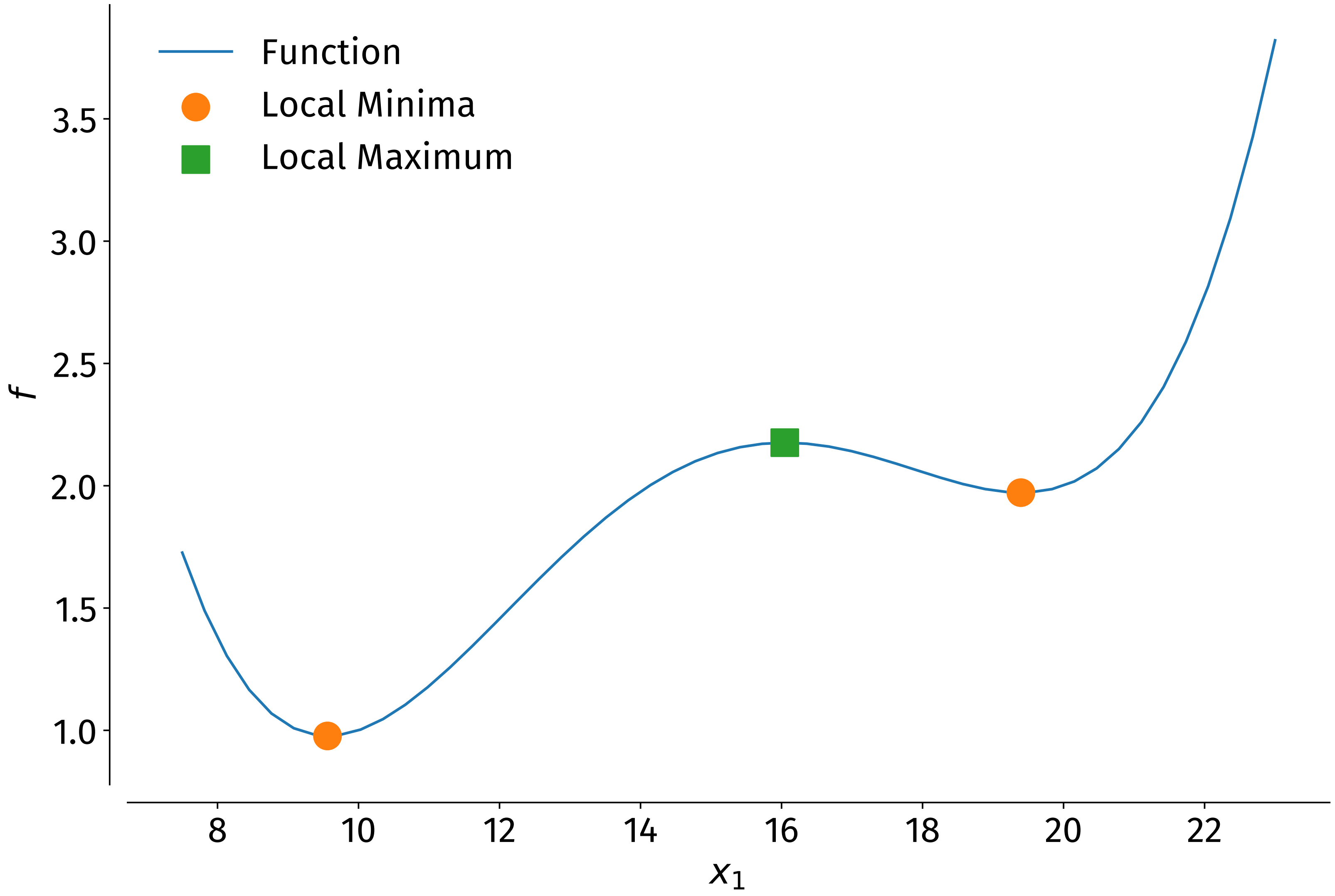
