Optimization

Nakul Gopalan
Georgia Tech

These slides are based on slides from Mykel Kochenderfer, Julia Roberts, Glaucio Paulino, and Rodrigo Borela Valente.

Outline

- Overview
- Unconstrained and constrained optimization
- Lagrange multipliers and KKT conditions
- Gradient descent

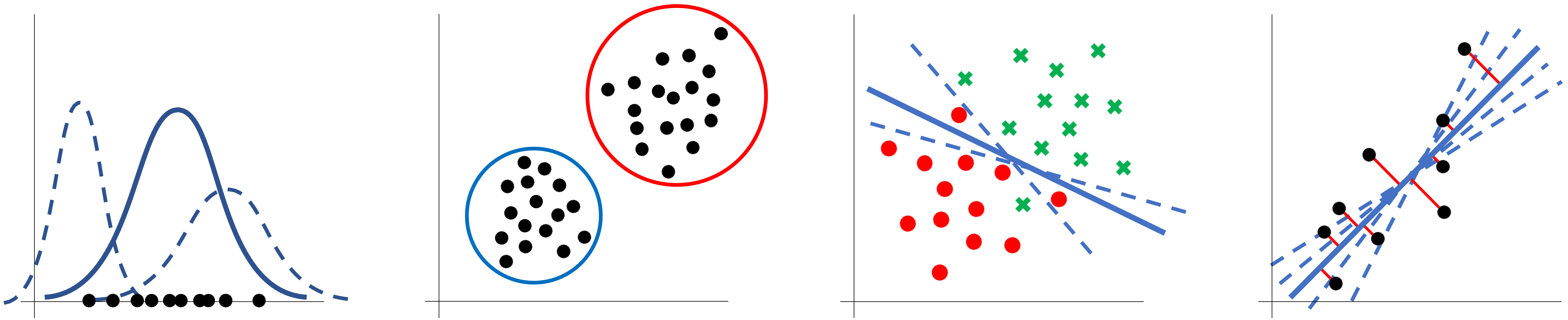
Complementary reading: Bishop PRML – Appendix E

Outline

- Overview
- Unconstrained and constrained optimization
- Lagrange multipliers and KKT conditions
- Gradient descent

Why optimization?

- Machine learning and pattern recognition algorithms often focus on the minimization or maximization of a quantity
  - Likelihood of a distribution given a dataset
  - Distortion measure in clustering analysis
  - Misclassification error while predicting labels
  - Square distance error for a real value prediction task



Basic optimization problem

- Objective or cost function $f(\mathbf{x})$ the quantity we are trying to optimize (maximize or minimize)
- The variables $x_1, x_2, \ldots, x_n$ which can be represented in vector form as $\mathbf{x}$ (Note: x_n here does NOT correspond to a point in our dataset)
- Constraints that limit how small or big variables can be. These can be equality constraints, noted as $h_k(\mathbf{x})$ and inequality constraints noted as $g_j(\mathbf{x})$

