HP Forum Archive 15

Ill conditioning and calculator precision

*Message #1 Posted by **Rodger Rosenbaum** on 7 Jan 2006, 9:54 p.m.*

Valentin Albillo gave an example a while ago of a very ill-conditioned linear system:

"If in doubt, you may want to consider this example I've set up, where the solution of some engineering problem requires solving this small 7x7 system of linear equations, where the coefficients are the result of some measurement, say Volts, with just one decimal of precision:

1.3 x1 + 7.2 x2 + 5.7 x3 + 9.4 x4 + 9.0 x5 + 9.2 x6 + 3.5 x7 = 45.3
4.0 x1 + 9.3 x2 + 9.0 x3 + 9.9 x4 + 0.1 x5 + 9.5 x6 + 6.6 x7 = 48.4
4.8 x1 + 9.1 x2 + 7.1 x3 + 4.8 x4 + 9.3 x5 + 3.2 x6 + 6.7 x7 = 45.0
0.7 x1 + 9.3 x2 + 2.9 x3 + 0.2 x4 + 2.4 x5 + 2.4 x6 + 0.7 x7 = 18.6
4.1 x1 + 8.4 x2 + 4.4 x3 + 4.0 x4 + 8.2 x5 + 2.7 x6 + 4.9 x7 = 36.7
0.3 x1 + 7.2 x2 + 0.6 x3 + 3.3 x4 + 9.7 x5 + 3.4 x6 + 0.4 x7 = 24.9
4.3 x1 + 8.2 x2 + 6.6 x3 + 4.3 x4 + 8.3 x5 + 2.9 x6 + 6.1 x7 = 40.7

which has the quite obvious, unique solution:

x1 = x2 = x3 = x4 = x5 = x6 = x7 = 1.0 (Volts)"

His point is well made, but I haven't seen much posted here about what one should do in a case like this. The HP48G can solve his example system, and gets the right answer. But, let's perturb the first element in the right hand column matrix so its value is 45.4 rather than 45.3; now the exact solution is:

Transpose[71083, -8, 63379, 63741, 45, -63738, -133356]

It doesn't seem reasonable that such a small perturbation should cause such a large change in the solution. This is what ill-conditioning does. What can we do about this? (I'll be rounding numerical results in what follows)

The statisticians have developed methods for dealing with this problem. Let the system to be solved be Ax = b, where A is the design matrix, Valentin's 7x7 matrix and b is the column matrix. The first thing to do is to calculate the correlation matrix of the A matrix; it is:

[1 .481 .77 .274 -.00652 -.033 .951]
[.481 1 .474 -.166 -.698 -.131 .391]
[.77 .474 1 .676 -.339 .526 .909]
[.274 -.166 .676 1 -.125 .924 .533]
[-.00652 -.698 -.339 -.125 1 -.347 -.052]
[-.033 -.131 .526 .924 -.347 1 .252]
[.951 .391 .909 .533 -.0519 .252 1]

We see that column 3 and column 7 are highly correlated, as are row 3 and row 7. I suspect this is a result of Valentin's method of construction.

One of the things a statistician would do is delete a row or column if it's highly correlated with another, and solve the reduced system. In this case if we drop row 7 (which is highly correlated with row 3) from A and b, the system can be solved as an underdetermined system (can your calculator do this easily?). The result is:

Transpose[1.09 .961, 1.01 .908 1.02 1.11 .964]

This result is much more reasonable and is actually not too (numerically) difficult to solve, because dropping the 7th row from A gave a matrix with a condition number of 189 (using the 2-norm) instead of the 3.17E12 condition number of the original A matrix.

Now for a small challenge. It is possible to perturb only two elements of the original A matrix, adding .1 to one and subtracting .1 from one, giving a result which is only a distance of .1414 (in R[7x7] hyperspace; subtract the two and compute the norm of the difference) away from the original. The perturbed matrix has a condition number of 399. Can you (and your calculator) find the perturbed matrix? If we solve the system with this matrix, get get a result:

Transpose[1.6 .961 1.47 1.37 1.02 .651 -.0000167]

This time we solved a full rank system, not an underdetermined one. The result is more reasonable than the unperturbed system, but we can do better.

Let's find a matrix which is a distance of .1 (in hyperspace) from the original, using at least 6 digits (up to 12, perhaps) to perturb ALL the elements of the original A matrix. The matrix of perturbations should have the minimum size (norm) to generate a new matrix just about .1 (in hyperspace) from the original. This perturbed matrix should have a condition number of about 399, and gives a full rank system which can be easily solved to give:

Transpose[1.09 .961, 1.01 .908 1.02 1.11 .964]

the same result we got when we deleted row 7 and solved the underdetermined system above. Can you and your calculator find the required perturbation matrix?

