1. Fixed point algorithm

a) For a fixed point iteration $|g'(x)| < 1$

so

$$g(x) = ax + b \Rightarrow g'(x) = a$$

$$|g'(x)| = |a| < 1$$

If this is true then $g(x)$ has a unique solution at $\bar{x}$,
$g(\bar{x}) = \bar{x}$

$$g(\bar{x}) = a\bar{x} + b = \bar{x} \Rightarrow \bar{x} = \frac{b}{1-a}$$

Suppose $|x_k - \bar{x}| = |g(x_{k-1}) - g(\bar{x})|$, for a fixed point iteration

$$x_{k+1} = g(x) \quad , \quad g(\bar{x}) = \bar{x} = \frac{b}{1-a}$$

$$|x_k - \bar{x}| = |a x_{k-1} + b - \bar{x}|$$

$$|x_k - \bar{x}| = \left|a x_{k-1} + b - \frac{b}{1-a}\right| = \left|a x_{k-1} - \frac{ab}{1-a}\right|$$

$$= \left|a\left(x_{k-1} - \frac{b}{1-a}\right)\right| = |a(x_{k-1} - \bar{x})|$$

$$|x_k - \bar{x}| = |a(x_{k-1} - \bar{x})|$$

Using triangular inequalities.

$$|x_k - \bar{x}| \leq |a|\,|x_{k-1} - \bar{x}| = |a|^2 |x_{k-2} - \bar{x}|$$

$$\vdots$$

$$|x_k - \bar{x}| \leq |a|^k |x_0 - \bar{x}|$$

As $k \to \infty$, since $|a| < 1$, then $|a|^k |x_0 - \bar{x}| \to 0$

therefore $|x_k - \bar{x}| \to 0$, hence the fixed point Iteration Converges to $\bar{x}$.

$$x_k = \bar{x} = \frac{b}{1-a}$$

b) Using the Intermidiate value Theorem, It Can be stated that

$$\frac{g(x_k) - g(\bar{x})}{x_k - \bar{x}} = g'(\xi)$$

$$g(x_k) - g(\bar{x}) = g'(\xi)\,(x_k - \bar{x})$$

for a fixed point Scheme

$$g(x_k) = x_{k+1}$$

$$g(\bar{x}) = \bar{x}$$

$$g'(\xi) = a$$

So

$$x_{k+1} - \bar{x} = a\,(x_k - \bar{x}) = a^2(x_{k-2} - \bar{x})$$

$$\vdots$$

$$x_{k+1} - \bar{x} = a^k(x_1 - \bar{x})$$

$$x_{k+1} - \bar{x} = a^{k+1}(x_0 - \bar{x})$$

but $x_{k+1} - \bar{x} = e_{k+1}$, $x_0 - \bar{x} = e_0$

$$e_{k+1} = a^{k+1} e_0$$

Hence satisfied.

c) $e_{k+1} \approx \frac{a}{a-1}(x_{k+1} - x_k)$

$e_{k+1} = x_{k+1} - \bar{x}$

Subtracting and adding x_k on the ~~f~~ left hand Side, we obtain

$e_{k+1} = x_{k+1} - \bar{x} - x_k + x_k$

$e_{k+1} = x_{k+1} - x_k + x_k - \bar{x}$

we have

$e_{k+1} = a e_k \Rightarrow e_k = \frac{a}{e_{k+1}}$ and $e_k = x_k - \bar{x}$

therefore

~~$e_{k+1} = x_{k+1} + x_k - \bar{x}$ becomes~~

~~$e_{k+1} \approx x_{k+1} - x_k$~~

$e_{k+1} = x_{k+1} - x_k + x_k - \bar{x}$ becomes,

$e_{k+1} \approx x_{k+1} - x_k + e_k$

$e_{k+1} \approx x_{k+1} - x_k + \frac{a}{e_{k+1}}$

$e_{k+1} \approx \frac{a}{a-1}(x_{k+1} - x_k)$

---

d) $e_{k+1} \approx \frac{a}{a-1}(x_{k+1} - x_k)$

we know that $g(x_k) = x_{k+1} = a x_k + b$

$e_{k+1} \approx \frac{a}{a-1}\left(g(x_k) - x_k\right) = \frac{a}{a-1}\left(a x_k + b - x_k\right)$

$e_{k+1} \approx a x_k + a\left(\frac{b}{a-1}\right)$, but $\bar{x} = b/1-a$

$e_{k+1} = a x_k - a\bar{x} = a(x_k - \bar{x})$

take $x_k - \bar{x} = e_k$

$e_{k+1} = a e_k = a^2 e_{k-1}$

$e_{k+1} = a^3 e_{k-2}$

$\vdots \qquad \vdots$

$e_{k+1} = a^k e_{1}$

$e_{k+1} = a^{k+1} e_0$

hence exactly equal to the true error.