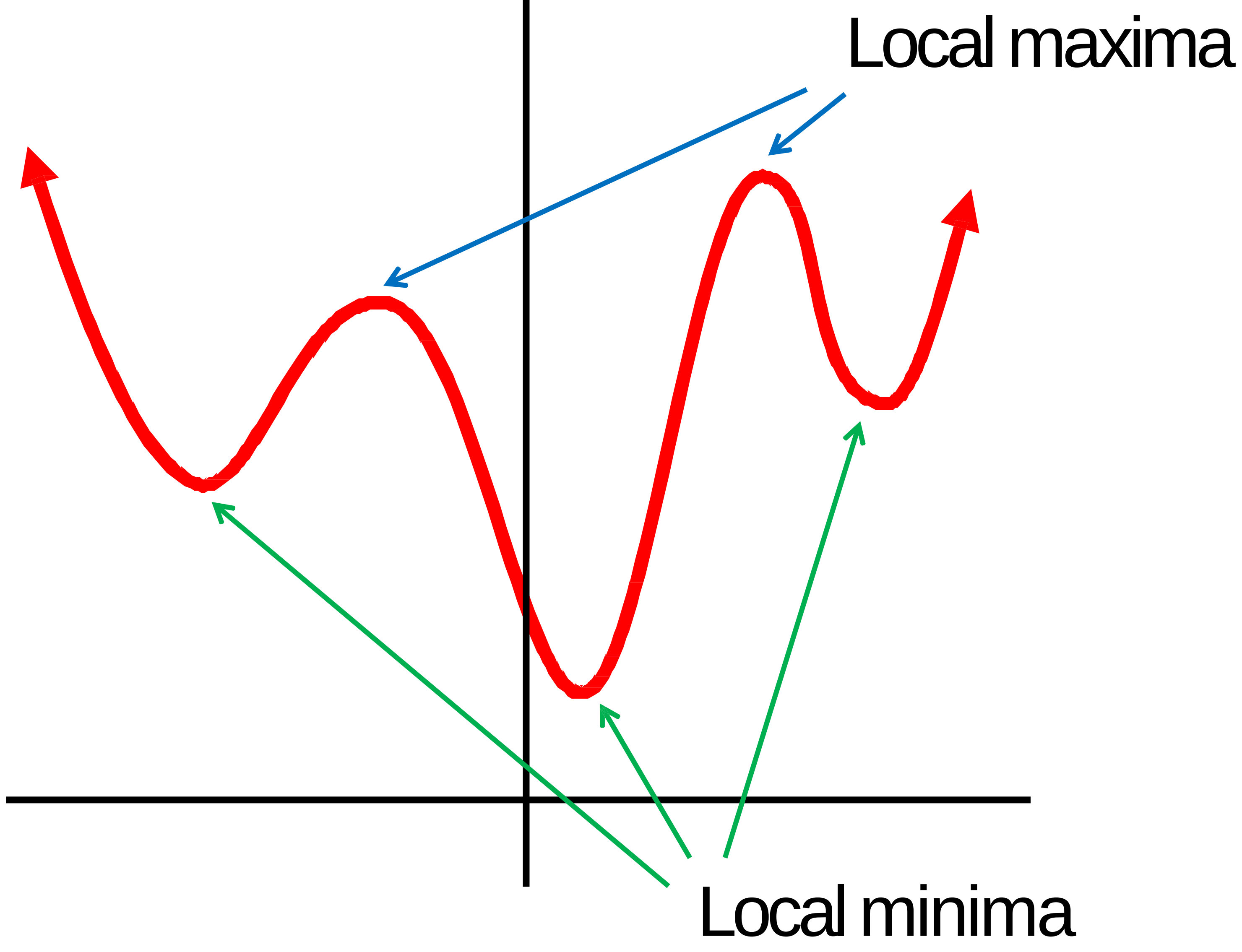
- An optimization problem is usually expressed as:

