MATH 167: APPLIED LINEAR ALGEBRA

Jesús De Loera, UC Davis

January 5, 2012

LINEAR ALGEBRA IS VERY USEFUL

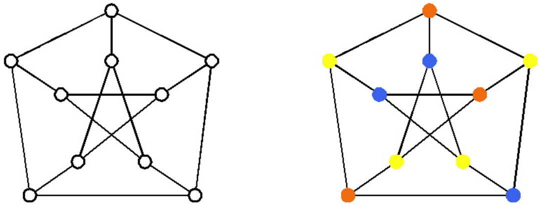
In this course we will try to highlight and discuss applications in three main areas:

- Networks and Graphs
  1. Electrical/Mechanical/Transportation networks
  2. World Wide Web Searching.
- Data Analysis
  1. Least Squares and Interpolation
  2. Vector Recognition and Machine Learning.
- Information Processing
  1. Error-correcting codes.
  2. Data compression and Noise removal.

In blue appear the possible final projects.

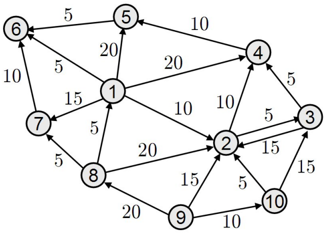
Networks and Graphs.

- A **graph** consists of nodes (or vertices) and edges (or connections). A typical example is the street network where the edges are the streets and the nodes are the intersections and of course, the internet gives some of the best examples of graphs! The facebook graph (nodes are people edges represent friends) or the wikipedia graph (nodes are concepts and edges represent relations via links).



- Graphs and networks are great tools in mathematical research, electrical engineering, computer programming and networking, business administration, sociology, economics, marketing,

- If a graph has numeric values on its edges or vertices is called a **network**. A graph is called directed or a **digraph** if its edges are directed (that means they have a specific direction). oriented edges are called **arcs** or **arrows** Often the edges of networks are directed.



- A **walk** joining two vertices X and Y of a graph is alternating sequence of incident nodes and edges (or arcs). A **path** is a walk without repeated vertices. A walk starting and ending at vertex v is called a **loop** . A loop without repeated nodes is a **cycle** or **circuit**. A graph G is **connected** if there is a path connecting any two vertices. Else G is **disconnected.**

- Matrices are a useful tool for studying graphs, since they turn the picture into numbers, and then one can use techniques from linear algebra.
  Given a graph G with n vertices $v_1, \ldots, v_n$, we define the **adjacency matrix** of G with respect to the labeling $v_1, \ldots, v_n$ of the vertices as being the $n \times n$ matrix $A_G = (a_{ij})$ whose entry $a_{ij} = 1$ if there is an edge between vertex v_i and v_j and zero otherwise.
- Note that for an undirected graph, the adjacency matrix is symmetric (that is it is equal to its transpose), but it is not necessarily the case for a digraph. Any square matrix with all entries 0 or 1 and 0's on the main diagonal determines a unique digraph. Here is a useful application (EXERCISE):
  Theorem: If A_G is the adjacency matrix of a graph G (with vertices $v_1, \ldots v_n$), the (i, j)-entry of $(A_G)^r$ represents the number of distinct walks from vertex v_i to vertex v_j in the graph of length r.

- There are important questions one can ask about a graph. Suppose we have a large graph G and we wish to cut off a piece of the vertex set by removing as few edges as possible (e.g., you want to attack an enemy's communication network).
- Let us call a partition of the vertex set V into two subsets A and $V \setminus A$ a **cut**. Denote by $E(A, V \setminus A)$ the edges going from one side to the other. We want to minimize the price to pay given by

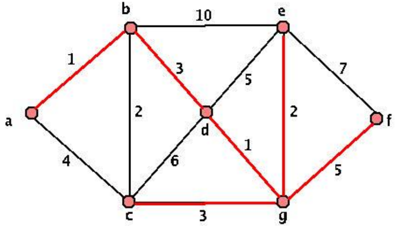
$$p(A, V \setminus A) = \frac{|V||E(A, V \setminus A)|}{|A||V \setminus A|}.$$

  Denote by p_G the value attained at the **sparsest cut.**
- To solve this problem one uses the **Laplace Matrix**. Consider the **degree matrix** D_G of the graph G which is the diagonal $n \times n$ matrix whose i-th diagonal entry is the number of edges incident on node i. The Laplace matrix L_G of the graph G is a defined as $D_G - A_G$, where A_G is the adjacency matrix of the graph.

