*Data Science Bootcamp, 10th January 2017*

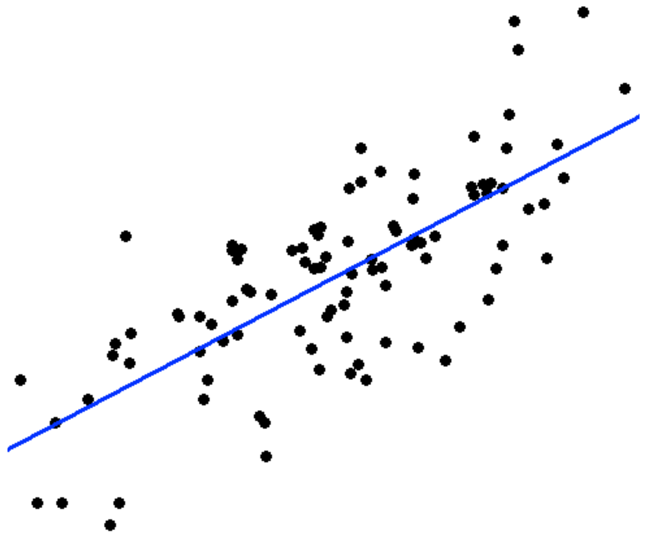
# Gradient Descent

Francisco J. R. Ruiz

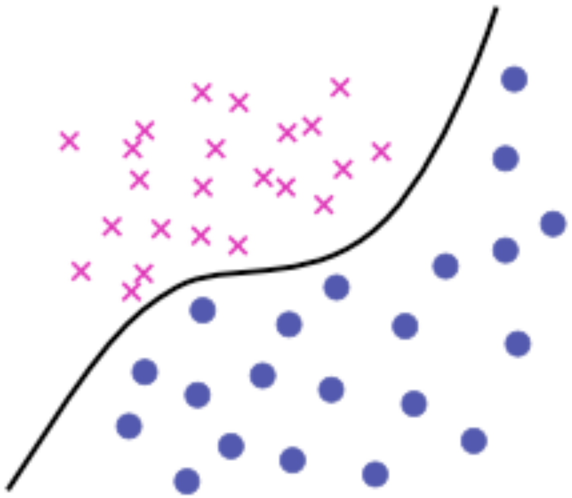
Columbia University
Data Science Institute

# Introduction

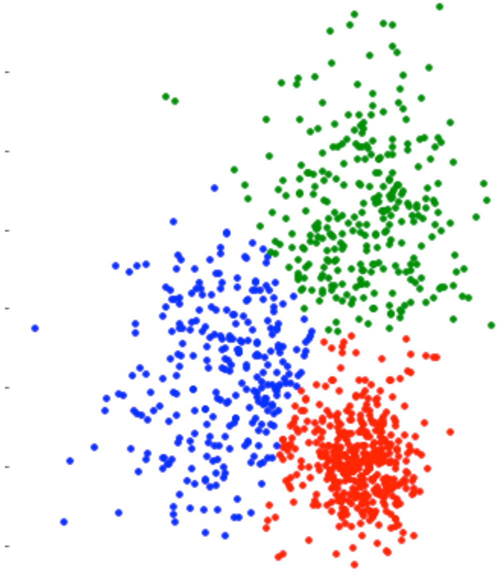
- What do these problems have in common?



Regression



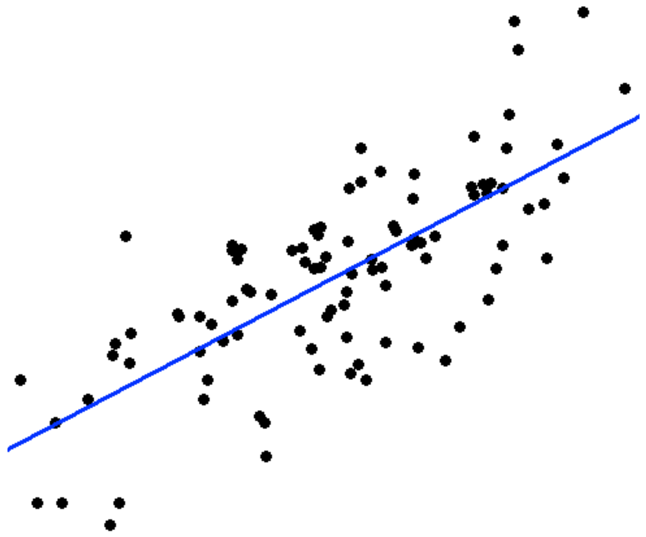
Classification



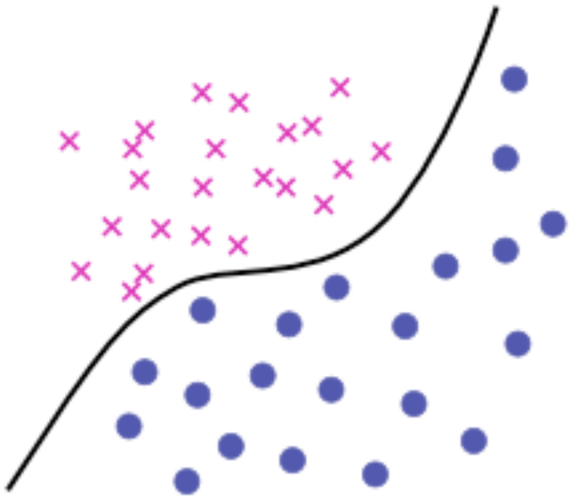
Clustering

# Introduction

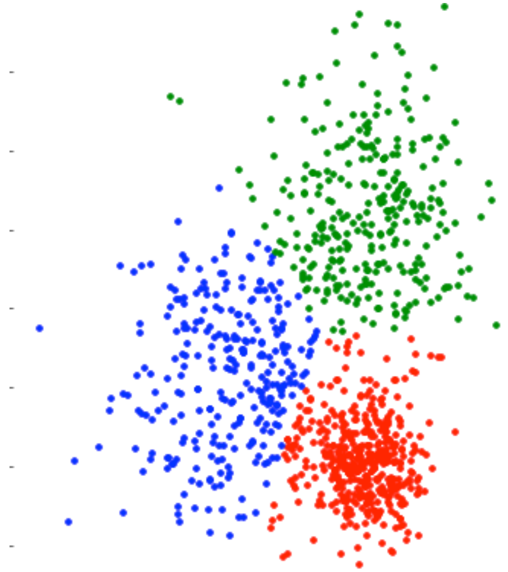
- In all of them, we define and **minimize** an error function