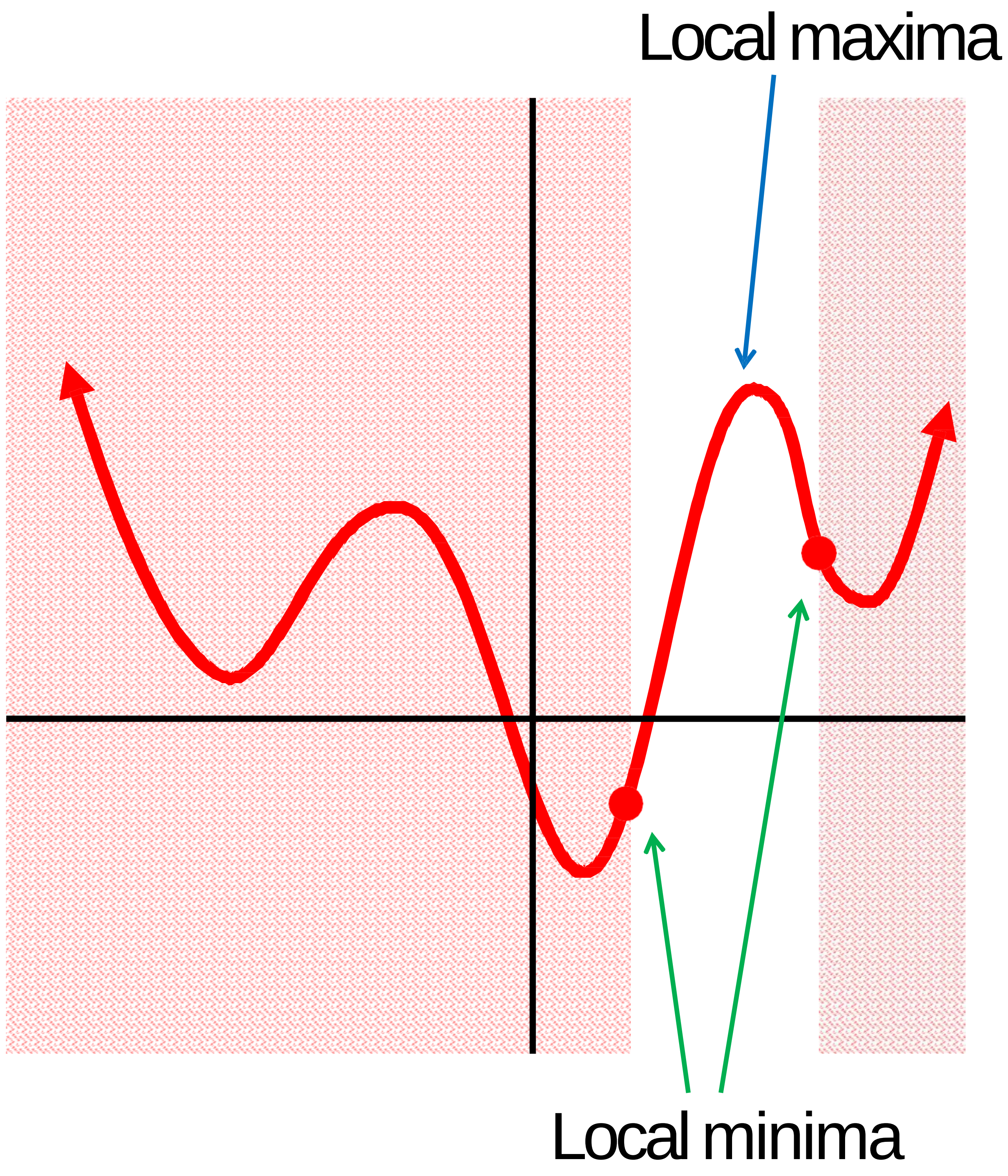
$$\max_{\mathbf{x}} f(\mathbf{x})$$
$$s.t. \quad \begin{aligned} \mathbf{g}(\mathbf{x}) &\geq 0 \\ \mathbf{h}(\mathbf{x}) &= 0 \end{aligned}$$

Outline

- Overview
- Unconstrained and constrained optimization
- Lagrange multipliers and KKT conditions
- Gradient descent

