Politecnico di Torino

Master's degree in Data Science and Engineering

Computational Linear Algebra

Homework 3
PageRank

Lisa Vandi 360247
Riccardo Polazzi 361352

Academic Year 2025/2026

Contents

Chapter 1

Introduction

The current report presents the solutions to a selection of exercises from the paper *"The $25,000,000,000 Eigenvector: The Linear Algebra Behind Google"*.

We have selected and solved the following exercises to demonstrate a comprehensive understanding of the mathematical principles underpinning the PageRank algorithm:

- **Exercise 4:** analysis of dangling nodes and substochastic matrices.
- **Exercise 5:** theoretical proof regarding pages with no backlinks.
- **Exercise 7:** stochasticity of the Google Matrix M.
- **Exercise 8:** properties of products of stochastic matrices.
- **Exercise 9:** minimum score assignment for isolated pages.
- **Exercise 12:** comparison of rankings with and without the damping factor.
- **Exercise 13:** resolution of disconnected web components.
- **Exercise 14:** computational convergence analysis of the Power Method.
- **Exercise 15:** spectral analysis of convergence rates.
- **Exercise 17:** the trade-off in choosing the damping factor m.

These exercises cover the core theoretical aspects (spectral theory, matrix properties) as well as practical implementation details (convergence speed, handling of edge cases).

Chapter 2

Exercises

2.1 Exercise 4

Text: In the web of Figure 2.1, remove the link from page 3 to page 1. In the resulting web page 3 is now a dangling node. Set up the corresponding substochastic matrix and find its largest positive (Perron) eigenvalue. Find a non-negative Perron eigenvector for this eigenvalue, and scale the vector so that components sum to one. Does the resulting ranking seem reasonable?

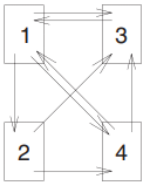

Figure 2.1: Network topology for Exercise 4

Solution: To solve this problem, we proceed in three analytical steps.
Firstly, we modify the link matrix A to reflect the topological change.
Secondly, we formulate the eigenvalue problem $A\mathbf{x} = \lambda\mathbf{x}$. By analyzing the dependencies between variables, we isolate the active subsystem to derive the condition for λ and compute the eigenvector.
Finally, we analyze the resulting probability distribution.

2.1.1 Step 1: matrix formulation

Removing the link $3 \to 1$ creates a dangling node at Page 3 ($n_3 = 0$). The third column of the adjacency matrix becomes entirely zero, making the matrix A

column-substochastic:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

We seek the largest positive eigenvalue λ and the corresponding eigenvector $\mathbf{x} = [x_1, x_2, x_3, x_4]^T$ such that $A\mathbf{x} = \lambda\mathbf{x}$. Writing this out as a system of linear equations, we get:

$$\frac{1}{2}x_4 = \lambda x_1 \tag{2.1}$$

$$\frac{1}{3}x_1 = \lambda x_2 \tag{2.2}$$

$$\frac{1}{3}x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_4 = \lambda x_3 \tag{2.3}$$

$$\frac{1}{3}x_1 + \frac{1}{2}x_2 = \lambda x_4 \tag{2.4}$$

We observe a critical structural property in the system above: the variable x_3 appears **only** in Equation 2.3, and nowhere else. This occurs because Page 3 is a dangling node: it receives probability mass (so x_3 depends on x_1, x_2, x_4) but does not redistribute it (so x_1, x_2, x_4 do not depend on x_3).

Consequently, equations 2.1, 2.2 and 2.4 form a self-contained **subsystem** governing the probability flow between pages 1, 2, and 4. We must solve this independent subsystem first to find the eigenvalue λ. Equation 2.3 will be used subsequently to determine x_3.

2.1.2 Step 2: finding λ

We solve the subsystem using the substitution method to express variables in terms of x_1:

1. From Equation 2.1, isolate x_4: $\quad x_4 = 2\lambda x_1$.
2. From Equation 2.2, isolate x_2: $\quad x_2 = \frac{1}{3\lambda}x_1$.

Substituting these into Equation 2.4:

$$\frac{1}{3}x_1 + \frac{1}{2}\left(\frac{1}{3\lambda}x_1\right) = \lambda(2\lambda x_1)$$

Assuming a non-trivial solution ($x_1 \neq 0$), we divide by x_1 and simplify:

$$\frac{1}{3} + \frac{1}{6\lambda} = 2\lambda^2 \implies 12\lambda^3 - 2\lambda - 1 = 0$$

This cubic equation represents the condition for the existence of a non-trivial eigenvector for the subsystem. Solving numerically (see Figure 2.2), the unique positive root is:

$$\lambda \approx 0.56135$$

```python
print("============================================")
print("                 EXERCISE 4")
print("============================================")

# Coefficients of the characteristic polynomial: 12*x^3 + 0*x^2 - 2*x - 1 = 0
# Order: [x^3, x^2, x^1, x^0]
coeffs = [12, 0, -2, -1]

# Compute the roots of the polynomial
roots = np.roots(coeffs)

real_components = roots.real

print("All roots found:")
print(real_components)

real_roots_mask = np.isreal(roots)
pure_real_roots = roots[real_roots_mask].real

# Filter to extract only the positive real root
lambda_val = pure_real_roots[pure_real_roots > 0][0]

print(f"The eigenvalue (positive real) is: {lambda_val:.5f}")
```

✓ 0.0s

```
============================================
                 EXERCISE 4
============================================
All roots found:
[ 0.56135324 -0.28067662 -0.28067662]
The eigenvalue (positive real) is: 0.56135
```

Figure 2.2: Python calculation for Exercise 4 roots.

