THE UNIVERSITY OF
CHICAGO

# Stat 310: Numerical Optimization

Mihai Anitescu

# Section 1: intro

# 1.1.Course Logistics

Course Website: canvas.

https://canvas.uchicago.edu/courses/54783

- Instructors: Mihai Anitescu (direct canvas message MUCH preferred, but anitescu@uchicago.edu)

- Course Assistants: Runxin Ni and Hongli Zhao (for emails see canvas)

- Their office hours are  TBD (though in the past the TA s held sessions on Tuesdays late afternoon/evening, since HWK is due on Wednesdays).

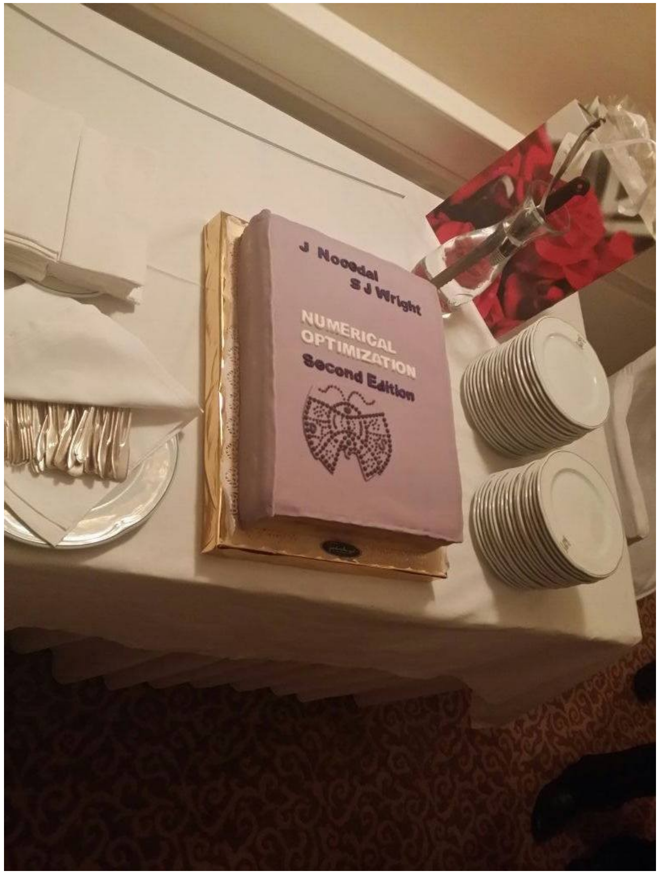
# About class organization and schedule

- Schedule
  - MW 1:30 – 2:50
  - Office Hour: M 3:00-4:00 (in flux)
  - Office Hour online: M 6-7 (zoom link to be posted).
- Online Material
  - Virtually all the instructions, syllabus, code, homeworks and most lecture slides should be posted on website/canvas.
  - Things change often (I find errors or typos in homework for example), so look at the online versions often.
  - I will *try* to post slides before using them in class so you can use them for notes.

# Book

- Book
  - Nocedal and Wright, freely available online in PDF from UC IP
  - (but I strongly recommend you get your own copy, at least get the $25 one from Springer)
  - For nonlinear programming (this class’ focus) it is probably the most used graduate textbook.

- The book has a devoted following ...



# Assignments and Exams

- Assignments (50%)
  - Combination of theoretical problems and computer projects using Matlab .
  - About 1 assignment per week.
  - May also contain you finishing theoretical arguments from class.
- Midterm exam (25%)
  - in-class midterm, probably open books and notes.
  - Mostly theory and conceptual questions
  - Probably in the 5-6th week.
- Final (25%)
  - Take-home, computing-intensive.
  - Due the day when the final is scheduled. (but no in-class final)

# What is Optimization?

- In essence, it is Min_{x \in X} f(x)
- However, the features and properties of X and f matter.
- X can be discrete, continuous or mixed.
- X most often represented as a mix of equality and inequality constraints with structural functions g,h.
  - Min_{x \in X, g(x)<=0, h(x)=0} f(x)
- f,g,h can be any combination of convex-nonconvex, linear-nonlinear.
- X can be R^n or some other set.
- There can be additional structure.
  - Scenario structure, stochastic programming (or nonlinear least squares)
  - h(x)=0 can contain the structural equation of an ODE (control) or PDE.

