MULTIPLE CLASSES

More than two classes

For some classifiers, multiple classes are natural. We have already seen one:

- Simple classifier fitting one Gaussian per class.

We will discuss more examples soon:

- Trees.
- Ensembles: Number of classes is determined by weak learners.

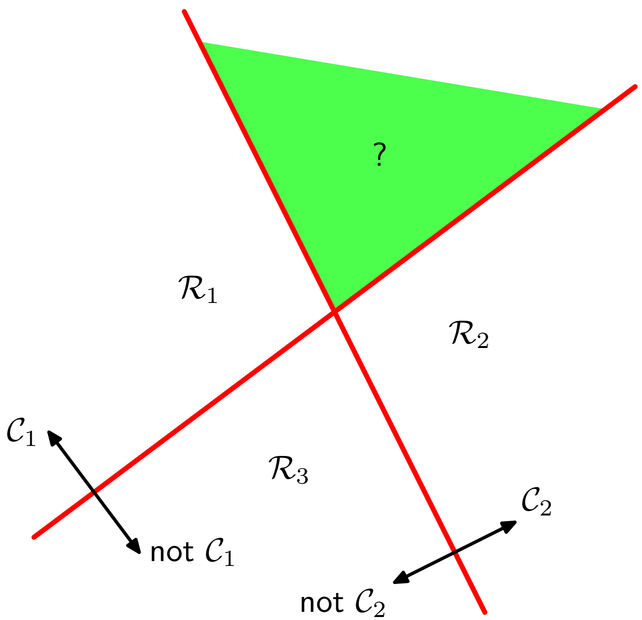
Exception: All classifiers based on hyperplanes.

Linear Classifiers

Approaches:

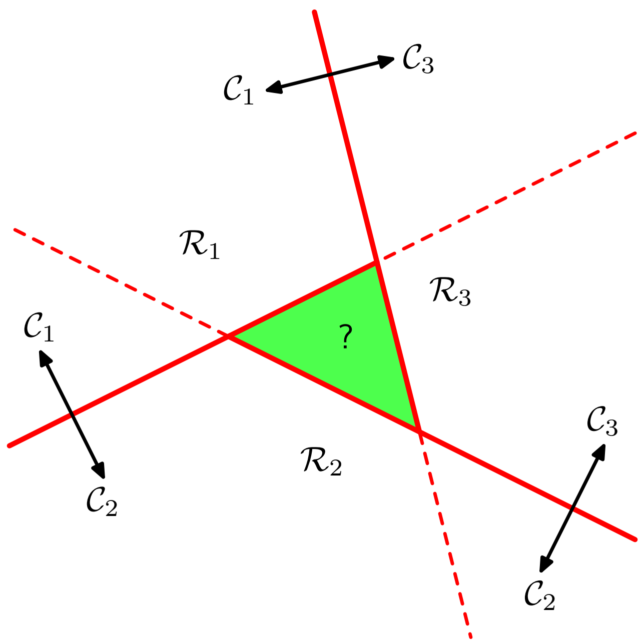
- One-versus-all (more precisely: one-versus-the-rest) classification.
- One-versus-one classification.
- Multiclass discriminants.

One-Versus-All Classification



- One linear classifier per class.
- Classifies "in class k" versus "not in class k".
- This is a two-class classifier that defines:
    - Positive class = $\mathcal{C}_k$.
    - Negative class = $\cup_{j \neq k} \mathcal{C}_j$.
- Problem: Ambiguous regions (green in figure).

One-Versus-One Classification



- One linear classifier for each pair of classes (i.e. $\frac{K(K-1)}{2}$ in total).
- Classify by majority vote.
- Problem again: Ambiguous regions.

MULTICLASS DISCRIMINANTS

Linear classifier

- Recall: Decision rule is $f(\mathbf{x}) = \operatorname{sgn}(\langle \mathbf{x}, \mathbf{v}_{\mathrm{H}} \rangle - c)$
- Idea: Combine classifiers *before* computing sign. Define

$$g_k(\mathbf{x}) := \langle \mathbf{x}, \mathbf{v}_k \rangle - c_k$$

Multiclass linear discriminant

- Use one classifier g_k (as above) for each class k.
- Trained e.g. as one-against-rest.
- Classify according to

$$f(\mathbf{x}) := \arg\max_k \{g_k(\mathbf{x})\}$$

- If $g_k(\mathbf{x})$ is positive for several classes, a larger value of g_k means that $\mathbf{x}$ lies "further" into class k than into any other class j.
- If $g_k(\mathbf{x})$ is negative for all k, the maximum means we classify $\mathbf{x}$ according to the class represented by the closest hyperplane.