2.1.3 Step 3: computing the eigenvector components

The system $A\mathbf{x} = \lambda\mathbf{x}$ defines the *direction* of the eigenvector, but not its magnitude. Because of linearity, if $\mathbf{x}$ is a solution, any scalar multiple $k\mathbf{x}$ is also a solution. The system therefore has one degree of freedom. To fix a specific numerical solution, we arbitrarily set the scale by choosing $\mathbf{x_1} = \mathbf{1}$.

Using $\lambda \approx 0.56135$ and $x_1 = 1$, we compute the remaining components:

- **Page 2:** $x_2 = \frac{1}{3\lambda} \approx 0.5938$
- **Page 4:** $x_4 = 2\lambda \approx 1.1227$
- **Page 3 (The Rank Sink):** Now we use Equation 2.3:

$$\lambda x_3 = \frac{1}{3}(1) + \frac{1}{2}(0.5938) + \frac{1}{2}(1.1227) \implies x_3 \approx \frac{1.1916}{\lambda} \approx 2.1227$$

The unnormalized eigenvector is $\mathbf{v} \approx [1.0, 0.5938, 2.1227, 1.1227]^T$.

2.1.4 Normalization and Result

To obtain valid probability scores, we normalize the vector such that $\sum x_i = 1$. The sum of components is $S \approx 4.8392$. Dividing each component by S, we obtain the final ranking vector:

$$\mathbf{x} \approx \begin{bmatrix} 0.207 \\ 0.123 \\ 0.439 \\ 0.232 \end{bmatrix}$$

The resulting hierarchy is Page $3 > 4 > 1 > 2$.

- **Mathematically:** the ranking is reasonable.
  Page 3 acts as a "Rank Sink": it receives links from all other nodes but has no outgoing edges. Probability mass flows into it and accumulates before "leaking" out of the system (reflected by $\lambda < 1$).
- **Web Search Context:** the ranking is not reasonable.
  A page that provides no outgoing links acts as a dead end for the user. It should not be considered the most important page on the web solely because it traps users. This highlights the "Rank Sink" problem and justifies the introduction of the rank distribution patch $(1/N)$ for dangling nodes used in the Google Matrix construction to fix this anomaly.

2.2 Exercise 5

Text: Prove that in any web the importance score of a page with no backlinks is zero.

Proof: Let A be the $n \times n$ link matrix of the web, where $A_{ij} \neq 0$ if and only if there is a link from page j to page i.
Let $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ be the eigenvector such that:

$$A\mathbf{x} = \mathbf{x}$$

The i-th component of this equation is given by:

$$x_i = \sum_{j=1}^{n} A_{ij} x_j$$

Assume page k has no backlinks (incoming links). This implies that no page j links to k. In terms of the matrix A, this means the k-th row contains only zeros:

$$A_{kj} = 0 \quad \forall j \in \{1, \ldots, n\}$$

Substituting this into the eigenvector equation for the index k:

$$x_k = \sum_{j=1}^{n} A_{kj} x_j = \sum_{j=1}^{n} 0 \cdot x_j = 0$$

Therefore, the importance score x_k must be zero.

Note: This limitation of the raw link matrix A is one of the reasons for introducing the Google Matrix $M = (1-m)A + mS$. In the modified model, every page receives a minimum probability mass from the teleportation matrix S, ensuring no page has a score of strictly zero.

2.3 Exercise 7

Text: Prove that if A is an $n \times n$ column-stochastic matrix and $0 \leq m \leq 1$, then $M = (1-m)A + mS$ is also a column-stochastic matrix.

Proof: A matrix is defined as *column-stochastic* if all its entries are non-negative and the sum of the entries in each column is equal to 1.

Let M be the matrix defined by the equation:

$$M = (1-m)A + mS$$

where S is the teleportation matrix with entries $S_{ij} = 1/n$ for all i, j.

2.3.1 Non-negativity

Since A is column-stochastic, $A_{ij} \geq 0$. The matrix S contains only positive entries ($1/n > 0$). Given that $0 \leq m \leq 1$, both coefficients $(1-m)$ and m are non-negative. Therefore, every entry $M_{ij} = (1-m)A_{ij} + mS_{ij}$ is a sum of non-negative terms, so $M_{ij} \geq 0$.

