Homeworks

Please make sure I return assignment #2 and collect assignment #3 at the end of class!

I will try to grade homework #3 before Friday (the last day on which students can withdraw from a course with the grade of "W"), but I can't promise I'll be able to. If you are considering withdrawing, send me an email, and I'll make grading your assignment by Friday a higher priority.

I'll post assignment #4 on the web in the next couple of days. It'll be due on December 6.

Final projects

Deadlines, etc.

If you haven't already done this, by Wednesday email me a proposal for your project!

The project is supposed to entail only about 10 hours of work; don't choose a topic that's too involved!

Homework problems from the final homework assignment (to be posted in the next few days) are yet more examples of what a final project could be.

Don't forget your class presentation is as important as your written submission!

You can use my Mac (I also have a tablet PC that I could bring in if you prefer), but I suggest that you use your own laptop if you have one, and that you do a rehearsal ahead of time to avoid technical glitches.

Questions?

The project write-ups are due on **November 29** .

Since there are 10 people enrolled in the class, we'll need 5 people to present on November 29 and 5 more to present on December 6. **You all should be prepared to present your work on either day.** If for some reason you can't present on November 29, you'll need to contact me by email ahead of time.

More possible topics

Consider random walk on {0,1,2,3,...} with $p_{0,0} = 1$ and $p_{i,i-1} = p_{i,i+1} = p_{i,i+2} = \frac{1}{3}$ for all $i > 0$. (An upcoming homework asks you to show that the probability that a particle that starts at 1 will ever hit 0 is $p = \sqrt{2} - 1$.) Derandomize this walk so that when n particles are put through the system starting at 1, the number that hit 0 differs from np by no more than a constant. (See the "Walk on finite graph C" mode of the Canary-Wong applet.)

In a somewhat similar vein, see http://faculty.uml.edu/jpropp/584/ladders.html

for a picture of a typical ladder graph and the derivation of the governing equations.

```
Clear[p, q, r, s, t]

Solve[{p == (1 + q + q) / 3, q == (0 + r + s) / 3,
  r == (1 + q + t) / 3, q + r == 1, s == r q + q r, t == q q + r r}, {p, q, r, s, t}]
```

$$\left\{\left\{p \to -\frac{1}{\sqrt{3}}, s \to -3 - 2\sqrt{3}, t \to 2\left(2 + \sqrt{3}\right), r \to \frac{1}{2}\left(3 + \sqrt{3}\right), q \to \frac{1}{2}\left(-1 - \sqrt{3}\right)\right\},\right.$$
$$\left.\left\{p \to \frac{1}{\sqrt{3}}, s \to 2\sqrt{3} - 3, t \to 2\left(2 - \sqrt{3}\right), r \to \frac{1}{2}\left(3 - \sqrt{3}\right), q \to \frac{1}{2}\left(\sqrt{3} - 1\right)\right\}\right\}$$

Other geometries are possible (ladders built of triangles instead of squares, etc.); they all give nice quadratic irrationals. Rotor-walk on these graphs should be susceptible to analysis, just as in the case of the "Goldbugs" walk (although the analysis for ladder-graphs is likely to be more complicated).

Diffusion-driven processes

Random walks in $\mathbb{Z}^2$ can be used to "build things" in the plane.

For instance, suppose we have some initial random coloring of the cells of an n-by-n array, where the "colors" are the numbers 1, 2, and 3. We'll repeatedly use random walks that start in the corners to modify the coloring.

Call the northwest corner, the northeast corner, and

the southwest corner the 1-corner, 2-corner, and 3-corner respectively.

First we'll have color 1 steal a cell from one of the other two colors.

To do this, we'll put a walker on the 1-corner and have her do a random walk until she encounters a cell that isn't colored 1 (say its color is c_1); she changes its color to 1. Let c_1' be the color that's neither 1 nor c_1. Now put a walker on the c_1' corner and have her do a random walk until she encounters a cell that isn't colored c_1' (say its color is c_2); she changes its color to c_1'. Let c_2' be the color that's neither c_1' nor c_2. Now put a walker on the c_2' corner and have her do a random walk until she encounters a cell that isn't colored c_2' (say its color is c_3); she changes its color to c_2'. Let c_3' be the color that's neither c_2' nor c_3.

