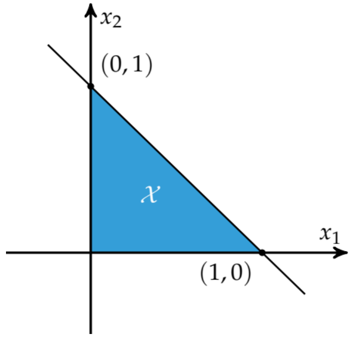
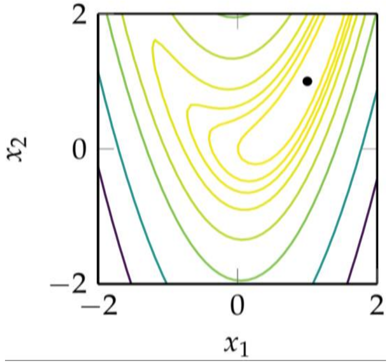
Tecnológico Nacional de México
Instituto Tecnológico de Orizaba/Celaya

Modern Computing Algorithms for Process Optimization with Julia Programming

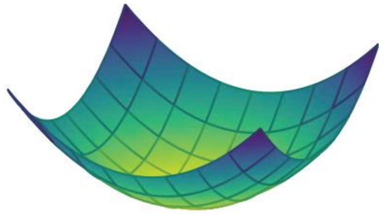


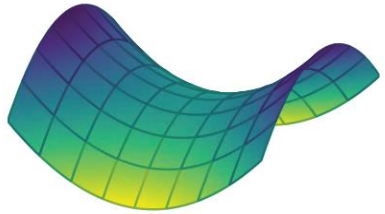


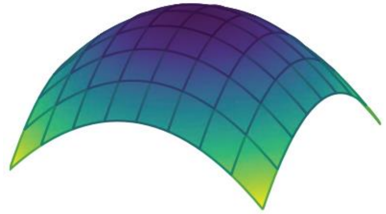
“5 – First-Order Methods”

By:

Dr. Kelvyn Baruc Sánchez Sánchez
Postdoctoral Researcher/I.T. Celaya







Gradient Descent

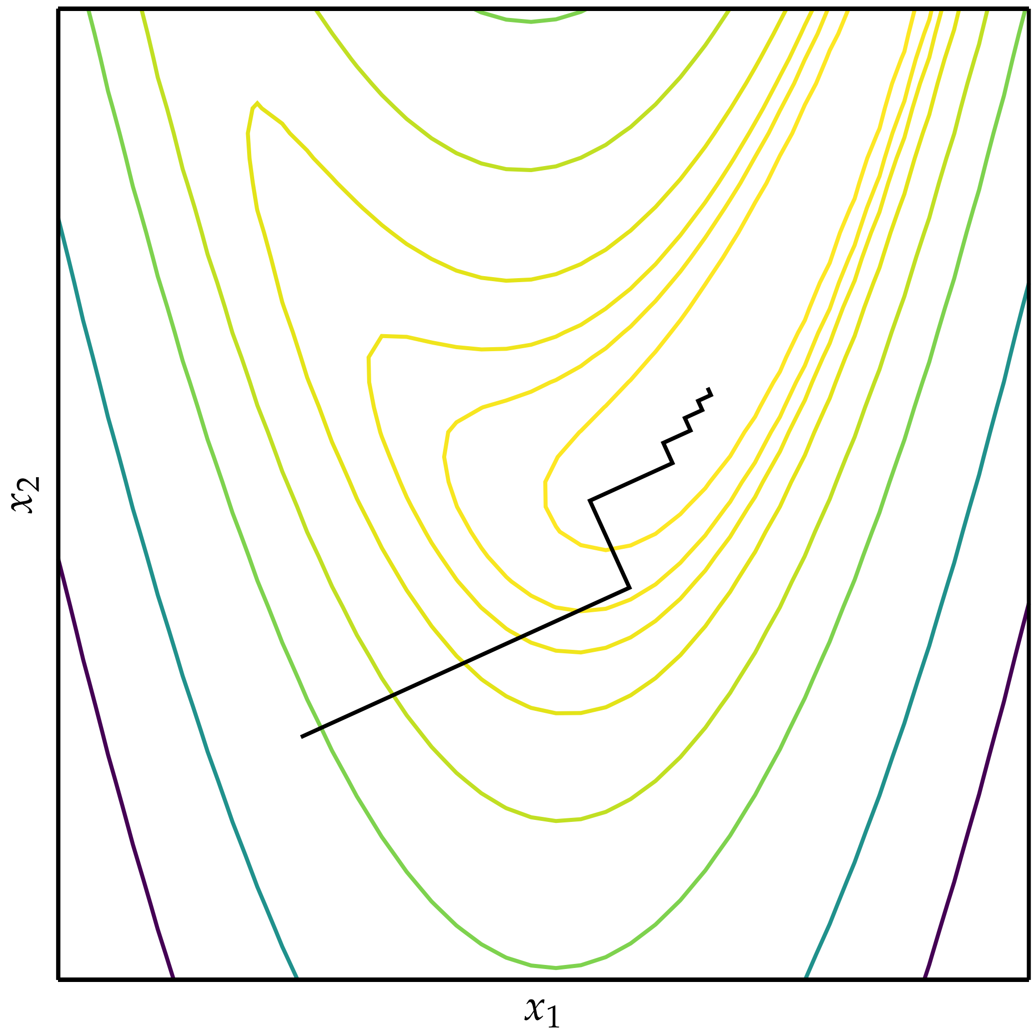
- Search in direction of steepest descent