2.3.2 Sum of Columns

We need to prove that for any column j, the sum of the elements $\sum_{i=1}^{n} M_{ij}$ is exactly 1.

Let's expand the summation for the generic column j:

$$\sum_{i=1}^{n} M_{ij} = \sum_{i=1}^{n} ((1-m)A_{ij} + mS_{ij})$$

Using the linearity of the summation:

$$\sum_{i=1}^{n} M_{ij} = (1-m)\sum_{i=1}^{n} A_{ij} + m\sum_{i=1}^{n} S_{ij}$$

Now we evaluate the two summations separately:

- Since A is column-stochastic by hypothesis, the sum of its j-th column is 1:

$$\sum_{i=1}^{n} A_{ij} = 1$$

- Since S is the matrix where every entry is $1/n$:

$$\sum_{i=1}^{n} S_{ij} = \sum_{i=1}^{n} \frac{1}{n} = n \cdot \frac{1}{n} = 1$$

Substituting these values back into the equation:

$$\sum_{i=1}^{n} M_{ij} = (1-m)(1) + m(1)$$

$$\sum_{i=1}^{n} M_{ij} = 1 - m + m$$

$$\sum_{i=1}^{n} M_{ij} = 1$$

In conclusion, since M contains non-negative entries and the sum of each column is 1, M is a column-stochastic matrix.

2.4 Exercise 8

Text: Show that the product of two column-stochastic matrices is also column-stochastic.

Proof: Let A and B be two $n \times n$ column-stochastic matrices. By definition, a matrix is column-stochastic if:

1. All its entries are non-negative: $A_{ij} \geq 0$ and $B_{ij} \geq 0$ for all i, j.
2. The entries in each column sum to one: $\sum_{i=1}^{n} A_{ij} = 1$ and $\sum_{i=1}^{n} B_{ij} = 1$ for all j.

Let C be the product matrix $C = AB$. The entry C_{ij} is defined by the standard matrix multiplication formula:

$$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

We must prove that C satisfies the two conditions of a column-stochastic matrix.

2.4.1 Non-negativity

Since every element A_{ik} and B_{kj} is non-negative, their product $A_{ik}B_{kj}$ is non-negative. The sum of non-negative terms is also non-negative. Thus, $C_{ij} \geq 0$ for all i, j.

2.4.2 Column Sums

We calculate the sum of the elements in the j-th column of C:

$$\sum_{i=1}^{n} C_{ij} = \sum_{i=1}^{n} \left(\sum_{k=1}^{n} A_{ik} B_{kj} \right)$$

We can swap the order of summation (finite sums):

$$\sum_{i=1}^{n} C_{ij} = \sum_{k=1}^{n} B_{kj} \left(\sum_{i=1}^{n} A_{ik} \right)$$

Since A is a column-stochastic matrix, the sum of its k-th column is 1 ($\sum_{i=1}^{n} A_{ik} = 1$). We substitute this into the equation:

$$\sum_{i=1}^{n} C_{ij} = \sum_{k=1}^{n} B_{kj}(1) = \sum_{k=1}^{n} B_{kj}$$

Since B is also a column-stochastic matrix, the sum of its j-th column is 1:

$$\sum_{k=1}^{n} B_{kj} = 1$$

Therefore:

$$\sum_{i=1}^{n} C_{ij} = 1$$

In conclusion: since $C = AB$ has non-negative entries and each column sums to 1, C is a column-stochastic matrix.

2.4.3 Relevance to PageRank

This result is crucial for the Power Method used in PageRank. The algorithm computes $x_k = Mx_{k-1} = M^k x_0$. Since M is column-stochastic, this exercise proves that any power M^k is also column-stochastic. This guarantees that the total probability of the system is conserved at every iteration step, ensuring that the surfer is always somewhere on the web and never disappears.

2.5 Exercise 9

Text: Show that a page with no backlinks is given importance score m/n by formula: $\mathbf{x}^{(k)} = (1-m)A\mathbf{x}^{(k-1)} + m\mathbf{s}$.

Proof: Let us recall the PageRank equation derived from the Google Matrix $M = (1-m)A + mS$:

$$\mathbf{x} = (1-m)A\mathbf{x} + mS\mathbf{x}$$

The component-wise form for the importance score of a specific page k, denoted as x_k, is:

$$x_k = (1-m)\sum_{j=1}^{n} A_{kj}x_j + m\sum_{j=1}^{n} S_{kj}x_j$$

where:

- n is the total number of pages in the web.
- A_{kj} is the entry of the link matrix (probability of moving from j to k).
- $S_{kj} = 1/n$ is the entry of the teleportation matrix.
- $\mathbf{x}$ is the probability vector, so $\sum_{j=1}^{n} x_j = 1$.