Regression



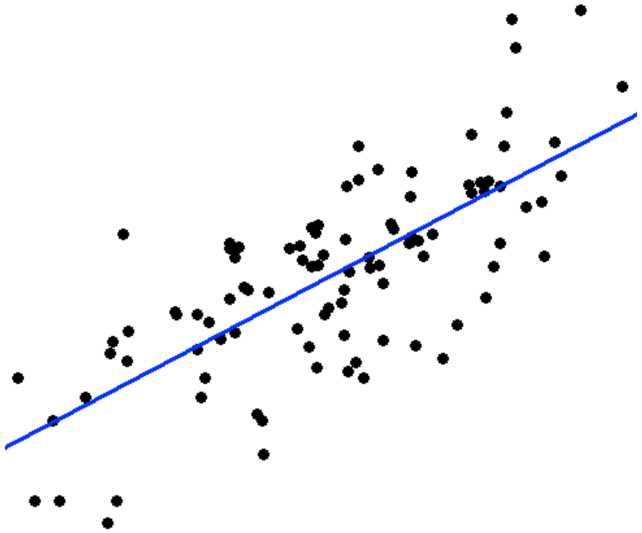
Classification



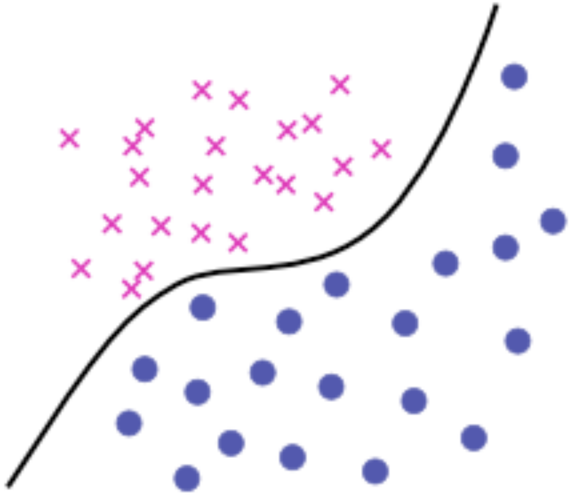
Clustering

# Introduction

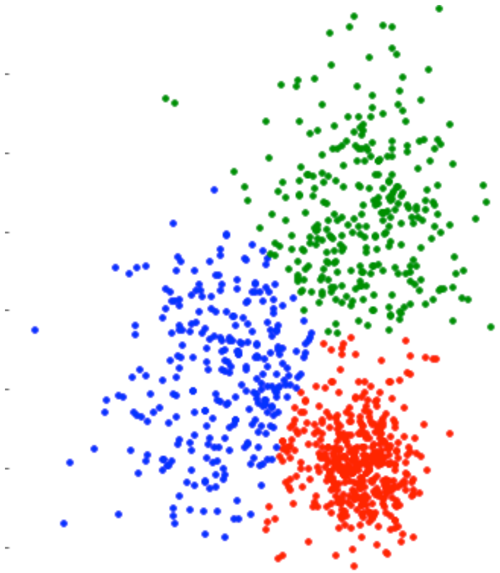
## Error function = Loss



Regression



Classification



Clustering

# Optimization in ML

- **Optimization** appears in many machine learning algorithms
  - Supervised and unsupervised learning
  - Basic and advanced methods

# Optimization in ML

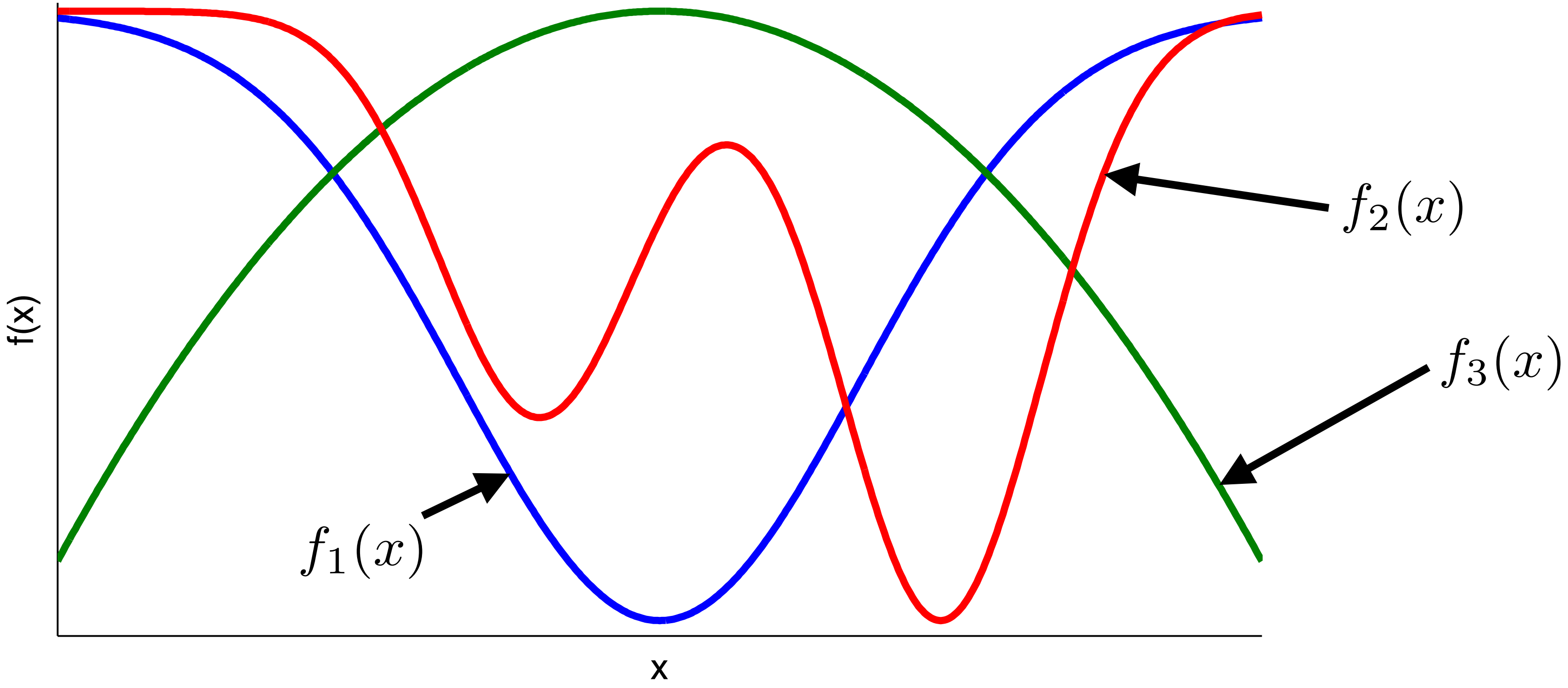
- **Optimization:** Minimize an objective function

$$\mathbf{x}^\star = \min_{\mathbf{x}} f(\mathbf{x})$$

- **Example:** Linear regression
  - The function is the sum of squared errors
  - Find the coefficients that minimize the function