# Our object of study-Nonlinear Programming

$$\boxed{\begin{array}{ll} \min & f(x) \\ \text{s.t.} & c(x)=0,\ h(x)\le 0 \end{array}} \quad \text{(or)} \quad \boxed{\begin{array}{ll} \min & f(x) \\ \text{s.t.} & c(x)=0,\ h(x)+y=0 \\ & y \quad 0 \end{array}} \quad \text{(or)} \quad \boxed{\begin{array}{ll} \min & f_1(z) \\ \text{s.t.} & c_1(z)=0 \\ & x \in K = \mathbb{R}^n \times \mathbb{R}^{+m} \end{array}}$$

- The variables *y* are called slacks.
- In the latter case, the “data” functions *f,c,* are not identical with the 2 preceding cases.
- We will assume f,g,h are twice continuously differentiable (sometimes 3 times)
- The problem is called **nonlinear** when either *f* or *(c,h)* or both are nonlinear.
- **The above is a powerful modeling paradigm, in which many problems may be rephrased or approximated, though it is important to exploit the particularities of the problem – the “structure”.**
- **Cousins: game theory/complementarity and nonlinear equations**

# Hierarchy of Optimization Encapsulations – Example: Augmented Lagrangian Approach

$$\min_{x,y;c(x)=0,h(x)+y=0} f(x) \Rightarrow$$

$$\min_{x,y\geq 0} f(x) + \lambda^T c(x) + \nu^T (h(x)+y) + c\|c(x)\|^2 + c\|h(x)+y\|^2$$

- Nonlinear Program with Inequality Constraints
- Nonlinear Program (e.g equality as well) with Bound Constraints (bad idea if convex)
- Minimization with Bound Constraints (~, Lagrange Theory)
- Unconstrained Minimization (~, Gradient Projection)
- Linear Algebra (~, Newton’s method), the actual computation workhorse.
- 1D Optimization (~, line search)
- I mean by “~” solved approximately by ….

CHICAGO | SSA

# The essential algorithm – Newton's method.

- Idea: approximate the problem with the quadratic version at the current iterate (there are constrained versions as well)

$$\min_x f(x) \Rightarrow \min_d f(x+d) \approx$$

$$\min_d f(x) + \nabla f(x)^T d + \frac{1}{2} d^t \nabla^2_{xx} f(x) d$$

$$x^+ = x - \left(\nabla^2_{xx} f(x)\right)^{-1} \nabla f(x)$$

- Solve the resulting quadratic approximation
- Do something with the result to make progress without blowing up

# General Aims of This Class

- Aim: define correct and efficient algorithms for nonlinear programming.
- These are also known as “solvers” of nonlinear programming
- Study the problem features that make the algorithmic choices to deal with the approximations ~ defensible, robust, and efficient.
- Will study:
  - Theory (more focus on the bottom of the hierarchy, with a bit of linear algebra) for unconstrained (~ ½ of class) and constrained NLP.
  - Algorithms (everything but linear algebra, with few exceptions)
  - Computations (Matlab ), though mostly for validation and not study in itself.

# Case Study of Computational Improvement: Empirical Bayes.

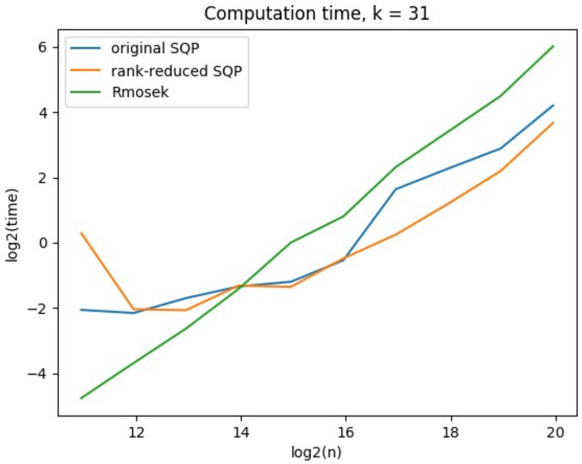
- We need to solve the problem

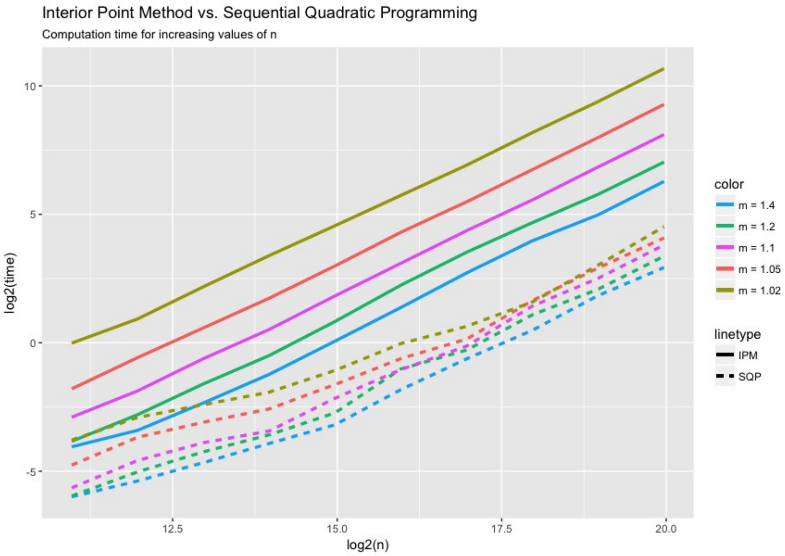
$$\begin{aligned} &\text{minimize} \quad && -\frac{1}{n}\sum_{j=1}^{n} \log\left(\sum_{k=1}^{m} L_{jk}x_k\right) + \sum_{k=1}^{m} x_k \\ &\text{subject to} \quad && x \in \mathbb{R}_+^m = \{x \in \mathbb{R}^m : x \succeq 0\} \end{aligned}$$

- Here L is the data matrix (can be up to m=10^3 and L=10^7).
- In the widely used REBayes package this is done by a call to the convex optimization solver MOSEK (one of the best around; with 20 years of development behind it)

# Reduction of Compute Time

- First, change the algorithm.
- Then, change the linear algebra.
- Code in Julia to get almost C-like performance (with Matlab-like expressivity).
- We started work in June 2017.
- The first results – July 2017
- The final results – 2018- a factor of 100 overall (for m=700) or so over a professional code.
- It is not always this stellar, but choosing and controlling the algorithm matters!





