*Informal Notes* for Mathematics for Machine Learning
Imperial College London

# Lecture 2: Towards Learning in Linear Models

*Lecturer: Matthew Wicker*

## 1 Learning Objectives

At the end of last lecture, we took a first step towards learning by defining a linear model. In this lecture, we will start by revisiting this model and will revisit vector calculus to observe how to fit this model to data.

## 2 Recall our Linear Models

**Supervised learning** Supervised learning settings assume that we have access input-output pairs, $(\boldsymbol{x}, \boldsymbol{y})$, with $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$. It is typically assumed that we have many such pairs comprising a dataset $\{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^N$.

**Linear Regression Model** Linear regression is exactly what the name suggests: performing regression with a linear model. Assuming domain of $\mathbb{R}^n$ and a one dimensional co-domain we can write our model as $f(x) = \boldsymbol{x}^\top \theta$. Thus our model can be written as:

$$\hat{y}^{(i)} = \boldsymbol{x}^{(i)\top} \theta$$

The key idea behind learning is to find $\theta$ such that $\hat{y}^{(i)} \approx y^{(i)}$. We also note that you may have previously seen linear models in prior education written as affine transformation: $y = mx + b$ where the $b$ is the bias or constant term interpreted often as the 'y intercept' or where the line intersects the y-axis. In fact, this is exactly what makes the model affine and not linear. When we speak of linear models what we mean is that the relationship between the model parameters and model predictions are given by a linear transformation. However, if you look carefully at the definition of a linear transformation, you see that our $y = mx + b$ model from child hood is not, as written, a linear transform.

**Definition 2.1. Linear transformation** A linear transformation between two vector spaces $V$ and $W$ is a map $T : V \to W$ such that the following hold:

- $T(v_1 + v_2) = T(v_1) + T(v_2)$ for any vectors $v_1$ and $v_2$ in $V$, and
- $T(\alpha v) = \alpha T(v)$ for any scalar $\alpha$.

You can see that the latter condition in this definition implies the preservation of the origin, that is, $T(\mathbf{0}) = \mathbf{0}$. But if we look at our $f(x) = mx + b$ model for fixed $m, b \in \mathbb{R}$, we have that $f(0x) = b \neq 0f(x)$. So, why is it that we want a model that is seemingly weaker than the linear models we studied very early in our science curriculum? It turns out, that with a very straight forward modification, we can expand our linear model to capture the affine transformation $y = mx + b$. We call this expansion of our model basis expansion. To recover $y = mx + b$, we simply can add a 1 entry to the end of each feature vector $x \in \mathbb{R}$ and then the corresponding weight for that feature vector becomes the bias. Observe that by introducing a function $\phi(x) : \mathbb{R} \to \mathbb{R}^2$ such that $\phi(x) := [x, 1]^\top$ we then need a parameter vector $\phi \in \mathbb{R}^2$ and so we end up with the model $\hat{y} = \phi(x)^\top \theta = x_1\theta_1 + 1\theta_2$. Thus $\theta_2$ is effectively our 'y intercept' from our previous model. This can be viewed as the most simplistic form of basis expansion.

## 2.1 Basis Expansion

Though our linear model is, as we have formulated it, restricted to lines through the origin, we now present basis expansion as a way of modelling non-linear functions. To begin, lets start with the simplest example, a one dimensional domain and a one dimensional co-domain. The key idea of basis expansion is to (1) expand our one-dimensional feature into many dimensions and (2) use non-linear functions to increase the expressiveness of our previously linear model. We start by considering a basis function $\phi(x) : \mathbb{R} \to \mathbb{R}^3$, specifically we will first consider polynomial basis of the form $\phi(x) = [1, x, x^2]^\top$. We can then assign weights $\theta \in \mathbb{R}^3$ to each of our expanded features. This makes our fully expanded function:

$$\hat{y} = \theta_0\phi(x)_0 + \theta_1\phi(x)_1 + \theta_2\phi(x)_1 \tag{1}$$

$$= \theta_0 + \theta_1 x + \theta_2 x^2 \tag{2}$$

We can see that the output of the function is linearly related to the expanded features, but of course the form of the model that we have expressed is actually a quadratic function. Of course this basis expansion does not just work in one dimension. For example, if our unexpanded feature vector is two dimensional (as opposed to the one dimensional example we just gave), then we can have a basis expansion $\phi : \mathbb{R}^2 \mapsto \mathbb{R}^6$, and following the specific example from before this expansion would return the vector $[1, x_1, x_2, x_1^2, x_2^2, x_1x_2]^\top$. As a minor exercise, you can expand this function as we have done above in the one dimensional case.