If we perturb elements of the Valentin's original A matrix and solve the system, we get wildly varying results, the result of the ill-conditioning. And this happens even if the perturbations are .5 LSD or less; this would be within the measurement error of the given numbers.

But if we use the matrix which is .1 away from the original to represent the original, we can perturb the elements by .5 LSD and still get results that are reasonably close to the unperturbed solution. It is backward error analysis championed by Wilkinson that leads us to this technique. A lot of this is explained in the HP15 Advanced Functions handbook.

How easy is this to do on your calculator? Can you guess which manufacturer's calculators make it easy? No, it's not Radio Shack.

Re: Ill conditioning and calculator precision

Message #2 Posted by ***Marcus von Cube, Germany*** *on 8 Jan 2006, 6:50 a.m.,*
in response to message #1 by Rodger Rosenbaum

Roger,

can you explain, in a few words, what a correleation matrix is and how it can be read. Another term asking for explanation is the matrix condition number. What do these numbers tell?

Marcus

Re: Ill conditioning and calculator precision

Message #3 Posted by ***Rodger Rosenbaum*** *on 8 Jan 2006, 8:25 a.m.,* *in response to message #2 by Marcus von Cube, Germany*

> Quote:
>
> Roger,
>
> can you explain, in a few words, what a correleation matrix is and how it can be read. Another term asking for explanation is the matrix condition number. What do these numbers tell?
>
> Marcus

In a few words, eh? Most of the later HP calcs had statistical functions which included the "corr" function. If you input a set of x,y pairs of data, the correlation function tells you if the x and y data sets are linearly related. The correlation matrix tells you all at once the same thing, pairwise, for the columns of a data matrix. The maximum absolute value for the correlation coefficient is 1, so looking at the correlation matrix can tell you which pairs (if any) of columns are (nearly) linearly related. If a pair of columns have a correlation coefficient nearly +-1, then the two columns carry nearly the same information, and you can probably delete one of them without changing the system solution much; you will be losing one variable in your model, but it's probably redundant.

The condition number tells you how numerically unstable the solution of the system will be with respect to perturbations in the system design matrix.

Use the search function at: http://mathworld.wolfram.com/
and you can find out more about these things.

Re: Ill conditioning and calculator precision

Message #4 Posted by ***Marcus von Cube, Germany*** *on 8 Jan 2006, 9:38 a.m.,*
in response to message #3 by Rodger Rosenbaum

Rodger,

I'm still trying to understand your answer fully.

This is the correlation matrix slightly reformatted:

```
[ 1       .481    .77     .274   -.00652 -.033    .951 ]
[ .481    1       .474   -.166   -.698   -.131    .391 ]
[ .77     .474    1       .676   -.339    .526    .909 ]
[ .274   -.166    .676    1      -.125    .924    .533 ]
[-.00652 -.698   -.339   -.125    1      -.347   -.052 ]
[-.033   -.131    .526    .924   -.347    1       .252 ]
[ .951    .391    .909    .533   -.0519   .252    1    ]
```

It is symmetric with respect to the diagonal and has just ones on the latter. This is clear, because each row is correlated perfectly to itself and the correlation shouldn't depend on the sequence of the rows (row 1 is correlated to row 5 in the same way as row 5 to row 1.) I can see, how to read the matrix for the correaltion between rows, but how do I see the correlation between columns? Is there another symmetry which I didn't detect yet?

> Quote:
>
> We see that column 3 and column 7 are highly correlated, as are row 3 and row 7.

Why did you choose rows 3 & 7. Isn't the correlation between rows 1 and 7 even higher? Can I say the same for the columns?

Marcus

Re: Ill conditioning and calculator precision

Message #5 Posted by ***Rodger Rosenbaum*** *on 8 Jan 2006, 1:02 p.m.,*
in response to message #4 by Marcus von Cube, Germany

> Quote:
>
> ---
>
> Rodger,
>
> I'm still trying to understand your answer fully.
>
> This is the correlation matrix slightly reformatted:
>
> ```
> [ 1       .481    .77     .274   -.00652 -.033    .951 ]
> [ .481    1       .474   -.166   -.698   -.131    .391 ]
> [ .77     .474    1       .676   -.339    .526    .909 ]
> [ .274   -.166    .676    1      -.125    .924    .533 ]
> [-.00652 -.698   -.339   -.125    1      -.347   -.052 ]
> [-.033   -.131    .526    .924   -.347    1       .252 ]
> [ .951    .391    .909    .533   -.0519   .252    1    ]
> ```
>
> It is symmetric with respect to the diagonal and has just ones on the latter. This is clear, because each row is correlated perfectly to itself and the correlation shouldn't depend on the sequence of the rows (row 1 is correlated to row 5 in the same way as row 5 to row 1.) I can see, how to read the matrix for the correaltion between rows, but how do I see the correlation between columns? Is there another symmetry which I didn't detect yet?

