Math 7H | Professor: Padraic Bartlett

Lecture 3: Discrete Dynamical Systems and the Internet

Week 3 | *UCSB 2015*

1 Discrete Dynamical Systems

In many branches of mathematics, we typically consider individual objects — the set of all prime numbers, the functions $\sin(x)$ and $\cos(x)$, a trefoil knot, the unit sphere in $\mathbb{R}^3$ — as if they are **fixed**, i.e. static or unchanging. This is often a useful abstraction to make, in that these objects are often difficult enough on their own; there is lots of interesting mathematics that we can do with a fixed knot in space without having to worry about it moving around! So we usually assume that these objects are unchanging, and study them with this assumption.

This assumption, however, is not a great way to model the world around us. Pretty much every object we encounter is constantly changing; materials degrade and reinforce themselves, sea levels rise and shrink, populations wax and wane. Accordingly, you would want to model this idea of "change" mathematically! The field of dynamical systems, roughly speaking, is the branch of mathematics that studies such changing objects.

Formally, we can define a **dynamical system**[1] as follows:

- An object X; typically something that can have a number of different states.
- An update map f, that when applied to X outputs a "new" set of states for X. We think of applying f to X as a way to advance one time-step, for whatever notion of time-step that we're simulating; that is, in some senses we would think of $f(X)$ as denoting what X will be after one second, or (for different maps f, X) one hour, or one year.

As with many objects in mathematics, this is best understood with an example.

Question. Consider the problem of modeling a population of rabbits and wolves on an island. Suppose that we start with 4000 rabbits and 1000 wolves. Moreover, suppose that if we have R rabbits and W wolves at month t, the number of rabbits and wolves at month $t+1$ can be written as

$$R_{\text{new}} = 2R_{\text{old}} - 1W_{\text{old}}, \qquad W_{\text{new}} = R_{\text{old}} + \frac{1}{2}W_{\text{old}}.$$

What is the long-term behavior for our populations of rabbits and wolves?

Answer. To answer this problem quickly, we use matrix[2] notation! Specifically, notice that we can describe our update rules as the following:

[1] Formally speaking, we think of this as a discrete dynamical system, because we have "discretized" time here: i.e. we have a sense in which we are advancing from a time t to a time $t+1$ all at once, as opposed to smoothly increasing t.

[2]

$$\begin{bmatrix} 2 & -1 \\ 1 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} R_{\text{old}} \\ W_{\text{old}} \end{bmatrix} = \begin{bmatrix} 2R_{\text{old}} - 1W_{\text{old}} \\ R_{\text{old}} + \frac{1}{2}W_{\text{old}} \end{bmatrix} = \begin{bmatrix} R_{\text{new}} \\ W_{\text{new}} \end{bmatrix}.$$

So, if we want to simulate our rabbit/wolf population, we can simply use this matrix to repeatedly update our population states! We do this here:

$$\begin{bmatrix} 2 & -1 \\ 1 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} 4000 \\ 1000 \end{bmatrix} = \begin{bmatrix} 7000, 4500 \end{bmatrix}$$
$$\begin{bmatrix} 2 & -1 \\ 1 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} 7000, 4500 \end{bmatrix} = \begin{bmatrix} 9500, 9250 \end{bmatrix}$$
$$\begin{bmatrix} 2 & -1 \\ 1 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} 9500, 9250 \end{bmatrix} = \begin{bmatrix} 9750, 14125 \end{bmatrix}$$
$$\begin{bmatrix} 2 & -1 \\ 1 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} 9750, 14125 \end{bmatrix} = \begin{bmatrix} 5375, 16812.5 \end{bmatrix}$$
$$\begin{bmatrix} 2 & -1 \\ 1 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} 5375, 16812.5 \end{bmatrix} = \begin{bmatrix} -6062.5, 13781.3 \end{bmatrix}.$$

In other words, sometime between months 4 and 5 our rabbit population cratered to 0 (after which, presumably, our model is no longer accurate[3].)

