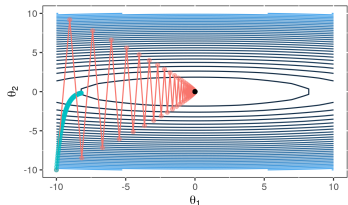


A 3x3 grid with a blue path starting at the top-left cell (0,0) and ending at the bottom-right cell (2,2). The path moves right to (0,1), then down to (1,1), then right to (1,2), and finally down to (2,2). The cells (0,1), (1,1), and (2,2) contain a grey circle, while the other cells contain a grey 'X'.

Step size and optimality



- Impact of step size
- Fixed vs. adaptive step size
- Exact line search
- Armijo rule & Backtracking
- Bracketing & Pinpointing

- Impact of step size
- Fixed vs. adaptive step size
- Exact line search
- Armijo rule & Backtracking
- Bracketing & Pinpointing

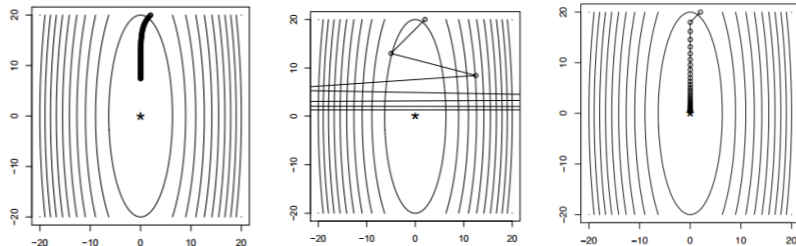
CONTROLLING STEP SIZE: FIXED & ADAPTIVE

Iteration t : Choose not only descent direction $\mathbf{d}^{[t]}$, but also step size $\alpha^{[t]}$

First approach: **Fixed** step size $\alpha^{[t]} = \alpha > 0$

- If α too small, procedure may converge very slowly (left)
- If α too large, procedure may not converge \rightarrow “jumps” around optimum (middle)

Adaptive step size $\alpha^{[t]}$ can provide better convergence (right)



Steps of line searches for $f(\mathbf{x}) = 10x_1^2 + x_2^2/2$



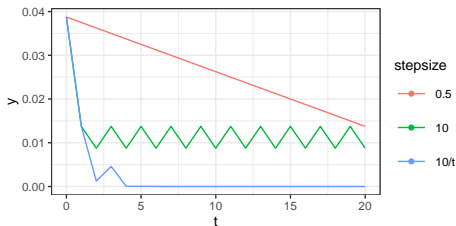
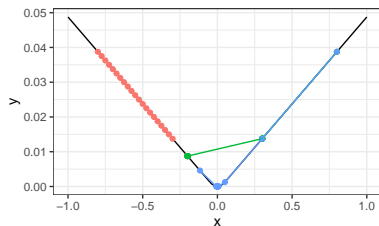
STEP SIZE CONTROL: DIMINISHING STEP SIZE

How can we adaptively control step size?

A natural way of selecting $\alpha^{[t]}$ is to decrease its value over time

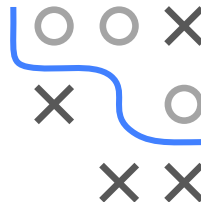
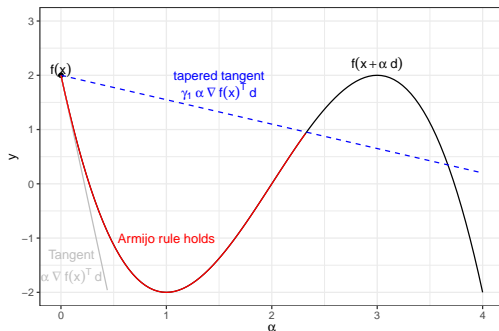
Example: GD on

$$f(x) = \begin{cases} \frac{1}{2}x^2 & \text{if } |x| \leq \delta, \\ \delta \cdot (|x| - 1/2 \cdot \delta) & \text{otherwise.} \end{cases}$$



GD with small constant (**red**), large constant (**green**), and diminishing (**blue**) step size

ARMIJO RULE



Inexact line search: Minimize objective “sufficiently” without computing optimal step size exactly

Common condition to guarantee “sufficient” decrease: **Armijo rule**

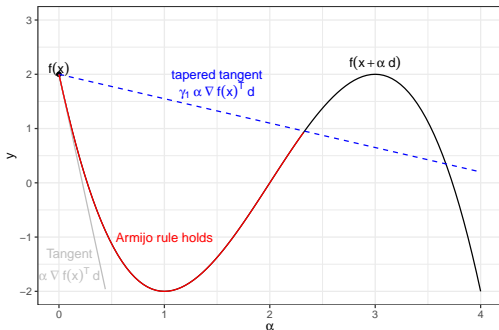
A 3x3 grid with a blue path starting at the top-left cell and ending at the bottom-right cell. The path moves right, then down, then right again. The cells along the path are empty. The other cells contain either a circle or an 'X'.