Actually, the matrix I have shown is for the correlation between columns. That is the convention for correlation matrices; they're always showing the correlation between columns. To see the correlation between rows, you need to transpose the original matrix before computing the correlation matrix. I should have shown another matrix for the row correlations. I did compute it; I just didn't display it. Sorry.

> Why did you choose rows 3 & 7. Isn't the correlation between rows 1 and 7 even higher? Can I say the same for the columns?

What is shown *is* the correlation between columns, as I mentioned above. I can't bring myself to type in the whole matrix of row correlations right now, but the row 3 to row 7 value is .999035, the highest of all the row or column correlations. You could type in those two rows as two-variable data in any late model HP calc and then use the "corr" function to see if you get the same value.

> Marcus
>
> ---

Re: Ill conditioning and calculator precision

Message #6 Posted by ***Marcus von Cube, Germany*** *on 8 Jan 2006, 3:17 p.m.,*
in response to message #5 by Rodger Rosenbaum

Rodger,

> Quote:
>
> ---

I can't bring myself to type in the whole matrix of row correlations right now, but the row 3 to row 7 value is .999035, the highest of all the row or column correlations. You could type in those two rows as two-variable data in any late model HP calc and then use the "corr" function to see if you get the same value.

I tried with the following value pairs:

(4.8, 0.3)
(9.1, 7.2)
(7.1, 0.6)
(4.8, 3.3)
(9.3, 9.7)
(3.2, 3.4)
(6.7, 0.4)

The correlation is about 0.6 so nowhere near 1 as you mentioned. Who is wrong?

Marcus

Re: Ill conditioning and calculator precision

Message #7 Posted by ***Rodger Rosenbaum*** *on 8 Jan 2006, 4:35 p.m.,*
in response to message #6 by Marcus von Cube, Germany

Quote:

Rodger,

I tried with the following value pairs:

(4.8, 0.3)
(9.1, 7.2)
(7.1, 0.6)
(4.8, 3.3)
(9.3, 9.7)
(3.2, 3.4)
(6.7, 0.4)

The correlation is about 0.6 so nowhere near 1 as you mentioned. Who is wrong?

It looks like you have got row 3 and row 6, not 3 and 7. For 3 and 6, I get .660085

Marcus

Re: Ill conditioning and calculator precision

Message #8 Posted by ***Marcus von Cube, Germany*** *on 9 Jan 2006, 2:35 a.m.,* *in response to message #7 by Rodger Rosenbaum*

Rodger, you are right, I can't count rows :-(

Let's try again with the correct data:

(4.8, 4.3)
(9.1, 8.2)
(7.1, 6.6)
(4.8, 4.3)
(9.3, 8.3)
(3.2, 2.9)
(6.7, 6.1)

The correlation is about 0.999. If you add the right hand sides (45.0, 40.7), the correlation jumps to 0.99998. This means, that the equations are *almost* linearly dependent and the system is in fact underdetermined.

Can you elaborate, how underdetermined linear systems can be tackled?

Marcus

Re: Ill conditioning and calculator precision

Message #9 Posted by ***Palmer O. Hanson, Jr.*** *on 9 Jan 2006, 4:35 a.m.,*
in response to message #8 by Marcus von Cube, Germany

Valentin issued the following challenge "Now, get your preferred HP calculator or computer software and try to solve it ... ... See what results you get and how they compare with the actual, unique solution." In the absence of other responses I decided to see what my stable of machines and software could do.

HP Machines

The only HP machines that I have which can handle this problem are the HP-41 and the HP-28S. I have two programs for the HP-41: the MATRIX routines in the Math Pac and Valentin's "seq" program from from V7N5P64 of the PPC Calculator Journal. The solutions for the two routines are

```
                 MATRIX                                 seq


        X                A*X                  X                A*X


    1.303525198         45.30000002         3.384562443         45.3
    0.999961733         48.4                 .999699401         48.40000002
    1.270628871         45.0                3.126121577         45.00000003
    1.272175068         18.60000001         3.138268917         18.60000002
    1.000187803         36.7                1.001475407         36.7
    0.727829119         24.9               -1.138236042         36.7
    0.4305555556        40.70000001        -3.473684211         40.70000001
```

where even though many of the calculated values of X are quite different from the exact solution of all ones, and some values of X are even of the wrong sign, the product of the matrix A and the solution vector X yields values which are never in error by more than 3e-8 and half are exact. I like to think of this situation as that the computer solution is correct in the computer's world even though it is far from correct in the mathematical world or in the real world..