# What is needed?

- What is needed (even for many research topics)?
  - Multivariate Calculus (and, perhaps, some analysis too,)
  - Linear Algebra
- What is nice to have a grasp of?
  - Some geometric intuition (particularly for convex bodies)
  - Some programming experience.
  - Matlab is one of the easiest things to pick up without programming experience, but some makes things faster;
- Check SIAM Journal on Optimization. There is recent research published in nonlinear programming that is accessible only with MV calculus and linear algebra.

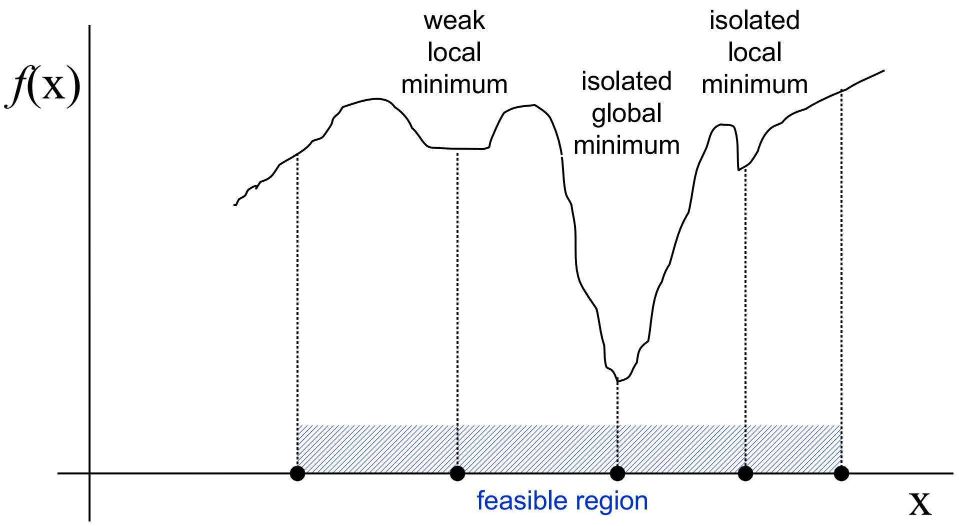
  Victor M. Zavala and Mihai Anitescu. "Scalable Nonlinear Programming Via Exact Differentiable Penalty Functions And Trust-Region Newton Method", *SIAM J. Optim.*, 24(1), 528–558, 2014. DOI: http://dx.doi.org/10.1137/120888181
- Much of the research is at the interface of optimization and other things (dynamics, data), but still MVC and LA are the core.

THE UNIVERSITY OF CHICAGO

# Section 2: Introduction to Unconstrained Optimization

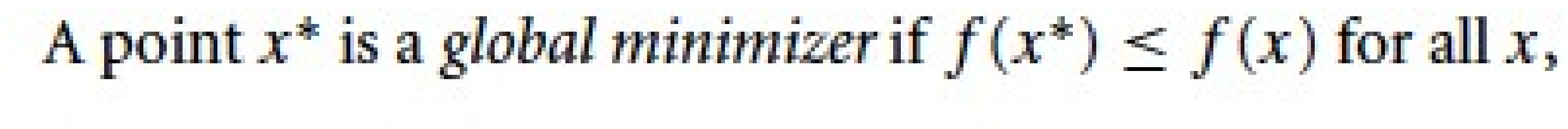
# 2.1 WHAT IS A SOLUTION

# Types of minima

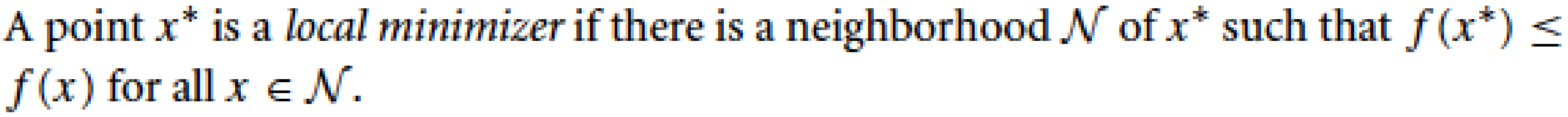


- which of the minima is found depends on the starting point
- such minima often occur in real applications

