ORF307 – Optimization
22. The role of optimization
Do we have new global upper/lower bounds for each iteration? Or, do we also take into account the local upper/lower bounds from previous iterations?

\[ L^k \leq c^* \leq U^k \]
Final project

- Longer coding exercise (similar to coding in homeworks)
- Topics on the whole course:
  - Least-squares
  - Linear optimization
  - Integer optimization
Today’s lecture
The role of optimization

• Geometry of optimization problems
• Solving optimization problems
• What’s left out there?
• The role of optimization
Basic use of optimization

Optimal decisions

Variables → Objective → Constraints → Decisions

Mathematical language
The algorithm computes them for you
Most optimization problems cannot be solved
Geometry of optimization problems
Least squares

minimize $\|Ax - b\|^2$
subject to $Cx = d$
Least squares

\[ f(x) \]

minimize \[ \| Ax - b \|^2 \]

subject to \[ Cx = d \]
Least squares

\[ f(x) \]

minimize \[ \|Ax - b\|^2 \]

subject to \[ Cx = d \]

Example

\[
\begin{bmatrix}
2 & 0 \\
-1 & 1 \\
0 & 2 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\end{bmatrix}
\approx
\begin{bmatrix}
1 \\
0 \\
-1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
0.6 & 1 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\end{bmatrix}
= \begin{bmatrix} d \\
-0.7 \end{bmatrix}
\]

\[ x^* = (0.05, -0.73) \]
Least squares

\[ f(x) \]

minimize \[ \|Ax - b\|^2 \]

subject to \[ Cx = d \]

**Example**

\[
\begin{bmatrix}
2 & 0 \\
-1 & 1 \\
0 & 2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
\approx
\begin{bmatrix}
1 \\
0 \\
-1
\end{bmatrix}
\]

\[
C
= \begin{bmatrix}
0.6 & 1 \\
1 & 0
\end{bmatrix}
\]

\[
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= \begin{bmatrix}
d \\
-0.7
\end{bmatrix}
\]

\[
x^* = (0.05, -0.73)
\]

**Optimal point properties**

- Minimum point of \( 2x^TA^TAx - 2(A^Tb)^T x \) over subspace \( Cx = d \)
Linear optimization

minimize \quad c^T x

subject to \quad Ax \leq b
\quad Cx = d
Linear optimization

\[ \begin{align*}
    \text{minimize} & \quad c^T x \\
    \text{subject to} & \quad Ax \leq b \\
                        & \quad Cx = d
\end{align*} \]
Linear optimization

\[ f(x) \]
\[ \text{minimize} \quad c^T x \]
\[ \text{subject to} \quad Ax \leq b \]
\[ Cx = d \]

Optimal point properties

- Extreme points are optimal
- Need to search only between extreme points
Duality

Dual function

\[ g(y) \]

Properties

- Lower bound \( g(y) \leq f(x) \)
  (\( x \) primal and \( y \) dual feasible)
- Always concave
  (minimum of linear functions of \( x \))
Duality

**Dual function**
\[ g(y) \]

**Properties**
- Lower bound \( g(y) \leq f(x) \) (\( x \) primal and \( y \) dual feasible)
- Always concave (minimum of linear functions of \( x \))

**Strong duality**
\[ g(y^*) = f(x^*) \]

It holds unless primal and dual infeasible
Optimality conditions

Linear optimization

\begin{align*}
\text{minimize} \quad & c^T x \quad \rightarrow \quad f(x) \\
\text{subject to} \quad & Ax \leq b \\
& Cx = d
\end{align*}

Least-squares

\begin{align*}
\text{minimize} \quad & \|Ax - b\|^2 \quad \rightarrow \quad f(x) \\
\text{subject to} \quad & Cx = d
\end{align*}
**Optimality conditions**

**Linear optimization**

minimize \( c^T x \) \( \leftrightarrow f(x) \)

subject to \( Ax \leq b \)
\( Cx = d \)