Two solutions are directly available using the keyboard functions on the HP-28S: dividing the vector B by the matrix A, and multiplying the vector B by the calculated inverse of matrix A, where page 66 of the HP-28S Reference Manual states that Matrix B divided by Matrix A "uses 16-digit internal computation precision to provide a more accurate result than obtained by applying INV to A and multiplying the result by B.". The solutions for the two methods are

```
                      B/A                                             INV A * B


          X                      A*X                              X                 A*X


    -2.4919254825           45.3                            -15               -87.195614
     1.00049786612          48.4                              0.99753        -181.673058
    -2.521175647            44.0000000002                    21              -164.100568
    -2.5412934091           18.5999999999                    11                -7.815059
     0.997556494395         36.7                              0.99913        -143.847882
     4.54123894334          24.9                            -19.1             -18.666223
     8.40909090909          40.7                            -37.5            -146.267475
```

I admit that I may be doing something wrong with my HP-28S. I have checked and rechecked my matrix entries. Also, when I enter X values obtained from some non-HP machines and multiply A times X I get correct results.

Results from Some non-HP Machines

Radio Shack Model 100:

The best results were obtained with my Radio Shack Model 100 using a simultaneous equation solution adapted from the "Mathematics Library: Application Software for the Sharp EL-5500 and PC-1403 Scientific Computers" by Swinnen and Thomas. The best results were obtained using the default double precision mode of the Model 100 which offers fourteen digits. The Model 100 offers an easy way to observe the results which would be obtained with a shorter word length by defining all variables as single precision variables, essentially changing the Model 100 to a six digit computer. The actual results were

```
    14 Digit              6 Digit


0.99819033841188          12.5676
1.0000002281400             0.998543
0.99838647109897          11.3157
0.99837725241896          11.3747
0.99999888029820           1.00717
1.0016227226229          - 9.37452
1.0033951110401          -20.7060
```

For the 14 digit case the product of matrix A with the solution vector yields values which are within 2e-12 of the values in vector B. For the 6 digit case the product yields values which are within 6e-04 of the values in vector B.

The TI-85:

The simultaneous equation routine of the TI-85, which is also a fourteen digit machine, yields results which are almost as good as the double precision results from the Model 100

0.99570145279720
1.0000005419083
0.99616733305820
0.99614543563180
0.99999734033633

1.0038545050843
1.0080645161290

The matrix routines of the TI-59 and TI-95, thirteen digit machines, yield solutions which are substantially degraded relative to those from the Model 100 and TI-85. For both machines the product of matrix A and the solution vector X yields values which are within 3e-11 of the values in matrix B..

The CC-40 and TI-74:

The CC-40 and TI-74 operate in base 100. As a result they are not truly 14 digit machines since some values will be limited to 13 digits. Even so, I was surprised by the results from the matrix routines in the Mathematics modules of those machines:

```
      X            A*X


    1.45         43.48608
    0.9999       48.21909
    1.16         47.14495
    1            18.51455
    1.0002       38.2778
    0.64         23.90822
    1            42.6484
```

where the differences between the A*X values and the corresponding B values vary between 0.18 and 2.14 (no negative exponents). At least the A*X calculations indicate that the solution isn't acceptable. I kind of remember encountering something like this back when the CC-40 first came out and finding that the solution was obtained by calculating the product of the inverse of matrix A with vector B. Note that the HP-28S results from that technique were also very bad.

With a simultaneous equation program adapted from the "Mathematics Library: Application Software for the Sharp EL-5500 and PC-1403 Scientific Computers" by Swinnen and Thomas the TI-74 and CC-40 obtain a much better solution

0.99484589988833
1.000000649766
0.99540450571417
0.99537824997733
0.99999681097517
1.004621678939
1.009669621273

which is only slightly less accurate than the solutions from the full fourteen digit machines such as the Model 100 and TI-85. The product of matrix A and the solution vector X yields values which are within 4e-12 of the vector B values. Tentative conclusion: for ill-conditioned problems the matrix inversion method of solution is a bad way to go.

