Lab Sheet 3

Report

Q1.

We see that the condition number varies steeply with the size of the matrix, it grows very quickly and in 1-Norm and Inf-Norm cases, MATLAB even generates warnings that the matrix is close to singular.

Q2.

s=16

For n= 8:

> cond(H)=1.52575755666280e+10 => t=10 => s-t=6
>
> norm(x,x1)/norm(x)=1.15902546907706e-07 <= 0.5e-06 => x and x1 agree to at least p=6 significant digits in their entries =>We lose 10 digits in computation but p = s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb
>
> norm(x,x2)/norm(x)=2.72088439903534e-07 <= 0.5e-06 => x and x2 agree to at least p=6 significant digits in their entries =>We lose 10 digits in computation but p = s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb
>
> norm(x,x3)/norm(x)=2.66000838391202e-07 <= 0.5e-06 => x and x3 agree to at least p=6 significant digits in their entries =>We lose 10 digits in computation but p = s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb

For n= 10:

> cond(H)= 1.60250281681132e+13 => t=13 => s-t=3
>
> norm(x,x1)/norm(x)= 3.87960365975955e-04 <= 0.5e-03 => x and x1 agree to at least p=3 significant digits in their entries =>We lose 13 digits in computation but p = s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb
>
> norm(x,x2)/norm(x)= 3.53494773231507e-04 <= 0.5e-03 => x and x2 agree to at least p=3 significant digits in their entries =>We lose 13 digits in computation but p = s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb
>
> norm(x,x3)/norm(x)= 4.74247171214652e-04 <= 0.5e-03 => x and x3 agree to at least p=3 significant digits in their entries =>We lose 13 digits in computation but p = s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb

For n= 12:

> **MATLAB even generates warnings that the matrix is close to singular**
>
> cond(H)= 1.62116390474750e+16 => t=16 => s-t=0

norm(x,x1)/norm(x)= 1.84776306362638e-01 <= 0.5e-0 => x and x1 agree to at least p=0 significant digits in their entries =>We lose 16 digits in computation but p >= s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb

norm(x,x2)/norm(x)= 2.12653093113878e-01 <= 0.5e-0 => x and x2 agree to at least p=0 significant digits in their entries =>We lose 16 digits in computation but p >= s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb

norm(x,x3)/norm(x)= 1.42179721015097e-01 <= 0.5e-0 => x and x3 agree to at least p=0 significant digits in their entries =>We lose 16 digits in computation but p >= s-t, hence loss of accuracy agrees with the value predicted by the Rule-of-thumb

So, loss of accuracy in all the cases agrees with the value predicted by the Rule-of-thumb. There isn't much difference between x1, x2, x3. In fact, we see that we get minimum error relative to x in x1 for n=8, x2 for n=10 and x3 for n=12. So we can't say which is better.

Q3.

||r||/||b||: 9.328815e-17

||x-x1||/||x||: 6.323861e-05 <= 0.5e-03

We see that even though r/b is very very small, x1 and x agree to only p=3 digits according to the previous question's rule. So, a small value of the norm of the residual is not enough to guarantee an accurate answer.

Q4.

n	MethodUsed	cond	forwardError	rBYb
"32"	"gepp"	"32"	"1.2218e-07"	"5.1733e-08"
"32"	"QR"	"32"	"3.5848e-16"	"3.735e-16"
"64"	"gepp"	"64"	"0.94184"	"0.37908"
"64"	"QR"	"64"	"1.7739e-15"	"5.4756e-16"

a) QR method gives lower forward error
b) s=16 and t=1, s-t=15
   For first forward error p=6<s-t
   For second forward error p=15>=s-t
   For third forward error p=-1<s-t
   For last forward error p=14>=s-t
   So, loss of accuracy in the case of QR method agrees with the value predicted by the Rule-of-thumb and loss of accuracy in the case of gepp method does not.
c) QR method gives lower value.
d) QR method has good backward stability because it gives low error and residual values and satisfies Rule-of-thumb.

Q5.

genp produces very large norm values.