	○	○	×
×			○
		×	×



Note: $\nabla f(\mathbf{x})^\top \mathbf{d} < 0$ (\mathbf{d} descent dir.) $\implies f(\mathbf{x} + \alpha \mathbf{d}) < f(\mathbf{x})$.

ARMIJO RULE



Feasibility: For descent direction \mathbf{d} and $\gamma_1 \in (0, 1)$, there exists $\alpha > 0$ fulfilling Armijo rule. In many cases, Armijo rule guarantees local convergence of GD and is therefore frequently used.

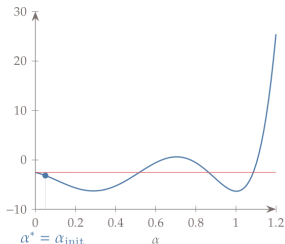
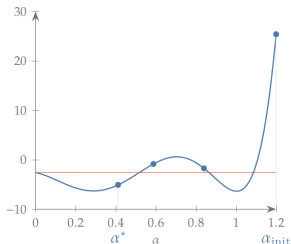
BACKTRACKING LINE SEARCH

Procedure to meet the Armijo rule: **Backtracking** line search

Idea: Decrease α until Armijo rule is met

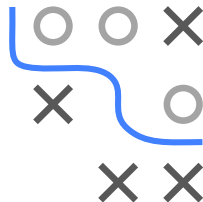
Algorithm Backtracking line search

- 1: Choose initial step size $\alpha = \alpha_{\text{init}}$, $0 < \gamma_1 < 1$ and $0 < \tau < 1$
 - 2: **while** $f(\mathbf{x} + \alpha \mathbf{d}) > f(\mathbf{x}) + \gamma_1 \alpha \nabla f(\mathbf{x})^T \mathbf{d}$ **do**
 - 3: Decrease α : $\alpha \leftarrow \tau \cdot \alpha$
 - 4: **end while**
-



(Source: Martins and Ning. *Engineering Design Optimization*, 2021.)

BACKTRACKING LINE SEARCH / 2



WOLFE CONDITIONS

Backtracking is simple and shows good performance in practice

But: Two undesirable scenarios

- ❶ Initial step size α_{init} is too large \Rightarrow need multiple evaluations of f
- ❷ Step size is too small with highly negative slopes

Solution for small step sizes:

- Fix γ_2 with $0 < \gamma_1 < \gamma_2 < 1$.
- α satisfies **sufficient curvature condition** in \mathbf{x} for \mathbf{d} if

$$|\nabla f(\mathbf{x} + \alpha \mathbf{d})^\top \mathbf{d}| \leq \gamma_2 |\nabla f(\mathbf{x})^\top \mathbf{d}|.$$

Armijo rule + sufficient curvature condition = **Wolfe conditions**

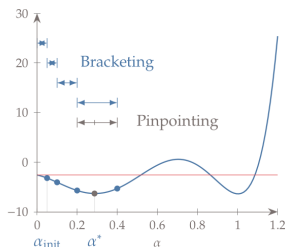
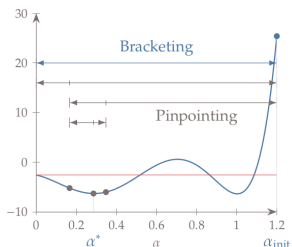


A 3x3 grid with a blue path starting at the top-left cell (0,0) and ending at the bottom-right cell (2,2). The path consists of the following cells: (0,0), (0,1), (1,1), (1,2), and (2,2). The cells (0,2), (1,0), and (2,0) are empty. The cells (0,1), (1,1), and (1,2) contain a grey 'X'. The cells (1,0) and (2,0) contain a grey circle.

BRACKETING & PINPOINTING

Example:

- Large initial step size results in quick bracketing but multiple pinpointing steps (**left**).
- Small initial step size results in multiple bracketing steps but quick pinpointing (**right**).



Source: Martins and Ning. *EDO*, 2021.