
LECTURE 1

Lecture Content

The course is designed to be dynamic and ever-evolving, incorporating the latest advancements in the field of machine learning and data science each year. This year, we've introduced new material on state-space models, including advanced concepts such as Mamba and Jamba models. Additionally, we've included the latest developments on transformers, diffusion models, and other cutting-edge topics. While these updates ensure the course remains current and relevant, they also present challenges, such as adjusting to the varying styles and evolving formats of lecture slides.

The lecture slides are central to the course, providing essential results and numerous citations for further exploration. Although lecture scribes from previous years are available, they may not fully align with this year's updates, as the course evolves annually to include new concepts and examples.

1 Lecture Logistics

This is a six-credit course designed with a balanced structure. For the first three weeks, lectures will be held on Mondays and Fridays to establish foundational knowledge before introducing exercises or homework. After this period, Mondays will be reserved for lectures, while Fridays will focus on exercises or homework sessions.

- **Course Format:** A six-credit course with lectures on Mondays and Fridays. Exercises and Jupyter Notebooks are provided for hands-on experience.
- **Lecture Duration:** Monday sessions typically end by 11 a.m., though occasional overruns may occur due to extended discussions or questions.
- **Exercise Sessions:** Initially scheduled from 3 p.m. to 6 p.m., exercise sessions have been shifted to 4 p.m. to 7 p.m. to accommodate other scheduling conflicts. Generally, these sessions conclude around 6 p.m., though teaching assistants (TAs) remain available for further support.
- **Supplementary Materials:** Background material on probability, linear algebra, and complexity notations is available on Moodle.
- **Collaboration Policy:** Students are encouraged to collaborate but must write their own solutions.
- **Grading:** Three homework assignments (2, 1, and 1 points respectively) and a final project (1 point).

Earlier versions of the slides are available for reference. To accommodate the extensive demand, a Zoom channel will also be available for exercise hours. However, it is important to note that online discussions cannot fully replicate the benefits of in-person interactions. Attending exercise sessions in the lab is highly encouraged, as they foster a collaborative and engaging learning environment. These sessions provide an opportunity to discuss problems, exchange ideas with peers, and work together while ensuring that each participant writes their own solutions.

1.1 Prerequisites and Resources

Students are expected to have a basic understanding of mathematical notation, linear algebra, and probability. For those needing a refresher, supplementary materials are available on Moodle to help bridge any gaps in background knowledge.

To enhance learning, the course emphasizes hands-on experience. In the first two weeks, handouts with exercises and their solutions will be provided, along with interactive Jupyter Notebooks for practical exploration. Homeworks, likewise, are structured around Jupyter Notebooks, allowing students to experiment and apply theoretical knowledge. Collaboration and Integrity

The field of machine learning thrives on collaboration, and this course adopts a similar philosophy. Students are encouraged to work in groups, typically of three to five, to discuss and solve problems. However, while collaboration is valued, it is essential that each student submits independently written solutions. This approach fosters both teamwork and individual understanding. Teaching Support

The course is supported by a dedicated team of eight teaching assistants and three postdoctoral researchers, all of whom possess deep expertise in the subject. Students can rely on this team for guidance during lectures, exercises, and beyond. Homework and Grading

The course includes three graded homework assignments:

The first homework is weighted at two points to encourage early engagement with foundational material. The second and third homeworks are each worth one point. The final exam contributes one point to the total grade.

The rationale behind the heavier weighting of the first homework is to provide an opportunity to tackle the relatively simpler material before the semester becomes more demanding with overlapping exams and assignments. While grades matter, the focus of this course is on mastering the material and gaining practical insights into the latest machine learning advancements.

This course is an opportunity to explore, apply, and internalize cutting-edge concepts in machine learning. It's not just about achieving a grade but also about acquiring knowledge and skills that will be invaluable in solving real-world problems.

1.2 Course Overview

This course takes a comprehensive view of the machine learning pipeline, covering how data is generated, learning objectives are defined, and these objectives are optimized to derive numerical solutions. It emphasizes that machine learning, much like life, is multidimensional, with various interconnected factors feeding into the process.