**Radial Basis Function Kernel** Importantly, polynomial basis expansion is just a single flavor of basis expansion. Another popular and widely used form of basis expansion is kernel basis expansion. Though we do not fully cover the extent of kernel basis expansion models here, they are good to work with to practice key skills (e.g., the vector calculus we will practice at the end of this lecture). A kernel is a function that takes in two points, $x$ and $x'$, and returns a non-negative distance between those points. The most common kernel is the *radial basis function kernel* (a.k.a. RBF kernel) which takes a fixed parameter $\gamma$ and is defined as $\kappa(x', x) = \exp(-\gamma||x - x'||^2)$. Typically, one picks fixed centers $x$ and then the basis expansion proceeds by computing the expanded feature set with respect to the distance to these centers.

# 3 Linear Algebra Review

In the lecture slides we do a far more detailed linear algebra review. For an expansion of this section, please see those slides.[1] Linear algebra concerns itself with different linear and affine transformations. The objects that describe such transformations are vectors, $\boldsymbol{x}$ which will be described as bold roman symbols with the distinct exception of $\theta$ which denotes the parameters of our machine learning models and may be a vector. A vector with $n$ real-valued entries we say "lives" in $\mathbb{R}^n$. Moreover, vectors are by default column vectors that is they have shape shape $(n \times 1)$. Matrices, can be viewed as 2D arrays:

$$A = \begin{bmatrix} A_{0,0} & A_{1,0} \\ A_{0,1} & A_{1,1} \end{bmatrix}$$

It is important that we quickly review a few facts that you should always keep in mind when doing algebra in this course:

1. Matrix multiplication is **not** commutative: $AB \neq BA$
2. Matrix multiplication is associative: $A(BC) = (AB)C$
3. Matrix multiplication is distributive: $A(B + C) = AB + AC$
4. $(AB)^\top = B^\top A^\top$
5. If $A$ is invertible then $A^{-1}A = I = AA^{-1}$ (it is useful to recall that a matrix is invertible when it is square and has all non-zero eigenvalues).

**Matrix Decomposition** As we have mentioned, matrix multiplication can sometimes be difficult to manipulate algebraically and it is useful to use linear algebra rules beyond the associativity and commutativity of matrices. The most common such rule is eigendecomposition. An eigenvector $\mathbf{v}$ and eigenvalue $\lambda$ of a matrix $A$ are a vector and real value satisfying:

$$A\mathbf{v} = \lambda\mathbf{v}$$

We will assume that you have solid working knowledge of linear algebra, but as a refresher take a moment to write out why it is the case (intuitively or algebraically) that an invertible matrix ($\in \mathbb{R}^{n \times n}$) has $n$ linear dependent eigenvectors. The eigendecomposition of a matrix $A$ is given by collecting its eigenvectors $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(n)}$ into a matrix $V$ and their corresponding eigenvalues into a vector $\lambda = [\lambda^{(1)}, \ldots, \lambda^{(n)}]^\top$. We then write out the eigendecomposition as:

$$A = V\text{diag}(\lambda)V^{-1}$$

Another very useful component of eigendecomposition is the ability to bound certain matrix products:

$$\lambda_{min}(A)||\boldsymbol{x}||_2^2 \leq \boldsymbol{x}^\top A\boldsymbol{x} \leq \lambda_{max}(A)||\boldsymbol{x}||_2^2$$

[1] Future versions of these lecture notes will contain all of the material from the slides.

Hopefully, this section has helped refresh your memory on linear algebra, but as with the rest of these lecture notes it is not an exhaustive list of concepts to remember. We refer those needed to brush up on their linear algebra skills to the linear algebra section of Ian Goodfellow's deep learning book which is linked at the bottom of lecture note 1.

### 3.1 Index Notation

As reasoning about matrix and vector products can sometimes be a bit cumbersome, it is often be useful to write out the operations we perform in index notation. The matrix product $C = AB$ can be written as: $C_{i,j} = \sum_k A_{i,k} B_{k,j}$ it can also be useful to adopt a more "pythonic " index notation where we consider the system $Ax = b$ and write the first entry of the vector as: $A_{1,:}x = b_1$ which indicates that the first value of the result is simply the dot product of the first row of matrix $A$ with the vector $x$. For some more on einstein/index notation that we will not have time to cover in lectures please reference the following video: https://www.youtube.com/watch?v=-hOhhRe2gSA&ab_channel=FacultyofKhan. Some quick reference rules to follow include:

- Free indices appear only once in an expression and thus are not summed over. Dummy indices appear twice, and are implicitly summed over.
- To help avoid confusion, one tip is to use roman letters ($i$, $j$, $k$) for free indices, and greek letters ($\lambda$, $\mu$, $\rho$) for dummy indicies.
- Dummy indices should never appear in the final answer.
- The free indices should always be the same in every term in an expression.
- An index should never appear more than twice in a single term.