Problem

- Multiclass discriminant idea: Compare distances to hyperplanes.
- Works if the orthogonal vectors $\mathbf{v}_{\mathrm{H}}$ determining the hyperplanes are normalized.
- For some of the best training methods for linear classifiers, that does not work well.

OPTIMIZATION

MOTIVATION

Recall from classification

- We "train" e.g. a linear classifier by finding the affine plane for which the empirical risk defined by a given loss function becomes as small as possible.

This is an example of phrasing a problem as an "optimization problem":

- There is a real-valued function (here: the empirical risk) that measures how good a given solution is.
- We choose that solution for which this function is minimal.

More generally

A variety of problems in statistics, machine learning and data mining are phrased as optimization problems:

- Fitting a parametric model: Maximum likelihood
- Training a classifier: Minimize an empirical risk under a given loss function
- Linear regression: Minimize a least squares error
- Sparse regression: Minimize a penalized least squares error
- Training neural networks: Minimize an empirical risk; loss can be chosen for classification or for regression task.

TERMINOLOGY

Min and Argmin

$$\min_x f(x) = \text{ smallest value of } f(x) \text{ for any } x$$

$$\arg\min_x f(x) = \text{ value of } x \text{ for which } f(x) \text{ is minimal}$$

Minimum with respect to subset of arguments

$$\min_x f(x,y) = \text{ smallest value of } f(x,y) \text{ for any } x \text{ if } y \text{ is kept fixed}$$

Optimization problem

For a given function $f : \mathbb{R}^d \to \mathbb{R}$, a problem of the form

$$\text{find} \quad \mathbf{x}^* := \arg\min_{\mathbf{x}} f(\mathbf{x})$$

is called an **minimization problem**. If arg $\min$ is replaced by arg $\max$, it is a **maximization problem**. Minimization and maximization problems are collectively referred to as **optimization problems**.

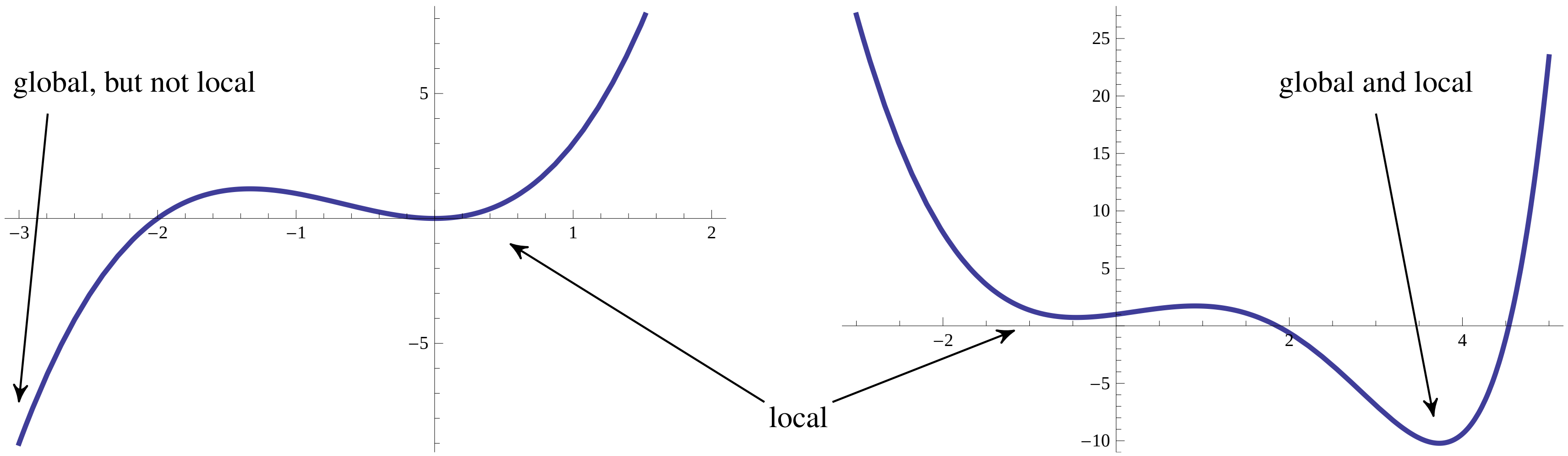
Minimization vs Maximization

For any function f, we have

$$\min f(x) = -\max(-f(x)) \qquad \text{and} \qquad \arg\min f(x) = \arg\max(-f(x))$$

