Lecture 10: FTLEs for the baker's map

Recall the (generalized) baker's map:

$$X = (x,y) \in [0,1]^2. \qquad X_{n+1} = \varphi(X_n)$$

$$x_{n+1} = \begin{cases} \alpha x_n & y_n < \alpha \\ \alpha + \beta x_n & y_n > \alpha \end{cases}$$

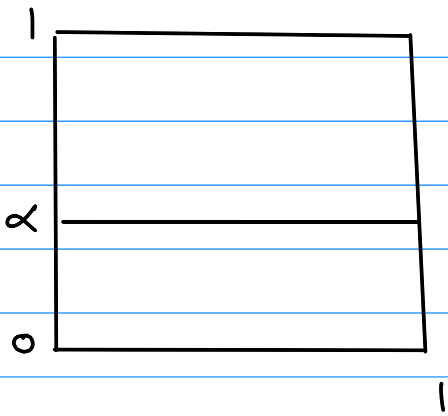
$$\alpha + \beta = 1$$

$$y_{n+1} = \begin{cases} \alpha^{-1} y_n & y_n < \alpha \\ \beta^{-1}(y_n - \alpha) & y_n > \alpha \end{cases}$$

We computed Lyapunov exponents

$$\lambda^{(2)} = -\lambda^{(1)} = \lambda = \alpha \log \alpha^{-1} + \beta \log \beta^{-1}$$

We used Birkhoff's ergodic thm, which suggest iterates land in lower box with probability α, upper box w. prob β.



Suggests writing

$$\lambda_n = \frac{1}{n} \sum_{k=1}^{n} h_k$$

where $h_k = \log \alpha^{-1}$ w. prob α

$\log \beta^{-1}$ w. prob β

are i.i.d.

If $n \to \infty$ the law of large numbers says $\lambda_n \to \lambda$.
What about for finite n? What is the PDF of λ_n?
(prob. dist. func.)
This is a standard problem in probability.
For small deviations the central limit thm holds, but we'll be interested in large deviations:

$$\mathbb{P}\left\{ |\lambda_n - \lambda| < \frac{c}{\sqrt{n}} \right\} \quad \text{vs} \quad \mathbb{P}\left\{ |\lambda_n - \lambda| < c \right\}$$

small deviations — large deviations

$$\boxed{\langle h \rangle = \lambda = \alpha \log \alpha^{-1} + \beta \log \beta^{-1}}$$

$$\sigma^2 = \langle (h-\lambda)^2 \rangle = \alpha\left(\log \alpha^{-1} - \lambda\right)^2 + \beta\left(\log \beta^{-1} - \lambda\right)^2$$

$$= \alpha\left(\underbrace{(1-\alpha)}_{\beta} \log \alpha^{-1} - \beta \log \beta^{-1}\right)^2 + \beta\left(\underbrace{(1-\beta)}_{\alpha} \log \beta^{-1} - \alpha \log \alpha^{-1}\right)^2$$

$$= \alpha\beta^2 \left(\log(\alpha^{-1}/\beta^{-1})\right)^2 + \beta\alpha^2\left(\log(\beta^{-1}/\alpha^{-1})\right)^2$$

$$= \alpha\beta(\alpha+\beta)\log^2(\alpha/\beta)$$

$$\boxed{\sigma^2 = \alpha\beta \log^2(\alpha/\beta)}$$

To find the distribution of λ_n, first find the generating function for h:

$$\langle e^{hs} \rangle = \alpha e^{s \log \alpha^{-1}} + \beta e^{s \log \beta^{-1}}$$

The generating function for λ_n is:

$$\langle e^{\lambda_n s} \rangle = \langle e^{\frac{s}{n} \sum h_n} \rangle$$

$$= \langle \prod_n e^{\frac{s}{n} h_n} \rangle$$

$$= \langle e^{\frac{s}{n} h} \rangle^n \quad \text{since } h_n \text{ are i.i.d.}$$

$$= \left(\alpha e^{\frac{s}{n} \log \alpha^{-1}} + \beta e^{\frac{s}{n} \log \beta^{-1}} \right)^n$$

$$= e^{n \Lambda(s/n)}$$

where

$$\Lambda(s) = \log \left(\alpha e^{s \log \alpha^{-1}} + \beta e^{s \log \beta^{-1}} \right)$$

We want to invert the two-sided Laplace transform to obtain the prob. density function of λ_n, for large n.

