ORDINARY LEAST SQUARES REGRESSION

MATH 196, SECTION 57 (VIPUL NAIK)

Corresponding material in the book: For the computational process, Section 5.3 of the book is relevant. For comparing notation:

- X in the notes corresponds to A in the book
- $\vec{\beta}$ in the notes corresponds to $\vec{x}$ in the book
- $\vec{y}$ in the notes corresponds to $\vec{b}$ in the book

Executive summary

Words ...

(1) Consider a model where the general functional form is linear in the parameters. Input-output pairs give a system of simultaneous linear equations in terms of the parameters. Each row of the coefficient matrix corresponds to a particular choice of input, and each corresponding entry of the augmenting column is the corresponding output. In the "no-error" case, what we would ideally like is that the coefficient matrix has full column rank (i.e., we get unique solutions for the parameters assuming consistency) and does *not* have full row rank (i.e., we have some extra input-output pairs so that consistency can be used as evidence in favor of the hypothesis that the given functional form is correct). If the model is correct, the system will be consistent (despite potential for inconsistency) and we will be able to deduce the values of the parameters.

(2) Once we introduce measurement error, we can no longer find the parameters with certainty. However, what we *can* hope for is to provide a "best guess" for the parameter values based on the given data points (input-output pairs).

(3) In the case where we have measurement error, we still aim to choose a large number of inputs so that the coefficient matrix has full column rank but does not have full row rank. Now, however, even if the model is correct, the system will probably be inconsistent. What we need to do is to replace the existin output vector (i.e., the existing augmenting column) by the vector closest to it that is in the image of the linear transformation given by the coefficient matrix. Explicitly, if $\vec{\beta}$ is the parameter vector that we are trying to find, X is the coefficient matrix (also called the design matrix), and $\vec{y}$ is the output vector, the system $X\vec{\beta} = \vec{y}$ may not be consistent. We try to find a vector $\vec{\varepsilon}$ of minimum length subject to the constraint that $\vec{y} - \vec{\varepsilon}$ is in the image of the linear transformation given by X, so that the system $X\vec{\beta} = \vec{y} - \vec{\varepsilon}$ is consistent. Because in our setup (if we chose it well), X had full column rank, this gives the unique "best"choice of $\vec{\beta}$. Note also that "length" here refers to Euclidean length (square root of sum of squares of coordinates) when we are doing an *ordinary least squares regression* (the default type of regression) but we could use alternative notions of length in other types of regressions.

(4) In the particular case that the system $X\vec{\beta} = \vec{y}$ is consistent, we choose $\vec{\varepsilon} = \vec{0}$. However, this does not mean that ths is the actual correct parameter vector. It is still only a guess.

(5) In general, the more data points (i.e., input-output pairs) that we have, the better our guess becomes. However, this is true only in a probabilistic sense. It may well happen that a particular data point worsens our guess because that data point has a larger error than the others.

(6) The corresponding geometric operation to finding the vector $\vec{\varepsilon}$ is orthogonal projection. Explicitly, the image of X is a subspace of the vector space $\mathbb{R}^n$ (where n is the number of input-output pairs). If there are m parameters (i.e., $\vec{\beta} \in \mathbb{R}^m$). and we chose X wisely, the image of X would be a m-dimensional subspace of $\mathbb{R}^n$. In the no-error case, the vector $\vec{y}$ would be in this subspace, and we would be able to find $\vec{\beta}$ uniquely and correctly. In the presence of error, $\vec{y}$ may be outside

the subspace. The vector $\vec{y} - \vec{\varepsilon}$ that we are looking for is the orthogonal projection of $\vec{y}$ onto this subspace. The error vector $\vec{\varepsilon}$ is the component of $\vec{y}$ that is perpendicular to the subspace.

(7) A fit is impressive in the case that m is much smaller than n and yet the error vector $\vec{\varepsilon}$ is small. This is philosophically for the same reason that consistency becomes more impressive the greater the excess of input-output pairs over parameters. Now, the rigid notion of consistency has been replaced by the more loose notion of "small error vector."

Actions ...

(1) Solving $X\vec{\beta} = \vec{y} - \vec{\varepsilon}$ with $\vec{\varepsilon}$ (unknown) as the vector of minimum possible length is equivalent to solving $X^T X\vec{\beta} = X^T\vec{y}$. Note that this process does not involve finding the error vector $\vec{\varepsilon}$ directly. The matrix $X^T X$ is a square matrix that is symmetric.

(2) In the case that X has full column rank (i.e., we have unique solutions *if* consistent), $X^T X$ also has full rank (both full row rank and full column rank – it is a square matrix), and we get a unique "best fit" solution.

1. The simple case of fitting a linear function

1.1. **Recalling the no-error case.** Suppose f is a function of one variable x, defined for all reals. Recall that f is called a *linear* function (in the affine linear, rather than the homogeneous linear, sense) if there exist constants m and c such that $f(x) = mx + c$. Suppose the only thing you have access to is a black box that will take an input x and return the value $f(x)$. You have no information about the inner working of the program.

You do not know whether f is a linear function, but you believe that that may be the case.

(1) Suppose you are given the value of f at one point. This is useful information, but does not, in and of itself, shed any light on whether f is linear.

(2) Suppose you are given the values of f at two points. This would be enough to determine f uniquely under the assumption that it is linear. However, it will not be enough to provide evidence in favor of the hypothesis that f is linear. Why? Because *whatever* the values of the function, we can always fit a straight line through them.

(3) Suppose you are given the values of f at three points. This allows for a serious test of the hypothesis of linearity. There are two possibilities regarding how the three points on the graph of f look:
- The points on the graph are non-collinear: This is *conclusive* evidence *against* the hypothesis of linearity. We can definitely *reject* the hypothesis that the function is linear.
- The points on the graph are collinear: This is evidence *in favor of* the hypothesis of linearity, but it is not conclusive. We can imagine a function that is not linear but such that the outputs for the three specific inputs that we chose happened to fall in a straight line. As we discussed earlier, there are good reasons to treat the collinearity of points on the graph as *strong evidence* in favor of linearity. But it is not conclusive. For instance, for the function $f(x) = x^3 + x$, the points $x = -1$, $x = 0$, and $x = 1$ appear to satisfy the collinearity condition, despite the function not being linear.