Unconstrained and constrained optimization

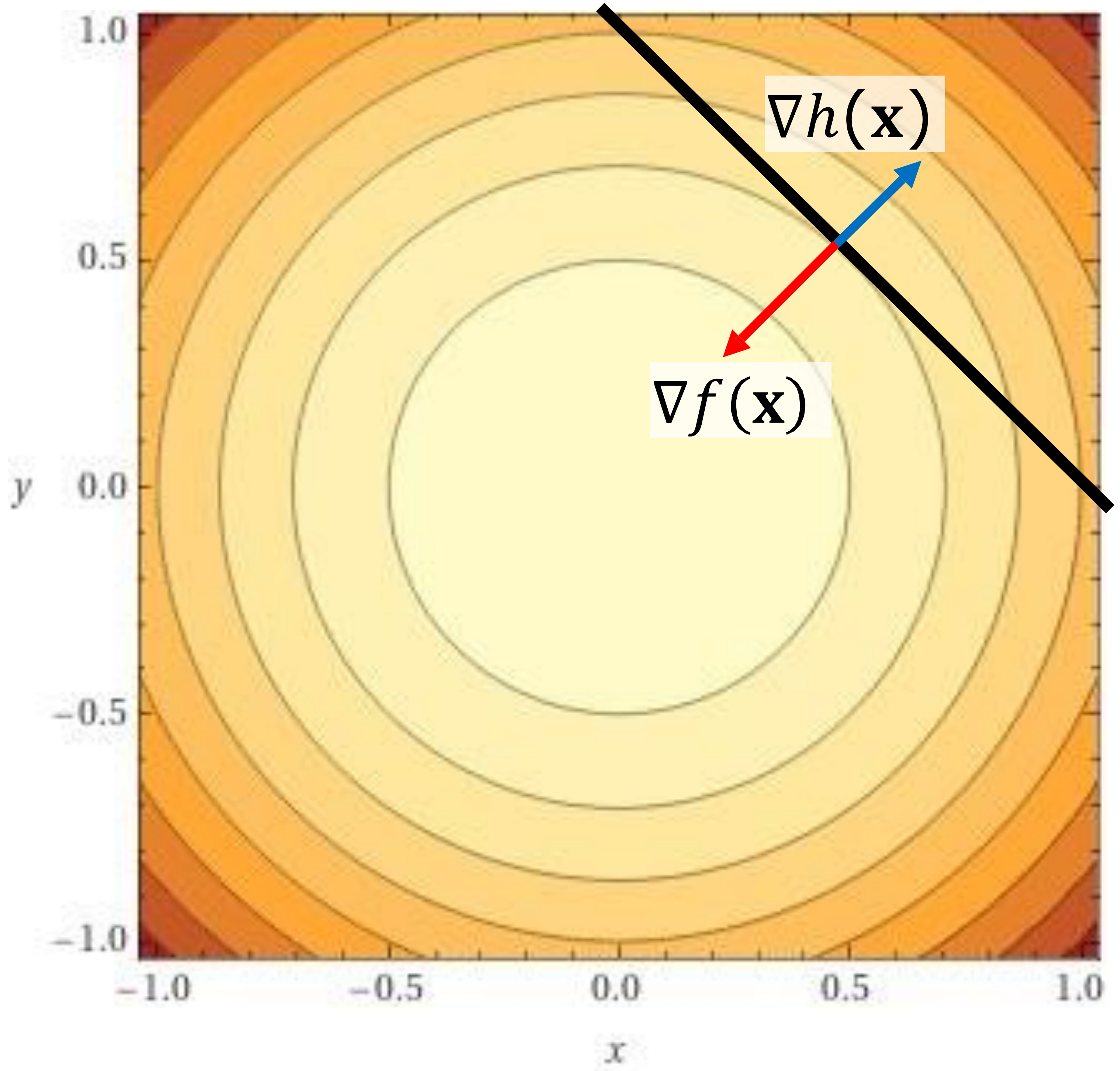




Outline

- Overview
- Unconstrained and constrained optimization
- Lagrange multipliers and KKT conditions
- Gradient descent

Lagrangian multipliers: equality constraint



$$\max_{\mathbf{x}} \quad 1 - x_1^2 - x_2^2$$

$$s.t. \;\; x_1 + x_2 - 1 = 0$$

Objective function: $f(x_1, x_2) = 1 - x_1^2 + x_2^2$

Equality constraint: $h(x_1, x_2) = x_1 + x_2 - 1 = 0$

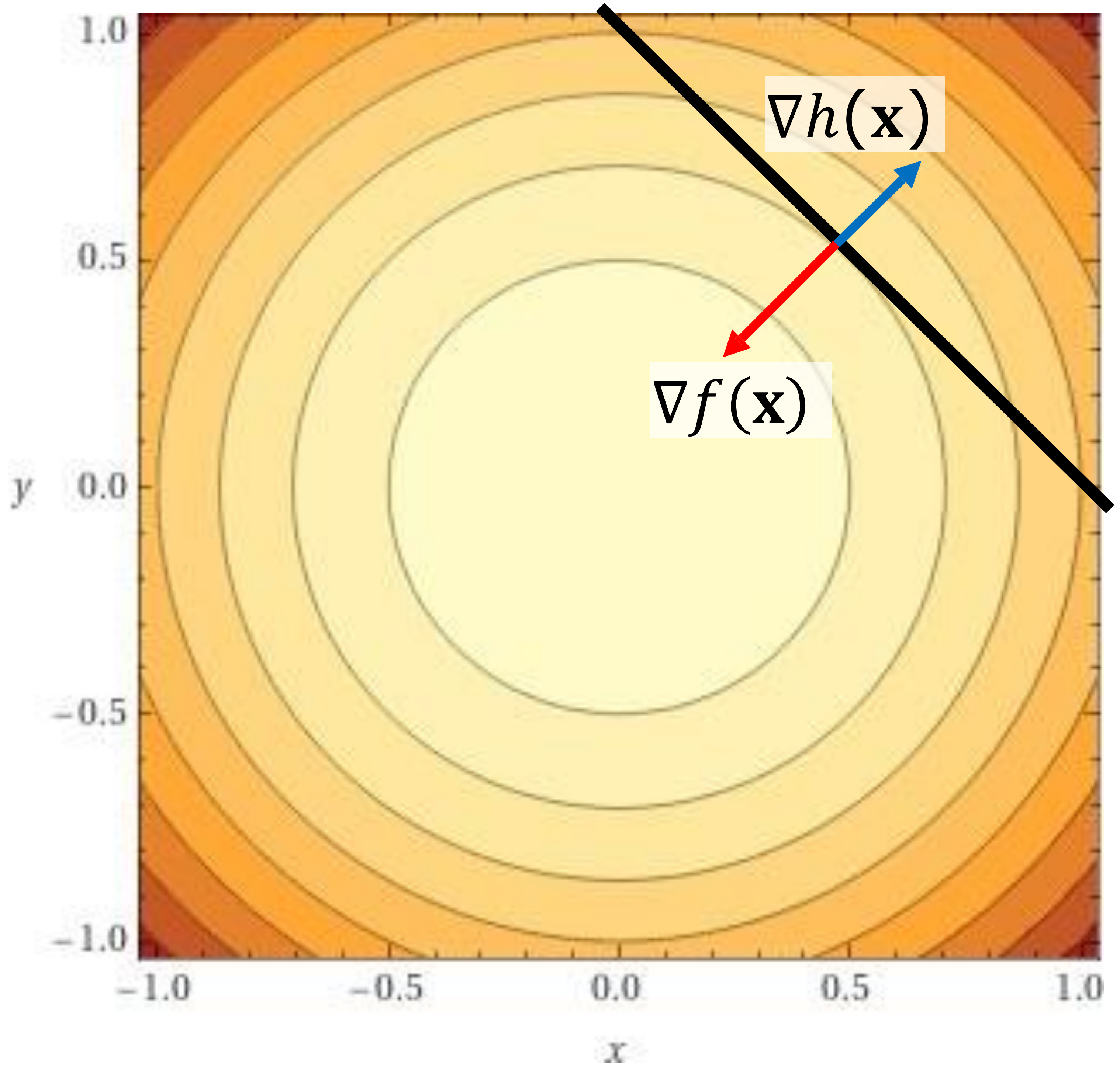
Intuition: $\nabla f(\mathbf{x}) + \mu \nabla h(\mathbf{x}) = 0$