2.5.1 Step 1: apply the no backlinks condition

If page k has no backlinks, it means no page j links to k. In terms of the link matrix A, this implies that the k-th **row** consists entirely of zeros:

$$A_{kj} = 0 \quad \forall j \in \{1, \ldots, n\}$$

Substituting this into the first summation term:

$$\sum_{j=1}^{n} A_{kj}x_j = \sum_{j=1}^{n} 0 \cdot x_j = 0$$

2.5.2 Step 2: simplify the teleportation term

Now we analyze the second term involving matrix S. Since $S_{kj} = 1/n$ for all j:

$$m\sum_{j=1}^{n} S_{kj}x_j = m\sum_{j=1}^{n} \frac{1}{n}x_j$$

We can factor out the constant $1/n$:

$$= \frac{m}{n}\sum_{j=1}^{n} x_j$$

Since $\mathbf{x}$ is a normalized PageRank vector, the sum of all importance scores must be 1 ($\sum x_j = 1$). Therefore:

$$= \frac{m}{n} \cdot 1 = \frac{m}{n}$$

In conclusion, combining the two parts:

$$x_k = (1-m)(0) + \frac{m}{n}$$

$$x_k = \frac{m}{n}$$

Thus, a page with no backlinks receives exactly the minimum score $\frac{m}{n}$.

2.6 Exercise 12

The primary objective of this exercise is to demonstrate how the **Google Matrix** M resolves the issue of nodes with no backlinks. In the standard eigenvector formulation based on the raw link matrix A, such pages are assigned a null importance score, effectively making them invisible to the ranking system. This exercise illustrates how the damping factor introduces a strictly positive lower bound for importance scores.

The network topology from the previous exercise is expanded by introducing a sixth node (Page 6). This new page is characterized by the following connectivity properties:

- **Outgoing Links:** It links to every other page in the network.
- **Incoming Links:** It receives no links from any other page (out-degree > 0, in-degree $= 0$).

We compute and compare the rankings derived from the raw link matrix A ($m = 0$) and the Google Matrix M ($m = 0.15$).

2.6.1 Comparative Analysis

The analysis reveals a fundamental divergence between the purely topological model and the random surfer model:

- **Model with Link Matrix A ($m = 0$):**
  In the raw matrix formulation, the entry A_{ij} represents the probability of moving from j to i. Since no page j links to Page 6, the 6th row of the matrix A consists entirely of zeros:

  $$A_{6j} = 0 \quad \forall j \in \{1, \dots, n\}$$

  Consequently, the eigenvector equation component for this page becomes $x_6 = \sum A_{6j} x_j = 0$. The page is assigned an importance score of zero, implying it has no relevance within the network.

- **Model with Google Matrix M ($m = 0.15$):**
  Even though the contribution from the link structure $(1 - m)A$ remains zero for Page 6, the teleportation component mS ensures a non-zero probability flow.
  For any page i, the contribution from S is $\sum_j \frac{m}{N} x_j = \frac{m}{N}$. Therefore, for a 6-page web with $m = 0.15$, Page 6 receives exactly the minimum baseline score:

  $$x_6 = \frac{0.15}{6} = 0.025$$

2.6.2 Theoretical Significance

Since the teleportation matrix S is strictly positive, the Google Matrix M satisfies the condition $M_{ij} > 0$ for all i, j. According to **Proposition 2**, the dominant eigenvector of a positive matrix must have strictly positive components.

This guarantees two critical properties for the search algorithm:

1. **Global visibility:** no page, regardless of its isolation or lack of backlinks, is ever assigned a score of zero.

2. **Minimum importance floor:** the term m/N acts as a universal lower bound for PageRank scores, ensuring that every indexed page retains a minimal probability of being visited by the random surfer.

```
A_ex12_dense = np.array([
    [0.0, 0.0, 0.5, 0.5, 0.0, 0.2],
    [1/3, 0.0, 0.0, 0.0, 0.0, 0.2],
    [1/3, 0.5, 0.0, 0.5, 1.0, 0.2],
    [1/3, 0.5, 0.0, 0.0, 0.0, 0.2],
    [0.0, 0.0, 0.5, 0.0, 0.0, 0.2],
    [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]  # Row Pg 6 (No backlinks ->
    Backlinks = 0)
])

A_ex12_sparse = sparse.csr_matrix(A_ex12_dense)
scores_A, iters_A = calculate_pagerank(A_ex12_sparse, 6, False, m
    =0.0, max_iter=1000)
```

Listing 2.1: Exercise 12 code

```
==========================================
 EXERCISE 12: PAGE WITHOUT BACKLINKS
==========================================

--- CASE A: Ranking with m=0.0 ---
Page 3: 0.3673
Page 1: 0.2449
Page 5: 0.1837
Page 4: 0.1224
Page 2: 0.0816
Page 6: 0.0000

--- CASE B: Ranking with m=0.15 ---
Page 3: 0.3402
Page 1: 0.2312
Page 5: 0.1738
Page 4: 0.1350
Page 2: 0.0948
Page 6: 0.0250

Check Page 6 (No Backlink):
With m=0.00 -> Score: 0.000000 (Should be 0)
With m=0.15 -> Score: 0.025000 (Minimum guaranteed = 0.15/6 = 0.025)
```

Figure 2.3: Results of Exercise 12

2.7 Exercise 13

2.7.1 Objective and Scenario

The objective of this exercise is to prove that the **Google Matrix** M enables a unique and globally comparable ranking even when the web structure consists of disconnected components. The analysis relies strictly on the spectral theory of positive matrices to show how the teleportation factor resolves the ambiguity inherent in disconnected graphs.
So, it proves that the damping factor m is not merely a numerical trick for convergence, but a structural necessity for defining a global ranking.

