Dt.
Pg. B+

SC Assignment - 1

2020349

Vibhor Agarwal

Q1 Here x_0 is the actual sol^n & x is the computed sol^n.

We know,

absolute condition number = $\frac{\text{absolute forward error}}{\text{absolute backward error}}$

$$= \frac{|x - x_0|}{|b - b_0|}$$

$$= \frac{|x - x_0|}{|ax - ax_0|}$$

$$= \frac{|x - x_0|}{a|x - x_0|}$$

$$= \frac{1}{a}$$

Relative cond^n number = $\frac{\text{relative forward error}}{\text{absolute backward error}}$

$$= \frac{|x - x_0|}{|x_0| \cdot \frac{|ax - ax_0|}{|ax_0|}} = \frac{|x - x_0|}{|x - x_0|}$$

$$= 1$$

Dt. B+
Pg.

Q2 We know, absolute cond^n number $= |f'(x)| = \hat{K}$

& Relative condition number, $K = \left| \frac{x\hat{K}}{f(x)} \right|$.

$$= \left| \frac{x f'(x)}{f(x)} \right|$$

So for

(a) $f(x) = (x-1)^{\alpha}$

$\Rightarrow f'(x) = \frac{d}{dx}(x-1)^{\alpha} = \alpha(x-1)^{\alpha-1}$

So, absolute condition no. $\hat{K} = |f'(x)|$

$$= |\alpha(x-1)^{\alpha-1}|$$

Case-1

Now if $\alpha > 1$ then

for large x ie $x >> 1$; the absolute no. will also be large ie $\hat{K} >> 1$. This is explainable as $\lim_{x \to \infty} |\alpha(x-1)^{\alpha-1}| \to \infty$

for x to be very small ie $x << 1$; absolute cond^n number we be large ie $K >> 1$.

as $\lim x \to -\infty$, $|\alpha(x-1)^{\alpha-1}| \to \infty$ for $\alpha > 1$.

Case-2 $\alpha < 1$

As $\lim_{x \to 1} |\alpha(x-1)^{\alpha-1}| \to \infty$

So for $\alpha < 1$, the x takes large values in the vicinity of 1.

Relative cond^n number $K = \left| \frac{x f'(x)}{f(x)} \right|$

$$= \left| \frac{x \alpha (x-1)^{\alpha-1}}{(x-1)^{\alpha}} \right|$$

$$= \left| \frac{\alpha x}{x-1} \right|$$

For it: $\lim_{x \to 1} \left| \frac{\alpha x}{x-1} \right| \to \infty$ (as $x-1 \to 0$), for $x \to 1$.

So the rel cond^n number takes large values in the vicinity of 1.

(b) $f(x) = \ln x$

, so $f'(x) = \frac{1}{x}$.

So Absolute cond^n No. $\hat{K} = |f'(x)|$

$$= \left| \frac{1}{x} \right|$$

We know,

as $x \to 0$ $\left| \frac{1}{x} \right| \to \infty$.

Dt.
Pg. B+

So $\tilde{K}$ takes large values in the neighbourhood of 0.

Relative cond^n no. $K = \left| \frac{x f'(x)}{f(x)} \right|$

$$= \left| \frac{x \cdot \frac{1}{x}}{\ln x} \right|$$

$$= \left| \frac{1}{\ln x} \right|$$

Here as $\lim_{x \to 1} \ln x \to 0 \Rightarrow$

$\lim_{x \to 1} \left| \frac{1}{\ln x} \right| \to \infty$. So when x takes values in the vicinity of 1 we have large values of K.

(c) $f(x) = x^{-1} e^x$.

$f'(x) = \frac{d}{dx}(x^{-1}) e^x$

$= x^{-1} e^x + \frac{e^x}{(-x^2)}$

$= e^x \left[\frac{1}{x} - \frac{1}{x^2} \right] = \frac{e^x [x-1]}{x^2}$

Dt.
Pg.

Absolute cond^{n} no. is $\bar{k} = |f'(x)|$

$$= \left| \frac{e^x (x-1)}{x^2} \right|$$

as if $x \to 0$: $\left| \frac{e^x (x-1)}{x^2} \right| \to \infty$ as $\frac{1}{x^2} \to \infty$ for $x \to 0$

So we have large absolute cod^{n} no. when x takes values close to 0.

