DEN: Linear algebra – numerical view

Algorithm (GEM: Gauss elimination method for reducing a full rank matrix to upper-triangular form, with partial pivoting)

Given: matrix $C = (c_{i,j})_{i,j=1}^{n,m}$ of real numbers, where $m \geq n$.
0. Set $k = 1$.
1. If $c_{i,k} = 0$ for all $i = k, \dots, n$, then $\mathrm{rank}(A) < n$. The algorithm fails and stops.
Otherwise, do the "pivoting": Among the rows $i = k, \dots, n$, choose the row k' that has the largest possible value of $|c_{i,k}|$. If $k' \neq k$, exchange rows k and k'.
The (new) number $c_{k,k}$ is called a "pivot". Continue with step **2**.
2. For $i = k+1, \dots, n$ do the following:
Let $l_{i,k} = \frac{c_{i,k}}{c_{k,k}}$, set $c_{i,k} = 0$ and for $j > k$ compute $c_{i,j} = c_{i,j} - l_{i,k} c_{k,j}$.
If $k < n$, increase k by one and go back to step **1**.
Otherwise the algorithm stops.
The **output** is the matrix $(c_{i,j})_{i,j=1}^{n,m}$.
Note: The output has the form

$$\begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,n-1} & c_{1,n} & \cdots & c_{1,m} \\ 0 & c_{2,2} & \cdots & c_{2,n-1} & c_{2,n} & \cdots & c_{2,m} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & c_{n-1,n-1} & c_{n-1,n} & \cdots & c_{n-1,m} \\ 0 & 0 & \cdots & 0 & c_{n,n} & \cdots & c_{n,m} \end{pmatrix}.$$

Fact.
Gaussian elimination applied to an $n \times (n+c)$ matrix requires at most $\frac{2}{3}n^3 + \left(c - \frac{1}{2}\right)n^2 - \left(c + \frac{1}{6}\right)n$ operations.

Corollary.
The computational complexity of GEM when reducing an $n \times n$ matrix or an $n \times (n+1)$ matrix is

$$\tfrac{2}{3}n^3 + O(n^2).$$

Algorithm (GJM: Gauss-Jordan elimination method for reducing a full-rank matrix to an extended diagonal matrix, with partial pivoting)

Given: matrix $C = (c_{i,j})_{i,j=1}^{n,m}$ of real numbers, where $m \geq n$.
0. Set $k = 1$.
1. If $c_{i,k} = 0$ for all $i = k, \dots, n$, then $\mathrm{rank}(A) < n$. The algorithm fails and stops.
Otherwise, do the "pivoting": Among the rows $i = k, \dots, n$, choose the row k' that has the largest possible value of $|c_{i,k}|$. If $k' \neq k$, exchange rows k and k'.
The (new) number $c_{k,k}$ is called a "pivot". Continue with step **2**.
2. For $j > k$ let $c_{k,j} = \frac{c_{k,j}}{c_{k,k}}$. Set $c_{k,k} = 1$.
For all $i \neq k$ do: For $j > k$ compute $c_{i,j} = c_{i,j} - c_{i,k}c_{k,j}$, set $c_{i,k} = 0$
If $k < n$, increase k by one and go back to step **1**.
Otherwise the algorithm stops.
The **output** is the matrix $(c_{i,j})_{i,j=1}^{n,m}$.
Note: The output has the form

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 & c_{1,n+1} & \cdots & c_{1,m} \\ 0 & 1 & \cdots & 0 & 0 & c_{2,n+1} & \cdots & c_{2,m} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 & c_{n-1,n+1} & \cdots & c_{n-1,m} \\ 0 & 0 & \cdots & 0 & 1 & c_{n,n+1} & \cdots & c_{n,m} \end{pmatrix}.$$

Fact.
Gauss-Jordan elimination applied to an $n \times (n+c)$ matrix requires at most $n^3 + \frac{1}{2}(4c-3)n^2 + \frac{1}{2}(1-2c)n$ operations.

