Matrix Calculus for Machine Learning

Alan Edelman
Steven G. Johnson

July 4, 2025

Contents

Chapter 1

Lecture 1

1.1 Introduction to the Course on Matrix Calculus

1.1.1 Course Overview

Welcome to this IAP class on matrix calculus. I am Professor Alan Edelman, and the other professor is Stephen Johnson. He will be delivering some of the lectures remotely due to out-of-town commitments, utilizing Zoom technology.

Both of us are in the Math Department. I am also a part of CSAIL, where I run the Julia Lab. My research focuses on both mathematics and computing software. Professor Johnson has a background in matrix calculus through his work, primarily with PDE-constrained optimization, and has also contributed to Julia programming.

1.1.2 Course Structure and Requirements

In this course, we have structured 16 hours of lectures. If any of you have taken 18.06, you may find one of the hours familiar. We will be making use of a GitHub repository that contains lectures and problem sets. The course assumes familiarity with basic linear algebra but does not require prior knowledge of Julia. However, for those not familiar yet, the Julia used in this class will be introductory and used mainly as a calculator.

1.2 Matrix Calculus in the Academic Sequence

At MIT, and in universities worldwide, calculus education begins with single-variable calculus (18.01), covering the derivatives and integrals of single-variable functions. This is followed by multivariable calculus (18.02), which introduces concepts of gradients and Jacobians.

It seems that the sequence of calculus education arbitrarily stops at vectors (1-dimensional). Logically, it should extend to matrices (2-dimensional) and beyond to higher-dimensional arrays. Programming languages often handle structures beyond vectors, and thus the ability to apply calculus to matrices and higher-dimensional arrays becomes crucial.

As someone who has taught at MIT for nearly 30 years, I have seen linear algebra's role transform significantly. Once a topic to be avoided, it has now become essential due to the rise of machine learning, statistics, and engineering applications. These fields require an understanding of linear algebra and the ability to perform calculus on matrices and higher-dimensional structures.

1.2.1 Relevance to Machine Learning

In machine learning, the necessity of taking gradients, especially of complex objects, is pivotal. Techniques like gradient descent require this capability. Traditionally, I have integrated matrix calculus concepts, marrying calculus with linear algebra, to facilitate applications in modern computational fields.

- **Single-variable Calculus (18.01):** Basics of derivatives and integrals.
- **Multivariable Calculus (18.02):** Gradients, Jacobians in vector spaces.
- **Matrix Calculus:** Extends concepts to 2D matrices and higher dimensions.

The incorporation of linear algebra into mainstream computational tools has enhanced the relevance and importance of understanding matrix calculus, especially in the context of the diverse applications present in machine learning and data science today.

purpose related to matrix calculus, whether it's for machine learning, physics, engineering, or any other field where advanced calculus concepts are crucial.

1.3 Introduction to Matrix Calculus

1.3.1 Overview and Objectives

Welcome to this IAP class on matrix calculus. I'm Professor Alan Edelman, and my colleague, Professor Stephen Johnson, will be joining remotely for the initial sessions. Our plan is to cover matrix calculus comprehensively over 16 hours of lectures, utilizing both our expertise in mathematics and programming, notably with the Julia programming language, though prior experience with Julia is not required.

This course builds on foundational knowledge in linear algebra. Its objective is to extend calculus beyond single-variable and multivariable systems (such as scalar and vector calculus) to matrix calculus, which is increasingly important given the rise of machine learning and data science.

1.3.2 Background and Context

Matrix calculus can be viewed as a natural extension of the calculus sequence offered at most universities, which starts with single-variable calculus (18.01 at MIT), progresses to multivariable calculus (18.02), and often omits further discussions into higher-dimensional structures like matrices. The structure of this course is designed to bridge the gap from standard calculus to matrix calculus, thereby addressing a crucial need in many scientific disciplines.

1.3.3 Why Matrix Calculus?

Over the past few decades, linear algebra has gained prominence due to its applicability in modern technological fields such as machine learning and statistical analysis. Understanding advanced calculus on matrices and higher-dimensional arrays is essential for complex applications, including but not limited to gradient descent and optimization in high-dimensional spaces.