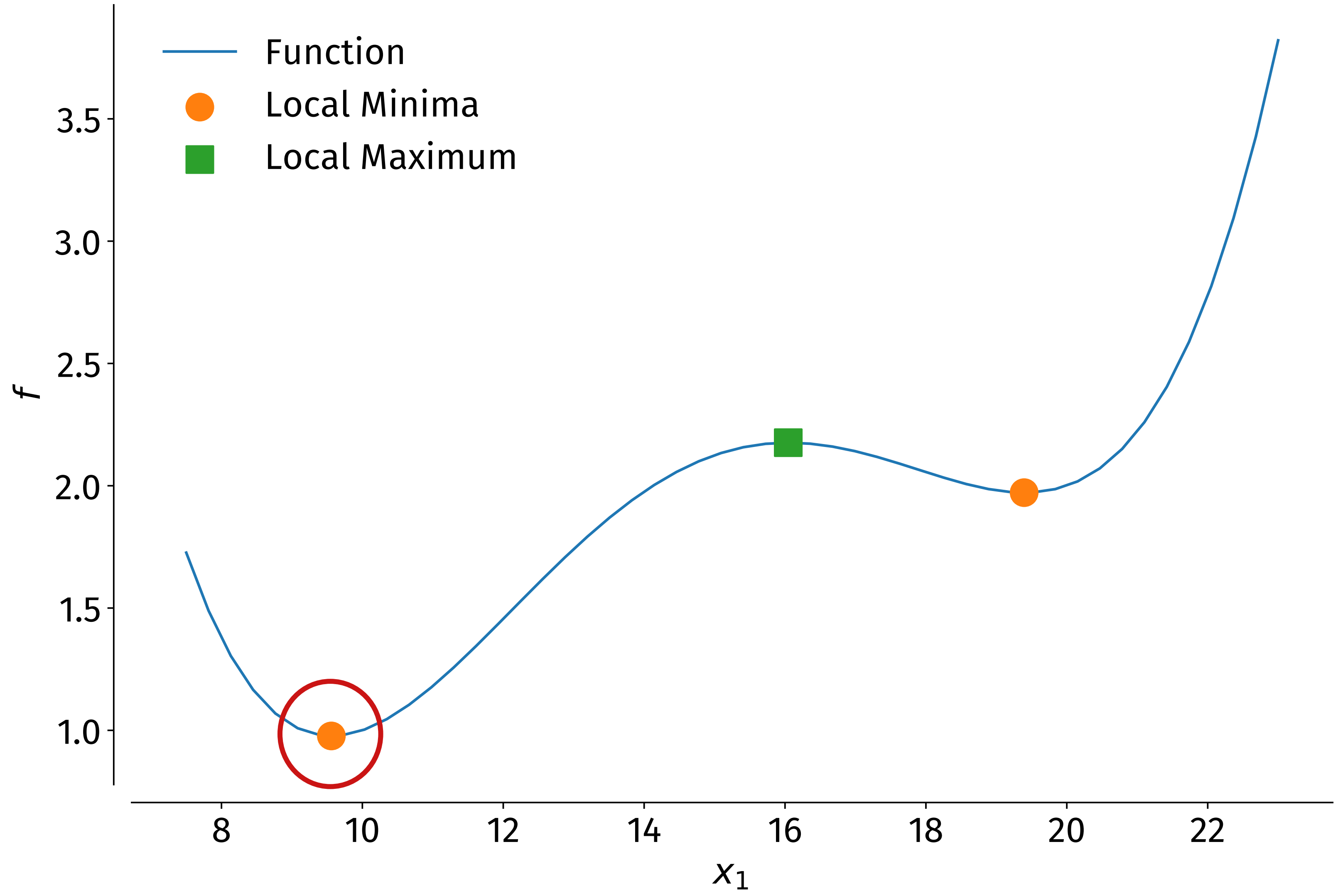
Local minimum

