Optimization

Guide to AI and Data Courses · Part V: Mathematical Foundations

Objective, constraints | Linear programming | Convexity | Gradient methods | Where in AI | Neighboring courses

Sercan KÜLCÜ · Giresun University · Computer Engineering

© Sercan Külcü. All rights reserved. · Interactive HTML version of these slides: https://sercankulcu.github.io/files/ai_courses/slides/Chapter_33_Optimization.html

INTRODUCTION

What Question Does It Ask?

> **How do we find the values that make an objective function best?**

Optimization studies how to find the values that minimize or maximize an **objective function** while respecting constraints. Training an AI model means searching for the weights that minimize a loss function, so an optimization method runs inside every learning algorithm.

At a glance

APPROACH mathematical model of objective and constraints	**INPUT** objective function and constraints	**OUTPUT** optimal or near-optimal solution
SUCCESS MEASURE objective value, convergence rate	**MATH LOAD** the theory itself	**DATA NEEDS** none

A Short History

1847
Cauchy proposes steepest descent (gradient descent) for solving systems of equations.

1947
Dantzig develops the simplex method for linear programming.

1951
Kuhn and Tucker publish optimality conditions for constrained problems (KKT).

1957
Bellman's book Dynamic Programming is published.

1984
Karmarkar introduces a polynomial-time interior point method for linear programming.

2014
Kingma and Ba propose Adam, soon widely used to train deep networks.

THE COURSE

Core Concepts

Objective function
The cost or loss to be minimized.

Constraints
Equalities and inequalities a solution must satisfy.

Linear programming
Linear objective and constraints; the simplex method.

Convexity
Every local optimum is also the global optimum.

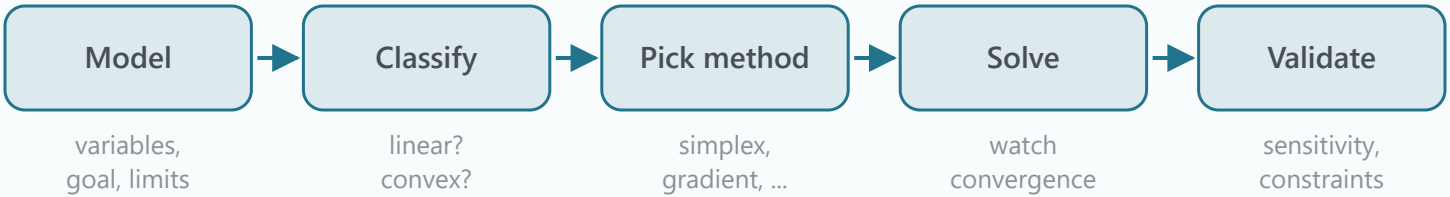
Gradient descent
Small steps against the slope.

Stochastic gradient
Using a small part of the data at each step.

Lagrange multipliers
Folding constraints into the objective; KKT conditions.

Integer programming
Integer variables; often very hard.

A Typical Workflow



Example: Training a neural network is an optimization problem: the variables are the weights and the objective is the average loss. The problem is not convex, so stochastic gradient descent on mini-batches or Adam is used, and the loss curve is watched on a validation set.

THE COURSE

What Does the Course Cover?

1. Formulating problems: variables, objective, constraints
2. Linear programming and the simplex method
3. Duality and sensitivity analysis
4. Convex sets and convex functions
5. Convex problems: QP, SOCP, SDP
6. Optimality conditions: Lagrange and KKT
7. Gradient descent and step size selection
8. Newton and quasi-Newton methods (BFGS)
9. Constrained optimization and interior point methods
10. Stochastic gradient, momentum and Adam
11. Integer programming and branch and bound
12. Dynamic programming