Re: Ill conditioning and calculator precision

Message #10 Posted by ***Rodger Rosenbaum*** *on 9 Jan 2006, 6:58 a.m.,*
in response to message #9 by Palmer O. Hanson, Jr.

"Tentative conclusion: for ill-conditioned problems the matrix inversion method of solution is a bad way to go. "

It would be interesting if you would try my suggestion of adding .1 to the first element of the column matrix, so that it is 45.4 rather than 45.3; I gave the exact solution in my first post.

With such a small perturbation, one would expect the solution to still be [1 1 1 1 1 1 1].

If the matrix inversion method is a bad way to go, how about some other method? Got any ideas?

Some results for the modified system

Message #11 Posted by ***Palmer O. Hanson, Jr.*** *on 10 Jan 2006, 4:56 a.m.,*
in response to message #10 by Rodger Rosenbaum

Rodger suggested that I redo the problem but with the b(1) value changed from 45.3 to 45.4. Some results follow.

HP-28S

```
                     B/A                                         INV A * B


        X                    A*X                    X                        A*X


 1010776.54039             45.399996             1010764.5                -86.78026
    -126.465124009         48.3999857               -126.681             -181.440996
  901227.742405            44.9999421             901251.3               -164.212648
  906376.678178            18.60000092            906390.2                 -7.617314
     626.42174829          36.6999899                626.42334           -143.940652
 -906360.718282            24.89999815         - 906384.3                 -18.533922
-1896317.61364             40.699875           -1896363.6                -146.364698
```

where the largest difference between the A*X and B values for the the B/A case 5.79e-5. The similarity of results for the solution matrix X by the two methods is more like other results that I recall when testing the 28S, say the 7x7 su-Hilbert with a vector of all ones. I don't recall seeing the lack of similarity between the A*X values, but I 'm not sure I did that calculation. I will have to search my files.

Again, I am concerned that I may be doing something wrong with my HP-28S.

Radio Shack Model 100:

Again, these results were obtained with my Radio Shack Model 100 using a simultaneous equation solution adapted from the "Mathematics Library: Application Software for the Sharp EL-5500 and PC-1403 Scientific Computers" by Swinnen and Thomas. The actual results were

```
     14 Digit                  6 Digit


 71057.6081486               45.1833
    -7.9967989123177          0.955531
 63356.360139385             40.3235
 63718.230789577             40.4464
    44.984289163027           1.04639
-62715.23113977             -38.4262
133308.36227832             -81.7585
```

For the 14 digit case the product of matrix A with the solution vector yields values which are within 3e-8 of the values in vector B. For the 6 digit case the product yields values which are within 8e-04 of the values in vector B even though the calculated X values have no obvious relationship to the expected X values..

Other Machines

As with Valentin's original problem the results from the CC-40 were not as good as with the Model 100 for the modified problem.

Results from the remaining machines mentioned in my earlier submission will take a little more time to obtain since for those machines I have to enter the entire problem rather than edit existing files.

Re: Some results for the modified system

Message #12 Posted by ***Rodger Rosenbaum*** *on 10 Jan 2006, 8:57 a.m.,*
in response to message #11 by Palmer O. Hanson, Jr.

Quote:

"Rodger suggested that I redo the problem but with the b(1) value changed from 45.3 to 45.4. Some results follow.

HP-28S

```
                     B/A                                         INV A * B


       X                    A*X                  X                        A*X


 1010776.54039            45.399996            1010764.5                -86.78026
    -126.465124009        48.3999857              -126.681             -181.440996
  901227.742405           44.9999421            901251.3               -164.212648
  906376.678178           18.60000092           906390.2                 -7.617314
     626.42174829         36.6999899               626.42334           -143.940652
 -906360.718282           24.89999815         - 906384.3                -18.533922
-1896317.61364            40.699875           -1896363.6               -146.364698
```

where the largest difference between the A*X and B values for the the B/A case 5.79e-5. The similarity of results for the solution matrix X by the two methods is more like other results that I recall when testing the 28S, say the 7x7 su-Hilbert with a vector of all ones. I don't recall seeing the lack of similarity between the A*X values, but I 'm not sure I did that calculation. I will have to search my files.

Again, I am concerned that I may be doing something wrong with my HP-28S."

