# Numerical Optimization with Python

Lecture 2: Unconstrained Optimization (Part 1/2)

# Lecture 02: Unconstrained Optimization (Part 1/2)

- Problem definition
- Necessary conditions (first and second order) for a local minimum
- Sufficient conditions
- Definition of convex functions (and global minimizers)
- Overview of algorithms: line search and trust regions
- Gradient Descent: naïve version

# Problem Definition

- Minimize an objective function that depends on real variables, with no restriction on their values:

$$\min_{x \in \mathbb{R}^n} f(x)$$

- Unless otherwise stated we will assume $f: \mathbb{R}^n \to \mathbb{R}$ is a smooth function (usually we will need continuous second derivatives)
- Typically: we do not have any global perspective of $f$, and only have local information (values of $f$ and perhaps its derivatives at points we can usually choose)

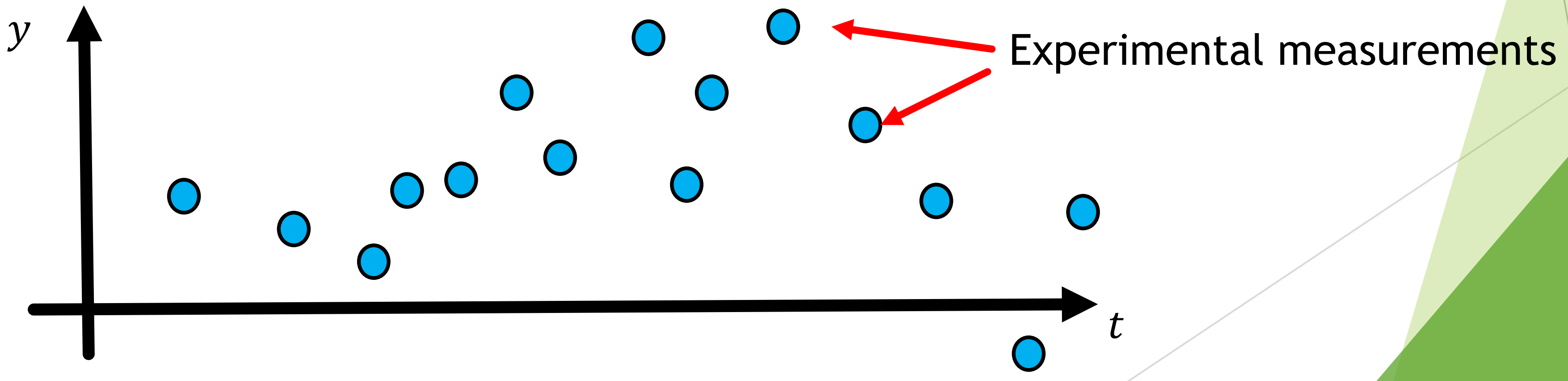
# Problem Definition

- We would like algorithms that:
  - Identify solutions efficiently (time, computer storage)
  - Do not evaluate $f$ or its derivatives unnecessarily, as sometimes evaluations are computationally expensive

# Problem Definition

**Example – nonlinear least squares:**

- Assume we have experimental data $(t_i, y_i)_{i=1,\dots,m}$ from some physical measurements ($t_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$)
- We seek a mapping $\phi: \mathbb{R}^d \to \mathbb{R}$ to model the measured physical phenomenon, that best fits the measured data (in *some* sense)



# Problem Definition

<u>**Example - nonlinear least squares:**</u>

- From prior knowledge (or analysis) we assume the model has a linear trend, two periodic components and an exponential component
- Hence we restrict our search to:

$$\phi(t) = \phi(t; w_0, w_1, w_2, w_3, w_4, w_5)$$

$$= w_0 + w_1 t + \sin w_2 t + \sin w_3 t + e^{-\left(\frac{w_4 - t}{w_5}\right)^2}$$

where $w = [w_0, w_1, w_2, w_3, w_4, w_5]$ is a vector of unknown variables that parameterize the family of models we investigate

# Problem Definition

<u>**Example - nonlinear least squares:**</u>

- Our usage of the observed data is to define the *residuals* (or *errors*) the assumed model has w.r.t the measurements:

$$r_i = y_i - \phi(t_i; w)$$

**Observed** value at $t_i$

**Modelled** value at $t_i$, given the vector $w$

# Problem Definition

**Example – nonlinear least squares:**

- We can formulate the problem of determining the unknown parameters $w$ that minimize the sum of the squared residuals:

$$\min_{w \in \mathbb{R}^6} r_1^2 + r_2^2 + \cdots + r_m^2$$

- The problem has 6 unknowns, and $m$ terms in the objective function (and no constraints)

# Problem Definition

**Example – nonlinear least squares:**

Some points to consider and variants:

- Is there a unique solution? If not, under which conditions will there be?
- What type of objective function are we minimizing? (Looks quadratic, does it?)
- Is the objective computationally expensive to evaluate? What does that depend on?