- The Laplace matrix L_G is very important and it has some nice properties. It is symmetric and it is positive semidefinite. This means that $x^T L_G x$ is non-negative for all real vectors x. So all eigenvalues of L_G are non-negative and real (WHY? EXERCISE!)
- It is interesting to note that the vector $\mathbf{1} = (1, 1, 1, 1, \ldots, 1)$ is is an **eigenvector** with eigenvalue zero! (CHECK). Now the amazing thing is that the second eigenvalue says a lot about finding a sparse cut:

  Theorem If μ is the second smallest eigenvalue and u is an eigenvector, then one can easily find a cut of price at most $4\sqrt{d_{max}\mu}$, where d_{max} is the maximum degree of the graph. Moreover the best price $p_G \geq \mu$.
- We are going to learn this algorithm and why it works!

- Another fascinating application is to count the number of **Spanning trees** of a graph: A spanning tree is a subgraph that is connected but has no cycles. Important for communications and planning.



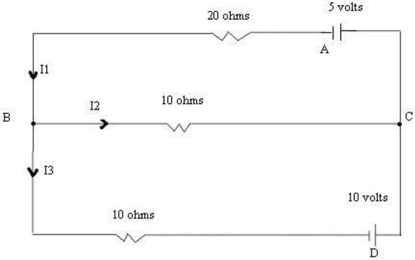
- How many different spanning trees are there? This is answered by the following cool theorem
  Theorem: Given a graph with n nodes and L_G the Laplace matrix of G then if we denote by L^- the $(n-1) \times (n-1)$ matrix obtained by deleting the last row and the last column of L_G, then the number of spanning trees of G equals $det(L^-)$.

- Another matrix for graphs is the node-edge **incidence matrix** B_G. Its columns are labeled by the edges or arcs on the graph and the rows correspond to the nodes. The entry $a_{ik} = 1$ if the k-th edge joins node v_i to another node. When using arcs, the value is $+1$ when v_i is at the head of the arrow and -1 when it is the tail of the arrow. For example, the complete graph K_5 has:

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

- Often, network problems can be modeled by a systems of linear equation coming from B_G. E.g., **Electrical Networks** we have three laws:
  1. **Ohm 's Law:** The voltage drop across a resistor is the product of the current and the resistance: V=IR
  2. **Kirchhoff's first Law:** The sum of the currents flowing into a node is equal to the sum of the current flowing out.
  3. **Kirchhoff's second Law:** The sum of the voltage drops around a closed loop is equal to the total voltage in the loop.

- **Example** In the network below



- Applying Kirchhoff's first Law to either of the nodes B or C, we find $I_1 = I_2 + I_3$ or $I_1 - I_2 - I_3 = 0$.
- Applying Kirchhoff's second Law to the loops $BDCB$ and $BCAB$, we obtain $-10I_1 + 10I_2 = 10$ and $20I_1 + 10I_2 = 5$.
- This gives a linear system of three equations: $I_1 - I_2 - I_3 = 0$, $-10I_1 + 10I_2 = 10$ and $20I_1 + 10I_2 = 5$ whose solution gives the current in each channel of the system. This is easy to solve, but in general?? **HOW SOLVE LARGE SYSTEMS**

Information Processing Error-Correction

- Transmitted messages, like data from a DVD, are always subject to noise. It is important to be able to encode a message in such a way that after noise scrambles it, it can be decoded back to its original form.
- This is done sometimes by repeating the message two or three times, something very common in human speech. However, copying data stored on a compact disk, or a floppy disk once or twice requires extra space to store.
- Messages are sent as sequences of 0's and 1's, such as 10101 or 1010011. Assume we want to send the message 1011. This **binary word** may stand for a real word, such as COOL, or a sentence such as MATH IS COOL.
- To encode 1011 we could attach a binary tail, so if the message gets distorted to, say, 0011, we can detect the error. Such tail could be a 1 or 0, depending on whether we have an odd or an even number of 1's in the word.
- This way all encoded words will have an even number of 1's. So 1011 will be encoded as 10111. If this is distorted to 00111 we know that an error has occurred (and two digits?).