e). from $g(x) = \frac{1}{10}x + 1$, $\Rightarrow a = 1/10$, $b = 1$

tolerance, $\epsilon_q = 10^{-8}$

Using $|e_{k+1}| \leq \epsilon_q$, but $e_{k+1} = a^{k+1} e_0$

$|a^{k+1} e_0| \leq \epsilon_q$

Using logarithms, $\log|a^{k+1}| + \log|e_0| \leq \log \epsilon_q$

$(k+1)\log|a| + \log|e_0| \leq \log \epsilon_q$

$(k+1)\log|a| \leq \log \epsilon_q - \log|e_0|$

$k\log|a| \leq \log\epsilon_q - \log|e_0| - \log|a|$

but $a = 1/10 = 10^{-1}$

$-k\log 10 \leq \log\epsilon_q - (\log|e_0| + \log 10^{-1})$

$k \geq \frac{8\log 10 + \log|e_0| - \log 10}{\log 10} = 8 + \log|e_0| - 1$

$k \geq 7 + \log|e_0|$

Hence it requires atleast 7 iterations, and this depends on $\log|e_0|$

2| Steffensen's Method.

$$x_{k+1} = x_k - \frac{(g(x_k) - x_k)^2}{g(g(x_k)) - 2g(x_k) + x_k}$$

a) Show ~~that~~ analytically that for any $g(x)$, the Iteration Used in Steffensen's Method Satisfies

$$\lim_{x \to \bar{x}} \left(x - \frac{(g(x_k) - x_k)^2}{g(g(x_k)) - 2g(x_k) + x_k} \right) = \bar{x}$$

Where $\bar{x}$ Satisfies $g(\bar{x}) = \bar{x}$

Using Lahopital's Rule.

$$\lim_{x \to \bar{x}} \left(x - \frac{2(g'(x) - 1)(g(x) - \bar{x})}{g'(x) g'(g(x)) - 2g'(x) + 1} \right)$$

but $g(\bar{x}) = \bar{x}$

$$= \bar{x} - \frac{2(g'(\bar{x}) - 1)(\bar{x} - \bar{x})}{g'(\bar{x}) g'(g(\bar{x})) - 2g'(\bar{x}) + 1}$$

$$= \bar{x} - 0$$

$$= \bar{x}$$

b) Show analytically that Steffensen's Iteration Converges in one Step for the fixed point problem.

$$g(x) = ax + b = x \quad , \text{for } |a| < 1$$

from $x_{k+1} = x_k - \dfrac{(g(x) - x)^2}{g(g(x)) - 2g(x) + x}$

at $k=0$

$$x_1 = x_0 - \frac{(g(x_0) - x_0)^2}{g(g(x_0)) - 2g(x_0) + x_0}$$

x_1 ~~A~~ $g(x_0) = ax_0 + b$

$$x_1 = x_0 - \frac{(ax_0 + b - x_0)^2}{a(ax_0+b) + b - 2(ax_0+b) + x_0}$$

$$x_1 = x_0 - \frac{(a-1)(x_0) + b}{(a-1)}$$

$$x_1 = x_0 - x_0 + \frac{b}{a-1} = x_0 - x_0 - b/a-1$$

$$x_1 = \frac{b}{1-a} = \bar{x}$$

$$\underline{\underline{x_1 = \bar{x}}}$$

hence it converges in one step.

No. 3 LU Decomposition

$$E_{21} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A = \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

a) Show that

$$E_{41}E_{31}E_{21} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{bmatrix}$$

$$E_{41}E_{31}E_{21} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$E_{41}E_{31}E_{21} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{bmatrix}$$

b) Choose multipliers l_{ij} so that applying $E_{41}E_{31}E_{21}$ to A zeros out the entries below a_{11}.

$$E_{41}E_{31}E_{21}A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

Choose
$l_{21} = -1$
$l_{31} = 2$
$l_{41} = -3$

$$E_{41}E_{31}E_{21}A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ -2 & 0 & 1 & 0 \\ 3 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

$$= \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 10 & 4 & -9 \\ 0 & -16 & -11 & 18 \end{bmatrix}$$

---

c) Show that the inverse of $E_{41}E_{31}E_{21}$ is

$$(E_{41}E_{31}E_{21})^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & 0 & 1 & 0 \\ l_{41} & 0 & 0 & 1 \end{bmatrix}$$

$$(E_{41}E_{31}E_{21})^{-1} = \frac{1}{\det(E_{41}E_{31}E_{21})}\left(\text{Adjoint}(E_{41}E_{31}E_{21})\right)$$

$$\det(E_{41}E_{31}E_{21}) = \begin{vmatrix} 1 & 0 & 0 & 0 \\ -l_{21} & 1 & 0 & 0 \\ -l_{31} & 0 & 1 & 0 \\ -l_{41} & 0 & 0 & 1 \end{vmatrix} = -1\begin{vmatrix} 1 & 0 & 0 \\ -l_{41} & 1 & 0 \\ -l_{31} & 0 & 1 \end{vmatrix} = \underline{\underline{1}}$$

the Adjoint $(E_{41}E_{31}E_{21})$ is the transpose of the Cofactor Matrix $(E_{41}E_{31}E_{21})$

$$\text{Cofactor Matrix} = \begin{bmatrix} 1 & l_{21} & l_{31} & l_{41} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Adjoint}(E_{41}E_{31}E_{21}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & 0 & 1 & 0 \\ l_{41} & 0 & 0 & 1 \end{bmatrix}$$

$$|E_{41}E_{31}E_{21}| = \frac{1}{1}\begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & 0 & 1 & 0 \\ l_{41} & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & 0 & 1 & 0 \\ l_{41} & 0 & 0 & 1 \end{bmatrix}$$