(4) Suppose you are given more than three points and the values of f at all these points. This allows for an even stronger test of the hypothesis of linearity. There are two possibilities regarding how the corresponding points on the graph of the function look:
- The points on the graph are not all collinear: This is *conclusive* evidence *against* the hypothesis of linearity. We can definitely *reject* the hypothesis that the function is linear.
- The points on the graph are collinear: This is evidence *in favor of* the hypothesis of linearity, but is not conclusive. After all, we can draw curves that are not straight lines to fill in the unknown gaps between the known points on the graph. However, it does constitute strong evidence in favor of the linearity hypothesis. And the more points we have evaluated the function at, the stronger the evidence.

Although the discussion that follows is general, we will use linear fitting as our motivating example.

We will use the term *data point* or *input-output pair* for a pair of values (x, y) with y the (actual or somewhat erroneous) value of $f(x)$. We will use the terms *line-fitting* (more generally, *curve-fitting*) and *parameter estimation*.

1.2. **The introduction of error.** Broadly speaking, there are two kinds of errors that introduce minor inaccuracies: *modeling errors (arising from the model being only an approximate representation of reality)* and *measurement errors*. Modeling errors refers to the fact that the linear model may be only an approximation to reality, albeit still a useful approximation. Measurement errors refers to the fact that the black box we use to measure the fuction values may not measure the values accurately. Note: There's another kind of modeling error, namely uncertainty about whether the model is right at all. We'll discuss that type of error later. The modeling errors we are talking about right now are errors where the model is a reasonable approximation of reality but not exactly correct.

In the physical science world, there are situations where modeling errors are zero (or near-zero): the world actually does conform to the actual model. The only errors we need to confront here are measurement errors. Note that measurement errors confound the process of *finding* the parameter values using input-output pairs. Once the parameters are found, though, the model can be applied perfectly.

In the social science world, both measurement and modeling errors occur. In fact, in some sense, it is hard to conceptually separate measurement errors from modeling errors. That's because in the social sciences, it is harder to pinpoint what the "true value" of something means. Consider, for instance, a construct of "happiness" that can be measured through surveys. A theoretical model might attempt to predict happiness in terms of socio-economic status, income, race, education levels, physiological measures (heart rate etc.) and other measures. The predicted values from the model may differ from the survey-measured happiness. One could argue that this difference arises from measurement error: the survey isn't measuring true happiness. One could also argue that there is modeling error: the model isn't calculating happiness correctly. Or, one could argue that it's a mix: the model is not correct, but there are also measurement errors. But a deeper critique would be to argue against the legitimacy of happiness as a construct itself, i.e., to argue against the idea that there is a "true value" of happiness that our model and measurement tools are getting at.

Explicitly, we can think of the relationship as follows:

Value predicted by model $\overset{\text{modeling error}}{\leftrightarrow}$ "True" value $\overset{\text{measurement error}}{\leftrightarrow}$ Measured value

The deeper critique would say that there is no true value we are getting at, and that the relationship is something of the form:

Value predicted by model $\overset{\text{all error}}{\leftrightarrow}$ Measured value

For our present purposes, the validity of the deeper critique is not relevant. All we are trying to do is use our measured value to find the parameters for the model (for instance, in the linear model case, we are using input-output pairs to find the values of m and c).

Thus, for our present purposes, *we will use the term measurement error for the total of measurement error and errors arising from the model being somewhat inaccurate.*

1.3. **The middle case for error is what's interesting.** There are three possibilities:

- The measurement and modeling errors are zero, or so close to zero, that for practical purposes they don't affect us at all.
- The measurement and modeling errors are so huge that the model is practically useless. For instance, huge measurement errors means that finding the parameters in the model from input-output pair observations is difficult. Huge modeling errors means that the model isn't that predictive of actual values anyway (over and above the problem of finding the parameters).
- The measurement and modeling errors are big enough that they do confound the process of determining the parameters and predicting outputs from inputs. They are not, however, big enough to make the model useless. So, it's hard enough to be challenging, but not hard enough to be useless.

1.4. **The core idea of regression.** Let's concentrate on the linear model $f(x) = mx + c$. In the absence of modeling and measurement errors, two input-output pairs suffice to determine f uniquely. Three or more input-output pairs can be used to obtain additional confirmation of the model, but they do not actually help with *finding* the parameters. In this sense, once we have the input-output values at two points, then additional input-output data is redundant.

In the *presence* of errors, just using two points is not good enough, because the values at those two points need not be the correct values. Sure, we *can* get a line through those two points, and that line would probably be somewhat related to the actual line we are trying to get. But once we introduce error, then we can use

more input-output pairs to *improve* the accuracy of our guesses for m and c. If we're not absolutely sure, more data is (in a probabilistic sense) helpful. It's *possible* that the new data could reduce the accuracy of our estimate, but in general we expect our estimate for the line to improve as we have more points.

Let's think about this a bit more. Suppose we are given three input-output pairs (x_1, y_1), (x_2, y_2), (x_3, y_3). We plot these three points on the xy-plane. In the absence of errors, these three points would be collinear, and we could join them and get the line that describes the function. In the presence of errors, however, it is quite likely that the points will not be collinear. Further, even if they happen to be collinear, we cannot be completely sure that the line of these three points is the correct line. Perhaps the measurement errors at the points caused the values to get distorted in a collinear fashion.