That means:

- If we know how to minimize, we also know how to maximize, and vice versa.
- We do not have to solve both problems separately; we can just generically discuss minimization.



Local and global minima

A minimum of f at x is called:

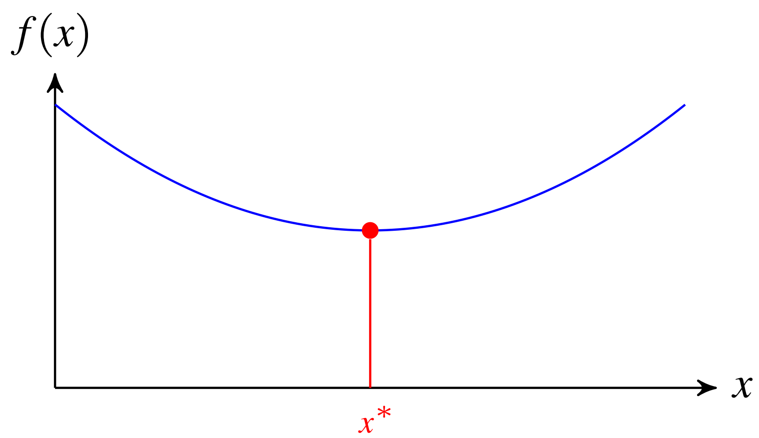
- **Global** if f assumes no smaller value on its domain.
- **Local** if there is some open interval (a, b) containing x such that $f(x)$ is a global minimum of f restricted to that interval.

Typical situation

- Given is a function $f : \mathbb{R}^d \to \mathbb{R}$.
- The dimension d is usually very large.
(In neural network training problems: Often in the millions.)
- We cannot plot or "look at" the function.
- We can only evaluate its value $f(x)$ point by point.

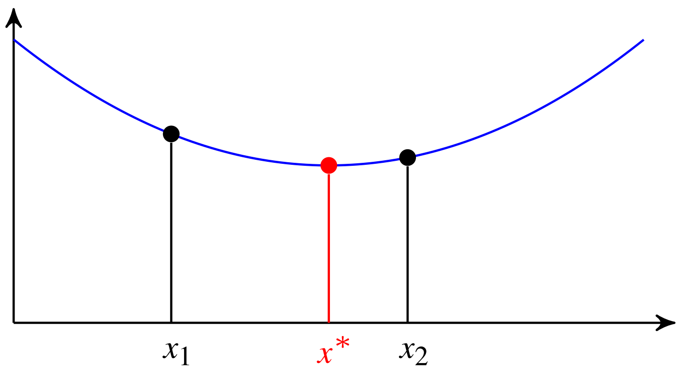
One-dimensional illustration

Here, $d = 1$ (but keep in mind we are interested in very large d.)

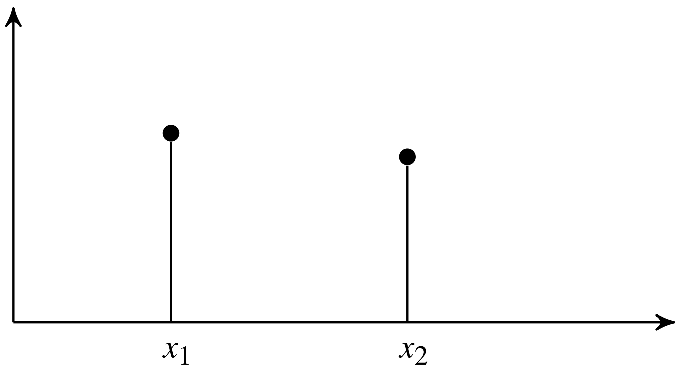


The minimizer we are interested in is x^*.