3d) <u>for U</u>

$$E_{41}E_{31}E_{21}A = U_1$$
$$(E_{42}E_{32})U_1 = U_2$$
$$E_{43}U_2 = U$$

therefore

$$\underline{U = E_{43}E_{42}E_{32}E_{41}E_{31}E_{21}A}$$

Hence U is obtained by computing the product $U = E_{43}U_2 = E_{43}E_{42}E_{31}U_1$ and $U_1 = E_{41}E_{31}E_{21}A$

<u>For L</u>

$$(E_{41}E_{31}E_{21})^{-1} = L_1$$
$$(E_{42}E_{32})^{-1}L_1 = L_2$$
$$(E_{43})^{-1}L_2 = L$$

So

$$L = (E_{43}E_{42}E_{32}E_{41}E_{31}E_{21})^{-1}$$

To compute this inverse, we can assume

$$(AB)^{-1} = A^{-1}B^{-1}$$

therefore

$$L = (E_{43}E_{42}E_{32}E_{41}E_{31}E_{21})^{-1} = (E_{43}E_{42}E_{32})^{-1}(E_{41}E_{31}E_{21})^{-1}$$

but

$$(E_{41}E_{31}E_{21})^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & 0 & 1 & 0 \\ l_{41} & 0 & 0 & 1 \end{bmatrix}$$

$$(E_{43}E_{42}E_{32})^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & L_{32} & 1 & 0 \\ 0 & L_{32}L_{43}+L_{42} & L_{43} & 1 \end{bmatrix}$$

$$L = (E_{43}E_{42}E_{32})^{-1}(E_{41}E_{31}E_{21})^{-1}$$

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & L_{32} & 0 & 0 \\ 0 & L_{32}L_{43}+L_{42} & L_{43} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ L_{21} & 1 & 0 & 0 \\ L_{31} & 0 & 1 & 0 \\ L_{41} & 0 & 0 & 1 \end{bmatrix}$$

Let $l_{21} = a$ $l_{32} = d$

$l_{31} = b$ $L_{42} = e$

$L_{41} = c$ $L_{43} = f$

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a & 1 & 0 & 0 \\ da+b & d & 1 & 0 \\ fb+a(df+e)+c & df+e & f & 1 \end{bmatrix}$$

And $U = E_{43}E_{42}E_{32}E_{41}E_{31}E_{21}A$

4a) Find the LU decomposition of the matrix in (3)

$$A = \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

$$A = LU$$

where

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & l_{32} & 1 & 0 \\ l_{41} & l_{42} & l_{43} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix}$$

Using Gaussian Elimination method on A, we obtain an upper triangular matrix U, as follows.

take $l_{21} = -1$

$$\begin{array}{c} R_1 \\ R_2 \\ R_3 \\ R_4 \end{array} \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix} \xrightarrow{R_2 \leftarrow R_2 + R_1} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 5 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

take $l_{31} = -2$ ~~In this case $L_{21} = -1$~~

$$\begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 5 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix} \xrightarrow{R_3 \leftarrow R_3 + 2R_1} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 5 \\ 0 & -18 & -4 & -1 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

~~In this case $L_{31} = -2$~~

$$\begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & -18 & -4 & -1 \\ -9 & 5 & -5 & 12 \end{bmatrix} \xrightarrow{R_4 \leftarrow R_4 + 3R_1} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & -18 & -4 & -1 \\ 0 & -16 & -11 & 18 \end{bmatrix}$$

In this case we took $L_{41} = -3$

~~for L_{32}~~ take $L_{32} = 9$

$$\begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & -18 & -4 & -1 \\ 0 & -16 & -11 & 18 \end{bmatrix} \xrightarrow{R_3 \leftarrow R_3 - 9R_2} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & -16 & -11 & 18 \end{bmatrix}$$

take $L_{42} = 8$

$$\begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & -16 & -11 & 18 \end{bmatrix} \xrightarrow{R_4 \leftarrow R_4 - 8R_2} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & 0 & -3 & 2 \end{bmatrix}$$

take $L_{43} = -3/5$

$$\begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & 0 & -3 & 2 \end{bmatrix} \xrightarrow{R_4 \leftarrow R_4 + \frac{3}{5}R_3} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & 0 & 0 & -\frac{47}{5} \end{bmatrix}$$

Hence we Obtain, U,

$$U = \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & 0 & 0 & -47/5 \end{bmatrix}$$

Corresponding to, L,

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -2 & 9 & 1 & 0 \\ -3 & 8 & -3/5 & 1 \end{bmatrix}$$

So $A = LU$

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -2 & 9 & 1 & 0 \\ -3 & 8 & -3/5 & 1 \end{bmatrix} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & 0 & 0 & -\frac{47}{5} \end{bmatrix}$$

So

$$LU = A$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -2 & 9 & 1 & 0 \\ -3 & 8 & -3/5 & 1 \end{bmatrix} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & 5 & -19 \\ 0 & 0 & 0 & -49/5 \end{bmatrix} = \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix}$$

hence $LU = A$

5. Jacobi Method

Assume that the following 2x2 matrix A is strictly diagonally dominant

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

a) Show that $\rho(I - \Delta^{-1}A) < 1$

take $\Delta = \begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix}$

$$\Delta^{-1} = \begin{bmatrix} 1/a_{11} & 0 \\ 0 & 1/a_{22} \end{bmatrix}$$

$$\Delta^{-1}A = \begin{bmatrix} 1/a_{11} & 0 \\ 0 & 1/a_{22} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} 1 & a_{12}/a_{11} \\ a_{21}/a_{22} & 1 \end{bmatrix}$$

then

$$I - \Delta^{-1}A = \begin{pmatrix} 0 & a_{12}/a_{11} \\ a_{21}/a_{22} & 0 \end{pmatrix}$$

the spectral radius, ρ, is the maximum absolute value of the eigen values, so to compute the eigen values we use the characteristic equation.