# Optimization in ML

- The function can have one, multiple of none **local optima**



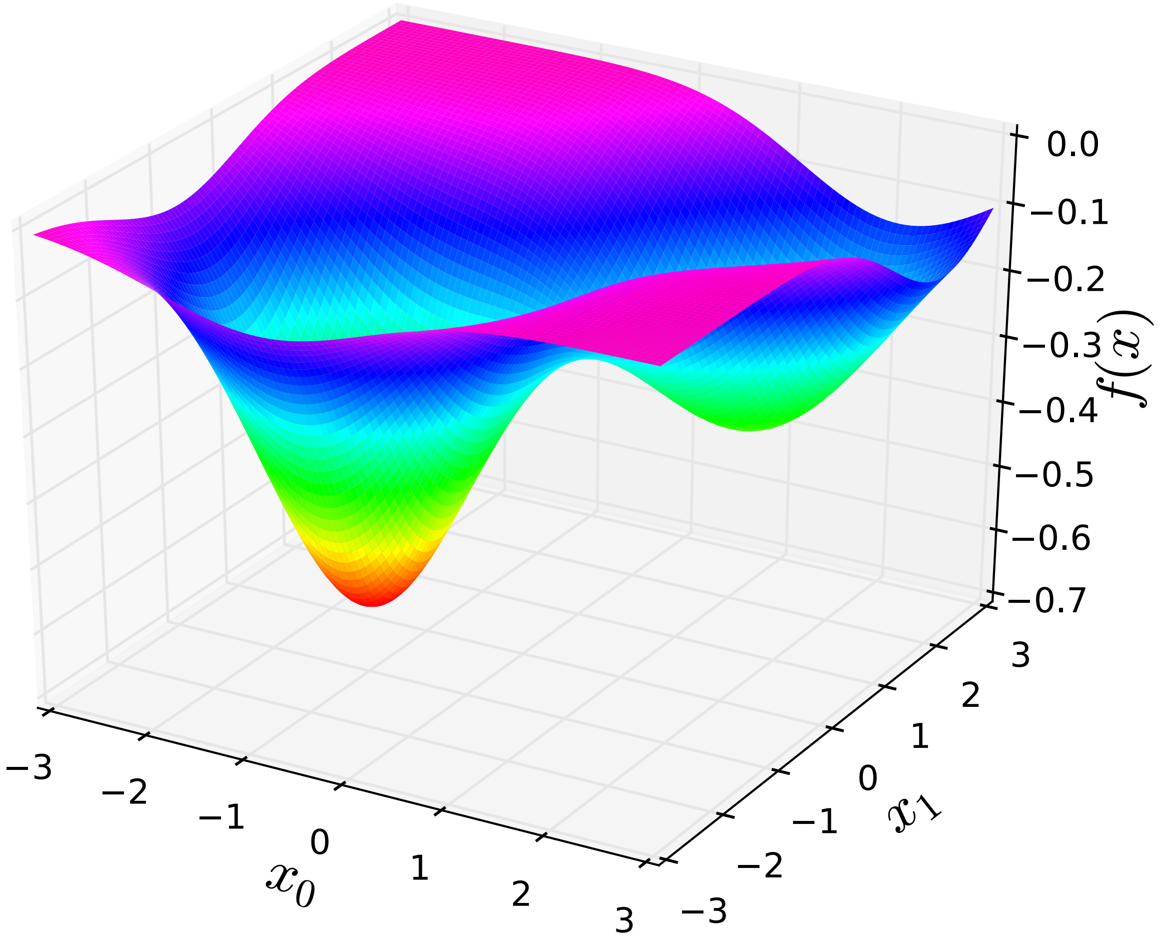
# Optimization in ML

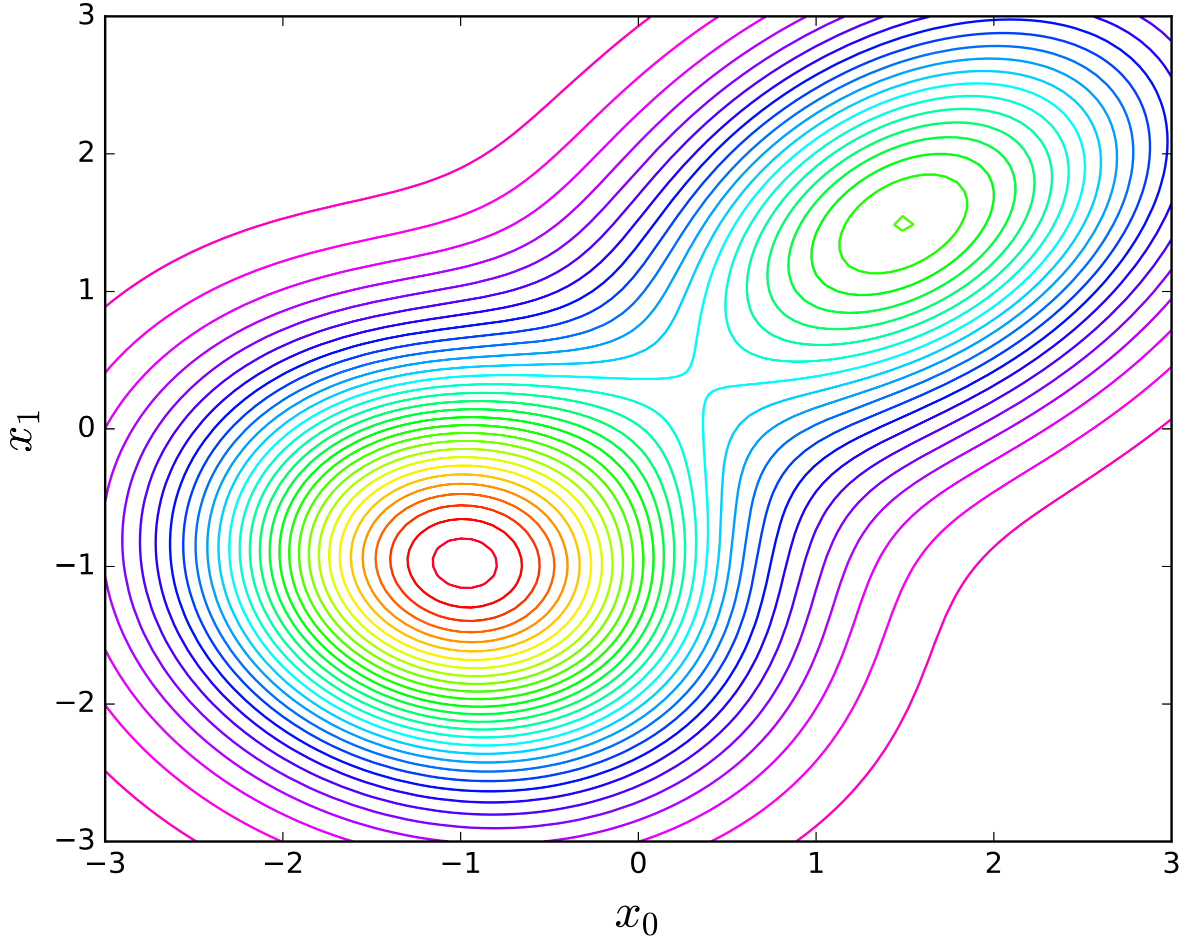
- In some cases, we can find the optima analytically
  - Example: linear regression
- In most practical cases, we need an iterative algorithm
  - Example: logistic regression

# Gradient Descent

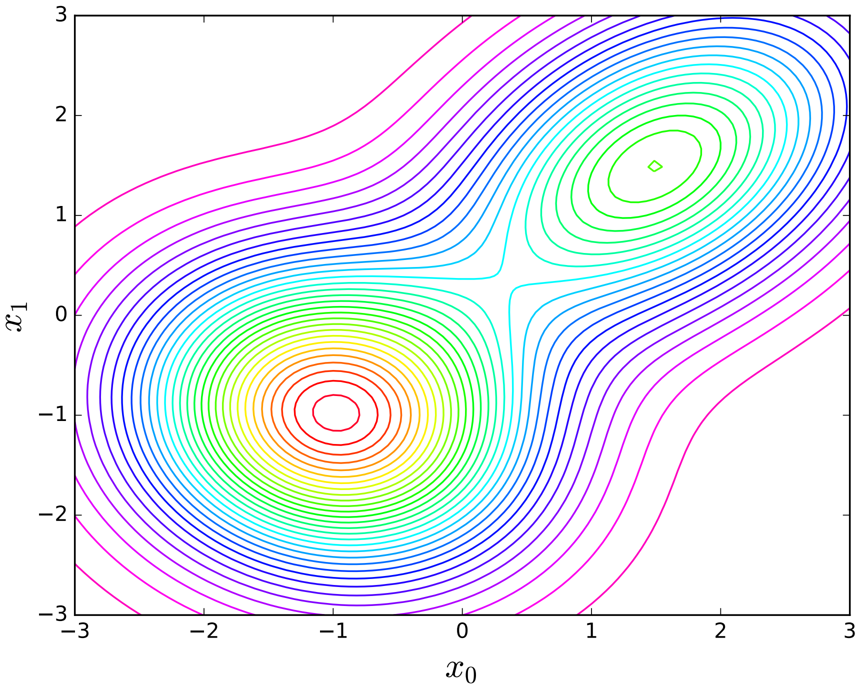
- Gradient descent is an algorithm for optimization
  - Simple to implement
  - Intuitive interpretation
  - Works also in high dimensions