Now
as if $x \to \infty$ $\left| \frac{e^x (x-1)}{x^2} \right| \to \infty$ as $e^x \to \infty$ as $x \to \infty$

so.
for very large values of x we will also have large absolute cond^{n} no.s.

Now, Relative cond^{n} no. $k = \left| \frac{x f'(x)}{f(x)} \right|$

$$= \left| \frac{x e^x (x-1)}{x^2 e^x} \right|$$

$$= |x-1|$$

Now, as $x \to \infty \Rightarrow |x-1| \to \infty$, and
as $x \to -\infty \Rightarrow |x-1| \to \infty$.

Dt.
Pg. B+

Hence we conclude relative cond^n number is very large for very small or very large values of x.

(d) $f(x) = \frac{1}{1+x^{-1}} = \frac{x}{1+x} = 1 - \frac{1}{1+x}$.

$f'(x) = \frac{d}{dx}\left(\frac{x}{1+x}\right) = \frac{d}{dx}\left(1 - \frac{1}{1+x}\right)$

$= \frac{1}{(1+x)^2}$

So $\tilde{K} = \left| \frac{1}{(1+x)^2} \right|$ $\quad \tilde{K}$ = Absolute cond^n no.).

Now as $x \to -1 \Rightarrow 1+x \to 0 \Rightarrow \frac{1}{(1+x)^2} \to \infty$.

So when x take values close to -1, we have very high value of absolute cond^n no.

Rel^n condition no. $K = \left| \frac{x f'(x)}{f(x)} \right|$

$= \left| \frac{x}{(1+x)^2 \cdot x} (1+x) \right|$

$= \left| \frac{1}{1+x} \right|$

Q3 (a) We are given,

$$f(x(\epsilon)) + \epsilon p(x(\epsilon)) = 0$$

$\Rightarrow$ On differentiating wrt ϵ

$\Rightarrow$ $$f'(x(\epsilon)) \cdot \frac{dx}{d\epsilon} + p(x(\epsilon)) + \epsilon \underbrace{p'(x(\epsilon))}_{\downarrow 0} \cdot \frac{dx}{d\epsilon} = 0$$

Since $\epsilon = 0 \Rightarrow x(0) = x^*$ as given

$$\Rightarrow \frac{dx}{d\epsilon} f'(x(\epsilon)) = p(x(\epsilon)) .$$

Dt.
Pg. B+

$$\left.\frac{dx}{d\epsilon}\right|_{\epsilon=0} = -\frac{p(x^*)}{f'(x^*)}.$$

(b) $$\left.\frac{dx}{d\epsilon}\right|_{\epsilon=0,\, x^*=j} = -\frac{p(x^*)}{f'(x^*)}$$

$$= -\frac{j^{19}}{f'(j)}.$$

$$f(x) = (x-1)(x-2)\cdots(x-j)\cdots(x-20)$$

$$f'(x^*)\Big|_{x^*=j} = (x-1)(x-2)\cdots(x-20)$$

$$f'(x^*)\Big|_{x^*=j} = \prod_{\substack{K=1 \\ K\neq j}}^{20} (j-K) \quad \text{where } K \neq j$$

$$\left.\frac{dx}{d\epsilon}\right|_{\epsilon=0,\, x^*=j} = \frac{-j^{19}}{\prod_{\substack{K=1 \\ K\neq j}}^{20} (j-K)}$$

$$\boxed{\left.\frac{dx}{d\epsilon}\right|_{\epsilon=0,\, x^*=j} = -\prod_{\substack{K=1 \\ K\neq j}}^{20} \frac{j}{j-K}}$$

Dt.
Pg. B+

(c) $x^* = 1$

$$\left.\frac{dx}{d\epsilon}\right|_{\epsilon=0,\, x^*=1} = -\frac{1}{(1-2)(1-3)\cdots(1-20)}$$

$$= \frac{1}{1\times 2\times \cdots 19}$$ (as 19 terms so taking - common one remains which interacts with outers)

$$= \frac{1}{19!}$$

So at $x^* = 1$ the change produced in the x would be very small even for a large change in ϵ.
So this root would be stable.

