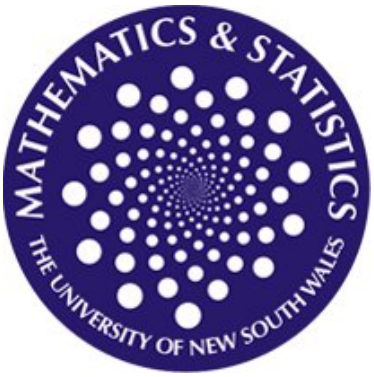
UNSW
AUSTRALIA



University of New South Wales

School of Mathematics and Statistics

Assignment 2

Ergodic Theory

Author:
Adam J. Gray

Student Number:
3329798

1

1.1

$$T(\theta, \nu) = (\theta + \nu \mod 2\pi, \alpha\nu + \gamma\cos(\theta + \nu))$$

1.2

In any neighbourhood not around $2n\pi$ for $n \in \mathbb{Z}$ we have that

$$J[T](\theta, \nu) = \begin{pmatrix} 1 & 1 \\ -\gamma\sin(\theta + \nu) & \alpha - \gamma\sin(\theta + \nu) \end{pmatrix}$$

and so

$$\begin{aligned} |J[T](\theta, \nu)| &= \alpha - \gamma\sin(\theta + \nu) + \gamma\sin(\theta + \nu) \\ &= \alpha. \end{aligned}$$

So $|J[T](\theta, \nu)| = 0$ iff $\alpha = 0$. So since we have chosen $\alpha > 0$ this map is invertible everywhere except around at $2n\pi$ by the inverse function theorem.

1.3

See code in *Question_3.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_1_Code`.

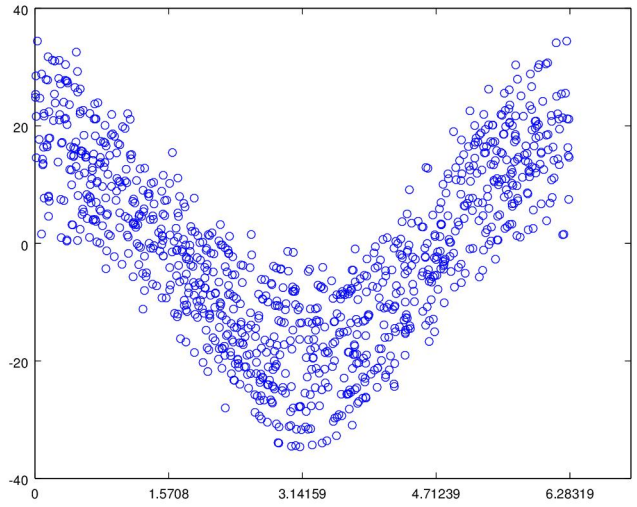


Figure 1: Phase plot of velocity vs table phase with $\theta_0 = 0.5, \nu_0 = 0.5$

1.4

The maximum post-bounce velocities occur near $\theta = 0$ ($\theta = 2\pi$). This makes sense because the piston is travelling in the opposite direction to the ball. This is happening fastest at $\theta = 0$ and so this results in the largest bounces. The minimum is found at $\theta = \pi$. This also makes sense because this is when the piston is travelling in the same direction as the ball. This is happening fastest at $\theta = \pi$ and this results in the largest slowdown in the bounce.

1.5

This does not preserve Lebesgue measure. To see why consider $\Omega = [\frac{\pi}{2}, \pi] \times [1, 2]$ and note that $\mu(\Omega) = \frac{\pi}{2}$. It is clear to see, however, that

$$T^{-1}T(\Omega) \supseteq [\frac{\pi}{2}, \pi] \times [1, 2] \cup [\pi, \frac{3\pi}{2}] \times [1, 2]$$

and thus

$$\mu(T^{-1}T(\Omega)) \geq 2\mu(\Omega)$$

and hence this map does not preserve Lebesgue measure. Is this a contradiction of ergodicity and mixing? No. While an ergodic map must be measure preserving, there is no requirement that it preserves the Lebesgue measure specifically.

1.6

See code in *Question_6.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_1_Code`.
What figure 2 shows is that a small ball of points *spreads out* in the phase space. Note that this ensemble of points roughly matches the orbit of one point. This would lead us to suspect that this process is Ergodic.