$$\mathbf{g}^{(k)} = \nabla f(\mathbf{x}^{(k)})$$

$$\mathbf{d}^{(k)} = -\frac{\mathbf{g}^{(k)}}{\|\mathbf{g}^{(k)}\|}$$

Gradient Descent

- Applying gradient descent and line search on Rosenbrock function



See example 5.1.ipynb

Conjugate Gradient

- Inspired by methods for optimizing quadratic functions
- Assumes problem takes the quadratic form

$$\underset{\mathbf{x}}{\text{minimize}}\, f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c$$

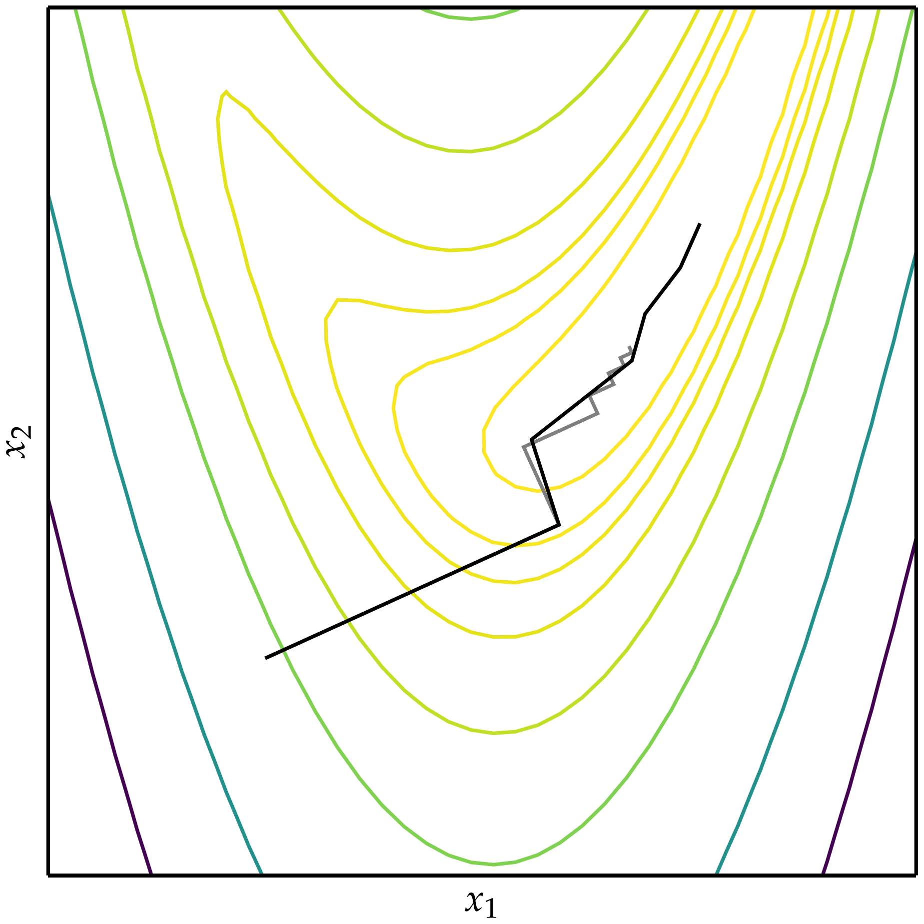