\( \nabla f(x^*) + A^T y^* + C^T z^* = 0 \)
\( y^* \geq 0 \)
\( Ax^* \leq b \)
\( Cx^* = d \)

\( y_i^*(Ax^* - b)_i = 0, \quad i = 1, \ldots, m \)

**Least-squares**

minimize \( \|Ax - b\|^2 \) \( \leftrightarrow f(x) \)

subject to \( Cx = d \)

\( \nabla f(x) = 2A^T(Ax-b) \)

**KKT optimality conditions**

**dual feasibility**

**primal feasibility**

**complementary slackness**
Integer optimization

minimize \[ c^T x \]
subject to \[ Ax \leq b \]
\[ x_i \in \mathbb{Z}, \quad i \in I \]

Optimal point properties

• Extreme points are not optimal in general
• If all integral variables, then finite set of solutions
• \( x_i \in \mathbb{Z} \quad \Rightarrow \quad \) Cannot use KKT optimality conditions
Optimality in integer optimization

 certify optimality \[ L \leq c^T x^* \leq U \] return feasible point “incumbent”

Lower bounds from direct relaxation

- Do not give integer feasible \( \bar{x} \)
- Different than the optimal objective \( c^T x^* \)
Optimality in integer optimization

certify optimality \[ L \leq c^T x^* \leq U \] return feasible point “incumbent”

Lower bounds from direct relaxation
- Do not give integer feasible \( \bar{x} \)
- Different than the optimal objective \( c^T x^* \)

Partition = Leaves
Optimality in integer optimization

certify optimality \[ \begin{array}{c} L \leq c^T x^* \leq U \end{array} \]

return feasible point “incumbent”

Lower bounds from direct relaxation
- Do not give integer feasible \( \bar{x} \)
- Different than the optimal objective \( c^T x^* \)

Partition = Leaves

Optimality certificate in integer optimization
- Partition \( S^j \)
- Bounds \( (L_j, U_j) \quad \forall j \)
Solving optimization problems
Numerical linear algebra

The core of optimization algorithms is linear systems solution

\[ Ax = b \]

Direct method

1. Factor \( A = A_1 A_2 \ldots A_k \) in “simple” matrices \( (O(n^3)) \)
2. Compute \( x = A_k^{-1} \ldots A_1^{-1} b \) by solving \( k \) “easy” linear systems \( (O(n^2)) \)
Numerical linear algebra

The core of optimization algorithms is linear systems solution

$$Ax = b$$

Direct method

1. Factor $A = A_1A_2 \ldots A_k$ in “simple” matrices ($O(n^3)$)
2. Compute $x = A_k^{-1} \ldots A_1^{-1}b$ by solving $k$ “easy” linear systems ($O(n^2)$)

Main benefit

factorization can be reused with different right-hand sides $b$
Numerical linear algebra

The core of optimization algorithms is linear systems solution

\[ Ax = b \]

Direct method

1. Factor \( A = A_1 A_2 \ldots A_k \) in “simple” matrices \( O(n^3) \)
2. Compute \( x = A_k^{-1} \ldots A_1^{-1} b \) by solving \( k \) “easy” linear systems \( O(n^2) \)

Main benefit

factorization can be reused
with different right-hand sides \( b \)

You never invert \( A \)
Solving least squares

minimize $\|Ax - b\|^2$
subject to $Cx = d$

KKT linear system solution

$$\begin{bmatrix} 2A^T A & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} x^* \\ z \end{bmatrix} = \begin{bmatrix} 2A^T b \\ d \end{bmatrix}$$
Solving linear optimization

minimize $c^T x$
subject to
$Ax \leq b$
$Cx = d$

$Ax \leq b$
$Cx = d$
Solving linear optimization

minimize \( c^T x \)
subject to
\[
\begin{align*}
Ax & \leq b \\
Cx &= d \\
Ax & \leq b
\end{align*}
\]