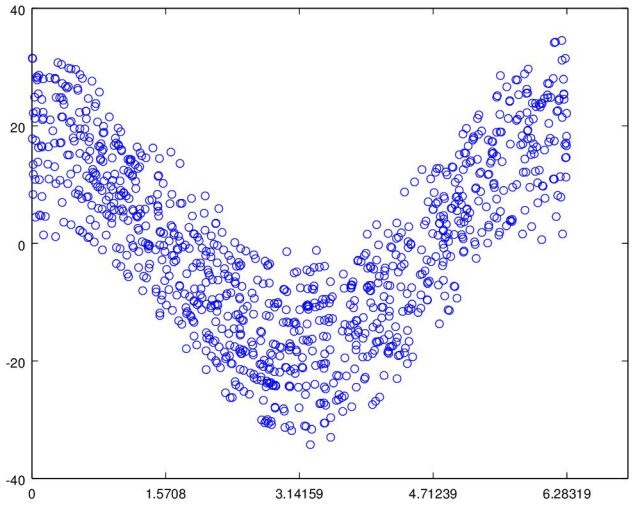


Figure 2: A bundle of 1000 points starting near $\theta_0 = 0.5, \nu_0 = 0.5$ were iterated through T 10000 times

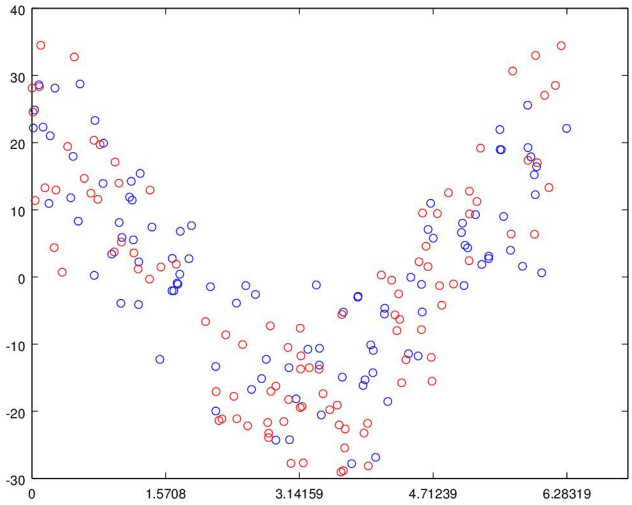


Figure 3: Two bundles of 200 points after iterating through T 10000 times. Blue started near $\theta_0 = 0.5, \nu_0 = 0.5$ while red started near $\theta_0 = 1, \nu_0 = 1$

Figure 3 suggests that the long term behaviour of the points is independent of their starting point. That is two sets of balls mix together. Although only testing with two bundles of balls is hardly rigorous, even empirically, this would suggest that this process is mixing.

1.7

See code in *Question_7.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_1_Code`. Since we suspect that this process is Ergodic (and mixing) this question can be answered by watching the long term behaviour of 1 point. We do this for a point starting at $\theta_0 = 0.5, \nu_0 = 0.5$ and see that 49.95% of the first 10000 bounces occur when $\theta < \pi/2$ or $\theta > 3\pi/2$, that is when the piston is moving up. We conclude that there is equal chance of the ball hitting on a down stroke compared to an up-stroke.

1.8

See code in *Question_8.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_1_Code`. Again since we suspect that this process is Ergodic we just study the evolution of one point. We see that with $\theta_0 = 0.5, \nu_0 = 0.5$ the maximum ν achieved is 34.821 and the 945 bounces are within 1% of this velocity over 10^6 iterations. This

suggests that ball is within 1% of its maximum post bounce speed about 0.945% of the time.
The above paragraph is actually wrong. Let me explain. If we consider the distribution of ν as shown in figure 4 we can see that by taking different orbit lengths we essentially get different values for the percentage of time that ν is within 1% of its observed maximum. This is because with an orbit length of say 10^4 we would get the first black line as the observed maximum whereas with 10^5 iterations we get the second black line as the observed maximum. We can easily see that the -1% brackets of these two lines correspond to different parts of the distribution and therefore correspond to different observed percentages. Precisely this problem occurs in the next two questions as well..

