♦MoHPC♠ *The Museum of HP Calculators*

HP Forum Archive 16

[Return to Index | Top of Index]

RREF on the HP-49 Revisited

Message #1 Posted by ***Palmer O. Hanson, Jr.*** *on 4 Apr 2006, 12:21 a.m.*

In a previous thread entitled "Even More Simultaneous Equation Solutions" I reported the following results from analyzing a problem by Valentin Albillo with three methods on an HP-49 :

```
         INV A * B              B/A                    RREF


       -1.076                 -0.6472              1.00148007239
        0.999786               1.00110019          0.999999813411
        0.407                 -0.7415              1.00131966086
        0.144                  0.6554              1.00132720057
        0.999124               0.9996398           1.00000091517
        0.553                  1.1592              0.998672819845
        0.61                   3.5385              0.997223232158
```

I noted that the RREF results were the first ones which were better than those I had obtained with a solver on my Model 100. Rodger Rosenbaum responded that he received different results with his HP-49.

```
          INV A * B                B/A                   RREF


          1.00018                1.00018             1.00124
          1.9043                 1.9043             -33231081.9872
          1.3779                 1.3779              4704.81157655
          1.00065                1.00065             4678.11919981
          0.7689                 0.7689              6770866.25288
          0.553                  0.553              -46565.78990985
         -0.785                 -0.785              -2226.81759925
```

After some playing around with the numbers I found that I can get a solution which looks very much like Rodger's RREF solution if I use a version of Valentin's problem in which all the elements have been multiplied by ten. The only differences are that my solution has the first element as 1.00124107685 and the seventh element as -4665.78990985 where Rodger's first element is a truncated version of my first element and his seventh element has a typographic error.

When I do a B/A solution using all of the elements of the problem multiplied by a factor of ten I get

```
   1.5286
   1.00018004
   1.9043
   1.3779
   1.0006513
   0.7689
  -0.785
```

where I note that six of my seven elements are the same as Rodger's if I allow for some truncation, but my elements are in a different order. For the element which is not the same my solution yielded 1.5286 while Rodger's solution yielded 0.553 . To date I have not been able to psych out why the different order of the elements together with one definitely different element occur.

Re: RREF on the HP-49 Revisited

Message #2 Posted by ***Rodger Rosenbaum*** *on 4 Apr 2006, 4:47 a.m.,*
in response to message #1 by Palmer O. Hanson, Jr.

Quote:

In a previous thread entitled "Even More Simultaneous Equation Solutions" I reported the following results from analyzing a problem by Valentin Albillo with three methods on an HP-49 :

```
         INV A * B              B/A                   RREF


       -1.076              -0.6472              1.00148007239
        0.999786            1.00110019          0.999999813411
        0.407              -0.7415              1.00131966086
        0.144               0.6554              1.00132720057
        0.999124            0.9996398           1.00000091517
        0.553               1.1592              0.998672819845
        0.61                3.5385              0.997223232158
```

I noted that the RREF results were the first ones which were better than those I had obtained with a solver on my Model 100. Rodger Rosenbaum responded that he received different results with his HP-49.

```
          INV A * B              B/A                   RREF


          1.00018               1.00018             1.00124
          1.9043                1.9043             -33231081.9872
          1.3779                1.3779              4704.81157655
```

```
        1.00065                1.00065             4678.11919981
        0.7689                 0.7689              6770866.25288
        0.553                  0.553              -46565.78990985
       -0.785                 -0.785              -2226.81759925
```

After some playing around with the numbers I found that I can get a solution which looks very much like Rodger's RREF solution if I use a version of Valentin's problem in which all the elements have been multiplied by ten. The only differences are that my solution has the first element as 1.00124107685 and the seventh element as -4665.78990985 where Rodger's first element is a truncated version of my first element and his seventh element has a typographic error.

When I do a B/A solution using all of the elements of the problem multiplied by a factor of ten I get

```
 1.5286
 1.00018004
 1.9043
 1.3779
 1.0006513
 0.7689
-0.785
```

where I note that six of my seven elements are the same as Rodger's if I allow for some truncation, but my elements are in a different order. For the element which is not the same my solution yielded 1.5286 while Rodger's solution yielded 0.553 . To date I have not been able to psych out why the different order of the elements together with one definitely different element occur.