No closed form solution

We need an iterative algorithm
Algorithms for linear optimization
Algorithms for linear optimization

Primal simplex
- Primal feasibility
- Zero duality gap
- Dual feasibility
Algorithms for linear optimization

Primal simplex
- Primal feasibility
- Zero duality gap
- Dual feasibility

Dual simplex
- Dual feasibility
- Zero duality gap
- Primal feasibility
Algorithms for linear optimization

- Primal simplex
  - Primal feasibility
  - Zero duality gap
  - Dual feasibility

- Dual simplex
  - Dual feasibility
  - Zero duality gap
  - Primal feasibility

Exponential worst-case complexity
Requires feasible point
Can be warm-started
Algorithms for linear optimization

**Primal simplex**
- Primal feasibility
- Zero duality gap
- Dual feasibility

**Dual simplex**
- Dual feasibility
- Zero duality gap
- Primal feasibility

**Exponential worst-case complexity**
- Requires feasible point
- Can be warm-started

**Dual simplex**
- Dual feasibility
- Zero duality gap

**Interior-point methods**
- Interior condition
- Primal feasibility
- Dual feasibility
- Zero duality gap
Algorithms for linear optimization

**Primal simplex**
- Primal feasibility

**Dual simplex**
- Dual feasibility

- Zero duality gap
- Dual feasibility

**Exponential worst-case complexity**
- Requires feasible point
- Can be warm-started

**Dual feasibility**
- Zero duality gap
- Primal feasibility

**Interior-point methods**
- Interior condition

- Primal feasibility
- Dual feasibility
- Zero duality gap

**Polynomial worst-case complexity**
- Allows infeasible start
- Cannot be warm-started
Linear optimization solvers

• Very **reliable** and **efficient** (many open source)
• Can solve problems in **milliseconds** on small processors
• **Simplex** and **interior-point solvers** are almost a technology
• **Used daily** in almost everywhere
Solving mixed-integer optimization

minimize \( c^T x \)
subject to \( Ax \leq b \)
\( x_i \in \mathbb{Z}, \quad i \in \mathcal{I} \)

Relaxation does not always give feasible solutions

Recursively partition the feasible space
Algorithms for mixed-integer optimization

Branch and bound

Iteratively \textit{branch} and \textit{bound} until $U - L \leq \epsilon$
Mixed-integer optimization solvers

• Can be slow (the only very good ones are commercial)
• Recent huge progress in hardware and software
• Still not a reliable technology
• Used daily in almost everywhere
What’s left out there?
What we did not cover in continuous optimization?

Convex optimization
- Quadratic optimization
- Second-order cone optimization
- Semidefinite optimization
- Convex relaxations of combinatorial problems

Optimization applications
- Stochastic Optimization and ML in Finance (ORF311)
- Modern Control (MAE434)
What we did not cover in machine learning?

Machine learning

- Analysis of big data (ORF350)
- Fundamentals of machine learning (COS424)

Decision-making under uncertainty

- Optimal learning (ORF418)
- Foundations of Reinforcement Learning (ELE524)
The role of optimization
Optimization as a surrogate for real goal

Very often, optimization is not the actual goal

The goal usually comes from practical implementation (new data, real dynamics, etc.)

Real goal is usually encoded (approximated) in cost/constraints
Optimization problems are just models

“All models are wrong, some are useful.”

— George Box
Optimization problems are just models

“All models are wrong, some are useful.”