So, finding the *actual* line for sure is not possible, but we can still ask a question such as: "what is the *best* line that fits the three points as closely as possible?" That's not necessarily the same as the actual line, but it's the closest fit we can find. In other words, what is the line such that the total distance between the observed points and the actual line is as little as possible? If that question sounds vague, that's a feature, not a bug. There are many ways of trying to make it precise, and we'll be getting into those in the next section. But you should first be on board with the *goal* that we have here: trying to find a line such that the observed points come as close to the line as possible.

What if we have more than three points? The idea above continues to work. As we add more new points, though, the best line fit may keep changing to incorporate the new data. In general, we expect the best line fit to come closer and closer to the *true* line. The process is probabilistic, so it may well happen that in some cases, adding new data leads to a somewhat worse fit. It's just that, overall, we expect the fit to get better and better as we add more and more data.

The description above seems wishy-washy and more like a test of artistic and visual skill than a procedure that can be handled mathematically. And, it's not clear where linear algebra comes in. Let's remedy that.

2. Getting linear algebra into the picture

2.1. **The no-error case with just enough information.** As before, we're trying to find a function f of one variable that is known to be of the form $f(x) = mx + c$ where m and c are real numbers. Let's write $y = f(x)$. It suffices to know the values (x_1, y_1) and (x_2, y_2) for $x_1 \neq x_2$. We could then use these input-output pairs to find the values of m and c. Explicitly, we'd set up the following linear system:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

The coefficient matrix is:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix}$$

If $x_1 \neq x_2$, the coefficient matrix has full rank. It's a square matrix, so this means full row rank and full column rank. Let's review what both these mean:

- Full *row* rank means that the system can always be solved for the parameters m and c. In particular, if there is any doubt about the model, just using two points cannot be used to test the validity of the model.
- Full *column* rank means that the solution, if it exists, is unique, i.e., we find a unique m and c satisfying the system.

Of course, in this case, we already knew this geometrically: given any two points, there is a unique line joining them. There is also an information-theoretic interpretation: there are two unknown parameters m and c, so we need two pieces of information to pin them down, and that information can be given in the form of two input-output pairs.

Let's be clear what is going on:

- Input values of input-output pairs $\leftrightarrow$ Rows of coefficient matrix
- Output values of input-output pairs $\leftrightarrow$ Entries of output vector, aka augmenting column

- Parameters for the line (namely, the slope and the intercept) $\leftrightarrow$ *coordinates* of the *parameter vector* $\begin{bmatrix} c \\ m \end{bmatrix}$

2.2. **The no-error case with extra information.** Continuing with the same setup as above, suppose that instead of just two input-output pairs, we are given n input-output pairs (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n). We need to find the values of the parameters m and c. The linear system we set up is:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ . & . \\ . & . \\ . & . \\ 1 & x_n \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}$$

We assume that $x_1, x_2, \ldots, x_n$ are all distinct points. The coefficient matrix is:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ . & . \\ . & . \\ . & . \\ 1 & x_n \end{bmatrix}$$

Note that although all the rows are distinct, the rank is still 2, since it is constrained by the number of columns. Thus, the coefficient matrix has full column rank but *not* full row rank. The significance is as follows:

- The fact that it has full column rank means that if a solution exists, then it is unique. This corresponds to the geometric fact that if the points are collinear, there is a unique line through them.
- The fact that it does *not* have full row rank means that a solution need not exist. What this means is that if our model is incorrect, it is potentially falsifiable using the model. This connects to our earlier observation that using more than two points gives us the opportunity of falsifying the model. If the system is consistent, that is evidence in favor of the hypothesis of linearity.

2.3. **The error case with just two points.** Return to the case of the linear model with just two input-output pairs (x_1, y_1) and (x_2, y_2) given to us. However, there is now measurement and/or modeling error. Thus, we have:

$$\begin{aligned} y_1 &= f(x_1) + \varepsilon_1 \\ y_2 &= f(x_2) + \varepsilon_2 \end{aligned}$$

where ε_1 and ε_2 are the error terms. Or equivalently:

$$\begin{aligned} f(x_1) &= y_1 - \varepsilon_1 \\ f(x_2) &= y_2 - \varepsilon_2 \end{aligned}$$

The problem, of course, is that we don't *know* what ε_1 and ε_2 are. We can now formulate the linear system:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} - \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix}$$

A reasonable assumption is that the errors are symmetrically distributed about zero, and the median and modal value of each error is zero. In this case, our "best fit" comes from solving:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

We can now determine a unique best fit line. This is simply the line joining the two points (x_1, y_1) and (x_2, y_2). Note that this is not *the* line for f: it's simply our best guess given the available information.

2.4. **More than two data points, with error.** Suppose now that we have more than two points. Due to the presence of error, new points *do* add value. As before, we have points (x_1, y_1), (x_2, y_2), ..., (x_n, y_n). The "true value" $f(x_i)$ equals $y_i - \varepsilon_i$ where ε_i is the (unknown) error term.

The linear system is:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 1 & x_n \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix} - \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \cdot \\ \cdot \\ \cdot \\ \varepsilon_n \end{bmatrix}$$

Let's say the points actually happen to be collinear. In that case, the best fit line is obvious: it is the line that passes through the points (x_i, y_i). What that means in the context of this system is that the system:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 1 & x_n \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}$$

i.e., the system assuming no error, is consistent. Note that full column rank means that if a solution exists, it is unique, which in this case translates to saying that the no-error solution is a unique line, i.e., that the values of m and c are uniquely determined. Note that these still need not be the actual values. But they are the best guess we can make given the information.

In the domain-range language, the case where the points are all collinear corresponds to the case where the observed output vector is in the range of the linear transformation corresponding to the coefficient matrix.