Well, I went back and rechecked my results and you can stop psyching. I see that I wasn't very careful when I posted this stuff. The reason for some of the truncation is that I wasn't trying to post exactly what I got; just to show the substantial differences from what you got. I think it must have been very late! Let me try again. Here's what my ver 1.05 HP49 gets:

```
    INV A * B                B/A              RREF
      1.5286               1.5286          1.00124107685
      1.00018004           1.00018004     -33231081.9872
      1.9043               1.9043          4704.81157655
      1.3779               1.3779          4678.11919981
      1.0006513            1.0006513       6770866.25288
       .7689                .7689         -4665.78990985
      -.785                -.785          -2226.81759925
```

As I mentioned in another post, I can't find the serial cable for my HP49, so I won't be able to update the firmware. The 1.05 firmware definitely had bugs.

Re: RREF on the HP-49 Revisited

Message #3 Posted by ***Werner*** *on 5 Apr 2006, 4:16 a.m.,* *in response to message #1 by Palmer O. Hanson, Jr.*

Please, Palmer & Rodger, for B/A I get [1 1 1 1 1 1 1] exactly, on any version of 48G and 49G.

How do you obtain your results?? (for INV(A)*B and RREF I get your results)

Cheers, Werner

Re: RREF on the HP-49 Revisited

Message #4 Posted by ***Rodger Rosenbaum*** *on 5 Apr 2006, 8:48 a.m.,*
in response to message #3 by Werner

Palmer is using an HP49 emulator, and I used my HP49 with the old 1.05 firmware.

If I use my HP48G or HP49G+, I also get {1 1 1 1 1 1 1]. The HP49 apparently had bugs that gave these different results.

Re: RREF on the HP-49 Revisited

Message #5 Posted by ***Palmer O. Hanson, Jr.*** *on 6 Apr 2006, 3:31 a.m.,*
in response to message #3 by Werner

The sequence that I use for the B/A solution is to place B in the stack, place A in the stack and press divide. The answer I get depends upon what I have in A and B and on whether I have numeric or approx checked in the CAS display.

I can have A and B defined in three different ways. The RREF results tha I originally published were obtained with A and B defined as in Valentin's original challenge; i.e., with values defined to one decimal place. To get Rodger's original RREF results I used Valentin's challenge numbers multiplied by ten, but without decimal points. What that essentially meant is that when Rodger's original RREF results did not agree with mine it was because we weren't even solving the same problem. To paraphrase Cassius in Shakespeare's Julius Caesar "The fault lay not in our 49's but in ourselves." One can also get solutions with A and B including Valentin's challenge numbers multiplied by ten but keeping the decimal points. So, what is in the first row of your A matrix?

```
1.3    7.2    5.7    9.4    9.0 ...  as in Valentin's challenge, or


13     72     57     94    90 ... , or


13.    72.    57.    94.    90. ...
```

Finally, I suggest that some of Rodger's original numbers must have been obtained with columns (or rows) interchanged in a manner which would change the sign of the determinant.

Re: RREF on the HP-49 Revisited

Message #6 Posted by ***Rodger Rosenbaum*** *on 6 Apr 2006, 4:56 a.m.,*
in response to message #5 by Palmer O. Hanson, Jr.

Palmer said:

Finally, I suggest that some of Rodger's original numbers must have been obtained with columns (or rows) interchanged in a manner which would change the sign of the determinant.

I'm pretty sure that didn't happen, Palmer. I just now checked the matrix (Valentin's original), and it's in the calculator just as you posted it again at the beginning of this thread. I have done some of that sort of fooling around, but only on the HP49G+, never on the HP49. In fact I only got out the HP49 to compare with your results on the emulator; otherwise, I never use it.

As I've mentioned before, my HP49 has a very old firmware, ver 1.05, which had a lot of bugs. I'm sure the firmware your emulator is using must be much newer. In fact, why don't you type VERSION and see what version you have?