At its core, the course focuses on algorithms—their design, functionality, and impact. Algorithms are explored as systems that take data and loss function models as inputs, process them, and produce meaningful results. However, the course goes beyond mere functionality to address critical considerations such as fairness, ethics, interpretability, and the trade-offs between computational resources (time, compute, storage) and performance.

Students will also evaluate algorithms based on their ability to converge, generalize, and remain robust against perturbations in inputs, models, or other factors, equipping them to understand and tackle real-world machine learning challenges effectively.

2 Introduction to Learning Paradigms

2.1 Supervised Learning

- **Objective:** Predict the label of unseen data based on labeled examples.
- **Examples:** Cancer prediction using DNA data, image classification, multiclass classification in photo tagging and next-word prediction.

2.2 Unsupervised Learning

- **Objective:** Identify patterns or structures in data without labels.
- **Applications:** Clustering and dimensionality reduction.

2.3 Reinforcement Learning

- **Objective:** Optimize an agent's interactions with an environment to maximize cumulative rewards.
- **Examples:** Optimal policy determination in complex systems, as for example in graph neural networks for travel time prediction.

3 An Overview of Statistical Learning by Vapnik

So, this is around 1996. He had this IEEE paper that used this block diagram that kind of explains the core of supervised learning.

3.1 A Basic Statistical Learning Framework

A statistical learning problem usually consists of three core elements:

1. **Generator:** A process that produces samples $a_i \in \mathbb{R}^P$, where a is a random variable with an unknown probability distribution P_a . The generator maps data into a space that we analyze and learn from.
2. **Supervisor:** For each a_i , the supervisor generates a corresponding sample b_i of a random variable b through an unknown conditional probability distribution $P_{b|a}$. This process establishes the mapping between inputs and outputs.
3. **Learning Machine:** A system that aims to approximate the supervisor's mapping by learning a function $h(a_i)$, where $h \in \mathcal{H}$ is part of a fixed function space \mathcal{H} . The goal is to accurately reproduce the relationship between inputs and outputs.

In this framework, the inputs a_i are typically represented as vectors in p dimensions (e.g., \mathbb{R}^P). For example, an image, though not inherently a vector, can be represented as one by choosing an appropriate ordering of its elements. While the probability distribution P_a governing the generator is unknown, this framework does not assume knowledge of it, making it robust for a variety of applications.

The supervisor's role involves producing outputs b_i (or labels) using a conditional probability distribution $P_{b|a}$. These outputs may represent labels, probabilities, or real numbers. Often, a_i and b_i pairs are collectively referred to as the data.

The learning machine's task is to approximate the supervisor's mapping by learning a function $h(a_i)$ from a function space \mathcal{H} . The learned function h aims to match the outputs b_i as accurately as possible.

Applications

- This framework provides the foundation for studying classification, regression, and density estimation problems.
- It allows us to evaluate learning algorithms based on their ability to generalize, converge, and remain robust to variations in input data, models, and other conditions.

Examples

Let's begin with some examples:

- **Cancer Prediction:** In this context, the generator produces DNA data (a_i), which are mathematical encodings of an individual's genetic material. These data may sometimes include mutations, which, unfortunately, can lead to diseases such as cancer. The output (b_i) represents whether or not a patient has cancer.

While there is no literal supervisor in this process, we assume the existence of a mapping—a functional relationship—that links DNA data (a_i) to the presence or absence of cancer (b_i). Our goal as learning machines is to approximate this mapping to enable accurate predictions.

This example illustrates the critical role of statistical learning in healthcare. By learning such mappings, we can assist doctors in diagnosing conditions, tailoring treatments, and developing targeted therapies. For example:

- Early detection of diseases such as cancer, Parkinson's, or diabetes from genetic markers.
- Predicting risks using genetic testing services like 23andMe or Ancestry.
- Helping researchers and clinicians design personalized treatments based on an individual's genetic profile.

Generalizing the Cancer Prediction Example

Beyond cancer prediction, this framework applies to numerous other domains:

- Identifying individuals at risk of Parkinson's or diabetes based on genetic data.
- Predicting outcomes from clinical data to enable targeted drug development.
- Developing tools for doctors to diagnose and intervene effectively.