$$|(I - \Delta^{-1}A) - \lambda I| = 0$$

$$\begin{vmatrix} -\lambda & a_{12}/a_{11} \\ a_{21}/a_{22} & -\lambda \end{vmatrix} = 0 \Rightarrow \lambda^2 = \frac{a_{21}a_{12}}{a_{22}a_{11}}$$

Since a_{22} and a_{11} are diagonally dominant row entries then $a_{22}a_{11} > a_{21}a_{12}$, therefore

$$\lambda^2 = \frac{a_{21}a_{12}}{a_{22}a_{11}} < 1$$

$$\lambda = \pm\sqrt{\frac{a_{21}\,a_{12}}{a_{22}\,a_{11}}}$$

ρ is the largest $|\lambda|$, so

$$|\lambda| = \left|\sqrt{\frac{a_{21}a_{12}}{a_{22}a_{11}}}\right| < 1$$

~~So th~~ So the spectral radius $\rho(I - D^{-1}A) < 1$

Jacobi Iteration will Converge, since for Jacobi, we take $M = D$, and we have shown that

$\rho(I - M^{-1}A) < 1$, hence the Iteration Converges

since according for ~~Converge~~ an Iteration Converges If and only if $\rho(I - M^{-1}A) < 1$

NO. 7

$$X_{k+1} = (I - M^{-1}A)X_k + M^{-1}b.$$

a) Show that if $M=A$, the iteration Converges in one step.

If $M=A$

$$X_{k+1} = (I - A^{-1}A)X_k + A^{-1}b$$

but $A^{-1}A = I$

$$X_{k+1} = (I - I)X_k + A^{-1}b$$

$$X_{k+1} = A^{-1}b$$

but $A\bar{X} = b \Rightarrow \bar{X} = A^{-1}b$

therefore $X_{k+1} = \bar{X}$, hence Converges in one step

b) Show that $Ae_k = -r_k$, where $r_k = b - AX_k$,

$e_k = x_k - \bar{x}$,

and ~~$\bar{x}$~~ solves $AX = b$ exactly

$$Ae_k = A(X_k - \bar{x}).$$

$$Ae_k = AX_k - A\bar{x}$$

we know that $A\bar{x} = b$

$$Ae_k = AX_k - b$$

$$Ae_k = -r_k$$ since $r_k = b - AX_k$.

c) Show that the iteration for the error e_k is given by

$$e_{k+1} = (I - M^{-1}A)\, e_k$$

from

$$x_{k+1} = (I - M^{-1}A)\, x_k + M^{-1}b$$

Subtract $\bar{x}$ from both sides

$$x_{k+1} - \bar{x} = (I - M^{-1}A)\, x_k - \bar{x} + M^{-1}b$$

we know that $x_{k+1} - \bar{x} = e_{k+1}$

$$e_{k+1} = (I - M^{-1}A)\, x_k - \bar{x} + M^{-1}b$$

note $A\bar{x} = b$.

$$e_{k+1} = (I - M^{-1}A)\, x_k - \bar{x} + M^{-1}A\bar{x}$$

$$e_{k+1} = (I - M^{-1}A)\, x_k - (I - M^{-1}A)\bar{x}$$

$$e_{k+1} = [I - M^{-1}A]\,(x_k - \bar{x})$$

but $x_k - \bar{x} = e_k$

$$e_{k+1} = (I - M^{-1}A)\, e_k$$

---

d) Show that

$$\|e_k\| \le \|I - M^{-1}A\|^k \|e_0\|$$

from $e_{k+1} = (I - M^{-1}A)\, e_k$

$$e_{k+1} = (I - M^{-1}A)(I - M^{-1}A)\, e_{k-1}$$

$$e_{k+1} = (I - M^{-1}A)^2 e_{k-1}$$

$$e_k = (I - M^{-1}A)^2 e_{k-2}$$

$$e_k = (I - M^{-1}A)^3 e_{k-3}$$

$$\vdots \qquad \vdots$$

$$e_k = (I - M^{-1}A)^k e_0$$

take the Modulus both sides, and then apply the triangular inequality

$$\|e_k\| = \|(I - M^{-1}A)^k e_0\|$$

$$\|e_k\| \leq \|(I - M^{-1}A)\|^k \|e_0\|$$

---

8) Show that

$$K \geq \frac{\log(\varepsilon)}{\log\|I - M^{-1}A\|}$$

from $\|e_k\| \leq \varepsilon$

~~but $\|e_k\| \leq \|(I - M^{-1}A)\|^k \|e_0\|$~~

but $e_k = (I - M^{-1}A)^k e_0$

$$\|(I - M^{-1}A)^k\| \|e_0\| \leq \varepsilon$$

taking log both sides we obtain

$$k \log\|I - M^{-1}A\| + \log\|e_0\| \leq \log \varepsilon$$

$$k \log \| I - M^{-1}A \| \leq \log \varepsilon - \log \| e_0 \|$$

Since $\log \| I - M^{-1}A \| < 0$, then

$$k \geq \frac{\log \varepsilon - \log \| e_0 \|}{\log \| I - M^{-1}A \|}$$

$$k \geq \frac{\log (\varepsilon / \| e_0 \|)}{\log \| I - M^{-1}A \|}$$

f) Since $\rho(I - M^{-1}A)$ is the largest absolute value of the eigen values of $(I - M^{-1}A)$, and the number of iterations is given by