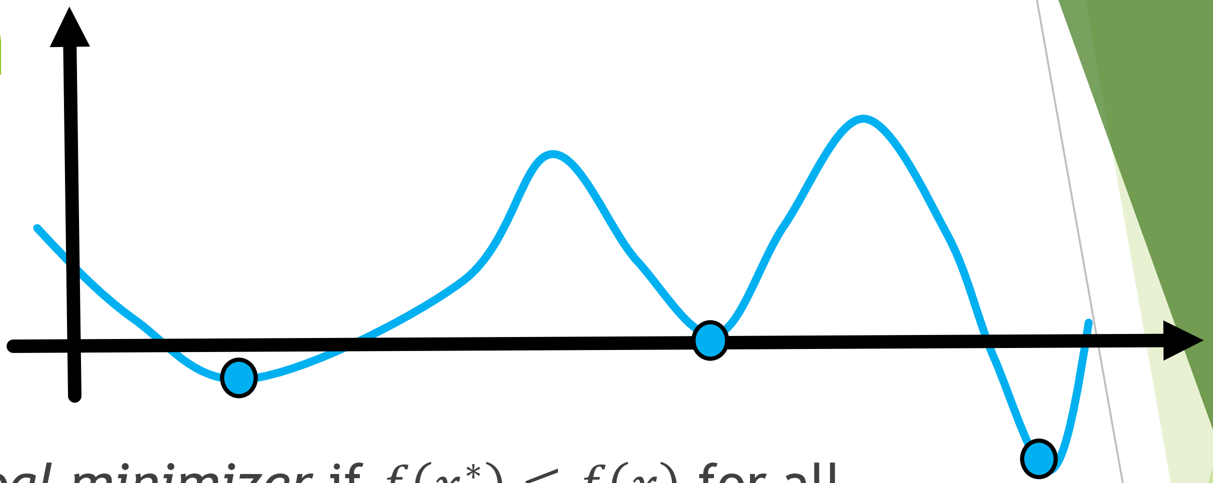
# Problem Definition

**Example – nonlinear least squares:**

Some points to consider and variants (cont.):

- Is the objective sensitive to outliers/erroneous measurements? What does that depend on? How can it be made more robust?
- The error measure we optimize: does it depend on $m$ (number of observations)? What can be done about that?
- In what units is our error measure? What can we do about that?

# Problem Definition



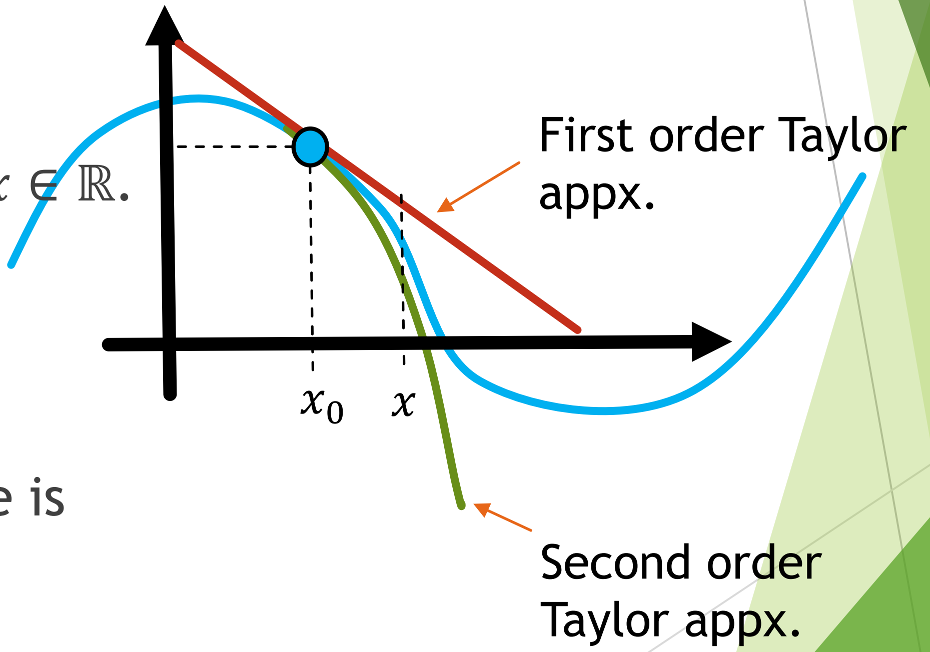
- **Definition**: a point $x^*$ is a *global minimizer* if $f(x^*) \leq f(x)$ for all $x \in \mathbb{R}^n$
- We typically have only local information on $f$ and most algorithms will be able to converge to *local minimizers*, defined next:
- **Definition**: a point $x^*$ is a *local minimizer* if there exists a neighborhood $N$ of $x^*$ such that $\forall x \in N, f(x^*) \leq f(x)$
- Strict (or strong) minimizers are defined as above but with strict inequalities (while the others may be referred to as weak minimizers)

# Necessary Conditions for a Local Min

**Recall Taylor's Theorem in one variable, for differentiable functions of first and second order:**

- Assume $f: \mathbb{R} \to \mathbb{R}$ is differentiable, and $x_0, x \in \mathbb{R}$. Then there is point $c \in (x_o, x)$ such that:

  $f(x) = f(x_0) + f'(c)(x - x_0)$




- If $f$ is twice differentiable, then then there is point $c \in (x_o, x)$ such that we can write:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2} f''(c)(x - x_0)^2$$

# Necessary Conditions for a Local Min

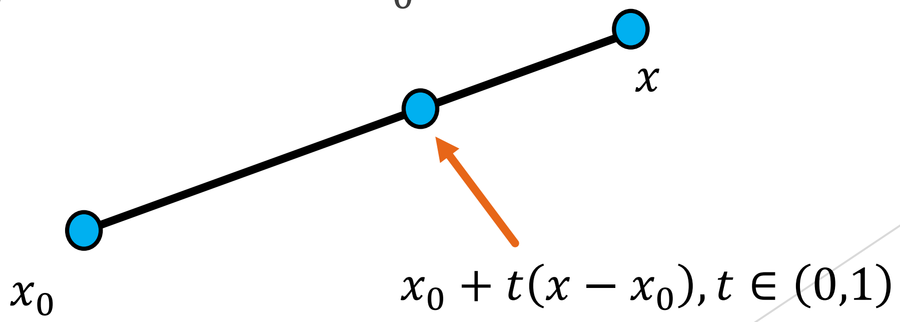
**Recall Taylor's Theorem in one variable, for differentiable functions of first and second order (cont.):**