In this example, we deal with a binary classification problem, where the outcomes are either "yes" (the patient has cancer) or "no" (the patient does not). The broader goal is to leverage these predictive models to make actionable decisions, whether it is aiding doctors in diagnosis or guiding researchers in drug discovery. This highlights the wide applicability of statistical learning frameworks in real-world healthcare challenges.

- **Multiclass Classification: Modern Photo Tools**

Let's consider a practical example of **multiclass classification**: modern photo tagging tools. Applications such as Apple Photos use neural networks to automatically tag photos, enabling users to search for specific individuals or objects in their photo library. For instance, you can search for a person's name, and the tool retrieves all their photos across various ages.

How It Works

This process involves offline classification. A database of tagged photos is used to train a model that learns to map images to labels, such as names of individuals in the photos. The function takes an image as input and outputs one or more labels corresponding to the people in the image. For example:

- A photo might be tagged as containing "Atakan" and "Damla."
- Another photo might include "Volkan" and "Damla."
- Searching for "Damla" will return all relevant photos tagged with her name.

Broader Implications

This example illustrates how classification models learn functions that map inputs (e.g., images) to outputs (e.g., labels). Such functions have applications beyond photo tagging:

- In healthcare, functions map DNA sequences to disease classifications (e.g., cancer or no cancer).
- In object recognition, functions map images to labels for objects, animals, or locations.
- In commercial tools, these models enable seamless search and retrieval of data, enhancing user experience.

Key Features of Multiclass Classification

- Multiclass classification handles multiple possible output labels (e.g., names of people in photos).
- These models often work offline, training on large datasets before being deployed for real-time searches.
- Such models must handle overlapping classes, where multiple labels may apply to a single input.

This example demonstrates the versatility of machine learning models in learning complex functions that solve real-world problems, from healthcare to entertainment and beyond.

• Next Word Prediction

Next word prediction is a classic application of language models, where a system predicts the most likely next word based on the context of preceding words. For instance, tools like Google search provide suggestions as you type, seemingly anticipating your thoughts.

This process involves several key steps:

- Representing words using a predefined vocabulary.
- Creating embeddings for words, which encode them as mathematical vectors.
- Training models to classify the next word (or token) over the vocabulary based on these embeddings.

Modern language models, such as GPT-4, scale this idea significantly. These models, composed of billions of parameters, learn intricate patterns in text and perform at unprecedented levels. By the end of this course, you will gain insights into tokenization, architectures, optimization techniques, and scaling laws that underpin such systems.

• Travel Time Prediction

Consider predicting travel time between two locations, such as from EPFL to a nearby city. The inputs are the start and destination points, and the output is the estimated travel duration.

Machine learning models, particularly those leveraging graph structures, can incorporate:

- Geographic and transportation data.
- Contextual factors such as time of day or metro schedules.
- Dynamic conditions like traffic or delays.

Graph neural networks are often employed to learn mappings between input points and travel durations, accounting for both spatial and temporal contexts.

• House Pricing and Mortgage Risk Assessment

Predicting house prices and assessing mortgage risks are prime examples of supervised learning. In this scenario:

- **Input Features:** Characteristics of the house, such as size, orientation, proximity to transportation, and year built.

- **Output:** A predicted price (in millions) for the house.

Banks and financial institutions use similar models to determine mortgage eligibility and risk:

- Buyers must provide a down payment (typically 20% of the house price).
- The model assesses additional risk factors, such as income and credit history.
- Machine learning systems compute interest rates and monthly payment feasibility.

These applications highlight the utility of machine learning in solving real-world problems, from determining house values to evaluating financial risk.

Broader Implications

Each of these examples illustrates how supervised learning models map inputs to outputs, enabling predictions across diverse domains:

- Language models predict the next word or phrase.
- Graph neural networks estimate travel times by analyzing spatial relationships.
- Regression models predict continuous outcomes like house prices.
- Risk assessment models classify loan applicants based on financial stability.

The overarching theme is the versatility of machine learning in capturing complex relationships and providing actionable insights across varied industries.

