Stat 310: Numerical Optimization

Mihai Anitescu
Today

• 1.1 Logistics for the class
• 1.2 Example of Optimization Problems and the Optimization Landscape
• 1.3 Modeling Optimization Problems
• 1.4 The object of continuous Optimization and course objectives.
• 1.5 Newton’s Method.
• 1.6 The role of linear algebra; recap of direct methods.
1.1. Course Logistics

- Instructors: Mihai Anitescu (anitescu@galton.uchicago.edu) and Lek-Heng Lim
- Course Assistant: Yunda Zhong (ydzhong@uchicago.edu)
- The course is divided in 2 parts (tentative);
  - MA: Nonlinear Programming ~ 10 lectures
  - LHL: Convex Optimization ~ 10 lectures
- **Textbook (when needed): Nocedal and Wright:**
1.1 Course Logistics

- **Assignments**
  - Combination of theoretical problems and (computer projects using Matlab)?
  - 4-5 assignments ~ 1 per week = your grade for this part.
- **Office Hours:** TTh, 4:30-5:20, Eck 104 (or by appointment).
- **Web site and contact:** (remember my name 😊).
- **Is email OK for me to communicate with you?**
- **Please answer the survey so that I can get an idea of the background and interest so that I can steer assignments.**
1.2 Context of Optimization

• What is nonlinear optimization optimization?
• Why? Example of optimization based on a subjective criterion.
• Why? Example of optimization problems derived from variational principles in physics.
• Thanks – Sven Leyffer, etc …
Nonlinear Optimization-Nonlinear Programming

$$\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad c(x) = 0, \ h(x) \leq 0 \\
\end{align*}$$ (or)  \hspace{3cm}
$$\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad c(x) = 0, \ h(x) + y = 0, \ y \geq 0 \\
\end{align*}$$ (or)  \hspace{3cm}
$$\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad c(x) = 0, \ x \in K \\
\end{align*}$$

• The variables $y$ are called slacks.

• In the latter case, the “data” functions $f,c$, are not identical with the 2 preceding cases.

• The problem is called **nonlinear** when either $f$ or $(c,h)$ or both are nonlinear.

• The set $K$ may include integrality constraints, MINLP.

• 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”.
ES: “Subjective” criteria of optimization.

- Trying to optimize an user defined criterion: time to accomplish a task, or cost, or matching criterion ….
- Generally a specialist makes a judgment and defines the criterion based on domain-specific knowledge.
- Optimal behavior and design of Engineering, Financial, Management applications
- Examples: structural design and phase problems in crystallography.
ES1: The Airbus wing

(From Sven Leyffer): Optimizing the inboard inner leading edge ribs.
Minimize weight of the structure subject to load and design restrictions
ES1: Final Design—"Truss topology"

STRUCTURE: linear objective function, nonlinear inequality and equality constraints, continuous variables.
ES2: Xray crystallography

• How do we obtain a 3D structure (right) from its diffraction pattern (left)?

• It is essentially the unique high resolution approach to detect protein structure irrespective of size.

• Problem: Find the atomic distributions that minimizes “discrepancy”.
ES2: Phase problem-centrosymmetric

(Sahinidis et al.)

Indices

\( m \) index used for reflections \((m = 1, \ldots, M)\)
\( t \) index used for triplet invariants \((t = 1, \ldots, T)\)

Variables

\( \phi_m \) phase of the \( m \)th reflection
\( \varphi_{m} \) normalized phase of the \( m \)th reflection equal to \( \varphi_{m} = \phi_m + \phi_{m}' + \phi_{m}'' \)
where \( h_m + h_{m}' + h_{m}'' = 0 \)
\( \alpha_t \) binary decision variable
\( \beta_t \) binary decision variable equal to \((1 - \cos \omega_t)\)

Parameters

\( M \) number of reflections
\( n \) number of atoms in the unit cell
\( T \) number of invariants
\( |E_m| \) structure factor amplitude associated with reflection \( h_m \)
\( A_t \) constant equal to \( 2n^{-1/2} |E_{m}| |E_{m}'||E_{m}''| \)
\( \bar{\omega}_t \) conditional expected value of the cosine of the triplet invariant, equal to \( I_1 (A_t)/I_0 (A_t) \)
(Germain et al., 1970)

