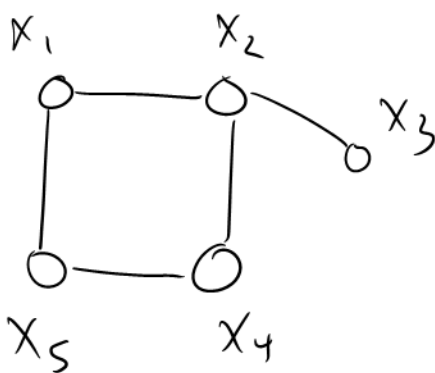


1 Outline

- Eliminate example
- Regression
 - Regression setup
 - Linear regression
 - Geometric picture for LMS
 - Normal eqns
 - Least Squares Cost
 - LMS: Stochastic gradient, Online vs. batch
 - Connections and probabilistic models
 - Applications of linear regression
 - Weighted linear least squares

2 Eliminate example



Calculate $p(x_3)$ using
Elimination ordering
(5, 1, 4, 2, 3)

for time/space, suppose
 $x_i \in \{1, \dots, k\}$

In this example we only have 2-cliques

some algebra

$$p(x_3) = \frac{1}{Z} \sum_{x_1} \sum_{x_2} \sum_{x_4} \sum_{x_5} \phi(x_1, x_2) \phi(x_1, x_5) \phi(x_2, x_3) \phi(x_2, x_4) \phi(x_4, x_5)$$

cost k^4 additions for each of k cells in the table for $p(x_3)$

⇓ rearrange and distribute sums

$$k^2 \text{ cost} \quad \times \quad k^2 \text{ cost} \quad + \quad k^3 \text{ cost} \quad + \quad k^3 \text{ cost}$$

$$p(x_3) = \frac{1}{Z} \sum_{x_4} \sum_{x_2} \phi(x_2, x_3) \phi(x_2, x_4) \sum_{x_1} \phi(x_1, x_2) \underbrace{\sum_{x_5} \phi(x_1, x_5) \phi(x_4, x_5)}_{m_5(x_1, x_4)}$$

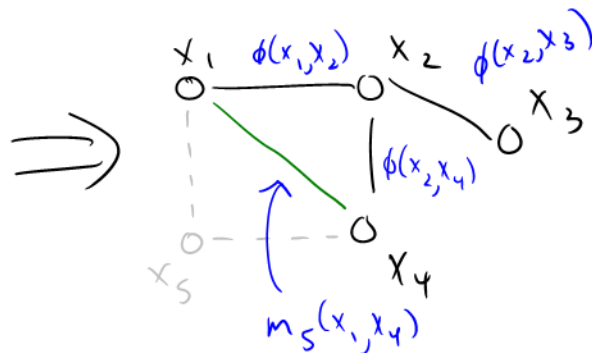
$$\underbrace{\phi(x_2, x_3) \phi(x_2, x_4)}_{m_1(x_2, x_4)}$$

$$\underbrace{\sum_{x_1} \phi(x_1, x_2) \phi(x_2, x_3) \phi(x_2, x_4)}_{m_2(x_3, x_4)}$$

$$\underbrace{\sum_{x_1} \phi(x_1, x_2) \phi(x_2, x_3) \phi(x_2, x_4) \sum_{x_5} \phi(x_1, x_5) \phi(x_4, x_5)}_{m_4(x_3)}$$

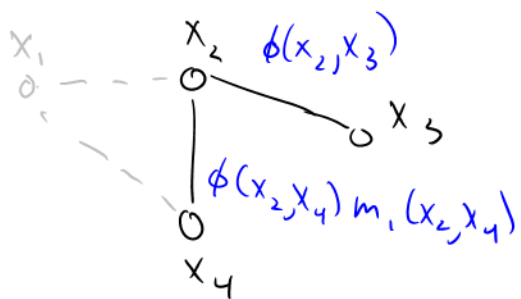
Graphically.