Re: RREF on the HP-49 Revisited

Message #7 Posted by ***Werner*** *on 6 Apr 2006, 8:38 a.m.,*
in response to message #6 by Rodger Rosenbaum

```
Rodger, I found out why you and Palmer get different results.
A or B is a type 29 array, not a type 3 array.
If both A and B are real arrays (type 3, like on the 48G), then
you'll get [1 1 1 1 1 1 1] as a result.
If either of them is a type 29 array (still with real items),
the result will be
[ -.6472
 1.00110019
 -.7415
 .6554
 .9996398
 1.1592
 3.5385 ]
I'm investigating why.
I've gotten to this point:
If either of the matrices are of type 29, the system is solved as
INV(A)*B.  Now, INV(A) is calculated as before, but the matrix-vector multiply
uses a different algorithm if one of its arguments is a type 29
matrix - even if all of its elements are reals.
Cheers, Werner
```

Re: RREF on the HP-49 Revisited

Message #8 Posted by ***Werner*** *on 6 Apr 2006, 9:32 a.m.,*
in response to message #7 by Werner

OK I know the gory details now ;-)

It's simple: if either of the input matrices is of type 29
(something you can't tell just by looking at it), the solution
is calculated as syMUL(INV(A),B) where syMUL stands for 'symbolic
multiply). For symbolic matrices containing real elements, it
boils down to a 12-digit DOT function, accumulating the inner
product starting with the *last* elements and moving to the beginning of the arrays.
You can simulate that in RPL as follows:

```
@ In: A nxn matrix
@     B vector of size n
@ Out: A*B, using reversed inner products
\<<
  OVER SIZE 1 GET
  \-> A B N
  \<<
    1 N FOR I
      0.
      N 1 FOR J
        'A(I,J)' EVAL B J GET * +
      -1 STEP
    NEXT
    N \->ARRY
  \>>
\>>
```

with INV(A) and B as arguments, this yields the exact same result
as doing A/B with type 29 arrays.

Now, I may understand why this particular decision has been taken;
What I don't understand is that you can have two different types
of arrays containing the same elements, yet obtain different
results - and no visual clue as to what type of array you're
actually working with.
BTW it's easy to transform a type 29 matrix into a type 3: \->NUM will do the trick.

```
Cheers, Werner
```

Re: RREF on the HP-49 Revisited

*Message #9 Posted by **Rodger Rosenbaum** on 6 Apr 2006, 12:53 p.m.,*
in response to message #8 by Werner

Just what is a type 29 array? It must something other than "exact" numbers, since you can tell those by looking. How does one input a bunch of numbers and get a type 29 rather than a type 3?

I'm being lazy here; rather than go look it up myself, I'm picking your brain, since you've already done it! :-)

Re: RREF on the HP-49 Revisited

*Message #10 Posted by **Werner** on 6 Apr 2006, 2:10 p.m.,*
in response to message #9 by Rodger Rosenbaum

Type 29 arrays have been created to allow arrays of other types besides real and complex numbers, or more specifically, to allow symbolic arrays. Internally, they are implemented as lists of lists with all elements of the same type, and all sublists the same length. (the prologue is not DOLIST of course, but DOMATRIX, but that aside, it's the same thing.)

Type 29 arrays are created when you enter an array with anything but reals; So, if you enter [5] in exact mode, you end up with a type 29 array. However, if you then do 1. * you get a type 29 array filled with reals (for some reason I prefer this to \->NUM - but now I know that it's not the same thing! and of course you can't tell by its looks that it's type 29, not type 3.

Another way to obtain it is to do AXL on a list-of-lists (AXL apparently always returns type 29 arrays - all it has to do is run some checks and change the prologues)

So, always do a \->NUM before doing numerical Linear Algebra on a 49.

Cheers, Werner

Re: RREF on the HP-49 Revisited

*Message #11 Posted by **James M. Prange (Michigan)** on 7 Apr 2006, 5:45 a.m.,*
in response to message #9 by Rodger Rosenbaum

Well, a "real array" (type 3) is an "atomic" object; it has one prologue for all of its elements, and each element is the "body" of a real number.

Similarly, a "complex array" (type 4) is an atomic object with one prologue for all of its elements, with each element being the body of a complex number.

But a "symbolic matrix" (type 29) is a "composite" object, containing other objects, each complete with its own prologue, so the elements can be various types; at least "real" numbers (type 0), complex numbers (type 1) , global names (type 6), local names (type 7), algebraic objects (type 9), and "exact integers" (type 28), or any mixture of these.

A symbolic matrix can be converted to a real array only if all of it's elements can be evaluated (or converted) to real numbers, or to a complex array only if all of it's elements can be evaluated to complex numbers.