Model M1

\[
\begin{align*}
\min & \quad f(\beta) = \frac{\sum_{t=1}^{T} A_t \left( 4 \beta_t \bar{\omega}_t + (1 + \bar{\omega}_t^2 - 2 \bar{\omega}_t) \right)}{\sum_{t=1}^{T} A_t} \\
\text{s.t.} & \quad \varphi_m + \varphi_{m}' + \varphi_{m}'' = 2 \alpha_t + \beta_t, \quad t = 1, \ldots, T \quad (1) \\
& \quad \varphi_m \in \{0, 1\}, \quad m = 1, \ldots, M \\
& \quad \alpha_t, \beta_t \in \{0, 1\}, \quad t = 1, \ldots, T.
\end{align*}
\]

STRUCTURE: nonlinear objective, linear equality constraints,
Mixed continuous and integer variables
ES3: Maximum Likelihood for Large Data Sets:

\[
\log p(y \mid \theta) = -\frac{1}{2} W(\theta_1)^T K(\theta_2)^{-1} W(\theta_1) - \frac{1}{2} \log |K(\theta_2)| - \frac{n}{2} \log 2\pi
\]

- **STRUCTURE**: Continuous Optimization, No Constraints.

- **Difficulty**, how do you compute the derivative of the determinant term when you have $10^6$-$10^{12}$ data sites?

- See Anitescu, Chen, Stein: [The ScalaGauss project](https://my.web.page) at my web page.
ESO: Other applications.

- ES4: Stochastic Unit Commitment and Dispatch
- Data assimilation in weather forecasting → PDE constraint
- Image reconstruction from acoustic wave data → PDE constraint
- Crew scheduling, vehicle routing → integer variables
- Reactor core reloading nonlinear → integer variables
- Radio therapy treatment planning → nonlinear integer
- Oil field infrastructure design → PDE c/s & integer var
- Simulation of competition in electricity markets equilibrium c/s.
EV: Variational Description of Phenomena in Physical Sciences

- In these problems, the “state” variable is the solution of an optimization problem, which is formulated based on a law of physics, rather than a subjective criterion.
- In electronic structure computation: the electronic density.
- In complex fluids the (density and species) distributions at thermodynamic equilibrium.
- In Hamiltonian systems, the trajectory is the solution of an optimization problem
- Fermat's variational principle states that a signal in anisotropic media propagates between two points along a curve which renders Fermat's functional $I(l)$ stationary
- ....
EV1: (Thomas-Fermi) Density Functional Theory

**Problem:** For a given atomic configuration, determine electronic density from using the variational principle. STRUCTURE: nonlinear objective, linear constraints, continuous variables.

\[
\min_{\rho} E[\rho, \{R_A\}]
\]

s.t. \( \int \rho = N_e \)

\[
E[\rho, \{R_A\}] = E_{ne}[\rho, \{R_A\}] + J[\rho] + K[\rho] + T[\rho] + V_{nn}\left(\{R_A\}\right)
\]

\[
T[\rho] = C_F \int \rho^{5/3}(r) \, dr, \quad K[\rho] = -C_x \int \rho^{4/3}(r) \, dr
\]

\[
E_{ne}[\rho, \{R_A\}] = \sum_{A=1}^{M} \int \frac{Z_A \rho(r)}{\|R_A - r\|} \, dr, \quad J[\rho] = \frac{1}{2} \int \int \frac{\rho(r)\rho(r')}{\|r - r'\|} \, dr \, dr'
\]
Computations carried out by Peter Zapol et al. from MSD-Argonne, using Kohn-Sham DFT.
• One of the “hot pursuits” in mechanical engineering: Simulating complex phenomena starting from first principles, as opposed to empirical potentials.

• Density Functional Theory based defect nucleation (Carter, Ortiz, et al.)
EV2: Multi-rigid-body dynamics with contact and friction

• A subject pursued by Mihai for some time.

• Essential in the study of robotics, granular materials, pharmaceutical drug processing (powders).

• The velocity of the system at the next step is the solution of the minimum energy problem subject to nonpenetration and frictional constraints.