Corollary.
The computational complexity of GJM when reducing an $n \times n$ matrix or an $n \times (n+1)$ matrix is

$$n^3 + O(n^2).$$

Algorithm (Solving an upper triangular system by back substitution)

Given: a system $U\vec{x} = \vec{d}$, where the matrix U is square, regular and upper triangular.
1. We find the solution $\vec{x}_0$ using the formulas

$$\begin{aligned} x_n &= \frac{d_n}{u_{n,n}} \\ x_{n-1} &= \frac{d_{n-1} - u_{n-1,n}x_n}{u_{n-1,n-1}} \\ x_{n-2} &= \frac{d_{n-2} - u_{n-2,n}x_n - u_{n-2,n-1}x_{n-1}}{u_{n-2,n-2}} \\ &\vdots \\ x_1 &= \frac{d_1 - u_{1,n}x_n - u_{1,n-1}x_{n-1} - \cdots - u_{1,2}x_2}{u_{1,1}} \end{aligned}$$

In general,

$$x_k = \frac{1}{u_{k,k}}\Big(d_k - \sum_{i=k+1}^{n} u_{k,i}x_i\Big).$$

Algorithm (Solving a lower triangular system by forward substitution)

Given: A system $L\vec{x} = \vec{d}$, where the matrix L is square, regular and lower triangular.
1. We find the solution $\vec{x}_0$ using the formulas

$$\begin{aligned} x_1 &= \frac{d_1}{l_{1,1}} \\ x_2 &= \frac{d_2 - l_{2,1}x_1}{l_{2,2}} \\ x_3 &= \frac{d_3 - l_{3,1}x_1 - l_{3,2}x_2}{l_{3,3}} \\ &\vdots \\ x_n &= \frac{d_n - l_{n,1}x_1 - l_{n,2}x_2 - \cdots - l_{n,n-1}x_{n-1}}{l_{n,n}} \end{aligned}$$

In general,

$$x_k = \frac{1}{l_{k,k}}\Big(d_k - \sum_{i=1}^{k-1} u_{k,i}x_i\Big).$$

Fact.
Any system $A\vec{x} = \vec{b}$ whose regular matrix A is upper (resp. lower) triangular can be solved by back (resp. forward) substitution with computational complexity n^2.

Definition.
Consider a real $n \times n$ matrix A. We say that $n \times n$ matrices L, U are an **LU decomposition** or **LU factorization** of A if U is an upper-triangular matrix, L is a lower-triangular matrix whose diagonal entries are all 1, and $A = LU$.

Theorem.
Let A be a real $n \times n$ matrix. If its rank is k and its first k leading principal minors are non-zero, then it has an LU decomposition.

Fact.
If GEM without pivoting applied to A yields a triangle matrix U, then A has an LU decomposition and U is the right matrix for it.

Algorithm (LU decomposition of a matrix)

Given: an $n \times n$ matrix $A = (a_{i,j})_{i,j=1}^{n}$ of real numbers.
0. Let $U = A$, set $k = 1$, $l = 1$.
1. If $u_{k,l} \neq 0$ then continue with step **2**. Otherwise:
If there is $i > k$ such that $u_{i,l} \neq 0$, then the given matrix does not have any LU decomposition. The algorithm fails and stops.
If $u_{k,l} = 0$ for all $i \geq k$, then go to step **3**.
2. For $i = k+1, \ldots, n$ do the following: Let $l_{i,k} = \frac{u_{i,l}}{u_{k,l}}$, set $u_{i,l} = 0$ and for $j > l$ compute $u_{i,j} = u_{i,j} - l_{i,k}u_{k,j}$.
Increase k by one and go to step **3**.
3. If $l < n$, increase l by one and go back to step **1**.
Otherwise the algorithm stops.
The **output** is matrices

$$L = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ l_{2,1} & 1 & \cdots & 0 & 0 \\ \vdots & & \vdots & & \vdots \\ l_{n-1,1} & l_{n-1,2} & \cdots & 1 & 0 \\ l_{n,1} & l_{n,2} & \cdots & l_{n,n-1} & 1 \end{pmatrix},$$

$$U = \begin{pmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,n-1} & u_{1,n} \\ 0 & u_{2,2} & \cdots & u_{2,n-1} & u_{2,n} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & u_{n-1,n-1} & u_{n-1,n} \\ 0 & 0 & \cdots & 0 & u_{n,n} \end{pmatrix}.$$

Fact.
For every square matrix A there exists a permutation matrix P, a lower-triangular matrix L and an upper-triangular matrix U such that $PA = LU$.