Now $x^* = 20$

$$\left.\frac{dx}{d\epsilon}\right|_{\epsilon=0,\, x=20} = -\frac{20^{19}}{19!} \cdot a$$

$$= \frac{-20^{19}}{(20-1)(20-2)\cdots(20-19)}$$

$$= \frac{-20^{19}}{19!}$$

So, $\left|\frac{dx}{dt}\right|_{\epsilon=0,\, x=20} = \frac{20^{19}}{19!}$

Hence at $x^* = 20$ even a very small change in ϵ has a huge change in the value of x, So the root is unstable at $x^* = 20$

Dt.
Pg. B+

Q6 (a) Given: Rank $(A) = n$, $A \in R^{m \times n}$.
$\| \ \|$ is a vector norm in R^m.

To show:- $\|x\|_A = \|Ax\|$ is a vector norm on R^n

ie. we need to prove.

(a) $\|x\|_A \geq 0$ & $\|x\|_A = 0$ iff $x = 0$

(b) $\|\alpha x\|_A = |\alpha| \, \|x\|_A$ $\forall \alpha \in R, x \in R^n$.

(c) $\|x+y\|_A \leq \|x\|_A + \|y\|_A$ $\forall y, x \in R^n$.

Proof:-

(a) $\|x\|_A \geq 0$ & $\|x\|_A = 0$ iff $x = 0$

$\|x\|_A = \|Ax\|$

Now we know $\|x\|_A$ is a vector norm so Ax should be a vector. Hence $x \in R^n$, $Ax \in R^m$.

we assume $Ax = b$, $b \in R^m$.

$\Rightarrow$ So, $\|x\|_A = \|Ax\|$
$= \|b\|$.

Now $\|b\|_p = \left(\sum_{i=1}^{m} |b_i|^p \right)^{1/p}$.

Here, since $|b_i| \geq 0$ so, $|b_i|^p \geq 0$ &
hence $\sum_{i=1}^{m} |b_i|^p \geq 0$

So the p^{th} root of a positive value must be positive.

So $\|b\| \geq 0 \Rightarrow \|x\|_A \geq 0$.

Now,

$\|x\|_A = \|b\|$

For $\|x\|_A = 0 \Rightarrow \|b\| = 0$

So, b's norm will be zero only when all elements of b are zero.

ie b is a zero vector.

But $Ax = b$ so Ax is a zero vector

Hence, x is a zero vector as A^- can't be zero matrix as it is assumed to full rank.

So, $x = 0$ iff $\|x\|_A = 0$. Hence cond" (i) is satisfied. & proved

(ii) To prove: $\|\alpha x\|_A = |\alpha| \|x\|_A$

$\|\alpha x\|_A = \|\alpha Ax\|$

Let $Ax = b$ so,

$\|\alpha x\|_A = \|\alpha b\|$

Dt. _____ B+
Pg. _____

$$= \sum_{i=1}^{m} |\alpha b_i|^p$$

$$= \left(\sum_{i=1}^{m} |\alpha b_i|^p \right)^{1/p}$$

$$= |\alpha| \left(\sum_{i=1}^{m} |b_i|^p \right)^{1/p}$$

$$= |\alpha| \|b\|$$

$$= \|\alpha\| \|x\|_A \qquad (\text{as } \|b\| = \|x\|_A)$$

So, $\|\alpha x\|_A = |\alpha| \|x\|_A$

This proved the second cond^n.

(iii) $\|x+y\|_A \leq \|x\|_A + \|y\|_A \quad \forall\, x, y \in R^n$.

$\|x+y\|_A = \|A(x+y)\|$

$= \|Ax + Ay\|$

Since $\|\cdot\|$ is a norm is satisfies
$\|x+y\| \leq \|x\| + \|y\|$

$\|x+y\|_A \leq \|Ax\| + \|Ay\|$

$\|x+y\|_A \leq \|x\|_A + \|y\|_A$

This proves the third cond^n.

Hence this shows $\|x\|_A$ is actualy a vector norm on R^n.

Dt.
Pg.

B+

Q6 (b) Given: $\|A\|_F = \left(\sum_{i=1}^{m} \sum_{i=1}^{n} |a_{ij}|^2 \right)^{1/2}$, $A \in R^{m \times n}$

To Prove: $\|A\|_F$ satisfies the properties of norms.

ie

(i) $\|A\|_F \geq 0$ & $\|A\|_F = 0 \Leftrightarrow A$ is zero matrix

(ii) $\|\alpha A\| = |\alpha| \|A\| \quad \forall \alpha \in R$.