• Density Estimation

Density estimation involves learning an unknown probability distribution from empirical samples. In this process:

- Regions with higher probabilities yield more samples.
- Regions with lower probabilities produce fewer samples.

The goal is to learn the underlying distribution to generate new data points or better understand the data's structure.

- **Image Generation:** Models like DALL-E and diffusion models generate images based on textual prompts by sampling from learned distributions. For example:
 - * A prompt such as "students listening to a mathematics of data lecture" might generate realistic images of such a scene.
 - * High-probability images (e.g., typical classroom setups) are generated more often, while rare prompts produce fewer samples.

However, it is important to note the environmental impact of these models. Each image generation can have a carbon footprint equivalent to driving several kilometers, emphasizing the need for efficiency and mindful usage.

- **Uncertainty Quantification in MRI Imaging:** In medical imaging, especially MRI, the machine captures data in the Fourier domain. By leveraging generative adversarial networks (GANs) or similar models, one can:
 - * Quantify uncertainty in the images.
 - * Optimize the sampling process to minimize this uncertainty.

For instance, identifying which Fourier samples are essential can reduce the time a patient spends in the MRI machine, improving comfort and efficiency.

Applications in Practice

- **Image Generation:** Modern models can generate high-resolution images that capture text prompts with remarkable accuracy. These models are continuously advancing in quality and interpretability.
- **Medical Imaging:** Automated systems that reduce sampling redundancy are particularly useful for patients undergoing lengthy or uncomfortable procedures, such as MRI scans.

3.2 Future Topics

We will explore diffusion models and advanced density estimation techniques in greater detail in Lecture 11. For MRI imaging and its Fourier sampling details, refer to Lecture 2.

4 Loss Function

4.1 Definition of a Loss Function

A loss function $L : \mathcal{B} \times \mathcal{B} \rightarrow \mathbb{R}$ on a set is a function that satisfies some or all properties of a metric. Loss functions are used in statistical learning to measure data fidelity, defined as $L(h(a), b)$. In simple terms, a loss function evaluates how well a model's predictions align with the actual data.

For example, if you make a prediction based on some data, the loss function quantifies the error in those predictions. It helps answer the question: "How well is the model performing?" or, for pessimists, "How poorly is the model performing?"

4.2 Definition of a Metric

Let \mathcal{B} be a set. A function $d(\cdot, \cdot) : \mathcal{B} \times \mathcal{B} \rightarrow \mathbb{R}$ is a metric if $\forall b_1, b_2, b_3 \in \mathcal{B}$:

- (a) $d(b_1, b_2) \geq 0$ (nonnegativity),
- (b) $d(b_1, b_2) = 0$ if and only if $b_1 = b_2$ (definiteness),
- (c) $d(b_1, b_2) = d(b_2, b_1)$ (symmetry),
- (d) $d(b_1, b_3) \leq d(b_1, b_2) + d(b_2, b_3)$ (triangle inequality).

Remarks

- A **pseudometric** satisfies (a), (c), and (d) but not necessarily (b). For example, the total variation norm looks at changes in a vector but can return zero for non-zero inputs (e.g., an all-ones vector).
- **Norms induce metrics**, while **pseudonorms induce pseudometrics**. For instance:

$$d(b_1, b_2) = \|b_1 - b_2\|$$

is a metric induced by a norm.

- A **divergence** satisfies (a) and (b) but not necessarily (c) or (d). For instance, Bregman divergences lose symmetry and the triangle inequality.

4.3 Examples of Loss Functions

4.3.1 Logistic Loss

For binary classification tasks, the logistic loss is a surrogate to the zero-one loss. It is defined as:

$$L(b_1, b_2) = \log_2(1 + \exp(-b_1 \cdot b_2)),$$

where $b_1 \in \mathbb{R}$ is a score value, and $b_2 \in \{\pm 1\}$ is the class label. Unlike the zero-one loss, which provides no gradient information, logistic loss is conducive to optimization and helps find the direction for improvement.