where **A** is positive-definite matrix

- All search directions are mutually conjugate with respect to **A**
- For n-dimensional problem, converges in n steps
- Can be used when function is locally approximated as quadratic

Conjugate Gradient

$$\mathbf{d}^{(k+1)} = -\mathbf{g}^{(k+1)} + \beta^{(k)}\mathbf{d}^{(k)}$$

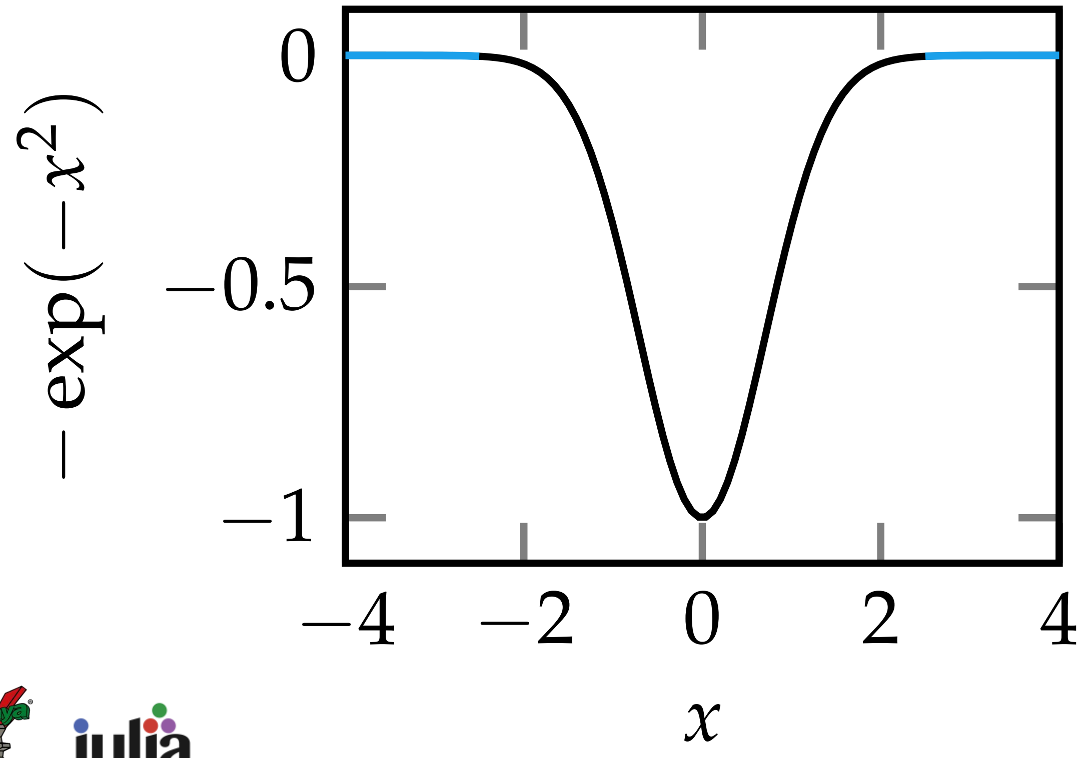
- Using Fletcher-Reeves or Polak-Ribière equations for $\beta^{(k)}$, a function can be locally approximated as quadratic and minimized using Conjugate Gradient