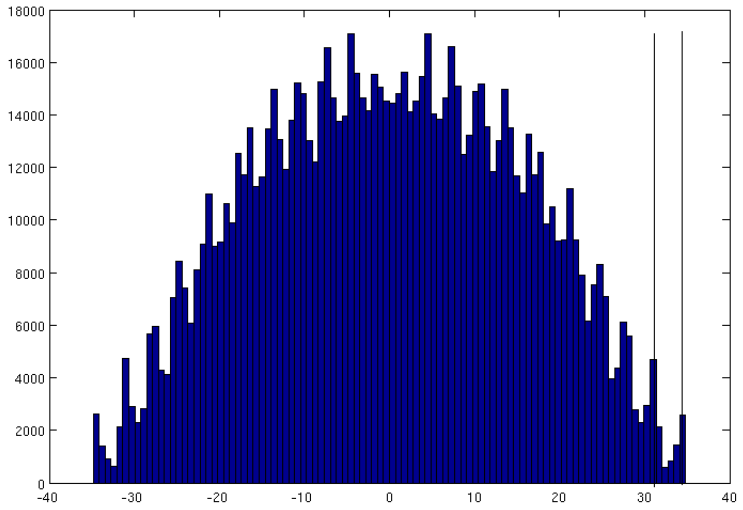


Figure 4: A histogram of ν taken from 10^6 iterations of T starting at $\theta_0 = 0.5$ and $\nu_0 = 0.5$

1.9

See code in *Question_9.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_1_Code`.
We use two methods for calculating the expected return time. The first method is by saying $\mathbb{E}[r] = \frac{N}{N_h+1}$ where r is the return time, N is the number of iterations of the T map in the orbit and N_h is the number of times it falls within the region of interest. This uses the fact that we suspect T is ergodic to say that n-th return times should be the same as the 1-st return time. Using this method with $\theta_0 = 0.5, \nu_0 = 0.5$ and using 10^6 iterations we get that $\mathbb{E}[r] = 1258$.

We can also do a similar thing for an ensemble of 1000 points but only measuring the 1-st return time (instead of every return). Doing this we get $\mathbb{E}[r] = 1140$. Notice that these two numbers are actually quite different. Despite the fact that we use the same orbit lengths (10^6) we most likely encountering this difference due to the issue outlined in the previous question.

1.10

See code in *Question_9.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_1_Code`. (Note we have code from question 9 and 10 together).
We have used the second method outlined above to produce the histogram below. We can see that this is probably an exponential distribution (as expected) with parameter = 1/1140.

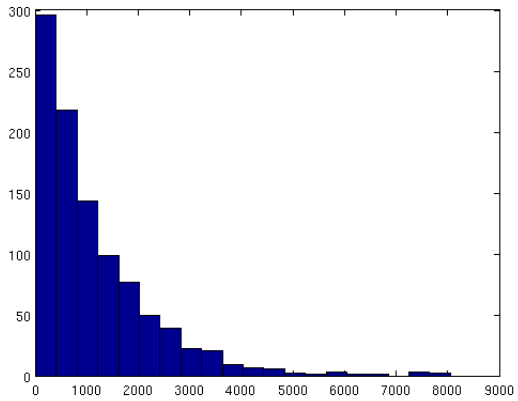


Figure 5: A histogram of the first return time to the high bounce region.

2

2.1

See code in *PageRank.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`

2.2

See code in *Question_2.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`.

The resultant vector is

$$q = (0.011878, 0.016927, 0.021698, 0.021698, 0.017105, 0.025485, 0.013031, 0.017105, 0.024870, 0.027504, 0.039333, 0.013031, 0.013898, 0.025485, 0.025485, 0.021174, 0.013031, 0.013031, 0.013031, 0.015050, 0.103970, 0.011878, 0.097454, 0.111770, 0.285079)$$

2.3

See code in *Question_3.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`.
For $p = 0.2$ the resultant vector is

$$q = (0.018891, 0.020780, 0.019820, 0.019820, 0.020144, 0.020634, 0.019139, 0.020144, 0.020823, 0.021390, 0.024838, 0.019139, 0.019647, 0.020634, 0.020634, 0.021618, 0.019139, 0.019139, 0.019139, 0.019895, 0.041789, 0.018891, 0.038559, 0.041962, 0.453391).$$