4.3.2 ℓ_q Loss

The ℓ_q loss is used to measure distances between points. For $q \geq 1$, it is defined as:

$$L(b_1, b_2) = \|b_1 - b_2\|_q,$$

where the ℓ_q norm is given by:

$$\|b\|_q = \left(\sum_{i=1}^n |b_i|^q \right)^{\frac{1}{q}}.$$

4.3.3 1-Wasserstein Distance

The 1-Wasserstein distance, also known as earth mover's distance, quantifies the effort required to transform one probability distribution into another. For two probability measures μ and ν on \mathbb{R}^d , it is defined as:

$$W_1(\mu, \nu) = \inf_{\pi \in \Gamma(\mu, \nu)} \mathbb{E}_{(x,y) \sim \pi} [\|x - y\|],$$

where $\Gamma(\mu, \nu)$ is the set of all couplings of μ and ν . This distance is particularly useful in applications like generative adversarial networks (GANs) and diffusion models.

Final Remarks

Loss functions are integral to the design and evaluation of machine learning models. While metrics and norms offer structured ways to measure distances, pseudometrics and divergences provide additional flexibility. Understanding the nuances of these mathematical tools is essential for developing effective optimization strategies. For further exploration, refer to the supplementary material on linear algebra and norms.

5 Statistical Learning Model and Risk

5.1 Statistical Learning Model

A statistical learning model consists of the following key elements:

1. A sample of n i.i.d. (independent and identically distributed) random variables $(a_i, b_i) \in \mathcal{A} \times \mathcal{B}$, $i = 1, \dots, n$, drawn from an unknown probability distribution P .
2. A function class \mathcal{H} , which is a set of candidate functions $h : \mathcal{A} \rightarrow \mathcal{B}$.
3. A loss function $L : \mathcal{B} \times \mathcal{B} \rightarrow \mathbb{R}$, which measures the data fidelity between the predicted output $h(a)$ and the actual label b .

The model aims to evaluate and minimize the **risk** of a function $h \in \mathcal{H}$. The risk quantifies the expected error (loss) over the joint distribution of (a, b) .

5.2 Definition of Risk

The (population) risk of a function $h \in \mathcal{H}$ is defined as:

$$R(h) = \mathbb{E}_{(a,b) \sim P}[L(h(a), b)],$$

where the expectation is taken over the unknown distribution P . In other words, the risk measures the average error the function incurs over all possible data points (a, b) .

Statistical learning seeks to find a function $h^* \in \mathcal{H}$ that minimizes the population risk:

$$h^* \in \arg \min_{h \in \mathcal{H}} R(h).$$

5.3 Understanding Risk and Its Challenges

To compute the risk $R(h)$, we need to:

- Know the probability distribution P of (a, b) , which is typically unknown.
- Evaluate the integral corresponding to the expectation over P . However, this integral is generally intractable in practice.

Thus, while minimizing the population risk is a theoretical goal, it is often impractical due to the lack of knowledge about P . Instead, we approximate the risk using data-driven methods.

Empirical Risk Minimization (ERM)

To address the challenges above, we use the principle of **empirical risk minimization (ERM)**. Instead of the population risk, we compute the **empirical risk**, which is the average loss over a given dataset $\{(a_i, b_i)\}_{i=1}^n$:

$$R_n(h) = \frac{1}{n} \sum_{i=1}^n L(h(a_i), b_i).$$

Why ERM Works:

- By the **law of large numbers**, the empirical risk $R_n(h)$ converges to the population risk $R(h)$ as $n \rightarrow \infty$. This means that, given sufficient data, the average loss is a good approximation of the true risk.
- Additionally, advanced statistical tools like **concentration inequalities** (e.g., Chernoff bounds) provide guarantees on how close $R_n(h)$ is to $R(h)$ with high probability, even for finite n .

5.4 The Goal of Statistical Learning

The goal of statistical learning is to find a function $h^* \in \mathcal{H}$ that minimizes the empirical risk:

$$h_n^* \in \arg \min_{h \in \mathcal{H}} R_n(h).$$

Practical Considerations:

- In practice, we do not know the true function class \mathcal{H} used by the supervisor. Thus, the choice of \mathcal{H} is often an approximation based on domain knowledge or assumptions.
- The quality of the learned function h_n^* depends on the representational power of \mathcal{H} and the size of the dataset n .

