pset8

September 7, 2017

1 18.06 pset 8

Due Wednesday, **April 19**.

1.1 Problem 1: Google PageRank

Consider the following network from pset 5:

Suppose that nodes represent 6 web pages, and the edges represent links (e.g. page 2 links to page 6 via edge 7).

Imagine a random web surfer who

- Starts on page 1 (node 1).
- With probability $p = 0.85$ (85%), she picks a link at random from the current node (with an equal chance for each edge starting at the current node) and follows it.
- With probability $1 - p = 0.15$ (15%), she jumps to a random web page (1 to 6) with equal probability, regardless of what is linked to from the current page.

For example, if she is at page 4, she will next visit page 1 with probability $(1 - p)/6 = 0.025$ (2.5%), page 2 with probability 0.025, page 3 with probability 0.025, page 4 with probability 0.025, page 5 with probability $p/2 + (1 - p)/6 = 0.45$ (45%), and page 6 with probability 0.45.

1.1.1 (a)

Express these transition probabilities in the form of a Markov matrix M. That is, the j-th column of M should be the probabilities of visiting each node starting from j.

Enter your matrix `M` in Julia. (Make sure to check that the sum of each column is 1 via `sum(M, 1)` as in class.)

(The adjacency matrix for this graph, from pset 5, might be helpful.)

1.1.2 (b)

Averaging over a long (infinite) time, what fraction of her visits go to each of the 6 nodes?

(You can use the Julia `eig` function to get the eigenvalues and eigenvectors of M, as in class. Be careful with your normalization.)

This fraction is the Google PageRank and is the basis for how Google ranks pages in its search results. Which is the "most important" (highest-ranking) page in your graph by this measure?

1.1.3 (c)

In practice, the Markov matrix represent the real web is too huge for Google to exactly compute the steady-state eigenvector. Instead, it starts with a random vector and just multiplies by M a few times to get an approximation for the steady state.

If you start with the "random" vector `x = [0.28,0.2,0.01,0.21,0.19,0.11]` (whose components sum to 1), how many times do you need to multiply it by M to get the correct pagerank to two decimal places for all 6 pages? (You can use `round.(x,2)` to round a vector to two decimal places in Julia.)

1.2 Problem 2: Ladders and Chutes

In class, we analyzed the expected number of moves to finish a 1-player "Chutes and Ladders" game. For variety, my daughter likes to change the rules sometimes. One variant that she likes is "Ladders and Chutes", in which the ladders and chutes are *reversed*: the ladders go down, and the chutes go up. How does this change the expected number of moves to finish?

Correction: Chutes & Ladders has a ladder from position 80 to position 100. If this becomes a chute, then the game would never finish. In the "Ladders and Chutes" game, simply delete this ladder.

(There is a simple way to change the transition matrix T — which is actually a form of adjacency matrix similar to pset 5 — from the lecture notes, in order to reverse the direction of the ladders/chutes, at least for the off-diagonal elements of T. However you do it, only trivial changes to the code from the lecture notes should be required.)

1.3 Problem 3

In class, we discussed the power method: starting with a random vector x and repeatedly computing $x \leftarrow Ax/\|Ax\|$, which converges to the eigenvector with the largest $|\lambda|$ (assuming that there is a single such eigenvector). Given an estimate x for the eigenvector, the Rayleigh quotient $x^T Ax/\|x\|^2 =$ `dot(x,A*x)/dot(x,x)` is an estimate for the eigenvalue.

For example, if we apply this process 100 times to the 5×5 matrix R given by

```
In [1]: R = [ 3.15042    -0.383941   0.469052  -1.92221   -2.0107
               0.299402    6.63455   -1.31991    1.20677   -1.34234
              -0.391874    0.36382    3.36915   -0.384912   0.821678
              -0.53268    -1.93047    1.25815    1.37172    0.888551
              -0.0655503   2.69435   -1.12379    0.874781   0.474167 ]
```

```
Out[1]: 5×5 Array{Float64,2}:
          3.15042    -0.383941   0.469052  -1.92221   -2.0107
          0.299402    6.63455   -1.31991    1.20677   -1.34234
         -0.391874    0.36382    3.36915   -0.384912   0.821678
         -0.53268    -1.93047    1.25815    1.37172    0.888551
         -0.0655503   2.69435   -1.12379    0.874781   0.474167
```

then we get

```
In [2]: x = randn(5)
        for i = 1:100
            x = R*x / norm(x)
        end
        dot(x,R*x)/dot(x,x)
```

```
Out[2]: 4.999998748076965
```

i.e. the estimated largest-magnitude eigenvalue is 5, which matches the "exact" largest-magnitude eigenvalue computed by `eigvals`:

```
In [3]: eigvals(R)
```

```
Out[3]: 5-element Array{Float64,1}:
         5.0
         4.0
         1.0
         3.00001
         2.0
```

Question: Suppose that instead we wanted the *smallest*-magnitude eigenvalue (minimum $|\lambda|$) and the associated eigenvector. How could we modify the power iteration above to give us this quantity? Try it out (modifying the code above) and make sure it works.

1.4 Problem 4

(From Strang, problem 6.3.)

A door is opened between rooms that hold $v(0) = 30$ people and $w(0) = 10$ people. The movement between the rooms is proportional to the difference $v - w$:

$\frac{dv}{dt} = w - v$

$\frac{dw}{dt} = v - w$

(a) Write the problem in matrix form $du/dt = Au$ for $u = (v, w)$. **(b)** Show that the total $v + w = 40$ is constant over time. **(c)** What are v and w at $t = 1$ and $t = \infty$?

1.5 Problem 5

1.5.1 (a)

Suppose $dx/dt = Ax$. Let y be a vector in the *left nullspace* of A. Show that $y^T x$ is constant over time (a "conserved quantity").

1.5.2 (b)

If M is a Markov matrix, explain why this means that the sum of the vector x components (`sum(x)` in Julia) is conserved for $dx/dt = (M - I)x$.

1.5.3 (c)

If M is a Markov matrix with positive entries (recall what this means about the eigenvalues), what is the solution $dx/dt = (M - I)x$ as $t \to \infty$, in terms of the initial condition $x(0)$ and the steady-state eigenvector x_0 of M? Explain.

1.5.4 (d)

Check your answers to (b) and (c) by computing $e^{(M-I)t}x$ (`expm((M-I)*t))*x` in Julia) for your Markov matrix from problem 1 and a random starting vector `x = rand(6); x /= sum(x)` whose components sum to 1, for $t = 1$ and $t = 100$.

1.6 Problem 6

(From Strang, section 6.3, problem 9.)

The matrix $A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ has eigenvalues $\pm i$ and eigenvectors $(1, \pm i)$. Use these to solve $du/dt = Au$ for $u(0) = (4, 0)$, by expanding the initial condition in terms of the eigenvectors. Use the Euler expansion $e^{\pm it} = \cos t \pm i \sin t$ to write your solution $u(t)$ in a form that is clearly real-valued (no i factors).

1.7 Problem 7

(From Strang, section 6.3, problem 26.) Give two reasons why the matrix exponential e^{At} is never singular. **(a)** Write down its inverse. **(b)** If $Ax = \lambda x$ then an eigenvalue of e^{At} is what, which is nonzero because why? Hence $N(e^{At}) = \{0\}$.