Skills you will gain

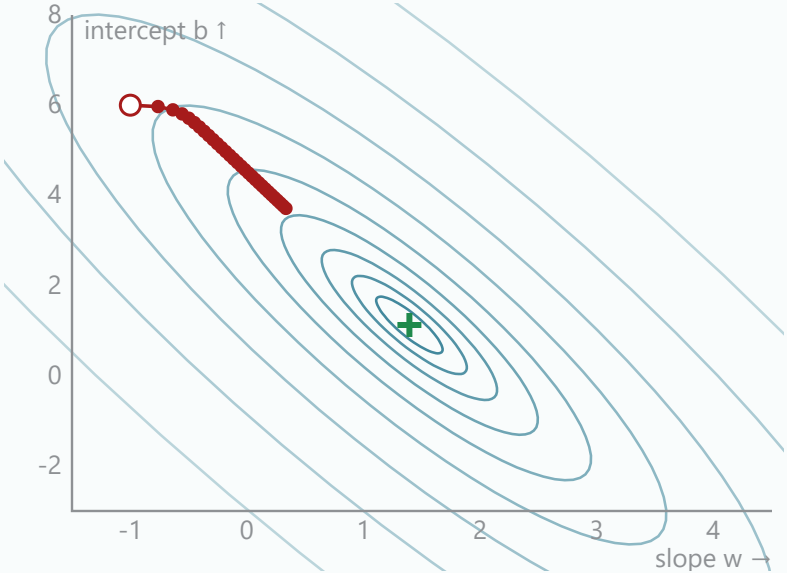
modelling convex problems in CVXPY | writing linear models with PuLP | using scipy.optimize | proving convexity | monitoring and tuning convergence

A typical semester; the order and coverage vary between universities.

TRY IT

Training a Price Model

A model predicts house prices as “price = w·area + b”; gradient descent moves its two weights toward the lowest loss. Click the map to set the start and change the learning rate.



Learning rate η = **0.05**

Raw area | Scaled area

1 step | Run | Reset

Step **30**: loss **1.89** (minimum 0.18)
w = 0.33, b = 3.71

Valley elongation (condition number) **26**; stable only if η < 0.18. It zigzags across the narrow direction and crawls along the long one.

With a small learning rate the model crawls; with a large one it bounces across the narrow valley and diverges. Scaling the features makes the valley round, which is why normalization speeds up neural network training.

CONTEXT

Prerequisites

Courses worth knowing before this one, and why they are needed.

Course	Why it is needed
Linear Algebra	gradient, Hessian, convexity
Calculus	derivatives, partial derivatives and Taylor series: the language of gradients and Hessians
Algorithms	dynamic programming, greedy methods and complexity

CONTEXT

Are You Ready for This Course?