Lagrangian: $L(\mathbf{x}, \mu) = f(\mathbf{x}) + \mu h(\mathbf{x}) = 0$

$$s.t. \quad \mu \neq 0$$

Solve $\nabla L(\mathbf{x}, \mu)$

Lagrangian multipliers: equality constraint



$$L(\mathbf{x}, \mu) = 1 - x_1^2 + x_2^2 + \mu(x_1 + x_2 - 1)$$

$$\frac{\partial L}{\partial x_1} = -2x_1 + \mu = 0$$

$$\frac{\partial L}{\partial x_2} = -2x_2 + \mu = 0$$

$$\frac{\partial L}{\partial \mu} = x_1 + x_2 - 1 = 0$$

Solution: $x_1, x_2, \mu = \left(\frac{1}{2}, \frac{1}{2}, 1\right)$

Lagrangian multipliers

- Maximization problem

$$\max_{\mathbf{x}} f(\mathbf{x})$$
$$s.t. \quad \begin{matrix} g(\mathbf{x}) \geq 0 \\ h(\mathbf{x}) = 0 \end{matrix}$$

- Lagrangian function:

$$L(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) + \lambda g(\mathbf{x}) + \mu h(\mathbf{x})$$

- KKT conditions:

$$g(\mathbf{x}) \geq 0$$
$$\lambda \geq 0$$
$$\lambda g(\mathbf{x}) = 0$$
$$\mu \neq 0$$

- Minimization problem

$$\min_{\mathbf{x}} f(\mathbf{x})$$
$$s.t. \quad \begin{matrix} g(\mathbf{x}) \geq 0 \\ h(\mathbf{x}) = 0 \end{matrix}$$

- Lagrangian function:

$$L(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) - \lambda g(\mathbf{x}) + \mu h(\mathbf{x})$$

- KKT conditions:

$$g(\mathbf{x}) \geq 0$$
$$\lambda \geq 0$$
$$\lambda g(\mathbf{x}) = 0$$
$$\mu \neq 0$$