An example is given in the lecture notes where we prove the claim about the loss and the $\ell_2$ norm.

## 4 A Minimal Optimization Formulation

Given the above discussion of basis expansion, we can now write down a more general version of linear regression (including an expanded basis) which takes the form:

$$\hat{y}^{(i)} = \phi(\boldsymbol{x}^{(i)})^\top \theta$$

The critical learning question in this model is then:

How do we pick the best $\theta$?

Unfortunately, outside of linear models, this question does not always have a straight-forward answer and depends on several critical modelling decisions that we will unravel throughout this

course. However, for now, we will take the most basic approach which is the standard, frequentist learning approach. This is where we model the "best" $\theta$ as the one that minimizes a loss or error function, $\mathcal{L}$. One such loss is:

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = ||y^{(i)} - \hat{y}^{(i)}||_2^2$$

This is known as the $\ell_2$ or mean squared error loss. Later in the course, we will pick apart exactly when and why someone might make this modelling choice. But, for now, as in previous courses, we will take this as a given. Thus, we can write down *learning* in the linear regression model as the following optimization problem:

$$\underset{\theta}{\text{argmin}} \left[\mathbb{E}_{(x,y)\sim\mathcal{D}}\mathcal{L}(y, \hat{y})\right]$$

That is, the best $\theta$ is the value of $\theta$ that minimizes the expected loss. Notice that though it does not appear in the above equation, $\hat{y}$ depends directly on $\theta$. To solve this optimization problem in general, i.e., for general losses and basis expansions, we must to turn to vector calculus as the foundation of general optimization techniques.

# 5 Calculus Revisited

Recall from your previous education in calculus that one can think about or define the derivative of a real-valued function, $f(x)$, as being the limit of the difference quotient:

$$f'(x) = \frac{df}{dx} = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h} \tag{3}$$

We highlight that this also prescribes one way of approximating a derivative when differentiation is computationally difficult. For interested readers, you may look up zeroth-order optimization to learn more. Of course, in general, the entire field of calculus prescribes exactly how functions change locally around a given parameter value. For example, several very useful derivatives of common functions include:

$$\begin{array}{ll} f(x) = x^n & f'(x) = nx^{n-1} \\ f(x) = \sin(x) & f'(x) = \cos(x) \\ f(x) = \tanh(x) & f'(x) = 1 - \tanh^2(x) \\ f(x) = \exp(x) & f'(x) = \exp(x) \\ f(x) = \log(x) & f'(x) = \frac{1}{x} \end{array}$$

Moreover, it is good to refresh yourself on the key rules of differentiation that allow us to easily differentiate compositions of functions. You will find practice exercises at the bottom of these lecture notes. I highly recommend working on these as this skill will prove critical to practical problems in class and on your exam.

- Sum Rule

$$\big(f(x) + g(x)\big)' = f'(x) + g'(x) = \frac{df(x)}{dx} + \frac{dg(x)}{dx}$$

- Product Rule

$$\big(f(x)g(x)\big)' = f'(x)g(x) + f(x)g'(x) = \frac{df(x)}{dx}g(x) + f(x)\frac{dg(x)}{dx}$$

- Chain Rule

$$(g \circ f)'(x) = \big(g(f(x))\big)' = g'(f(x))f'(x) = \frac{dg(f(x))}{df}\frac{df(x)}{dx}$$

- Quotient Rule

$$\Big(\frac{f(x)}{g(x)}\Big)' = \frac{f'(x)g(x) - f(x)g'(x)}{(g(x))^2} = \frac{\frac{df}{dx}g(x) - f(x)\frac{dg}{dx}}{(g(x))^2}$$

## 5.1 Gradients and Partial Derivatives

The above rules that we have recalled from your early calculus education can be generalized to real-valued functions $f$ that take vectors as inputs, i.e., $f : \mathbb{R}^n \mapsto \mathbb{R}$. Notice first that if we simply look at the $i^{th}$ index of our input vector and hold all other indices constant, then, as before, we can compute the derivative which we now denote $\partial f/\partial x_i$ which is known as the partial derivative:

**Partial Derivative** For a function $f : \mathbb{R}^n \to \mathbb{R}$, $x \mapsto f(x)$, $x \in \mathbb{R}^n$ of $n$ variables $x_1, \ldots, x_n$, we define the partial derivatives as follows:

$$\begin{aligned}
\frac{\partial f}{\partial x_1} &= \lim_{h\to 0} \frac{f(x_1 + h, x_2, \ldots, x_n) - f(x)}{h}, \\
\frac{\partial f}{\partial x_2} &= \lim_{h\to 0} \frac{f(x_1, x_2 + h, \ldots, x_n) - f(x)}{h}, \\
&\vdots \\
\frac{\partial f}{\partial x_n} &= \lim_{h\to 0} \frac{f(x_1, x_2, \ldots, x_n + h) - f(x)}{h},
\end{aligned}$$

and collect them in the row vector $\nabla f$ (gradient of $f$):

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix}.$$

Throughout the course we will always assume that the gradient of a vector valued function is a row vector. Now that we have seen how the gradient of a function is the collection of its partial derivatives, lets look at gradient field i.e., functions of the form : $f : \mathbb{R}^n \to \mathbb{R}^m$, and consider their gradients. In this case, we can again think about isolating the $i^{th}$ output of the function and know that gradient with respect to the $i^{th}$ output is simply a row vector of partial derivatives. So, it follows that the gradient of our function with respect to the input $\partial f/\partial x$ would be a matrix

$\in \mathbb{R}^{m \times n}$ because we have a row vector for each of the outputs so we have $m$ rows an $n$ columns. This Jacobian matrix can be expressed:

$$\nabla f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Where, $f_i$ represents the $i^{th}$ component of the vector-valued function $f$, and $x_j$ represents the $j$-th component of the input vector. Finally, let's write out what the shapes of a matrix value function with respect to a matrix of outputs. It is clear that for every input and output we need to have a partial derivative so if we have a function $f : \mathbb{R}^{n \times m} \to \mathbb{R}^{k \times l}$ then we will have a gradient which is a tensor of shape $(k \times l) \times (n \times m)$.

## 5.2 Vector Calculus Identities to Remember

Before moving on to seeing how vector calculus can help us in our goal of learning, let's revisit a few helpful identities:

$$\begin{aligned}
\frac{\partial \boldsymbol{x}^\top \boldsymbol{a}}{\boldsymbol{x}} &= \boldsymbol{a}^\top \\
\frac{\partial}{\partial X} f(X)^\top &= \left(\frac{\partial f(X)}{\partial X}\right)^\top \\
\frac{\partial}{\partial X} \text{tr}(f(X)) &= \text{tr}\left(\frac{\partial f(X)}{\partial X}\right) \\
\frac{\partial}{\partial X} f(X)^{-1} &= -f(X)^{-1} \frac{\partial f(X)}{\partial X} f(X)^{-1} \\
\frac{\partial \boldsymbol{a}^\top X \boldsymbol{b}}{\partial X} &= \boldsymbol{a}\boldsymbol{b}^\top \\
\frac{\partial}{\partial \boldsymbol{s}} (x - A\boldsymbol{s})^\top W (\boldsymbol{x} - A\boldsymbol{s}) &= -2(\boldsymbol{x} - A\boldsymbol{s})^\top W A
\end{aligned}$$

where the last identity assumes $W$ is symmetric. See lecture slides 3 for some basic proofs and strategies for proving these identities.

# A Lecture 2: Vector Differentiation

**Question 1** (Circle)**.** Consider a vector function $\mathbf{x}(t) = \begin{bmatrix}\cos t & \sin t\end{bmatrix}^\intercal$.

a. Draw the set of points that this function passes through.

b. To build intuition, draw the velocity vector at a few points by considering the direction that the point moves in.

c. Find the derivative $\mathrm{d}\mathbf{x}/\mathrm{d}t$. Draw this vector for some point t.

**Question 2** (Index notation)**.** Turn the following matrix-vector expressions into index notation:

a. $\mathrm{ABC}\mathbf{x}$

b. $\mathrm{Tr}(\mathrm{A})$

c. $\mathrm{Tr}(\mathrm{AB})$

d. $\mathbf{y}^\intercal \mathrm{A}^\intercal \mathbf{x}$

Turn the following index expressions back to matrix-vector notation:

a. $\sum_{ijk} A_{ij} B_{jk} C_{ki}$

b. $b_i + \sum_j A_{ij} b_j$

c. $x_i x_j$

d. $\sum_j \delta_{ij} a_j$

**Question 3** (Index notation proofs)**.** Using index notation, show that

1. $\mathbf{x}^\intercal \mathrm{A}\mathbf{y} = \mathbf{y}^\intercal \mathrm{A}\mathbf{x}$ if A is symmetric, i.e. $\mathrm{A} = \mathrm{A}^\intercal$.
2. $\mathbf{x}^\intercal \mathbf{y} = \mathrm{Tr}(\mathbf{x}^\intercal \mathbf{y}) = \mathrm{Tr}(\mathbf{y}^\intercal \mathbf{x})$, for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^D$.
3. $\mathrm{Tr}(\mathrm{ABC}) = \mathrm{Tr}(\mathrm{CAB})$.