(iii) $\|A + B\| \leq \|A\| + \|B\|$

Proof

(i) $\|A\|_F \geq 0$ & $\|A\|_F = 0 \Leftrightarrow A$ is zero matrix.

So, $\|A\|_F = \left(\sum_{i=1}^{m} \sum_{i=1}^{n} |a_{ij}|^2 \right)^{1/2}$

Now since $|a_{ij}| \geq 0$. so, its sum would be ≥ 0. Hence the square root of +ve is also 0 +ve.

So, $\|A\|_F \geq 0$.

Now, if $\|A\|_F = 0$ ie $\left(\sum_{i=1}^{m} \sum_{i=1}^{n} |a_{ij}|^2 \right)^{1/2} = 0$

But since $|a_{ij}| \geq 0$. So sum of positive is zero iff each entry is zero.

Dt. Pg. B+

So $a_{ij} = 0 \iff A$ is a zero matrix.

Hence, $\|A\|_F = 0 \iff A$ is a zero matrix.
as if A is a zero then $a_{ij} = 0 \left(\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}\right) = 0$

This completes proof of cond^{n} (i).

(ii) $\|\alpha A\|_F = |\alpha| \|A\|_F$.

$$\|\alpha A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (\alpha a_{ij})^2 \right)^{1/2}$$

$$= (\alpha^2)^{1/2} \left(\sum_{j=1}^{m} \sum_{j=1}^{n} (a_{ij})^2 \right)^{1/2}$$

$$= |\alpha| \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij})^2 \right)^{1/2}.$$

$$\|\alpha A\|_F = |\alpha| \|A\|_F$$

This shows the second cond^{n}.

(iii) $\|A + B\| \leq \|A\| + \|B\|$

$$\text{So } \|A+B\| = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij} + b_{ij})^2 \right)^{1/2}$$

$$\|A+B\| = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij})^2 + \sum_{i=1}^{m} \sum_{j=1}^{n} (b_{ij})^2 + 2 \sum_{j=1}^{m} \sum_{j=1}^{n} a_{ij} b_{ij} \right)^{1/2}$$

Now we know cauchy schwarz inequality is,

Dt. _____ B+

Pg. _____

$$\left(\sum a_i b_i\right)^2 \leq \sum a_i^2 \sum b_i^2$$

$$\sum a_i b_i \leq \left(\sum a_i^2\right)^{1/2} \left(\sum b_i^2\right)^{1/2}$$

So,

$$\|A+B\| \leq \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij})^2 + \sum_{i=1}^{m} \sum_{j=1}^{n} (b_{ij})^2 + 2 \left(\sum a_i^2\right)^{1/2} \left(\sum b_i^2\right)^{1/2} \right)^{1/2}$$

$$\|A+B\| \leq \left(\left(\left(\sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij})^2 \right)^{1/2} + \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (b_{ij})^2 \right)^{1/2} \right)^2 \right)^{1/2}$$

$$\|A+B\| \leq \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij})^2 \right)^{1/2} + \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (b_{ij})^2 \right)^{1/2}$$

$$\boxed{\|A+B\| \leq \|A\| + \|B\|}$$

This completes the proof for condition 3.

Hence $\|A\|_f$ satisfies the conditions for matrix norms.

Dt.
Pg. B+

Q6 (c)

Given

$E = uv^T$, $u \in R^m$, $v \in R^n$ $\Rightarrow E \in R^{m \times n}$.

To Prove:- $\|E\|_F = \|E_2\| = \|u\|_2 \|v\|_2$.

$\|E\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |e_{ij}|^2 \right)^{1/2}$ by def^n of Frobenius Norm.

we know $tr(A) = \sum_{i=1}^{m} \sum_{j=1}^{n} |e_{ij}|$.

ie the s

we know that $tr(A) = \sum_{i=1}^{m} a_{ii}$ where $A \in R^{m \times m}$.

Now $A = EE^T \in R^{m \times m}$, & we can easily observe

$\sum_{i=1}^{m} a_i = \sum_{i=1}^{m} \sum_{j=1}^{n} |e_{ij}|^2 = tr(A) = tr(EE^T)$ — (i)

$\|E\|_F = \sqrt{Tr(EE^T)}$

$= \sqrt{tr\{(uv^T)(uv^T)^T}$

$= \sqrt{tr[uv^T vu^T]}$

Now we know trace of a matrix is associative over product.

Dt.
Pg.