Cramér's theorem then tells us we need to find:

$$I(x) = \sup_s \{ xs - \Lambda(s) \}$$

Assume $0 < \alpha < 1/2$, so $\alpha < \beta$ and $\log \alpha^{-1} > \log \beta^{-1}$.

$$\Lambda(s) = s \log \beta^{-1} + \log\left(\alpha e^{s(\log \alpha^{-1} - \log \beta^{-1})} + \beta\right)$$

We can find $I(x)$ using calculus:

$$\frac{d}{ds}\left(xs - \Lambda(s)\right) = x - \Lambda'(s)$$

$$= x - \log \beta^{-1} - \frac{\alpha(\log \alpha^{-1} - \log \beta^{-1})\, e^{(\log \alpha^{-1} - \log \beta^{-1}) s}}{\alpha e^{s(\log \alpha^{-1} - \log \beta^{-1})} + \beta}$$

$$= 0$$

Solve for $s = s_*(x)$!

$$s_*(x) = \frac{\log\left(\frac{\beta(x - \log \beta^{-1})}{\alpha(\log \alpha^{-1} - x)}\right)}{\log \alpha^{-1} - \log \beta^{-1}}, \quad \log \beta^{-1} < x < \log \alpha^{-1}.$$

Hence,

$$I(x) = \log\left(\frac{\log \alpha^{-1} - x}{\beta(\log \alpha^{-1} - \log \beta^{-1})}\right) + \left(\frac{x - \log \beta^{-1}}{\log \alpha^{-1} - \log \beta^{-1}}\right) \log\left(\frac{\beta(x - \log \beta^{-1})}{\alpha(\log \alpha^{-1} - x)}\right)$$

This is the rate function (or Cramér function, or entropy function)

Note that $I(x)$ has a minimum at $x = \lambda$, and that

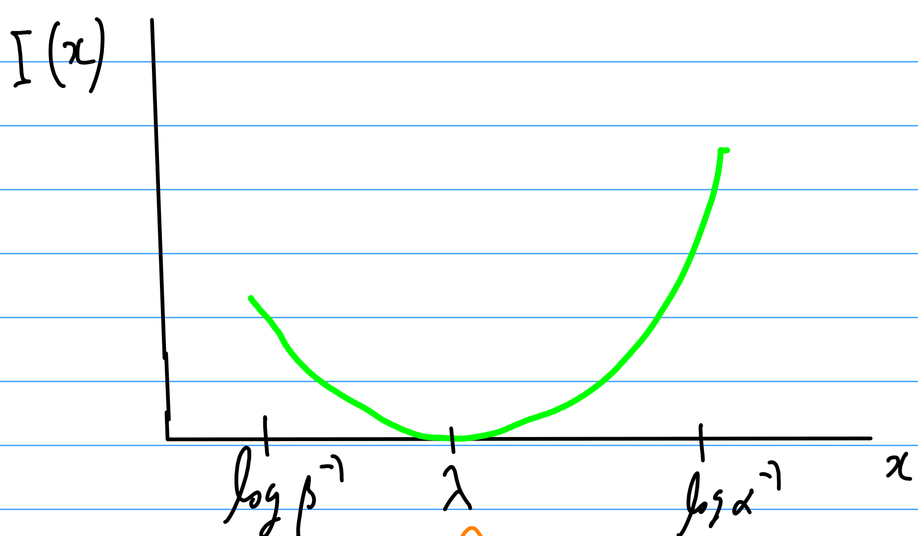
$$I''(\lambda) = 1/\sigma^2$$

We have finally that

$$\boxed{\rho_{\lambda_n}(h) \sim \sqrt{\frac{n}{2\pi\sigma^2}}\, \exp\left(-n\, I(h)\right)} \qquad n \to \infty$$

$$\log \beta^{-1} < h < \log \alpha^{-1}$$

Sketch for $\alpha = 1/4$, $\beta = 3/4$:



$I(x) \sim \frac{(x-\lambda)^2}{2\sigma^2}$ Gaussian approximation

This is normalized properly, since for large n:

$$\int_{\log\beta^{-1}}^{\log\alpha^{-1}} \rho_{\lambda_n}(h)\,dh \sim \int_{-\infty}^{\infty} \sqrt{\frac{n}{2\pi\sigma^2}} \exp\left(-n\frac{I''(\lambda)(h-\lambda)^2}{2}\right) dh$$

$$\xrightarrow{n\to\infty} = 1 \quad (+\text{ exponentially small terms})$$

This PDF matches simulations very well.

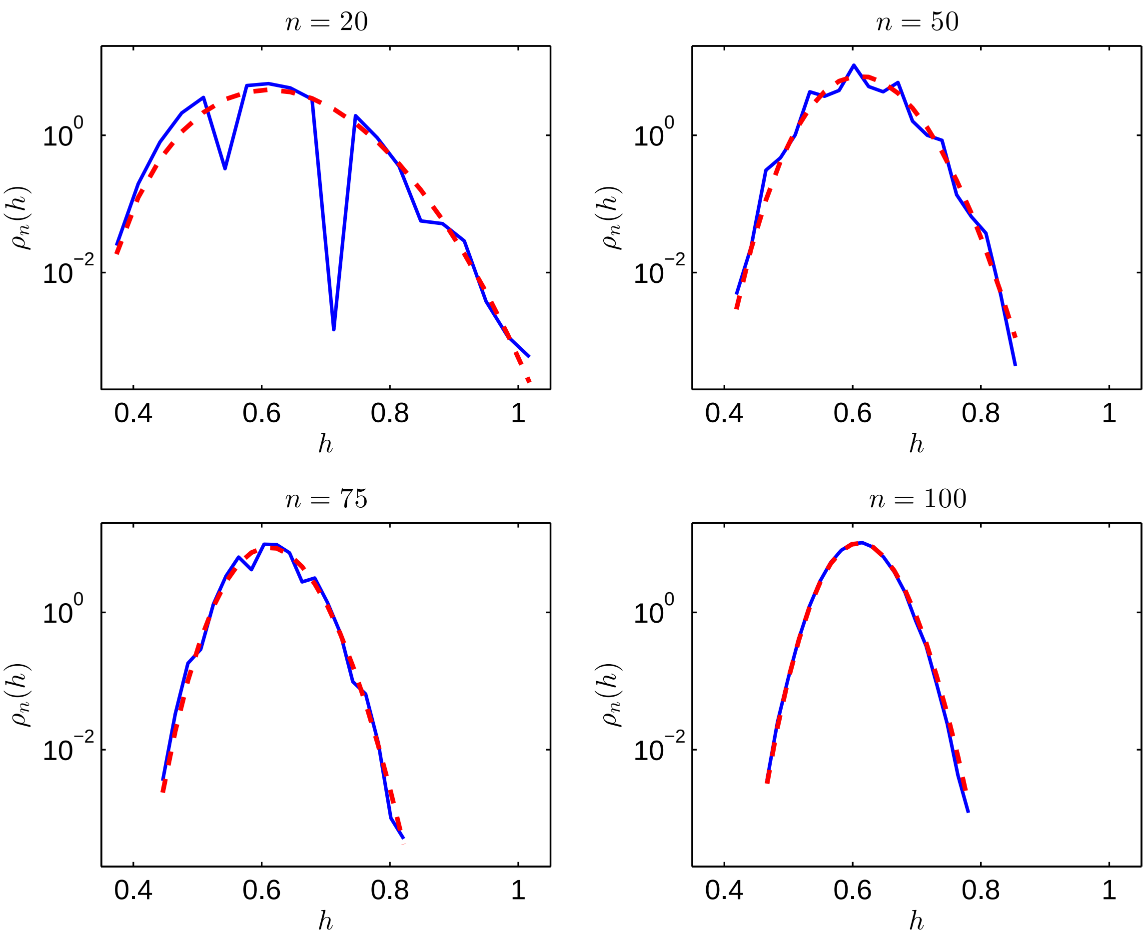


Figure 1: The probability distribution of finite-time Lyapunov exponents for the baker's map with $\alpha = 0.3$. As the iterate n increases, the distribution converges to the large-deviation probability density, $\rho_n(h)$ (dashed red). The distribution was computed using 10^5 randomly-distributed trajectories. See Figs. 2 and 3 for the Matlab code used to generate these figures.

```
function dist_baker

Npts = 100000; Niter = 100; % number of initial points and iterates
alpha = .3; beta = 1-alpha;

% The large-deviation form of the PDF.
mh = @(h) (h + log(beta))/(log(beta/alpha));
G = @(h) mh(h).*log(mh(h)) + (1-mh(h)).*log(1-mh(h)) ...
    + mh(h)*log(beta/alpha) - log(beta);
sigma2 = alpha*beta*log(alpha/beta)^2;
Pld = @(h,n) sqrt(n/2/pi/sigma2) * exp(-n*G(h));

% Specify which 4 values of iterate n to plot.
plotn = [20 50 75 100]; plotgeom = [2 2]; nplot = 0;
% Generate random initial conditions.
rng('default'); X = rand(Npts,2);
lstr = zeros(Npts,1); % the log-stretch
for n = 1:Niter
  [X,str] = baker(X,alpha);  % apply baker's map (vectorized)
  lstr = lstr + log(str);
  if any(plotn == n)
    % Plot the results.