$$\exists \epsilon > 0 : \forall y \in [x_0 - \epsilon, x_0 + \epsilon] : f(x_0) \leq f(y)$$

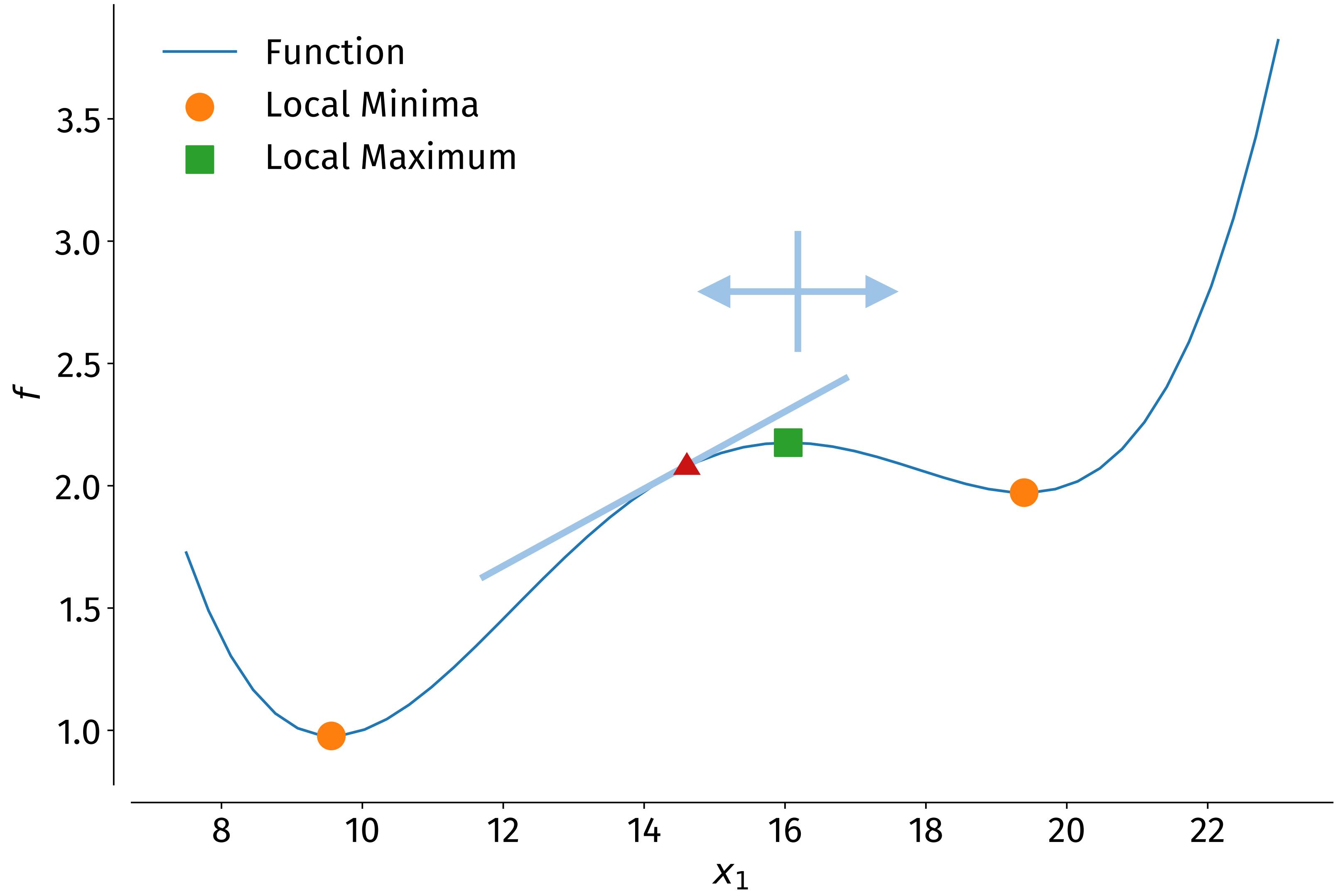


Global minimum

$$\forall y \in X : f(x_0) \leq f(y)$$

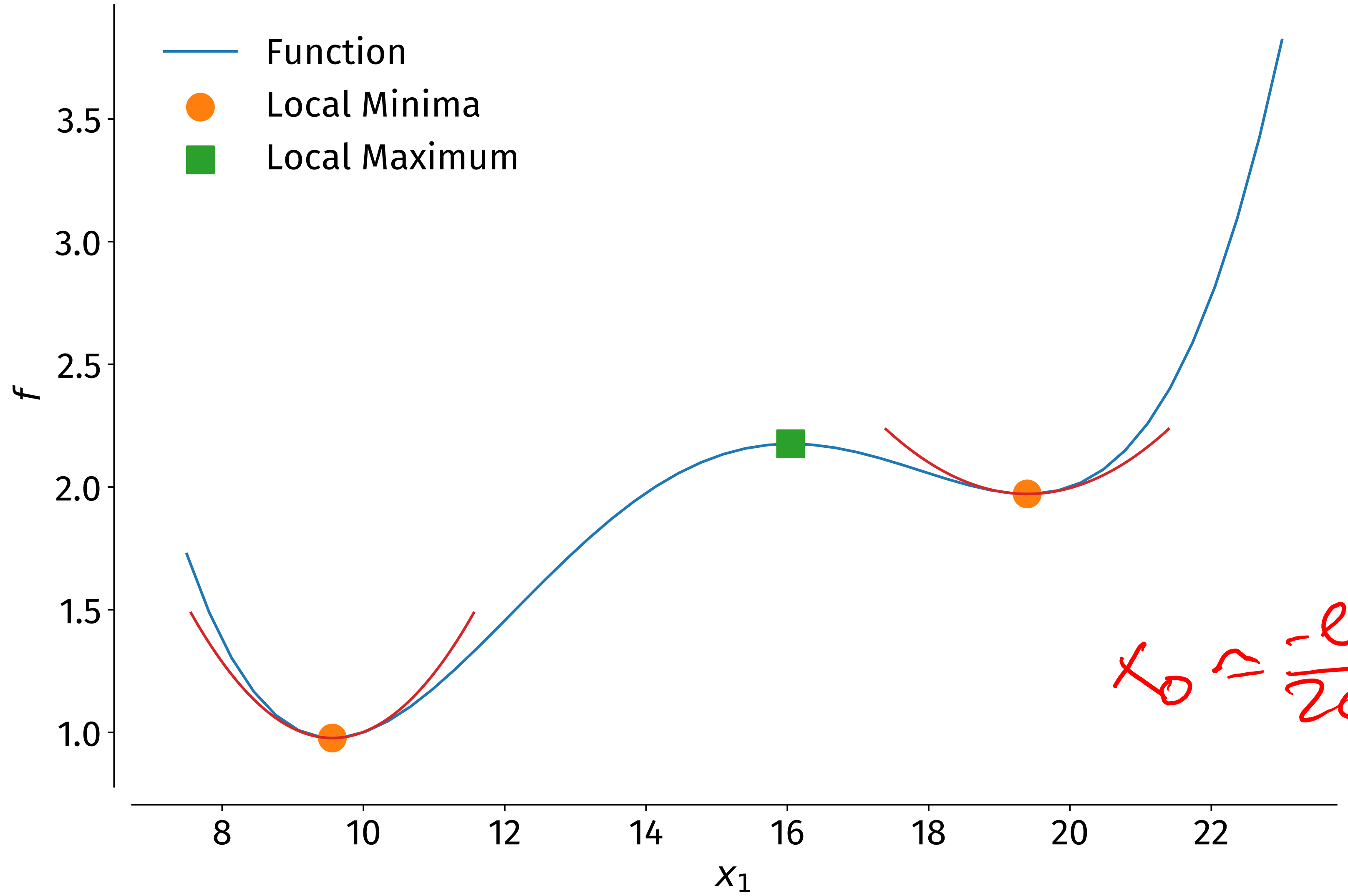


Definition: Attractive basin

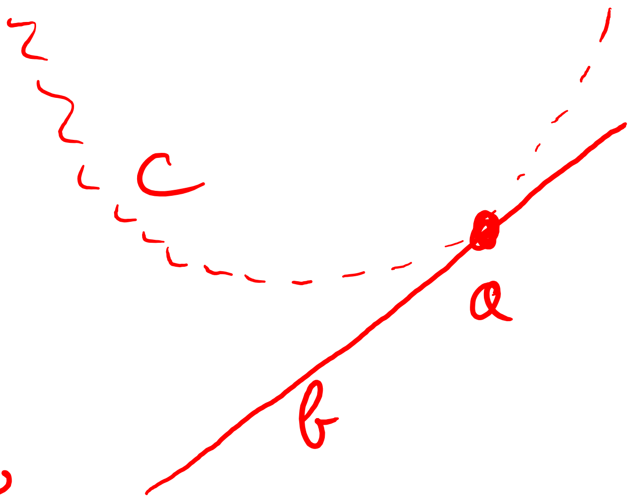


All values that if the gradient is followed reach a given minimum.

Definition: Quadratic region



All values where a Taylor series expansion up to second order around the minimum is a “good” approximation.