Algorithm (LUP decomposition of a matrix, with partial pivoting)

Given: an $n \times n$ matrix $A = (a_{i,j})_{i,j=1}^{n}$ of real numbers.
0. Let $U = A$ and $L = P = E_n$ (unit matrix). Set $k = 1$, $l = 1$.
1. If $u_{k,l} = 0$ for all $i \geq k$, then go to step **3**. Otherwise:
Among the rows $i = k, \ldots, n$, choose the row k' that has the largest possible value of $|u_{i,l}|$. If $k' > k$, then exchange rows k and k' in matrices U and P; in the matrix L exchange the first $k-1$ entries of rows k and k'.
Continue with step **2**.
2. For $i = k+1, \ldots, n$ do the following: Let $l_{i,k} = \frac{u_{i,l}}{u_{k,l}}$, set $u_{i,l} = 0$ and for $j > l$ compute $u_{i,j} = u_{i,j} - l_{i,k}u_{k,j}$.
Increase k by one and continue with step **3**.
3. If $l < n$, increase l by one and go back to step **1.**
Otherwise the algorithm stops.
The **output** is the matrices P, L, U.

Algorithm (solving systems of linear equations using LUP factorization)

Given: a system $A\vec{x} = \vec{b}$, where A is a regular square matrix.
1. Find the LUP factorization $LU = AP$.
2. Using the forward substitution, solve the system $L\vec{y} = P\vec{b}$ for $\vec{y}$.
3. Using the back substitution, solve the system $U\vec{x} = \vec{y}$ for $\vec{x}$.

$$A \mapsto L, U, P \qquad (L|P\vec{b}) \mapsto \vec{y} \qquad (U|\vec{y}) \mapsto \vec{x}$$

Iterative improvement of solution:
1. Find a solution $\vec{x}$ of the system $A\vec{x} = \vec{b}$,
2. Determine the residuum $\vec{r} = \vec{b} - A\vec{x}$. Solve the system $A\vec{E}_x = \vec{r}$. If the error $\vec{E}_x$ is not sufficiently small, do the correction $\vec{x} := \vec{x} + \vec{E}_x$.

Traditional norms on $\mathbb{R}^n$:

$$\|\vec{x}\| = \sqrt{\sum_{k=1}^{n} |x_k|^2} \qquad \text{(Euclidean norm)},$$

$$\|\vec{x}\|_\infty = \max_{k=1,\dots,n} |x_k| \qquad \text{(max norm)},$$

$$\|\vec{x}\|_1 = \sum_{k=1}^{n} |x_k| \qquad \text{(sum norm)}.$$

Definition.
Let V be a vector space. A mapping $\|\!|\cdot|\!\|\colon V \mapsto \mathbb{R}$ is called a **norm** if it has the following properties:
- $\|\!|\vec{x}|\!\| \geq 0$ for all $\vec{x} \in \mathbb{R}^n$;
- $\|\!|\vec{x}|\!\| = 0$ if and only if $\vec{x} = \vec{0}$;
- $\|\!|c\vec{x}|\!\| = |c| \cdot \|\!|\vec{x}|\!\|$ for all $\vec{x} \in \mathbb{R}^n$ and $c \in \mathbb{R}$;
- $\|\!|\vec{x} + \vec{y}|\!\| \leq \|\!|\vec{x}|\!\| + \|\!|\vec{y}|\!\|$ for all $\vec{x}, \vec{y} \in \mathbb{R}^n$ (triangle inequality).

Definition.
Let V be a vector space of matrices. A mapping $\|\!|\cdot|\!\|\colon V \mapsto \mathbb{R}$ is called a **matrix norm** if it is a norm and also satisfies
- $\|\!|AB|\!\| \leq \|\!|A|\!\| \cdot \|\!|B|\!\|$ for all $A, B \in M_{n\times n}$.