$$k \geq \frac{\log (\varepsilon / \| e_0 \|)}{\log \| I - M^{-1}A \|}$$

$$k \geq \frac{\log \varepsilon}{\log \| I - M^{-1}A \|} - \frac{\log \| e_0 \|}{\log \| I - M^{-1}A \|}$$

So for larger values of $\rho(I - M^{-1}A)$ implies the term $\frac{\log \|e_0\|}{\log \|I - M^{-1}A\|}$ is also large. This is because $\log \|I - M^{-1}A\|$ ~~is~~ will be very small to reduce $\log \|e_0\|$.

But again, as $\log \|I - M^{-1}A\|$ approaches to zero from the left side, it means that this term $\frac{\log \|e_0\|}{\log \|I - M^{-1}A\|}$ increases and eventually become undefined at $\log \|I - M^{-1}A\| = 0$, this case the solution no longer exists, so the the estimate iterations k will seem to underestimate the actual number of k i.e.

$$k \geq \frac{\log(\varepsilon)}{\log \|I - M^{-1}A\|} + \text{a very big term}$$

after which become undefined.

```
function [xroot, en] = steffensens(g,x0,tol,kmax)


xkm1 = x0;

for k = 1:kmax
   gk = g(xkm1);
   ggk = g(gk);
   D = (ggk - 2*gk +xkm1);
   if (D==0)
       fprintf('Tolerance achived\n')
       xroot = g(xkm1);
       break;
   else
       xk = xkm1 - (gk-xkm1)^2/D;

   end


   en(k) = abs(xk - xkm1);

   fprintf('%5d %20.16e, %12.4e\n',k,xk,en(k));
   if (en(k) < tol)
       fprintf('Tolerance achieved\n');
       xroot = xk;
       break;
   end
   xkm1 = xk;

end
xroot = xk;
end
```

```
Not enough input arguments.

Error in steffensens (line 6)
xkm1 = x0;
```

Published with MATLAB® R2020a

```
%Code solves the fixed point iteration problem g(x) = 0.1x +1 using
%steffensens method.

clear all;
close all;

%tolerance
tol = 1e-8;

%intial guess
x0 = 0;

kmax = 20;

%function g(x,y)
g=@(x) 0.1*x+1;

fprintf('Below is the solution for the fixed point problem;\n');

fprintf('    k          x_k                e_n\n');

[xroot, en] = steffensens(g,x0,tol,kmax)

fprintf('We get convergence in one step since xk =xroot is achived only in one step k=1\n');

fprintf('In this case we require only one iteration to converge to the true solution while \n in 1(e) we require atleast k = 8 iterations dependi
```

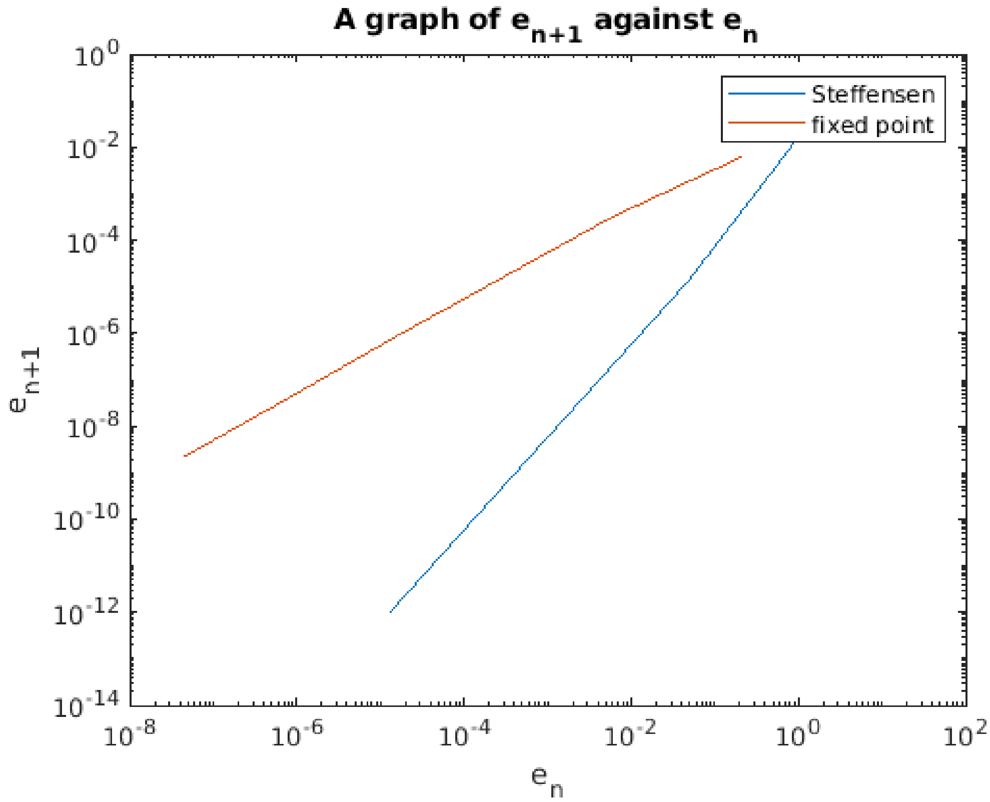
```
Below is the solution for the fixed point problem;
    k          x_k                e_n
    1 1.1111111111111112e+00,   1.1111e+00
Tolerance achived

xroot =

    1.1111


en =

    1.1111

We get convergence in one step since xk =xroot is achived only in one step k=1
In this case we require only one iteration to converge to the true solution while
 in 1(e) we require atleast k = 8 iterations depending on the the magnitude of log|eo|.
```