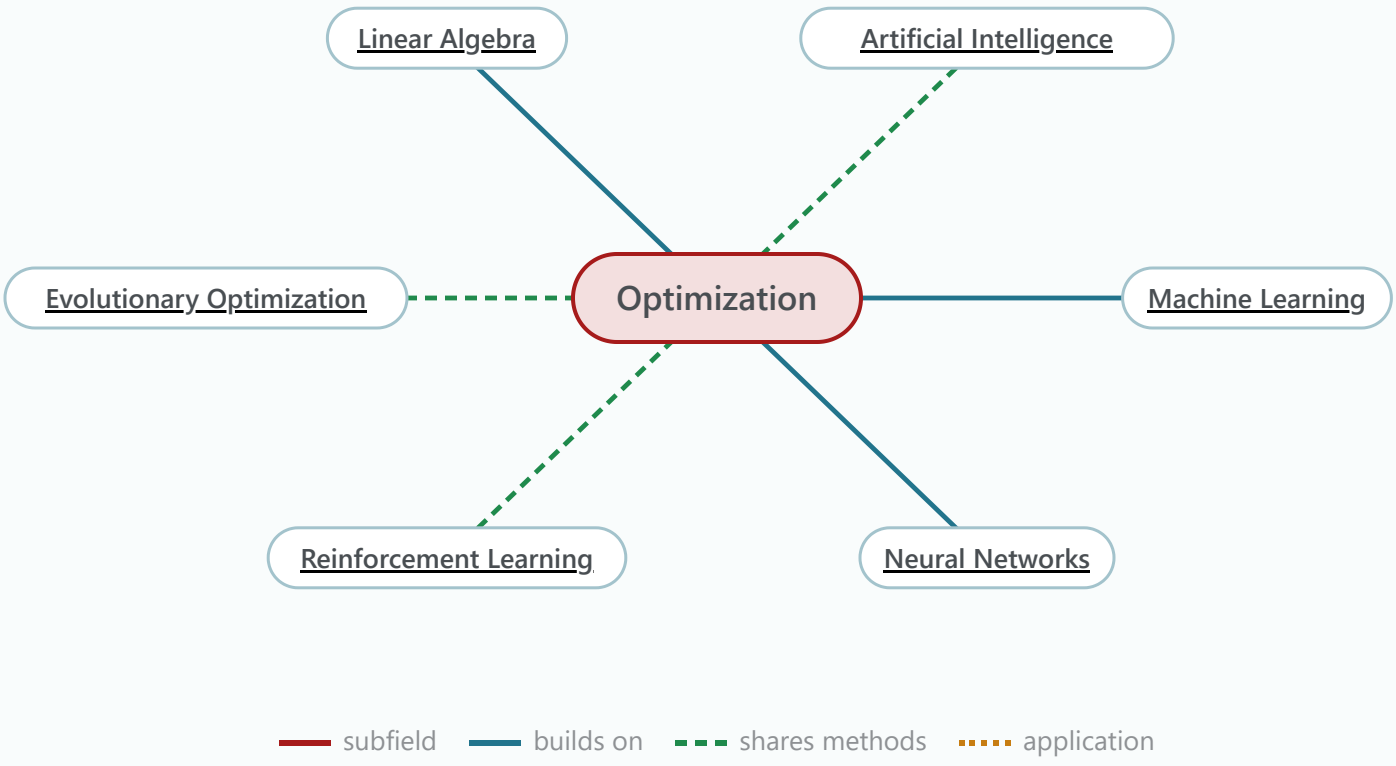
Ask yourself; for every "no", the right-hand column shows where to start.

Ask yourself	If not, start with
Can you compute the gradient of a function of several variables?	Partial derivatives and the chain rule from calculus
Can you write the Taylor expansion of a function up to the second term?	Taylor series from calculus
Can you multiply a matrix by a vector and solve a linear system?	Linear Algebra
Can you update NumPy arrays step by step inside a loop in Python?	An intro programming course, then a short NumPy tutorial

If you can say "yes" to most of these, you are ready; one or two gaps can be closed during the semester.

CONTEXT

Neighbouring Courses



Overlapping Courses

Relation	Course	Shared ground
builds on	Machine Learning builds on this course	loss minimization, regularization
builds on	Artificial Neural Networks builds on this course	gradient descent, learning rate
builds on	This course builds on Linear Algebra	gradient, Hessian, convexity
shares methods	Shares methods with Artificial Intelligence	state-space search, constraint satisfaction
shares methods	Shares methods with Reinforcement Learning	dynamic programming, policy gradient
shares methods	Shares methods with Evolutionary and Metaheuristic Optimization	derivative-free and global optimization

Where It Differs (1/2)

Criterion	Optimization	Evolutionary Optimization	Artificial Intelligence
Core question	How do we find the values that make an objective function best?	How do we find a good solution when there is no gradient or the search space is huge?	How does a machine act intelligently by searching, reasoning and planning?
Approach	mathematical model of objective and constraints	population-based heuristic search	search, logic, probability and learning combined
Input	objective function and constraints	objective, solution encoding	problem definition: states, actions, goal
Output	optimal or near-optimal solution	good, not proven optimal, solution	action sequence, plan or decision
Success measure	objective value, convergence rate	quality, evaluation count	reaching the goal, cost, rationality
Math load	the theory itself	low-medium	medium: logic, probability, graphs
Data needs	none	none; evaluations suffice	little; often a model and knowledge suffice

- **Evolutionary Optimization:** Evolutionary methods search without derivatives; this course teaches methods that exploit problem structure (derivatives, convexity).
- **Artificial Intelligence:** AI starts by modeling a problem as an agent and an environment; optimization solves a mathematically stated problem.