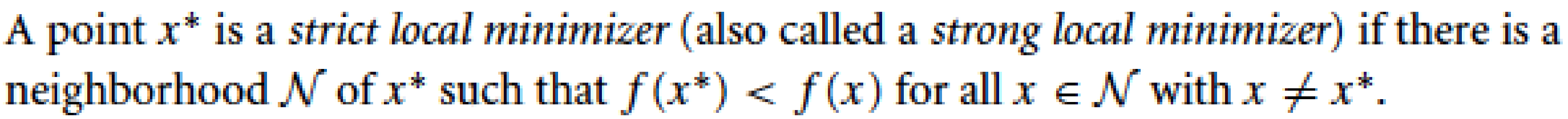
# Types of minima



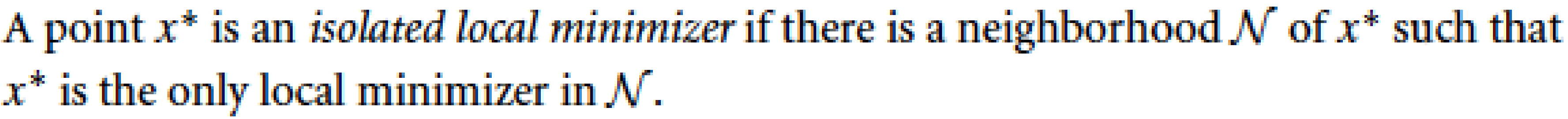
A point $x^*$ is a *global minimizer* if $f(x^*) \leq f(x)$ for all $x$,



A point $x^*$ is a *local minimizer* if there is a neighborhood $\mathcal{N}$ of $x^*$ such that $f(x^*) \leq f(x)$ for all $x \in \mathcal{N}$.



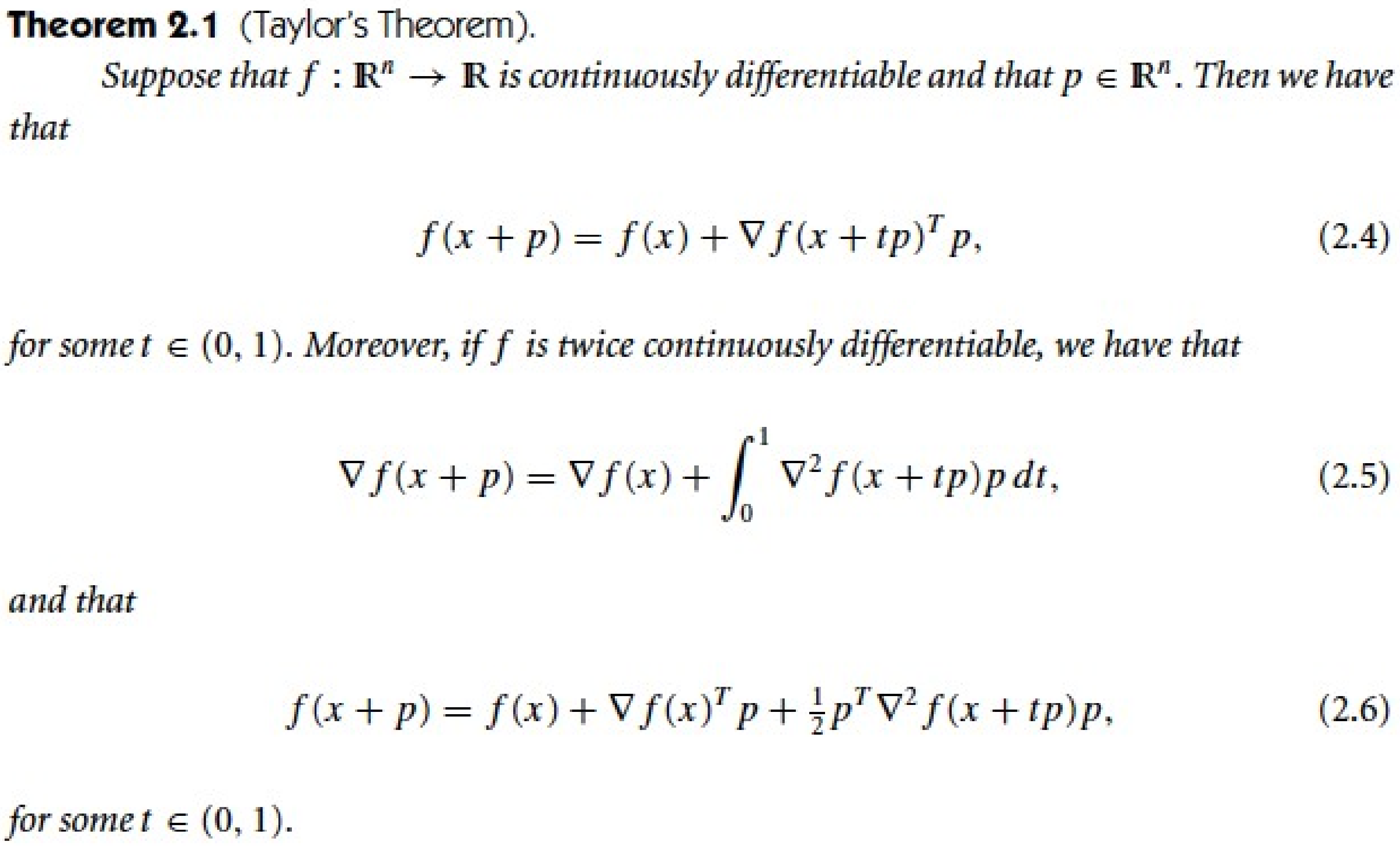
A point $x^*$ is a *strict local minimizer* (also called a *strong local minimizer*) if there is a neighborhood $\mathcal{N}$ of $x^*$ such that $f(x^*) < f(x)$ for all $x \in \mathcal{N}$ with $x \neq x^*$.



