भारतीय प्रौद्योगिकी संस्थान गुवाहाटी
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

MA 423: Matrix Computations Lab

Lab 03

AB Satyaprakash (180123062)

24 Aug 2021

Some general points about the report

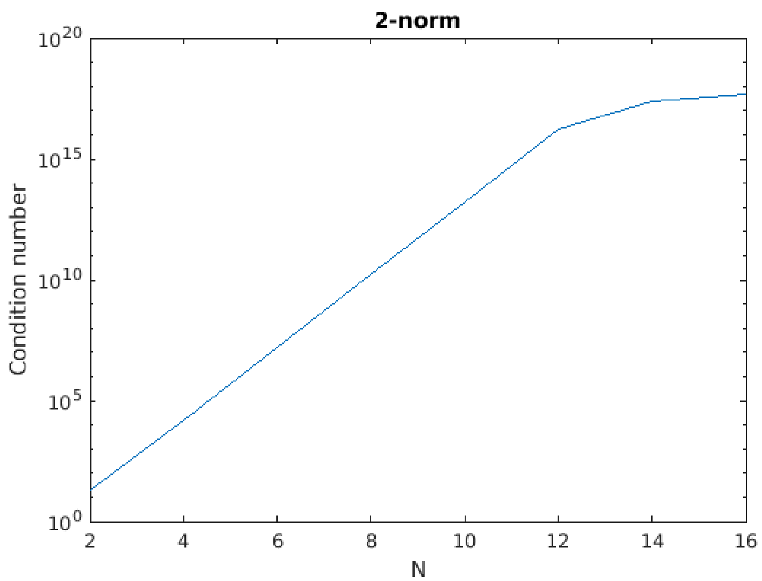
- *All workspaces are saved as *.mat files in **Workspaces** folder and matlab scripts are saved as *.m files.*
- *All graphs are saved as *.fig files in **Graphs** folder.*

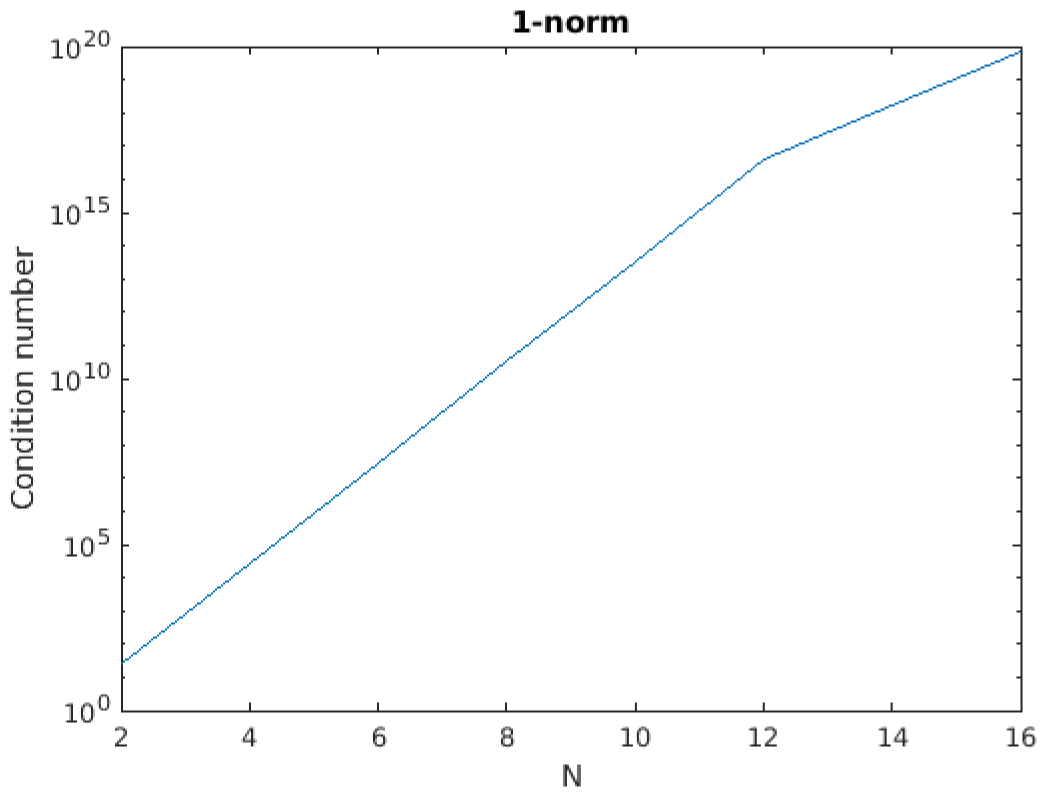
Question 1.