The concepts of gradient and Jacobian, typically covered in vector calculus (18.02), are fundamental in understanding the calculus of matrices and higher structures. Many programming languages, including Julia, readily support operations on multi-dimensional arrays, further necessitating the understanding of matrix calculus.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_V CN4izB$

1.4 Key Questions in Matrix Calculus

1.4.1 Example Problems

Consider the function that squares a matrix. An immediate question is: What should the derivative be? Unlike the scalar case where the derivative is simply $2x$, the derivative for matrices is not the same when x is not a scalar.

Other questions include the derivative of the matrix inverse, the matrix inverse squared, and related operations. These operations reveal that matrix calculus involves complexities that are not immediately predictable from scalar or vector calculus.

1.4.2 Complexities and Challenges

Matrix calculus is inherently more complicated than its scalar or vector counterparts. While it is a subject that can be mastered, it requires understanding nuances that aren't trivial extensions of the simpler cases.

1.4.3 Applications

Applications of matrix calculus are varied and impactful. From optimization problems constrained by partial differential equations to foundational techniques in machine learning, its utility spans numerous scientific and engineering disciplines.

As the course progresses, we will explore these mathematical concepts and their real-world applications, equipping you with the skills needed to solve complex problems using matrix calculus.

1.5 Introduction to Matrix Calculus Applications and Automatic Differentiation

In contemporary mathematics and computing, buzzwords like machine learning, parameter optimization, stochastic gradient descent, automatic differentiation, and backpropagation are ubiquitous. These signify the rapid developments that necessitate an understanding of matrix calculus. The applications of matrix calculus extend beyond theoretical pursuits to address physical problems and machine learning statistics.

For instance, consider topology optimization, a procedure used in the design of aircraft wings. Historically, engineers would select a wing design and iteratively refine it by assessing aerodynamic performance. Currently, with advancements in computing and machine learning, it's possible to computationally optimize for various objectives, such as minimizing fuel consumption or material use. The computer determines the optimal shape, transforming it into a significant optimization challenge. This method is also applicable to fluid dynamics and other domains, facilitating simulations that embed physical models within optimization loops—a paradigm shift from past methodologies.

Data science and multivariate statistics represent additional areas where these principles are being actively applied. Here, computer scientists leverage these methods extensively to glean insights from complex data sets.

1.5.1 Automatic Differentiation: Overview and Implications

Automatic differentiation, or autodiff, represents a noteworthy technological advancement. Understanding autodiff requires discerning what it is—and, crucially, what it is not. While MIT attendees are proficient in traditional differentiation methods, automatic differentiation diverges from these conventional techniques. It is not numerical differentiation, as seen in numerical analysis, where $\Delta y/\Delta x$ quantifies small changes. Nor is it akin to symbolic differentiation, available via tools like Wolfram Alpha or Mathematica.

Instead, autodiff functions more like a compiler technology, applying differentiation rules algorithmically. This distinguishes it from classical approaches and renders it an intriguing subject of study, often misleading new learners with its computational intricacies.

Analogous to long division's educational shift—where understanding the concept is prioritized over manual execution—the pedagogy of calculus may see a transition. Traditional methods establish foundational understanding, yet the necessity for manual differentiation is diminishing as computational tools evolve.

This course aims to demonstrate these methodologies, linking classical calculus through modern computational lenses. By doing so, it paves the way for a nuanced comprehension of matrix calculus's role in solving contemporary computational problems.

1.6 Introduction to Modern Optimization and Differentiation Techniques

In contemporary times, the buzzwords such as machine learning, parameter optimization, stochastic gradient descent, automatic differentiation, and backpropagation, highlight the integration of computational methods with matrix calculus. These are crucial in solving complex optimization problems across various applications.