A point $x^*$ is an *isolated local minimizer* if there is a neighborhood $\mathcal{N}$ of $x^*$ such that $x^*$ is the only local minimizer in $\mathcal{N}$.

# 2.2 OPTIMALITY CONDITIONS

# Taylor's Theorem



**Theorem 2.1** (Taylor's Theorem).

*Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and that $p \in \mathbb{R}^n$. Then we have that*

$$f(x+p) = f(x) + \nabla f(x+tp)^T p, \tag{2.4}$$

*for some $t \in (0,1)$. Moreover, if $f$ is twice continuously differentiable, we have that*

$$\nabla f(x+p) = \nabla f(x) + \int_0^1 \nabla^2 f(x+tp)p\,dt, \tag{2.5}$$

*and that*

$$f(x+p) = f(x) + \nabla f(x)^T p + \tfrac{1}{2} p^T \nabla^2 f(x+tp)p, \tag{2.6}$$

*for some $t \in (0,1)$.*

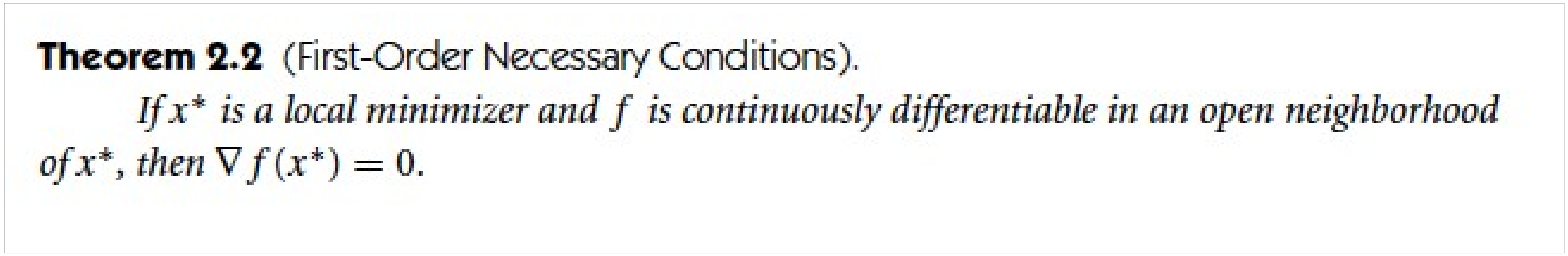
# Strict Local Minimizers are not Isolated

- Example:

$$f(x) = x^4 cos\left(\frac{1}{x}\right) + 2x^4, \quad f(0) = 0$$

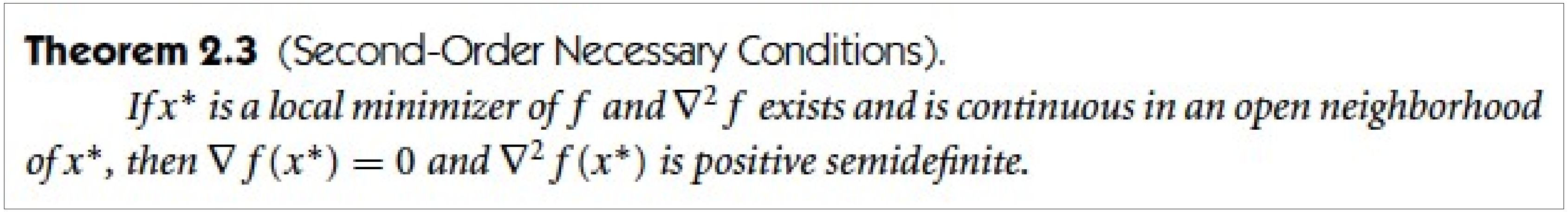
- … but all isolated local minimizers are strict.

# Optimality Conditions



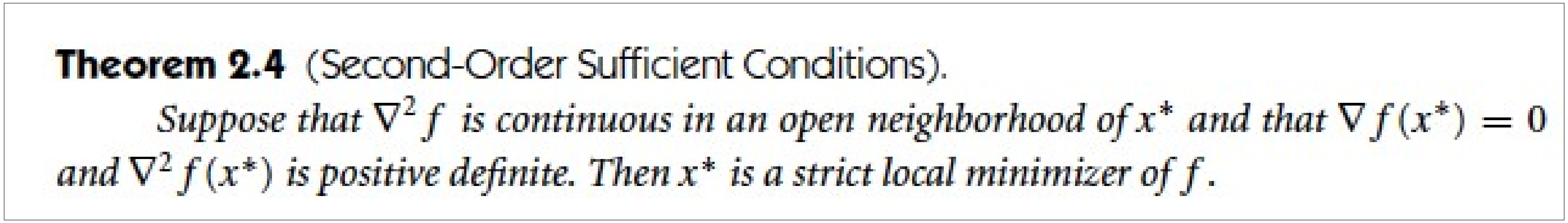
**Theorem 2.2** (First-Order Necessary Conditions).