Consider a web W partitioned into two disjoint subwebs, W_1 and W_2, such that there are no links from W_1 to W_2 and vice versa. In graph theory terms, the graph is not **strongly connected**. This structure reflects the reality of the global web, where distinct clusters of websites may exist without direct reciprocal links.
By indexing the pages of W_1 first and those of W_2 second, the link matrix A

assumes a **block-diagonal form**:

$$A = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \tag{2.5}$$

where A_1 and A_2 are the column-stochastic matrices representing the internal links of the two subwebs. The off-diagonal zero blocks mathematically represent the total isolation between the two communities.

2.7.2 Analysis of Matrix M: uniqueness via positivity

To resolve this ambiguity, the algorithm introduces the Google Matrix $M = (1-m)A + mS$.
Physically, the teleportation matrix S (with entries $1/n$) acts as a perturbation that fills the zero blocks of A, modeling a random surfer who can jump between isolated subwebs.
Since the teleportation matrix S has strictly positive entries $(1/n)$ and $m > 0$, the matrix M becomes a **strictly positive matrix** ($M_{ij} > 0$ for all i, j). According to **Proposition 2** and **Lemma 3.2**, a positive column-stochastic matrix possesses a unique eigenvector with eigenvalue $\lambda = 1$ and strictly positive components.
Thus, the introduction of the teleportation factor mS forces the existence of a unique ranking vector $\mathbf{x}$. This allows the algorithm to objectively compare the importance of pages in W_1 against pages in W_2, merging them into a single global scale.

2.7.3 Computational Verification

To validate these theoretical findings, we performed a numerical simulation using a web of $N = 5$ pages partitioned into two disconnected subwebs: Group A (Pages 1, 2 and 3) and Group B (Pages 4, and 5).
Using the standard teleportation factor $m = 0.15$, the algorithm successfully converged to a unique importance vector (Fig. 2.4), resolving the ambiguity of the block-diagonal link matrix.

```
A_ex13_dense = np.array([
    [0.0,0.0,0.5,       0.0,0.0],
    [1.0,0.0,0.5,       0.0,0.0],
    [0.0,1.0,0.0,       0.0,0.0],
    [0.0,0.0,0.0,       0.0,1.0],
    [0.0,0.0,0.0,       1.0,0.0]
])

A_ex13_sparse = sparse.csr_matrix(A_ex13_dense)
scores_13, iters_13 = calculate_pagerank(A_ex13_sparse, 5, False, m
    =0.15)
```

Listing 2.2: Exercise 13 code

The computational results (presented in Figure 2.4) confirm the egalitarian nature of the teleportation term: the Matrix M distributes importance across disconnected components. The total probability mass is divided between the groups exactly proportional to their size: Group A (3 pages) received a total aggregated score of 0.6000 (matching the expected 3/5 ratio), while Group B (2 pages) received 0.4000 (matching 2/5).
However, the internal distribution differs based on topology.
Group B, which forms a symmetric cycle, assigned identical scores to its members (Pages 4 and 5 both at 0.2000). Conversely, Group A exhibits an internal hierarchy due to its asymmetric link structure, with Page 2 emerging as the most authoritative node (0.2384) compared to Page 1 (0.1289). This confirms that, while teleportation unifies the scale based on component size, the algorithm preserves the specific structural ranking within each subweb.

```
==========================================
 EXERCISE 13: DISCONNECTED SUBNETWORKS
==========================================
Matrix A (5x5) constructed with two disconnected components.
Group A: Pages 1-2-3 (Internal links only)
Group B: Pages 4-5   (Ping-pong link)

--- Ranking Calculation (m=0.15) ---
Page 2 (Group A): 0.2384
Page 3 (Group A): 0.2327
Page 4 (Group B): 0.2000
Page 5 (Group B): 0.2000
Page 1 (Group A): 0.1289

--- Probability Distribution Analysis ---
Total Probability Group A (3 pages): 0.6000
Total Probability Group B (2 pages): 0.4000
Approximate Expected Ratio (3/5 vs 2/5):
A (Expected ~0.60): 0.6000
B (Expected ~0.40): 0.4000

[SUCCESS] The PageRank distribution matches the theoretical size ratio!
```

Figure 2.4: Rank obtained for Exercise 13

2.8 Exercise 14

2.8.1 Objective of the Exercise

The primary objective of this exercise is to analyze the computational efficiency of the PageRank algorithm by monitoring the convergence of the **Power Method**. It demonstrates how the error between the iterative vector x_k and the steady-state vector q decays over time, and how this decay rate is strictly governed by the spectral properties of the Google Matrix M.