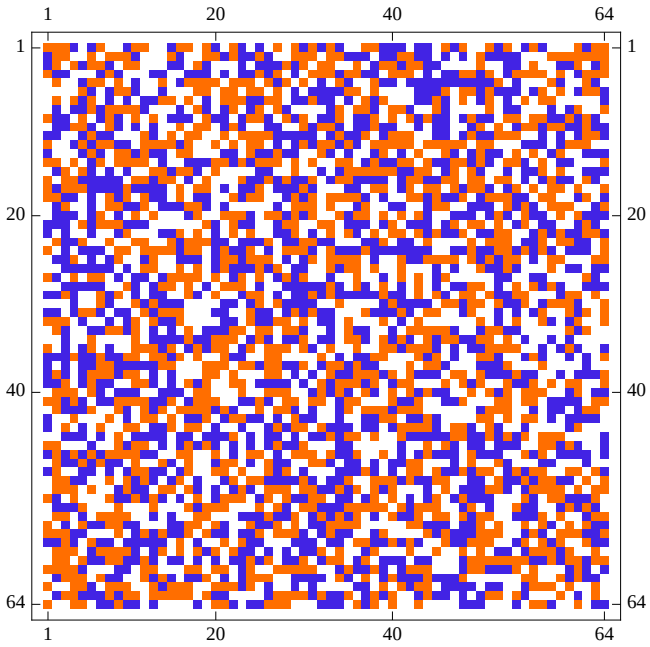
Et c.

What do you expect to happen?

```
n := 64

Board = Table[RandomInteger[{1, 3}], {n}, {n}];

MatrixPlot[Board - 2]
```

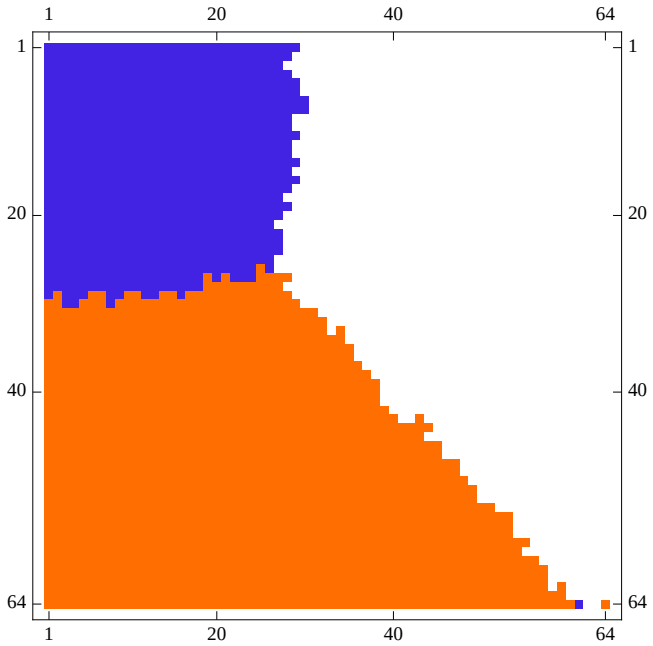



```
Revise[i_] := Module[{row, col, newrow, newcol, j}, {row, col} = {{1, 1}, {1, n}, {n, 1}}
    [[i]];
  While[Board[[row, col]] == i,
   {newrow, newcol} = {row, col} + {{1, 0}, {-1, 0}, {0, 1}, {0, -1}}
      [[RandomInteger[{1, 4}]]]; If[newrow ≥ 1 && newrow ≤ n && newcol ≥ 1 && newcol ≤ n,
    {row, col} = {newrow, newcol}]]; j = Board[[row, col]]; Board[[row, col]] = i; Return[j]]

Compete[n_] :=
 Module[{i = 1, j, m}, For[m = 1, m ≤ n, m++, j = Revise[i]; i = 6 - i - j]; Return[Board]]

Compete[10 000];
```

```
MatrixPlot[Board - 2]
```



Do the three regions stabilize, with interfaces whose fluctuations are small as a function of *n* (so that most cells of the grid are either nearly-certain-to-be-1's, nearly-certain-to-be-2's, or nearly-certain-to-be-3's?

Do the interfaces meet at 120 degree angles?

Ask me in about three years!

A simpler version of this process that has been studied is Internal Diffusion-Limited Aggregation, or Internal DLA.

In this stochastic model, we start with all of the cells of $\mathbb{Z}^2$ colored white. We repeatedly let a walker leave (0,0) until she encounters a white square; she turns the white square black and starts again from (0,0).