**Question 4** (MML 5.5-5.6)**.** First find the dimensions, then the Jacobian. It's probably easiest here to use index notation.

a. $f(\mathbf{x}) = \sin(x_1)\cos(x_2)$, find $\mathrm{d}f/\mathrm{d}\mathbf{x}$.

b. $f(\mathbf{x}) = \mathbf{x}^\intercal \mathbf{y}$, find $\mathrm{d}f/\mathrm{d}\mathbf{x}$.

c. $f(\mathbf{x}) = \mathbf{x}\mathbf{x}^\intercal$, find $\mathrm{d}f/\mathrm{d}\mathbf{x}$.

d. $f(\mathbf{t}) = \sin(\log(\mathbf{t}^\intercal \mathbf{t}))$, find $\mathrm{d}f/\mathrm{d}\mathbf{t}$.

e. $f(\mathrm{X}) = \mathrm{Tr}(\mathrm{AXB})$ for $\mathrm{A} \in \mathbb{R}^{D\times E}$, $\mathrm{X} \in \mathbb{R}^{E\times F}$, $\mathrm{B} \in \mathbb{R}^{F\times D}$, find $\mathrm{d}f/\mathrm{dX}$.

**Question 5** (MML 5.7-5.8: Chain rule)**.** Comupte the derivatives $\mathrm{d}f/\mathrm{d}\mathbf{x}$ of the following functions.

- First, write out the chain rule for the given decomposition.

- Give the shapes of intermediate results, and make clear which dimension(s) will be summed over.
- Provide expressions for the derivatives, and describe your steps in detail. Providing an expression means specifying everything up to the point where you could implement it.
- Give the results in vector notation if you can.

a. $f(z) = \log(1+z), \qquad z = \mathbf{x}^\intercal\mathbf{x}, \qquad \mathbf{x} \in \mathbb{R}^D.$

b. $f(\mathbf{z}) = \sin(\mathbf{z}), \qquad \mathbf{z} = \mathrm{A}\mathbf{x} + \mathbf{b}, \qquad \mathrm{A} \in \mathbb{R}^{E\times D}$. What sizes are $\mathbf{x}$ and $\mathbf{b}$?

c. $f(z) = \exp(-\frac{1}{2}z), \qquad z = \mathbf{y}^\intercal\mathrm{S}^{-1}\mathbf{y}, \qquad \mathbf{y} = \mathbf{x} - \boldsymbol{\mu}.$

d. $f(\mathrm{A}) = \mathrm{Tr}(\mathrm{A}), \qquad \mathrm{A} = \mathbf{x}\mathbf{x}^\intercal + \sigma^2\mathrm{I}.$

e. $\mathbf{f}(\mathbf{z}) = \tanh(\mathbf{z}), \qquad \mathbf{z} = \mathrm{A}\mathbf{x} + \mathbf{b}, \qquad \mathrm{A} \in \mathbb{R}^{M\times N}.$

f. $f(\mathrm{A}) = \mathbf{x}^\intercal\mathrm{A}\mathbf{x}, \qquad \mathrm{A} = \mathbf{x}\mathbf{x}^\intercal.$

Remember: Generally, scalar functions are applied elementwise to vectors/matrices.

**Question 6** (Hessian of Linear Regression)**.** For the stationary point of linear regression, find the Hessian, and prove that it is positive definite, perhaps by making some assumptions. Discuss your assumptions.

**Question 7.** Prove the following theorem:

**Theorem A.1.** *Given a positive semi-definite matrix $S \in \mathbb{R}^{m\times m}$ that defines a Mahalanobis metric: $d_S(a,b) = \sqrt{(a-b)^\top S(a-b)}$, a feature vector $x' \in \mathbb{R}^m$, and a similarity theshold $\delta$, all vectors $x'' \in \mathbb{R}^n$ satisfying $d_S(x', x'') \leq \delta$ are contained within the axis-aligned orthotope:*

$$\left[x' - \delta\sqrt{d}, x' + \delta\sqrt{d}\right]$$

*where $d = diag(S)$, the vector containing the elements along the diagonal of $S$.*

**Question 8.** A colleague of yours suggests to study the transfer learning set up we discussed in class in a purely linear setting using linear models with basis expansion. Does this idea make sense? Formulate the problem and write down if and why you think transfer learning can be studied using linear models.