• Example: the study of size-based segregation in granular materials. STRUCTURE: quadratic objective, quadratic constraints.

\[
v^{(i+1)} = \arg\min_{\hat{v}} \frac{1}{2} \hat{v}^T M \hat{v} + k^{(i)}^T \hat{v}
\]

subject to \[
\frac{1}{h} \Phi^{(j)}(q^{(i)}) + \nabla \Phi^{(j)^T} \hat{v} + \mu^{(j)} d_k^{(j)}^T \hat{v} \geq 0,
\]

\[
j \in \mathcal{A}(q^{(i)}, \epsilon), \quad k = 1, 2, \ldots, m^{(j)},
\]

(16)
Multi-body dynamics simulation: the pebble-bed reactor
1.3 Modeling Optimization Problems

• Here comes the big decision.

• Do we save the same problem over and over
  – Then performance is what matters.
  – Use C++, Fortran, MPI, PERL.

• Do we solve the problem once or only a few times? (e.g. algorithmics class like this)
  – Productivity matters; use higher-level language.
  – Matlab for general scientific computing
  – AMPL or GAMS for optimization

• Domain-specific languages may blur the line …
Modeling: Ingredients

- Objective function
- Variables
- Constraints

Find values of the variables that minimize or maximize the objective function while satisfying the constraints
Isn’t any problem reducible to NLP? Sure, but it is very efficient to recognize and exploit structure.
Modeling (Nonlinear) Optimization Problems

AMPL & GAMS

• high level languages for nonlinear optimization
• interpret problem description, interface to solvers & returns results
• details of solver, derivatives & pre-solve are hidden from user
• modeling language (e.g. var, minimize, subject to, ...)
• programming language (e.g. while, if, ...)
AMPL

• Has an exquisitely simple syntax, reminiscent of C (Kernighan is one of the authors), but adapted to optimization.
• Versions of it can be used even in parallel computation if done wisely.
• AMPL can be run:
  – Student version is free and easy to use.
  – Or you can run it using one of the online servers.
\[
\begin{align*}
\min_{(x,y,z)} \ z \\
\text{subject to} \quad g_0(x,y,z) &= x^2 + 3x^4 - 2y^2 - z &\leq 0, \\
g_1(x,y,z) &= -\frac{1}{2}(x^2 + y^2) + y^4 + 3xy - z &\leq 0, \\
g_2(x,y,z) &= -2x^2 + y^2 - z &\leq 0, \\
g_3(x,y,z) &= -\frac{1}{2}(x^2 + y^2) + x^2 y^2 - 3xy - z &\leq 0.
\end{align*}
\]
var x1;
var x2;
var x3;
minimize objective: x3;
subject to constraint_1: x2**2-2*x1**2+3*x1**4-x3 <= 0;
subject to constraint_2: -2*x2**2+x1**2+x2**4-x3 <= 0;
subject to constraint_3: -0.5*x2**2-0.5*x1**2+3*x1*x2-x3<=0;
subject to constraint_4: -0.5*x2**2-0.5*x1**2-3*x1*x2+x1**2*x2**2-x3<=0;
let x1:=1;
let x2:=1;
let x3:=1;
solve;
display x1, x2, x3;
display constraint_1.dual, constraint_2.dual, constraint_3.dual, constraint_4.dual;
AMPL: standalone

- EXPAND AND DEMO
• **EXPAND.** Discuss taxonomy. How do different solvers behave?
• One-dimensional Thomas-Fermi problem.
• Once you have created the model, you can even run it over the internet with the NEOS server.
**Problem:** For a given atomic configuration, determine electronic density from using the variational principle.

\[
\min_{\rho} E[\rho, \{R_A\}] \\
\text{s.t. } \int \rho = N_e
\]

\[
E[\rho, \{R_A\}] = E_{ne}[\rho, \{R_A\}] + J[\rho] + K[\rho] + T[\rho] + V_{nn}(\{R_A\})
\]

\[
T[\rho] = C_F \int \rho^{5/3}(r) \, dr, \quad K[\rho] = -C_x \int \rho^{4/3}(r) \, dr
\]

\[
E_{ne}[\rho, \{R_A\}] = \sum_{A=1}^{M} \int \frac{Z_A \rho(r)}{\|R_A - r\|} \, dr, \quad J[\rho] = \frac{1}{2} \int \int \frac{\rho(r) \rho(r')}{\|r - r'\|} \, dr \, dr'
\]
#user-defined parameters
param n integer;
param ma integer;
param mg integer;
param dist;
param ratioGap;
param delta;
param cutoff integer;
param pi;
param Z;