— George Box

Implications

• Problem formulation does not need to be “accurate”
• Objective function and constraints “guide” the optimizer
• The model includes parameters to tune

We often do not need to solve most problems to extreme accuracy
Data fitting

**Goal** learn model $y \approx f(x)$

from **training data**

$(x^{(i)}, y^{(i)})$ for $i = 1, \ldots, N$

**Data**

- The goal of model is not to predict outcome for *given data* (Train)
- Instead, it is to predict the outcome on *new, unseen data* (Test)
Data fitting

**Goal** learn model

\[ y \approx f(x) \]

from **training data**

\[ (x^{(i)}, y^{(i)}) \text{ for } i = 1, \ldots, N \]

**Data**

- The goal of model is not to predict outcome for *given data* (Train)
- Instead, it is to predict the outcome on *new, unseen data* (Test)

\[ \downarrow \]

- A model generalizes if it makes reasonable predictions on unseen data
- A model **overfits** if it makes poor predictions on unseen data
Regularization as proxy for generalization

Regularized fitting LP

minimize $\|Ax - b\|_1 + \gamma \|x\|_1$
Regularization as proxy for generalization

\[ \text{Regularized fitting LP} \]

\[ \text{minimize} \quad \|Ax - b\|_1 + \gamma \|x\|_1 \quad \text{Proxy} \]
Train vs test error across regularization

Regularized fitting LP

\[
\text{minimize } \|Ax - b\|_1 + \lambda \|x\|_1
\]

- Minimum test error $\lambda \approx 1.15$
- Dashed lines: true values
- $x \to 0$ as $\lambda \to \infty$
Portfolio optimization

Goal: maximize average future returns
\[ \text{avg}(\tilde{R}_w) = \tilde{\mu}^T w \]

from historical returns
\[ T \times n \text{ matrix of asset returns: } R \]
Portfolio optimization

Goal: maximize average future returns
\[ \text{avg}(\tilde{R}w) = \tilde{\mu}^T w \]
from historical returns
\[ T \times n \text{ matrix of asset returns: } \underline{R} \]

Our model generalizes if a good \( w \) on past returns
leads to good future returns

Example
\begin{itemize}
  \item Pick \( w \) based on last 2 years of returns
  \item Use \( w \) during next 6 months
\end{itemize}
Portfolio optimization

Minimize risk-return tradeoff
(on historical data)

minimize $-\mu^T w + \gamma \|Rw - \mu^T w \mathbf{1}\|_1$
subject to $\mathbf{1}^T w = 1$
$w \geq 0$
Portfolio optimization

Minimize risk-return tradeoff
(on historical data)

Returns

\[
\begin{align*}
\text{minimize} & \quad -\mu^T w + \gamma \| R w - \mu^T w 1 \|_1 \\
\text{subject to} & \quad 1^T w = 1 \\
& \quad w \geq 0
\end{align*}
\]
Portfolio optimization

Minimize risk-return tradeoff
(on historical data)

\[
\begin{align*}
\text{Returns} & \quad \text{Risk} \\
\text{minimize} & \quad -\mu^T w + \gamma \| R w - \mu^T w 1 \|_1 \\
\text{subject to} & \quad 1^T w = 1 \\
& \quad w \geq 0
\end{align*}
\]
Portfolio optimization

Minimize risk-return tradeoff
(on historical data)

Returns
\[ -\mu^T w + \gamma \| R w - \mu^T w \mathbf{1} \|_1 \]

Risk
\[ 1^T w = 1 \]
\[ w \geq 0 \]

Proxy

Risk is a proxy to perform well in the future
Past vs future returns on portfolio optimization

Minimize risk-return tradeoff

\[
\text{minimize} \quad -\mu^T w + \gamma \| R w - \mu^T w 1 \|_1 \\
\text{subject to} \quad 1^T w = 1 \\
\quad w \geq 0
\]

- As \( \gamma \to 0 \), more aggressive
- As \( \gamma \to \infty \), risk-averse
- Future is unclear
Conclusions

In ORF307, we learned to:

• **Model decision-making problems** across different disciplines as mathematical optimization problems.

• **Apply the most appropriate optimization tools** when faced with a concrete problem.

• **Implement** optimization algorithms

• **Understand** the limitations of optimization
Optimization cannot solve all our problems
It is just a mathematical model

But it can help us making better decisions

Thank you!

Bartolomeo Stellato