1.6.1 Applications in Engineering and Data Science

Matrix calculus naturally extends to fields like physical problem-solving and machine learning statistics. One intriguing application is the topology optimization of aircraft wings. Traditionally, designing an aircraft wing involved selecting a design and iteratively testing its aerodynamics. Today, with advanced computing and machine learning, optimization can focus on objectives like minimizing fuel or material usage. The computer determines optimal shapes, converting this task into a significant optimization problem, applicable not only to aircraft wings but also to fluid dynamics.

Data science and multivariate statistics form another vast area, with computer scientists leveraging these techniques extensively.

1.6.2 The Role of Automatic Differentiation

Automatic differentiation, or autodiff, represents a fascinating advancement in computational technology. Unlike traditional numerical differentiation, which involves approximating derivatives by computing differences, or symbolic differentiation as performed by tools like Wolfram Alpha or Mathematica, autodiff offers a new approach. It's similar to a compiler technology rather than a mathematical coursework methodology.

- **Numerical Differentiation:** Involves the calculation of derivatives through finite differences, $\Delta y/\Delta x$, providing approximations.
- **Symbolic Differentiation:** Utilizes symbolic manipulation software to derive expressions, unlike autodiff.
- **Automatic Differentiation:** Focuses on evaluating derivatives accurately and efficiently by decomposing functions into elementary operations and applying the chain rule systematically.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_{V}CN4izBK$

Many at MIT are proficient in manual differentiation; however, the intrinsic complexity of modern derivatives suggests that automatic differentiation tools are essential. Indeed, complex derivatives exceed human computational capabilities, making autodiff an invaluable tool for dealing with such intricacies.

1.6.3 Educational Implications

The evolution of computational tools also impacts educational methods. Conventional calculus courses, such as MIT's 18.01, focus predominantly on symbolic methods (approximately 90%). While foundational understanding is crucial, the practical application increasingly leans on computational tools like autodiff.

For effective usage of autodiff, understanding the underlying technology is important, akin to knowing how an engine works to drive a car efficiently. Knowledge of when to employ forward or reverse mode differentiation and understanding the significance of vector-Jacobian products are essential for adept usage.

1.6.4 Linearization and Derivatives

An essential concept in calculus is linearization — approximating nonlinear functions locally as linear. This viewpoint, while trivial in one dimension, facilitates understanding multivariate calculus where complex surfaces are approximated by tangent planes (or hyperplanes in higher dimensions).

For a point (x_0, y_0) on a curve, the linear approximation or tangent line can be expressed as:

$$y - y_0 = f'(x_0)(x - x_0),$$

where $(\Delta y/\Delta x) = f'(x_0)$. This linear perspective is foundational as we extend understanding to higher dimensions, emphasizing that calculus abstracts complex shapes as locally flat entities.

1.7 Notations and Linearization

In this discussion, we introduce some important notations. We denote finite perturbations using the symbol Δ. When considering derivatives, we often encounter expressions like $f'(x)\Delta x$, or, taking the infinitesimal limit, the familiar notation $\frac{dy}{dx} = f'(x)$.

The traditional calculus notation $\frac{dy}{dx}$ is often taught with the caveat that it's not actual division. Nevertheless, it is a useful conceptual tool, often depicted as such for ease of understanding and application in linearization.

1.7.1 Linearization Concept

Linearization involves approximating a nonlinear function with a linear function at a point. This can be illustrated with the expression:

$$df = f'(x)\,dx$$

where df is the infinitesimal change in f, $f'(x)$ is the derivative of f at x, and dx is the infinitesimal change in x.

In one-dimensional calculus, $f'(x)$ simplifies to a constant, representing a linear relationship between the change in x and the change in f. This linearization is fundamental and often serves as an introduction to the concept of derivatives.

> We need to be cautious about dividing by dx, especially when transitioning to vectors and higher dimensions. In such contexts, division is not straightforward, much like the non-meaningfulness of dividing vectors by vectors.

1.7.2 Example: The Square Function

Let's apply these concepts to a simple example: the function $f(x) = x^2$. Consider the specific point $(3, 9)$:

- The derivative, $f'(x)$, is given by:

$$f'(x) = 2x$$

- At $x = 3$, the derivative is:

$$f'(3) = 2 \times 3 = 6$$

Hence, when x changes by an infinitesimal amount, the linear approximation of the change in the function value at $x = 3$ is described by the equation of the tangent line. If you calculate this in practice, either by hand or on a computer, it becomes evident how well this linear approximation works as x approaches 3.