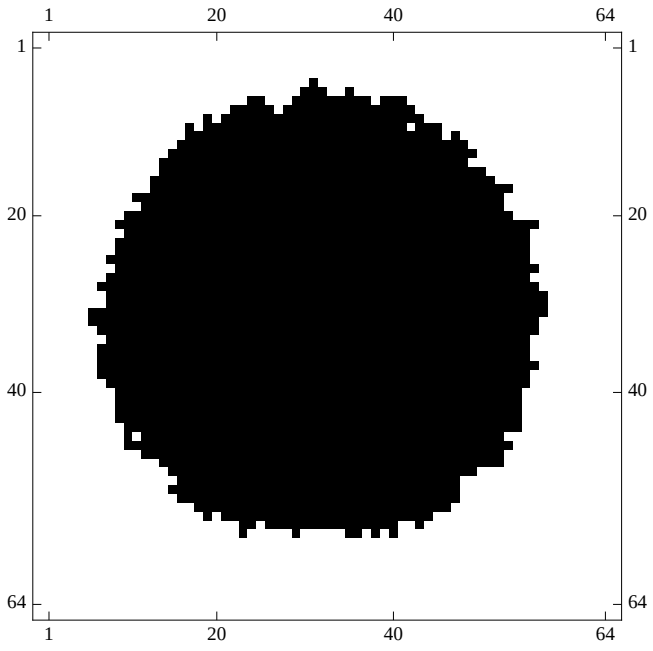
```
n := 64

IDLABoard = Table[0, {n}, {n}];

Grow[] := Module[{row, col, newrow, newcol, j}, {row, col} = {Floor[n / 2], Floor[n / 2]}; While[
   IDLABoard[[row, col]] == 1 , {newrow, newcol} = {row, col} + {{1, 0}, {-1, 0}, {0, 1}, {0, -1}}
      [[RandomInteger[{1, 4}]]]; If[newrow ≥ 1 && newcol ≤ n && newrow ≥ 1 && newcol ≤ n,
    {row, col} = {newrow, newcol}]]; IDLABoard[[row, col]] = 1]

Do[Grow[], {2000}]

MatrixPlot[IDLABoard, ColorFunction → "Monochrome"]
```



It's been proved by Jerison and Levine and Sheffield that as an IDLA blob gets bigger and bigger, it gets rounder and rounder.

If we use rotor-walk instead of random walk, we get even rounder blobs (as proved by Levine and Peres), and the coloring of the cells shows beautiful patterns that nobody understands yet. Run the 2-D Aggregation Mode of
http://www.cs.uml.edu/~jpropp/rotor-router-model/
to see what this looks like dynamically, and look at the pictures at http://rotor-router.mpi-inf.mpg.de/ to see what rotor-router aggregation blobs look like after they have grown to occupy millions or even billions of cells in the grid.

```
Clear[n]
```

Knowing when to stop

Total variation distance

We know that if $\mathbf{P}$ is the transition matrix of a regular Markov chain, then $\mathbf{P}^n \to \mathbf{W}$ as $n \to \infty$, where $\mathbf{W}$ is the square matrix each of whose rows is the unique probability vector $\mathbf{w}$ that satisfies $\mathbf{wP} = \mathbf{w}$ (the stationary probability distribution for the Markov chain).

How quickly?

Note that the i,j th entry of $\mathbf{P}^n$ is the probability, if you start in state i, that you're in state j after exactly n steps.

That is, the i th row of $\mathbf{P}^n$ gives the probability distribution that governs where you are after n steps, if you started in state i .

(Special case: If $\mathbf{u}$ is the row-vector with 1 in the i th position and with 0 everywhere else, then $\mathbf{uP}^n$ is the i th row of $\mathbf{P}^n$.)

So if we knew how quickly $\mathbf{P}^n$ goes to $\mathbf{W}$, we'd know how quickly $\mathbf{uP}^n$ goes to $\mathbf{w}$, and vice versa.

Example: Random walk on a path of length 3, with semi-reflecting endpoints.

```
SR3 = {{1 / 2, 1 / 2, 0}, {1 / 2, 0, 1 / 2}, {0, 1 / 2, 1 / 2}};
```

```
MatrixForm[SR3]
```

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

```
Eigenvalues[SR3]
```

$$\left\{1, -\frac{1}{2}, \frac{1}{2}\right\}$$

Since there are no repeated eigenvalues, the matrix is definitely diagonalizable.

```
EV = Eigenvectors[SR3]
```