What, now, if the system:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 1 & x_n \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}$$

is inconsistent? This means that we cannot get a line that fits all the data points. What we'd ideally like is a line that comes close to fitting the data points. In other words, we are trying to find an error vector $\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \cdot \\ \cdot \\ \cdot \\ \varepsilon_n \end{bmatrix}$ such that the system below is consistent:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 1 & x_n \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix} - \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \cdot \\ \cdot \\ \cdot \\ \varepsilon_n \end{bmatrix}$$

In other words, we want an error vector such that the difference:

$$\begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix} - \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}$$

is in the range (image) of the linear transformation given by the coefficient matrix.

Now, let's be clear: the choice of error vector is far from unique. An obvious choice that would work is to take the error vector $\vec{\varepsilon}$ to equal $\vec{y}$, so that the difference is the zero vector, giving $m = c = 0$. One could also imagine many other choices that work (this will become clearer in the subsequent discussion). Given lots of possible error vectors we can subtract, what vector should we choose?

The idea of the *best fit* is that we want to choose the error vector to be *as small as possible*. Small in what sense? That depends on the type of regression we are interested in doing. The most typical type of regression (both due to its computational ease and its conceptual significance) is called *ordinary least squares* regression: we try to find a candidate error vector of minimum possible length (in the sense of the square root of the sum of squares of the coordinates) so that subtracting it off gives us an output vector that is in the image.

2.5. **Taking stock: closest vector in the range.** Let's give names to the vectors. Suppose:

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{bmatrix}, \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}, \vec{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ . \\ . \\ . \\ \varepsilon_n \end{bmatrix}$$

The coefficient matrix can be written as:

$$X = \begin{bmatrix} \vec{1}_n & \vec{x} \end{bmatrix}$$

where $\vec{1}_n$ is used as shorthand for the all ones vector with n coordinates.

Define a new vector:

$$\vec{z} = \vec{y} - \vec{\varepsilon}$$

The vector $\vec{z}$ can be thought of as our *best estimate* for the "true value" of the output vector. $\vec{z}$ is the vector we want in the image of the linear transformation given by the coefficient matrix. Pictorially, $z_1, z_2, \ldots, z_n$ are such that the points (x_1, z_1), (x_2, z_2), $\ldots$, (x_n, z_n) are collinear, and the line we use to fit these points gives us our best guesses for m and c.

Note that finding the true vector $\vec{z}$ is equivalent to finding the vector $\vec{\varepsilon}$ once $\vec{x}$ and $\vec{y}$ are known. We can reformulate our goal as follows:

> Given the vectors $\vec{x}$ and $\vec{y}$, find the vector $\vec{z}$ in the image of the linear transformation given by the matrix $\begin{bmatrix} \vec{1}_n & \vec{x} \end{bmatrix}$ that is closest to the vector $\vec{y}$.

The matrix:

$$X = \begin{bmatrix} \vec{1}_n & \vec{x} \end{bmatrix}$$

defines a linear transformation $\mathbb{R}^2 \to \mathbb{R}^n$, so its image is a subspace of $\mathbb{R}^n$. Since the matrix has rank 2, the image is a two-dimensional subspace of $\mathbb{R}^n$, i.e., it is a plane in $\mathbb{R}^n$. Explicitly, it is a plane through the origin with basis given by the vectors:

$$\begin{bmatrix}1\\1\\.\\.\\.\\1\end{bmatrix}, \begin{bmatrix}x_1\\x_2\\.\\.\\.\\x_n\end{bmatrix}$$

The first basis vector corresponds to the outputs that we would get for the line $y = 1$ (i.e., the case $c = 1, m = 0$). The second basis vector corresponds to the outputs that we would get for the line $y = x$ (i.e., the case $c = 0, m = 1$).

The case $r = 3$ may be particularly easy to visualize: the image of the linear transformation (the set of possible outputs that could actually arise from lines) is a plane in $\mathbb{R}^3$, spanned by the vectors:

$$\begin{bmatrix}1\\1\\1\end{bmatrix}, \begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$$

The first basis vector corresponds to the outputs that we would get for the line $y = 1$ (i.e., the case $c = 1, m = 0$). The second basis vector corresponds to the outputs that we would get for the line $y = x$ (i.e., the case $c = 0, m = 1$). Various linear combinations of these possibilities correspond to various values of m and c.

The actual observed output $\vec{y}$ is a vector in $\mathbb{R}^3$, but it is not necessarily in the plane. We need to find the vector $\vec{z}$ in the plane that is "closest" to $\vec{y}$.

Pictorially, how would one find the closest vector in a plane to a vector not in the plane? We know the answer: orthogonal projection. Drop a perpendicular. That's the fundamental idea that generalizes, even to cases where we are operating in a generic $\mathbb{R}^n$ instead of $\mathbb{R}^3$.

Let's make the correspondence clear:

- Set of all vectors $\begin{bmatrix}c\\m\end{bmatrix} \leftrightarrow$ Set of all possible linear functions (we are trying to pick the best fit from this set of possibilities)
- Image of the linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^n \leftrightarrow$ Set of all possible outputs for the given inputs for which the data points are collinear.
- Observed vector $\vec{y}$ is in the image plane $\leftrightarrow$ the data points are already collinear, so the "best fit" is obvious (even though it still may not be the correct fit)
- Observed output vector $\vec{y}$ is not in the image plane $\leftrightarrow$ the data points are not already collinear.
- Closest vector $\vec{z}$ in the image plane to the observed output vector $\vec{y} \leftrightarrow$ the best line fit for the data points.

2.6. **Regressions, summarized.** Here is a reasonable high-level summary of what we have done:

- The original "no-error case" goal involved solving a linear system with a given coefficient matrix and a given output vector. The absence of error guaranteed that the output vector would indeed be in the image of the coefficient matrix.
- The "with-error case" has a coefficient matrix and an output vector. The output vector is no longer necessarily in the range of the coefficient matrix. We replace the output vector by the closest vector to it that is in the image of the coefficient matrix. The vector that we subtract in the process is our guess for the error vector. Another way of putting it is that we are trying to choose the error vector to be as small as possible. Here, we define "small" as meaning that the length (the square root of the sum of squares of coordinates) is as small as possible.
- Finding the closest vector geometrically means "dropping a perpendicular" which can be operationalized using matrix multiplications (we will not go into the computational details here).