1.8 Finite Perturbations and Linearization

In this section, we delve into various notations used for finite perturbations and the concept of linearization in calculus. We introduce the notion of finite perturbations using the delta (δ) notation and link it with familiar concepts such as differentiation.

1.8.1 Notation and Finite Perturbations

The traditional notation $\frac{dy}{dx} = f'(x)$ is widely recognized, though occasionally misunderstood. When considering finite changes, we use $f'(x)\delta x$ as a representation of changes in the function. This notation emphasizes that $f'(x)$ is a linear functional, which in a one-dimensional context acts as a constant multiplier, linking changes in x to changes in the function f.

For a function $y = f(x)$, the differential df is represented as $df = f'(x)dx$. This signifies the change in the function with respect to changes in x, which is foundational to calculus.

1.8.2 Differentiating the Square Function

As an example, consider the square function, particularly at the point $(3, 9)$. The derivative of the square function is $f'(x) = 2x$. At $x = 3$, the derivative is 6, meaning a small change δx at this point results in a change $6\delta x$.

$$\delta y = f'(3)\delta x = 6\delta x$$

If δx is a small increment from 3, the corresponding function value will be close to 9, with the increment defined by the linear relation above.

1.8.3 Infinitesimals and Practical Computation

In practical computations, dx and dy are often thought of as very small numbers. While the concept of infinitesimals can be rigorously defined mathematically, for computational purposes, they are treated as the limits of very small numbers—this is an immensely useful approach in evaluations and simulations.

1.9 Introduction to Matrix Calculus

With the foundation on scalar calculus laid, we now transition into the realm of matrix calculus. This involves understanding notation such as the element-wise product in vector and matrix contexts, and the notion of a trace of a matrix.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_VCN4izBK$

1.9.1 Element-wise Products and Trace

The element-wise product, referred to in some tools as "broadcasting," is denoted by a dot, ·, in some languages. This operation is essential for manipulating matrices and vectors in a point-wise fashion.

The trace of a square matrix is the sum of its diagonal entries. This scalar quantity might be familiar from linear algebra and plays a role in certain matrix calculations.

1.9.2 Matrix Calculus Queries

An example from the practical point of view is the derivative of a scalar trace function. Given matrices Y, X, and Θ, the problem might involve computing the gradient of an expression involving these matrices:

$$\text{The trace of } (Y - X\Theta)(Y - X\Theta)^\top$$

The derivative is denoted by:

$$-2X^\top(Y - X\Theta)$$

Tools like matrixcalculus.org provide computational help in such analysis, allowing experiments with various matrix operations.

In upcoming discussions, we will consider gradients of expressions like $x^\top x$ or $x^\top Ax$, demonstrating approaches that are more comprehensive than traditional component-wise analysis. Here's an invitation to view matrices holistically as we explore further topics in matrix calculus.

1.10 Matrix Calculus

1.10.1 Introduction to Derivative Notations

We begin by discussing some notations used in calculus. The notation δ is utilized to indicate finite perturbations, represented as $f'(x)\delta x$. In cases where we consider infinitesimal limits, many of you are familiar with the notation for the derivative $dy/dx = f'(x)$. However, it's often debated whether this represents true division. Despite this, it is useful to see $df = f'(x)dx$, noting that f' is a linear function, and in one variable, it acts as a constant multiplied by the change in x to equal the change in f.

1.10.2 Linearization and Practical Computations

Consider the square function at the point $(3, 9)$, with derivative $2x$. If $x = 3$, then $2x = 6$. Adding a small perturbation δx leads to a change $f(x + \delta x) \approx 9 + 6\delta x +$ (higher order term). Here, $\delta y = 6\delta x$, highlighting the linear relationship as a multiplier of 6. In practical applications, dx and dy can be thought of as very small numbers, rather than infinitesimals, useful in computational settings.