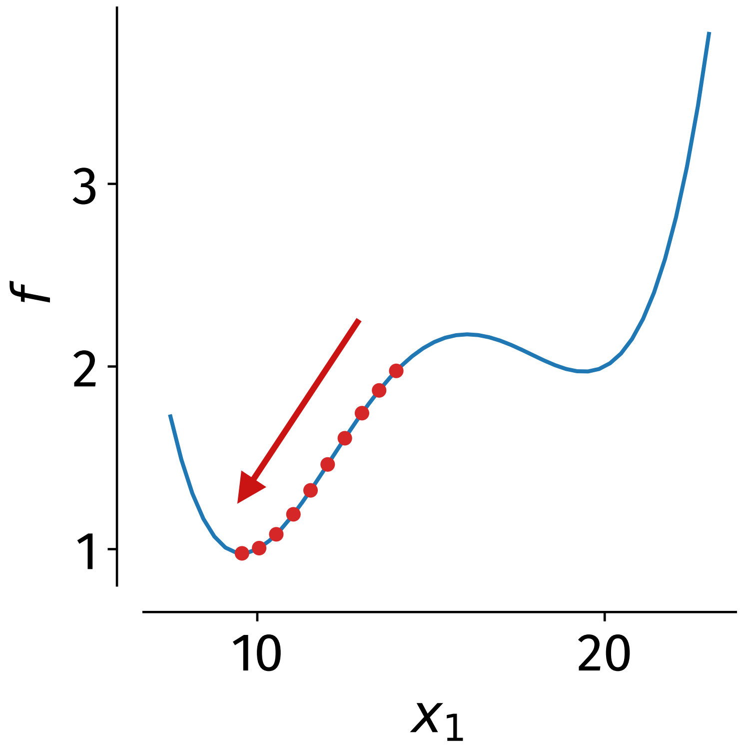


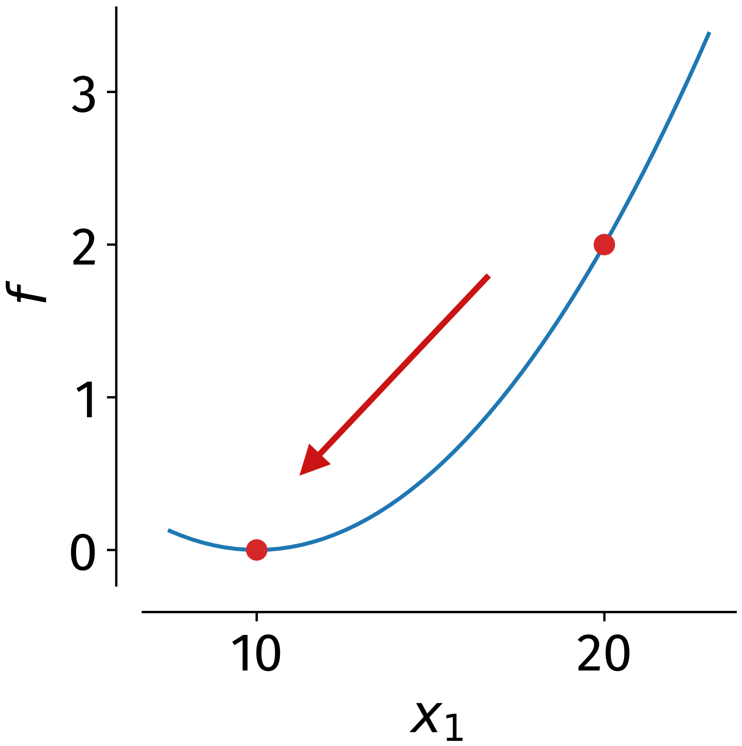
$$x_0 \simeq \frac{-b}{2c}$$

$$f \sim a + bx + cx^2$$

$$f' \sim b + 2cx$$

Definition: Iterative vs direct





Iterative

- Edging closer to the minimum
- Continue until close enough

Direct

- One-step optimization
- Analytical expression
- Note: “direct method” = no gradients

Optimization strategies

Pure strategies:

- Follow gradient and/or Hessian — (Quasi-)Newton methods
  - Transition states: eigenmode following
- Reduce dimensionality — Subspace methods
- (Quasi-)randomly pick points — Stochastic optimisation
- Regularly pick points — Grid refinement
- …

Hybrid:

- Problem specific
- Typically global optimization
  - E.g. stochastic first, then Newton