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & 0 & 1 \end{pmatrix}$$

(Note that *Mathematica* has different cell-formats for expressions like this, such as StandardForm and TraditionalForm. You can convert between them with Cell → Convert To → ... Try it:

```
EV = Eigenvectors[SR3]
```

```
{{1, 1, 1}, {1, -2, 1}, {-1, 0, 1}}
```

We can make the change of format permanent with *Mathematica* → Preferences → Format type of new output cells.)

```
EVInverse = Inverse[%]
```

$$\begin{pmatrix} \frac{1}{3} & \frac{1}{6} & -\frac{1}{2} \\ \frac{1}{3} & -\frac{1}{3} & 0 \\ \frac{1}{3} & \frac{1}{6} & \frac{1}{2} \end{pmatrix}$$

```
Diag = EV.SR3.EVInverse
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} \end{pmatrix}$$

```
EVInverse.Diag.EV
```

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

```
MatrixPower[SR3, 5]
```

$$\begin{pmatrix} \frac{11}{32} & \frac{11}{32} & \frac{5}{16} \\ \frac{11}{32} & \frac{5}{16} & \frac{11}{32} \\ \frac{5}{16} & \frac{11}{32} & \frac{11}{32} \end{pmatrix}$$

```
MatrixPower[EVInverse.Diag.EV, 5]
```

$$\begin{pmatrix} \frac{11}{32} & \frac{11}{32} & \frac{5}{16} \\ \frac{11}{32} & \frac{5}{16} & \frac{11}{32} \\ \frac{5}{16} & \frac{11}{32} & \frac{11}{32} \end{pmatrix}$$

```
EVInverse.MatrixPower[Diag, 5].EV
```

$$\begin{pmatrix} \frac{11}{32} & \frac{11}{32} & \frac{5}{16} \\ \frac{11}{32} & \frac{5}{16} & \frac{11}{32} \\ \frac{5}{16} & \frac{11}{32} & \frac{11}{32} \end{pmatrix}$$

```
EVInverse.MatrixPower[Diag, n].EV
```

$$\begin{pmatrix} \frac{1}{3} + 2^{-n-1} + \frac{1}{3}(-1)^n 2^{-n-1} & \frac{1}{3} - \frac{1}{3}\left(-\frac{1}{2}\right)^n & \frac{1}{3} - 2^{-n-1} + \frac{1}{3}(-1)^n 2^{-n-1} \\ \frac{1}{3} - \frac{1}{3}\left(-\frac{1}{2}\right)^n & \frac{1}{3} + \frac{1}{3}(-1)^n 2^{1-n} & \frac{1}{3} - \frac{1}{3}\left(-\frac{1}{2}\right)^n \\ \frac{1}{3} - 2^{-n-1} + \frac{1}{3}(-1)^n 2^{-n-1} & \frac{1}{3} - \frac{1}{3}\left(-\frac{1}{2}\right)^n & \frac{1}{3} + 2^{-n-1} + \frac{1}{3}(-1)^n 2^{-n-1} \end{pmatrix}$$

