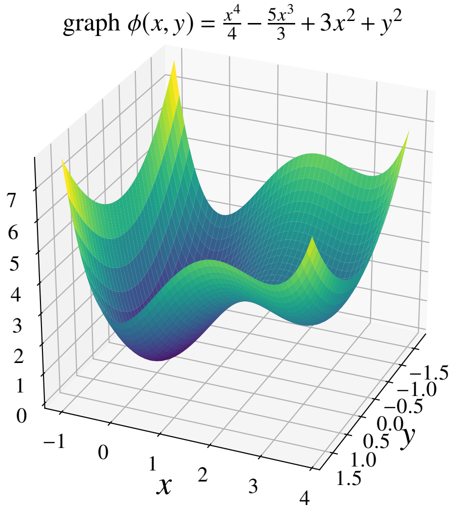
MAT 423: Homework 8 (11/07)

In the three exercises, we consider the following function to minimize:

$$\phi(x,y) = \frac{x^4}{4} - \frac{5x^3}{3} + 3x^2 + y^2.$$



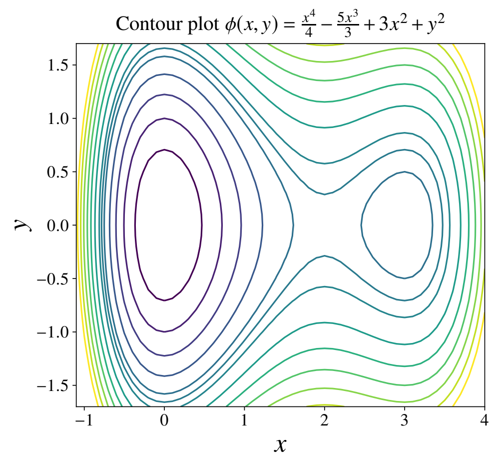


Figure 1: Surface plot of the function $\phi(x,y)$ (left) and contour plot (right), i.e. curves where $\phi(x,y) = $ Constant.

Ex 1. [**Warm up**]

a) Show that ϕ is *coercive*[1], i.e.:

$$\phi(x,y) \overset{\|(x,y)\|\to+\infty}{\longrightarrow} +\infty.$$

Deduce that there exists an infinimum of ϕ on $\mathbb{R}^2$:

$$\inf_{(x,y)\in\mathbb{R}^2} \phi(x,y) = \min_{(x,y)\in\mathbb{R}^2} \phi(x,y).$$

b) Find and study the critical points of ϕ.
Deduce the global minimum of ϕ on $\mathbb{R}^2$.

[1]In other words, ϕ *goes to infinity at infinity.*

Ex 2. [**Newton vs BFGS**]

We would like to compare the Newton's method and the quasi-Newton BFGS to find critical points of ϕ (i.e. solving $\nabla\phi = \mathbf{0}$).

a) Starting from $\mathbf{x}^{(0)} = (\frac{1}{2}, 1)$, compute explicitly the first iteration of the Newton's method:

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} - [D^2\phi(\mathbf{x}^{(0)})]^{-1}\nabla\phi(\mathbf{x}^{(0)}).$$

b) Implement the BFGS method (see next page).
Compare the evolution of $\phi(\mathbf{x}^{(k)})$ with $\mathbf{x}^{(k)}$ computed using **i)** the Newton's method and **ii)** the quasi-Newton method BFGS (plot in logarithm scale).

c) Find another initial condition $\mathbf{x}^{(0)}$ such that both methods do **not** converge to the global minimum.

Ex 3. [**Gradient descent**]

a) Implement the gradient descent algorithm:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \lambda\nabla\phi(\mathbf{x}^{(k)}), \tag{1}$$

where $\lambda > 0$ is a fixed parameter (the "learning rate").

b) Starting from $\mathbf{x}^{(0)} = (\frac{1}{2}, 1)$, study the convergence of the algorithm for different learning rates, i.e. taking $\lambda = 0.05$, 0.1, 0.2, 0.5.

Extra) One can *adapt* the learning rate λ while performing the gradient descent. The idea is to reduce λ when the updated value $\phi(\mathbf{x}^{(k+1)})$ does not decay. The algorithm is as followed:

1) Starting from $\mathbf{x}^{(k)}$ and λ, compute:

$$\widetilde{\mathbf{x}} = \mathbf{x}^{(k)} - \lambda\nabla\phi(\mathbf{x}^{(k)}).$$

2) If $\phi(\widetilde{\mathbf{x}}) < \phi(\mathbf{x}^{(k)})$ then:

let $\mathbf{x}^{(k+1)} = \widetilde{\mathbf{x}}$, keep λ

else:

let $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}$, $\lambda = \frac{\lambda}{2}$.

Go back to 1)

Apply the algorithm starting with $\mathbf{x}^{(0)} = (\frac{1}{2}, 1)$ and $\lambda = 0.5$.

Broyden–Fletcher–Goldfarb–Shanno (BFGS) method

a) Initialization: $\mathbf{x}^{(0)}$, $\mathbf{x}^{(1)}, A^{(0)}$

   – Pick $\mathbf{x}^{(0)}$ and define $A^{(0)} = D^2\phi(\mathbf{x}^{(0)})$.
   – Define the first iteration: $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} - [D^2\phi(\mathbf{x}^{(0)})]^{-1}\nabla\phi(\mathbf{x}^{(0)})$.

b) "$n \rightsquigarrow n+1$". Denote $\Delta\mathbf{x} = \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}$ and $\Delta\mathbf{y} = \nabla\phi(\mathbf{x}^{(k)}) - \nabla\phi(\mathbf{x}^{(k-1)})$.

   ◦ Update $A^{(k)}$ (to satisfy the secant formula):

$$A^{(k)} = A^{(k-1)} + \frac{\Delta\mathbf{y} \otimes \Delta\mathbf{y}}{\langle \Delta\mathbf{y}, \Delta\mathbf{x} \rangle} - \frac{A^{(k-1)}\Delta\mathbf{y} \otimes \Delta\mathbf{y} A^{(k-1)}}{\langle A^{(k-1)}\Delta\mathbf{x}, \Delta\mathbf{x} \rangle}$$

   ◦ Update $\mathbf{x}^{(k+1)}$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - [A^{(k)}]^{-1}\nabla\phi(\mathbf{x}^{(k)}).$$

The inverse of $A^{(k)}$ should be computed using the formula:

$$[A^{(k)}]^{-1} = [A^{(k-1)}]^{-1} \ + \ \alpha\Delta\mathbf{x} \otimes \Delta\mathbf{x} \ + \ \beta(\Delta\mathbf{x} \otimes \mathbf{v} + \mathbf{v} \otimes \Delta\mathbf{x})$$

with :

$$\mathbf{v} = [A^{(k-1)}]^{-1}\Delta\mathbf{y} \quad , \quad \alpha = \frac{\langle \Delta\mathbf{x}, \Delta\mathbf{y} \rangle + \langle \mathbf{v}, \Delta\mathbf{y} \rangle}{\langle \Delta\mathbf{x}, \Delta\mathbf{y} \rangle^2} \quad , \quad \beta = -\frac{1}{\langle \Delta\mathbf{x}, \Delta\mathbf{y} \rangle}.$$

This formula is obtained using the Woodbury matrix identity (after some computations...).