#parameters of Thomas Fermi model
param CF:=0.3*(3*pi^2)^(2/3);
param CX:=0.75*(3/pi)^(1/3);
param indexCutoff:=2*(ma+mg)*cutoff;

#total number of nodes
param N:=(n+2)*(2*ma+2*mg);
param ZA{i in 1..n}=Z;

param Nelec:=sum{i in 1..n} ZA[i];

param atomicPosition{i in 1..n}:=dist*i; #atomic positions
param xloc{i in 1..(ma+mg)}:= if i <= ma then (i-0.5)/ma*ratioGap*dist*0.5 else (ma-0.5)/ma*ratioGap*dist*0.5 + 0.5*(1-ratioGap)*dist *(i-ma)/mg;

param xi{i in 1..N}:= if (i-2*(ma+mg)*floor((i-1)/(2*(ma+mg)))-ma-mg-0.5) < 0
then floor((i-1)/(2*(ma+mg)))*dist-xloc[abs(i-2*(ma+mg)*floor((i-1)/(2*(ma+mg)))-ma-mg-1)]
else floor((i-1)/(2*(ma+mg)))*dist+xloc[i-2*(ma+mg)*floor((i-1)/(2*(ma+mg)))-ma-mg];

var x;
var rho{i in 1..N} >=0;
minimize obj: sum\{i in 1..(N-1)\} 0.5*(CF*(rho[i]+delta)^(5/3)-CX*(rho[i] +delta)^(4/3) + CF*(rho[i+1]+delta)^(5/3)-CX*(rho[i+1]+delta)^(4/3))*(xi [i+1]-xi[i])

#Kinetic and Exchange
- sum\{i in 1..(N-1), j in 1..n\} ZA[j]*(rho[i+1]/sqrt((xi[i+1]-atomicPosition [j])^2+delta)+rho[i]/sqrt((xi[i]-atomicPosition[j])^2+delta))*0.5*(xi[i+1]-xi [i])

# Electron-Nucleus
+ 0.5 * 0.25*sum\{i in 1..(N-1), j in max(1,i-indexCutoff)..min(i +indexCutoff,N-1)\} (xi[i+1]-xi[i])*(xi[j+1]-xi[j])* (rho[i]*rho[j]/sqrt((xi [i]-xi[j])^2+delta) + rho[i]*rho[j+1]/sqrt((xi[i]-xi[j+1])^2+delta) + rho[i+1]*rho[j]/sqrt((xi[i+1]-xi[j])^2+delta) + rho[i+1]*rho[j+1]/sqrt((xi[i+1]-xi[j+1])^2+delta)));

# Electron-electron

subject to
constr: sum\{i in 1..(N-1)\} 0.5*(xi[i+1]-xi[i])*(rho[i]+rho[i+1])=Nelec;
Then why study the algorithms at all if modeling is so advanced?

- Some problems CANNOT be solved well by current high-level languages.
  - Problems that have non smooth data (and need to understand limitations of algorithms if approximating them).
  - Problems that use non-intrinsics (e.g. max likelihood).
  - The solvers for the problems we want to solve do not exist (ES4)

- One can take an enormous performance hit if the problem is large or has to be solved many times (ES4).
ES4: Stochastic Unit Commitment with Wind Power (SAA)

\[
\begin{align*}
\text{min} \quad \text{COST} &= \frac{1}{N_s} \sum_{s \in S} \left( \sum_{j \in N} \sum_{k \in T} c_j^p + c_j^u + c_j^d \right) \\
\text{s.t.} \quad \sum_{j \in N} p_{sjk} + \sum_{j \in N_{\text{wind}}} p_{sjk}^{\text{wind}} &= D_s, s \in S, k \in T \\
\sum_{j \in N} p_{sjk} - \sum_{j \in N_{\text{wind}}} p_{sjk}^{\text{wind}} &\geq D_s + R_k, s \in S, k \in T
\end{align*}
\]

- Ramping constr., min. up/down constr.