B+

Now $u^T u$ is a scalar & $v^T v$ is scalar.

$$\|E\|_f = \sqrt{tr(u^T u) \cdot tr(v^T v)} \quad [\text{since } tr(u^T u) \text{ is a scalar}]$$

$$= \sqrt{tr(u^T u)} \cdot \sqrt{tr(v^T v)}.$$

$$= \sqrt{tr(uu^T)} \cdot \sqrt{tr(vv^T)}$$

$$= \|u\|_2 \|v\|_2 \quad \text{from (i)}$$

$$\therefore \|E\|_f = \|E\|_2 \quad (\text{as } \|E\|_f = \|E\|_2)$$

So $\|E\|_f = \|u\|_2 \|v\|_2 = \|E\|_2$.

Q4

(A)

We observe that the numbers printed keep getting divided by 2 and the code finally stops after printing zero. The number that we get before exiting at 0 is 5e-324. This number is approximately the **UFL or Underflow of IEEE double precision**. So, on dividing this by 2 further we get zero as its smaller than the least IEEE double precision could represent

(B)

We get 1.1102230246251565e-16 as the output. The printed value of eps is the value which when added to 1 does not change its value and so the loop breaks. In IEEE double precision the value of ε_M is roughly 2.2204460492503 e-16. As we observe the printed number is roughly half of ε_M and hence 1 remains unchanged on addition.

(C)

We get inf as the last number in the output. The code snippet keeps doubling the numbers untill the numbers become larger than **OFL or overflow of IEEE double precision**. So, when the number becomes equal to OFL the while loop breaks.

Q7

A

```
current n is 1700 and the value of term is 0.5775097537135299
current n is 1800 and the value of term is 0.5774934169591495
current n is 1900 and the value of term is 0.5774787997122512
current n is 2000 and the value of term is 0.5774656440682016
current n is 2100 and the value of term is 0.577453741243179
current n is 2200 and the value of term is 0.5774429204111771
current n is 2300 and the value of term is 0.5774330404528918
current n is 2400 and the value of term is 0.5774239837672805
current n is 2500 and the value of term is 0.5774156515681996
current n is 2600 and the value of term is 0.5774079602664202
current n is 2700 and the value of term is 0.5774008386555254
current n is 2800 and the value of term is 0.5773942257008384
current n is 2900 and the value of term is 0.5773880687857824
current n is 3000 and the value of term is 0.5773823223089227
current n is 3100 and the value of term is 0.5773769465525795
current n is 3200 and the value of term is 0.5773719067634975
current n is 3300 and the value of term is 0.5773671724007521
current n is 3400 and the value of term is 0.5773627165162711
current n is 3500 and the value of term is 0.5773585152416505
current n is 3600 and the value of term is 0.5773545473603523
current n is 3700 and the value of term is 0.5773507939494777
current n is 3800 and the value of term is 0.5773472380778735
current n is 3900 and the value of term is 0.5773438645508655
current n is 4000 and the value of term is 0.5773406596931707
current n is 4100 and the value of term is 0.5773376111636459
current n is 4200 and the value of term is 0.5773347077964406
current n is 4300 and the value of term is 0.577331939464333
current n is 4400 and the value of term is 0.5773292969607322
current n is 4500 and the value of term is 0.5773267718973916
current n is 4600 and the value of term is 0.5773243566154296
current n is 4700 and the value of term is 0.5773220441077846
current n is 4800 and the value of term is 0.5773198279512748
current n is 4900 and the value of term is 0.5773177022470506
current n is 5000 and the value of term is 0.577315661568166
The final estimated value of eulers constant is 0.577315661568166
```