- We do not know where $c$ is exactly, but usually bounds on derivative values in the interval are useful for bounds on the approximation err
- If the derivatives are continuous (also denoted $f \in C^1$ or $f \in C^2$) we can guarantee bounds in some neighborhood (we will use in a few slides)

# Necessary Conditions for a Local Min

**For multivariate functions – we now show Taylor's theorem using the chain rule:**

- Assume $f: \mathbb{R}^n \to \mathbb{R}$ is differentiable, and we would like first and second order approximations that are analog to the univariate case
- For that we consider the line segment between $x_0$ and $x$ that can be parametrized as follows:



# Necessary Conditions for a Local Min

**A comment on notation:**

- In what follows, we will both use both $\nabla^2 f(x)$ and $H(x)$ to denote the Hessian matrix of a scalar valued, twice differentiable function $f$ at a point $x$
- We will use them interchangeably, with no confusion, as they mean the same thing: the matrix with $\frac{\partial^2 f(x)}{\partial x_i \partial x_j}$ in its $i, j$'th entry
- Note that $\nabla^2$ does make sense: go ahead and differentiate that vector valued function $x \mapsto \nabla f(x)$, and obtain the Hessian matrix. We may think of this as applying the $\nabla$ operator twice

# Necessary Conditions for a Local Min

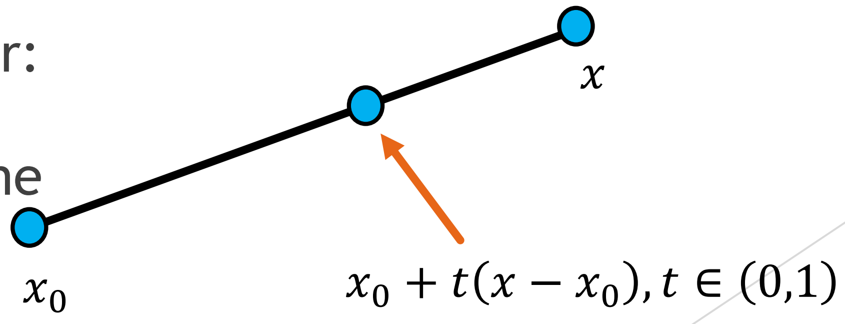
**For multivariate functions - we now show Taylor's theorem using the chain rule (cont.):**

- The restriction of $f$ to the line segment is a function of a single variable

  $t \in [0,1]$, defined by: $g(t) = f\big(x(t)\big) = f\big(x_0 + t(x - x_0)\big)$

- For $g$ we have the first and second order Taylor approximations from the previous slides. First order:

- $f(x) = g(1) = g(0) + g'(t_c)$ for some number $t_c \in (0,1)$ (and $t - t_0 = 1$)



# Necessary Conditions for a Local Min

**For multivariate functions - we now show Taylor's theorem using the chain rule (cont.):**

- Using the chain rule, we differentiate:

$$g'(t) = \nabla f\big(x(t)\big)^T \frac{dx}{dt} = \nabla f\big(x(t)\big)^T (x - x_0)$$

- Hence the point $c = x(t_c)$ is the unknown point and we have our first order analog:

$$f(x) = f(x_0) + \nabla f(c)^T (x - x_0)$$

Make sure the dimensions and matrix multiplication makes sense here! Note that:
$\frac{dx}{dt} = [x_1'(t), \ldots, x_n'(t)]^T$ (column vector)

# Necessary Conditions for a Local Min

**For multivariate functions - we now show Taylor's theorem using the chain rule (cont.):**

- Using the chain rule, we differentiate:

$$g'(t) = \nabla f\big(x(t)\big)^T \frac{dx}{dt} = \nabla f\big(x(t)\big)^T (x - x_0)$$

- Hence the point $c = x(t_c)$ is the unknown point and we have our first order analog:

$f(x) = f(x_0) + \nabla f(c)^T (x - x_0)$

This should look familiar: the explicit equation for the tangent plane to $f$ at $x_0$ is:

$$L(x) = f(x_0) + \nabla f(x_0)^T (x - x_0)$$

# Necessary Conditions for a Local Min

**For multivariate functions - we now show Taylor's theorem using the chain rule (cont.):**

- To obtain our second order analog, we need $g''(t)$. We already have:

$$g'(t) = \nabla f\big(x(t)\big)^T \frac{dx}{dt} = \nabla f\big(x(t)\big)^T (x - x_0) = (x - x_0)^T \nabla f\big(x(t)\big)$$

- Differentiating again w.r.t $t$, recall $d[\nabla f(x)] = \nabla^2 f(x) = H(x)$:

$$g''(t) = (x - x_0)^T H\big(x(t)\big) \frac{dx}{dt} = (x - x_0)^T H\big(x(t)\big)(x - x_0)$$

The expression for $g''$ is a quadratic form of the Hessian matrix, quadratic in the vector $x - x_0$

# Necessary Conditions for a Local Min

**For multivariate functions - we now show Taylor's theorem using the chain rule (cont.):**

- Applying the second order appx for $t$, we have our multivariate analog:

$$f(x) = f(x_0) + \nabla f(x_0)^T (x - x_0) + \frac{1}{2}(x - x_0)^T H(c)(x - x_0)$$

Take a minute to make sure dimensions make sense. This is a scalar equation!