Published with MATLAB® R2020a

```
%Code accelerates the convergence of a fixed-point algorithm using
%steffensens method.

clear all;
close all;

%tolerance
tol = 1e-8;

%intial guess
x0 = 0.2;

kmax = 100;

%function g(x,y)
g=@(x) (3+3*x-x^2)^(1/3);

fprintf('Below is the solution for the root finding problem;\n');

fprintf('    k          x_k                e_n\n');

[xroot, en] = steffensens(g,x0,tol,kmax)

%Computing e_n
en0 = [];
for k = 1:length(en)-1
   en3 = en(k);
   en0 = [en0,en3];
end

%computing e_n+1
en1 = [];
for k = 2:length(en)
   en2 = en(k);
   en1 = [en1,en2];
end

figure(1);
loglog(en0,en1);
title("A graph of e_n_+_1 against e_n");
ylabel("e_n_+_1");
xlabel("e_n");


slope_steffensens=polyfit(log(en0),log(en1),1);
slope_steffensens = slope_steffensens(1);
fprintf('slope_steffensens = %f\n',slope_steffensens(1));
fprintf('Hence the steffensens is quadratically convergent since its slope is approximately 2.\n');
%fixed point
[en] = fixed_point(g,x0,tol,kmax);

%computing e_n
enf0 = [];
for k = 1:length(en)-1
   en3 = en(k);
   enf0 = [enf0,en3];
end

%computing e_n+1
enf1 = [];
for k = 2:length(en)
   en2 = en(k);
   enf1 = [enf1,en2];
end

 hold on
```

```
loglog(enf0,enf1);
legend('Steffensen','fixed point')

slope_fixed_point=polyfit(log(enf0),log(enf1),1);
slope_fixed_point = slope_fixed_point(1);
fprintf('slope_fixed_point = %f\n',slope_fixed_point(1));
fprintf('Hence the fixed point is linearly convergent since its slope is approximately 1.\n');

%fixed point algorithm
function [en]=fixed_point(g,x0,tol,kmax)

xk = x0;
for k = 1:kmax
    xkp1 = g(xk);
    if abs(xkp1 - xk) < tol
        fprintf('Tolerance achieved\n');
        xroot = xkp1;
        break;
    end
    xk = xkp1;
    en(k) = abs(xkp1 - sqrt(3));
end
%fprintf('\n');
%fprintf('Root is %24.16f\n',xkp1);
%fprintf('Number of iterations : %d\n',k);

end
```

```
Below is the solution for the root finding problem;
    k          x_k                 e_n
    1 1.7778344886912885e+00,   1.5778e+00
    2 1.7320380917493903e+00,   4.5796e-02
    3 1.7320508075679841e+00,   1.2716e-05
    4 1.7320508075688774e+00,   8.9329e-13
Tolerance achieved

xroot =

    1.7321


en =

    1.5778    0.0458    0.0000    0.0000

slope_steffensens = 2.086567
Hence the steffensens is quadratically convergent since its slope is approximately 2.
Tolerance achieved
slope_fixed_point = 0.971838
Hence the fixed point is linearly convergent since its slope is approximately 1.
```



Published with MATLAB® R2020a

```
 %A = [3 -7 -2 2; -3 5 1 0; 6 -4 0 -5; -9 5 -5 12];

 %[L,U,P,pv] = lu_bug_pp(A)

function [L,U,P,pv] = lu_bug_pp(A)

N = size(A,1);

U = A;
L = eye(N);    % Initialize using identity matrix
P = eye(N);
pv = 1:N;

% Decomposition
for k = 1:N-1
    % Find largest pivot in the columnx
    [m,p] = (max(abs(U(k:end,k))));
    U([k,p+k-1],:) = U([p+k-1,k],:);   % Swap rows
    L([k,p+k-1],1:k-1) = L([p+k-1,k],1:k-1);

    % Store permutations
    pv([k,p+k-1]) = pv([p+k-1,k]);

    % Get multiplier, vectors and submatrix
     m  = U(k,k);             % Multiplier
    ck = U(k+1:end,k);       % column vector
    ak = U(k,k+1:end)';      % Use transpose to get a column vector
    Ak = U(k+1:end,k+1:end); % Submatrix

    % Update L
    lk = ck/m;
    L(k+1:end,k) = lk;

    % Update U
    U(k+1:end,k) = 0;                    % Zero out variables
    U(k+1:end,k+1:end) = Ak - lk*ak';   % Outer product used
end
P = P(pv,:);
end
```

```
Not enough input arguments.

Error in lu_bug_pp (line 7)
N = size(A,1);
```