See example 5.2.ipynb

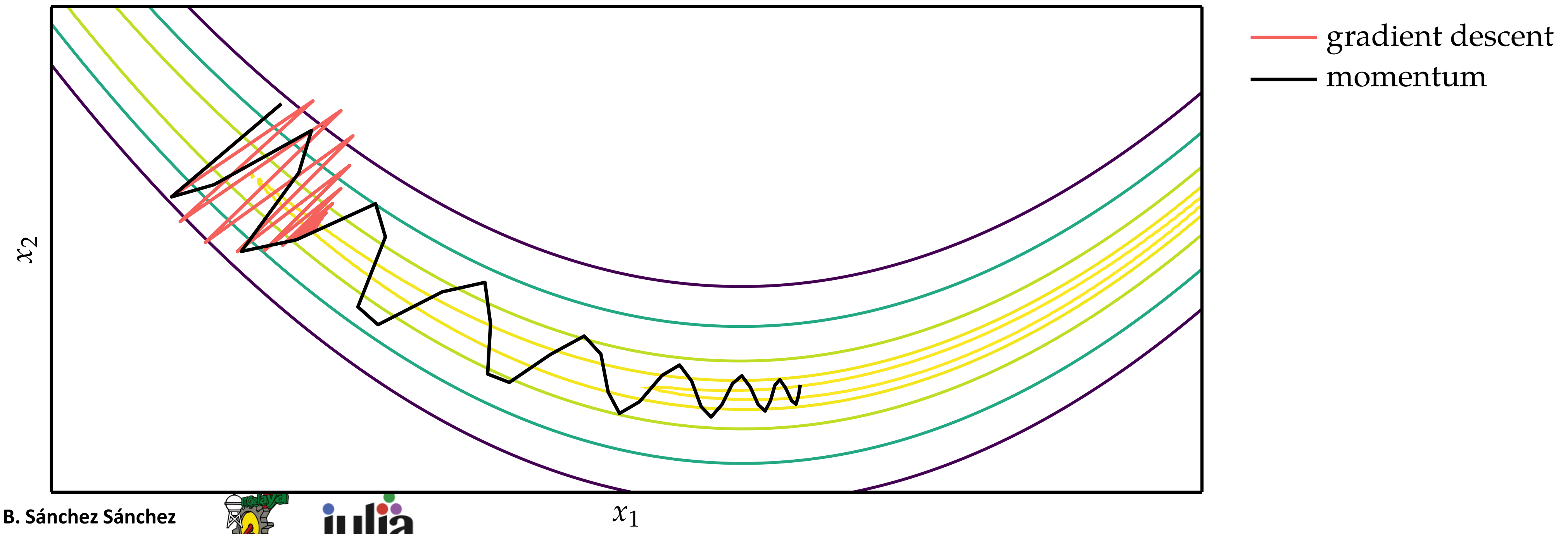
Momentum

- Addresses common convergence issues
- Some functions have regions with very small gradients



Momentum

- Some functions cause gradient descent to get stuck
- Momentum overcomes these issues by replicating the effect of physical momentum



Momentum

- Momentum update equations

$$\mathbf{v}^{(\mathrm{k}+1)} = \beta \mathbf{v}^{(\mathrm{k})} - \alpha \mathbf{g}^{(\mathrm{k})}$$