1.10.3 Transition to Matrix Calculus

Moving from scalar to matrix calculus, we introduce some notations. Element-wise multiplication is represented in Julia's notation as 2 $.*3 = 6, oftenreferredtoasbroadcasting.Also, thetraceofamatrixis$

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_{V}CN4izBK$

1.10.4 A Motivational Problem

Consider a practical matrix calculus problem:

- Let Y be an $n \times m$ matrix.
- X is an $n \times k$ matrix.
- Θ is a $k \times m$ matrix.

The derivative of the scalar trace$(Y - X\Theta)$ with respect to Y can be computed, exemplifying matrix differentiation. The result is a matrix, $-2X^\top(Y - X\Theta)$.

1.10.5 Holistic Approach to Matrix Computations

When dealing with expressions like $x^\top x$ or $x^\top Ax$, it is common to compute gradients in a traditional element-wise manner. However, this course invites you to adopt a holistic perspective, considering vectors and matrices as whole entities rather than a collection of elements.

1.11 Matrix Calculus and Notation in Derivatives

Consider a matrix as more than just an $n\times n$ table, akin to moving beyond understanding a person merely by their physical components like hands, feet, nose, and mouth. A matrix should be seen as a holistic object. In this light, matrix calculus can be performed without necessarily breaking things down to an element-by-element level.

1.11.1 Rethinking Matrix Calculus

Traditionally, in many linear algebra classes, matrix problems are often written out in scalar terms with indices, covering blackboards with lengthy calculations. While this method can be reassuring by providing a tangible step-by-step solution, it is often more elegant and efficient to treat matrices as whole entities using matrix notation.

1.11.2 Understanding Derivatives: Different Scenarios

1. **Scalar to Scalar Case (Standard Calculus)**: This is the basic single-variable calculus (commonly taught in courses like 18.01), where the derivative represents the rate of change of a scalar function with respect to its scalar input.

2. **Scalar to Vector Case**: In simple physics, one might encounter the trajectory of a point in three-dimensional space as a function of time. Here, time is the scalar input, and position (a vector) is the output. The derivative with respect to time yields the velocity vector, which is tangent to the trajectory and indicates the speed and direction of movement.

3. **Scalar to Matrix Case**: It is also feasible to define a trajectory in matrix space, where each element of an $m\times n$ matrix is a function of time. The derivative of such a matrix with respect to time would be another matrix, offering a visualization of change in high-dimensional matrix space.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_{V}CN4izBK$

1.11.3 Exploring Multivariate Derivatives

1. **Vector to Scalar Case (Gradients)**: In courses like multivariate calculus (18.02) and areas such as machine learning, one often encounters functions with many variables as input (vector) and a single output (scalar), often referred to as the "loss function." The derivative here is the gradient, frequently represented with the nabla symbol (∇) and is a vector.

The gradient of a function F where the input is a vector x, maintains the same shape as the input. Thus, if the input is a vector, the gradient is also a vector. In notation:

$$\text{If x is a vector, } \nabla F = \frac{\partial F}{\partial \mathbf{x}}$$

2. **Row Vector vs. Column Vector Debate**: There has been discussion about whether to treat the gradient as a row or column vector. A consensus that has emerged from such discussions (and practical experience in programming environments like Julia) is that while input vectors might conventionally be column vectors, the derivative (F prime) is treated as a row vector.

For a quadratic form example $F(\mathbf{x}) = \mathbf{x}^T\mathbf{x}$, the calculations are:

$$\nabla F = 2\mathbf{x}$$

The differential is expressed as:

$$dF = 2\mathbf{x}^T d\mathbf{x} \quad \text{or} \quad F'(\mathbf{x}) = 2\mathbf{x}^T$$

This treatment ensures that the derivative aligns as a linear operator, where multiplying the transpose vector (a row vector) by a small change vector ($d\mathbf{x}$) outputs a scalar.

This approach to derivatives and matrix calculus emphasizes the utility of maintaining a holistic consideration of matrices and vectors, rather than breaking them into components without necessity.