Definition. A $n \times n$ **matrix** A, for the purposes of this lecture, is a particular kind of function from $\mathbb{R}^n \to \mathbb{R}^n$. A matrix is specified by giving a $n \times n$ array of elements from $\mathbb{R}$, drawn as follows:

$$A := \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$$

With this array of elements specified, we can define the function $A : \mathbb{R}^n \to \mathbb{R}^n$ as follows: for any vector $\vec{v} = (v_1, v_2, \dots v_n) \in \mathbb{R}^n$, write

$$A(\vec{v}) = \left(\sum_{i=1}^{n} a_{1i} v_i, \ \sum_{i=1}^{n} a_{2i} v_i, \ \sum_{i=1}^{n} a_3 v_i, \ \dots \ , \sum_{i=1}^{n} a_{ni} v_i \right).$$

For example, the following object

$$A = \begin{bmatrix} \pi & \pi & 0 \\ 0 & 2 & 1 \\ 1 & 3 & -21 \end{bmatrix}$$

can be thought of as a 3×3 matrix over $\mathbb{R}$, and thus is a function from $\mathbb{R}^3 \to \mathbb{R}^3$. If we wanted to know where A sends the vector $(1, 2, 1)$, we could just use the definition above to calculate

$$\begin{aligned} A((1,2,1)) &= \left(\sum_{i=1}^{3} a_{1i} v_i, \ \sum_{i=1}^{3} a_{2i} v_i, \ \sum_{i=1}^{3} a_3 v_i \right) \\ &= (\pi \cdot 1 + \pi \cdot 2 + 0 \cdot 1, 0 \cdot 1 + 2 \cdot 2 + 1 \cdot 1, 1 \cdot 1 + 3 \cdot 2 + (-21) \cdot 1) \\ &= (3\pi, 5, -14). \end{aligned}$$

[3] Or there are antimatter rabbits.

This is a simple example of a dynamical system. Additionally, we've illustrated one of the key questions that we like to study for any given dynamical system: given a starting state along with an update rule, what does our dynamical system eventually go towards? In this case, the rabbit population cratered (with presumably the wolf population cratering shortly thereafter;) in general, you can see how you would simulate out such a system with starting state equal to some vector $\vec{v}$ and update rule given by a matrix A by just calculating $A^n \cdot \vec{v}$ for increasingly large values of n.

This is not the only kind of problem we like to study with dynamical systems, though! Consider the second sort of problem:

Question. On a second island, we want to study the long-term behavior of wolves and moose over time. As before, we have a set update rule: if we have M moose and W wolves at month t, the number of moose and wolves at month $t+1$ can be written as

$$M_{\text{new}} = 4M_{\text{old}} - 1W_{\text{old}}, \qquad W_{\text{new}} = 6M_{\text{old}} - W_{\text{old}},$$

and we can denote this update map with a matrix $A = \begin{bmatrix} 4 & -1 \\ 6 & -1 \end{bmatrix}$. This time, however, we're not trying to measure the long-term trajectory of some predetermined population. Instead, we want to solve a different problem: suppose that we get to seed the island with our own preselected population (M, W) of moose and wolves. Can we pick M, W so that our population is **stable**: in other words, such that $A(M, W) = (M, W)$?

Answer. We can do this! This is not too hard to do. We simply want a solution to the equation

$$\begin{bmatrix} 4 & -1 \\ 6 & -1 \end{bmatrix} \cdot \begin{bmatrix} M \\ W \end{bmatrix} = \begin{bmatrix} M \\ W \end{bmatrix};$$

in other words, by expanding the LHS, we can see that we want M, W such that

$$\begin{bmatrix} 4M - W \\ 6M - W \end{bmatrix} = \begin{bmatrix} M \\ W \end{bmatrix}.$$

In other words, we want $4M - W = M$ and $6M - W = W$; i.e. in both cases we want the relation $3M = W$. So any population that has this proportion (three wolves for each moose) is a stable population!

The above sort of problem — where we have a matrix A and want to find a vector $\vec{v}$ such that $A\vec{v} = \vec{v}$ — is one of the (many) reason we care about **eigenvectors** and **eigenvalues**! If you haven't seen them before, here's a definition:

Definition. Let A be a $n \times n$ matrix over $\mathbb{R}$. We say that $\vec{v} \in F^n$ is an **eigenvector** and $\lambda \in F$ is an **eigenvalue** if

$$A\vec{v} = \lambda\vec{v}.$$

We usually ask that $\vec{v} \neq \vec{0}$, to avoid silly trivial cases.