Series notation for iterative approaches:

$$\{a_n\} \qquad \lim_{n\to\infty} = x_0$$

Optimization Algorithms

Follow gradient and/or Hessian

When to use

- Local minima
- Reasonable initial guess
- Wide attractive basins

When not to use

- Noisy function evaluations
- High dimensionality

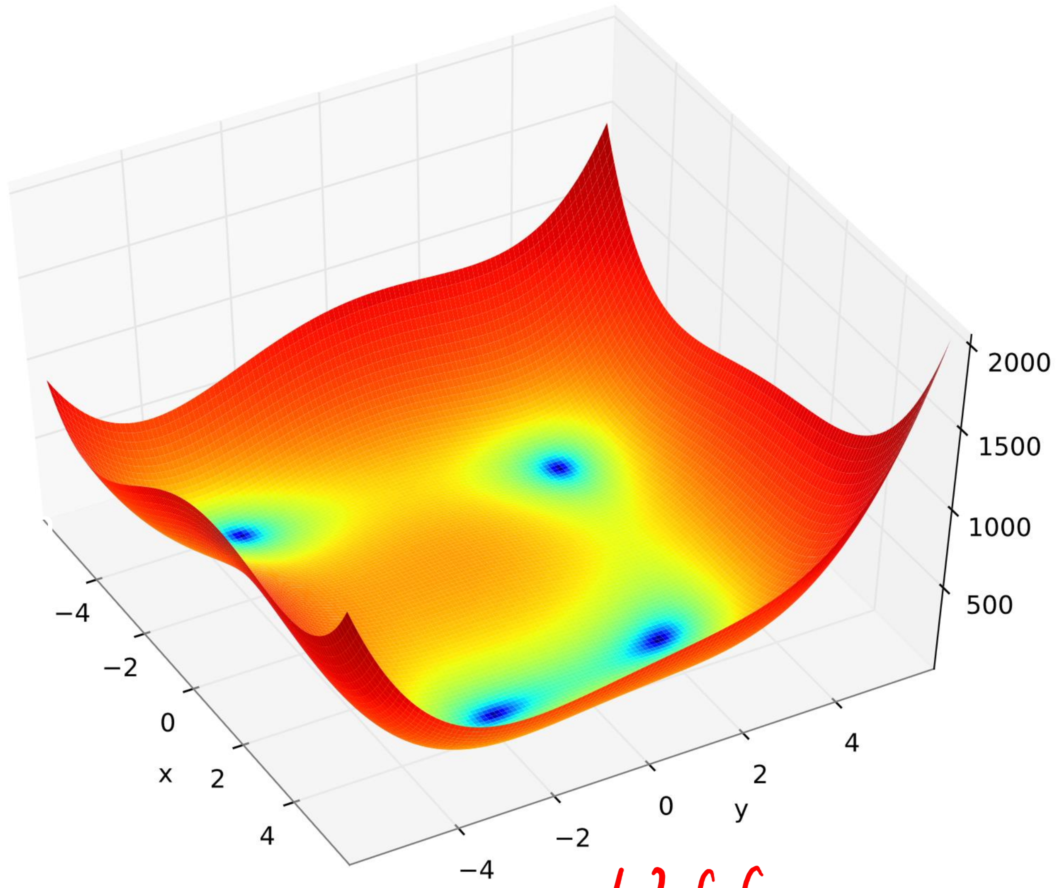
Popular representatives

- Newton
- Steepest descent
- BFGS

  ```
  scipy.optimize.minimize(method='BFGS')
  ```