5.5 Illustrative Example

Consider the problem of predicting whether DNA data indicates the presence of cancer. The data consists of input-output pairs (a_i, b_i) , where a_i represents a DNA sample, and b_i is a binary label indicating whether the sample corresponds to a cancer diagnosis. The goal is to learn a function $h(a)$ that maps DNA data to the correct diagnosis with minimal error.

In this setting:

- The loss function $L(h(a), b)$ measures the error in predicting b from a using h .
- The empirical risk $R_n(h)$ quantifies the average error on a finite dataset of DNA samples.
- Statistical learning aims to minimize $R_n(h)$ over a function class \mathcal{H} to find the best predictor h_n^* .

5.6 The Faces of ERM and Challenges of Function Classes

There are multiple ways to explain the same concept, just as there are many faces to a polytope. When it comes to empirical risk minimization, averages provide a good estimate of the expected loss, giving us valuable information. However, a key challenge remains: we do not know the true function class \mathcal{H} .

To address this, we choose a function class \mathcal{H} , but the problem becomes complex because there are infinitely many possible functions to consider. Mapping inputs to outputs without constraints leads to an overwhelming search space. To make the problem solvable, we introduce assumptions about the nature of the function class, such as whether it is parametric or non-parametric.

Non-Parametric Models: Non-parametric models are not “parameter-free.” Instead, the number of parameters grows with the size of the data. A popular example of non-parametric models is kernels, which use a data-driven approach to define mappings. For those interested, we provide a supplementary lecture on kernels via Moodle. While kernels were once widely popular, they have declined in use due to the rise of parametric alternatives that offer fixed parameter sizes.

Parametric Models: In parametric models, the number of parameters is fixed, regardless of the dataset size. A simple example is a polynomial function, where the coefficients act as parameters. For instance, a polynomial model of the form:

$$h(a) = x_1 + a_1x_2 + a_2x_3^2$$

has its mapping defined by the polynomial coefficients a_1, a_2 , etc. Parametric models are widely used in statistical learning because they simplify the search space and enable effective parameterized mappings.

5.7 Empirical Risk Minimization in Practice

Statistical learning uses ERM to estimate functions when the underlying distribution is unknown. Instead of computing expectations, which require knowledge of the true distribution P , ERM computes the average loss over observed data. For example:

- Parametric models, such as polynomials or neural networks, parameterize the function class \mathcal{H} and allow us to learn mappings effectively.
- By constraining the parameters (e.g., bounding polynomial coefficients), we reduce the complexity of the optimization problem.

5.8 Final Thoughts on ERM

ERM is an essential tool in statistical learning, allowing us to approximate the population risk through data-driven averages. While it is a powerful framework, its success relies heavily on the assumptions about the function class \mathcal{H} and the quality of observed data. With parametric and non-parametric models offering distinct advantages, the choice of function class plays a critical role in ensuring generalization and performance.

5.9 Insights into Parametric Estimation Models

Parametric estimation models are introduced as a practical approach to statistical learning. Starting with simple cases, such as low-order polynomials, these models build understanding while maintaining tractability. For instance:

- When the true conditional distribution $P(b|a)$ is unknown, parametric models can approximate the mapping using observed data.
- Neural networks, which are advanced parametric models, offer significant flexibility and power by parameterizing complex function classes.

6 Gaussian Linear Model: Overview and Applications

The Gaussian linear model represents one of the simplest yet widely applicable frameworks in statistical estimation. It incorporates the following key elements:

- 1. Parameter Space:** This defines the set of unknown parameters to estimate, such as the coefficients in a linear function or polynomial. The true parameter, denoted by x^\dagger (or $x^\#$), is the ground truth we aim to approximate.
- 2. Data and Conditional Distributions:** The observed data, b_i , are modeled as samples generated through a linear transformation of the parameter x^\dagger , often with additive Gaussian noise. This results in a conditional distribution $P(b_i | a_i, x^\dagger)$, where a_i represents input features or sampling information.