1.12 Holistic Understanding of Matrices and Derivatives

In an early linear algebra class, one might see a matrix as an $n \times n$ table. However, similar to how humans are more than just the sum of their body parts, matrices should be viewed as holistic objects. There are methods in matrix calculus that avoid descending to the level of individual elements. Consider this: I walked into classrooms where professors, using chalk and blackboards, wrote out entire matrices element-wise, filling the entire board. This practice often omits using matrix notation, opting instead for scalar notation with abundant indices. While indices can sometimes offer clarity, there is elegance in transcending that approach.

1.12.1 Types of Derivatives

Let's explore the different cases for taking derivatives.

Scalar Input and Output

Recall your first semester calculus, 18.01, where a scalar input produces a scalar output. This represents the most basic form of a derivative.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_VCN4izBK$

Vector Outputs in Physics

In simple physics contexts, for instance, the position in three dimensions (a vector) is a function of time (a scalar). Here, the derivative is a velocity vector, tangent to the trajectory's curve, with magnitude indicating speed.

Matrix as Function of Time

Though perhaps less common in earlier courses, we can consider a trajectory in matrix space. An $m \times n$ matrix function of time has elements that change over time. Its derivative, in this context, would be a tangent in this high-dimensional matrix space.

1.13 Gradients and Machine Learning Applications

In multivariate calculus (18.02) and in machine learning, gradients are key. Consider functions with many variables (a vector) resulting in a scalar, often called the loss function in machine learning.

1.13.1 Gradient Notation

For a scalar function F with vector input x, the gradient has the same shape as the input. This is denoted using the nabla symbol (∇) in LaTeX as \nabla, commonly written as ∇F or the gradient of F.

1.13.2 Transforming Dimensions

Within this framework, it simplifies to denote gradients as vectors (column vectors) and derivatives or Jacobians as row vectors. For example, for the function $F(\mathrm{x}) = \mathrm{x}^T\mathrm{x}$: - The gradient $\nabla F = 2\mathrm{x}$. - The total differential $dF = 2\mathrm{x}^T d\mathrm{x}$. - The derivative $F'(\mathrm{x}) = 2\mathrm{x}^T$.

This maintains consistency, as multiplying a row vector by a column vector results in a scalar, representing a small change dx in the vector x impacting the scalar function F.

1.14 Dimensionality in Derivatives

There's a noticeable color coding for derivative dimensionality: - Green indicates zero-dimensional (scalars on the diagonal). - Blue indicates one-dimensional (vector and gradient). - Matrix derivatives, highlighted diagonally from northeast to southwest, such as in machine learning when parameters form an array.

Finally, the Jacobian matrix, a crucial concept often underrepresented, connects many of these aspects. Some may recall its role in linearization and integrals from 18.02. The Jacobian matches the shape of the inputs, thus reflecting an essential tool in the broader course of dimensional analysis and matrix derivatives.

1.15 Introduction to Jacobian Matrices and Higher-Order Derivatives

In this segment, we will delve into the concept of Jacobian matrices and their applications particularly focusing on the transition from vector inputs to vector outputs. It's

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_VCN4izBK$

worth noting that students typically learn about Jacobian matrices through the lens of multi-dimensional integrations, often only in the context of determining determinants of Jacobians, which can sometimes lead to misconceptions.

1.15.1 Jacobian Matrices: Vector Inputs and Outputs

Consider a function with a vector input and a vector output. For instance, imagine a 3-dimensional input function producing an output in either a lower or higher number of dimensions. If we have n dimensions going into the function and m dimensions coming out, the derivative of the function is best expressed as an $m \times n$ matrix. Such matrix representations become particularly important as we delve into higher order arrays, where derivatives extend beyond the constraints of two-dimensional matrices.

1.15.2 Operator Notation and Small Changes

Let us revisit the basic derivative calculation using operator notation. Consider the derivative of x^3, which is $3x^2\,dx$. This is widely recognized, but it serves as a reminder that for a small change in x, the corresponding change in the function will be approximately $3x^2\times$ the small change. For example, if $x = 7$ and the change dx is 0.01, the change in the function is approximately $3 \cdot 7^2 \cdot 0.01$.