3. Ordinary least squares regression: the general setting

3.1. **Linearity in parameters is what is crucial.** The example we discussed above involves the case of a linear function of one variable (in the affine linear sense, so that there may be a nonzero intercept).

Describing such a function required two parameters, one a slope parameter m and one an intercept parameter c. Moreover, the example was also linear in the parameters. Explicitly, it has the form:

$$f(x) = (1)c + (x)m$$

Thus, a particular input x_i with output y_i gives an equation of the form:

$$1c + x_i m = y_i$$

The corresponding row of the augmented matrix is thus:

$$\begin{bmatrix} 1 & x_i & | & y_i \end{bmatrix}$$

The corresponding row of the coefficient matrix is:

$$\begin{bmatrix} 1 & x_i \end{bmatrix}$$

Now, we could consider some more complicated examples. For instance, consider a model with a functional form:

$$f(x) = A\cos x + B\sin x$$

The function f described here is not a linear function at all. However, it is linear in the parameters A and B, and those are what we are trying to determine using input-output pairs. Each input-output pair thus gives a linear equation. Explicitly, each piece of information of the form $f(x_i) = y_i$ gives rise to a linear equation of the form:

$$(\cos x_i)A + (\sin x_i)B = y_i$$

The corresponding row of the augmented matrix is:

$$\begin{bmatrix} \cos x_i & \sin x_i & | & y_i \end{bmatrix}$$

The corresponding row of the coefficient matrix is:

$$\begin{bmatrix} \cos x_i & \sin x_i \end{bmatrix}$$

We can thus apply the linear regression methods to this situation. However, this is a relatively unimportant case, so we will turn instead to another functional form that is linear in the parameters and is extremely important: the *polynomial*. But first, let's formulate the general picture.

3.2. **The full general setting: functions of one variable with multiple parameters.** Consider a function of one variable of the form:

$$f(x) = \beta_1 g_1(x) + \beta_2 g_2(x) + \cdots + \beta_m g_m(x)$$

where the functions $g_1, g_2, \ldots, g_m$ are all known functions, and the coefficients $\beta_1, \beta_2, \ldots, \beta_m$ are the parameters of the model. These are the parameters that we are trying to determine.

Each input-output pair (x_i, y_i) gives rise to an equation of the form:

$$\beta_1 g_1(x_i) + \beta_2 g_2(x_i) + \cdots + \beta_m g_m(x_i) = y_i$$

The corresponding row of the augmented matrix is:

$$\begin{bmatrix} g_1(x_i) & g_2(x_i) & \ldots & g_m(x_i) & | & y_i \end{bmatrix}$$

The corresponding row of the coefficient matrix is:

$$\begin{bmatrix} g_1(x_i) & g_2(x_i) & \ldots & g_m(x_i) \end{bmatrix}$$

Suppose we are given a collection of input-output pairs (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n). The linear system that we want to solve is:

$$\begin{bmatrix} g_1(x_1) & g_2(x_1) & \dots & g_m(x_1) \\ g_1(x_2) & g_2(x_2) & \dots & g_m(x_2) \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ g_1(x_n) & g_2(x_n) & \dots & g_m(x_n) \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \cdot \\ \cdot \\ \beta_m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}$$

Let's use vector notation:

$$\vec{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \cdot \\ \cdot \\ \beta_m \end{bmatrix}, \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}, X = \begin{bmatrix} g_1(x_1) & g_2(x_1) & \dots & g_m(x_1) \\ g_1(x_2) & g_2(x_2) & \dots & g_m(x_2) \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ g_1(x_n) & g_2(x_n) & \dots & g_m(x_n) \end{bmatrix}$$

We are essentially trying to solve the linear system:

$$X\vec{\beta} = \vec{y}$$

where the matrix X is determined completely by the *input* parts of the input-output pairs, and the vector $\vec{y}$ is the outputs.

If we were trying to solve this directly, we would try to make sure that the matrix X has full column rank m, i.e., we have enough input-output pairs to be able to uniquely determine all the parameters. If it does, then that's good enough. Obviously, a necessary condition for full column rank is that $n \geq m$. If the inputs are chosen wisely, this is also sufficient (to be more specific, we'd need to know the functions g_i).

Once we introduce error, though, the model changes. We're actually trying to solve the system:

$$X\vec{\beta} = \vec{z}$$

where $\vec{z}$ is the closest vector to $\vec{y}$ in the range of X. In other words:

- We first perform the orthogonal projection of $\vec{y}$ on the subspace of $\mathbb{R}^n$ that is the image of X. Note that this is a m-dimensional subspace of $\mathbb{R}^n$.
- If $\vec{z}$ is this orthogonal projection, we then solve the equation $X\vec{\beta} = \vec{z}$. Note that full column rank guarantees a unique solution.
- This solution is our best estimate for the parameter values.

3.3. **Functions of more than one variable.** We could have the same setup for a functional form that uses more than one variable:

$$f(x_1, x_2, \dots, x_r) = \beta_1 g_1(x_1, x_2, \dots, x_r) + \beta_2 g_2(x_1, x_2, \dots, x_r) + \dots + \beta_m g_m(x_1, x_2, \dots, x_r)$$

Everything in the description proceeds as with the single variable case. In some sense, the original input variables do not matter. What gets "seen" as far as linear regression is concerned is the coefficient matrix. Note that we now have to use *double subscripts*, one subscript to refer to which input we are talking about, and one subscript to refer to the coordinate of the input that we are referring to. Explicitly, the input-output pair information is:

$$\begin{aligned} f(x_{1,1}, x_{1,2}, \dots, x_{1,r}) &= y_1 \\ f(x_{2,1}, x_{2,2}, \dots, x_{2,r}) &= y_2 \\ \cdot &= \cdot \\ \cdot &= \cdot \\ \cdot &= \cdot \\ f(x_{n,1}, x_{n,2}, \dots, x_{n,r}) &= y_n \end{aligned}$$