- Wind Forecast – WRF (Weather Research and Forecasting) Model
  - Real-time grid-nested 24h simulation
  - 30 samples require 1h on 500 CPUs (Jazz@Argonne)
Wind power forecast and stochastic programming

- Unit commitment & energy dispatch with uncertain wind power generation for the State of Illinois, assuming 20% wind power penetration, using the same windfarm sites as the one existing today.

- Full integration with 10 thermal units to meet demands. Consider dynamics of start-up, shutdown, set-point changes.

- Does uncertainty matter? ... Yes. The solution is only 1% more expensive than the one with exact information. Solution on average infeasible at 10%.
Two-stage stochastic programming with recourse ("here-and-now")
Min \( \min_{x_0} \left\{ f_0(x_0) + \mathbb{E} \left[ \min_x f(x, \omega) \right] \right\} \)
subj. to. \( A_0x_0 = b_0 \)
\( A(\omega)x_0 + B(\omega)x = b(\omega) \)
\( x_0 \geq 0, \quad x(\omega) \geq 0 \)
\( \xi(\omega) := (A(\omega), B(\omega), b(\omega), Q(\omega), c(\omega)) \)

Sample average approximation (SAA)
Min \( \min_{x_0, x_1, x_2, \ldots, x_S} \left\{ f_0(x) + \frac{1}{S} \sum_{i=1}^{S} f_i(x_i) \right\} \)
subj. to. \( A_0x_0 = b_0 \)
\( A_kx_0 + B_kx_k = b_k \), \( k = 1, \ldots, S \)
\( x_0 \geq 0, \quad x_k \geq 0, \quad k = 1, \ldots, S \)
Convex quadratic problem

\[
\text{Min } \frac{1}{2} x^T Q x + c^T x \\
\text{subj. to. } A x = b \\
x \geq 0
\]

IPM Linear System

\[
\begin{bmatrix}
Q + \Lambda & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y
\end{bmatrix} = \text{rhs}
\]

Multi-stage SP

Two-stage SP

\[
\begin{bmatrix}
\hat{H}_1 & B_1^T \\
B_1 & 0
\end{bmatrix}
\begin{bmatrix}
\hat{H}_2 & B_2^T \\
B_2 & 0
\end{bmatrix}
\ldots
\begin{bmatrix}
\hat{H}_S & B_S^T \\
B_S & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 \\
A_1 & 0
\end{bmatrix}
\ldots
\begin{bmatrix}
0 & 0 \\
A_S & 0
\end{bmatrix}
\begin{bmatrix}
A^T & 0 \\
0 & A^T
\end{bmatrix}
\begin{bmatrix}
H_0 & A_0^T \\
A & 0
\end{bmatrix}
\]

nested arrow-shaped linear system (via a permutation)

Mihai Anitescu -- Stochastic Programming 39
Do we really need to do our own solver? Well…..:

- AMPL needs 1 week only to preprocess the problem.
- Some instances of the problem have
  - 3 billion variables
  - Needs to be solved in 1 hour
- We need *\times 100 000 CPU-hours
- So only BG/L – Argonne Intrepid – has that kind of power around (100K CPUs), but software for Optimization problems on it did not exist on this scale.
- Plus, the solution MUST SCALE; commercial solutions have not been run on more than hundreds of processors ….
- Conclusion: YES, WE NEED A NEW SOLVER.
The Direct Schur Complement Method (DSC)

- Uses the arrow shape of $H$

$$
\begin{bmatrix}
H_1 & G_1^T \\
H_2 & G_2^T \\
& \ddots \\
H_S & G_S^T \\
G_1 & G_2 & \ldots & G_S & H_0
\end{bmatrix}
= \begin{bmatrix} L_1 & & & & \\
& L_2 & & & \\
& & \ddots & & \\
& & & L_S & \\
L_{i0} & L_{i2} & \ldots & L_{iS0} & L_{ic}
\end{bmatrix}
\begin{bmatrix} D_1 & & & & \\
& D_2 & & & \\
& & \ddots & & \\
& & & D_N & \\
& & & & D_c
\end{bmatrix}
\begin{bmatrix} L_1^T & & & & \\
& L_2^T & & & \\
& & \ddots & & \\
& & & L_S^T & \\
& & & & L_c^T
\end{bmatrix}
$$