1.15.3 Dot Products and Matrix Derivatives

Next, consider a vector input and a scalar output, where x is a vector and dx is a small change to that vector. In such cases, the dot product is significant. Recall that the expression $x^T y$ (where y could be another vector) represents a dot product, multiplying all the corresponding elements and summing the results.

For example, with the function $x^T x$ and a small change dx, the change in the function is represented as a dot product between $2x$ and dx. Hence, $\Delta(x^T x) = 2x^T dx$.

1.15.4 Matrix Squaring and Change in Function

As we extend into matrix derivatives, consider squaring a matrix X. The derivative here is not simply $2X$ due to non-commutative properties of matrix multiplication. The change in the square of the matrix due to a small change dX must account for the non-commutativity. Thus, the change is computed as:

$$\Delta(X^T X) = X^T dX + (dX)^T X$$

This formula captures the necessity to consider both multiplication orders, highlighting the complex nature of derivatives in matrices. Understanding these concepts is key to grasping higher-dimensional derivative calculations.

1.16 Advanced Differentiation Techniques with Matrix Functions

In this section, we delve into the understanding of Jacobian matrices, especially when dealing with vector functions that extend across multiple dimensions. A common misconception among students is to equate Jacobians directly with their determinants. However, the Jacobian is much more insightful as it encapsulates derivative information for vector-valued functions.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_VCN4izBK$

1.16.1 Jacobian Calculation in Multi-Dimensional Spaces

When we transition into multi-dimensional calculus, handling vector inputs and outputs requires us to represent derivatives as matrices. Consider a vector function where the input is an n-dimensional vector and the output is an m-dimensional vector. For such functions, the derivative should be expressed as an $M \times N$ matrix, aligning with the dimensions of the output and input respectively.

Matricial differentiation offers a more consolidated framework compared to treating derivatives as linear operators. This becomes crucial when expanding into higher order arrays, where derivatives transcend simple two-dimensional matrices.

1.16.2 Operator Notation and Function Changes

To illustrate the differentiation process, consider the derivative of a simple polynomial: the derivative of x^3 is $3x^2\,dx$. This indicates that a small change in x results in a corresponding change in the function, calculated as $3x^2$ times the small change.

For a vector input and scalar output, the derivative could be expressed using a dot product notation. For example, if x is a vector and dx is its small change, the derivative is given by the dot product $2x \cdot dx$, representing the change in the squared function $x^T x$.

1.16.3 Matrix Differentiation: Non-commutative Properties

In matrix functions, unlike scalar functions, the commutative property does not hold. When squaring a matrix, the differential dx does not commute with x. The proper expression for the change in a matrix square, x^2, given a small perturbation dx, is the linear operator $(x \cdot dx + dx \cdot x)$. This accounts for both multiplication directions due to non-commutativity.

To visualize, consider a matrix x with $n \times n$ dimension. The function $x \to x^2$ involves n^2 variables. The derivative would scale up to involve n^4 terms, but it's more efficient to think in terms of linear operators rather than attempting to manage a massive $n^2 \times n^2$ matrix.

1.16.4 Numerical Example

Consider the quadratic function $f(x) = x^T x$, which sums the squares of its components. At point $x = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$, the function evaluates to 25. With a small change $dx = \begin{pmatrix} 0.001 \\ 0.002 \end{pmatrix}$, applying the calculus formula, the resulting change in f, Δf, can be precisely predicted by $2x^T dx$.

This approach underlines the elegance of calculus in process automation-ensuring predictions align with numerical calculations.

1.16.5 Practical Demo in Julia

To test our understanding with matrices, consider an arbitrary 3×3 matrix x, initialized with some integers. Define the differential dx through a random perturbation. Compute the change dy, both numerically as $(x+dx)^2 - x^2$, and using the matrix derivative $(x \cdot dx + dx \cdot x)$. Comparing these methods validates the consistency of our matrix calculus approach.