No, you're ok. I get exactly the same thing on the HP48SX, except that the value -126.681 in the table above should be -126.4681, an obvious typo

"Radio Shack Model 100:

Again, these results were obtained with my Radio Shack Model 100 using a simultaneous equation solution adapted from the "Mathematics Library: Application Software for the Sharp EL-5500 and PC-1403 Scientific Computers" by Swinnen and Thomas. The actual results were

```
     14 Digit                  6 Digit


 71057.6081486               45.1833
    -7.9967989123177          0.955531
 63356.360139385             40.3235
 63718.230789577             40.4464
    44.984289163027           1.04639
```

```
-62715.23113977               -38.4262
133308.36227832               -81.7585
```

For the 14 digit case the product of matrix A with the solution vector yields values which are within 3e-8 of the values in vector B. For the 6 digit case the product yields values which are within 8e-04 of the values in vector B even though the calculated X values have no obvious relationship to the expected X values.."

I think the Radio Shack 100 is running a Microsoft Basic equivalent to the common GWBASIC. I think it is doing 56 bit arithmetic for the high precision calculations, which is equivalent to about 17 digit precision. Would you compute 1000000/7-142857 on the 100 and see what you get? From this we can tell how many equivalent digits are available.

Re: Some results for the modified system

Message #13 Posted by ***Palmer O. Hanson, Jr.*** *on 11 Jan 2006, 9:48 a.m.,*
in response to message #12 by Rodger Rosenbaum

For 1000000/7142857 I get .1400000028 .

I am not familiar with that test but I presume that the result means that the machine uses 14 digits.

I have tested the Model 100 years ago with a derivative of a Kahan program known as "Little Bits of Paranoia" which reported that it was a fourteen digit, base ten machine with a guard digit. Recently, when I decided to do some of the single precision work I tested in single precision and got six digits, base ten with a guard digit.

I have completed testing of your revised problem with the HP-41 Math Pak and with the TI-85. I should be able to get those results out tonight.

Re: Some results for the modified system

Message #14 Posted by ***Rodger Rosenbaum*** *on 11 Jan 2006, 10:03 a.m.,*
in response to message #13 by Palmer O. Hanson, Jr.

"For 1000000/7142857 I get .1400000028 .

I am not familiar with that test but I presume that the result means that the machine uses 14 digits."

Make that 1000000/7 - 142857

The exact result of 1000000/7 should be:
142857.142857142857142857142857......

Subtracting off the leading 142857 should leave 14-6 = 8 digits after the decimal point if you have a 14 digit machine, and 17-6 = 11 digits after the decimal point if you have a 17 digit machine.

If it's a 14 digit machine, you should get .14285714; if it's a 17 digit machine, you should get .14285714286, or something close to that.

Re: Some results for the modified system

Message #15 Posted by ***Palmer O. Hanson, Jr.*** *on 11 Jan 2006, 11:17 p.m.,*
in response to message #14 by Rodger Rosenbaum

Sorry about not seeing the mionus sign.

On the Model 100 1000000/7 - 142857 = .14285714 which says it is a fourteen digit machine. That agrees with my paranoia result.

Some more results

Message #16 Posted by ***Palmer O. Hanson, Jr.*** *on 12 Jan 2006, 9:24 a.m.,*
in response to message #15 by Palmer O. Hanson, Jr.

In his submission "Calculator Precision: Sure" of December 21 Valentin Albillo proposed the following set of linear equations for investigation of the effects of word length on the accuracy of the solution. His system has the unique solution:

x1 = x2 = x3 = x4 = x5 = x6 = x7 = 1.0

In a response Rodger Rosenbaum proposed examination of a modified set of equations where the only modification was the change of the B1 value from 45.3 to 45.4 The exact solution for the modified set of equations is

```
x1 =       71083
x2 =          -8
x3 =       63379
x4 =       63741
x5 =          45
x6 =      -63738
x7 =     -133356
```

My first response to Rodger's proposal included results obtained with the Radio Shack Model 100 and the HP-28S. Results from additional machines follow.

The HP-41 MATRIX Routine in the Math Pak with the Modified Set of Equations:

```
           X                  A*X


     -822.2517531          45.399997
        1.0649402          48.4
     -733.098423           45.0
     -737.3942465          18.6
        0.50966796         36.699997
      739.402556           24.9000016
     1545.638889           40.699995
```

Many of the calculated values of X are grossly different from the exact solution listed above; however, the product of the matrix A and the solution vector X yields values which are never in error by more than 5e-6 from the vector B values and three are exact.