B

```
current n is 1400 and the value of term is 0.5772156861448545
current n is 1500 and the value of term is 0.5772156834077089
current n is 1600 and the value of term is 0.5772156811674058
current n is 1700 and the value of term is 0.5772156793105871
current n is 1800 and the value of term is 0.5772156777544755
current n is 1900 and the value of term is 0.5772156764374801
current n is 2000 and the value of term is 0.5772156753129938
current n is 2100 and the value of term is 0.5772156743452568
current n is 2200 and the value of term is 0.577215673506438
current n is 2300 and the value of term is 0.5772156727746101
current n is 2400 and the value of term is 0.5772156721323229
current n is 2500 and the value of term is 0.5772156715655328
current n is 2600 and the value of term is 0.5772156710628664
current n is 2700 and the value of term is 0.5772156706149998
current n is 2800 and the value of term is 0.5772156702142466
current n is 2900 and the value of term is 0.5772156698542288
current n is 3000 and the value of term is 0.5772156695296005
current n is 3100 and the value of term is 0.5772156692358834
current n is 3200 and the value of term is 0.5772156689692576
current n is 3300 and the value of term is 0.5772156687264989
current n is 3400 and the value of term is 0.5772156685048309
current n is 3500 and the value of term is 0.5772156683019034
current n is 3600 and the value of term is 0.5772156681156311
current n is 3700 and the value of term is 0.577215667944273
current n is 3800 and the value of term is 0.5772156677862554
current n is 3900 and the value of term is 0.5772156676402354
current n is 4000 and the value of term is 0.5772156675050191
current n is 4100 and the value of term is 0.5772156673795781
current n is 4200 and the value of term is 0.5772156672629993
current n is 4300 and the value of term is 0.5772156671544533
current n is 4400 and the value of term is 0.5772156670532187
current n is 4500 and the value of term is 0.5772156669586632
current n is 4600 and the value of term is 0.5772156668702006
current n is 4700 and the value of term is 0.5772156667873283
current n is 4800 and the value of term is 0.5772156667095789
current n is 4900 and the value of term is 0.5772156666365351
current n is 5000 and the value of term is 0.5772156665678327
The final estimated value of eulers constant is 0.5772156665678327
```

We can clearly see from the above two images that the function in B converges faster than the function in A.

Function in A converged to 0.577315661568166 in 5000 iterations while B converged to 0.5772156665678327 in 5000 iterations which is much closer to the actual value of gamma.

Q8

In part C as observed by me none of the solves fail.

The output table printed by the code is as follows

```
for n= 10
Matrix       Condition Number     Error GE    Error GPE    Error Numpy    Residual GE    Residual GPE    Residual Numpy
-----------  -----------------  ----------  -----------  -------------  -------------  --------------  ----------------

Random             938.723       3.21044e-14  1.55858e-14  1.28138e-14  5.96611e-16    8.48844e-17     9.67831e-17
Hilbert            1.6025e+13    0.000422352  0.000500639  0.000417071  4.81637e-17    1.7254e-16      4.39672e-17
One Matrix         6.31375       0            0            0            0              0               0

for n= 20
Matrix       Condition Number     Error GE     Error GPE    Error Numpy    Residual GE    Residual GPE    Residual Numpy
-----------  -----------------  ----------  -----------  -------------  -------------  --------------  ----------------

Random             271.305       2.85983e-13  2.97656e-15  4.85372e-15  1.58287e-14    1.08952e-16     1.47186e-16
Hilbert            2.10653e+18   3.97508      44.4832      30.6275      1.06939e-16    6.39899e-16     4.00526e-16
One Matrix         12.7062       0            0            0            0              0               0

for n= 30
Matrix       Condition Number     Error GE     Error GPE    Error Numpy    Residual GE    Residual GPE    Residual Numpy
-----------  -----------------  ----------  -----------  -------------  -------------  --------------  ----------------

Random             580.574       1.54108e-13  5.98848e-15  1.65142e-14  1.62753e-15    1.2953e-16      2.10168e-16
Hilbert            5.10189e+18   38.8873      35.1388      11.3935      4.16574e-16    4.73653e-16     1.6588e-16
One Matrix         19.0811       0            0            0            0              0               0

for n= 40
Matrix       Condition Number     Error GE     Error GPE    Error Numpy    Residual GE    Residual GPE    Residual Numpy
-----------  -----------------  ----------  -----------  -------------  -------------  --------------  ----------------

Random             415.74        6.62246e-14  1.29369e-14  1.33508e-14  2.49228e-15    1.25872e-16     1.41768e-16
Hilbert            6.1556e+18    109.177      25.7024      15.2146      9.88671e-16    6.52894e-16     1.72774e-16
One Matrix         25.4517       0            0            0            0              0               0
```

Explanation for Random Matrix

The generated random matrices of size 10 ,20,30,40 have relatively small condition numbers. This explains that the relative error in the solution of Random matrices is very less and of the order 1e-14. This being a random matrix can have a large difference between the values in a row and in the same column which might produce roundoff errors in floating point system. We can also observe that the error in each size of random matrix decrease when we use pivoting compared to without pivoting.