While this example was conducted using Julia in a REPL environment, the demonstrated principles are crucial in both practical computational mathematics and theoretical explorations.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_VCN4izBK$

1.17 Numerical Derivation and Matrix Derivative Formulas

In this segment, we explore the numerical approach to calculating derivatives, especially in the context of matrix functions. Let's consider a practical illustration using a numerical method to better understand the computation of derivatives.

After performing a calculation, we compared the present state of the matrix to its initial state. This change aids in understanding the derivative numerically. To demonstrate:

$$\texttt{Numerical Derivative} = \texttt{Final State} - \texttt{Initial State}$$

However, this is not the only method. We also discussed a "magic formula" which provides the derivative of a squared matrix x^2 in terms of its differential dx. Some common misconceptions about this derivative include incorrect assumptions such as:

$$2x \cdot dx, \quad \texttt{or} \quad 2 \cdot dx \cdot x$$

Neither of these expressions correctly represents the derivative. The correct formula that determines how to approach changes in a squared matrix is:

$$\Delta(x^2) = x \cdot dx + dx \cdot x$$

While this has not yet been derived in detail, it serves as an accurate model for computing such derivatives. This visual or numerical evidence creates an intuitive understanding. Personally, viewing this concrete demonstration aids in grasping the validity of the formula.

The derivation of this approach will be explored further shortly, providing a deeper mathematical basis for why these computations are valid.

1.18 Numerical Derivation and Product Rule Extensions

In our previous discussion, we explored how to numerically calculate a derivative by comparing the change in position. We emphasized that certain formulas commonly thought to be correct, such as $2x\,\mathrm{d}x$ or $2\,\mathrm{d}x{\cdot}x$, are incorrect. The correct formula will be derived in our next sessions, but it is important to understand its application experimentally even before we reach the complete theoretical framework.

To illustrate, consider computing a derivative for scalars using calculus on a computer, such as $(x{+}\mathrm{d}x)^2$ for approximating derivatives and arriving at $2x$. Assume $x = 3$ and $\mathrm{d}x = 0.001$. When compared with the well-known formula from ordinary calculus, $2x \times \mathrm{d}x$, they agree to the correct number of significant digits.

This experimental agreement isn't merely coincidental; it suggests that matrix calculus extends certain rules we observed in standard calculus. These include the product rule which many of you learned as $\mathrm{d}(uv) = u\,\mathrm{d}v + v\,\mathrm{d}u$.

1.18.1 Product Rule for Matrices and Vectors

The product rule holds true in the context of matrices and vectors as well, provided the dimensions are compatible. For example, the differentiation:

$$\frac{\mathrm{d}}{\mathrm{d}t}(AB) = \frac{\mathrm{d}A}{\mathrm{d}t}B + A\frac{\mathrm{d}B}{\mathrm{d}t}$$

It is important to note, matrices do not commute; thus, the order must be maintained. If A is on the left in the original formula, it should remain on the left.

Source:
https://www.youtube.com/watch?v=0YqjeqLhDDElist=PLUl4u3cNGP62EaLLH92E$_VCN4izBK$

To see this applied, consider $x^T x$. Differentiating, we get:

$$d(x^T x) = d(x^T)x + x^T dx$$

Yet, these terms simplify to $2x^T dx$. Why can they be combined, violating our order rule? This is permissible because the result is a scalar, which is commutative: $x^T dx = dx^T x$.

For example, consider vectors $x = [1, 2, 3]$ and $dx = 0.001 \cdot \texttt{rand}(3)$. Whether using $x^T dx$ or $dx^T x$, results confirm their equality.

1.18.2 Old and New Derivation Techniques

An old technique to derive ∇f for $f(x) = x^T x = \sum x_i^2$ involves computing partial derivatives:

$$\frac{\partial f}{\partial x_i} = 2x_i \quad \texttt{implying} \quad \nabla f = [2x_1, \dots, 2x_n] = 2x$$

This manual method is valid but simplified by holistically applying the matrix rule, sparing us from handling indices. This illustrates both clarity and efficiency in vector calculus.