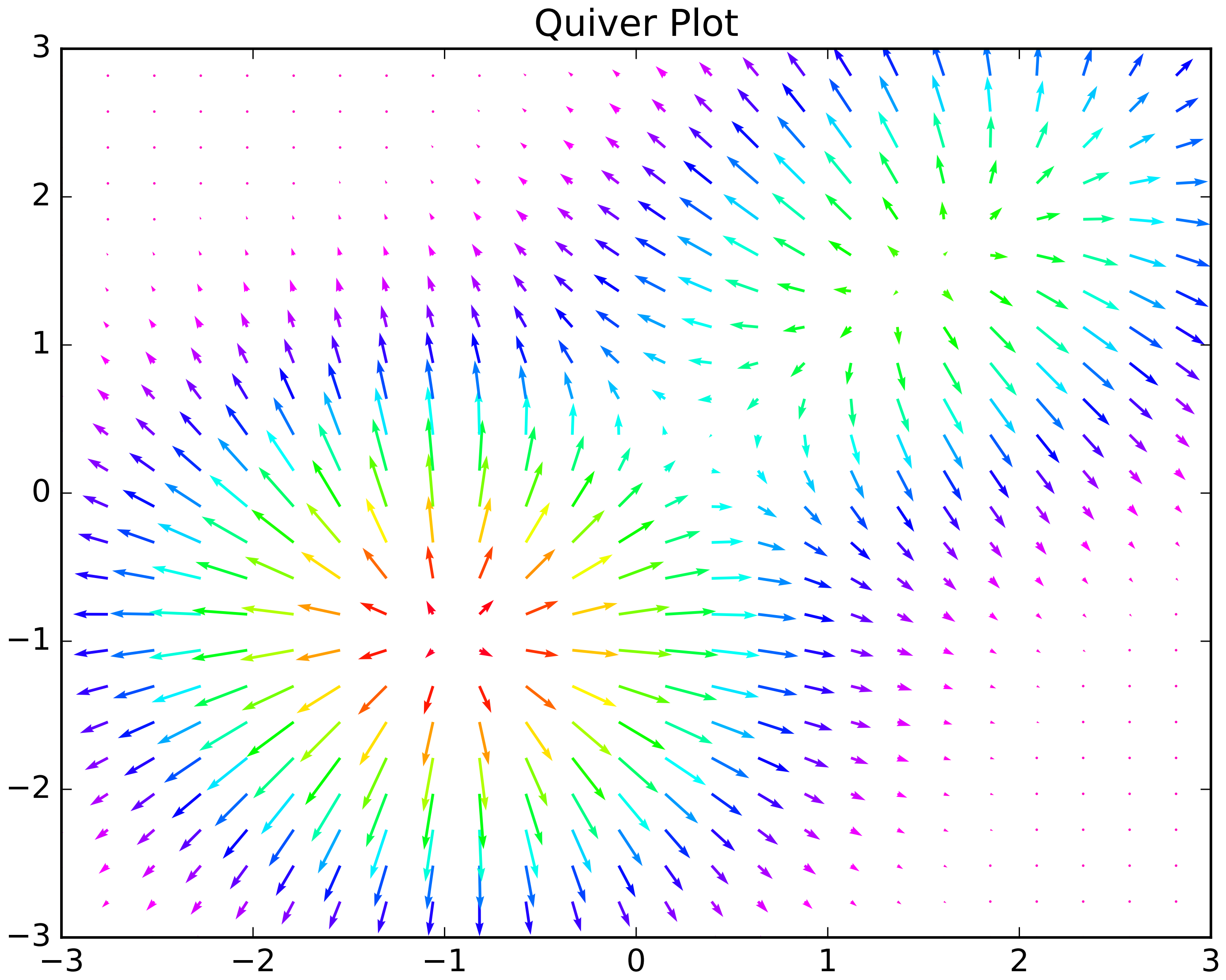
# Gradient



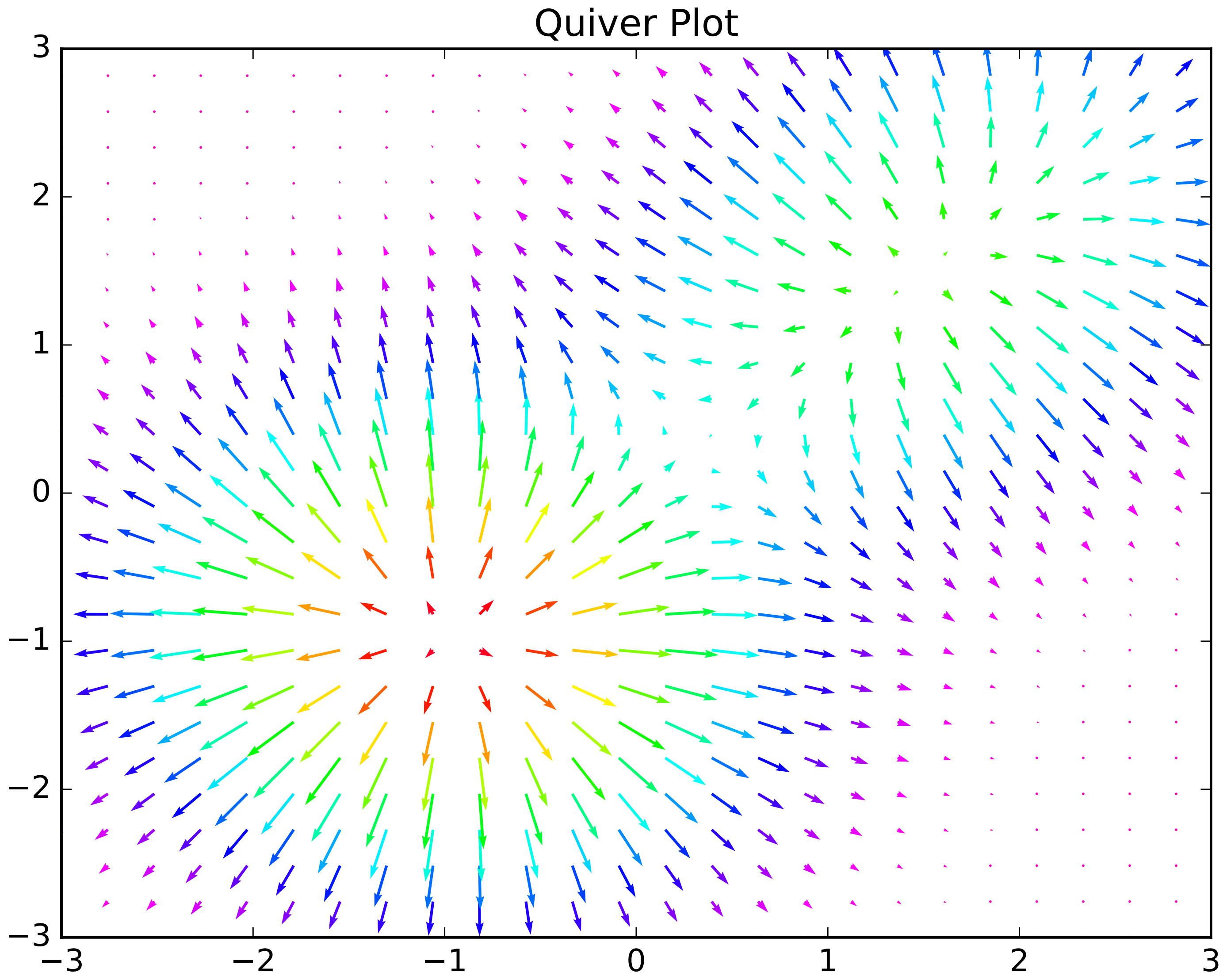
# Gradient



# Gradient

The gradient gives the direction of steepest ascent

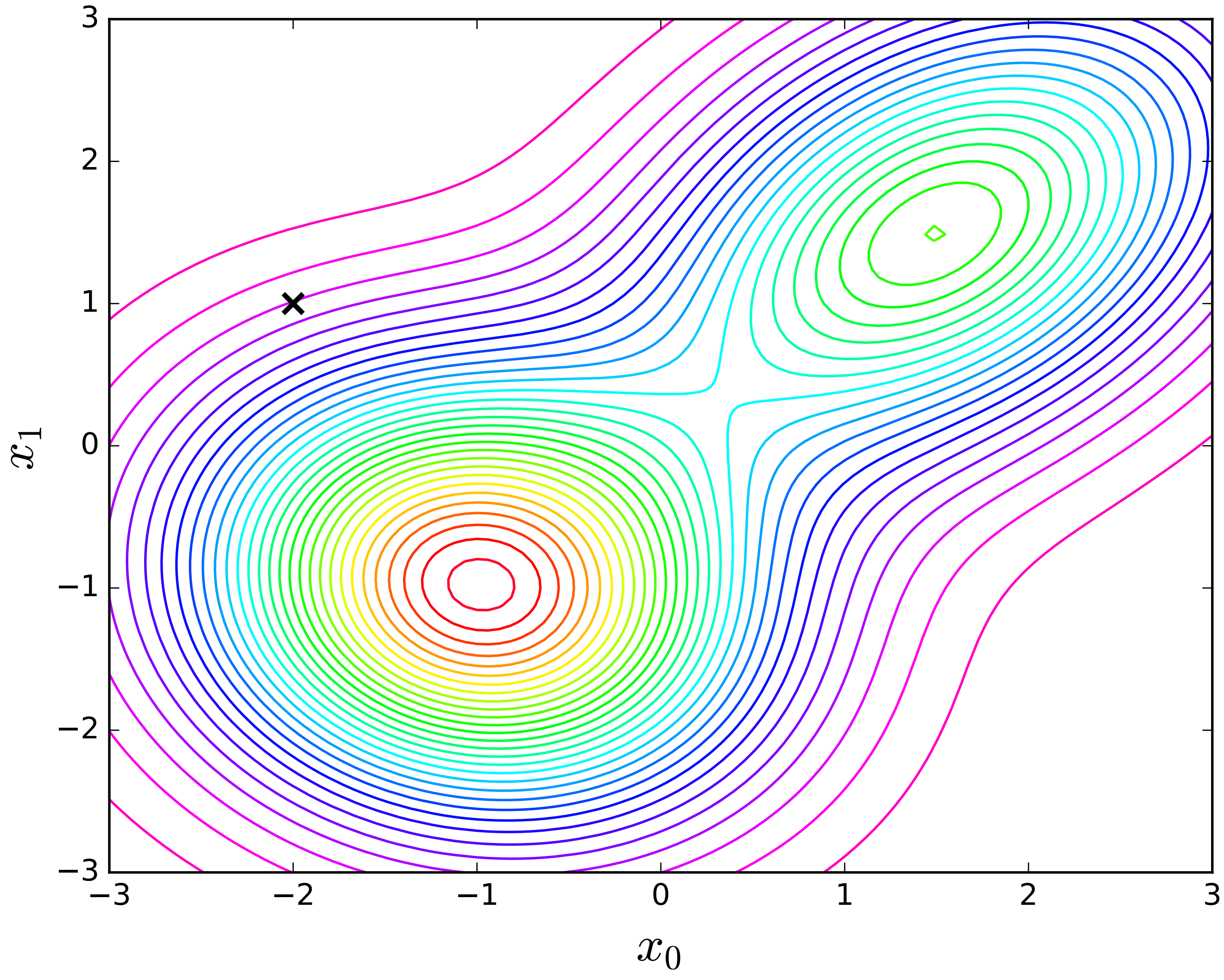