Example: MRI Imaging

In the context of MRI:

- x^\dagger represents the true image (e.g., a heart scan) that we want to reconstruct.
- The machine acquires b_i , Fourier coefficients of x^\dagger , through linear transformations.
- Additive Gaussian noise is often included in this model due to its ubiquity in real-world measurements.
- The goal is to estimate x^\dagger by minimizing the error between the observed b_i and the predicted values.

6.1 Robustness and Noise Assumptions

While Gaussian noise is a common assumption due to properties like asymptotic normality, real-world scenarios may involve other types of noise, such as Poisson or multiplicative noise. Despite these mismatches, simple models like the Gaussian linear model are robust and can provide reasonable approximations under mild perturbations in assumptions.

6.2 Estimation with Least Squares

A popular estimator in the Gaussian linear model is the **least squares estimator**, which minimizes the ℓ_2 norm of the error:

$$\hat{x} = \arg \min_x \frac{1}{n} \|b - Ax\|_2^2,$$

where A is the design matrix formed by concatenating the input data a_i , and b is the vector of observations b_i . This approach uses a loss function based on the ℓ_2 error and is computationally efficient for solving linear inverse problems.

6.3 Key Observations:

- **Random Nature of the Data:** The observed data b_i is random due to noise, making the estimator's output a random variable. The goal is to design an estimator that closely approximates x^{\dagger} while minimizing data and computational requirements.
- **Applications and Generality:** The Gaussian linear model has widespread applications, including image reconstruction, signal processing, and general parameter estimation problems. Its simplicity makes it a foundational tool in statistical learning and inverse problems.

6.4 Practical Insights

- Increasing model complexity (e.g., quadratic fits or higher-order polynomials) can initially degrade performance, but phenomena like *double descent* may lead to performance improvements at extreme levels of complexity.
- Despite its simplicity, the Gaussian linear model remains robust and versatile, offering a strong starting point for understanding statistical estimation.

6.5 Performance of the Estimator

- The performance of the least squares estimator is quantified by its expected error:

$$\mathbb{E}[\|\hat{x} - x^{\dagger}\|_2^2].$$

- Theorem: If $A \in \mathbb{R}^{n \times p}$ has independent and identically distributed Gaussian entries, then as $n, p \rightarrow \infty$, the following holds:

$$\frac{\mathbb{E}[\|\hat{x} - x^{\dagger}\|_2^2]}{n - p} \rightarrow \sigma^2.$$

This result demonstrates the asymptotic properties of the least squares estimator under Gaussian assumptions.

6.6 Robustness and Practical Considerations

While Gaussian noise is a common assumption, real-world scenarios may involve other types of noise, such as Poisson or multiplicative noise. Despite these deviations, the Gaussian linear model often remains robust and provides reasonable approximations.

Alternative estimators, such as the **least-absolute deviation (LAD)** estimator, minimize the ℓ_1 -norm of the residuals:

$$\hat{x}_{\text{LAD}} = \arg \min_x \frac{1}{n} \sum_{i=1}^n |b_i - a_i^{\top} x|,$$

which is more robust to outliers compared to the least squares estimator.

6.7 Verification of Fidelity

The fidelity of the estimated parameter \hat{x} to the true parameter x^{\dagger} can be evaluated using various metrics. A standard approach is to use the ℓ_2 -error:

$$d(\hat{x}, x^{\dagger}) = \|\hat{x} - x^{\dagger}\|_2^2.$$

Fidelity can be verified under the following conditions:

1. $\mathbb{E}[d(\hat{x}, x^{\dagger})] < \epsilon$ (expected error).
2. $\mathbb{P}[d(\hat{x}, x^{\dagger}) > \epsilon] \rightarrow 0$ (consistency).
3. $\sqrt{n}(\hat{x} - x^{\dagger}) \rightarrow \mathcal{N}(0, \Sigma)$ (asymptotic normality).

6.8 Applications and General Insights

The Gaussian linear model and its estimators form the basis for numerous applications:

- **Image Reconstruction:** Estimating high-resolution MRI images from noisy Fourier samples.
- **Signal Processing:** Reconstructing signals from compressed measurements.
- **General Parameter Estimation:** Approximating unknown parameters in systems governed by linear mappings.