Traditional matrix norms:

$$\|A\|_\infty = \|A\|_R = \max_{i=1,\dots,n} \sum_{j=1}^{n} |a_{i,j}| \qquad \text{(row-sum norm)},$$

$$\|A\|_1 = \|A\|_C = \max_{j=1,\dots,n} \sum_{i=1}^{n} |a_{i,j}| \qquad \text{(column-sum norm)},$$

$$\|A\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} |a_{i,j}|^2} \qquad \text{(Frobenius norm)}.$$

Definition.
Consider a norm $|||\vec{x}|||$ for vectors from $\mathbb{R}^n$ and a norm $|||A|||_M$ for matrices from $M_{n\times n}$. We say that these norms are **compatible** if $|||A\vec{x}||| \leq |||A|||_M \cdot |||\vec{x}|||$ for all $A \in M_{n\times n}$ and $\vec{x} \in \mathbb{R}^n$.

Definition.
We define the **spectral radius** of A as $\varrho(A) = \max_j(|\lambda_j|)$, where the maximum runs through all eigenvalues λ_j of A (including possible complex ones).

Theorem.
(i) Let $|||\cdot|||$ be a vector norm. The number defined for $A \in M_{n\times n}$ by the formula

$$|||A|||_M = \sup\Big\{\frac{|||A\vec{x}|||}{|||\vec{x}|||};\ \vec{x} \in \mathbb{R}^n \setminus \{\vec{0}\}\Big\} = \sup\{|||A\vec{x}|||;\ \vec{x} \in \mathbb{R}^n \wedge |||\vec{x}||| \leq 1\}$$

determines a matrix norm compatible with $|||\cdot|||$. We call it the matrix norm **induced** by $|||\cdot|||$.
(ii) Let $|||\cdot|||_M$ be a matrix norm. The number defined for $\vec{x} \in \mathbb{R}^n$ by the formula

$$|||\vec{x}||| = \left|\left|\left| \begin{pmatrix} x_1 & 0 & \dots & 0 \\ x_2 & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots \\ x_n & 0 & \dots & 0 \end{pmatrix} \right|\right|\right|_M$$

is a norm on $\mathbb{R}^n$ compatible with $|||\cdot|||_M$. It is called the norm induced by $|||\cdot|||_M$.

Definition.
For an $n \times n$ matrix A we define its **condition number** as $\mathrm{cond}(A) = |||A||| \cdot |||A^{-1}|||$.

Theorem.
Assume that vectors $\vec{x}_0, \vec{x}$ a $\vec{b}_0, \vec{b}$ are related by the formulas $A\vec{x}_0 = \vec{b}_0$ and $A\vec{x} = \vec{b}$. Denote $\vec{E}_x = \vec{x}_0 - \vec{x}$ and $\vec{E}_b = \vec{b}_0 - \vec{b}$. Then

$$\frac{|||\vec{E}_x|||}{|||\vec{x}|||} \leq \mathrm{cond}(A) \frac{|||\vec{E}_b|||}{|||\vec{b}|||}.$$

That is,

$$\varepsilon_x \leq \mathrm{cond}(A)\varepsilon_b.$$

Fact.
If a vector $\vec{b}$ is reliable to k significant digits, then the solution $\vec{x}$ of an equation $A\vec{x} = \vec{b}$ is reliable to $k - \log_{10}(\mathrm{cond}(A))$ significant digits.

Theorem.
Assume that matrixec A_0, A and vectors $\vec{x}_0, \vec{x}$ and $\vec{b}_0, \vec{b}$ are related by the formulas $A_0\vec{x}_0 = \vec{b}_0$ and $A\vec{x} = \vec{b}$. Denote $E_A = A_0 - A$, $\vec{E}_x = \vec{x}_0 - \vec{x}$ and $\vec{E}_b = \vec{b}_0 - \vec{b}$. Then

$$\varepsilon_x \leq \mathrm{cond}(A)\Big(\varepsilon_b + \varepsilon_A \cdot \frac{|||\vec{x}|||}{|||\vec{x}_0|||}\Big).$$

Theorem.
If a matrix B satisfies $|||B|||_M < 1$ for some compatible matrix norm, then the corresponding iterative method $\vec{x}_{k+1} = B\vec{x}_k + \vec{c}$ converges to $\vec{x}_f$ for arbitrary choice of $\vec{x}_0$ and we have

$$|||\vec{x}_f - \vec{x}_{k+1}||| \leq |||B|||_M |||\vec{x}_f - \vec{x}_k|||, \qquad |||\vec{x}_f - \vec{x}_{k+1}||| \leq \frac{|||B|||_M}{1 - |||B|||_M} |||\vec{x}_{k+1} - \vec{x}_k|||.$$

Fact.
Consider an $n \times n$ matrix B.
The inequality $\varrho(B) \leq |||B|||$ is true for all induced matrix norms.
Conversely, for every $\varepsilon > 0$ there is an induced matrix norm $|||\cdot|||$ such that $|||B||| - \varepsilon < \varrho(B)$.