By the way, if you convert an exact integer with over twelve digits (exclusive of trailing zeroes), or over 500 digits including trailing zeroes, to a real number, then it will be rounded.

Internally, a symbolic matrix is much like a list or a list of lists, although of course it has its own prologue, there are restrictions on which object types it can contain, and each row must be the same length. See the AXL command for converting arrays to lists, and suitable lists to symbolic matrices.

The distinction between composite and atomic objects is also relevant for the size of the array, and having many objects extracted from a symbolic matrix on the stack causes the same kind of garbage collection slow-down as happens with lists.

Regards,
James

Edited: 7 Apr 2006, 8:48 p.m.

Re: RREF on the HP-49 Revisited

*Message #12 Posted by **gene** on 6 Apr 2006, 8:38 a.m.,*
in response to message #6 by Rodger Rosenbaum

the emulator I'm using gets palmer's results. Has ROM 2.05 in it.

Re: RREF on the HP-49 Revisited

*Message #13 Posted by **James M. Prange (Michigan)** on 7 Apr 2006, 3:32 a.m.,* *in response to message #5 by Palmer O. Hanson, Jr.*

Well, these threads are beyond my math skills, so I don't know whether it has anything to do with it, but maybe try changing flag -126. Clear: rref w/ last col, set: rref w/o last col.

Regards,
James

It's my turn!

Message #14 Posted by ***Palmer O. Hanson, Jr.*** *on 7 Apr 2006, 10:41 p.m.,*
in response to message #13 by James M. Prange (Michigan)

It's my turn to ask "Am I doing something wrong?" Or, maybe "Am I doing something that that in my Navy days we used to call DUMB S _ _ _ ; namely knowing better, but doing it anyway?"

I have the times ten version of Valentin's problem in the calculator as integers without decimal points; i.e. I made the matrix and vector in numeric mode without entering decimal points. I have the calculator in numeric mode. I enter the matrix A and invert it. I enter the vector B and change it to the decimal point format by multiplying by 1. I hit multiply which should give me INV A * B. the answer I get is

[1. 1. 1. 1. 1. 1. 7.]

Where did the seven come from?

I did have some good fortune this week. I was at a rummage sale and purchased a early version Made in the USA, near mint condition HP-12C with a manual for two dollars.

Re: It's my turn!

Message #15 Posted by ***Rodger Rosenbaum*** *on 8 Apr 2006, 2:45 a.m.,*
in response to message #14 by Palmer O. Hanson, Jr.

> Quote:
>
> ---
>
> It's my turn to ask "Am I doing something wrong?" Or, maybe "Am I doing something that that in my Navy days we used to call DUMB S _ _ _ ; namely knowing better, but doing it anyway?"
>
> I have the times ten version of Valentin's problem in the calculator as integers without decimal points; i.e. I made the matrix and vector in numeric mode without entering decimal points. I have the calculator in numeric mode. I enter the matrix A and invert it. I enter the

> vector B and change it to the decimal point format by multiplying by 1. I hit multiply which should give me INV A * B. the answer I get is
>
> [1. 1. 1. 1. 1. 1. 7.]
>
> Where did the seven come from?

Are you doing this on the HP49 emulator? I'm really beginning to mistrust the HP49 arithmetic.

On my HP49G+, I can multiply A and B by 10, 100, 1000 or divide them by 10, 100, 1000 and solve by INV(A)*B, or B/A, or RREF and I always get exactly the same result (for each method).

By the way Palmer, if you have B A on the stack, another method of solution available on the HP48G, HP49 and HP49G+ is the command LSQ; it is intended for least squares problems, but it works just fine for a non least squares system.

Re: It's my turn!

*Message #16 Posted by **Werner** on 8 Apr 2006, 2:58 a.m.,*
in response to message #14 by Palmer O. Hanson, Jr.

Hi Palmer. It's a rounding error, no less.

Remember that multiplication of a symbolic array with real numbers is done using 12-digit arithmetic.

The last row of INV(A) reads:

[-133357 952047 3994038146 68560103 -497464609 -6667765 -3995675528]

and 'B':

[453 484 450 186 367 249 407]

Now, 407*-3995675528 = -1626239939896, a thirteen-digit number that is rounded to twelve digits, and that's the cause of the difference in 7 and the correct 1.