# Necessary Conditions for a Local Min

**Theorem (First Order Necessary Conditions):**

- If $f \in C^1$ in a neighborhood of $x^*$ and $x^*$ is a local minimizer, then $\nabla f(x^*) = 0$

**Proof:**

- The underlying idea: a non-zero gradient enables decrease in function values for a small enough step size in the direction $-\nabla f(x^*)$.
- Formally: in our first order approximation, choose $x - x_0$ to be the vector $-\alpha \nabla f(x^*)$, $\alpha$ is a positive scalar (the step size) we will soon choose appropriately

# Necessary Conditions for a Local Min

**Proof (cont.):**

- Note that $-\nabla f(x^*)^T \nabla f(x^*) = -\|\nabla f(x^*)\|^2 < 0$ and since $\nabla f$ is continuous, there is an entire neighborhood of $x^*$ for which $-\nabla f(x)^T \nabla f(x^*) < 0$ (as a function of $x$ in the first term)
- Hence we may choose a small enough $\alpha$ (displacement from $x^*$ along $-\nabla f$) for which $-\alpha \nabla f(c)^T \nabla f(x^*) < 0$ and $c$ is the intermediate point of the approximation: $f(x) = f(x^*) - \alpha \nabla f(c)^T \nabla f(x^*) < f(x^*)$, a contradiction.

(**Note**: a point $x^*$ for which $\nabla f(x^*) = 0$ is called a *stationary point*)

# Necessary Conditions for a Local Min

**Definition:** a (symmetric) matrix $A$ is called *positive semidefinite* if the quadratic form is non-negative, namely: for all $x \in \mathbb{R}^n$, $x^T Ax \geq 0$

The matrix is called *positive definite* if for all $x \neq 0$, the inequality is strict.

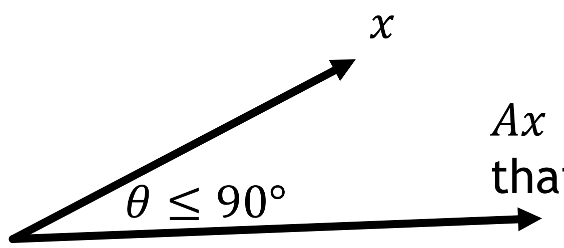
(Recall that the above has a criterion: all eigenvalues are non-negative/positive, respectively)

**Notation**: we sometimes denote by $A \succcurlyeq 0$ that $A$ is positive semidefinite and by $A \succ 0$ that $A$ is positive definite.

# Necessary Conditions for a Local Min

**Positive definite matrices**

- Thinking of scalars as operators via multiplication, in fact positive definite matrices are a generalization of positive numbers
- The image $Ax$ is in the same half space as $x$, due to the positive inner product $x^T Ax$
- The sign remains positive when operating on squared (positive) quantities




$Ax$ points in the same half-space
that $x$ is pointing ("same sign" analog)

# Necessary Conditions for a Local Min

**Theorem (Second Order Necessary Conditions):**

- If $f \in C^2$ in a neighborhood of $x^*$ and $x^*$ is a local minimizer, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite

**Proof:**

- From the previous theorem we have $\nabla f(x^*) = 0$. Now, assume the opposite, namely $\nabla^2 f(x^*)$ is not positive semidefinite.

# Necessary Conditions for a Local Min

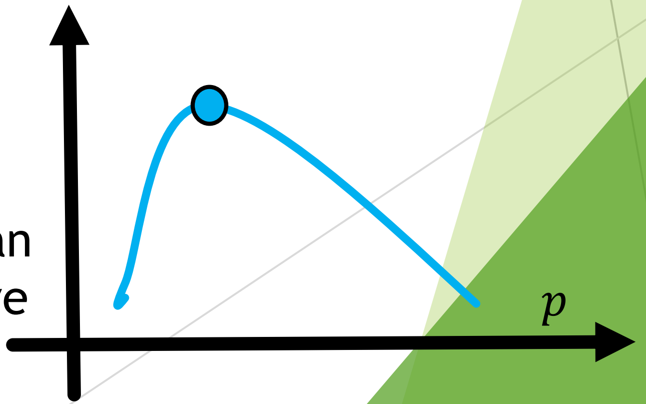
**Proof (cont.):**

- Then we can choose a direction $p$ such that $p^T \nabla^2 f(x^*) p < 0$ and since $\nabla^2 f(x)$ is continuous, an entire neighborhood of $x^*$ enables choosing a small enough displacement along $p$ such that:

$$f(x) = f(x^*) + \nabla f(x^*)^T p + \frac{1}{2} p^T H(c) p = f(x^*) + \frac{1}{2} p^T H(c) p < f(x^*)$$

which is again a contradiction.

For at least one direction $p$ we can decrease function values (negative second derivative along $p$)



# Sufficient Conditions for a Local Min

**Theorem (Second Order Sufficient Conditions):**

- Assume $f \in C^2$, $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. Then $x^*$ is a strict local minimizer of $f$.