*If $x^*$ is a local minimizer and $f$ is continuously differentiable in an open neighborhood of $x^*$, then $\nabla f(x^*) = 0$.*



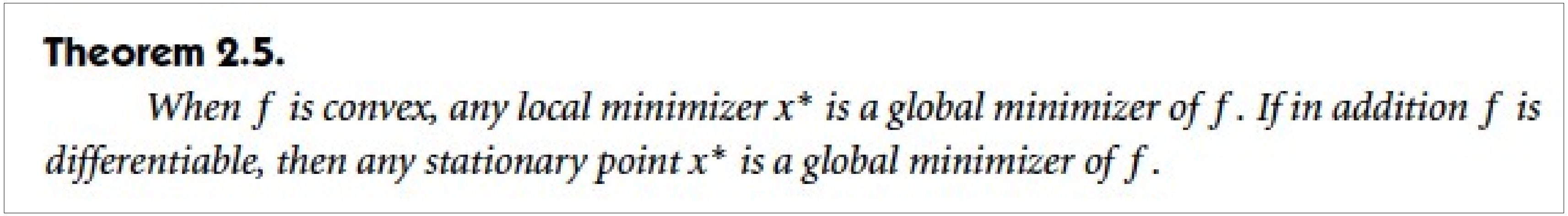
**Theorem 2.3** (Second-Order Necessary Conditions).

*If $x^*$ is a local minimizer of $f$ and $\nabla^2 f$ exists and is continuous in an open neighborhood of $x^*$, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite.*



**Theorem 2.4** (Second-Order Sufficient Conditions).

*Suppose that $\nabla^2 f$ is continuous in an open neighborhood of $x^*$ and that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. Then $x^*$ is a strict local minimizer of $f$.*



**Theorem 2.5.**

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

# Summary LOCAL optimality conditions

- Conditions for *local* minimum of unconstrained problem:

$$\min_x f(x), \quad f \in \mathcal{C}^2$$

- First Order Necessary Condition $\nabla f = \mathbf{0}$
- Second Order Necessary Condition: $\nabla_{xx}^2 f(x) \succeq 0$

- Second Order Sufficient Condition: $\nabla_{xx}^2 f(x) \succ 0$

- Proof: by reasoning on every line passing through the solution.

# How about global optimality?

- There is no simple criterion; extremely hard question (most such problems are NP hard).
- One exception f is convex:

$$\nabla^2_{xx} f(x) \succeq 0, \ \forall x \in \mathbb{R}^n$$

- But we will consider the general case.

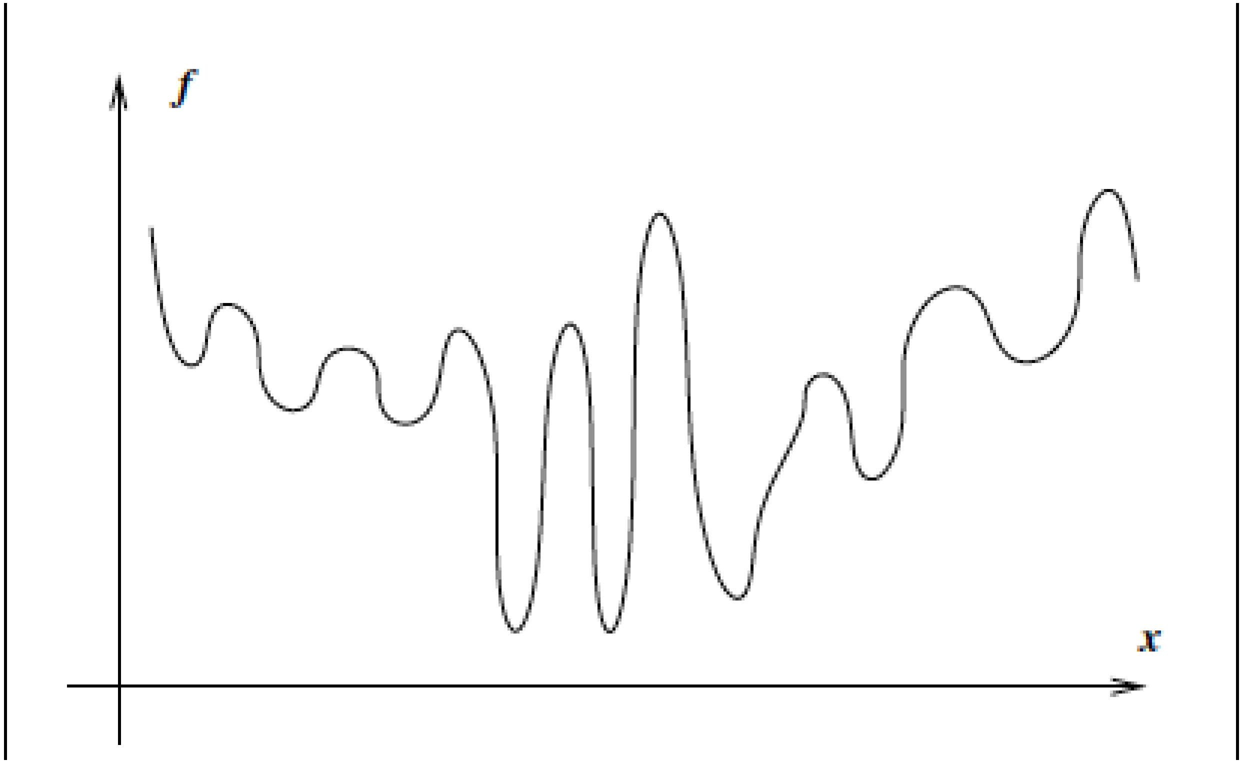
# Consequence:

- If function f is twice continuously differentiable, then solving for a local solution is getting close to solving a differentiable system of nonlinear equations:

$$x = \arg\min f \Leftrightarrow \nabla f(x) = 0$$

- For such situations we have Newton’s method (next section) and we replace an analytical problem with a linear algebra one (which we have some ideas how to solve).
- However, we cannot hope to get global minima this way, in general.

# Difficulty of finding a global minimum



# 2.3 RATES OF CONVERGENCE

# Orders of convergence: Q convergence

- What separates iterative method? Speed of convergence for one.
- Q-convergence – Quotient convergence
- Q-linear $\dfrac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \le r, \quad \text{for all } k \text{ sufficiently large.}$
- Q-superlinear $\displaystyle\lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0.$
- Q-quadratic $\dfrac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} \le M, \quad \text{for all } k \text{ sufficiently large,}$

# Order of Convergence: R-convergence

- The distance to solution is dominated by Q-convergent sequence.

$$\|x_k - x^*\| \leq v_k, \quad v_k \text{ converges } \mathrm{Q} - \text{order}$$

Where order is linear, superlinear or quadratic.

- The origin of the name comes from root convergence, for linear it implies that

$$\limsup \sqrt[k]{\|x_k - x^*\|} \leq r < 1$$

# Course objectives for unconstrained optimization

- Derive efficient iterative algorithms to “solve” the problem(and its constrained form) “fast “.

$$\min_{x} f(x), \quad f \in \mathcal{C}^2$$

- Solve.1 = guarantee convergence to a point that satisfies the first-order NECESSARY conditions.
  Solve.2 = guarantee convergence to a point that satisfies the second-order NECESSARY conditions.
- Fast.1=Typically, if point also SUFFICIENT, then local convergence should be Netwton-like (e.g. quadratic or superlinear),
- Fast.2=Make sure the linear algebra is efficient and fits with the optimization (e.g. solving for the Newton direction results in DESCENT).

# 2.4 OVERVIEW OF ALGORITHMS

# The fundamental building blocks

- First-order methods:
  - Compute gradients.
  - Those are descent directions for the function, go along them to make progress.
- Second-order methods
  - Compute Hessians
  - Rely on the fact that certain subproblems are tractable (I.e we know how to solve efficiently)

- Tractable subproblems

  Linear systems of equations

  - e.g Newton’s direction

$$\nabla^2_{xx} f(x) d + \nabla f(x) = 0$$

  - If the matrix is positive definite, d is descent direction

  Quadratic optimization problems with ONE quadratic constraint

  - e.g.

$$\min f(x) + \nabla f(x)^T d + \frac{1}{2} d^t \nabla^2_{xx} f(x) d$$

$$\text{s.t.} d^t d \leq \rho^2$$

# Line Search Versus Trust Region

- Line search:
  - Find a ”good” direction p_k.
  - Need to be guaranteed to decrease f, at least for a while.
  - Solve approximately the 1D minimization problem