I have the full 'dot' worked out as well if you want to see it..

Cheers, Werner

Re: It's my turn!

Message #17 Posted by **Rodger Rosenbaum** *on 8 Apr 2006, 6:36 a.m.,*
in response to message #16 by Werner

It's the old type 29 versus type 3 thing again, eh?

Werner, are you doing these calculations on a real HP49, HP49G+ or an emulator?

This type 29 vs. 3 thing explains some of these latest puzzling behaviors, but there is definitely something wrong with my HP49 (old 1.05 firmware). For example, the determinant of Valentin's matrix (it's a type 3 object, and I triple checked that it's typed in correctly) as calculated on the HP49 is -1.00000533622E-7, but on the HP49G+ and HP48G, it's 9.98918882E-8. But, Hilbert matrices up to order 15 give exactly the same result for DET on my old HP49 and my HP49G+. Passing strange.

Palmer, and Werner, what do you get for this determinant (of Valentin's matrix) on your HP49 (real or emulated)?

Re: It's my turn!

Message #18 Posted by **Werner** *on 8 Apr 2006, 11:46 a.m.,*
in response to message #17 by Rodger Rosenbaum

Rodger, I use the emulator most of the time, but just for double-checking, I got out my real 49G (1.19-6)

They all get 9.98918882E-8.

BTW, to quadruple-check your matrix, do 'BYTES' on it:

# 29Ah check

407. bytes

(That's for the [[1.3 7.2 5.7 ...]] matrix)

Cheers, Werner

Re: It's my turn!

Message #19 Posted by **Rodger Rosenbaum** *on 8 Apr 2006, 6:30 p.m.,*
in response to message #18 by Werner

All of this discussion will be for the HP49, except where otherwise noted.

Well, the problem is the type 29/3 bug. I had thought this might be the problem when you first posted about it, and I checked the type of the matrix and it was 3. But when I checked the checksum, it was wrong. Looking carefully at the matrix, I noticed that some of the entries didn't have a decimal point, such as the {2 1} entry. Examining it further, I found that all the numbers that were single digit had no decimal point (except the single digit numbers that had a decimal in front, such as .1). Obviously, the calculator had been in exact mode when I typed in the matrix, and I didn't type a decimal point for the single digit numbers like the 4 in the {2 1} position.

In spite of the presence of these apparently "exact" integers, the type of the matrix is 3. To try to find out what's happening, I typed in a small matrix, [[1 2.][3. 4.]], and checked its type; the type was 29. Some more testing seemed to indicate that if a single element of an otherwise approximate matrix is exact, then the type of the matrix is 29. So why is my matrix type 3 when there are several "exact" elements?

Next, I put the matrix on the stack and executed ->ROW and checked the type of each row. The type of the individual rows was type 29(!}, *except row 2*, for which the type was 3. I then decomposed the matrix with ->COL and checked the type of the columns. They were all 29 except for column 5.

I put the calculator into exact mode, and typed in the matrix all over again, without typing a decimal point for the single integer entries. This matrix also exhibited the behavior I described above.

So, the type of a matrix isn't *always* 29 if there are some "exact" elements in it (at least, not on my HP49}.

Some more testing (entering these objects with the calculator in exact mode), gave the following results:

```
Object              Type


[[1. 2]]             29
[[11. 2]]            3
[[2 11.]]            29
[[1.][2]]            29
[[11.][2]]           3
[[2][11.]]           29
```

The HP49G+ gave none of this anomalous behavior. If a matrix contains even one exact element, the whole matrix is type 29.

Re: It's my turn!

Message #20 Posted by ***James M. Prange (Michigan)*** *on 8 Apr 2006, 9:30 p.m.,*
in response to message #19 by Rodger Rosenbaum

Quote:

> So, the type of a matrix isn't *always* 29 if there are some "exact" elements in it (at least, not on my HP49}.
>
> Some more testing (entering these objects with the calculator in exact mode), gave the following results:
>
> ```
> Object               Type
>
>
> [[1. 2]]              29
> [[11. 2]]             3
> [[2 11.]]             29
> [[1.][2]]             29
> [[11.][2]]            3
> [[2][11.]]            29
> ```
>
> The HP49G+ gave none of this anomalous behavior. If a matrix contains even one exact element, the whole matrix is type 29.