Explanation for Hilbert Matrix

The hilbert matrix of different sizes have very large condition numbers since it can have large roundoff errors in floating point system as the entries are of the form 1/(i+j-1) which will require very high precision for storing and needs to be rounded for storing in the computers. The high condition number explains a very high relative error in the solution. Hilbert matrix have a large relative error even for numpy.linalg.solve function. Pivoting is needed in the Hilbert matrix because as the i increases along the column the entries decrease and might have large difference between the first and the last entry of a column for large n. This is also evident in the above solution that the error of Gaussian elimination with pivoting is less than without pivoting in almost all cases.

Explanation for One's Matrix

The given one's matrices have very small condition number. This is because they only contains 1 and -1 which don't suffer any roundoff error and hence the solution is very accurate. This explains the fact that they have zero relative error in their solutions for all the three methods. Since the entries are just 1 and -1 , Gaussian elimination without pivoting already gives very accurate results. Pivoting hence is not very necessary in this case although can be done.

Q5

The output that we get from our code is as follows

```
Condition number is 125664458272343.4218750000000000
Relative backward error is 2.74847887190819777006591376 9E-17
Error in the input is 0.00172692020650673283087998 7643
The value of j is 14
(x,tan(x) = (628318530717959.3750000000000000,0.8900802593986616)
Condition number is 1265166139489326.2500000000000000
Relative backward error is 8.68816649216572838402769258 4E-17
Error in the input is 0.05458936004990577559242804031
The value of j is 15
(x,tan(x) = (6283185307179587.0000000000000000,0.5766517306609554)
Condition number is 14519188958191302.0000000000000000
Relative backward error is 2.91578455627305426464328445 5E-17
Error in the input is 0.18320414682876006236727783 48
The value of j is 16
(x,tan(x) = (62831853071795864.0000000000000000,-0.9682197486526081)
Condition number is 125729248279028768.0000000000000000
Relative backward error is 1.56544302586186766656903960 8E-17
Error in the input is 0.98359686193220402115846044 78
The value of j is 17
(x,tan(x) = (628318530717958656.0000000000000000,-2.0518446488895292)
Condition number is 1595433312584221184.0000000000000000
Relative backward error is 1.91286255891590311860845819 5E-18
Error in the input is 1.20188699248343487301846471 9
The value of j is 18
(x,tan(x) = (6283185307179586560.0000000000000000,5.5747653335805616)
Condition number is 3615435983118129971 2.0000000000000000
Relative backward error is 1.26534264607143139485677225 0E-19
Error in the input is 0.79503823223437575456518526 49
The value of j is 19
(x,tan(x) = (62831853071795863552.0000000000000000,-8.7118803024535953)
Condition number is 5545957865811806781 44.0000000000000000
Relative backward error is 1.75116373716481331133887491 4E-20
Error in the input is 1.10028862638196499481182749 6
The value of j is 20
(x,tan(x) = (628318530717958668288.0000000000000000,-1.0506941034214516)
Condition number is 1258173865379857956864.0000000000000000
Relative backward error is 1.62989723427637912438071047 4E-21
Error in the input is 1.02409463546179899292093866 9
```

Ideally tan(x + 2nπ) = tan(x)

So tan(π/4+2π ×10^j) = tan (π/4) = 1. So, the output of each value should have been 1.

However, we observe that as the j increases the rounding off error due to rounding pi to 16 decimal places becomes much more prominent and hence the condition number as evident from above results increases with the increase in j.

Now the condition number of tan(x) is |x/(sinx*cosx)| , which helps us to get the value of condition number at a particular x. As x increase the condition number also increases as the |sinx*cosx| varies from 0 to 1.

Now lets analyse our computation at j=20

Here x = $\pi/4+2\pi \times 10^{20}$ and the computed value of x is 628318530717958668288.00000000000000

So the error in the computation is | $\pi/4+2\pi \times 10^{20}$ - 628318530717958668288.0|. Now error in the output is roughly condition number * backward error.
Since the condition number for this computation is 1258173865379857956864 , so even a small error in input produces a large relative error in the input in this case is changes the value from 1 to -1.0506941034214516.

So we conclude that as the number increases the rounding off error increases because the bits to make mistake increases and the condition number also increases as the x increases. Hence this causes a significant change in the value of tan(x) relatively.