# Gradient Descent: Algorithm

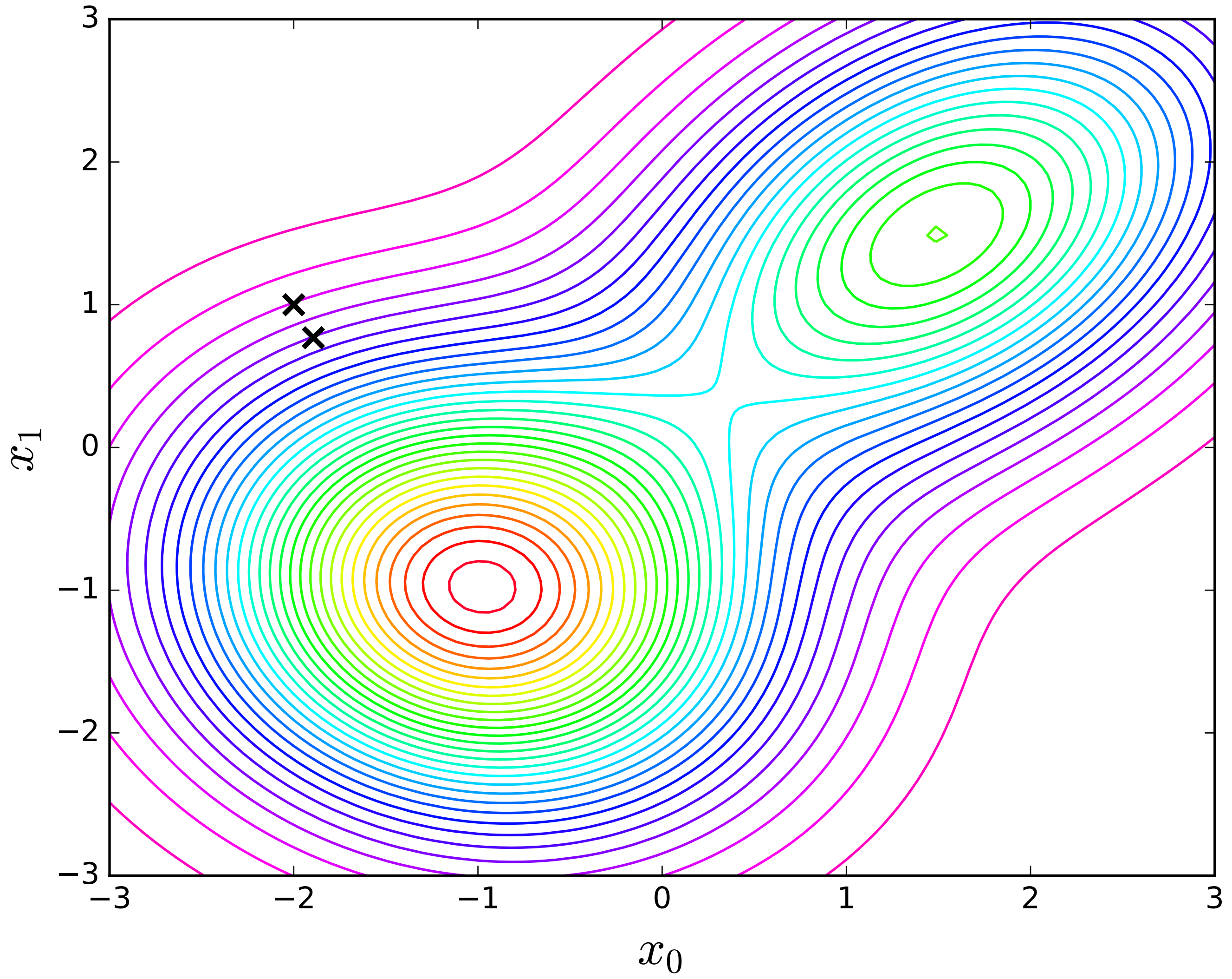
- **Algorithm:**
  1. Set **x** to initial guess
  2. Refine the current value of **x**
  3. If not converged, go back to step 2