Solve the optimization problem by resolving: $\nabla L = 0$

Outline

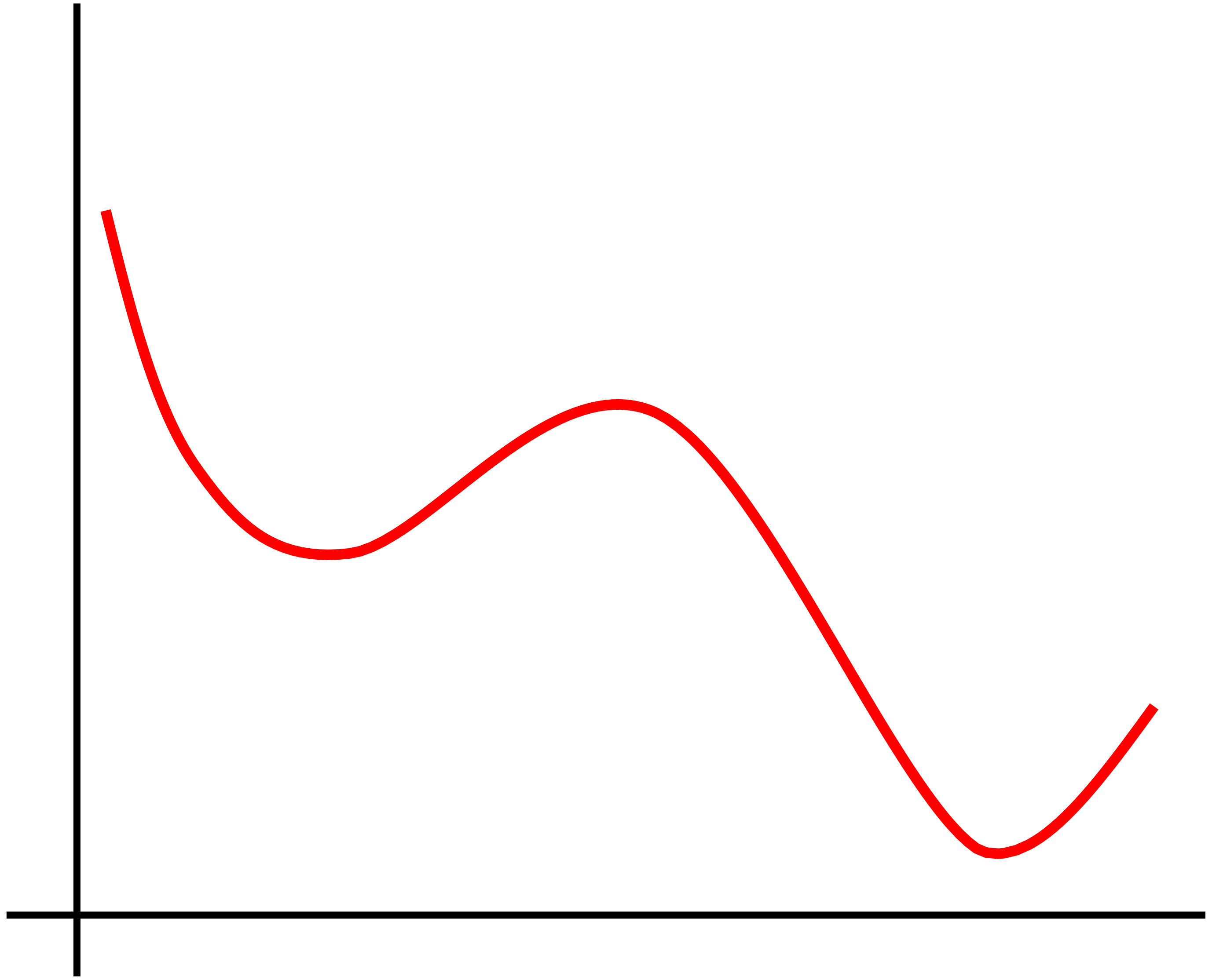
- Overview
- Unconstrained and constrained optimization
- Lagrange multipliers and KKT conditions
- Gradient descent

Search

Gradient descent

- Common in machine learning problems when not all of the data is available immediately or a closed form solution is computationally intractable
- Iterative minimization technique for differentiable functions on a domain

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \gamma \nabla F(\mathbf{x}_n)$$



Closed or Symbolic differential

$$f(x) = x^2 + x + 1$$

$$\frac{\Delta f(x)}{\Delta x} = 2x + 1$$

Method of Finite differences

$$f(x) = x^2 + x + 1$$

$$\frac{\Delta f(x)}{\Delta x} = \frac{f(x+\Delta) - f(x)}{2\Delta}$$

Autodiff

Automatic differentiation (AD): A method to get exact derivatives efficiently, by storing information as you go forward that you can reuse as you go backwards

• Takes code that computes a function and returns code that computes the derivative of that function.