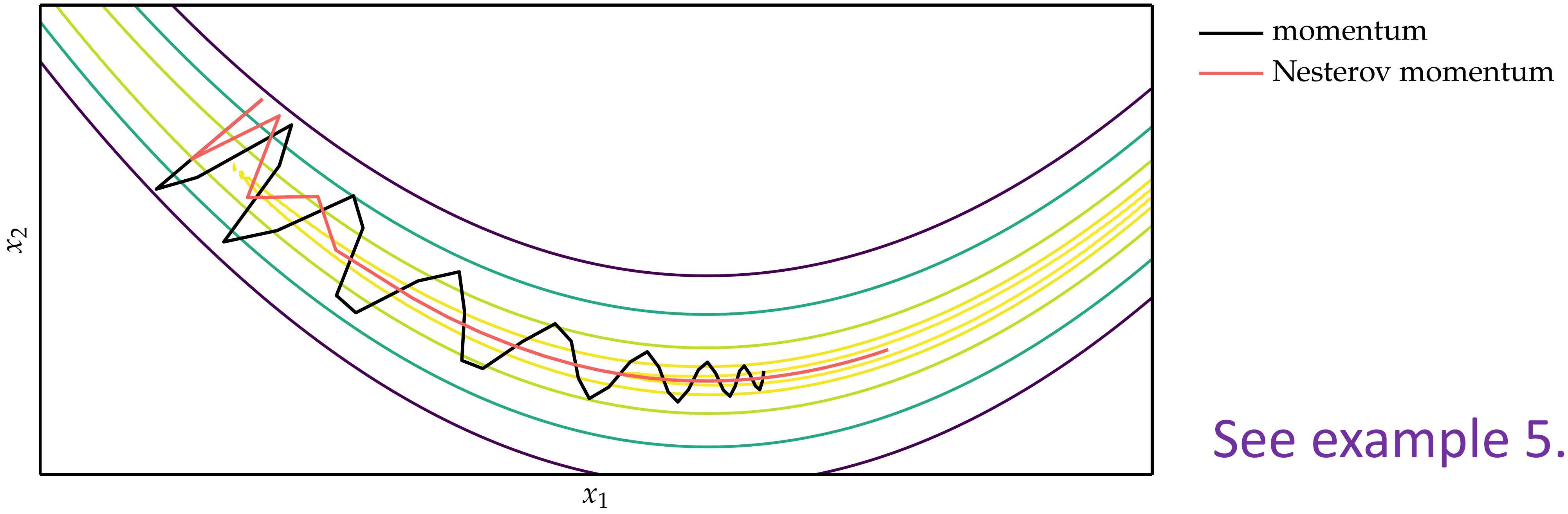
$$\mathbf{x}^{(\mathrm{k}+1)} = \mathbf{x}^{(\mathrm{k})} + \mathbf{v}^{(\mathrm{k}+1)}$$

See example 5.3.ipynb

Nesterov Momentum

$$\mathbf{v}^{(k+1)} = \beta\mathbf{v}^{(k)} - \alpha\nabla f\left(\mathbf{x}^{(k)} + \beta\mathbf{v}^{(k)}\right)$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{v}^{(k+1)}$$



See example 5.4.ipynb

Dr. Kelvyn B. Sánchez Sánchez
Instituto Tecnológico de Celaya

Adagrad

- Instead of using the same learning rate for all components of **x**, Adaptive Subgradient method (Adagrad) adapts the learning rate for each component of **x**.
- For each component of x, the update equation is

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\alpha}{\epsilon + \sqrt{s_i^{(k)}}} g_i^{(k)}$$

where

$$s_i^{(k)} = \sum_{j=1}^{k} \left(g_i^{(j)}\right)^2 \qquad \epsilon \approx 1 \times 10^{-8}$$

See example 5.5.ipynb

RMSProp

- Extends Adagrad to avoid monotonically decreasing learning rate by maintaining a decaying average of squared gradients

$$\hat{\mathbf{s}}^{(k+1)} = \gamma\hat{\mathbf{s}}^{(k)} + (1-\gamma)\left(\mathbf{g}^{(k)} \odot \mathbf{g}^{(k)}\right)$$

- Update Equation

$$\begin{aligned} \mathrm{x}_\mathrm{i}^{(\mathrm{k}+1)} &= x_\mathrm{i}^{(\mathrm{k})} - \frac{\alpha}{\epsilon + \sqrt{\hat{s}_i^{(k)}}} g_i^{(k)} \\ &= x_\mathrm{i}^{(\mathrm{k})} - \frac{\alpha}{\epsilon + \mathrm{RMS}(g_i)} g_i^{(k)} \end{aligned}$$

See example 5.6ipynb

Adadelta

- Modifies RMSProp to eliminate learning rate parameter entirely

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\mathrm{RMS}(\Delta x_i)}{\epsilon + \mathrm{RMS}(g_i)} g_i^{(k)}$$

See example 5.7.ipynb

Hypergradient Descent

- Many accelerated descent methods are highly sensitive to hyperparameters such as learning rate.
- Applying gradient descent to a hyperparameter of an underlying descent method is called hypergradient descent
- Requires computing the partial derivative of the objective function with respect to the hyperparameter

