- (ex 1.6) Compare and comment on the the performances of the different methods.

  Answer: First of all, when there is a fixed stepsize (i.e. $\beta = 1$), the gradient norms approach infinity regardless of the methods. However, when the methods are implemented with the backtracking, their gradient norms converge to the tolerable level. From the number of iterations taken to converge the norms into a tolerable level, newton's method took far less number of iterations in comparison to gradient method. Which is expected in case of a well behaved function due to the second-order information provided through Hessian matrix. Although the newton's method performed better in our case, the gradient method is more robust (in case Hessian matrix is ill-conditioned or impractical to calculate) and less computationally costly (both in terms of efficiency and memory usage due to Hessian matrix).
  (Please run script ex1.m to view the graphs for ex 1.4, 1.5)

- (ex 2.5) Produce a table in which you compare the number of iterations required by BFGS, by Newton's method (with backtracking) and by Steepest descent method (with backtracking). You can use the results from the previous exercise. Comment the results by comparing the different methods.

Method	Number of Iterations
Newton's Method	14
BFGS	212
Steepest Descent (Gradient) Method	2245

(Please run script ex1.m to view the graphs for ex 2.3, 2.4)

BFGS is a quasi-Newton's method which approximates the Hessian matrix without directly computing it. Hence, it converges faster (lesser iteration) than gradient method but slower than the usual Newton's method, although it requires less memory consumption than Newton's. However, it may not converge to the global minimum and it's performance is relevant to the point of initialization.

- Steepest Descent

$$f = \frac{1}{2}x^T Ax - b^T x \tag{1}$$

Where A is positive definite. How many iterations does the SD method take to minimize the function f if we use the optimal step length? Please, prove your answer.

The optimal step length for minimizing a function using the Steepest Descent method can be found by solving the following one-dimensional optimization problem where α_k is the optimal step length at the k^{th} iteration and x_k is the current iterate, and $\nabla f(x_k)$ is the gradient. Steepest Descent method is considered to have converged when the gradient shrinks below the pre-defined tolerance level (Approximately 0):

$$\alpha_k = argmin_\alpha f(x_k - \alpha \nabla f(x_k)) \tag{2}$$

According to the gradient function which is:

$$\nabla f(x) = Ax - b \tag{3}$$

With the minimization function:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) \tag{4}$$

We know A is positive definite so our f is strictly convex, and has a unique minimum. Hence, the optimal α_k can be found by setting the function to 0.

$$\frac{d}{d\alpha} f(x_k - \alpha \nabla f(x_k)) = 0 \tag{5}$$

$$\alpha_k = \frac{\nabla f(x_k)^T \nabla f(x_k)}{\nabla f(x_k)^T A \nabla f(x_k)} \tag{6}$$

while we can't determine the exact number of iterations required for convergence without additional information, we can be confident that the Steepest Descent method with the optimal step length will converge to the minimum of f in a finite number of iterations for convex functions like the one described.

From what we know, we can rewrite the minimization function as follwing:

$$x_1 = x_0 - \alpha_0(Ax_0 - b) = A^{-1}b \tag{7}$$

$$x_1 = A^{-1}b \tag{8}$$

$$A * x_1 = A * A^{-1}b \tag{9}$$

By expanding this, we can realize the minimum of the f is equivalent to solving:

$$A * x = b \tag{10}$$