2.8.2 Scenario and Error Measurement

Starting with an initial probability vector x_0, the Power Method generates a sequence of approximations through the iteration $x_k = Mx_{k-1}$. The convergence is quantified using the L_1 **norm** of the error:

$$e_k = \|x_k - q\|_1 \tag{2.6}$$

By computing the ratio of successive errors, $\frac{e_k}{e_{k-1}}$, for $k = 1, 5, 10, 50$, the exercise shows that the sequence stabilizes. This ratio provides the asymptotic rate of convergence, indicating how much "error" is eliminated in each step of the algorithm.

2.8.3 Analysis: The Second Eigenvalue and the Ergodicity Coefficient

The exercise verifies two critical theoretical bounds discussed in the paper:

- **The Spectral Gap:** The asymptotic convergence rate is determined by the absolute value of the second largest eigenvalue, $|\lambda_2|$. For the matrix M under examination, the ratio $\frac{e_k}{e_{k-1}}$ stabilizes around a specific value (e.g., ≈ 0.61 or ≈ 0.85 depending on the specific link structure), confirming that $|\lambda_2|$ dictates the rhythm of the algorithm.
- **The Ergodicity Coefficient $c(M)$:** The exercise requires calculating $c(M) = \max_j |1-2\min_i M_{ij}|$. This coefficient provides a guaranteed upper bound for the convergence rate, ensuring that $|\lambda_2| \leq c(M)$. By showing that the actual convergence ratio is less than or equal to $c(M)$, the exercise experimentally validates the matrix M as a **contraction mapping** in the space of probability vectors.

2.8.4 Technical Significance: Scalability and the Damping Factor

This analysis provides the mathematical justification for why Google can process billions of pages efficiently. The convergence speed is intrinsically linked to the **damping factor** m. Because $|\lambda_2| \leq 1 - m$, setting $m = 0.15$ guarantees that $|\lambda_2| \leq 0.85$, ensuring that the error decreases by at least 15% at every iteration. This explains why the system reaches a stable state in relatively few iterations, making PageRank computationally tractable for the global scale of the World Wide Web.

2.8.5 Results

```
Iter (k)   | L1 Error (e_k)       | Ratio (e_k/e_k-1)
------------------------------------------------------------
1          | 0.7819626529         | 0.5125208082
5          | 0.0068714107         | 0.3227085474
10         | 0.0003043717         | 0.5245287480
50         | 0.0000000000         | 0.6114691824
------------------------------------------------------------
THEORETICAL RESULTS:
Second Dominant Eigenvalue |lambda_2|:   0.611269
Ergodicity Coefficient c(M):  0.940000

Check Proposition 4: |lambda_2| <= c(M) is True
Check Convergence: Final ratio (0.6115) tends to |lambda_2| (0.6113)
```

Figure 2.5

2.9 Exercise 15

2.9.1 Objective

The primary objective of this exercise is to provide a rigorous mathematical foundation for the convergence behavior observed in the Power Method. By applying **spectral decomposition**, we analyze how any initial starting vector eventually converges to the unique PageRank vector $\mathbf{q}$, and demonstrate why the speed of this convergence is strictly governed by the second largest eigenvalue, $|\lambda_2|$.

2.9.2 Eigenvector Representation and Proofs

To analyze the convergence, we consider an $n \times n$ positive column-stochastic matrix $\mathbf{M}$ that is **diagonalizable**. This ensures the matrix possesses a complete set of n linearly independent eigenvectors forming a basis for $\mathbb{R}^n$.

Let the set of eigenvectors be $\{\mathbf{q}, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{n-1}\}$. Any initial probability vector $\mathbf{x}_0$ can be uniquely expressed as a linear combination:

$$\mathbf{x}_0 = a\mathbf{q} + \sum_{k=1}^{n-1} b_k \mathbf{v}_k \tag{2.7}$$

In this decomposition, each term has a specific meaning:

- **The Signal ($\mathbf{q}$):** The steady-state vector corresponding to $\lambda_1 = 1$.
- **The Noise ($\mathbf{v}_k$):** The eigenvectors associated with $|\lambda| < 1$, representing initial errors or deviations.

Proof that $a = 1$ and components of $\mathbf{v}_k$ sum to 0

The exercise requires showing that $a = 1$ and that the entries of each deviation vector $\mathbf{v}_k$ sum to zero. We utilize the property that column-stochastic matrices preserve the sum of components (probability mass). Let $\mathbf{1}^T = [1, 1, \ldots, 1]$. Since $\mathbf{x}_0$ and $\mathbf{q}$ are probability vectors, $\mathbf{1}^T\mathbf{x}_0 = 1$ and $\mathbf{1}^T\mathbf{q} = 1$. Multiplying the basis equation by $\mathbf{1}^T$:

$$\mathbf{1}^T\mathbf{x}_0 = a(\mathbf{1}^T\mathbf{q}) + \sum_{k=1}^{n-1} b_k(\mathbf{1}^T\mathbf{v}_k)$$
$$1 = a(1) + \sum_{k=1}^{n-1} b_k(\mathbf{1}^T\mathbf{v}_k)$$

For this to hold for *any* arbitrary starting vector $\mathbf{x}_0$ (where coefficients b_k vary), it implies:

1. $a = 1$: The coefficient of the steady state is always 1.
2. $\mathbf{1}^T\mathbf{v}_k = 0$: The sum of components of any eigenvector $\mathbf{v}_k$ (for $\lambda \neq 1$) is zero. This confirms that $\mathbf{v}_k$ represents "zero-sum" noise (shifts in probability that do not change the total).