ONE-DIMENSIONAL ILLUSTRATION



- Our goal is to find x^*.
- We can evaluate the function at points of our choice, say x_1 and x_2.



- However, we cannot "see" the function.
- All we know are values at a few points.

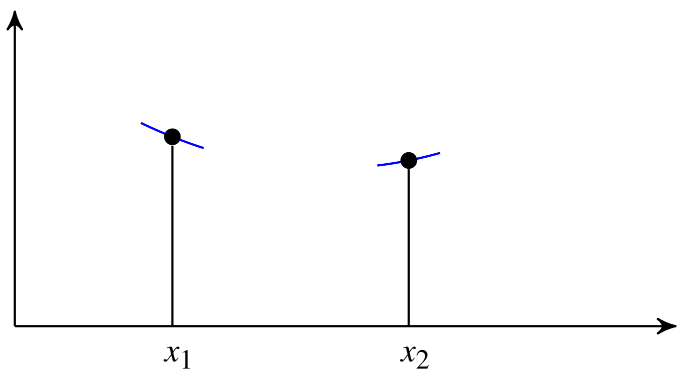
Task

Based on the values we know, we have to:

- Either make a decision what x^* is.
- Or gather more information, by evaluating f at additional points. In that case, we have to decide which point to evaluate next.

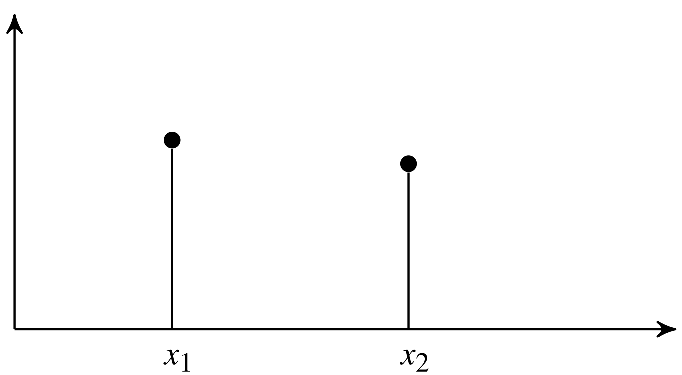
Next steps

- We will first consider how we would proceed if we had access to the entire function in a small neighborhood around each of the points $x_1, x_2, \ldots$, i.e. if we could see something like this:

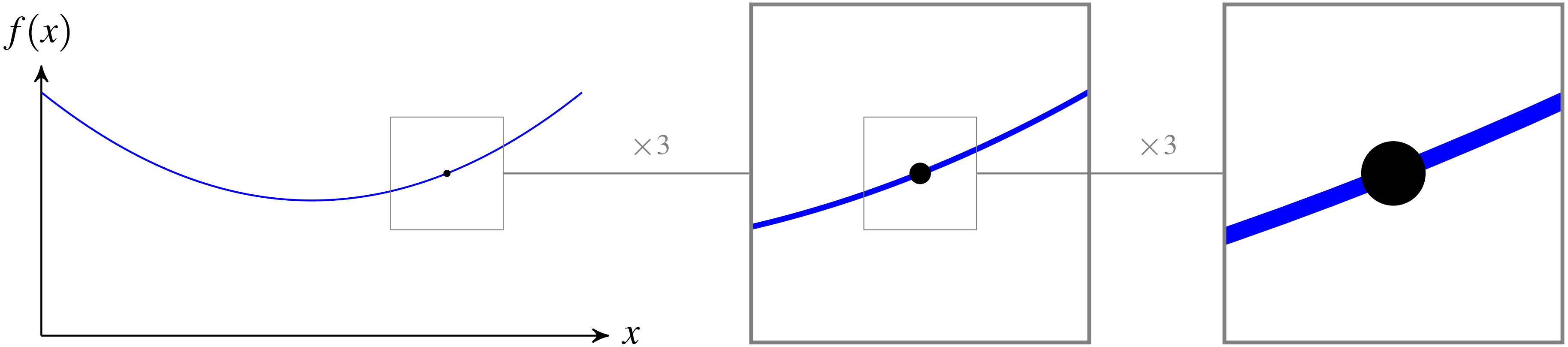


  To this end, we discuss the concept of a derivative.