So, in the example above, a **stable state** was precisely an eigenvector with eigenvalue 1! Studying eigenvectors and eigenvalues is something you'll do a lot more of in Math 4a/108a-b; instead, I want to talk about one more problem that relates the dynamical systems concepts we've talked about to an area of modern research!

2 The Googles

Basically, before Google came along, web search engines were **atrocious**; many search results were not very-sophisticated massive keyword-bashes plus some well-meaning but dumb attempts to improve these results by hand. In the 90's, Brin and Page (the two founders of Google) came onto the scene, with the following simple idea:

Important websites are the websites other important websites link to.

This seems kinda circular, so let's try framing this in more of a graph-theoretic framework: Take the internet. Think of it as a collection of webpages, which we'll think of as "vertices," along with a collection of hyperlinks, which we'll think of as directed lines[4] going between webpages. Call these webpages $\{v_1, \dots v_n\}$ for shorthand, and denote the collection of webpages linking to some v_i as the set $\text{LinksTo}(v_i)$.

In this sense, if we have some quantity of "importance" $rank(v_i)$ that we're associating to each webpage i, we still want it to obey the entire "important websites are the websites other important websites link to" idea. However, we can refine what we mean by this a little bit. For example, suppose that we know a website is linked to by Google. On one hand, this might seem important — Google is an important website, after all! — but on the other hand, this isn't actually that relevant, because Google basically links to **everything.** So we don't want to simply say something is important if it's linked to by something important — we want to **weight** that importance by how many **other** things that important website links to! In other words, if you're somehow important and also only link to a few things, we want to take those links very seriously — i.e. if something is linked to by the front page of the New York Times or the Guardian, that's probably pretty important!

If we write this down with symbols and formulae, we get the following equation:

$$\text{rank}(v_i) = \sum_{v_j \in \text{LinksTo}(v_i)} \frac{\text{rank}(v_j)}{\text{number of links leaving}(v_j)}.$$

In other words, to find your rank, we add up all of the ranks of the webpages that link to you, scaling each of those links by the number of other links leaving those webpages. This is ...still circular. But it looks mathier! Also, it's more promising from a linear-algebra point of view. Suppose that we don't think of each ranking individually, but rather take them all together as some large rank vector $\vec{r} = (\text{rank}(v_1), \dots \text{rank}(v_n))$.

As well, instead of thinking of the links one-by-one, consider the following $n \times n$ "link-matrix" A, formed by doing the following:

- If there is a link to v_i from v_j, put a $\frac{1}{\text{number of links leaving}(v_j)}$ in the entry (i, j).
- Otherwise, put a 0.

This contains all of the information about the internet's links, in one handy $n \times n$ matrix!

[4] A "series of tubes," if you will.

Now, notice that if we multiply this matrix A by our rank vector $\vec{r}$, we get

$$A \cdot \vec{r} = \begin{bmatrix} \sum_{v_j \in \text{LinksTo}(v_1)} \frac{\text{rank}(v_j)}{\text{number of links leaving}(v_j)} \\ \sum_{v_j \in \text{LinksTo}(v_2)} \frac{\text{rank}(v_j)}{\text{number of links leaving}(v_j)} \\ \vdots \\ \sum_{v_j \in \text{LinksTo}(v_n)} \frac{\text{rank}(v_j)}{\text{number of links leaving}(v_j)} \end{bmatrix}.$$

In other words, if we the "mathy" version of the importance rule we derived earlier, we have

$$A \cdot \vec{r} = \vec{r}.$$

In other words, the vector $\vec{r}$ that we're looking for is an **eigenvector** for A, corresponding to the eigenvalue 1! The entries in this eigenvector then correspond to the "importance" ranks we were looking for. In particular, the coordinate in the vector $\vec{r}$ with the highest value corresponds to the "most important" website, and should be the first page suggested by the search engine.

Up to tweaks and small modifications, this is precisely how search works nowadays; people come up with quick and efficient ways to find eigenvectors for subgraphs of the internet that correspond to the eigenvalue 1. (Actually finding this eigenvector in an efficient manner is a problem people are still working on; much of modern search is a study for "better/faster" ways to find this eigenvector!)