The workspaces are saved as q1_workspace_N={n}.mat corresponding to different values of n = 2,4,6,..16.

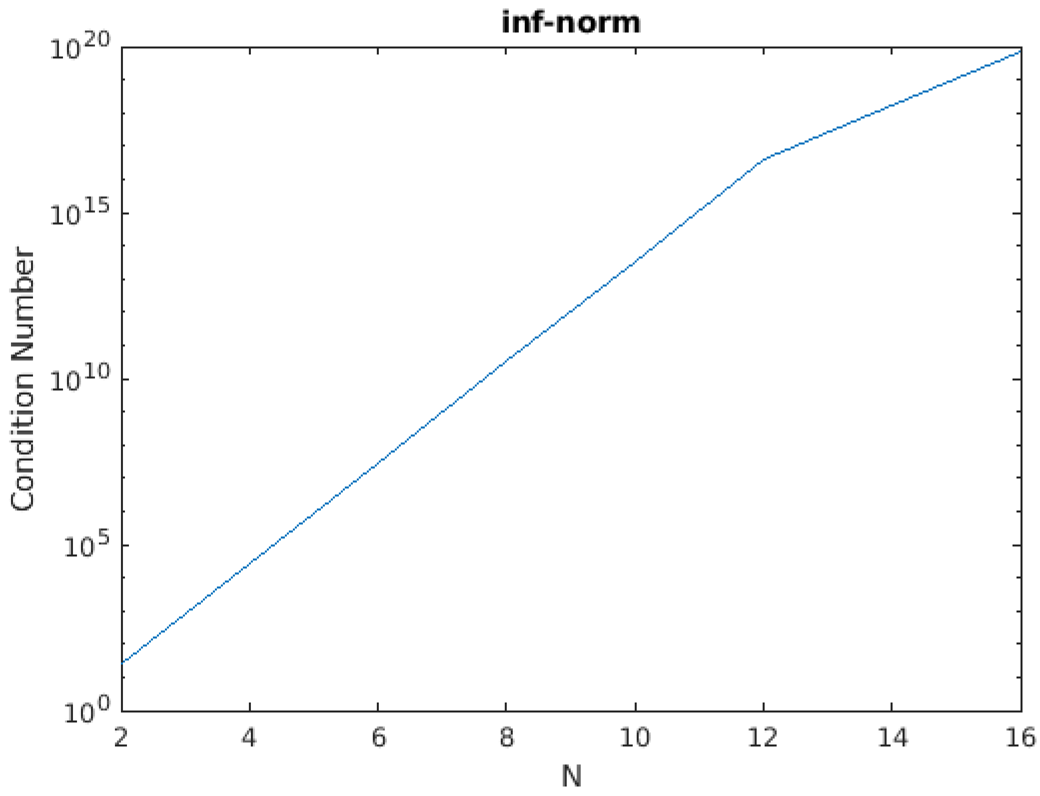
- cond(H,1) and cond(H,∞) are used for finding the 1 and ∞-norm condition numbers.
- From the graphs in our experiment, we observe that -
  - There is a linear relationship between n and log(cond(H)) till the condition number of H is ~10^17, i.e, cond(H) grows exponentially with n.
  - For even larger values of cond(H), due to floating point precision errors, the relationship is no longer linear. The sizes for which the condition number is 10^17 or more are extremely ill conditioned and behave like singular matrices numerically.