Presumably it's a bug with your old revision #1.05 ROM.

Testing on my 49Gs, both with the old ROM "Version HP49-C Revision #1.18" and with "Version HP49-C Revision #2.00"

```
Object               Type


[[1. 2]]              29
[[11. 2]]             29
[[2 11.]]             29
[[1.][2]]             29
[[11.][2]]            29
[[2][11.]]            29
```

Regards,
James

Edited: 8 Apr 2006, 9:38 p.m.

Re: It's my turn!

Message #21 Posted by ***Werner*** *on 9 Apr 2006, 2:36 p.m.,*
in response to message #19 by Rodger Rosenbaum

Hi Rodger, can you do me a favor and for the matrix

[[11. 2]] (that is of type 3 in your 49)

return the result of ->H ? (in 256 MENU)

e.g. in the emulator (where it is type 29), the result is:

686206862033920100000000000001102C372B2130B2130

or

```
02686 DOMATRIX
02686 DOMATRIX
02933 DOREAL
0110000000000001 1.1e01
273C2 ZINT 2 (pointer to ROM object)
B2130 SEMI
B2130 SEMI
```

Cheers, Werner

Re: It's my turn!

Message #22 Posted by ***Rodger Rosenbaum*** *on 10 Apr 2006, 8:41 p.m.,*
in response to message #21 by Werner

Quote:

Hi Rodger, can you do me a favor and for the matrix

[[11. 2]] (that is of type 3 in your 49)

return the result of ->H ? (in 256 MENU)

e.g. in the emulator (where it is type 29), the result is:

686206862033920100000000000001102C372B2130B2130

or

```
02686 DOMATRIX
02686 DOMATRIX
02933 DOREAL
0110000000000001 1.1e01
273C2 ZINT 2 (pointer to ROM object)
B2130 SEMI
B2130 SEMI
```

Cheers, Werner

I got: 8E920930003392020000100002000010000000000001100000000000000020

Re: It's my turn!

Message #23 Posted by ***Werner*** *on 12 Apr 2006, 3:26 a.m.,*
in response to message #22 by Rodger Rosenbaum

Thanks, Rodger!

```
029E8 DOARRY
00039 length field
02933 type (DOREAL)
00002 nr of dimensions
00001 1st dim
00002 2nd dim
0110000000000001 11.
0200000000000000  2.
```

looks perfectly normal. This means the bug is in the display driver for that object type.

Still, I wonder how you can then get a value for DET(A) that I can only get using a type 29. matrix with exact elements (ZINTs), evaluated in approx mode.

Either way, we are left with essentially the same matrix that can return one of three values as its DET, depending on how you entered it, your CAS settings and the status of flag -54. Ironically, the best answer is obtained in approx mode with reals. (and of course also in exact mode with ZINTs)

Sigh

Cheers, Werner

Re: It's my turn!

Message #24 Posted by ***Rodger Rosenbaum*** *on 12 Apr 2006, 4:10 a.m.,*
in response to message #23 by Werner

> Quote:
>
> (Snip)
>
> Still, I wonder how you can then get a value for DET(A) that I can only get using a type 29. matrix with exact elements (ZINTs), evaluated in approx mode.
>
> (Snip)
>
> Sigh
>
> Cheers, Werner

I get the result 1.00000533622E-7 when I use the matrix containing all exact elements, evaluated in approx mode on my HP49G+. But on my HP49 when using the funky version of the matrix with the several exacts mixed in with mostly approximate elements, I get -1.00000533622E-7. The minus sign is really there. On comp.sys.hp48, a search of old postings indicates that there were lots of bugs in the 1.05 firmware.

Re: It's my turn!

Message #25 Posted by ***Palmer O. Hanson, Jr.*** *on 10 Apr 2006, 2:02 a.m.,*
in response to message #17 by Rodger Rosenbaum

Rodger:

I apologize for the delay in responding to your request for determinant values. I have been busy packing for our move to western North Carolina for the summer.

There are four combinations of numeric and approx which can be set in CAS mode. both, approx, numeric, neither:

For Valentin's original matrix A I get .0000001 in each of the four modes.

For Valentin's matrix A multiplied by ten (with decinal points) I get 1. in each of the four modes.