The CC-40 and TI-74 with the Modified Set of Equations:

The CC-40 and TI-74 operate in base 100. As a result they are not truly 14 digit machines since some values will be limited to 13 digits. With a simultaneous equation program adapted from the "Mathematics Library: Application Software for the Sharp EL-5500 and PC-1403 Scientific Computers" by Swinnen and Thomas the CC-40 yielded the following results

```
x1 =    71675.9882
x2 =       -8.07475664
x3 =    63907.71966
x4 =    64272.7403
x5 =       45.36690284
x6 =   -64269.7322
x7 =  -134468.506
```

which is only slightly less accurate than the solutions from the full fourteen digit machines such as the Model 100 and TI-85. The product of matrix A and the solution vector X yields values which are within 7.9e-7 of the vector B values.

Some Thirteen Digit Results:

The TI-59 and TI-95 are thirteen digit machines. The TI-59 exhibits the famous anomaly in which multiplication is not always commutative, and in addition does not use an add/subtract guard digit. In the TI-95 multiplication is commutative and there is a guard digit. Solutions for Valentin' original problem from the ML-02 program in the Master Library module of the TI-59 and from the matrix routines in the Mathematics module for the TI-95 follow:

```
      TI-59                 TI-95


 1.740824864135         0.6835858242542
 0.9999066058071        1.000039889642
 1.660533625121         0.7178790660364
 1.664307494855         0.7162672061226
 1.000458374646         0.9998042233199
 0.3357027223239        1.283728430019
-0.3898635477583        1.593625498008
```

Both solutions are substantially degraded relative to those from the Model 100 and TI-85. For both machines the product of matrix A and the solution vector X yields values which are within 3e-11 of the values in matrix B.

How Many Exact Answers to Valentin's original problem from the HP-28S (or from any other machine)

Multiply the A matrix by a vector of all ones and it is not surprising that the result will be exactly the B matrix. Multiply the A matrix by a vector which includes the solution from the TI-95 rounded to twelve digits, Voila! The result is exactly the B matrix. I'm suggesting that on any machine there are probably an infinite number of vectors which can be multiplied by the A matrix and yield the exact B matrix.

Re: Some more results

Message #17 Posted by ***Rodger Rosenbaum*** *on 13 Jan 2006, 7:26 a.m.,*
in response to message #16 by Palmer O. Hanson, Jr.

Palmer said:

"The TI-59 and TI-95 are thirteen digit machines. The TI-59 exhibits the famous anomaly in which multiplication is not always commutative, and in addition does not use an add/subtract guard digit. In the TI-95 multiplication is commutative and there is a guard digit. Solutions for Valentin' original problem from the ML-02 program in the Master Library module of the TI-59 and from the matrix routines in the Mathematics module for the TI-95 follow:

```
      TI-59                 TI-95
```

```
1.740824864135          0.6835858242542
0.9999066058071         1.000039889642
1.660533625121          0.7178790660364
1.664307494855          0.7162672061226
1.000458374646          0.9998042233199
0.3357027223239         1.283728430019
-0.3898635477583        1.593625498008
```

Both solutions are substantially degraded relative to those from the Model 100 and TI-85. For both machines the product of matrix A and the solution vector X yields values which are within 3e-11 of the values in matrix B.

How Many Exact Answers to Valentin's original problem from the HP-28S (or from any other machine)

Multiply the A matrix by a vector of all ones and it is not surprising that the result will be exactly the B matrix. Multiply the A matrix by a vector which includes the solution from the TI-95 rounded to twelve digits, Voila! The result is exactly the B matrix. I'm suggesting that on any machine there are probably an infinite number of vectors which can be multiplied by the A matrix and yield the exact B matrix."

Valentin's original problem has, as he says, a unique solution. This is when doing continuous, traditional arithmetic. The discretized arithmetic of calculators spawns the spurious solutions of which you have given an example. Another is:

```
x1 = .220440311016
x2 = 1.00009827737
x3 = .304929663701
x4 = .300958473026
x5 = .999517658758
x6 = 1.6990307756
x7 = 2.46253405849
```

Notice that this one is even further from [1 1 1 1 1 1 1]. I was able to generate lots of them.

The product of A times the X you gave, and the one I gave, has non-zero digits out to the 15th place, and the loss of these in the rounding process makes it appear that the X is a solution, but only in the discretized arithmetic of the calculator; it's not a true solution.

However, I doubt that there are an infinite number of these false solutions. The HP28, for example, has a 12 digit mantissa; this means that it can represent all possible 12 digit integers, of which there are 10^12 positive and (10^12)-1 negative. The HP28 has 999 possible exponents. Therefore a total of about 2*999*10^12 numbers can be represented on it, somewhat short of infinity. :-)

But, back to the original purpose I had when I started this thread. I noticed that a slightly different perturbation of the column matrix has even greater effect. Change the third element of the column matrix in Valentin's original problem from 45 to 45.1--now the exact solution is:

```
x1 = -2128901625
x2 = 268387
x3 = -1898169427
x4 = -1909014362
x5 = -1317226
x6 = 1908985003
x7 = 3994038147
```

This is a tremendous change in the solution for an apparently insignificant change in one element.