The design matrix becomes:

$$X = \begin{bmatrix} g_1(x_{1,1}, x_{1,2}, \ldots, x_{1,r}) & g_2(x_{1,1}, x_{1,2}, \ldots, x_{1,r}) & \ldots & g_m(x_{1,1}, x_{1,2}, \ldots, x_{1,r}) \\ g_1(x_{2,1}, x_{2,2}, \ldots, x_{2,r}) & g_2(x_{2,1}, x_{2,2}, \ldots, x_{2,r}) & \ldots & g_m(x_{2,1}, x_{2,2}, \ldots, x_{2,r}) \\ \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \ldots & \cdot \\ g_1(x_{n,1}, x_{n,2}, \ldots, x_{n,r}) & g_2(x_{n,1}, x_{n,2}, \ldots, x_{n,r}) & \ldots & g_m(x_{n,1}, x_{n,2}, \ldots, x_{n,r}) \end{bmatrix}$$

3.4. **Terminology used.** We use the following terminology:

- The vector $\vec{\beta}$ is termed the *parameter vector*. Its coordinates β_i are termed the *parameters*, *effects*, or *regression coefficients*.
- The coefficient matrix X is termed the *design matrix*. The entries of the matrix are sometimes termed the *regressors* (there are some complications here worth a separate discussion, that we cannot afford right now).
- The vector $\vec{\varepsilon}$ is termed the *error term*, *disturbance*, or *noise*.

4. Polynomial regression: a special case of linear regression

4.1. **Wait, why are polynomials a special case of linear things?** An obvious point of confusion is how *polynomial* regression can be a special case of *linear* regression. After all, polynomial functions are *more* general than linear functions.

Answer: the *polynomial* refers to the nature of the function in terms of its input. The term *linear* refers to the nature of the function in terms of its parameters. Specifically, polynomial regression is used for situations where our model predicts a polynomial of bounded degree with unknown coefficients, and the goal of regression is to determine these coefficients.

4.2. **The setup.** Consider the general case of a polynomial of degree $\leq m$. Note that the number of arbitrary coefficients here is $m+1$. We will number the subscripts starting from 0, so the notation will differ somewhat from earlier. Explicitly, the general functional form is:

$$f(x) := \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_m x^m$$

Suppose we are given n distinct input-output pairs (x_i, y_i). The coefficient matrix X for the linear system (called the design matrix in linear regression terminology) has the form:

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^m \\ 1 & x_2 & x_2^2 & \ldots & x_2^m \\ \cdot & \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot \\ 1 & x_n & x_n^2 & \ldots & x_n^m \end{bmatrix}$$

X is a $n \times (m+1)$ matrix.

The vector $\vec{\beta}$ has $m+1$ coordinates and is:

$$\vec{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \cdot \\ \cdot \\ \cdot \\ \beta_m \end{bmatrix}$$

In the no-error case, we are trying to solve the equation:

$$X\vec{\beta} = \vec{y}$$

In the case with an error vector $\vec{\varepsilon}$, we are trying to solve the equation:

$$X\vec{\beta} = \vec{y} - \vec{\varepsilon}$$

where we are trying to keep $\vec{\varepsilon}$ as small as possible.

The matrix X defines a linear transformation $\mathbb{R}^{m+1} \to \mathbb{R}^n$.

The matrix X is called the *Vandermonde matrix* for the given collection of inputs x_i. Note that X is determined by the inputs alone, and is independent of the outputs.

4.3. **The no-error case.** In the case that there is no error, we are trying to solve:

$$X\vec{\beta} = \vec{y}$$

It turns out from some easy-to-verify-but-hard-to-think-of facts in polynomial theory that if all the values x_i are distinct, then the rank of X is $\min\{m+1, n\}$. In particular, the bare minimum number of inputs that we need is $m+1$: with $m+1$ distinct inputs, we get a square matrix X that has full rank. We can thus uniquely recover $\vec{\beta}$ from $\vec{y}$. Note, however, that $m+1$ inputs do not allow for independent confirmation of the hypothesis that f is a polynomial of degree $\leq m$, because any $m+1$ input-output pairs will fit a polynomial. To obtain independent confirmation, we need $m+2$ or more input-output pairs. Note that if we take $m+2$ or more input-output pairs, the coefficient matrix has full column rank $m+1$, so we still get a unique solution. However, since the row rank is not full, the possibility of inconsistency has now been opened up. If the system is still consistent, that is independence in favor of the hypothesis that f is indeed a polynomial of degree $\leq m$.

4.4. **The case of error.** The goal is to solve the system:

$$X\vec{\beta} = \vec{y} - \vec{\varepsilon}$$

where we are trying to keep $\vec{\varepsilon}$ as small as possible. We consider the image of X, and attempt to find the vector in that image that is closest to $\vec{y}$. In other words, we determine the orthogonal projection of $\vec{y}$ on the image of X. If we call that vector $\vec{z}$, we must then solve the equation:

$$X\vec{\beta} = \vec{z}$$

Note also that $\vec{\varepsilon} = \vec{y} - \vec{z}$.

Geometrically, the image of X is a $(m+1)$-dimensional subspace of $\mathbb{R}^n$.

5. Against overfitting

5.1. **Combining measurement error with uncertainty about the model.** We consider overfitting in connection with two related concerns: *measurement error* and *the fear that we have picked a completely wrong model*. We have dealt with both issues separately. It's now time to connect the dots.

