U_{j-1}}{\Delta x^2} + v(x_j)U_j = EU_j$$

$$-\frac{1}{\Delta x^2}U_{j+1} + (\frac{2}{\Delta x^2} + v(x_j))U_j - \frac{1}{\Delta x^2}U_{j-1} = EU_j$$

Therefore, the solution of the Schrodinger equation is given by the eigenvectors of a matrix.

$$n = 1920 \quad \Delta x = 24/n$$

$$\begin{bmatrix} \frac{2}{\Delta x^2} + v(x_0) & -\frac{1}{\Delta x^2} & 0 & \dots & 0 \\ -\frac{1}{\Delta x^2} & \frac{2}{\Delta x^2} + v(x_1) & -\frac{1}{\Delta x^2} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & -\frac{1}{\Delta x^2} & \frac{2}{\Delta x^2} + v(x_{n-1}) & -\frac{1}{\Delta x^2} \\ 0 & \dots & 0 & -\frac{1}{\Delta x^2} & \frac{2}{\Delta x^2} + v(x_n) \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_{n-1} \\ U_n \end{bmatrix} = E \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_{n-1} \\ U_n \end{bmatrix}$$

The 5 eigenmodes for $v_0(x) = x^2/10$, $v_1(x) = |x|$, $v_2(x) = 12(\frac{x}{10})^4 - \frac{x^2}{18} + \frac{x}{8} + \frac{13}{10}$, $v_3(x) = 8|||x| - 1| - 1|$ corresponding to their 5 lowest eigenvalues are plotted as $y_i(x) = 3\Psi_i(x) + E_i$ for i = 1, ..., 5 in Figure 8.

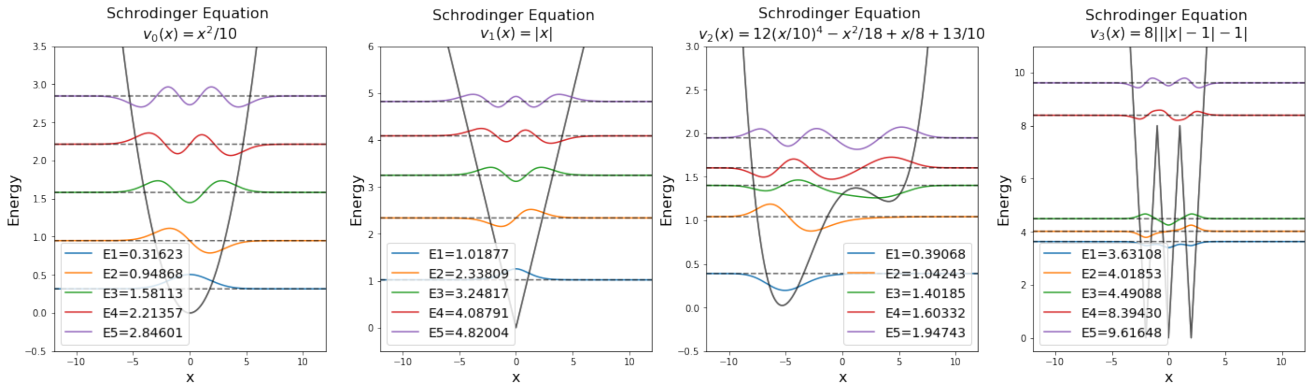

Figure 8: Problem 3: (a)

(b) I parsed in as parameters the indices of the grid points to python's routine `scipy.integrate.simps` and calculated the probability given by

$$\frac{\int_a^b |\Psi(x)|^2 dx}{\int_{-12}^{12} |\Psi(x)|^2 dx}$$

The index of the grid points at a=0 and b=6 can be found at

```
ia = int(12/dx)
ib = int(18/dx)+1
```

For potential function $v_2(x) = 12(\frac{x}{10})^4 - \frac{x^2}{18} + \frac{x}{8} + \frac{13}{10}$, the 5 probabilities corresponding to the first 5 eigenmodes are:

- E = 0.39068: p = 0.00032
- E = 1.04243: p = 0.03036
- E = 1.40185: p = 0.78730
- E = 1.60332: p = 0.39990
- E = 1.94743: p = 0.53251