2.9.3 Dynamics of Convergence: Why λ_2 dominates

When we apply the matrix $\mathbf{M}$ iteratively for k steps, we obtain:

$$\text{iteration 1:} \quad \mathbf{M}\mathbf{x}_0 = \mathbf{q} + b_1\lambda_2\mathbf{v}_1 + \ldots$$
$$\text{iteration k:} \quad \mathbf{x}_k = \mathbf{M}^k\mathbf{x}_0 = \mathbf{q} + \sum_{k=1}^{n-1} b_k\lambda_{k+1}^k\mathbf{v}_k$$

The error vector $\mathbf{e}_k = \mathbf{x}_k - \mathbf{q}$ is strictly composed of the "noise" terms:

$$\mathbf{e}_k = b_1\lambda_2^k\mathbf{v}_1 + b_2\lambda_3^k\mathbf{v}_2 + \cdots + b_{n-1}\lambda_n^k\mathbf{v}_{n-1} \tag{2.8}$$

Since eigenvalues are ordered $1 > |\lambda_2| \geq |\lambda_3| \ldots$, the components associated with smaller eigenvalues decay exponentially faster. To visualize this, let us assume $|\lambda_2| \approx 0.61$ and a smaller eigenvalue $|\lambda_3| \approx 0.30$. After just 10 iterations:

- The λ_3 component shrinks to: $0.30^{10} \approx 0.000005$ (vanished).
- The λ_2 component remains at: $0.61^{10} \approx 0.007$ (still significant).

Consequently, for large k, the error becomes dominated by the single term associated with the second eigenvalue:

$$\mathbf{e}_k \approx b_1\lambda_2^k\mathbf{v}_1 \tag{2.9}$$

2.9.4 Evaluation of the Limit

Using the dominant term approximation derived above, we can now evaluate the limit requested by the exercise:

$$\lim_{k\to\infty} \frac{\|\mathbf{M}^k\mathbf{x}_0 - \mathbf{q}\|_1}{\|\mathbf{M}^{k-1}\mathbf{x}_0 - \mathbf{q}\|_1} = \lim_{k\to\infty} \frac{\|\mathbf{e}_k\|_1}{\|\mathbf{e}_{k-1}\|_1} \tag{2.10}$$

Substituting $\mathbf{e}_k \approx b_1\lambda_2^k\mathbf{v}_1$:

$$\begin{aligned}\lim_{k\to\infty} \frac{\|b_1\lambda_2^k\mathbf{v}_1\|_1}{\|b_1\lambda_2^{k-1}\mathbf{v}_1\|_1} &= \lim_{k\to\infty} \frac{|\lambda_2|^k \cdot \|b_1\mathbf{v}_1\|_1}{|\lambda_2|^{k-1} \cdot \|b_1\mathbf{v}_1\|_1} \\ &= \frac{|\lambda_2|^k}{|\lambda_2|^{k-1}} = |\lambda_2|\end{aligned}$$

This rigorous derivation confirms that the asymptotic convergence rate is $|\lambda_2|$.

2.9.5 Graphical Confirmation

The figure below (Fig. 2.6) illustrates this convergence on a semi-logarithmic scale.

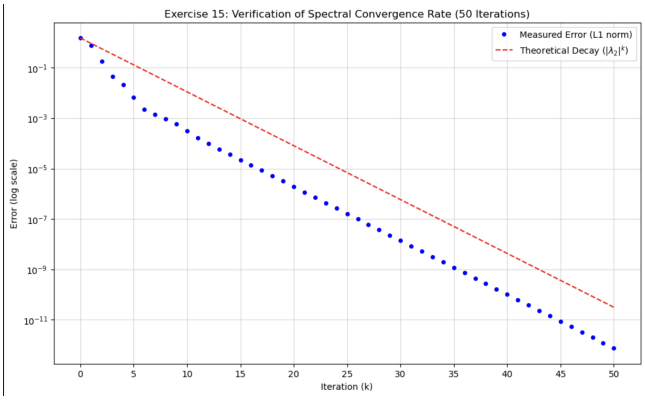

Figure 2.6: Convergence of the error term in the Power Method.