- We then consider what we can actually implement on a computer, given that we only have access to point-wise information:



Zooming in on a smooth function



Observation

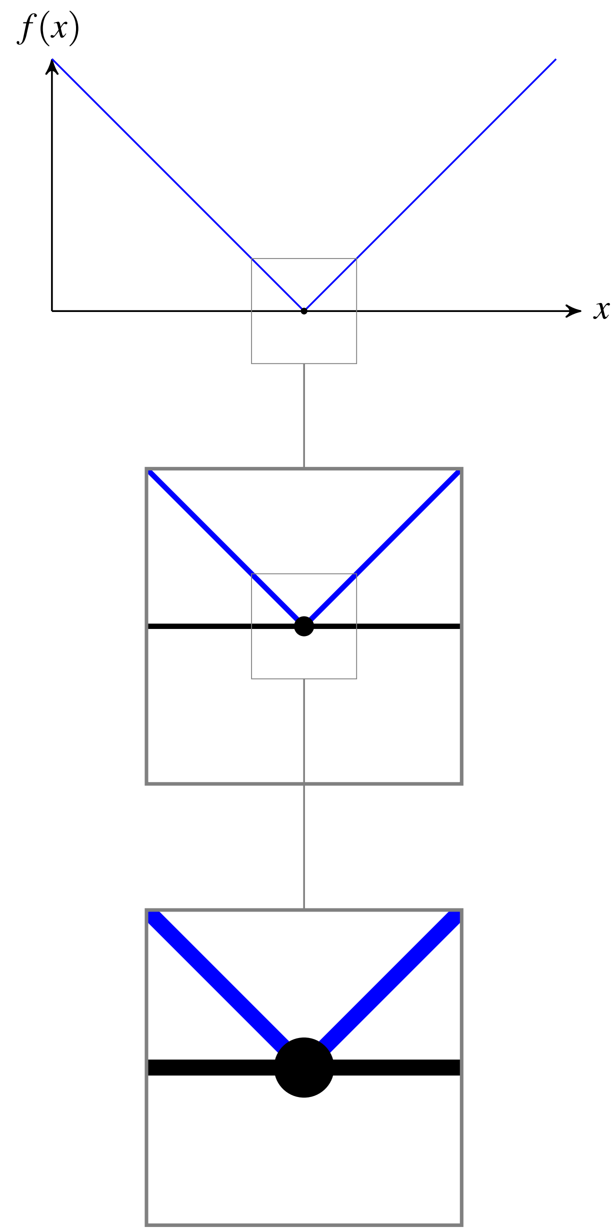
- Each time we zoom in, the curve looks more like a straight line.
- If we zoom in far enough, we can replace the curve in a small area around the marked point by a straight line.
- In mathematical jargon, that is called an *approximation*: We replace the curve around the marked point by a surrogate curve. If that surrogate is a straight line (i.e. a linear function), it is a *linear approximation*.

A counter example

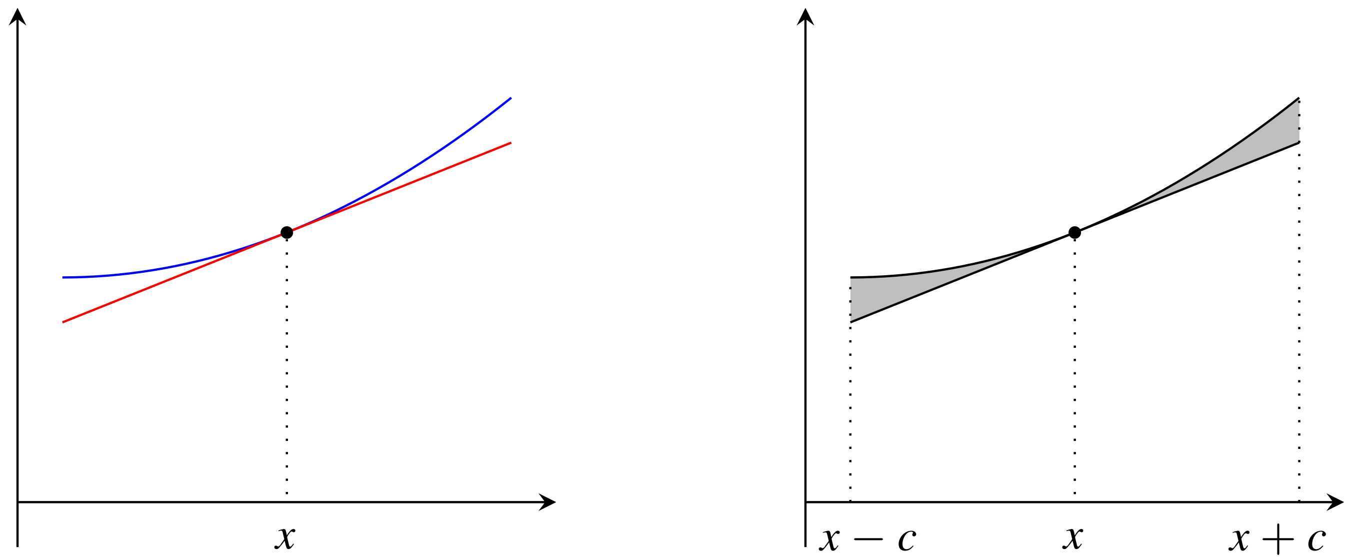
- Not every function has this property.
- Here, we consider the absolute value function $f(x) = |x|$, and zoom in on the point $x = 0$.
- In this case, the shape of f never seems to change.
- Note this would be different if we had picked any other point than $x = 0$.