For Valentin's matrix A multiplied by ten (without decimal points; i.e., assembled in numeric mode) I get the answer 1 (no decimal point) in numeric only or with neither numeric or approx set. I get 1.00000533622 (plus, not the minus you reported) with approx only or with approx and numeric both set. It was the difference in our signs that suggested to me that you may have interchanged some columns or rows.

Re: It's my turn!

Message #26 Posted by ***Rodger Rosenbaum*** *on 10 Apr 2006, 4:35 a.m.,*
in response to message #25 by Palmer O. Hanson, Jr.

> Quote:
>
> Rodger:
>
> For Valentin's matrix A multiplied by ten (without decimal points; i.e., assembled in numeric mode) I get the answer 1 (no decimal point) in numeric only or with neither numeric or approx set. I get 1.00000533622 (plus, not the minus you reported) with approx only or with approx and numeric both set. It was the difference in our signs that suggested to me that you may have interchanged some columns or rows.

My HP49 really did get that difference in sign, but now that Werner has clarified the type 29/3 problem, and the fact that my HP49 has that old 1.05 firmware, I'm not surprised. I'm not going to be using my HP49 for any of this kind of thing anymore, after I answer Werner's request. It's just too buggy, and I can't find the serial cable to update it (assuming I ever had one).

Re: It's my turn!

Message #27 Posted by ***Werner*** *on 10 Apr 2006, 5:29 a.m.,*
in response to message #25 by Palmer O. Hanson, Jr.

Palmer, there's another variable to consider: the status of flag -54 (use tiny element). I suspect it is Clear in your case (as it is per default).

With the flag clear, the calculator will fiddle with the calculated value of the determinant (getting the exact results in your case). With the flag set, it will leave the calculated value alone, and you get .998918882 in the first four cases.

Cheers, Werner

Re: It's my turn!

Message #28 Posted by ***Palmer O. Hanson, Jr.*** *on 15 Apr 2006, 10:42 a.m.,*
in response to message #27 by Werner

What terrifies me about all of this is that there are too many options which too frequently really aren't needed but which if inadvertently set can lead to unhappy results. I don't think that I will be investing in an HP-49. I'll keep the emulator on my computer primarily for the availability of the exact mode.

Question: It occurred to me that the TI-89 may be just as bad? Does anyone know?

Re: It's my turn!

Message #29 Posted by ***Rodger Rosenbaum*** *on 15 Apr 2006, 9:56 p.m.,*
in response to message #28 by Palmer O. Hanson, Jr.

> Quote:
>
> What terrifies me about all of this is that there are too many options which too frequently really aren't needed but which if inadvertently set can lead to unhappy results. I don't think that I will be investing in an HP-49. I'll keep the emulator on my computer primarily for the availability of the exact mode.
>
> Question: It occurred to me that the TI-89 may be just as bad? Does anyone know?

Not to worry; this kind of thing only happens when the condition number of the matrix is very high. The condition number of Valentin's A matrix is about 3E12. The rule of thumb is that you lose LOG10(condition number) digits in your computations; in this case that says that you lose 12 digits. It's a wonder you get anything at all!

You shouldn't be trying to solve a system with a condition number of 3E12 with elimination algorithms, which is what B/A, RREF and INVERSE(A)*B all do. Try out some of those little programs I posted to help you use the SVD on your HP49 emulator.

Here's a system with an A matrix having the same determinant as Valentin's, but with a much smaller condition number. Play around with this for a change.

```
                  A                      b


 [[ 6.6 1.9 6.3 8.1 7.1 6.4 6.1 ]   [ 101.9 ]
```

```
   [ 6.4 2.1 6.1 7.2 6.9 6.2 6.0 ]   [  96.8 ]
   [ 5.6 1.8 5.3 6.4 5.9 5.4 5.2 ]   [  84.4 ]
   [ 5.7 1.9 5.6 6.2 6.5 5.7 5.2 ]   [  87.2 ]
   [ 3.9 1.3 3.6 4.4 3.9 3.7 3.7 ]   [  57.7 ]
   [ 8.4 2.7 7.8 9.7 8.5 8.0 7.9 ]   [ 125.4 ]
   [ 7.4 2.4 7.2 8.3 8.3 7.3 6.8 ]]  [ 113.3 ]
```

Notice that you get good results no matter what the setting of flag -54, or what method you use.

[Return to Index | Top of Index]

GTO Go back to the main exhibit hall