University of Washington
AMATH 301 Spring 2017
Instructor: Dr. King-Fai Li

Week 3

1. Fixed-point iteration

Purpose: Find the roots of $F(\mathbf{x}) = 0$

1. Rewrite $F(\mathbf{x}) = 0$ into the form of $\mathbf{x} = G(\mathbf{x})$. i.e. if $\mathbf{x}$ is a solution, so will be $G(\mathbf{x})$.
2. Make an initial guess of $\mathbf{x}$, call it $\mathbf{x}^{(0)}$.
   a. For linear problems, if convergence is assured, any $\mathbf{x}^{(0)}$ will do. Try $\mathbf{x}^{(0)} = \mathbf{0}$.
3. For self-consistency, $G\left(\mathbf{x}^{(0)}\right)$ should be close to the root. So take $\mathbf{x}^{(1)} = G\left(\mathbf{x}^{(0)}\right)$.
4. Iterate $\mathbf{x}^{(2)} = G\left(\mathbf{x}^{(1)}\right)$, $\mathbf{x}^{(3)} = G\left(\mathbf{x}^{(2)}\right)$, … until

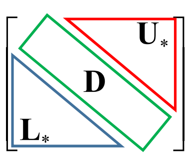
$$\underbrace{\max\left|\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}\right| = \max_k \left\{\left|x_k^{(n+1)} - x_k^{(n)}\right|\right\}}_{\texttt{norm}\left(\texttt{x}^{(\texttt{n+1})}\texttt{-x}^{(\texttt{n})}\texttt{,inf}\right)} < \varepsilon$$

Fixed-point iteration is applicable to linear and non-linear equations. For our purposes, we only consider linear systems $F(\mathbf{x}) = \mathbf{Ax} - \mathbf{b}$. Two forms of $G(\mathbf{x})$ are considered below.

2. Jacobi's Method

Purpose: Solve $\begin{bmatrix} & & \\ & \mathbf{A} & \\ & & \end{bmatrix}_{m\times m} \begin{bmatrix} \\ \mathbf{x} \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \mathbf{b} \\ \\ \end{bmatrix}$ for $\mathbf{x}$ using iterative methods.

Decompose $\mathbf{A}$ into three components:



$$\begin{bmatrix} & & \\ & \mathbf{A} & \\ & & \end{bmatrix} = [\text{diagram}] = \underbrace{\mathbf{L}_*}_{\texttt{tril(A-D)}} + \underbrace{\mathbf{D}}_{\texttt{diag(diag(A))}} + \underbrace{\mathbf{U}_*}_{\texttt{triu(A-D)}}$$

$\mathbf{D}$ is the diagonal part of $\mathbf{A}$, $\mathbf{L}_*$ is the strictly lower triangular part and $\mathbf{U}_*$ is the strictly upper triangular part.

Jacobi's method keeps only the diagonal part on the left hand side:

$$\mathbf{D}\mathbf{x} = \mathbf{b} - \left(\mathbf{L}_* + \mathbf{U}_*\right)\mathbf{x}$$
$$\mathbf{x} = \mathbf{D} \backslash \left[\mathbf{b} - \left(\mathbf{L}_* + \mathbf{U}_*\right)\mathbf{x}\right]$$

Thus the iteration equation is

$$\mathbf{x}^{(n+1)} = \mathbf{D} \backslash \left[\mathbf{b} - \left(\mathbf{L}_* + \mathbf{U}_*\right)\mathbf{x}^{(n)}\right]$$

Physical Meaning: All new estimates $x_j^{(n+1)}$ are based on old estimates $x_{k \neq j}^{(n)}$.

In the video, Prof. Brunton defined $\mathbf{T} = \mathbf{L}_* + \mathbf{U}_*$.

In Matlab,

```
x = zeros(m,1);
iter = 0;
eps = 1e-4;
while iter <= 100   % To avoid infinite loops
  xold = x;
  iter = iter + 1;
  x = D\(b-(Ls+Us)*xold);
  if norm(x-xold,inf)<eps
    break
  end
end
```

3. Gauss-Seidel Method

Gauss-Seidel method keeps the diagonal and the strictly lower triangular parts on the left hand side:

$$\left(\mathbf{L}_* + \mathbf{D}\right)\mathbf{x} = \mathbf{b} - \mathbf{U}_*\mathbf{x}$$
$$\mathbf{x}^{(n+1)} = \left(\mathbf{L}_* + \mathbf{D}\right) \backslash \left[\mathbf{b} - \mathbf{U}_*\mathbf{x}^{(n)}\right]$$

Physical Meaning: New estimates $x_j^{(n+1)}$ are progressively used whenever they are available.