We observe

- Whether a function is "locally straight" is a property that may be true at some points, but not at others.
- Clearly it matters whether the function is "smooth" around the point we focus on.



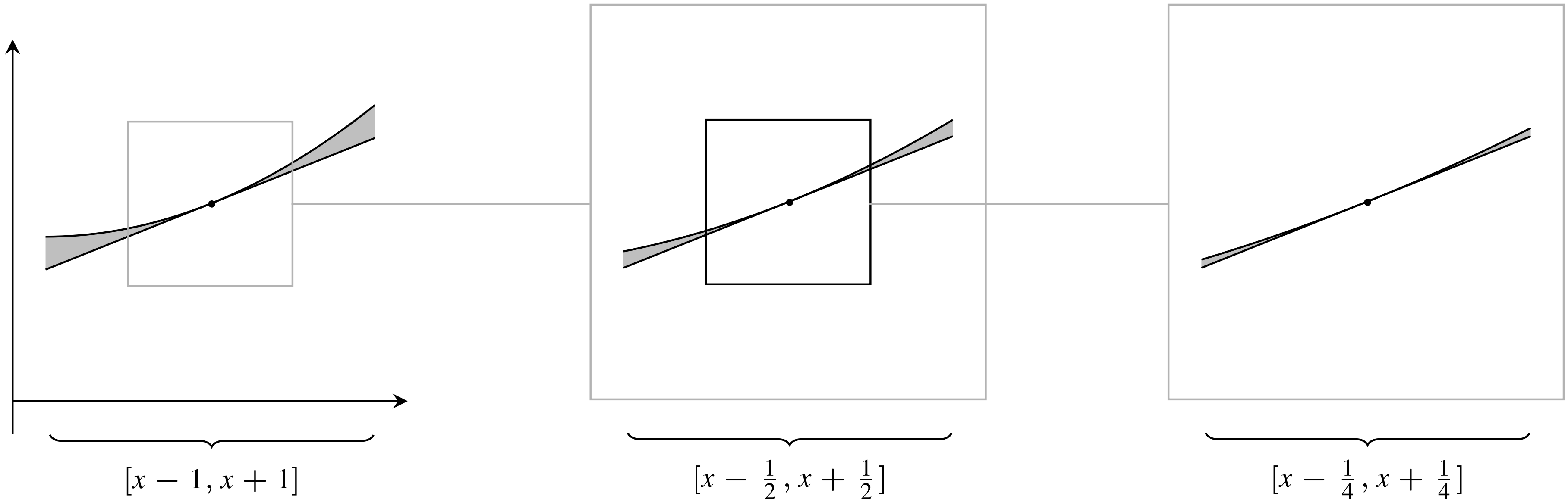
APPROXIMATING BY A STRAIGHT LINE



- We consider a function (blue) and approximate it at a point x by a straight line (red).
- To measure how good the approximation is, we fix a constant $c > 0$ and enclose x in the interval $[x - c, x + c]$.
- On this interval, we compute the area between the two functions (shaded in gray). Suppose this area is $A(x, c)$.
- Of course, $A(x, c)$ will grow if we make c larger. To make the area comparable for different values of c, we use the *relative* approximation error