Eliminate x_5
gives the
"induced" graph
and potentials

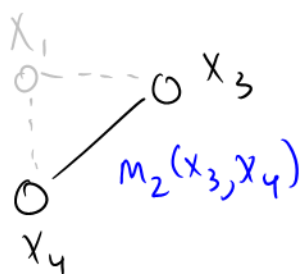


note that the
clique $\{x_1, x_2, x_4\}$
has a clique
potential that
factorizes as
edge potentials

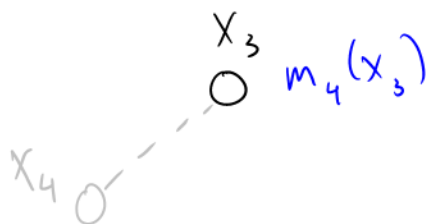
Eliminate X_1



Eliminate X_2



Eliminate X_4



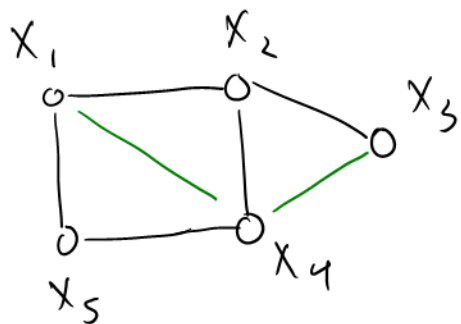
This gives

$$p(x_3) = \frac{1}{Z} m_4(x_3)$$

$$\text{and } 1 = \sum_{x_3} p(x_3) = \frac{1}{Z} \sum_{x_3} m_4(x_3)$$

$$\Rightarrow Z = \sum_{x_3} m_4(x_3)$$

The reconstituted graph is then



which has a maximal clique size 3

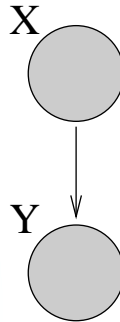


Figure 1: Graphical model illustrating linear regression

3 Regression Setup

We have observed a set of data:

$$D = \{(x_1, y_1), \dots, (x_N, y_N)\}$$

and we would like to model the data as in figure 1. What are the probabilities associated with the graphical model? $p(x)$ and $p(y|x)$.

Often you will find a problem where you don't care about modeling $p(x)$, but you are interested in modeling $p(y|x)$. (prediction, etc.)

In the discrete case with sufficient data, we did not need to make any assumptions about the relationship between x_i and y_i . What if we're in the continuous case? if there is insufficient data? We must make some assumptions about this relationship.

Linear Regression:

(Draw some picture of data and a regression line)

Assume there is a sort of linear relationship.

$$Y_n = \beta^T x_n + \epsilon_n$$

and further assume that the “error terms” ϵ_n are i.i.d. with mean 0.

Note this implies that the expected value of Y_n is a linear function of x_n :

$$E[Y_n|x_n] = \beta^T x_n$$

Our task now is to estimate the regression coefficients β .

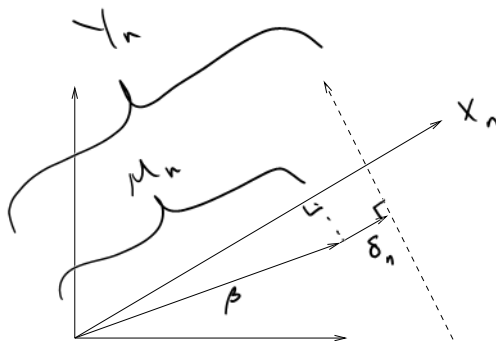


Figure 2: Illustration of the ideas behind LMS.

4 LMS: geometric perspective

We can take a small step back and ignore the probabilistic assumptions. Instead take a geometric view. A “good” estimate for β will result in the observed y_i 's being close to the predictions, the $\beta^T x_i$'s. Our task is then to find $\hat{\beta}$ such that Euclidean distance $\|y - \hat{\beta}^T x\|$ is minimized. We will connect this back to probabilistic models later.

4.1 Single point geometry for the *parameter* β

First we consider an example with only a single observation (x_n, y_n) .

For this example, let $x_n = [x_n^1, x_n^2]^T$ and y_n continuous, scalar. Suppose we have an initial guess for β

In figure 2 (draw it out), you need to move β exactly δ to get $y_n = \beta^T x_n$. What is δ_n ?

If $\|x\| = 1$ then $\beta^T x_n$ is the projection of β onto x_n and

$$\begin{aligned}\mu_n &= \beta^T x_n \\ \delta_n &= (y_n - \mu_n)x_n\end{aligned}$$

or, when $\|x_n\| \neq 1$:

$$\begin{aligned}\mu_n &= \beta^T x_n / \|x_n\| \\ \delta_n &= (y_n / \|x_n\| - \mu_n)x_n / \|x_n\|\end{aligned}$$

Then:

$$\beta^* \leftarrow \beta + \frac{1}{\|x_n\|} (y_n - \beta^T x_n)x_n / \|x_n\|$$

Note that in this case we have many possible solutions for β that would give us $y_n = \beta^T x_n$ (all the points on the dashed line).