- The graphs are shown here -





1-norm
Condition number
N
10^{20}
10^{15}
10^{10}
10^{5}
10^{0}
2
4
6
8
10
12
14
16



inf-norm
Condition Number
N
10^{20}
10^{15}
10^{10}
10^{5}
10^{0}
2
4
6
8
10
12
14
16

Question 2.

n = 8

[x	x1	x2	x3]
3.688115016889260e-01	3.688115016891051e-01	3.688115016666416e-01	3.688115016806969e-01
5.614429412226414e-01	5.614429411844426e-01	5.614429416600615e-01	5.614429416401895e-01
6.432092460025051e-01	6.432092467417192e-01	6.432092413306236e-01	6.432092407409826e-01
3.881099340331140e-02	3.881098849982087e-02	3.881099820137024e-02	3.881102118134311e-02
4.600074285676184e-01	4.600074435513943e-01	4.600072503089905e-01	4.600073551221878e-01
5.272264484567957e-02	5.272262171824994e-02	5.272269248962402e-02	5.272274735096781e-02
7.876652679325113e-01	7.876652854713754e-01	7.876651287078857e-01	7.876651957646704e-01
9.633490553694127e-01	9.633490501746340e-01	9.633490368723869e-01	9.633490755562203e-01

[cond(H)	norm(x - x1)/norm(x)	norm(x - x2)/norm(x)	norm(x - x3)/norm(x)]
1.525757556662796e+10	2.062590103098352e-08	1.430665018421441e-07	9.212863342344688e-08

n = 10

[x	x1	x2	x3]
6.216711263002072e-01	6.216711266077526e-01	6.216711258748546e-01	6.216711263123680e-01
5.096902641015577e-02	5.096899867944871e-02	5.096897482872009e-02	5.096902457924434e-02
7.607881335500398e-01	7.607887440062672e-01	7.607886791229248e-01	7.607881857299530e-01
5.166017793741676e-01	5.165960829273497e-01	5.165987014770508e-01	5.166012083475215e-01
8.776650369749158e-01	8.776927879606891e-01	8.777313232421875e-01	8.776681391210078e-01
8.652841215936570e-01	8.652064938232488e-01	8.653259277343750e-01	8.652747259573316e-01
5.998473664832698e-01	5.999766030084448e-01	5.997009277343750e-01	5.998639865225804e-01
4.608392544295856e-01	4.607128030135361e-01	4.609069824218750e-01	4.608221959231614e-01
1.624213286835777e-01	1.624884268924825e-01	1.624298095703125e-01	1.624307375098126e-01
3.126696629160608e-01	3.126547695001159e-01	3.126792907714844e-01	3.126675066526327e-01

[cond(H)	norm(x - x1)/norm(x)	norm(x - x2)/norm(x)	norm(x - x3)/norm(x)]
1.602502816811318e+13	1.132444176678409e-04	9.682873562616109e-05	1.482851977345812e-05