For $p = 0.5$ the resultant vector is

$$q = (0.015820, 0.019775, 0.019782, 0.019782, 0.018879, 0.021813, 0.016438, 0.018879, 0.021982, 0.023395, 0.030867, 0.016438, 0.017402, 0.021813, 0.021813, 0.021951, 0.016438, 0.016438, 0.016438, 0.018020, 0.071322, 0.015820, 0.065319, 0.073702, 0.379674).$$

In figure 6 we see that there are basically 3 important nodes, 21, 23 and 24. When choosing a smaller value for p we see that across the first 24 nodes the distribution becomes more uniform. If we want to make sure that we keep these nodes as *important* we might choose some intermediate value of p. Say $p = 0.50$. To see a comparison of different weightings given for different values of p see figure 7. Note that in this figure we have included the 25-th *Google* node and because this node is connected to every other node by definition it outranks all other nodes. In some sense one should ignore the rank of the *Google* node when assessing the results.

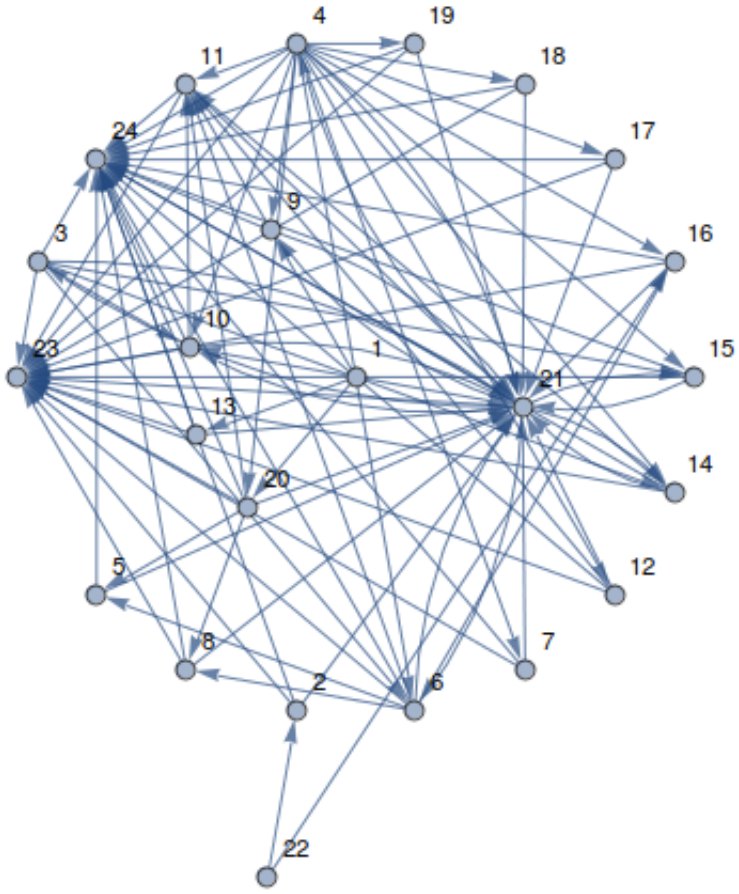


Figure 6: The layout of the network

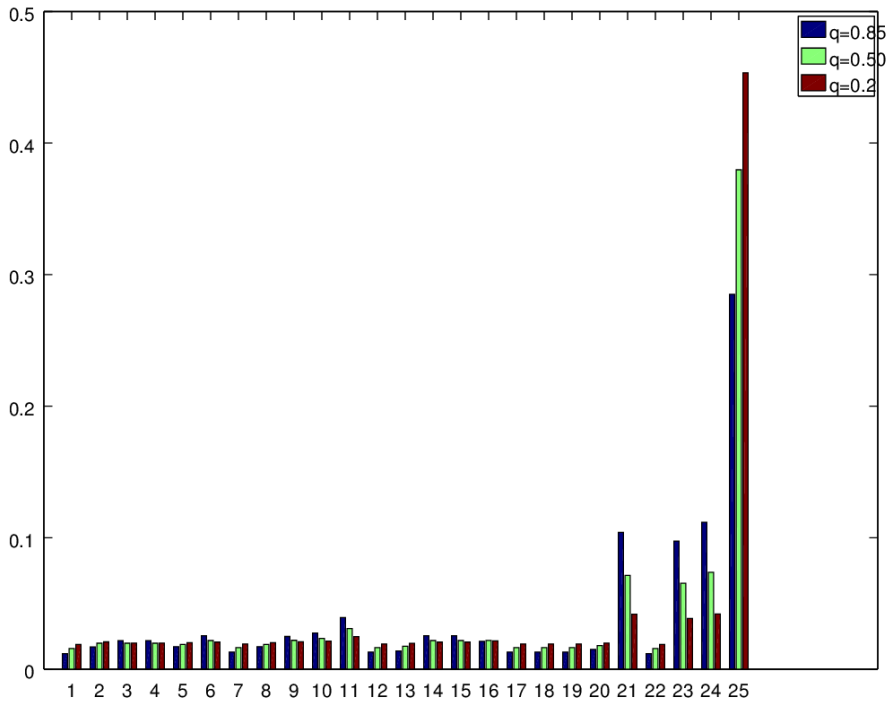


Figure 7: Different weightings for different values of p.