Try something I described in an earlier post; add .1 to the {3,7} element of the A matrix, and subtract .1 from the {7,7} element. This tiny change reduces the condition number of the A matrix from about 3E12 to about 400, and using this slightly perturbed A matrix with the perturbed column matrix mentioned just above gives a HP48G solution:

```
x1 = 1.20999247798
x2 = 1.01325966115
x3 = 1.26616078717
x4 = 1.29157926088
x5 = .997263593179
x6 = .690061410742
x7 = .499897531935
```

This isn't as close to [1 1 1 1 1 1 1] as we would like, but given what the solution was with just the third element in the column matrix perturbed and using the original A matrix, this inprovement is an astounding result considering that only two tiny perturbations were made to the A matrix.

My purpose here is to show that with proper techniques we can find reasonable solutions to very ill-conditioned systems. Systems as badly conditioned as Valentin's example are unlikely to occur in practice; they must be artificially constructed. But, if we can handle *them*, then we can certainly handle the ones we encounter in real life problems.

Palmer, it would be good if you tried solving the system on your various calculators, consisting of the column matrix perturbed in the third element, and the {3,7} and {7,7} elements of the A matrix perturbed as I described. I think you will find that you don't get wildly varying results with these changes. For example, on my HP48S, which uses the same matrix routines as the HP28, I get about 9 or 10 digit agreement with the solution posted above, whether I use the divide key or invert the A matrix and multiply times the column matrix.

We've seen repeatedly how ill-conditioned systems are not correctly solved (exactly), and the more digits your calculator has, the better the results. This is to be expected. But now we should move on to new pastures and learn how to deal with these ill-conditioned systems, even with calculators that don't do arithmetic with lots of digits.

My favorite way to handle them involves the use of the singular value decomposition. Routines written in Fortran for the SVD can be found at:

http://www.netlib.org/lapack/single/

These could be most easily implemented on a machine that has Basic rather than keystroke programming. Or, you could just get an HP48G which has the SVD and lots more built-in. I find the HP48G very handy for experimenting with advanced matrix algebra.

The book, "Numerical Recipes", has a good section on the use of the SVD for dealing with ill-conditioned systems. I hope I can stir up some interest in this.

Re: Ill conditioning and calculator precision

Message #18 Posted by ***Rodger Rosenbaum*** *on 9 Jan 2006, 6:36 a.m.,*
in response to message #8 by Marcus von Cube, Germany

Quote:

Rodger, you are right, I can't count rows :-(

Let's try again with the correct data:

(4.8, 4.3)
(9.1, 8.2)
(7.1, 6.6)
(4.8, 4.3)
(9.3, 8.3)
(3.2, 2.9)
(6.7, 6.1)

The correlation is about 0.999. If you add the right hand sides (45.0, 40.7), the correlation jumps to 0.99998. This means, that the equations are *almost* linearly dependent and the system is in fact underdetermined.

The convention is that the terms "overdetermined" and "underdetermined" refer only to whether the system has more equations than unknowns, or fewer equations than unknowns. So Valentin's system is neither. But the fact that system exhibits near linear dependencies (another way to put it is to say that the system is near rank deficient) means that it is ill-conditioned. This can be true even *without* any high correlations, but high correlation can sometimes indicate where linear dependencies lie. In this case, it gave us the clue that dropping row 7 might be a good thing to do.

Can you elaborate, how underdetermined linear systems can be tackled?

To my mind, one of the most elegant methods uses the pseudoinverse. There is some discussion of the pseudoinverse at mathworld.wolfram.com.

But an underdetermined system can still be ill-conditioned, and may need to be attacked by techniques suggested by backward error analysis. What I like to do is to find another matrix only a small distance from the original A, with much smaller condition number and use *it* to solve the system.

The Lawson and Hanson reference at:
http://mathworld.wolfram.com/LeastSquaresFitting.html
is a good place to get detailed discussion of these things.

Marcus

Re: Ill conditioning and calculator precision

Message #19 Posted by ***Marcus von Cube, Germany*** *on 9 Jan 2006, 10:50 a.m.,*
in response to message #18 by Rodger Rosenbaum

Thanks for the answer!

It'll take some time to digest it all.

Marcus

[Return to Index | Top of Index]