```
MatrixPower[SR3, n]
```

$$\begin{pmatrix} \frac{1}{3}+2^{-n-1}+\frac{1}{3}(-1)^n 2^{-n-1} & \frac{1}{3}-\frac{1}{3}\left(-\frac{1}{2}\right)^n & \frac{1}{3}-2^{-n-1}+\frac{1}{3}(-1)^n 2^{-n-1} \\ \frac{1}{3}-\frac{1}{3}\left(-\frac{1}{2}\right)^n & \frac{1}{3}+\frac{1}{3}(-1)^n 2^{1-n} & \frac{1}{3}-\frac{1}{3}\left(-\frac{1}{2}\right)^n \\ \frac{1}{3}-2^{-n-1}+\frac{1}{3}(-1)^n 2^{-n-1} & \frac{1}{3}-\frac{1}{3}\left(-\frac{1}{2}\right)^n & \frac{1}{3}+2^{-n-1}+\frac{1}{3}(-1)^n 2^{-n-1} \end{pmatrix}$$

Note that *Mathematica* does at least one unhelpful thing: it has expressions like the

$$(-1)^n 2^{1-n}$$

in the middle entry that hide the fact that the relevant eigenvalue is -1/2.

Every entry of $\mathbf{P}^n$ is of the form

(*) $$\frac{1}{3} + A\left(\frac{1}{2}\right)^n + B\left(-\frac{1}{2}\right)^n$$

for constants A, B, so we can see that as $n \to \infty$, $\mathbf{P}^n$ converges to

$$\begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{pmatrix}$$

exponentially.

It is also easy to see that for any **u**, the three entries of $\mathbf{u}\mathbf{P}^n$ are all of the form (*), and hence converge to **w** exponentially fast.

It turns out that a good way to measure the distance between $\mathbf{u}\mathbf{P}^n$ and $\mathbf{w}$ is not the usual "ell-two" distance

$$d_2((a,b,c),(d,e,f)) = \sqrt{(a-d)^2 + (b-e)^2 + (c-f)^2}$$

but instead the "ell-one" distance

$$d_1((a,b,c),(d,e,f)) = |a-d| + |b-e| + |c-f|$$

or rather half of the ell-one distance, which is called the total variation distance between two probability distributions:

$$||(a,b,c)-(d,e,f)||_{TV} = \frac{1}{2}(|a-d| + |b-e| + |c-f|).$$

As you'll show on the homework, if π and π' are two

probability distributions on a finite set S, then the total variation distance

$\|\pi - \pi'\|_{TV} = \frac{1}{2} \sum_{s \text{ in } S} |\pi(s) - \pi'(s)|$

is also equal to the maximum of $|\pi(E) - \pi'(E)|$ over all subsets E of S; that is, it's the answer to the question "If we're computing the probability of an event, how much might it matter whether we use π or π'?"

Going back to our example, for any initial distribution **u**, each entry of the vector $\mathbf{u}\mathbf{P}^n - \mathbf{w}$ can be written in the form

$$A\left(\frac{1}{2}\right)^n + B\left(-\frac{1}{2}\right)^n$$

and hence is bounded by $(|A| + |B|)2^{-n}$. This implies that the total variation distance between $\mathbf{u}\mathbf{P}^n$ and **w** goes to 0 like $C2^{-n}$ for some C.

More generally, if we have a regular Markov chain on a finite state space with transition matrix **P** and stationary distribution **w**, and **u** is an arbitrary initial distribution, the total variation distance between $\mathbf{u}\mathbf{P}^n$ and **w** goes to 0 like r^n, where

$r = \max\{|\lambda|: \lambda \neq 1 \text{ is an eigenvalue of } \mathbf{P}\} < 1$.

If we take an n such that $r^n \approx \frac{1}{2}$, it may not look like we've made much progress towards stationarity, but if we take twice as many steps, we get $r^{2n} \approx \frac{1}{4}$, and if we take ten times as many steps, we get $r^{10n} \approx \frac{1}{1024}$, which means we're quite close to stationarity (unless our multipliers A,B,... are huge, which isn't likely, since all the numbers in sight are probabilities and hence between 0 and 1); so in some sense, the first n for which $r^n \leq \frac{1}{2}$ measures the time at which we first begin to measurably approach equilibrium, and if we take multiples of that n, we get quite close to equilibrium. But there's nothing special about the number $\frac{1}{2}$.

The mixing time for a regular Markov chain is often defined as the smallest n such that

$$|\lambda|^n \leq e^{-1} \text{ for all eigenvalues } \lambda \neq 1.$$

This notion of mixing time has some defects (it underestimates the amount of time it takes for the total variation distance to get really small), but it's good for certain sorts of qualitative predictions.

Example: Unbiased random walk on the vertices of an m-gon, or if you prefer, on $\mathbb{Z}/m\mathbb{Z}$.

Here's the matrix **P** for random walk on the 5-gon:

```
MatrixForm[{{0, 1/2, 0, 0, 1/2}, {1/2, 0, 1/2, 0, 0},
  {0, 1/2, 0, 1/2, 0}, {0, 0, 1/2, 0, 1/2}, {1/2, 0, 0, 1/2, 0}}]
```

$$\begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}$$

Claim: The eigenvalues are $(\zeta + \zeta^{-1})/2$ where ζ ranges over the m mth roots of 1 in $\mathbb{C}$. (These eigenvalues are real, since ζ and ζ^{-1} are complex conjugates; indeed, writing $\zeta = \exp(2\pi ij/m)$ with $0 \le j \le m$-1, we get $(\zeta + \zeta^{-1})/2 = \cos 2\pi j/m$.)

Proof: Check that the column vector **u** with components $\zeta^1, \zeta^2, \zeta^3, \ldots, \zeta^m$ is an eigenvector of the matrix **P** with eigenvalue $(\zeta + \zeta^{-1})/2$. Indeed, if the kth component of **u** is ζ^k, the kth component of **Pu** is $(1/2)\,\zeta^{k+1} + (1/2)\zeta^{k-1}$;

the eigenvector equation $\mathbf{Pu} = \lambda \mathbf{u}$ turns into

$$(1/2)\zeta^{k+1} + (1/2)\zeta^{k-1} = \lambda\, \zeta^{k} \text{ for all } k,$$

and dividing the equation by ζ^k gives just

$$(1/2)\zeta^{+1} + (1/2)\zeta^{-1} = \lambda,$$

so that $\lambda = (\zeta + \zeta^{-1})/2$.

Now let's compute the mixing time.

Take the case where m is odd (this rules out the annoying eigenvalue -1); write $m = 2k+1$. Then the eigenvalue $\lambda \neq 1$ for which $|\lambda|$ is largest is $\lambda = \cos 2\pi k/(2k+1) = \cos \pi - \frac{\pi}{2k+1} = -\cos \frac{\pi}{2k+1}$, with $|\lambda| = \cos \frac{\pi}{2k+1} \approx 1 - x^2/2 \approx \exp(-x^2/2)$ with $x = \frac{\pi}{2k+1}$. So the error between $\mathbf{uP}^n$ and $\mathbf{w}$ goes like $[\exp(-x^2/2)]^n = \exp(-nx^2/2)$, and starts to equilibrate when $nx^2/2 \approx 1$. Ignoring constants, we have $x^2 \approx 1/k^2$, and equilibration happens when $nx^2 \approx 1$, i.e., $n/k^2 \approx 1$, i.e., when $n \approx k^2 \approx m^2$.

That is, the time it takes for our distribution to begin to approach stationarity is about m^2, where m is the size of our system.

Note that $O(m^2)$ is also the time it typically takes for a particle doing random walk on the m-gon to first reach the other side. (Turn this into a one-dimensional random walk with a walker who starts in the middle and with two targets, both k steps away, located in opposite directions.)

It makes intuitive sense that the time it takes for the Markov chain to begin to equilibrate should be roughly the same order of magnitude as the time it takes before it's likely that the walker has gotten to the opposite side.

If our ergodic chain is not regular, then we have (real or complex) eigenvalues other than 1 that lie on the unit circle in the complex plane. In this case, $\mathbf{P}^n$ does not go to $\mathbf{W}$. However, we can introduce a "lazy" version of the chain that at each step tosses a fair coin

and stays put if the coin comes up heads and advances according to the transition matrix **P** is the coin comes up tails. The transition matrix for this lazy chain is **P'** = $\frac{1}{2}$(**I** + **P**), so its eigenvalues are of the form $\lambda' = \frac{1}{2}(1 + \lambda)$ where λ ranges over the eigenvalues of **P**. Since λ lies in the disk of radius 1 centered at 0 in the complex plane, λ' lies in the disk of radius $\frac{1}{2}$ centered at $\frac{1}{2}$, and if we omit the eigenvalue 1, we see that all the other eigenvalues lie strictly inside the unit disk. This is good news: it means that we get exponential convergence.

Alternatively, recall that if **P** is ergodic, $\frac{1}{2}$(**I** + **P**) is regular.

Stopping rules

We've seen that if you just run the chain for a preset number of steps, you get exponentially close to the stationary distribution (with the caveat that if one of the eigenvalues other than 1 has magnitude close to 1, then it could take a very long time for this exponential law to kick in).

What if you run the chain for a variable number of steps, in some adaptive fashion?

Then it's possible that you might hit the stationary measure on the nose.

Example 1: Lazy random walk on the vertices of the n-dimensional cube. (The vertices correspond to bit-strings of length n, and two vertices are adjacent if the corresponding bit-strings disagree in exactly one location.)

Toss a fair coin. If it's heads, stay put; if it's tails, choose randomly from one of the n edges incident to the vertex you currently occupy, and travel in that direction.

In terms of bit-strings:

Toss a coin. If it's heads, do nothing; if it's tails, choose randomly from one of the *n* positions in the current bit-string, and complement that bit.

Equivalently:

Choose randomly from one of the *n* positions, and then toss a penny; if it's heads, do nothing, and if it's tails, complement that bit.

Equivalently:

Choose randomly from one of the *n* positions, and then toss a nickel; if it's heads, replace that bit by a 0, and if it's tails, replace that bit by a 1.