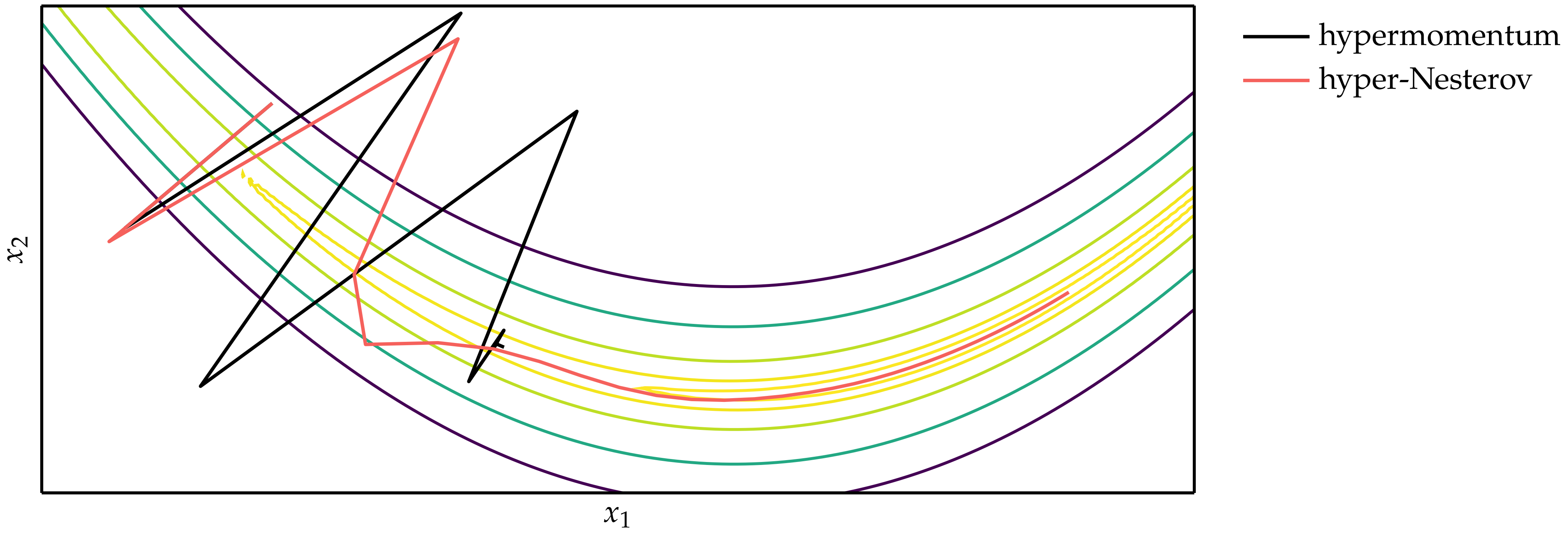
Hypergradient Descent

$$\begin{aligned}\frac{\partial f(\mathbf{x}^{(k)})}{\partial \alpha} &= (\mathbf{g}^{(k)})^\top \frac{\partial}{\partial \alpha}\left(\mathbf{x}^{(k-1)} - \alpha \mathbf{g}^{(k-1)}\right) \\ &= (\mathbf{g}^{(k)})^\top \left(-\mathbf{g}^{(k-1)}\right)\end{aligned}$$

$$\begin{aligned}\alpha^{(k+1)} &= \alpha^{(k)} - \mu \frac{\partial f(\mathbf{x}^{(k)})}{\partial \alpha} \\ &= \alpha^{(k)} + \mu (\mathbf{g}^{(k)})^\top \mathbf{g}^{(k-1)}\end{aligned}$$

See example 5.8 .ipynb

Hypergradient Descent



Summary

- Gradient descent follows the direction of steepest descent.
- The conjugate gradient method can automatically adjust to local valleys.
- Descent methods with momentum build up progress in favorable directions.
- A wide variety of accelerated descent methods use special techniques to speed up descent.
- Hypergradient descent applies gradient descent to the learning rate of an underlying descent method.