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

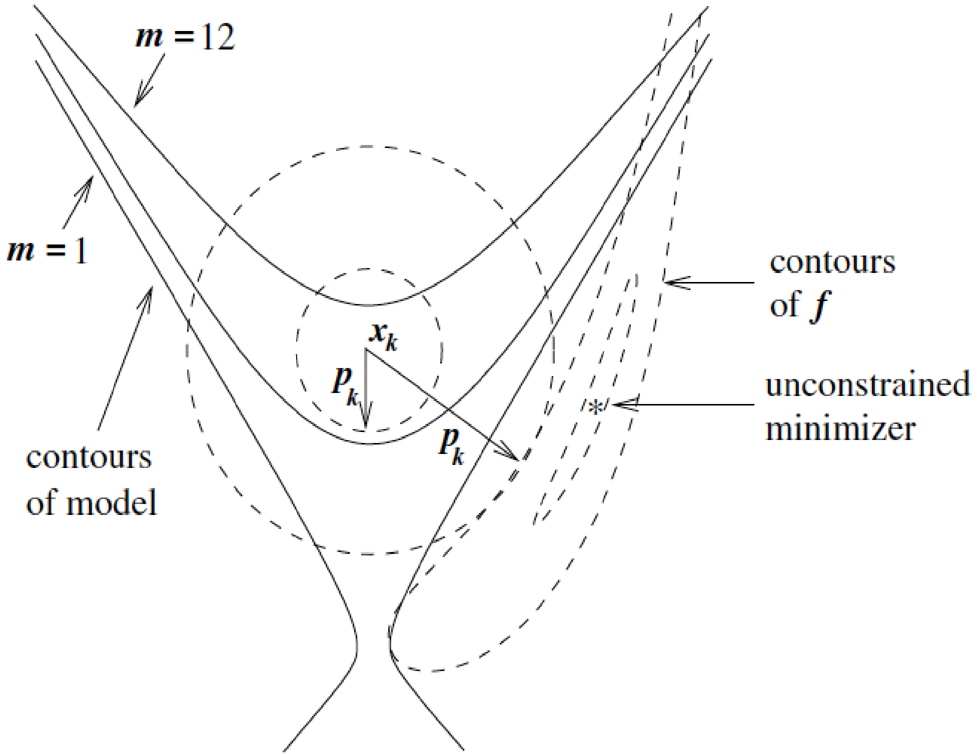
- Trust Region
  - Construct a quadratic model of the problem.
  - Minimize the model subject to a trust-region constraint (most commonly of spherical or elliptical shape)
  - Manage the trust-region

$$m_k(x_k + p) = f_k + p^T \nabla f_k + \tfrac{1}{2} p^T B_k p$$

- Both are iterative methods

# Trust-region

- Note that for the trust-region, the "search" direction depends on the size of the trust-region.
- Particularly pronounced is the situation of far from spherical level sets and model countours



# Line Search: Steepest Descent

- Taylor series:

$$f(x_k + \alpha p) = f(x_k) + \alpha p^T \nabla f_k + \tfrac{1}{2}\alpha^2 p^T \nabla^2 f(x_k + tp)p, \qquad \text{for some } t \in (0, \alpha)$$

- The fastest *immediate* descent direction is the solution of the problem

$$\min_p \; p^T \nabla f_k, \qquad \text{subject to } \|p\| = 1.$$

- Of course, that is proportional to the negative gradient and is called the “steepest descent direction”
- Pluses: needs only first derivative information. Minuses: horribly slow (zig/zag effect). Q-linearily convergent.
- However, it is the basis of other methods, so it is studied extensively.

# Newton Direction

- For line step to work, all we need is a descent direction and not the steepest one (which may be bad)
- You want to ”point to the minimum”
- Approximate the function with a quadratic model, and that is Hessian is positive definite (true in the limit).

$$f(x_k + p) \approx f_k + p^T \nabla f_k + \tfrac{1}{2} p^T \nabla^2 f_k p \overset{\text{def}}{=} m_k(p).$$

- Its minimizer is the Newton direction and it is a descent direction.

$$p_k^{\mathrm{N}} = -\left(\nabla^2 f_k\right)^{-1} \nabla f_k$$

- Pluses: Blazingly fast close to solution. Q-superlinearily convergent.
- Minuses: needs second derivatives, and may not work initially.
- When not doing line search, it is called *Newton's method*.

# The approximate secant property of Hessian

- What if we have no appetite to code or compute the Hessian?
- Idea: try to mimic only *some* properties of the Hessian.
- Do Taylor series *of the gradient*.

$$\nabla f_{k+1} = \nabla f_k + \nabla^2 f_k (x_{k+1} - x_k) + o(\|x_{k+1} - x_k\|)$$

- This indicates that the Hessian has the following important (approximate) secant property

$$\nabla^2 f_k (x_{k+1} - x_k) \approx \nabla f_{k+1} - \nabla f_k$$

- Hint: Look for matrices that have the same approximate relationship

# Secant Condition: Quasi-Newton Methods

- Hint: Satisfy the secant property

$$B_{k+1} s_k = y_k, \ s_k = x_{k+1} - x_k \quad y_k = \nabla f_{k+1} - \nabla f_k$$

- These are quasi-Newton methods (e.g BFGS), give descent directions.

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}. \qquad p_k = -B_k^{-1} \nabla f_k.$$

- Pluses: no second derivatives, superlinear convergence.
- Minuses: storage for large problems.

# Why use trust region

- All descent methods would get stuck at saddle points.
- Including Newton’s method (even if it is not a descent method in that case).
- It is less affected by bad scaling compared with line search.
- *It ends up being the only method that can in principle converge to second-order stationary points.*

# NM/Abstraction/Implementations

- Descent Methods, and Trust-Region methods may be seen as "Newton-Like"
- "Minimizing a quadratic model iteratively"
- All "Newton-like" methods need to solve a linear system of equations.
- All "Newton-like" methods need the implementation of derivative information (unless a modeling language provides it for free, such as AMPL, or is otherwise computed with divided differences, for example). .