What's good about this last way of running the Markov chain (tossing a nickel instead of tossing a penny) is

that there's an obvious rule for when to stop: As soon as you've picked (and re-randomized) every position in the bit-string, it's as likely to be any bit-string as any other, so you can stop; the state at that (random!) time is uniform on the set of bit-strings of length *n* (that is, it is governed by the stationary probability distribution exactly, without any error).

(Moral: Sometimes it helps to think of a Markov chain in a different way that gives the same transition probabilities.)

Note that the time it takes before this stopping rule delivers an output is a random variable of the coupon-collector kind: at each step, we pick one of the *n* positions uniformly at random, and we stop when we've picked each of them at least once. This is the same as the question "How many times do you have to roll an *n*-sided die before each of its faces has come up at least

once?" As we saw in an earlier lecture, if we call this random time τ, then
Prob($\tau > n \ln n + cn$) $\leq e^{-c}$, so we say it takes on the order of $n \ln n$ steps for our stopping rule to give us an output governed by the stationary distribution.

Stopping rules and total variation distance

It's can be shown that if we have a stopping rule, then $||\mathbf{u}\mathbf{P}^n - \mathbf{w}||_{TV}$ is bounded above by the probability that someone who runs the stopping rule will still be going (i.e., will not yet have stopped) after n steps. (See section 6.4 of Levin, Peres, and Wilmer's book.)

So if we can show that this probability is small (as in our estimate Prob($\tau > n \ln n + cn$) $\leq e^{-c}$), then we have a bound on the total variation distance between $\mathbf{u}\mathbf{P}^n$ and $\mathbf{w}$.

Shuffling cards

In a riffle shuffle, we divide a deck of cards into two equal (or roughly equal) stacks and then combine the stacks, preserving the order of the cards in each stack but interleaving the two stacks in a somewhat random way.

In a "backward riffle shuffle" (not actually used in casinos, but easier to analyze!), we randomly pull apart our deck into two stacks and then put one stack atop the other.

More specifically, we assign each card independently (with probability 1/2,1/2 respectively) to either the left stack or the right stack, and then put the left stack on top of the right stack.

How many backward riffle shuffles does it take to randomize a deck of 52 cards?

Imagine that when you backward riffle shuffle the deck, you randomly mark each card with a 0 or a 1; then you toss a nickel, putting the cards marked 0 into the left stack and the cards marked 1 into the right stack if the nickel came up heads, and vice versa if the coin came up tails.

You do this each time you shuffle, so that after two shuffles, each card is marked 00, 01, 10, or 11; after three shuffles, each card is marked with a bit-string of length 3; etc.

Note that if after some number of shuffles, two cards are marked with different bit-strings, then the first has a probability of $\frac{1}{2}$ of being above the other.

(Proof: Look at the last position in which the bit-strings differ; this is the time at which their order in the current deck got established. But at that time, the two stacks were randomly put one atop the other, according to the toss of a nickel.)

In fact, it can be shown that when all 52 cards are marked with different bit-strings, each of the 52! dif - ferent orderings of the cards are equally likely.

So, a stopping rule would be "Stop when all the cards are marked with different bit-strings."

How long will this take to happen?

An equivalent model: 52 walkers start at the root of a binary tree (where each node has an edge marked 0 and an edge marked 1). At each node, the walkers proceed to a random child of that node. How many levels do we need to go down before all the walkers are at different nodes? (Obviously at least six, since five levels down we have only 32 nodes, which are not enough to fit $52 > 32$ walkers.)

Write this random variable as X. Note that, exploiting the recursive structure of the binary tree, we can write the recursive relation

$$X = 1 + \max(Y, Z)$$

where Y is the number of additional levels the walkers who take the 0-branch need to take before they're all at different nodes, and Z is the number of additional levels the walkers who take the 1-branch need to take before they're all at different nodes.

The Y and Z variables are just like the X variable, except that instead of starting with 52 walkers, they start with r walkers and 52-r walkers, respectively, where r is a random integer distributed according to Binomial$(52, \frac{1}{2})$.

Applying this down the tree recursively, we get a very simple way to simulate the X-process:

```
Shuffle[n_] := Module[{r},
  If[n ≤ 1, 0, r = RandomInteger[BinomialDistribution[n, 1 / 2]];
   1 + Max[Shuffle[r], Shuffle[n - r]]]]

N[Sum[Shuffle[52], {k, 1000}] / 1000]
```

```
11.74
```

It can be shown rigorously (see e.g. www.dartmouth.edu/~chance/teaching_aids/books_articles/Mann.pdf) that the expected value of this random variable is given by a particular infinite sum whose value is about 11.7.

True casino card-shuffling is quite a bit less random than a riffle shuffle or a backward riffle shuffle.
So, if you gamble, make sure the dealer shuffles the deck at least a dozen times after taking it out of the box!

Exact sampling

The pinned steppingstone model

Recall the stepping stone model (Example 11.12in Grinstead and Snell): We have a 20-by-20 array of squares "with toroidal boundary conditions" (i.e., we glue the top edge to the bottom edge and the left edge to the right edge), where each square is initially black or white, and where at each step, we choose a square at random and change the color of that square to the color of one of the square's 8 neighbors.

This is an absorbing Markov chain with two absorbing states (all-white and all-black).

To stop the chain from being absorbing, we can pick

some squares that must start white and stay white and some other squares that must start black and stay black. I call this the "pinned steppingstone model"; see Lec05. (As far as I'm aware, nobody's written anything about this model in the research literature; I'm interested in understanding it better.)

The new chain is not absorbing, because from any state you can get to the all-the-squares-that-are-allowed-to-be-white-are-white state, AND from any state you can get to the all-the-squares-that-are-allowed-to-be-black-are-black state, in some finite number of steps.

These are recurrent states (recall the classification of states as transient versus recurrent). More importantly, they are **universally accessible** recurrent

states: you can get to them (with positive probability) from any other state, in a finite number of steps.

The new chain is not ergodic either, but it turns out that if you throw out the transient states, the chain that you're left with is ergodic; this is a consequence of the existence of universally accessible recurrent states.

So the pinned stepping stone model, pruned of its transient states, is ergodic and has a unique stationary measure π.

I've done simulations of a version of this model, in which the 20-by-20 torus is replaced by a 21-by-20 cylinder, the "colors" are 0 and 1, all the squares on the upper edge are pinned at 0, all the squares on the lower edge are pinned at 1, and each site has only 4 neighbors, not 8. Here's a typical state:

```
00000000000000000000
00000000000000000000
00000000000000000000
00000000000000000000
00000000000000000000
00000000000000000000
00000000000000000000
00000000000000000010
00000000000000001110
00000000000000000110
00000000000000001110
11110000000000001110
11110000000000000110
10011110000000000100
10111101000000001111
10001101100000000111
10001100100100010111
10001100101101111111
11111100011111111111
11111111011111111111
11111111111111111111
```

Harmonic functions again

The preceding picture shows clumping: the state of a site (0 versus 1) and the states of its neighbors are highly correlated. Indeed, with high probability, there are two large clumps with a few small islands. Being more quantitative requires non-trivial analysis of the stationary measure π. But there is at least one assertion about π that requires almost no work.

Claim: Let s be a particular square, and let $h(s)$ be the probability (with respect to the probability measure π) that square s has color 1. Then the function h is harmonic on the set of squares (that is, its value at any square equals the average of the values of its neighbors), leaving aside the pinned squares.

Proof: Look at two successive moments in time, say time n and time $n+1$, where the system is equilibrium at time n and hence is also in equilibrium at time $n+1$. The probability that s has color 1 at time $n+1$ equals $h(s)$, but it also equals

$p\ a(s) + (1-p)\ h(s)$, where p is the probability that square s gets recolored from time n to time $n+1$, and $a(s)$ is the average of $h(s)$ over the neighbors of s. So

$$h(s) = p\ a(s) + (1-p)\ h(s)$$

Subtracting $(1-p)\ h(s)$ from both sides and dividing by p gives

$$h(s) = a(s)$$

which is what we needed to show.

For the case of the 21-by-20 cylinder, it's easy to say what $h(s)$ is: the unique harmonic function that takes the value 0 at the top and 1 at the bottom is a linear function of the vertical coordinate of the square (and doesn't depend on the horizontal coordinate at all). So $h(s)$ is 0 in the top row, 1/20 in the next row, 2/20 in the row after that, and so on, and is 20/20 = 1 in the final (21st) row.

Already this has a non-obvious consequence, namely, that the border between the two "voting blocs" (the white bloc and the black bloc) does not stay poised halfway between the pinned squares, but moves back and forth as the Markov chain evolves. That is, the statistical equilibrium (aka stationarity) does not lead to a large-scale equilibrium in the location of the interface between the two clusters. Putting it differently: unlike the case of the three-competing-colors model

from the start of the lecture, for which the stationary measure appears to have the property that most cells have a favored color that they are almost certain to have, the stationary distribution for the pinned stepping stone model has the property that most cells spend some time having each of the two colors. It's a very different sort of interface.