In the video, Prof. Brunton defined $\mathbf{S} = \mathbf{L}_* + \mathbf{D}$.

In Matlab,

```
x = zeros(m,1);
iter = 0;
eps = 1e-4;
while iter <= 100   % To avoid infinite loops
  xold = x;
  iter = iter + 1;
  x = (Ls+D)\(b-Us*xold);
  if norm(x-xold,inf)<eps
    break
  end
end
```

4. Theory of convergence

One must check the convergence of the above methods before proceeding.

A sufficient (but not necessary) convergence condition for Jacobi's method:

Strictly diagonally dominance: The sum of absolute values of row-wise off-diagonal elements is less than the diagonal element of the same row: $\sum_{j \neq i} |a_{ij}| < |a_{ii}|$.

However, this condition is not necessary. i.e. There can be cases where **A** is not strictly diagonally dominant but the Jacobi method still converges.

e.g. From Prof. Kutz's book

$$\mathbf{A} = \begin{pmatrix} -2 & 1 & 5 \\ 4 & -8 & 1 \\ 4 & -1 & 1 \end{pmatrix} \rightarrow \begin{array}{l} \text{row 1: } |-2| < |1| + |5| = 6 \\ \text{row 2: } |-8| > |4| + |1| = 5 \\ \text{row 3: } |1| < |4| + |-1| = 5 \end{array}$$

is not strictly diagonally dominant. But swapping the first and third equations will do:

$$\mathbf{A} = \begin{pmatrix} 4 & -1 & 1 \\ 4 & -8 & 1 \\ -2 & 1 & 5 \end{pmatrix} \rightarrow \begin{array}{l} \text{row 1: } |4| > |-1| + |1| = 2 \\ \text{row 2: } |-8| > |4| + |1| = 5 \\ \text{row 3: } |5| > |2| + |1| = 3 \end{array}$$

In this case, the Jacobi method is guaranteed to be convergent.

General theory of convergence:

The above two iterative methods can be recast as

$$\mathbf{x}^{(n)} = \mathbf{M}\mathbf{x}^{(n-1)} + \mathbf{c} = \mathbf{M}\left(\mathbf{M}\mathbf{x}^{(n-2)} + \mathbf{c}\right) + \mathbf{c} = \cdots = \mathbf{M}^n \mathbf{x}^{(0)} + \left(\sum_{k=0}^{n-1} \mathbf{M}^k\right)\mathbf{c}$$

Therefore, we need $\lim_{n\to\infty} \mathbf{M}^n = 0$ for convergence.

Since $\mathbf{M}^n = \underbrace{\mathbf{V}\mathbf{S}^n\mathbf{V}^{-1}}_{[\mathbf{V},\mathbf{S}]=eig(\mathbf{M})}$, where $\mathbf{S} = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{pmatrix}$ is a diagonal matrix containing the eigenvalues and $\mathbf{V} = \begin{pmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \\ | & | & & | \end{pmatrix}$ has the eigenvectors as columns,

$$\left\|\mathbf{M}^n\right\| = \left\|\mathbf{V}\mathbf{S}^n\mathbf{V}^{-1}\right\| \le \left\|\mathbf{V}\right\|\left\|\mathbf{S}^n\right\|\left\|\mathbf{V}^{-1}\right\| = \left\|\mathbf{S}^n\right\| \le \left(\max_k \left|\lambda_k\right|\right)^n$$

Thus if $\max|\lambda| < 1$, then the iteration converges.

For Jacobi's Method, $\mathbf{M} = -\mathbf{D} \backslash \left(\mathbf{L}_* + \mathbf{U}_*\right)$

For Gauss-Seidel Method, $\mathbf{M} = -\left(\mathbf{L}_* + \mathbf{D}\right) \backslash \mathbf{U}_*$

5. Conjugate Gradient Method (Optional)

Given a symmetric **A, m**inimize $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{x}^T\mathbf{b}$ with respect to **x** such that at the minimum, $f'(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b} = 0$. Therefore **x** is the solution of $\mathbf{A}\mathbf{x} - \mathbf{b}$ at the minimum point.

In Matlab, `bicg` uses a similar algorithm solve for $\mathbf{x}$ with non-symmetric **A**.