- L-BFGS

  ```
  scipy.optimize.minimize(method='L-BFGS-B')
  ```



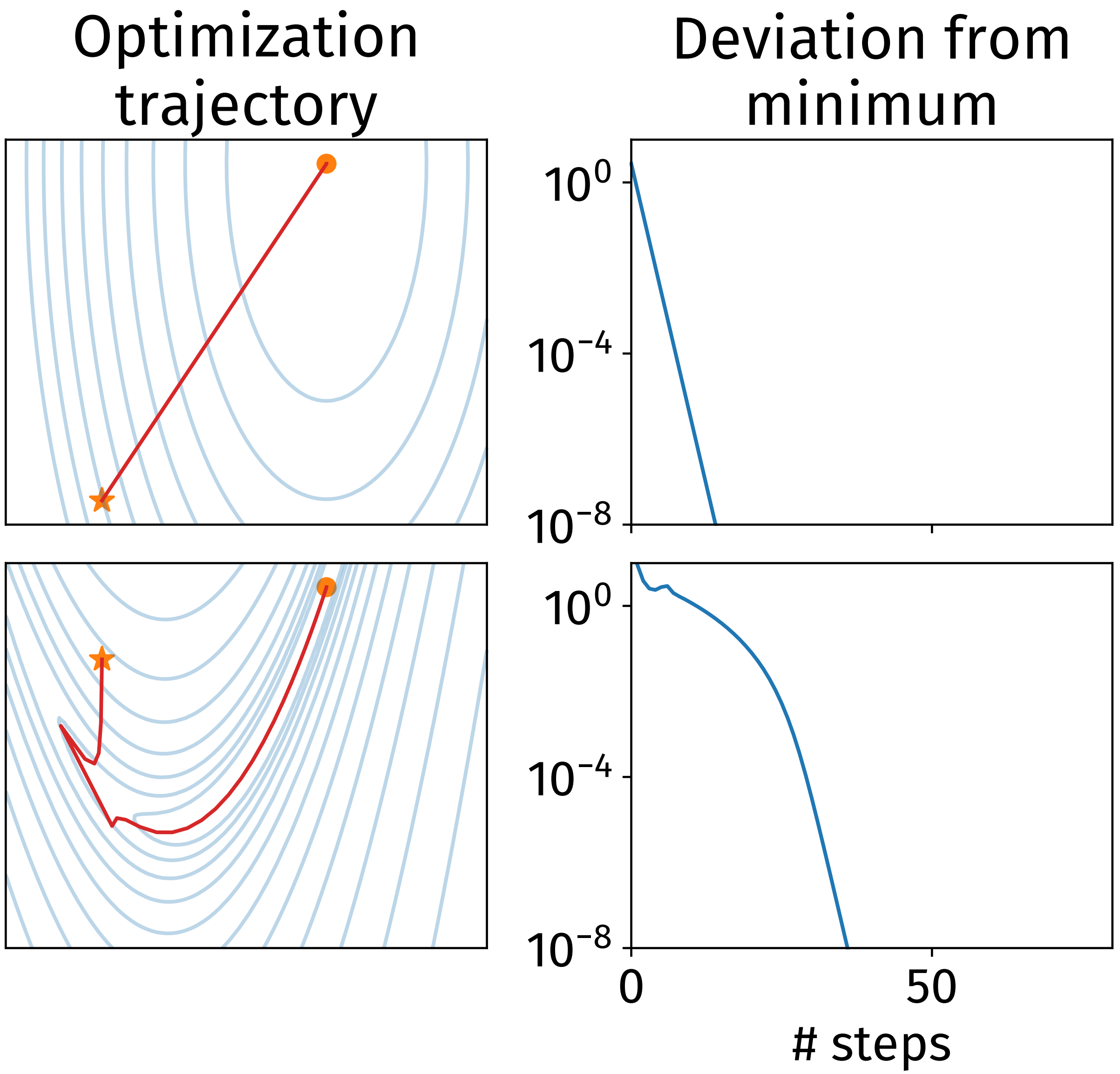
Wolfe's cond

Newton’s method

$$a_n - s\left[\nabla^2 f(a_n)\right]^{-1}\nabla f(a_n)$$

Variants
- Scale step size s
- Stochastic Newton

Problems
- Large Hessian and inversion expensive
- Slow with a fixed step

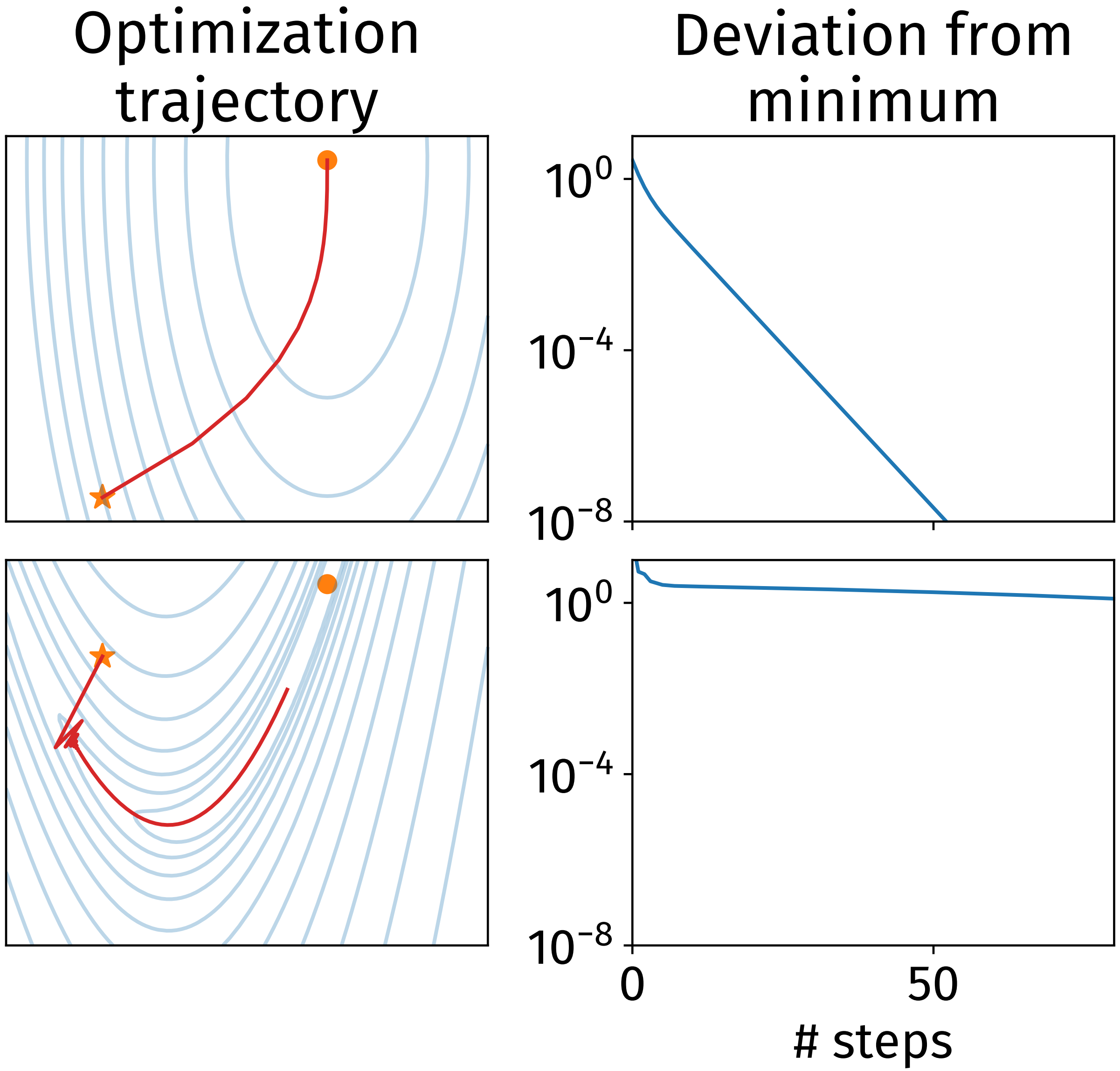


Steepest descent

$$a_n - s\nabla f(a_n)$$

Variants
- Adjust step size
- Line search

Problems
- Slow with fixed step
- Oscillations



BFGS

Like Newton's method

$$p_{n+1} = -B_n^{-1} \nabla f(a_n)$$

Line search

$$\alpha_{n+1} = \arg\min f(a_n + \alpha p_{n+1}) \qquad s_{n+1} = \alpha_{n+1} p_{n+1}$$

Update optimisation

$$a_{n+1} = a_n + s_{n+1}$$

Get gradient response

$$y_{n+1} = \nabla f(a_{n+1}) - \nabla f(a_n)$$

Update approximate Hessian

$$B_{n+1} = B_n + \frac{y_{n+1} y_{n+1}^{\mathrm{T}}}{y_{n+1}^{\mathrm{T}} s_{n+1}} - \frac{B_n s_{n+1} s_{n+1}^{\mathrm{T}} B_n^{\mathrm{T}}}{s_{n+1}^{\mathrm{T}} B_n s_{n+1}}$$

BFGS

Newton with approximate Hessian

Variants

- L-BFGS keeping only subset of Hessian

Problems

- Approximate Hessian update expensive
- High memory requirements