The plot confirms our theoretical analysis:

- **Initial Transient Phase (Fast Drop):** In the first few iterations, the error drops faster than predicted by $|\lambda_2|$. This corresponds to our calculation above where terms like 0.30^{10} vanish almost immediately.
- **Asymptotic Parallelism:** Once the smaller components vanish, the error curve becomes perfectly parallel to the red dashed line ($|\lambda_2|^k$). This geometric parallelism validates that the ratio of successive errors has converged to the constant $|\lambda_2|$.

2.10 Exercise 17

The choice of m represents a fundamental trade-off in the design of the PageRank algorithm. The Google Matrix is defined as a weighted average:

$$M = (1 - m)A + mS \tag{2.11}$$

The parameter m balances two opposing forces: the "democratic" nature of the web structure (represented by A) against the mathematical stability and convergence speed provided by the teleportation matrix (S).

2.10.1 Effect on Computation Time (Convergence Speed)

As established in the analysis of the Power Method, the convergence rate is strictly governed by the magnitude of the **second largest eigenvalue** $|\lambda_2|$. The error at step k decays according to the ratio $|\lambda_2|^k$.

For the Google Matrix M, the paper implies a direct relationship between the damping factor and the spectral gap:

$$|\lambda_2| \approx 1 - m \tag{2.12}$$

This relationship dictates the computational cost:

- **Small m (e.g., $m = 0.01$):** the second eigenvalue is large ($|\lambda_2| \approx 0.99$). The error decays very slowly (only 1% per iteration). This would require thousands of iterations to reach convergence, which is computationally prohibitive for a web-scale graph.
- **Large m (e.g., $m = 0.50$):** the second eigenvalue is small ($|\lambda_2| \approx 0.50$). The error is halved at every step, leading to extremely fast convergence.

2.10.2 Effect on rankings (Quality and "Democracy")

If speed were the only factor, we would choose a large m. However, increasing m degrades the quality of the information. The term $(1 - m)$ represents the **weight of the true web topology**: it reflects the principle of the web where hyperlinks act as votes conferred by human authors.

- **High fidelity ($m \to 0$):** the ranking is determined almost exclusively by actual links. However, without the damping factor, the system suffers from critical topological issues: **Rank Sinks** trap probability mass, **disconnected components** lead to non-unique rankings, and pages with **no backlinks** receive a score of zero (invisibility).
- **High noise ($m \to 1$):** as m increases, the matrix is increasingly dominated by S, which distributes probability uniformly ($1/n$).
  If we were to choose a high value like $m = 0.50$, the algorithm would derive 50% of a page's score from random teleportation rather than its actual backlinks. This would flatten the ranking vector, reducing the distinction between authoritative pages and obscure ones, effectively destroying the relevance of the search results.

2.10.3 Conclusion: the choice of $m = 0.15$

The standard value of $m = 0.15$ is the industry-standard compromise that optimally resolves this trade-off. It is often cited as a "threshold" value because it maintains a specific physical property of navigation, derived from the statistical behavior of the random surfer.

1. **The Path Length threshold:** the expected number of steps (clicks) a random surfer performs before teleporting is governed by the probability of continuing the chain, which is $(1 - m)$.
   Mathematically, the number of steps k until the first teleportation follows a geometric distribution with success probability $p = m$. The expected value (mean) of this distribution is calculated as the sum of the probabilities of the surfer surviving at least k steps. This forms a geometric series:

   $$E[\text{steps}] = \sum_{k=0}^{\infty} P(\text{surviving } k \text{ steps}) = \sum_{k=0}^{\infty} (1 - m)^k$$

   This summation corresponds to the famous **geometric series formula** $\sum_{k=0}^{\infty} r^k$. For $|r| < 1$, it converges as:

   $$\sum_{k=0}^{\infty} r^k = \frac{1}{1 - r}$$

   So, since $0 < 1 - m < 1$, substituting our ratio $r = 1 - m$ into the formula, we obtain the exact relationship:

   $$E[\text{steps}] = \frac{1}{1 - (1 - m)} = \frac{1}{m}$$

   This derivation proves why the choice of m is critical for the depth of exploration:

   - With $m = 0.15$, the average path is $1/0.15 \approx \mathbf{6.66}$ clicks. This depth allows the algorithm to propagate voting authority through roughly 6 layers of the network, capturing the deep structure of the web.
   - If we increased m to just 0.30, the average path would drop to $1/0.30 \approx \mathbf{3.33}$ clicks. This represents a "short-sighted" exploration: the ranking would become too local, failing to distinguish global authorities that are reachable only via longer paths.

2. **Structural Integrity:** by setting $1 - m = 0.85$, the algorithm preserves the vast majority of the true link structure. The "noise" introduced (15%) is sufficient to ensure strong connectivity (resolving disconnected components) and guarantee a minimum score for every page (fixing the zero-score problem for pages with no backlinks), without overpowering the human-generated votes.

3. **Acceptable Speed ($|\lambda_2| \approx 0.85$):** while not instant, a decay factor of 0.85 ensures that the error is reduced by roughly 15% at each iteration. This allows the algorithm to converge to a high degree of precision in a reasonable number of steps, making the computation feasible even for billions of pages.