- We have dealt with the issue of measurement error using the idea of linear regression: we collect a lot more points than the bare minimum needed to ascertain the values of the parameters, and then use ordinary least squares regression to determine the parameter values. Note that we need a lot more data points than the actual number of parameters once we are combating measurement errors.
- We have dealt with the concern that the model itself is completely wrong in the hypothesis testing discussion: The idea is to pick one or more input over and above the minimum number of inputs needed to determine parameters, and see if the system is still consistent. For instance, for the original example of a linear function of one variable with possibly nonzero intercept, we just need two points to find the line, but we need three or more points to obtain evidence in favor of the hypothesis of linearity.

Note that we are solving two very different types of problems with roughly the same solution. The problem of uncertainty about whether the model is correct is solved by using extra data points, but subject to a very stringent test: we demand that the extra data points keep the system consistent. The problem of measurement error is solved by using extra data points, but mostly just to obtain a better estimates.

What if both problems occur together? In other words, we need to contend with measurement error and with the problem that our model may be just wrong. In the presence of both measurement error and

uncertainty, we need not just a few more data points, we need a *lot more* data points. Instead of a stringent test of fitting the line or curve, we impose a requirement that the total error in the fit is small. By "total error" here we mean the length of the error vector $\vec{\varepsilon}$. There is an alternative related way of measuring goodness of fit called R^2 (also called the *coefficient of determination*).

5.2. **Trying to fit a generic polynomial.** The preceding discussion of polynomial regression assumed that we are trying to fit using a polynomial with a fixed bound on its degree. Let's consider a somewhat more generic setting.

We are given a function f and n input-output pairs for f. We have some reason to believe that f may be polynomial.

If we didn't have measurement error, the process would be simple: try fitting f with a constant polynomial, a polynomial of degree ≤ 1, a polynomial of degree ≤ 2, and so on, and continue until we get a perfect fit. If we manage to get a perfect fit using a polynomial of degree m where $m+1$ is less than n (the number of data points we have), that is strong evidence in favor of the hypothesis that f is polynomial, because the remaining $n-(m+1)$ outputs agreeing with the predictions on sheer chance would be unlikely. If we think in terms of solving linear systems, this is equivalent to requiring that the appropriate augmented entries just happen to be 0 even though the entries were chosen randomly. The larger the gap $n-(m+1)$, the stronger the evidence in favor of the hypothesis. If, on the other hand, we only achieve perfect fit at $n = m+1$, that's no evidence in favor of the polynomial hypothesis.

Suppose we do have measurement error. Let's say we have 1000 data points. In that case, we can always fit a polynomial of degree 999 perfectly without measurement error. But that's cheating. What would be more convincing would be a polynomial of small degree (such as 3) that, even if not a perfect fit, gives a low error. The idea, therefore, would be to start with constant polynomials, polynomials of degree ≤ 1, polynomials of degree ≤ 2, polynomials of degree ≤ 3. Each time we increase the degree of the polynomial, we add in a new parameter, allowing for more wiggle room to fit the data points. So, our fit keeps getting better and better. But, *it's not clear whether the improved fit is fitting the signal or the noise.*

The general idea in this case is to see each time we allow one more parameter whether that additional parameter adds significant explanatory value, i.e., whether it pushes down the measurement error considerably. In general, the fewer parameters we have (which in this case means "the smaller the degree of the polynomial we are using") the more impressive a good fit is. This is also related to Occam's Razor.

5.3. **Interpretation in terms of the innards of linear regression.** Let's consider the above in terms of how linear regression is actually executed. As above, consider that we have 1000 input-output pairs for f. The output vector for f is a vector in $\mathbb{R}^{1000}$.

Trying to fit this with a constant function involves finding the orthogonal projection on a one-dimensional subspace of $\mathbb{R}^{1000}$. Note that the specific description of the one-dimensional subspace actually depends on the input values.

Trying to fit this with a linear function involves finding the orthogonal projection on a two-dimensional subspace containing that one-dimensional subspace. Trying to fit with a polynomial of degree ≤ 2 involves finding the orthogonal projection on a three-dimensional subspace that contains the two-dimensional subspace. In general, trying to fit a polynomial of degree $\leq m$ involves projecting onto a subspace of dimension $m+1$. And the subspace for smaller degree is contained in the subspace for bigger degree.

Since the subspaces are growing, the distance from the vector $\vec{y}$ to the subspace is decreasing, because the subspace is spreading out more and more over the place. What we want is a situation where $\vec{y}$ already gets close enough to the subspace that going up in dimension does not bring one much closer.

5.4. **The general caution against overfitting.** A revealing quote is by mathematician and computer scientist John von Neumann: "With four parameters I can fit an elephant. And with five I can make him wiggle his trunk." Another is by prediction guru Nate Silver: "The wide array of statistical methods available to researchers enables them to be no less fanciful and no more scientific than a child finding animal patterns in clouds." In other words, don't be overimpressed by impressive fits if they use lots of parameters.

The concern behind *data mining* is a discrete version of the same concern.

6. The computational procedure for ordinary least squares regression

6.1. **The transpose of a matrix.** For a $n \times m$ matrix A, we denote by A^T the $m \times n$ matrix that is obtained by interchanging the role of rows and columns in A. Explicitly, the $(ij)^{th}$ entry of A^T equals the $(ji)^{th}$ entry of A, where $1 \leq i \leq m$ and $1 \leq j \leq n$. A^T is pronounced *A-transpose* and is called the *transpose* of A. Note that the letter T is a fixed letter (chosen because it is the first letter of the word "transpose") and should not be mistaken for a variable in the exponent.

The transpose of a matrix is a useful construction because taking dot products involves an interplay of rows and columns.

Note also that in the above scenario, A describes a linear transformation from $\mathbb{R}^m$ to $\mathbb{R}^n$, whereas A^T describes a linear transformation from $\mathbb{R}^n$ to $\mathbb{R}^m$. However, although it might be tempting to believe this, these linear transformations are not merely inverses of one another. The relationship between them is far more subtle and deep. There are some special kinds of square matrices, called *orthogonal matrices*, for which the transpose coincides with the inverse.