6.8.1 Example: MLE for Quantum Tomography

Quantum tomography is a practical application of MLE for reconstructing the state of a quantum system. A quantum state is represented by a density operator X , which is a Hermitian, positive semidefinite matrix. For a system of q -qubits, $X \in \mathbb{C}^{p \times p}$, where $p = 2^q$.

Problem Setup: Given a set of independent random variables b_k , the probability distribution is expressed as:

$$P(b_k = k) = \text{Tr}(A_k X),$$

where A_1, A_2, \dots, A_m are positive operator-valued measures (POVM), a set of Hermitian, positive semidefinite matrices summing to the identity matrix I . The goal is to estimate X using the measurements A_k and observations b_k .

MLE Formulation: The MLE for X is formulated as:

$$X_{\text{MLE}}^* = \arg \min_{X \geq 0} - \sum_{k=1}^m b_k \log(\text{Tr}(A_k X)).$$

This formulation minimizes the negative log-likelihood while ensuring that X remains positive semidefinite.

Performance of the MLE: The performance of the MLE is evaluated using numerical simulations, particularly for systems of three qubits. The error metric involves the Frobenius norm of the difference between the estimated X_{MLE}^* and the true X^\dagger . Observations include:

- The error decreases proportionally to $\frac{1}{n}$, where n is the number of measurements.
- Simulations confirm the theoretical bound, demonstrating consistency and accuracy of the MLE as the sample size increases.

Limitations of the MLE While the MLE is a powerful estimation tool, it does not always yield optimal performance under all circumstances. Some notable limitations include:

Comparison with James-Stein Estimator: For dimensions $p \geq 3$, the James-Stein estimator dominates the MLE in expectation. This estimator shrinks the estimates towards the origin, improving robustness against noise. It is expressed as:

$$x_{\text{JS}}^* = \left(1 - \frac{p-2}{\|b\|_2^2}\right) b,$$

where b represents the observed data.

Dimension is your enemy, and data is your friend: For quantum tomography, this relationship remains consistent. The behavior of the estimator aligns with the theory:

$$\|\hat{x}_{\text{MLE}} - x^\dagger\| \propto \sqrt{\frac{p}{n}}.$$

This behavior has been verified using both theoretical analysis and real-world data, where quantum tomography state estimates and their errors exhibit the expected trends. The results are robust and align well with theoretical predictions.

Underdetermined Systems ($n < p$): In scenarios where the number of samples n is less than the dimensionality p , the least-squares solution may become arbitrarily large. For example, in linear models:

$$x_{\text{LS}}^* = \arg \min_x \|Ax - b\|_2^2,$$

the estimation error scales poorly with n , leading to overfitting.

Proposition on Overfitting: For A with standard Gaussian entries:

$$\|x_{\text{LS}}^* - x^\dagger\|_2^2 \propto \frac{p}{n},$$

showing that the error grows with dimensionality p and decreases with the number of samples n .

Counterintuitive Insights About Estimators It is important to note that the maximum likelihood estimator (MLE), while widely used, is not always the best choice. Alternatives like the James-Stein estimator often perform better, particularly when robustness to random noise is critical.

Challenges with Least Squares Estimation In least squares estimation, underdetermined problems ($p > n$) introduce additional complexities:

- **Infinite Solutions:** There are infinitely many solutions due to the underdetermined nature of the problem.
- **Error Scaling:** The error scales as:

$$\text{Error} \propto \frac{\text{Data (friend)}}{\text{Dimension (enemy)}} \times \left(1 - \frac{n}{p}\right).$$

These challenges highlight the importance of balancing data quantity and dimensionality to ensure meaningful and reliable estimates.

6.9 Final Remarks

While maximum likelihood estimation and least squares approaches are powerful tools, they are not always optimal or without limitations. Alternative estimators like the James-Stein provide better performance under specific conditions, and understanding the trade-offs between dimensionality and data availability is key in statistical estimation.

In conclusion, the fundamental takeaway is clear:

Data helps reduce error, but higher dimensionality amplifies it.

Despite its simplicity, the Gaussian linear model is a robust and versatile framework, with extensions to more complex models covered in subsequent lectures.