2.4

See code in *Question_4.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`. The solution converges exponentially with the error at step n given by

$$Error(n) = \exp(-0.48640n - 0.25550).$$

This is true for about $n \leq 65$, after that the error has reached machine precision and cannot improve any further. See figure 8 for a plot of $\log(Error(n))$.

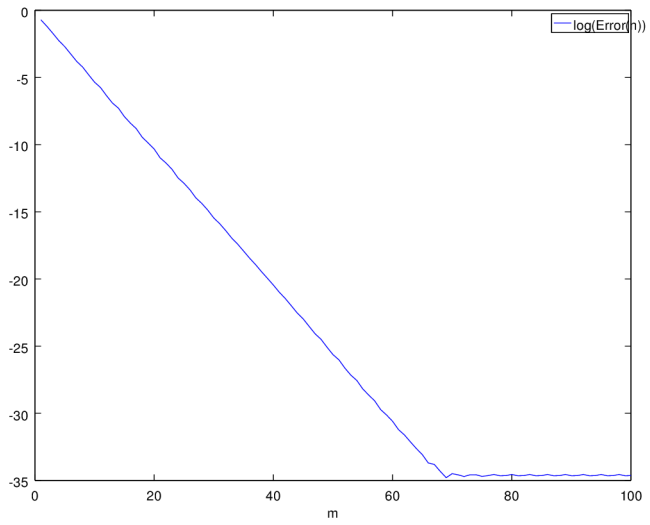


Figure 8: Exponential convergence to the stationary vector. Note that at about $n = 65$ the graph flattens out. This is because the error has reached machine precision.

2.5

2.5.1

See code in *PageRank2.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`.

2.5.2

See code in *Question_5_b.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`.

$$q = (0.023011, 0.029826, 0.031190, 0.031190, 0.028626, 0.034984, 0.024166, 0.028626, 0.035070, 0.037710, 0.051047, 0.024166, 0.025737, 0.034984, 0.034984, 0.033940, 0.024166, 0.024166, 0.024166, 0.026891, 0.124281, 0.023011, 0.114333, 0.129734)$$

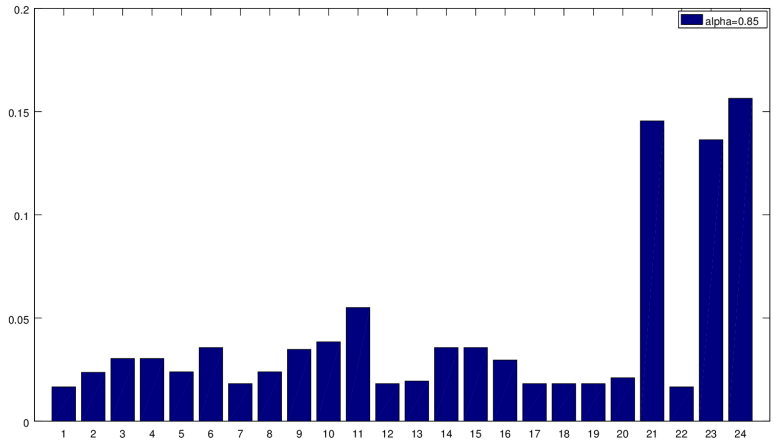


Figure 9: The page rank weightings for this network using the second page rank algorithm.