n = 12

[x	x1	x2	x3]
9.550881793675079e-01	9.550882240801253e-01	9.550882074981928e-01	9.550882353345431e-01
6.338195075234686e-01	6.338138527065653e-01	6.338112354278564e-01	6.338124363919878e-01
8.656963067695501e-01	8.658737891009578e-01	8.658180236816406e-01	8.659181018914640e-01
1.439299922296653e-01	1.415162836578187e-01	1.428833007812500e-01	1.409148715519637e-01
5.214287354077745e-01	5.390928916287441e-01	5.415039062500000e-01	5.434883208402863e-01
9.021265061079632e-01	8.246377933454059e-01	7.871093750000000e-01	8.053726863246543e-01
4.723857520562298e-01	6.879697519778982e-01	6.562500000000000e-01	7.415385691290839e-01
3.596461934920641e-01	-3.005083587595362e-02	0	-1.268519741515322e-01
8.173979290662142e-01	1.273680933166386e+00	1.218750000000000e+00	1.387002411182652e+00
6.388167754857278e-01	3.050458717275458e-01	2.968750000000000e-01	2.221566284169527e-01
5.580941649671349e-01	6.967102785795775e-01	6.875000000000000e-01	7.311348666684124e-01
1.240987541922712e-01	9.914993350800184e-02	1.005859375000000e-01	9.295360672581328e-02

[cond(H)	norm(x - x1)/norm(x)	norm(x - x2)/norm(x)	norm(x - x3)/norm(x)]
1.621163904747500e+16	3.323673346035590e-01	3.095944077743285e-01	4.149207132766070e-01

The workspaces are saved as q2_workspace_N={n}.mat corresponding to the values of n = 8, 10 and 12. In each of them x is the generated solution.

(a) In the first 2 cases the agreement of the number of significant digits for the exact and computed solutions in each case is up to at least 6 significant figures. However for n = 12, for some cases all 16 significant digits are lost.

(b) All x1,x2,x3 give pretty much the same results so there is no difference between the methods used to generate them.

(c) Yes the loss in accuracy agrees with the value predicted by the *Rule of thumb*. This is because we have cond(A) = 10^t, for t=16 and s=16, so the number of significant digits = s-t

Question 3.

The workspace is saved as q3_workspace.mat with the generated solution x.

We find that -

- norm(r)/norm(b) = 7.637682e-17
- norm(x - xt)/norm(x) = 6.389715e-05

Conclusion: *A small ‖r‖/‖b‖ does not imply that the perturbation in the solution ‖x−ˆx‖/‖x‖ will be small too.*

Question 4.

The workspaces are saved as q4_workspace_N={n}.mat corresponding to the values of n = 16 and 32. In each of them x is the generated solution.

n = 32

<u>GEPP method</u>

forward error norm(x - xcap,inf)/norm(x,inf): 5.140399e-09

cond(W): 1.421555e+01

norm(r,inf)/norm(b,inf): 2.897165e-09

<u>QR method</u>

forward error norm(x - xcap,inf)/norm(x,inf): 8.295685e-16

cond(W): 1.421555e+01

norm(r,inf)/norm(b,inf): 5.913120e-16

n = 64

<u>GEPP method</u>

forward error norm(x - xcap,inf)/norm(x,inf): 2.598663e+00

cond(W): 2.860298e+01

norm(r,inf)/norm(b,inf): 8.964456e-01

<u>QR method</u>

forward error norm(x - xcap,inf)/norm(x,inf): 2.071474e-15

cond(W): 2.860298e+01

norm(r,inf)/norm(b,inf): 6.782461e-16

(a) For both n, QR method gives a much smaller forward error O(10^-16)
(b) Also for both n, QR method gives a much lower value of $\|r\|_\infty/\|b\|_\infty$
(c) The rule of thumb is predicting correctly for the QR method as s = 16, t = 1 and s – t = 15. We have x^ correct upto 15 significant digits as seen in forward error.
(d) From the experiment we can say that the **QR method** with lower forward error and lower $\|r\|_\infty/\|b\|_\infty$ compared to GEPP **is more backward stable** algorithm.

Question 5.

The workspaces are saved as q5_workspace_N={n}_Norm={norm}.mat corresponding to the values of n = 20, 40, 60,..160 and norm = 1,2 and inf. In each of them A is the generated coefficient matrix.

The values of norms of L, U and the required ratios are computed for different values of n, and the corresponding results are tabulated in the table T of the workspace (too big to attach here).
Conclusion - The norm values for GENP are much larger than GEPP, for all n.