4.2 Extension to more than one datapoint

In the previous case: $K = 2$ (dimension of x_n), $N = 1$ (number of x_n). Extend to $N = 2$, as in figure 3.

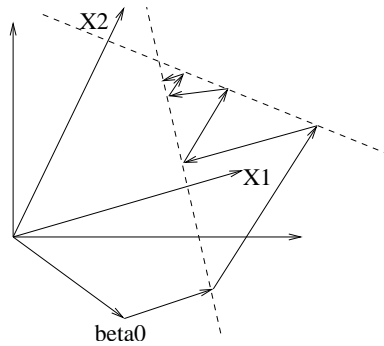


Figure 3: How the LMS algorithm proceeds for two datapoints.

Explain: with two points, there is an exact solution for β . When there are more than two points (i.e., $N > K$), the problem is *overconstrained* and we hope to converge to some point within a small area of space.

(Draw figure 6.3 in book)

4.3 LMS equation

Instead of using $\frac{1}{\|x_n\|}$, replace it with a “step size” ρ .

This gives us the LMS equation:

$$\beta^{(t+1)} \leftarrow \beta^{(t)} + \rho(y_i - \beta^{(t)T} x_n)x_n.$$

Now, for $\rho > 0$ sufficiently small, if we run the update a number of times, it will converge eventually to some β .

(I am cheating a bit since I haven’t specified which data point x_n is, but we’ll see in a moment.)

How size does ρ need to be? Answer $0 < \rho < 2/\lambda_{max}(X^T X)^2$. (Give some brief intuition using single point example. $\|x_n\|^2$ is then the max eigenvalue and $2/\|x_n\|^2$ ensures that at each step we get closer to the “right” point).

We skip the proof of the convergence results which are given in the book.

5 Normal equations: geometric view of the predicted values

Linear Algebra: good reference is Strang.

Let’s generalize this to N large.

Let $\hat{y} = [\hat{y}_1, \dots, \hat{y}_N]^T$, where $\hat{y}_n = \beta^T x_n$.

Let X be an N by K matrix, where each row is x_n^T (draw out).

Now, $\hat{y} = X\beta$.

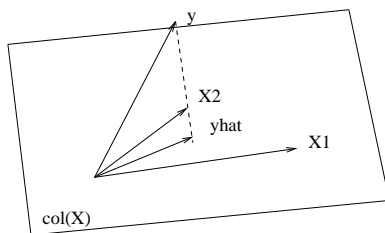


Figure 4: Column space and the normal equations

Column space of X : let X_1 be the first column of X , X_2 is the second column. Plot these two points (in N dimensional space), as in figure 4. Note:

$$\hat{y} = X_1\beta_1 + X_2\beta_2$$

When $\epsilon = 0$, y would lie on the column space of X (it is a linear function of X).

In reality, $\epsilon \neq 0$, so y is not on column space of X .

Intuition: choose a \hat{y} that best captures y but lies on the column space. Using Euclidean distance, you know that the shortest distance between a point and a plane is the orthogonal line:

Orthogonal: if X and Y are orthogonal, then $X^T Y = 0$.

So $X^T(y - X\beta) = 0$, or $X^T X\beta = X^T y$. This is the **normal equation**.

Quick note on normal equation: you can use e.g. Matlab to solve for β directly here:

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

6 Least Squares Cost Function

Our geometric Euclidean distance perspective immediately translates into a least square cost function perspective.

$$\begin{aligned} \arg \min_{\beta} \|y - \beta^T x\| &= \arg \min_{\beta} \|y - \beta^T x\|^2 \\ &= \arg \min_{\beta} \sum_n (y_n - \beta^T x_n)^2 = \arg \min_{\beta} \sum_n \epsilon(\beta)^2 \end{aligned}$$

Define: $J(\beta) \equiv \frac{1}{2} \sum_{n=1}^N \epsilon_n^2 = \frac{1}{2} \sum_{n=1}^N (y_n - \beta^T x_n)^2$. This is called the least squares cost function.