- Solving $Hz = r$

\[
L_i D_i L_i^T = H_i, \quad L_{i0} = G_i L_i^{-T} D_i^{-1}, \quad i = 1, \ldots, S,
\]

\[
C = H_0 - \sum_{i=1}^{S} G_i H_i^{-1} G_i^T, \quad L D_c L_c^T = C.
\]

Back substitution

$$w_i = L_i^{-1} r_i, \quad i = 1, \ldots, S,$$

$$w_0 = L_c^{-1} \left( r_0 - \sum_{i=1}^{S} L_{i0} w_i \right)$$

Diagonal solve

$$v_i = D_i^{-1} w_i, \quad i = 0, \ldots, S$$

Forward substitution

$$z_0 = L_c^{-1} v_0$$

$$z_i = L_i^{-T} \left( v_i - L_i^{-T} z_0 \right), \quad i = 1, \ldots, S.$$
Large-scale performance (with Miles Lubin),

- Comparison of ScaLapack (LU), Elemental(LU), and $LDL^T$ (2048 cores)

<table>
<thead>
<tr>
<th>Units (Q+A)</th>
<th>1st Stage Size</th>
<th>Factor (Sec.)</th>
<th>Reduce (Sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$LU$ (S)</td>
<td>$LU$ (E)</td>
</tr>
<tr>
<td>300</td>
<td>23436+1224</td>
<td>16.59</td>
<td>20.04</td>
</tr>
<tr>
<td>640</td>
<td>49956+2584</td>
<td>60.67</td>
<td>83.24</td>
</tr>
<tr>
<td>1000</td>
<td>78030+4024</td>
<td>173.67</td>
<td>263.53</td>
</tr>
</tbody>
</table>

- Strong scaling
  - 90.1% from 64 to 1024 cores;
  - 75.4% from 64 to 2048 cores.
  - > 4,000 scenarios.
  - New DENSE PARALLEL LDLT (implemented by Miles)

SAA problem: $189$ million variables

Total Walltime

Mihai Anitescu -- Stochastic Programming
Argonne Leadership Computing Facility (ALCF) – BG/P system

BG/P Surveyor System
13.6 TF/s 1 rack BG/P
1024 compute nodes
(4096 CPUs)

BG/P Intrepid System
557.1 TF/s 40 rack BG/P
40960 compute nodes
(163840 CPUs)
Results on BG/P - Miles Lubin

- Now include transmission.
- 3B variables, 100K primal variables, 32K scenarios -- one problem solved in about 10 hours – we are working on “real-time” – 1 hour.
1.3 What is state of art in optimization?

- We can solve problems with $10^9$-$10^{12}$ variables LOCALLY.
- We have designed algorithms that make excellent use of massive parallelism (see unit commitment example).
- We can take advantage “smartly” of the latest architectures.
- Commercial or Free Software SOA: $\sim 10^6$-$10^7$ variables and constraints, one processor, rarely multi-threaded. Very few parallel implementations exist (and above this you may not be able to fit in memory due to Hessian size).
1.4 COURSE OBJECTIVES
Types of minima

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

- Conditions for local minimum of unconstrained problem:

  \[ \min_x f(x); \quad f \in C^2 \]

  - First Order Necessary Condition (WHY?): \[ \nabla f = 0 \]
  - Second Order Sufficient Condition:
    \[ \nabla^2 f \geq 0 \]

- EXPAND: Geometry.
  - Second Order Sufficient Condition:
    \[ \nabla^2 f > 0 \]
How about global optimality?

- There is no simple criterion; extremely hard question (most such problems are NP hard).
- One exception $f$ is convex:
  \[ \nabla^2_{xx} f > 0 \quad \text{EVERYWHERE} \]
- But we will consider the general case.
Course objectives

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

\[\min_x f(x); \quad f \in C^2\]

• Solve = guarantee convergence to a point that satisfies the NECESSARY conditions.

• Fast.1 = Typically, if point also SUFFICIENT, then local convergence should be Newton-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).

• NOTE: WE WILL DO NO SIMULATION.