Theorem.
An iterative method $\vec{x}_{k+1} = B\vec{x}_k + \vec{c}$ converges if and only if $\varrho(B) < 1$.

Algorithm (JIM: Jacobi iteration method)

Given: a system $A\vec{x} = \vec{b}$ of linear equations and tolerance ε.
0. Choose arbitrary initial vector $\vec{x}_0$. Set $k = 0$.
1. Compute

$$(\vec{x}_{k+1})_i = -\frac{1}{a_{i,i}}\Big(\sum_{j=1}^{i-1} a_{i,j}(\vec{x}_k)_j + \sum_{j=i+1}^{n} a_{i,j}(\vec{x}_k)_j\Big) + \frac{b_i}{a_{i,i}}.$$

If $\|\vec{x}_{k+1} - \vec{x}_k\|_\infty \geq \varepsilon$, increase k by one and go back to step **1**.

Algorithm (GSM: Gauss-Seidel iteration)

Given: a system $A\vec{x} = \vec{b}$ of linear equations and tolerance ε.
0. Choose arbitrary initial vector $\vec{x}_0$. Set $k = 0$.
1. Compute

$$(\vec{x}_{k+1})_i = -\frac{1}{a_{i,i}}\Big(\sum_{j=1}^{i-1} a_{i,j}(\vec{x}_{k+1})_j + \sum_{j=i+1}^{n} a_{i,j}(\vec{x}_k)_j\Big) + \frac{b_i}{a_{i,i}}.$$

If $\|\vec{x}_{k+1} - \vec{x}_k\|_\infty \geq \varepsilon$, increase k by one and go back to step **1**.

Definition.
Consider an $n \times n$ matrix A.
We say that A is **strictly diagonally dominant** if

$$|a_{i,i}| > \sum_{j \neq i} |a_{i,j}|$$

for all $i = 1, \ldots, n$.
We say that A is **positive definite** if $\vec{x}^T A\vec{x} > 0$ for all non-zero vectors $\vec{x} \in \mathbb{R}^n$.

Theorem.
If A is strictly diagonally dominant, then both JIM and GSM converge for arbitrary choice of initial vector.
If A is symmetric and positive definite, then GSM converges for arbitrary choice of initial vector.

Algorithm (Relaxation (SOR, Successive OverRelaxation method))

Given: a system $A\vec{x} = \vec{b}$ of linear equations, tolerance ε, and parameter of relaxation λ.
0. Choose arbitrary initial vector $\vec{x}_0$. Set $k = 0$.
1. Compute

$$(\vec{x}_{k+1})_i = (1-\lambda)(\vec{x}_k)_i - \frac{\lambda}{a_{i,i}}\Big(\sum_{j=1}^{i-1} a_{i,j}(\vec{x}_{k+1})_j + \sum_{j=j+1}^{n} a_{i,j}(\vec{x}_k)_j\Big) + \frac{\lambda b_i}{a_{i,i}}.$$

If $\|\vec{x}_{k+1} - \vec{x}_k\|_\infty \geq \varepsilon$, increase k by one and go back to step **1**.

Theorem. (Ostrovsky)
Let A be a symmetric $n \times n$ matrix with positive diagonal entries. Then $\varrho(B_\lambda) < 1$ if and only if A is positive definite and $0 < \lambda < 2$.

Definition.
Consider an $n \times n$ matrix A. A number λ is called an **eigenvalue** of A if there is a non-zero vector $\vec{x}$ such that $A\vec{x} = \lambda\vec{x}$. Vectors $\vec{x}$ with this property are then called **eigenvectors** associated with λ.

Fact.
If a real $n \times n$ matrix is symmetric, then its eigenvalues are real.
Moreover, there exists a basis of $\mathbb{R}^n$ composed of its eigenvectors.