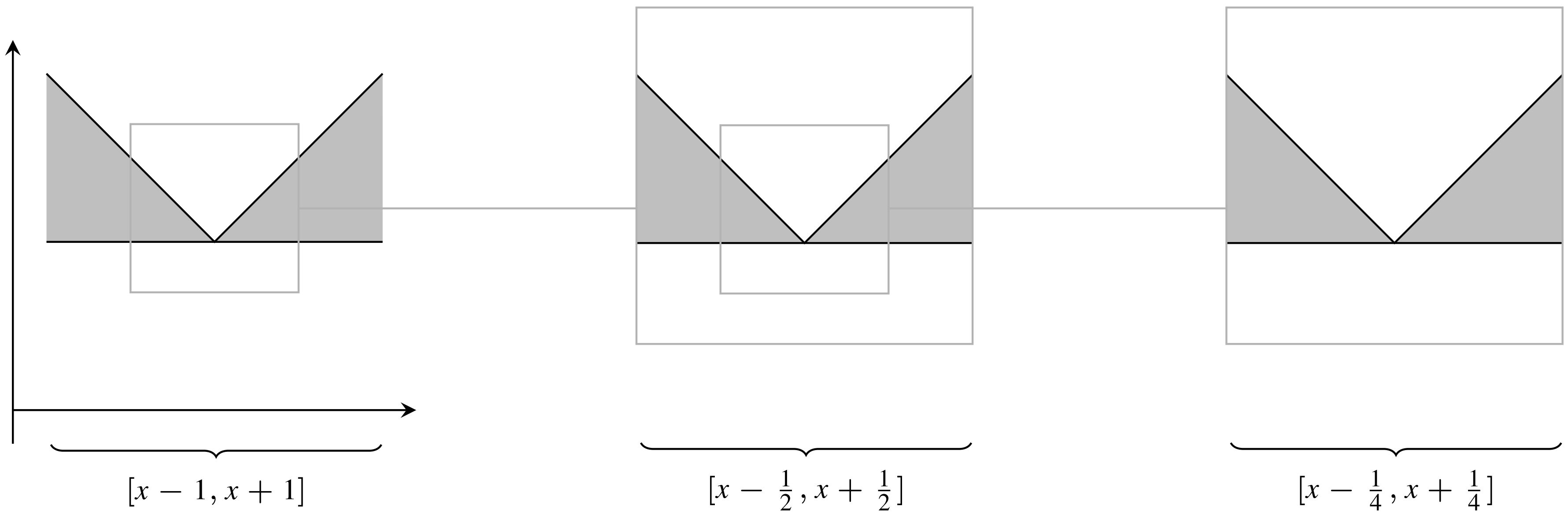
$$r(c) = \frac{A(x,c)}{|[x-c, x+c]|} = \frac{A(x,c)}{2c} \;.$$

Approximating by a Straight Line



- Now consider what happens if we zoom in, by making c smaller and smaller.
- If the function is smooth, we observe the relative error becomes smaller each time.
- The function can be approximated by the line to arbitrary precision, that is: If we are permitted *any* error $\varepsilon > 0$, we can always find a small enough c such that $r(c) < \varepsilon$.
- In this sense, the linear approximation (= approximation by a straight line) is *locally exact*.
- If a straight line can be chosen for f and x such that the relative approximation error can be made arbitrarily small by making the intervall sufficiently small, then f is called **differentiable at** x.