• “The goal isn’t to obtain closed-form solutions, but to be able to write a program that efficiently computes the derivatives.” • Autograd, Torch Autograd

From slides of Paul Vicol, and Ryan Adams

Autodiff

An autodiff system will convert the program into a sequence of primitive operations which have specified routines for computing derivatives

Original program:

$$z = wx + b$$

$$y = \frac{1}{1 + \exp(-z)}$$

$$\mathcal{L} = \frac{1}{2}(y - t)^2$$

Sequence of primitive operations:

$$t_1 = wx$$

$$z = t_1 + b$$

$$t_3 = -z$$

$$t_4 = \exp(t_3)$$

$$t_5 = 1 + t_4$$

$$y = 1/t_5$$

$$t_6 = y - t$$

$$t_7 = t_6^2$$

$$\mathcal{L} = t_7/2$$

From slides of Roger Grosse

Gradient descent: Himmelblau's function

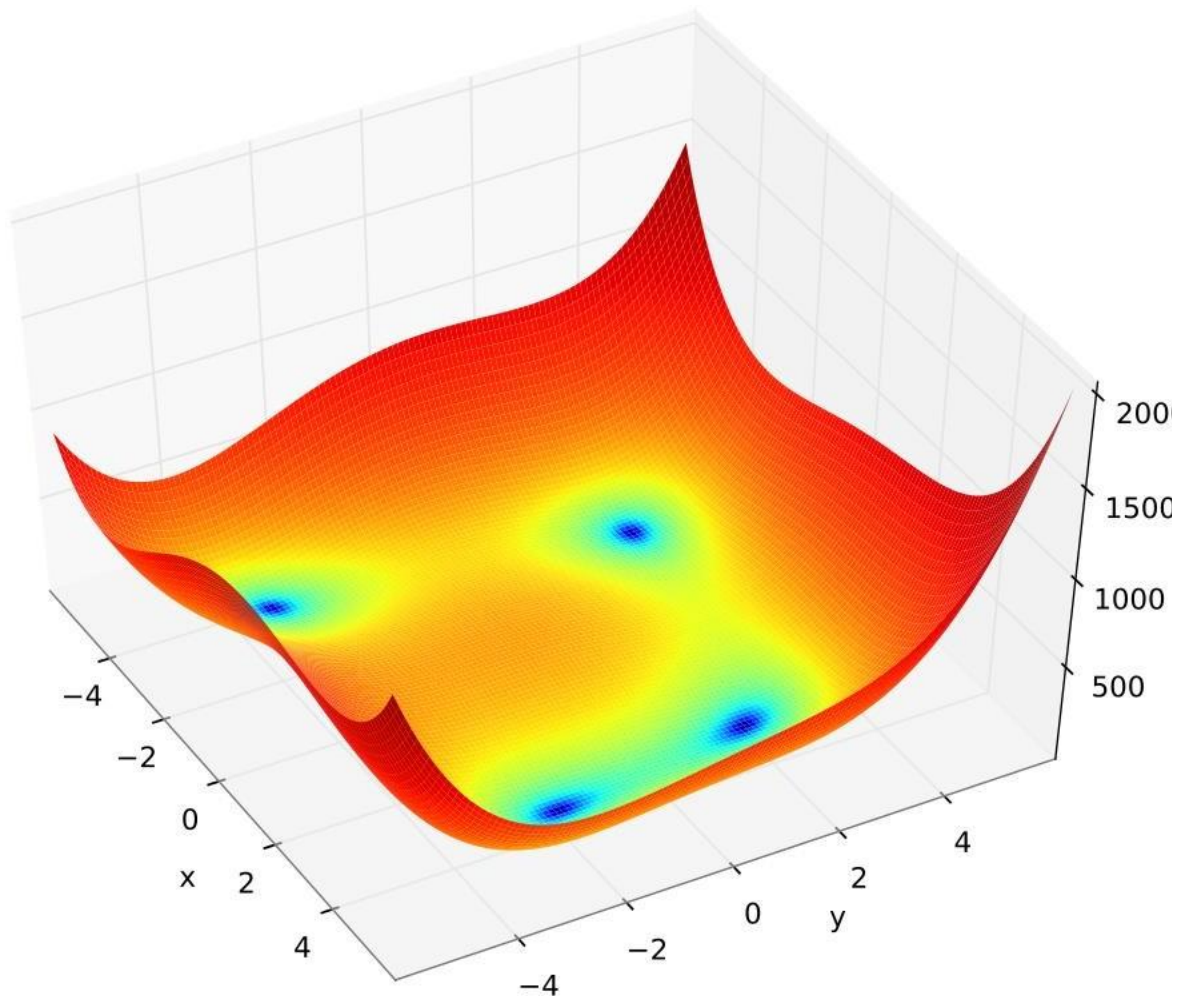


Image credit: Wikimedia

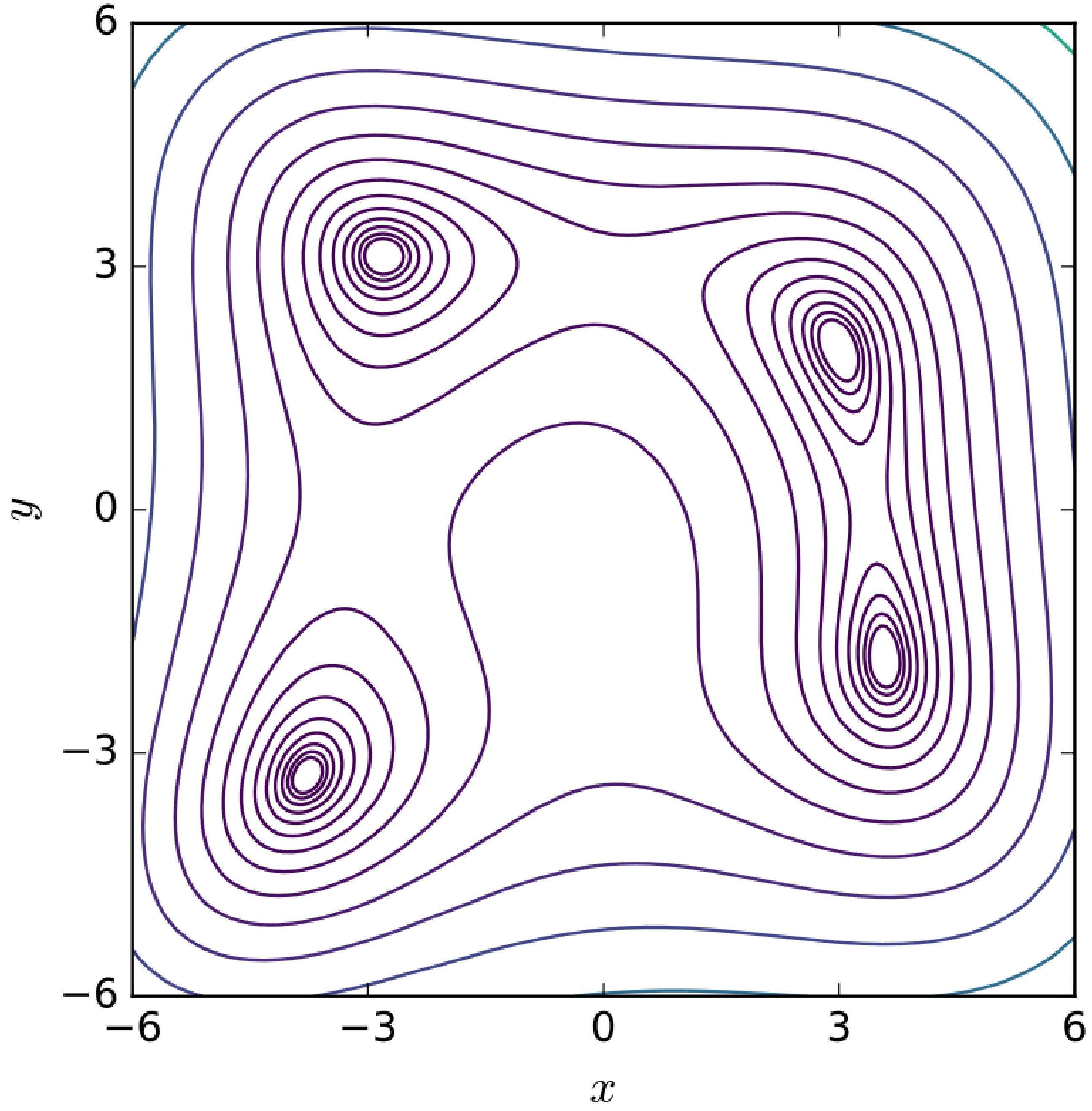


Image credit: Wikimedia