- In the 1950's, R.H. Hamming introduced an interesting single error-correcting code that became known as the **Hamming code.** The key of the method is to use linear algebra not over the real numbers but over a **finite field** of two elements Z_2.
- $Z_2 = 0, 1$ and has operations $0 + 1 = 1, 0 + 0 = 0, 1 + 1 = 0$, $0 \cdot 0 = 0, 1 \cdot 0 = 0, 1 \cdot 1 = 1$. Using the field Z_2 we can construct the **vector space** $(Z_2)^n$. A big difference between the real vector space $\mathbb{R}^n$ and $(Z_2)^n$ is that the latter is finite, it has only 2^n possible vectors!
- Given two integers $k \leq n$ a subspace of $(Z_2)^n$. of dimension k is called an (n, k)-**linear code**. The elements of the linear code are the **encoded words**. Consider for example, the matrix H over Z_2 whose columns are the non-zero vectors of $(Z_2)^3$:

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

- The nullspace of H, $N(H)$ is called a **Hamming code**. In this case this is a $(7,4)$ code (WHY?). We say that H is a **parity check matrix** for the code $N(H)$. Using Gaussian elimination we get the a basis for the code
  $B =$
  $\{(1,0,0,0,0,1,1),(0,1,0,0,1,0,1),(0,0,1,0,1,1,0),(0,0,0,1,1,1,$
- If e_i is the standard vector in $(Z_2)^7$ it is not a member of $N(H)$, thus if $v \in N(H)$ then $v + e_i$ is not in $N(H)$. Also if $Hv = c_i$ then note $v + e_i \in N(H)$. and $v + e_j$ is not in $N(H)$ for $j \neq i$.
- The matrix G whose rows are the elements of the basis B is called the **generator matrix** of the Hamming (7,4)-code. Using these matrices and vector spaces we have an **algorithm for error-correction**.

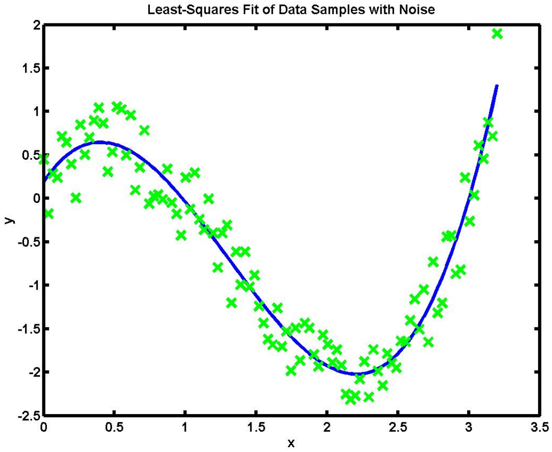
Say we want to send a word u consisting of four binary digits $u_1 u_2 u_3 u_4$. Assume the encoded word might get distorted by noise changing **no more than one of its components**. Let w be the received word.

- To encode u, form the linear combination v of the elements of the basis B above with the four digits of u as coefficients. Note that $v = [u_1 u_2 u_3 u_4]G$, where G is the generator matrix. By construction, the vector v is in $N(H)$. Note also that v would give a seven digit vector whose first four digits represent the original word.
- Compute Hw, where H is the matrix above.
- If $Hw = 0$, then w is in $N(H)$. A single error would mean w is not in $N(H)$ by the first part. We conclude that there is no distortion, and u is the first four digits of w.
- If $Hw = c_i$ for some i, then $v + e_i$ is a vector of $N(H)$, and $v + e_j$is not in $N(H)$ for all $j \neq i$. Thus changing the i-th component of w (from 0 to 1 or from 1 to 0) and get a new vector w'. The first four digits of w' represent the original word u.

- Suppose we received $w = 1100011$. The procedure gives Hw is $(0,1,0)^T$. Since Hw is equal to the second column of H, changing the second component of w give the word originally encoded. We conclude that the original message was 1000.
- Suppose we received $w = 0101010$. Since in this case $Hw = 0$, there was no error of transmission, thus the original word was 0101.
- The words we sent are small but in real life the words have many more digits. There are many other kinds of codes that have better performance.
- LINEAR ALGEBRA IS VERY POWERFUL AND USEFUL....

Regression
Least Squares

One of the important topics we will discuss is the so called method of regression or curve fitting. The process tries to find equations of approximating curves to a set of raw data. We desire to have a curve with minimal deviation from all data points.



- Suppose that the data points are $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where x is the independent variable and y is the dependent variable the **fitting curve** $f(x)$ has a **deviation error** for each point of $d_i = y_i - f(x_i)$. We are looking for the curve $f(x)$ that minimizes

$$d_1^2 + d_2^2 + d_3^2 + \cdots + d_n^2$$

- The approximation by a polynomial $f(x)$ will depend on the degree of the polynomial but as we will see this is done via system of linear conditions. This is related to minimizing quadratic constraints over the points inside a linear space.
- We will learn how and why this method works so well. Invented by Legendre Gauss back in the 1800's