6.2. **The normal equation.** Recall that we are trying to solve:

$$X\vec{\beta} = \vec{y} - \vec{\varepsilon}$$

where we are attempting to pick $\vec{\varepsilon}$ to be as small as possible. In our typical setup, X is a $n \times m$ matrix, where n is the number of input-output pairs and m is the number of parameters. $\vec{\beta} \in \mathbb{R}^m$, whereas $\vec{y}$ and $\vec{\varepsilon}$ are both vectors in $\mathbb{R}^n$.

As discussed previously, this is achieved when $\vec{\varepsilon}$ is perpendicular to the image of the linear transformation given by X. In fact, $\vec{\varepsilon}$ is the *unique* choice of vector satisfying *both* these conditions:

(1) The system $X\vec{\beta} = \vec{y} - \vec{\varepsilon}$ is consistent (i.e., $\vec{y} - \vec{\varepsilon}$ is in the image of the linear transformation with matrix X).
(2) The vector $\vec{\varepsilon}$ is perpendicular to all the columns of the matrix X. In other words, $X^T\vec{\varepsilon} = \vec{0}$. Here, X^T is a $m \times n$ matrix, and the $\vec{0}$ appearing on the right is now a vector in $\mathbb{R}^m$.

Thus, if we multiply both sides of the equation in Condition (1) by X^T, we obtain:

$$X^T(X\vec{\beta}) = X^T(\vec{y} - \vec{\varepsilon})$$

This simplifies to:

$$X^TX\vec{\beta} = X^T\vec{y} - X^T\vec{\varepsilon}$$

By Condition (2), the term $X^T\vec{\varepsilon}$ becomes $\vec{0}$, and we are left with:

$$(X^TX)\vec{\beta} = X^T\vec{y}$$

Note that X^TX is a $m \times m$ square matrix, and both $\vec{\beta}$ and $X^T\vec{y}$ are vectors in $\mathbb{R}^m$. In other words, the system of equations that we are now solving for $\vec{\beta}$ has an equal number of variables and equations.

One small note. When we multiply by a matrix, we potentially lose information. In this case, we may be worried that solving the new system might give many solutions that are not solutions to the old system. However, it turns out that there is no loss of information in this case: the solution vectors $\vec{\beta}$ to the above are precisely the same as the solution vectors for $X\vec{\beta} = \vec{y} - \vec{\varepsilon}$ for the correctly chosen $\vec{\varepsilon}$.

Recall that our eventual goal is to find the vector $\vec{\beta}$, not the vector $\vec{\varepsilon}$. Our ability to obtain a new form of the equation that does not feature $\vec{\varepsilon}$ is therefore a feature, not a bug. The equation above is called the *normal equation*. We can attempt to solve this equation for the vector $\vec{\beta}$ given our knowledge of X and $\vec{y}$, without intermediately attempting to discover $\vec{\varepsilon}$.

In the case that the matrix X^TX is invertible, we can write the general solution as:

$$\vec{\beta} = (X^TX)^{-1})X^T\vec{y}$$

Note that the invertibility of X^TX is equivalent to X having full column rank (the statement is obvious in only one direction, so you'll have to take me on faith). Note, however, that unless the inverse of X^TX

is pre-computed, it is generally easier to solve the normal equation directly using Gauss-Jordan elimination rather than first compute the inverse and then multiply.

Finally, note that we *can* find the error vector $\vec{\varepsilon}$ once we know $\vec{\beta}$, as follows:

$$\vec{\varepsilon} = \vec{y} - X\vec{\beta}$$

6.3. **Multiple outputs for the same input.** We might sometimes be in a situation where we are performing a linear regression where the same input appears with two different outputs in different input-output pairs. The idea here is that we have attempted measuring the output for a given input multiple times, and came up with different values each time. We need to be a little more careful regarding the validity of general statements about rank, but otherwise, the procedure stays intact.

6.4. **A baby case: the line through three points.** We consider a very special case of linear regression: the case where we have three input-output pairs (x_1, y_1), (x_2, y_2), (x_3, y_3) and we are trying to fit $y = f(x)$ for a linear function $f(x) = \beta_0 + \beta_1 x$. We take β_0 as the first variable and β_1 as the second variable (note: in our earlier discussion, we had used $mx + c$ as the general description, but we're now using better notation). In the no-error case, we would be trying to solve:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

In the case that we have measurement error, we wish to project $\vec{y}$ onto the two-dimensional subspace that is the image of the (linear transformation given by the) coefficient matrix. The projection is the vector $\vec{y} - \vec{\varepsilon}$ and the difference vector $\vec{\varepsilon}$ is perpendicular to the image.

The image of the linear transformation given by the coefficient matrix is a plane in $\mathbb{R}^3$ with basis given by the column vectors of the matrix, i.e., the vectors:

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

In the language we used in the general description, we have:

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{bmatrix}, \vec{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}, m = 2, n = 3$$

The transposed matrix X^T is given as:

$$X^T = \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \end{bmatrix}$$

The product matrix is a 2×2 matrix given as follows:

$$X^T X = \begin{bmatrix} 3 & x_1 + x_2 + x_3 \\ x_1 + x_2 + x_3 & x_1^2 + x_2^2 + x_3^2 \end{bmatrix}$$

We also have:

$$X^T \vec{y} = \begin{bmatrix} y_1 + y_2 + y_3 \\ x_1 y_1 + x_2 y_2 + x_3 y_3 \end{bmatrix}$$

Thus, the system that we are trying to solve for β_0 and β_1 is:

$$\begin{bmatrix} 3 & x_1 + x_2 + x_3 \\ x_1 + x_2 + x_3 & x_1^2 + x_2^2 + x_3^2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} y_1 + y_2 + y_3 \\ x_1 y_1 + x_2 y_2 + x_3 y_3 \end{bmatrix}$$

The general version of this is described in Section 5.4 of the text, Example 5 (Page 243-244 in the 4th Edition).