**Proof:**

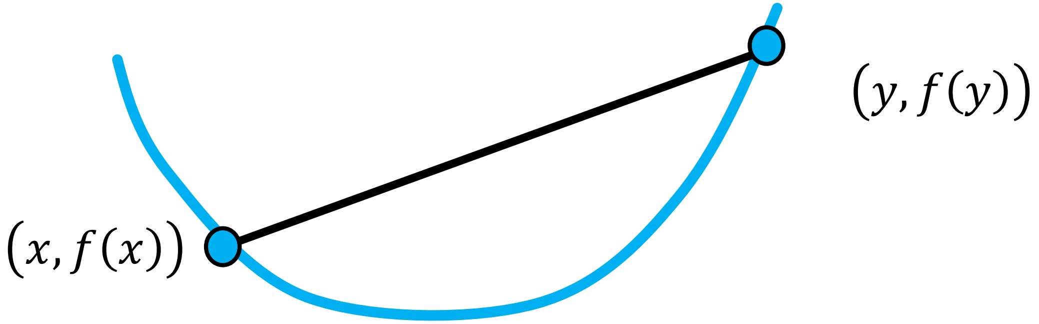
- From the continuity of $\nabla^2 f$ we have an entire neighborhood of $x^*$ for which $\nabla^2 f$ is positive definite
- Hence in any direction $p$, for small enough step $||p|| < r$, we have:

$$f(x) = f(x^*) + p^T \nabla^2 f(c) p > f(x^*)$$

# Convex Functions

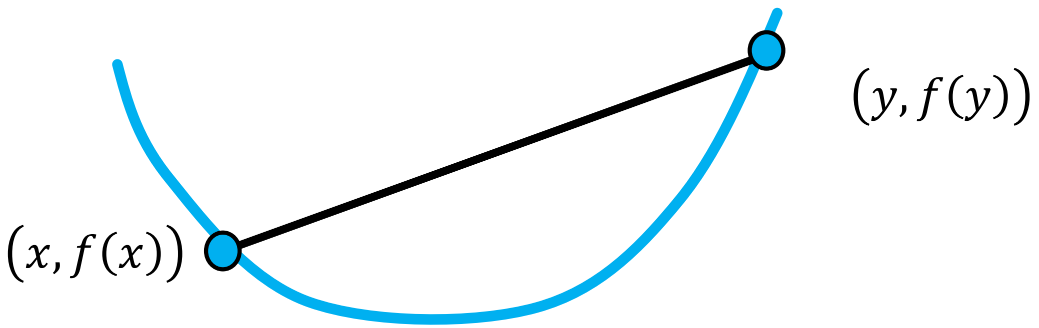
- When the objective function is convex, global and local minimizers will be easy to characterize
- **Definition:** let $f: \mathcal{D} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$ and assume its domain $\mathcal{D}$ is convex. The function $f$ is *convex* if for all $x, y \in \mathcal{D}$ and for any $\alpha \in [0,1]$:

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y)$$



# Convex Functions

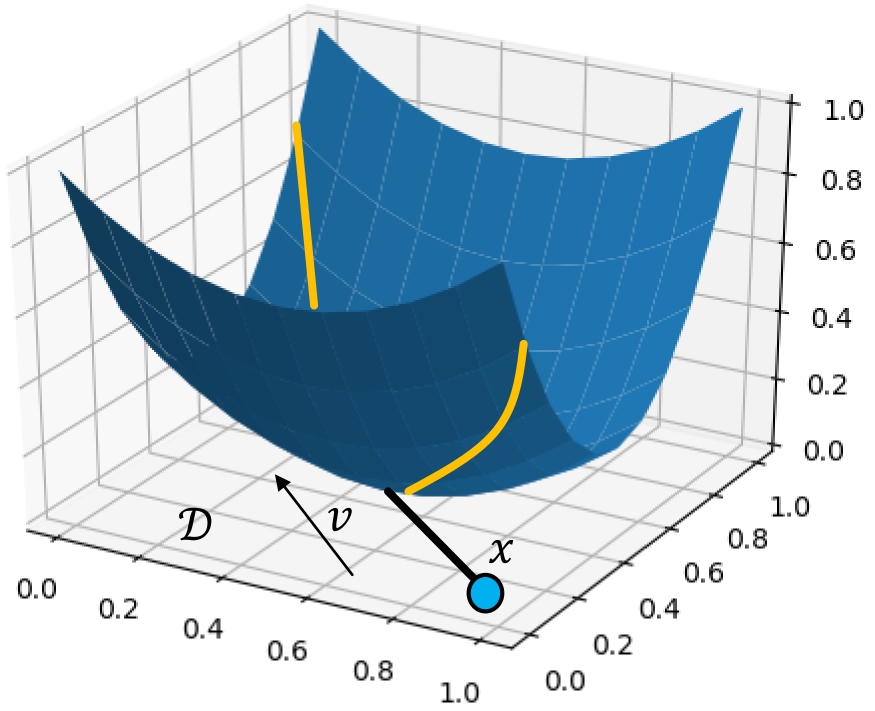
- Geometrically, this means that the line segment between $(x, f(x))$ and $(y, f(y))$ lies above the graph of $f$
- A function is called *strictly convex* if strict inequality holds whenever $x \neq y$ and $\alpha \in (0,1)$
- (We say that $f$ is *concave* if the opposite in equalities hold)



# Convex Functions

- Linear and affine functions are both convex and concave
- A function is convex if and only if its restriction to any line is convex, as a function of a single variable:

$g(t) = f(x + tv)$ where $x + tv \in \mathcal{D}$



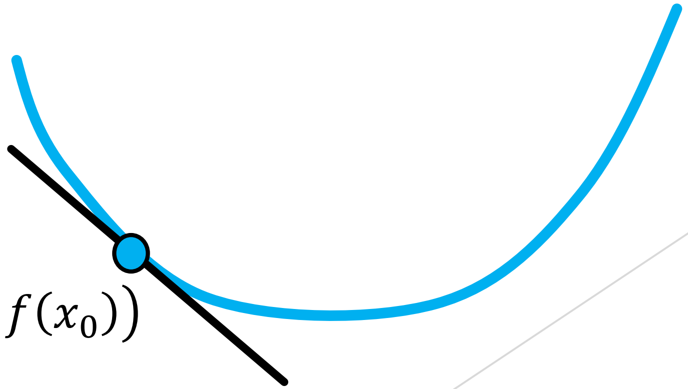
# Convex Functions

**First order conditions for convexity**

- Assume $f$ is differentiable, that is: $\nabla f$ exists for all points of the (open) domain $\mathcal{D}$. Then $f$ is convex if and only if for all $x, y \in \mathcal{D}$:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x)$$

$$L(x) = f(x_0) + \nabla f(x_0)^T (x - x_0)$$



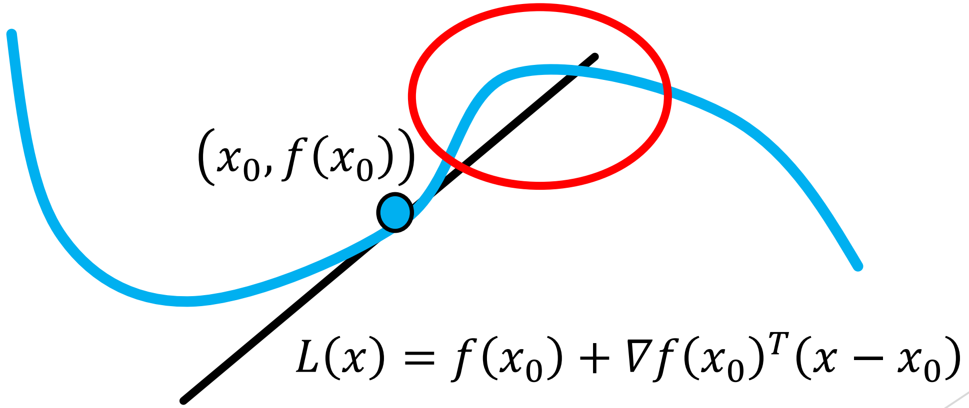
$\left(x_0, f(x_0)\right)$

Geometrically: The tangent plane at any point lies entirely below the graph

# Convex Functions

**First order conditions for convexity - geometry behind the proof:**

- A tangent plane that is not entirely below the graph of $f$, enables selecting a cord that will violate the definition of convexity (cord above graph):
- (Read the full proof:
  Boyd, Ch03 p.70)

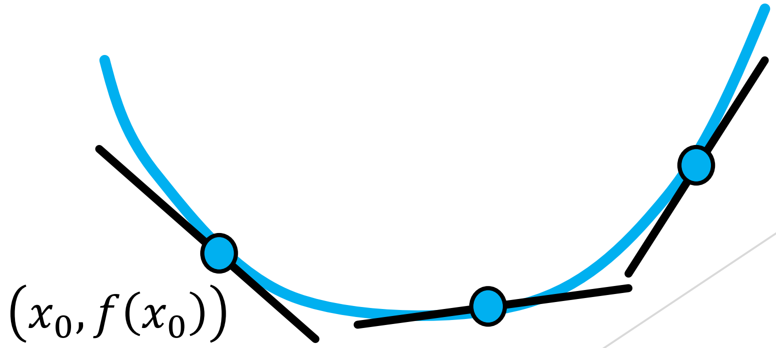


# Convex Functions

**Second order conditions for convexity**

- Assume $f$ is twice differentiable, that is: $\nabla^2 f$ exists for all points of the (open) domain $\mathcal{D}$. Then $f$ is convex if and only if $\nabla^2 f(x) \succcurlyeq 0$

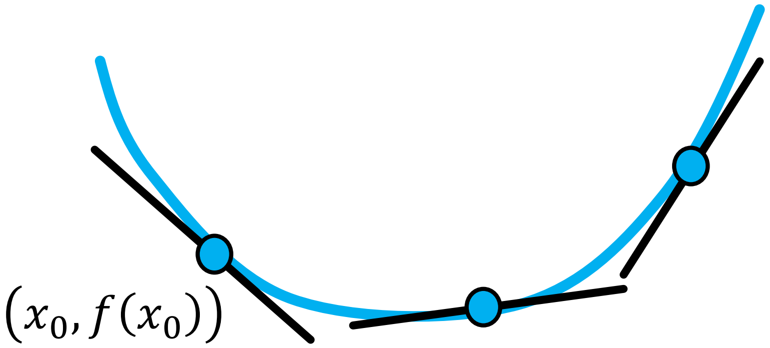
Geometrically: the graph has *positive curvature* – slopes are changing upwards



# Convex Functions

**Second order conditions for convexity – proof main idea:**

- First make the positive curvature intuition formal in 1D, using the first order conditions:

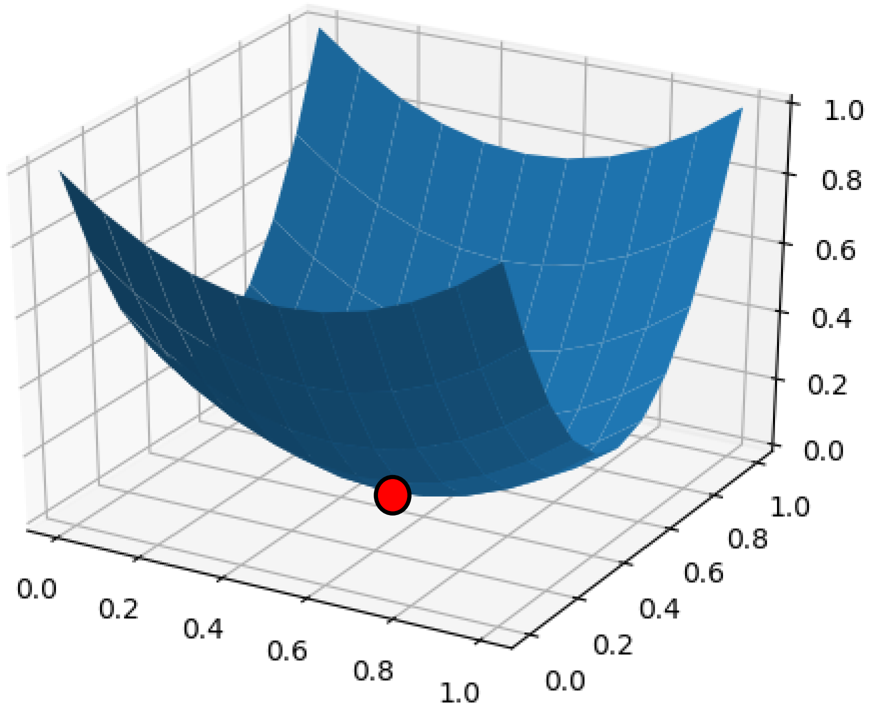



- Then use the chain rule to show that that in any direction $v$ in space, the second derivative is exactly the quadratic form of the Hessian in $v$ (positive!)
- Show that 1D convexity in every direction is equivalent to convexity in $\mathbb{R}^n$

# Convex Functions

**Theorem**: if $f$ is convex, any local minimizer $x^*$ is a global minimizer. If, in addition, $f$ is differentiable, then any stationary point $x^*$ is a global minimizer of $f$.

- Underlying idea: if it is not a global minimizer – a cord will violate convexity
- (proof: Nocedal & Wright Ch02)



# Convex Functions

- It is sometimes useful to define the extended value version of a convex function as follows:

$$\bar{f}: \mathbb{R}^n \to \mathbb{R} \cup \infty$$

$$\bar{f}(x) = \begin{cases} f(x), & x \in \mathcal{D} \\ \infty, & x \notin \mathcal{D} \end{cases}$$

- This is convenient as we do not have to always explicitly state the domain and we can do arithmetic with functions without explicitly defining the intersection of their domains, etc.
- The extension is convex, allowing extended arithmetic and ordering in the definition!

# Convex Functions

**Examples:**

- Exponents: $e^{ax}$ is convex on $\mathbb{R}$ for any $a$
- Powers: $x^a$ is convex on $\mathbb{R}_+$ for $a \geq 1$ or $a \leq 0$ and concave for $a \in [0, 1]$
- Powers of absolute value: $|x|^p$ is convex on $\mathbb{R}$ for $p \geq 1$
- Logarithms: $\log x$ is concave on $\mathbb{R}$

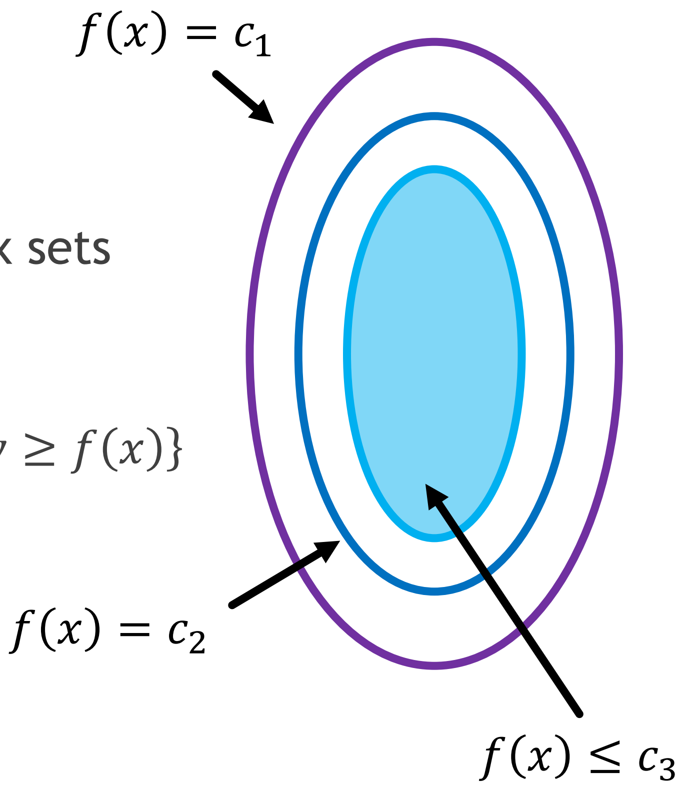
# Convex Functions

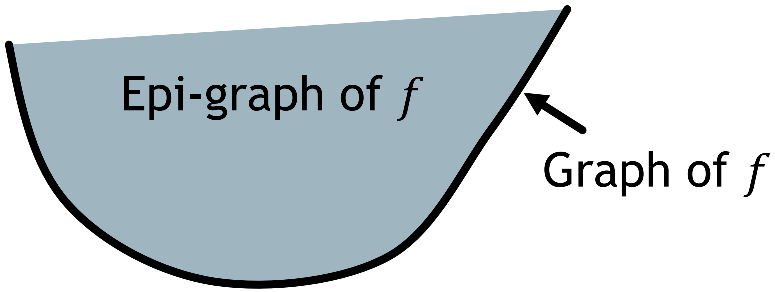
**(Some) connections to convex sets:**

- Sub-level sets of convex functions, are convex sets
- **Definition**: the epi-graph of $f$:

$$\mathrm{epi}(f) = \{(x, y) \in \mathbb{R}^{n+1} : y \geq f(x)\}$$

- A function is convex if and only if its epi-graph is a convex set



# Overview of Algorithms

- **Line search methods:**
  - At the current iterate $x_k$ find a search direction $p_k$
  - Along the search direction $p_k$ find a new iterate $x_{k+1}$ such that the objective function value is lower
  - The step length $\alpha > 0$ along the direction $p_k$ is selected by ***approximately*** solving the (univariate!) minimization problem:

$$\min_{\alpha>0} f(x_k + \alpha p_k)$$

# Overview of Algorithms

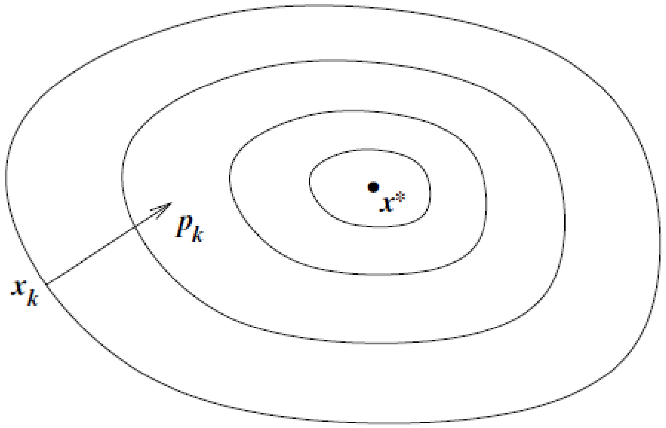


Fig: Nocedal & Wright, Ch02

**Two first examples of search directions:**

- The direction of gradient descent: $p_k \coloneqq -\nabla f(x_k)$
- The Newton direction: $p_k \coloneqq -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$

- The direction of gradient descent gives rise to the family of gradient descent methods, and is a special cast of the direction of steepest descent (more on that later)
- The Newton direction solves the second order appx. minimization problem – we will understand it next week

# Overview of Algorithms

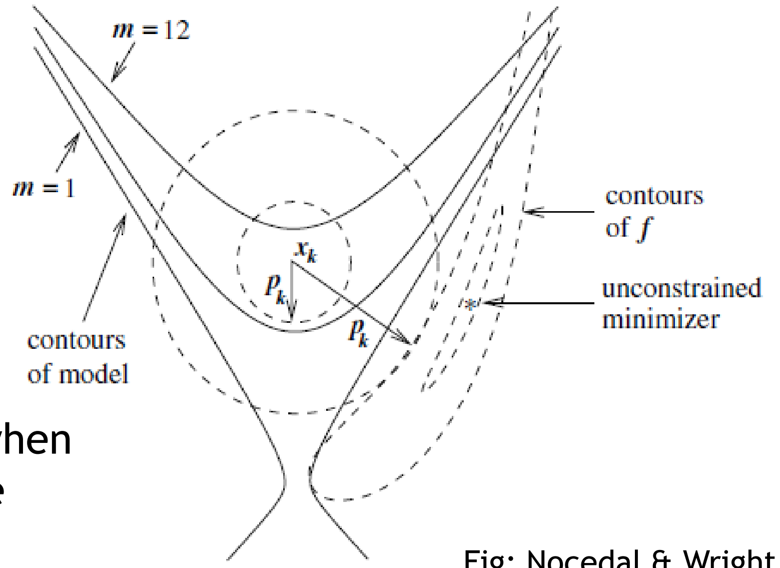
- Trust region methods:
  - At the current iterate $x_k$ construct a model $m_k$ of the objective function $f$
  - The model is similar to $f$ in near $x_k$, and may not be a good approximation far from $x_k$
  - Restrict the search for a minimizer of $m_k$ to some region around $x_k$, namely find a candidate step $p$ to obtain $x_{k+1} = x_k + p$ by ***approximately*** solving:

$$\min_p m_k(x_k + p)$$

where $x_k + p$ lies in the trust region

# Overview of Algorithms

- **Two examples of models and trust regions:**
  - Minimize the first order approximation of $f$ at $x_k$ in a Euclidean ball of radius $\Delta_k$
  - Minimize the second order approximation of $f$ at $x_k$ in a Euclidean ball of radius $\Delta_k$




Note: Directions may vary when region is smaller (unlike line search methods)

Fig: Nocedal & Wright, Ch02

# Gradient Descent – a First Naïve Version

```
def gradient_descent(obj_func, x0, alpha, max_iter):
    x_prev = x0
    f_prev, df_prev = obj_func(x0)
    i = 0
    success = False
    while not success and i <= max_iter:
        x_next = x_prev - alpha * df_prev
        f_next, df_next = obj_func(x_next)
        i += 1
        success = check_converge(x_next, x_prev,
                                 f_next,f_prev, df_next)
    return x_next, success
```

# Gradient Descent – a First Naïve Version

**Discussion:**

- Step size (now fixed `alpha`)
- Convergence:
  - Does the algorithm converge?
  - If so, to what point?
  - If so, at what rate?
- Easy/hard setups for the algorithm? Coordinate scaling, etc.
- Termination conditions?