Published with MATLAB® R2020a

```
function x = lu_solve(L,U,b,pv)

N = size(L,1);

% Forward Solve
y = zeros(N,1);
for i = 1:N
    lk = L(i,1:i-1)';
    yk = y(1:i-1);
    y(i) = b(pv(i)) - lk'*yk;

end

% Backward Solve
x = zeros(N,1);
for i = N:-1:1
    m = U(i,i);
    xk = x(i+1:end);
    ak = U(i,i+1:end)';
    x(i) = (y(i) - ak'*xk)/m;
end

end
```

```
Not enough input arguments.

Error in lu_solve (line 4)
N = size(L,1);
```

Published with MATLAB® R2020a

```
clear all;
close all;

A = [1e-10 4;2 1];

b = [1;1];

%Exact solution
fprintf('True Solution;\n');
U_exact = A\b

%LU without partial pivoting
[L, U] = lu_wopp(A);
fprintf('Solution Obtained without partial pivoting;\n');
x_opp = lu_solve1(L,U,b)

%LU with partial pivoting
[L,U,P,pv] = lu_bug_pp(A);
fprintf('Solution Obtained with partial pivoting;\n');
x_wpp = lu_solve(L,U,b,pv)

%Error of LU with partial pivoting
fprintf('Error Obtained with partial pivoting;\n');
error_wpp = abs(U_exact - x_wpp)
fprintf('Error Obtained without partial pivoting;\n');
error_wopp = abs(U_exact - x_opp)

fprintf('Using partial pivoting we obatin exact values because we obtain zero error,\n while without partial pivoting we obtained a slightly smal

fprintf('d). While doing LU decomposition, we need to create an upper triangular matrix U, by making\n  entry a_21 = 0 in matrix A. We shall have
```

```
True Solution;

U_exact =

    0.3750
    0.2500

Solution Obtained without partial pivoting;

x_opp =

    0.3750
    0.2500

Solution Obtained with partial pivoting;

x_wpp =

    0.3750
    0.2500

Error Obtained with partial pivoting;

error_wpp =

     0
     0

Error Obtained without partial pivoting;

error_wopp =

   1.0e-07 *

    0.3102
         0

Using partial pivoting we obatin exact values because we obtain zero error,
 while without partial pivoting we obtained a slightly smaller error
d). While doing LU decomposition, we need to create an upper triangular matrix U, by making
  entry a_21 = 0 in matrix A. We shall have to perform a calculation R2 <-- (1e-10)R2  - 2R1,
 which will become (1e-10)(1) -2(4), so we shall have a very small number in magnitude minus
 a big number in magnitude: 8. Normally this must give us -8, but due to catastrophic loss
 of accuracy we obtain -7.9999999998 hence catastrophic cancellation.
```

Published with MATLAB® R2020a

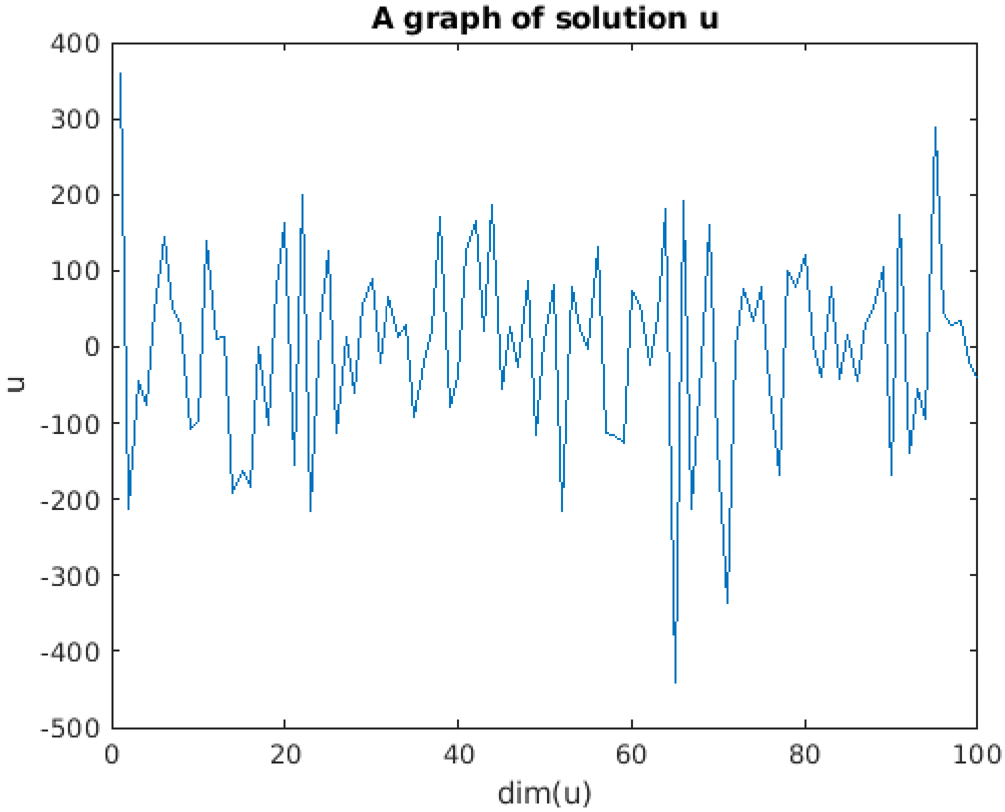
```
clear all;
close all;

%rng('default')
B = rand(4,4);
A = B'*B; %To make A symmetrically positive definite
b = rand(4,1);
tol = 10^(-8);   %relative residual
kmax = 10^5;


u = Gauss(A,b,tol,kmax)

%Eigen values of A
lambda = eig(A)

fprintf("Since the eigen vlaues of A are all positive hence A is symmetric positive definite,\n hence given any vector x , x'Ax > 0.\n")

function x = Gauss(A,b,tol,kmax)

n = size(A,1);

% Intial guesss
xk = zeros(n,1);

%compute an intial residual
rk = b - A*xk;

D = diag(diag(A));
L = tril(A) -D;
M = D + L;

zk = M\rk; %intial approx

for k = 1:kmax

    xkp1 = xk +zk;
    rkp1 = b - A*xkp1;
    zkp1 = M\rkp1;

    if norm(zkp1) < tol
        break;
    end
    xk = xkp1;
    zk = zkp1;

end
fprintf('The number of iterations k = %3d\n', k);
x = xkp1;

end
```

```
The number of iterations k = 298

u =

   -0.4800
    1.0048
   -6.5607
   11.4476


lambda =

    0.0197
    0.0700
    0.3937
    4.1868

Since the eigen vlaues of A are all positive hence A is symmetric positive definite,
 hence given any vector x , x'Ax > 0.
```