- In step 2, follow the negative gradient

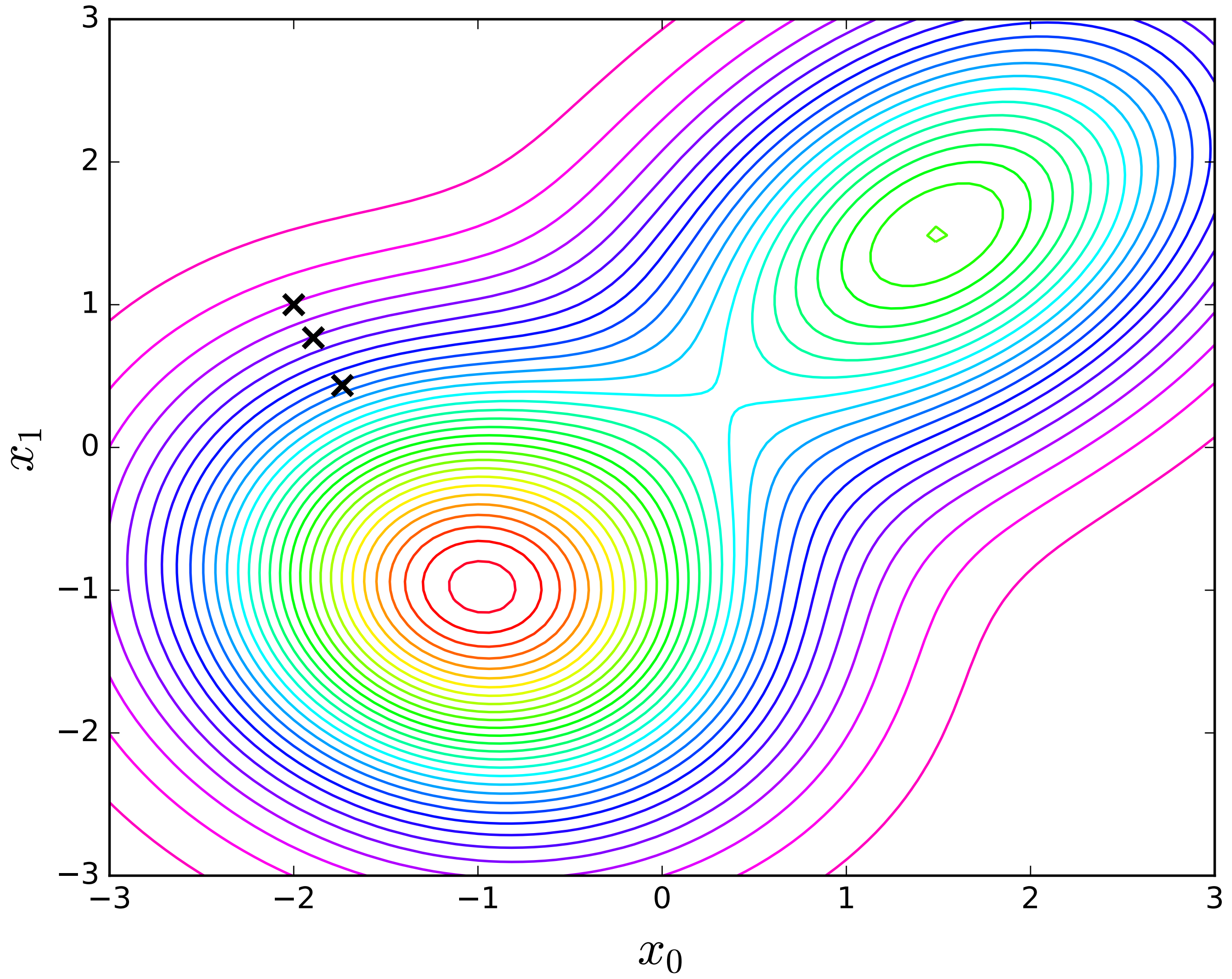
# Gradient Descent: Algorithm



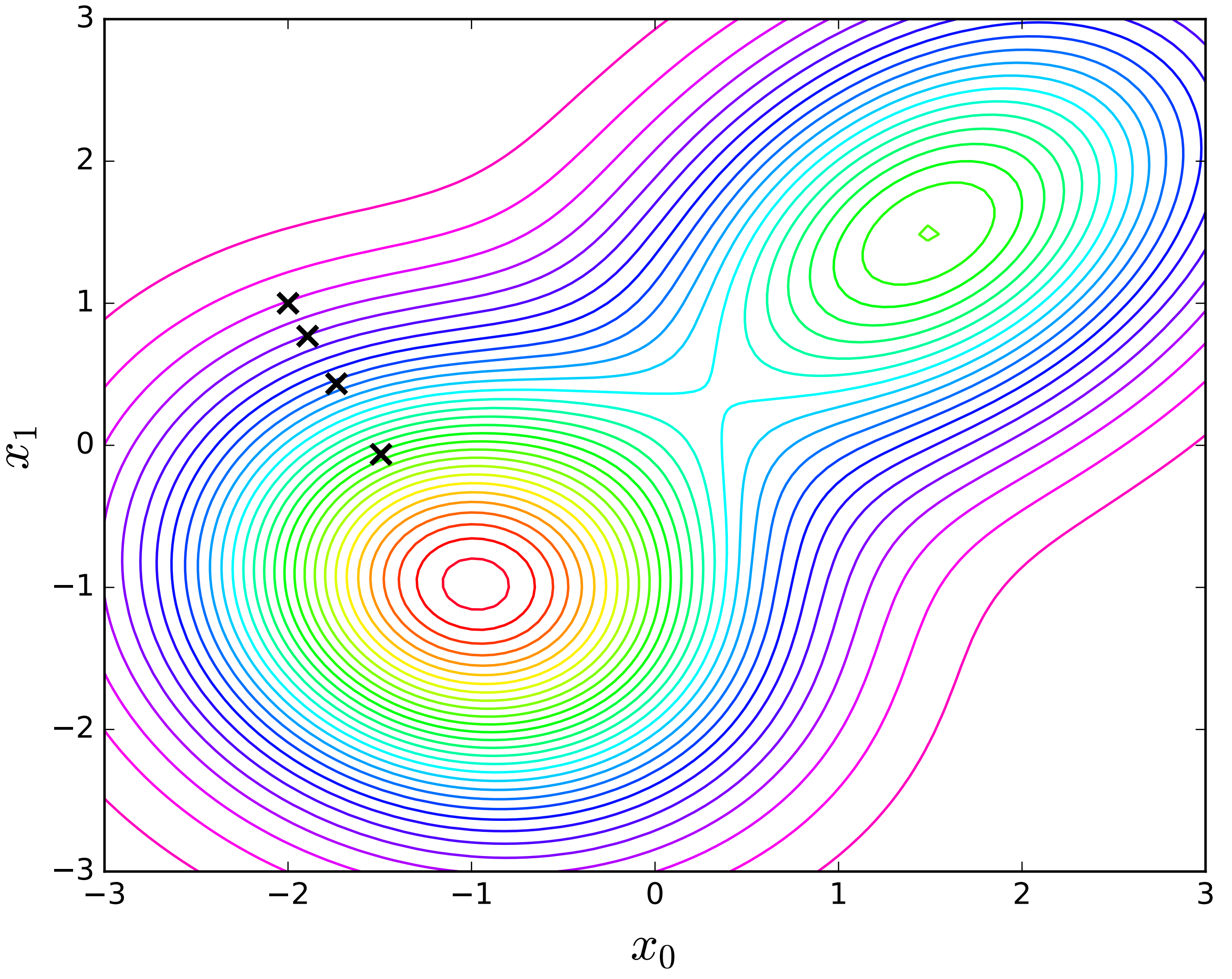
# Gradient Descent: Algorithm



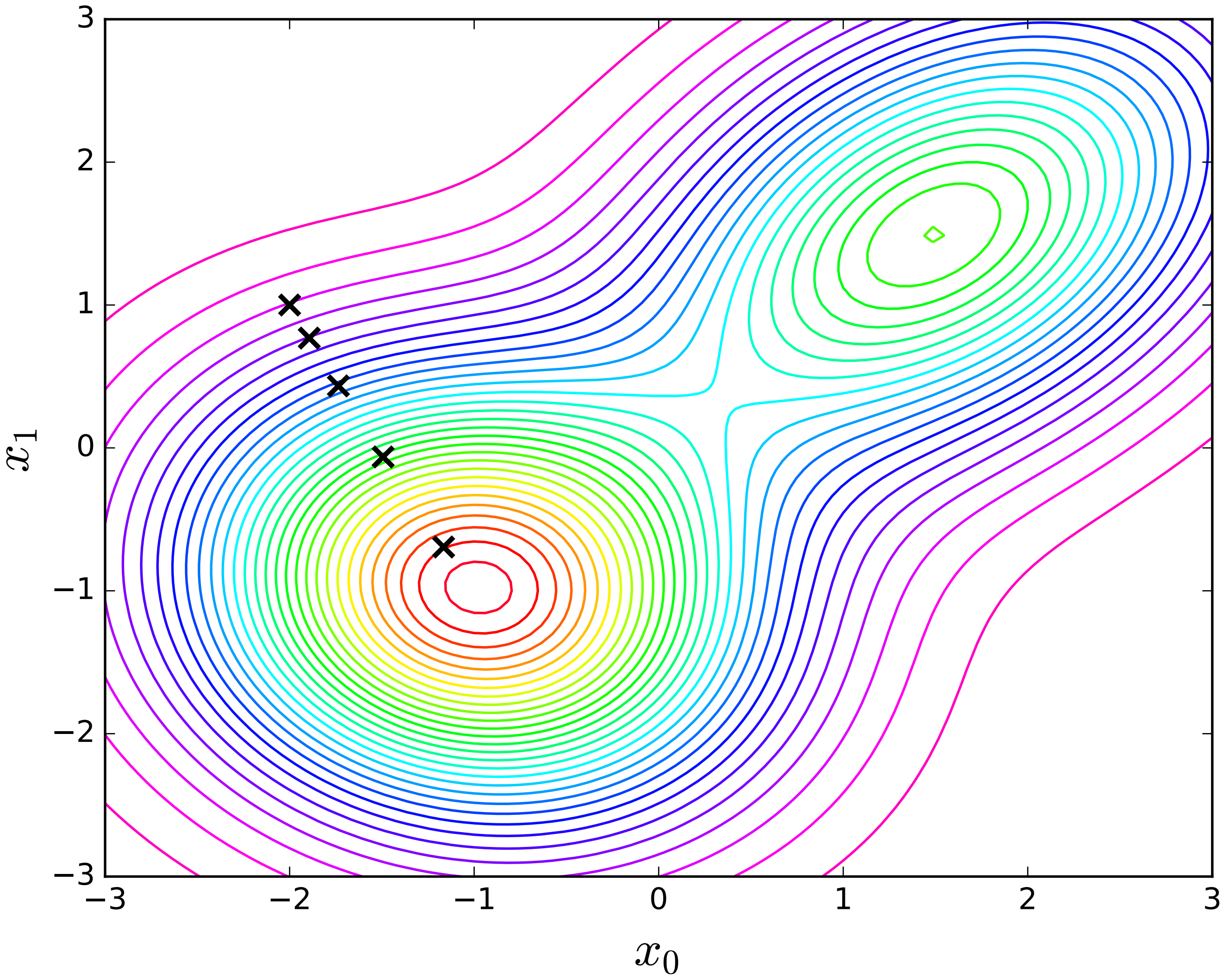
# Gradient Descent: Algorithm



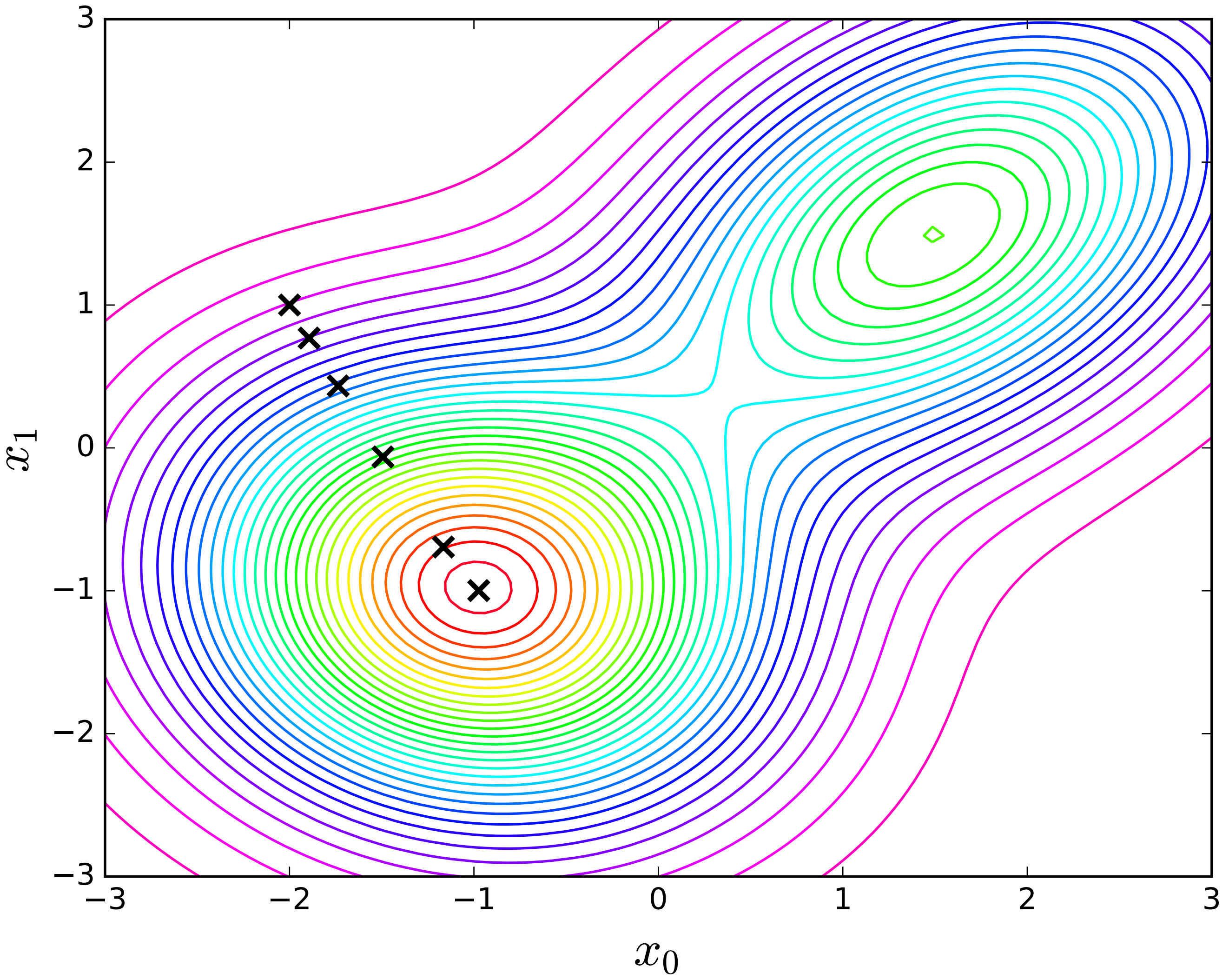
# Gradient Descent: Algorithm



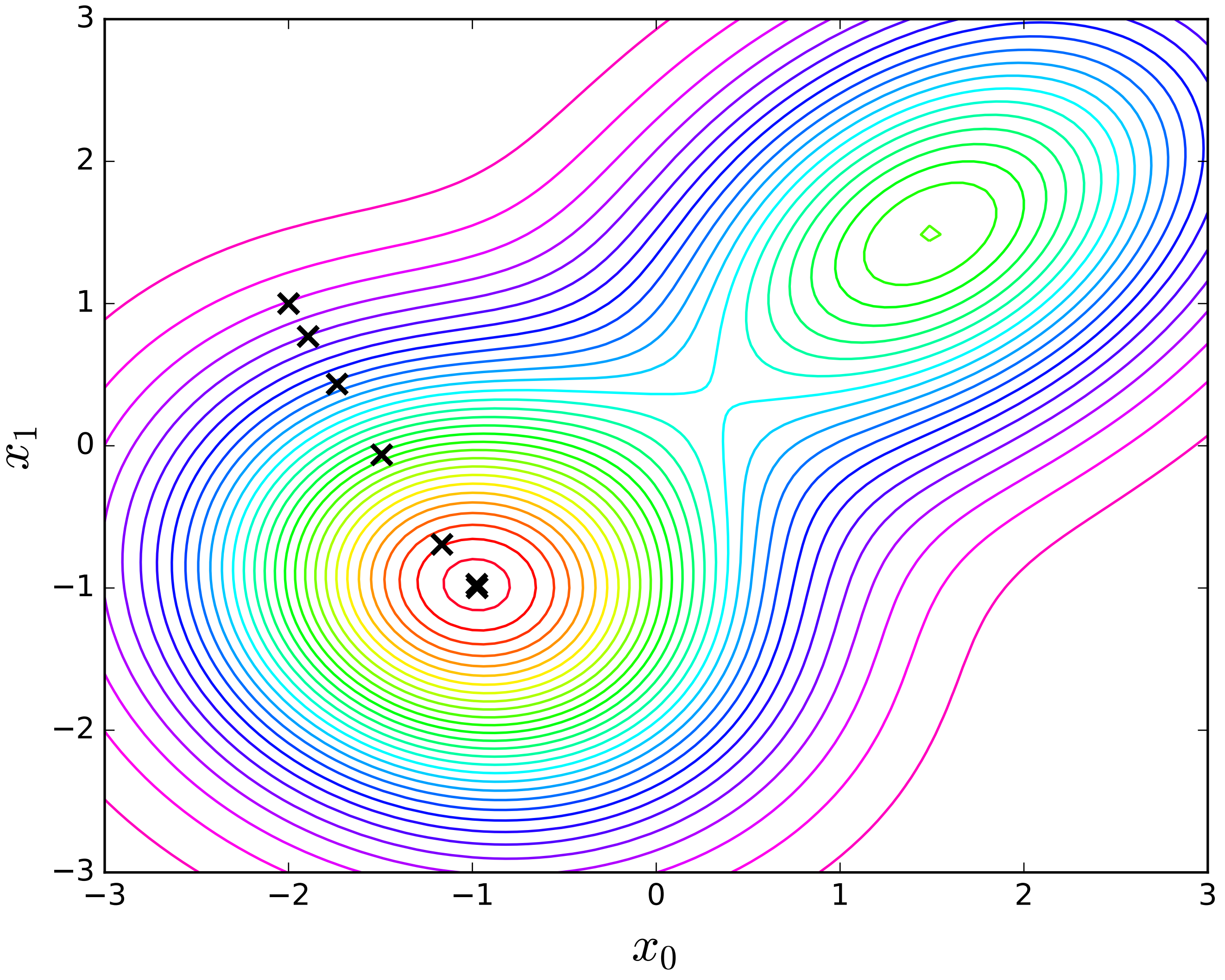
# Gradient Descent: Algorithm



# Gradient Descent: Algorithm



# Gradient Descent: Algorithm



# Gradient Descent: Algorithm

- Each update is:

$$\mathbf{x}^{\text{new}} = \mathbf{x}^{\text{old}} + \rho \cdot \nabla_{\mathbf{x}} f(\mathbf{x}^{\text{old}}) \qquad \text{(Maximization)}$$

$$\mathbf{x}^{\text{new}} = \mathbf{x}^{\text{old}} - \rho \cdot \nabla_{\mathbf{x}} f(\mathbf{x}^{\text{old}}) \qquad \text{(Minimization)}$$

# Gradient Descent: Algorithm

- Each update is:

$$\mathbf{x}^{\text{new}} = \mathbf{x}^{\text{old}} + \rho \cdot \nabla_{\mathbf{x}} f(\mathbf{x}^{\text{old}}) \qquad \text{(Maximization)}$$

$$\mathbf{x}^{\text{new}} = \underbrace{\mathbf{x}^{\text{old}}}_{\text{Current value of } \mathbf{x}} - \underbrace{\rho}_{\text{Step size}} \cdot \underbrace{\nabla_{\mathbf{x}} f(\mathbf{x}^{\text{old}})}_{\text{Gradient}} \qquad \text{(Minimization)}$$

# Convergence

- We stop the algorithm:
  - When **x** does not change much
  - When the gradient is small
  - After a fixed number of iterations

# Limitations

- Not guaranteed to find the global optimum
- Choosing the step size is a nuisance
- Only takes into account gradient information

# Today's Session

- Implement gradient descent
- Explore different step sizes and initial points
- Understand the effect of the step size
- IPython Notebook: *GradientDescent*

COLUMBIA UNIVERSITY
Data Science Institute