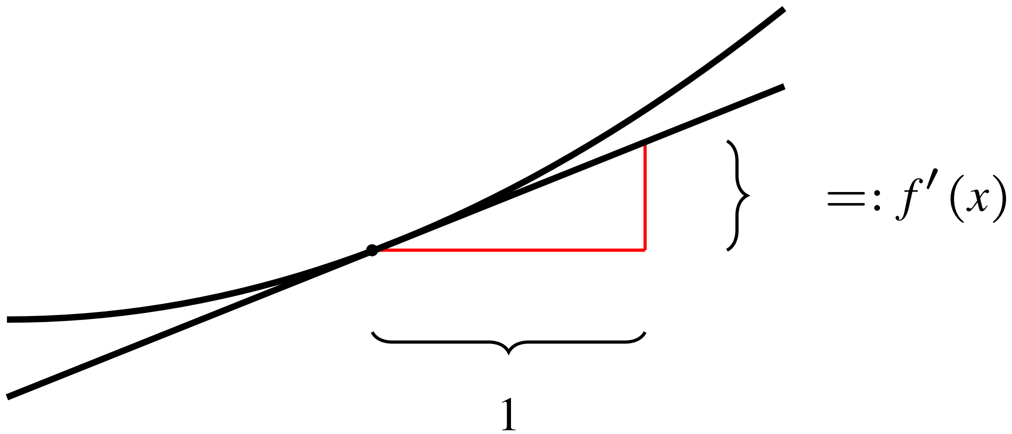
ZOOMING IN ON NON-SMOOTH FUNCTION



Now try the same for the absolute value function:

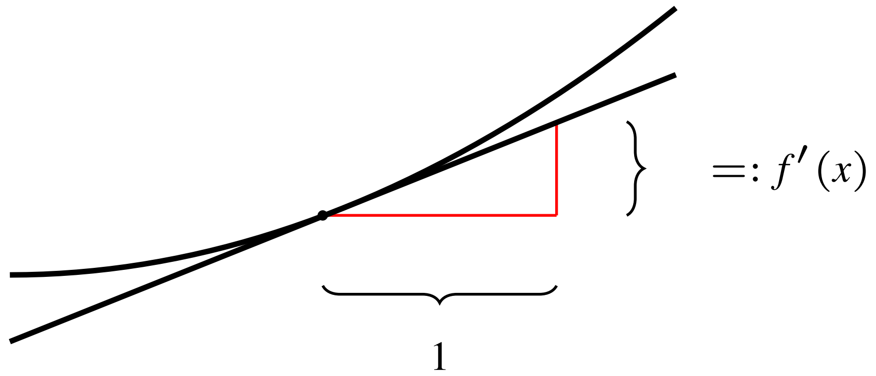
- Approximate it at $x = 0$ by a horizontal line.
- Here, the relative error around $x = 0$ remains the same regardless of how we choose c.
- We could also use an approximating line with a different slope, and would encounter the same problem.
- Thus, $|x|$ is not differentiable at $x = 0$ (although it is differentiable at every other point x).

THE DERIVATIVE

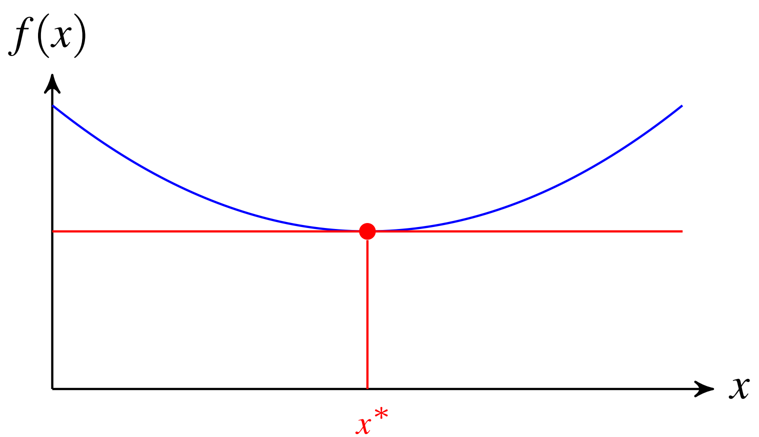


- If f is differentiable at x, there is a unique approximating line at x for which the relative error is minimal as c gets smaller.
- We can measure the slope of this line by substracting its values at $x + 1$ and x.
- We denote this slope by $f'(x)$ and call it the **derivative** of f at x.
- If f is differentiable at every point x, we can compute the value $f'(x)$ at every point, so f' is again a function. In general, it takes different values at different points x.

Some Properties of the Derivative



- If f increases around x, then $f'(x) > 0$. If f decreases, then $f'(x) < 0$.
- Recall that we are interested in finding minima and maxima. If f is differentiable at x and x is a local minimum or maximum, the approximating line is horizontal:



That means: At a (differentiable) maximum or minimum x^*, we have $f'(x^*) = 0$.