Published with MATLAB® R2020a

```
clear all;
close all;

%rng('default')
B = rand(100,100);
A = B'*B;
b = rand(100,1);
tol = 10^(-8);   %relative residual

kmax = 1000;

u = CG(A, b, tol,kmax);

plot(u);
xlabel('dim(u)');
ylabel('u');
title('A graph of solution u');
%Conjugate gradients

function u = CG(A, b, tol,kmax)

n = size(A,1);

% Intial guesss uo
uo = zeros(n,1);

ro = b - A*uo;
po = ro;


for k = 1:kmax
    wo = A*po;
    alphao = (ro'*ro)/(po'*wo);
    uk = uo + alphao*po;
    rk = ro - alphao*wo;

    if norm(rk,2)<tol*norm(b,2)
        break;
    end

    betao = (rk'*rk)/(ro'*ro);
    pk = rk + betao*po;

    uo = uk;
    ro = rk;
    po = pk;
end

fprintf('The number of iterations k = %3d\n', k);

u = uk;

end
```

```
The number of iterations k = 188
```



Published with MATLAB® R2020a

```
%Compares the number of iterations needed using CG and Gauss-Seidel

clear all;
close all;

%rng('default')
B = rand(100,100);
A = B'*B;
b = rand(100,1);
tol = 10^(-8);   %relative residual
kmax = 10^8;


u = CG(A, b, tol,kmax); %Conjugate Gradient
u = Gauss(A,b,tol,kmax); %Gauss-Seidel

fprintf('Therefore CG converges faster than Gauss-Seidel\n');
```

```
CG takes k = 186
Gauss-Seidel takes k = 262526
Therefore CG converges faster than Gauss-Seidel
```

Published with MATLAB® R2020a

a) Suppose

$$X_{k+1} - X_k = \alpha_k P_k \quad — ①$$

The residual $r_{k+1} = r_k - \alpha_k A P_k$, where A is a positive symmetric definite matrix.

Using equation ① and since α_k is a constant

$$r_{k+1} = r_k - A\alpha_k P_k = r_k - A(X_{k+1} - X_k)$$

$$r_{k+1} = r_k - AX_{k+1} - \cancel{AX_k}$$

$$r_{k+1} = r_k - AX_k - AX_{k+1}$$

but the residual $r_k = b - AX_k$, given $AX_k = b$

so $r_{k+1} = b - AX_{k+1}$

b) - The directions P_k satisfy $P_{k+1}^T A P_k = 0$, iff P_{k+1} is A-conjugate to P_k with A a positive symmetric definite Matrix.

- This means that P_{k+1} is mutually orthogonal to P_k, given that P_{k+1} and P_k are eigen vectors of A with corresponding eigen values λ_{k+1} and λ_k

- If this is true then

$$AP_k = \lambda_k P_k \quad \text{for } k = 1, \dots, n$$

Since $A \in \mathbb{R}^{n \times n}$ and is symmetric positive definite then $\exists$ n eigen vectors $P_1, P_2, \dots, P_n$ which are mutually orthogonal i.e. $(P_i, P_j) = 0$ if $i \neq j$

therefore

$$P_{k+1}^T A P_k = (P_{k+1}, AP_k)$$
$$= (P_{k+1}, \lambda P_k)$$
$$= \lambda (P_{k+1}, P_k)$$

Since k is a positive integer, so $k+1 \neq k$,

$$P_{k+1}^T A P_k = \lambda(0) = 0$$

Hence the directions P_k, P_{k+1} satisfy $P_{k+1}^T A P_T = 0$

Since they are mutually orthogonal, hence A - conjugate to each other.