We can see from figure 9 and by comparing to figure 7, that the this second page rank method results in a more *uniform* distribution or page ranks (we include consideration of the *Google* node in this remark). This is because there is a *latent* connectivity between all pages introduced by the probability of just going to a random page, regardless of the links. This means that each page is *one hop* away from every other page, instead of two hops (via the *Google* node) in the case of the first method.
If we discard the *Google* node and re-normalise in the $p = 0.85$ in the Page Rank vector from the first setup we can observe a perhaps better comparison between the two methods in figure 10. From we can see that these two methods are essentially equivalent if we discount the *Google* node.

2.5.3

See code in *Question_5_c.m* available at `https://github.com/adamjoshuagray/Honours_Ergodic_Theory/tree/master/Assignment_2/Project_2_Code`.
The solution converges exponentially with the error at step n given by

$$Error(n) = \exp(-1.04629n - 0.25217).$$

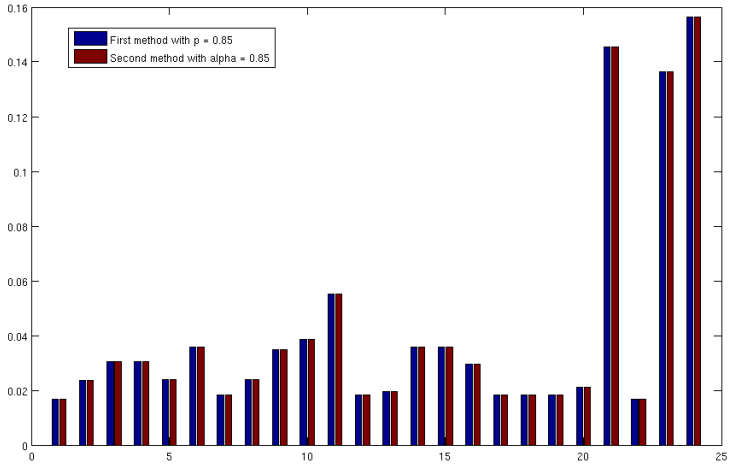


Figure 10: A comparison between the first and second methods with $p = 0.85$ and $\alpha = 0.85$ respectively.

This is true for about $n \leq 30$, after that the error has reached machine precision and cannot improve any further. See figure 11 for a plot of $\log(Errror(n))$.
It converges faster because the second eigenvalue of G is $|\lambda_2| = 0.3511$ and the second eigenvalue of Q is $|\lambda_2| = 0.4538$ and so the power method will converge for G faster than Q. It also converges faster because the uniform distribution is in a *1 norm* sense, *closer* to q than in the first case (because the first case has a *Google* node.

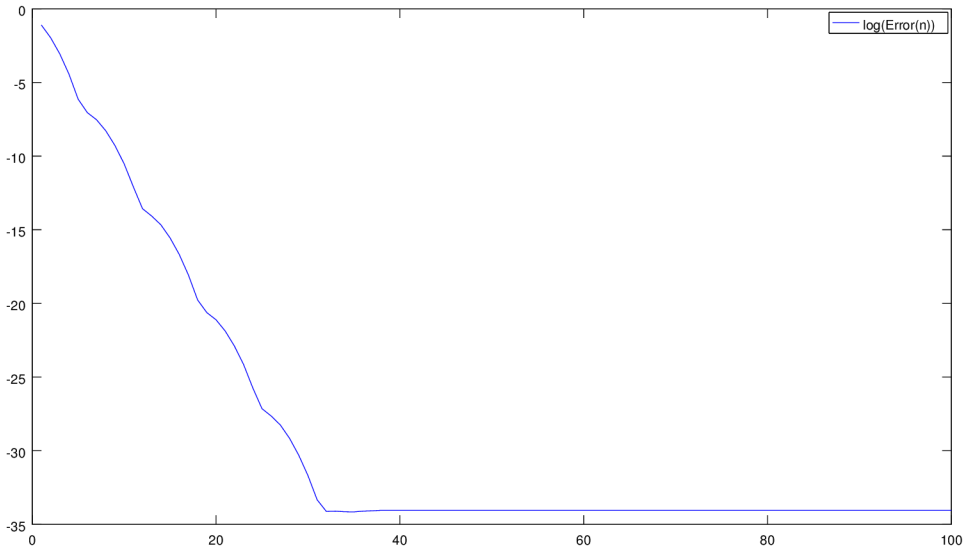


Figure 11: Exponential convergence to the stationary vector. Note that at about $n = 30$ the graph flattens out. This is because the error has reached machine precision.