Where It Differs (2/2)

Criterion	Optimization	Machine Learning	Reinforcement Learning
Core question	How do we find the values that make an objective function best?	How does a program generalize from examples?	How can good decisions be learned by trial and error from a reward signal alone?
Approach	mathematical model of objective and constraints	statistical learning	trial and error with rewards
Input	objective function and constraints	labeled or unlabeled examples	environment: state, reward
Output	optimal or near-optimal solution	a model that makes predictions	policy: state → action
Success measure	objective value, convergence rate	error on unseen data	total expected reward
Math load	the theory itself	high: linear algebra, probability, optimization	high: probability, dynamic programming
Data needs	none	moderate to large	generated through interaction

- **Machine Learning:** ML lowers training loss but also wants success on new data; optimization only makes the given objective as good as possible.
- **Reinforcement Learning:** Reinforcement learning maximizes reward by trial and error; here the objective function is written down explicitly from the start.

PRACTICE

Applications

Model training
Finding the weights that minimize the loss.

SVM
The maximum-margin separator: a convex problem.

Hyperparameter search
Choosing learning rate and regularization.

Logistics
Routing, transport and warehouse location.

Scheduling
Shifts, production and timetables.

Portfolio selection
Balancing return against risk.

PRACTICE

Risks and Ethics

The wrong objective

Optimizing what is measurable can defeat the real goal: a system that maximizes clicks alone promotes clickbait.

A forgotten constraint

A constraint left out of the model is ignored: the cheapest route may use a banned road, the best rota may skip legal rest.

Fragile optima

An optimum computed from estimated parameters can degrade badly under small errors; Markowitz portfolios are notoriously sensitive.

Poor scaling

Variables on wildly different scales slow a solver down or can lead it to a false "infeasible" verdict.

In depth: Explainable AI and AI Ethics.

PRACTICE

Tools and Datasets

Tools and libraries

SciPy (scipy.optimize) | CVXPY | Google OR-Tools | PuLP | Gurobi | PyTorch (torch.optim)

Classic example problems

Diet problem | Transportation problem | Knapsack problem | Traveling salesman | Markowitz portfolio | Rosenbrock function

PRACTICE

Common Misconceptions

Common belief	In fact
Gradient descent always finds the best solution.	On non-convex problems it can get stuck in a local optimum or at a saddle point.
The larger the learning rate, the faster.	Too large a step overshoots the minimum and the loss diverges; too small a step drags training out.
Integer problems are solved by rounding.	Rounding the linear solution can break constraints or land far from the optimum.
The lowest loss means the best model.	Driving training loss to zero can mean overfitting; in AI the goal is generalization.

NEXT STEPS

After This Course

Courses that build on or extend this one

Machine Learning | Artificial Neural Networks

Related careers

Operations researcher | Machine learning engineer | Logistics and planning analyst | Quantitative analyst

After this course, how training works in machine learning and neural networks becomes clear. From here you can move to evolutionary and metaheuristic optimization for problems without derivatives, reinforcement learning for sequential decisions, and operations research for allocating resources.

Further Reading

- Boyd and Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- Nocedal and Wright, *Numerical Optimization*, 2nd ed., Springer, 2006.
- Bertsimas and Tsitsiklis, *Introduction to Linear Optimization*, Athena Scientific, 1997.
- Bottou, Curtis and Nocedal, "Optimization Methods for Large-Scale Machine Learning", *SIAM Review* 60(2), 2018.

The End

Chapter 33: Optimization

← Previous chapter · Map of the field · Next chapter →