    nplot = nplot + 1; subplot(plotgeom(1),plotgeom(2),nplot)
    % Histogram of average stretching, normalized.
    [P,bins] = hist(lstr/n,20); P = P/trapz(bins,P);
    semilogy(bins,P,'b','LineWidth',1.5), hold on
    semilogy(bins,Pld(bins,n),'r--','LineWidth',2), hold off
    xlabel('$h$','Interpreter','LaTeX')
    ylabel('$\rho_{n}(h)$','Interpreter','LaTeX')
    axis([.35 1.05 2e-4 2e1])
    title(sprintf('$n = %d$',n),'Interpreter','LaTeX')
  end
end
print -dpdf dist_baker.pdf
```

Figure 2: The Matlab code `dist_baker.m`.

```
function [Xn,stretch] = baker(X,al)
%BAKER   Baker's map.
%   XN = BAKER(X) returns the image of X=(x,y) under the action of the
%   baker's map.  BAKER(X,ALPHA) returns the generalized baker's map, where
%   0 < ALPHA < 1.  X must be in the unit square [0,1]^2.  X can also be an
%   array with a 2-vector on each row.
%
%   [XN,STRETCH] = BAKER(X) returns a vector STRETCH that records the
%   vertical stretching experienced by the particle (1/alpha or
%   1/(1-alpha)).  This is used to reconstruct the tangent map.

if nargin < 2
  % Default is the uniform baker's map.
  al = 0.5;
end

if al > 1 || al < 0
  error('Baker''s map requires 0 < alpha < 1.')
end

be = 1-al;
x = X(:,1); y = X(:,2);
xn = zeros(size(x)); yn = zeros(size(y));  % Allocate arrays.

% Formula for y <= alpha.
ila = find(y <= al);
xn(ila) = al*x(ila);
yn(ila) = y(ila)/al;
% Formula for y > alpha.
iga = find(y >  al);
xn(iga) = al + be*x(iga);
yn(iga) = (y(iga) - al)/be;

Xn = [xn yn];

if nargout > 1
  stretch = 1/al*ones(size(xn));
  stretch(iga) = 1/be;
end
```

Figure 3: The Matlab code `baker.m`.