Goal: find β such that $J(\beta)$ is minimized:

$$\min_{\beta} \frac{1}{2} \sum_{n=1}^N (y_n - \beta^T x_n)^2$$

Exactly like section from week 1, this is a convex function. Take the derivative wrt β , set it to 0, and solve for β :

$$\frac{\partial J}{\partial \beta^i} = - \sum_{n=1}^N (y_n - \beta^T x_n) x_n^i$$

If we put all of these partial derivatives together in a vector, setting to zero, we get the gradient:

$$\begin{aligned} \nabla J &\rightarrow - \sum_{n=1}^n (y_n - \beta^T x_n) x_n = 0 \\ -X^T (y - X\beta) &= 0 \\ X^T X \beta &= X^T y \end{aligned}$$

which is again the normal equations. This gradient represents the direction to step β in in order to move the most directly to the optimal solution. The point of the exercise: both derivations minimize the Euclidean cost of the error term.

7 LMS and stochastic gradient descent

From above, we see that $\sum_{n=1}^N (y_n - \beta^T x_n) x_n$ is the steepest decent for a batch (N) datapoints.

Lots of redundancy; sometimes don't have all of them at the same time. (eg. a stream of data) Want to reduce the time/space you spend on this. Use *stochastic gradient*:

- Choose random (x_n, y_n) from dataset.
- Update β according to LMS update:

$$\beta^{(t+1)} \leftarrow \beta^{(t)} + \rho (y_n - \beta^{(t)T} x_n) x_n$$

Since we are choosing our datapoint randomly, we see that, in expectation, we are traveling in the direction of the gradient.

8 Online vs Batch

Batch algorithm: From the normal equations, one can derive the solution for minimizer β^* .

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

This is a good way of doing a batch calculation for β^* .

Online algorithm (LMS): Suppose you receive a stream of datapoints, or it is cumbersome to store all the datapoints in memory. For these cases, it is appropriate or faster to use an online algorithm. In the stream case, you can perform LMS update as you receive new datapoints. You would not want to do the batch calculation every time you receive a new datapoint, especially if the number of covariates K is large.

(For other problems where a nice closed form solution does not exist or is difficult to calculate, LMS or ideas from LMS may make the problem easier. eg. neural net backprop)

9 Connections

We have thus far focused on the geometric side of things. Let us turn to the probabilistic side.

9.1 Maximum Likelihood Interpretation

We return to our probabilistic model, and we assume normally distributed errors:

$$Y_n = \beta^T x_n + \epsilon_n$$

where $\epsilon_n \stackrel{iid}{\sim} N(0, \sigma^2)$.

Equivalently, $Y_n - \beta^T x_n = \epsilon_n \stackrel{iid}{\sim} N(0, \sigma^2)$ which gives us:

$$p(y_n|x_n, \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{\frac{-1}{2\sigma^2}(y_n - \beta^T x_n)^2\right\}$$

Let's write out likelihood for entire dataset with N points:

$$\begin{aligned} p(y|x, \theta) &= \prod_{n=1}^N p(y_n|x_n, \theta) \\ &= \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left\{\frac{-1}{2\sigma^2} \sum_{n=1}^N (y_n - \beta^T x_n)^2\right\} \end{aligned}$$

Again, like the first week of section, let's look at the log likelihood:

$$\begin{aligned} l(\theta; x, y) &= \log p(y|x, \theta) \\ &= \frac{-1}{2\sigma^2} \sum_{n=1}^N (y_n - \beta^T x_n)^2 + c \\ &= -\frac{1}{\sigma^2} J(\beta) + c \end{aligned}$$

where c is some constant that does not depend on β .

Thus maximizing the likelihood is equivalent to minimizing $J(\beta)$, the least squares cost function.

Note that it be of interest to estimate σ^2 as well. One may easily calculate the MLE, $\hat{\sigma}^2 = RSS/N$ where $RSS = 2J(\beta)$, the residual sum of squares. The bias corrected estimate is $RSS/(N - K)$ where K is the degrees of freedom i.e. number of columns in X . Likewise, estimates can be obtained for predicted values \hat{y}_n and also for the regression coefficients $\hat{\beta}$.

9.2 Quasi-likelihood

Suppose we only assume that the ϵ_i 's are uncorrelated and have equal variance (homoskedacity). We have weakened the assumptions. We no longer assume iid Gaussian. We do not even assume independence or that the errors are identically distributed.