Algorithm (power method for finding the largest eigenvalue and an associated eigenvector)
Given: an $n \times n$ matrix A and tolerance $\varepsilon > 0$.
0. Choose arbitrary initial vector $\vec{x}_0$ and set $k = 0$.
1. Compute

$$\vec{x}_{k+1} = \frac{A\vec{x}_k}{\|A\vec{x}_k\|_\infty}.$$

If $\|\vec{x}_{k+1} - \vec{x}_k\|_\infty \geq \varepsilon$, increase k by one and go back to step **1**.

Definition.
For an $n \times n$ matrix A and a vector $\vec{x} \in \mathbb{R}^n$ we define their **Rayleigh quotient** by the formula

$$\frac{\vec{x}^T A\vec{x}}{\vec{x}^T\vec{x}}.$$

Fact.
If $\vec{x}$ is an eigenvector of a matrix A, then $\frac{\vec{x}^T A\vec{x}}{\vec{x}^T\vec{x}}$ is equal to the associated eigenvalue.

Algorithm (power method for finding the largest eigenvalue and an associated eigenvector)

Given: an $n \times n$ matrix A and tolerance $\varepsilon > 0$.

0. Choose arbitrary initial vector $\vec{x}_0$.

If complex eigenvalues are needed, choose a vector with non-zero imaginary part in each component.

Set $k = 0$, let $l_0 = \frac{\vec{x}_0^* A \vec{x}_0}{\vec{x}_0^* \vec{x}_0}$.

1. Compute

$$\vec{x}_{k+1} = \frac{|l_k| \cdot A\vec{x}_k}{l_k \cdot \|A\vec{x}_k\|_\infty}, \qquad l_{k+1} = \frac{\vec{x}_{k+1}^* A \vec{x}_{k+1}}{\vec{x}_{k+1}^* \vec{x}_{k+1}}.$$

If $\|\vec{x}_{k+1} - \vec{x}_k\|_\infty \geq \varepsilon$, increase k by one and go back to step **1**.

Alternative: Use Euclidean norm, that is,

$$\vec{x}_{k+1} = \frac{|\lambda_k| \cdot A\vec{x}_k}{\lambda_k \cdot \|A\vec{x}_k\|_2}, \qquad l_{k+1} = \vec{x}_{k+1}^* A \vec{x}_{k+1}.$$

Theorem.

Let A be an $n \times n$ matrix with eigenvalues λ_j, where $|\lambda_1| > |\lambda_j|$ for $j \geq 2$. Let $M = \max_{j \geq 2} |\lambda_j|$.

For $\vec{x}_0$ let $\{l_k\}$, $\{\vec{x}_k\}$ be sequences generated by the power method. Then $\{l_k\}$ converges to λ_1 and $\{\vec{x}_k\}$ converges to some eigenvector $\vec{v}$ associated with λ_1.

Moreover, $|\lambda_1 - l_k| = O\big(\big[\frac{M}{|\lambda_1|}\big]^{2k}\big)$ and $\|\vec{v}_1 - \vec{x}_k\| = O\big(\big[\frac{M}{|\lambda_1|}\big]^{k}\big)$.

Algorithm (inverse power method for finding eigenvalue and an associated eigenvector)

Given: an $n \times n$ matrix A, arbitrary real number μ that is not an eigenvalue of A and tolerance $\varepsilon > 0$.
1. Set $B = (A - \mu E_n)^{-1}$.
2. Applying power iteration to the matrix B, find the largest eigenvalue λ_B of the matrix B and an associated eigenvector $\vec{v}$.
Output: The number $\mu + \frac{1}{\lambda_B}$ is the eigenvalue of A closest to μ with eigenvector $\vec{v}$.

Algorithm (deflation method for finding eigenvalue and an associated eigenvector)

Given: a symmetric $n \times n$ matrix A and its eigenvalues λ_1 through λ_N with eigenvectors $\vec{v}_1$ through $\vec{v}_N$ satisfying $\|v_i\| = 1$ and tolerance $\varepsilon > 0$.
1. Set up the matrix $B = A - \sum_{j=1}^{N} \lambda_j \vec{v}_j \cdot \vec{v}_j^T$.
2. Apply power iteration to the matrix B and obtain λ_{N+1}, $\vec{x}_{N+1}$.