A famous result, the Gauss-Markov Theorem, states that our linear least squares regression estimate $\hat{\beta}$ is the best linear unbiased estimator. Note that this is a statement about the variance of $\hat{\beta}$ and not of the squared error of the predicted values \hat{y}_n .

9.3 Implications

As frameworks for estimating β we have observed the equivalences:

- Gaussian maximum likelihood framework \iff Least squares cost
- Least squares cost \iff Geometric perspective

This means the Gaussian ML framework \iff Geometric perspective.

The different perspective bring different idea to the table though.

- Probabilistic viewpoints give methods for estimating uncertainty in estimates.
- Least squares regression makes sense even under weaker assumptions than what we commonly assume.
- Gauss-Markov under quasi-likelihood assumptions gives an optimality characterization.
- Idea: what if we changed the cost function? What if we consider $|y_n - \beta^T x_n|$ instead of $(y_n - \beta^T x_n)^2$ (getting into robust regression). What if we add a penalty on β to obtain a regularized estimate of β ? eg. consider $J(\beta) + \lambda \|\beta\|^2$ (getting into ridge regression, LASSO)
- What happens if we consider another error distributions for ϵ_n ?

10 Applications (skip)

Moving onto what you can do with linear regression...

The assumption of linearity looks very restrictive. (Draw some example plots, strictly convex function, periodic)

Instead of using the x 's directly, we can use functions of x .

$$Y_n = \beta^T \phi(x_n) + \epsilon_n$$

examples: polynomials/interaction terms, spline regression, basis functions

This still gives us a problem that is linear in β and all the methods discussed still apply.

11 Weighted least squares (skip)

Suppose we have “important” observations and “unimportant” ones. We may think of giving higher weight to the residuals from the important observations. We do this by adding weights to the cost function

$$J(\beta) = \frac{1}{2} \sum_n w_n (y_n - \beta^T x_n)^2 = \frac{1}{2} (y - X\beta)^T W (y - X\beta)$$

Solving for the minimum cost gives normal equations for weighted least squares.

$$X^T W X \hat{\beta} = X^T W y$$

Another way of viewing weighted least squares is to drop the assumption that the errors have equal variance. Under Gaussianity assumptions, you get that the MLE maximizes

$$\sum_n \frac{1}{\sigma_n^2} (y_n - \beta^T x_n)^2$$

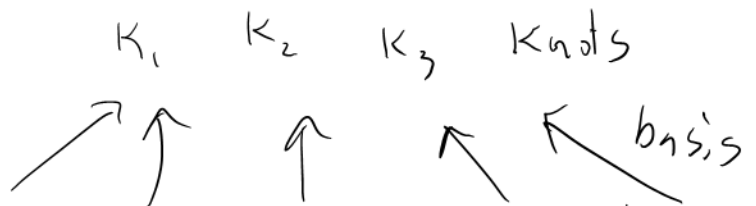
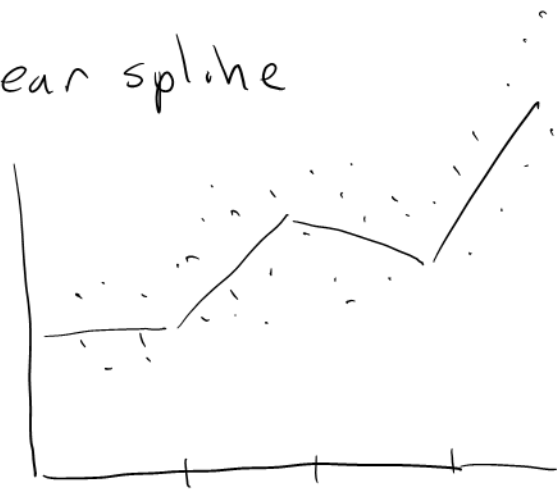
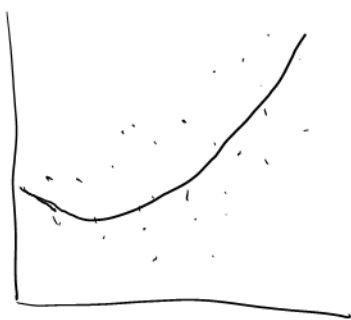
We will see this later when we do generalized linear models (GLMs)

univariate

Examples of basis function regression

polynomial

linear spline



linear basis