HP Forum Archive 14

[Return to Index | Top of Index]

e^A where A is a Matrix

Message #1 Posted by ***Harry*** *on 2 Dec 2004, 4:57 p.m.*

Ok, I didn't know this calculation is possible until a few month ago. But it realy makes sence (or sense? Whats the correct spelling there?) to calculate e^A and I need it for an exam at univerity. So how do I get my HP42s or Hp48 to do that? The HP48 says "Bad Argument Type"

Regards, Harry from Germany

Re: e^A where A is a Matrix

Message #2 Posted by ***Vieira, Luiz C. (Brazil)*** *on 2 Dec 2004, 5:36 p.m.,*
in response to message #1 by Harry

Hi, Harry;

I'm curious, now. Is there any e-place where one (I, in this case) can read about e^A? I was also not aware of the fact it is possible (makes sense... the correct spelling, as you asked for), too.

If there is no e-place that you know there is additional info, could you point out the calculations involved in e^A? I'd like to try a program in the HP42S, if applicable.

Best regards.

Luiz (Brazil)

Re: e^A where A is a Matrix

Message #3 Posted by ***Harry*** *on 2 Dec 2004, 6:17 p.m.,*
in response to message #2 by Vieira, Luiz C. (Brazil)

As we know e^x = 1 + x/1! + x^2/2! + ... + x^n/n!

Now e^A is defined the same way. This gives a very easy result for a diagonal matrix (not sure if this is the correct word, I don't usualy talk about math in english :)):

Things aren't that easy anymore for other nxn matrizes, because then you will have to calculate A^2, A^3 and so on until you get to a point where your result is good enough.

I hope you understand what I am trying to tell you. I didn't talk about mathmatics in english a lot before, and I need to practice that :)

Re: e^A where A is a Matrix

Message #4 Posted by ***Harry*** *on 2 Dec 2004, 6:25 p.m.,*
in response to message #3 by Harry

I think I will try and write a program for that too. Maybe first step would be to write a a program that calculates e^x. I'm am not sure how many steps I need to get a result that is correct for all displayed digits. Does anyone know how the HP calculaters internally calculate e^x? Do they also use e^x= 1 + x^1/1! + x^2/2! + ... ?

Best regards, Harry

Re: e^A where A is a Matrix

Message #5 Posted by ***Veli-Pekka Nousiainen*** *on 2 Dec 2004, 7:19 p.m.,*
in response to message #4 by Harry

49 series (48gII, 49G, 49g+): MAP
[VPN]

Re: e^A where A is a Matrix

Message #6 Posted by ***Harry*** *on 2 Dec 2004, 7:26 p.m.,*
in response to message #5 by Veli-Pekka Nousiainen

Does that mean I finaly found a reason to buy a 49G+?!?

Thanks for the info!

Re: e^A where A is a Matrix

Message #7 Posted by ***Karl Schneider*** *on 2 Dec 2004, 10:52 p.m.,*
in response to message #5 by Veli-Pekka Nousiainen

The original question was, which HP calculators can perform exponentials of *(square)* matrices.

So, the "49 series (48gII, 49G, 49g+): MAP" can? I assume that MAP stands for "Mathematical Applications Package" or similar. My unenhanced 49G also returns, "EXP Error: Bad Argument Type".

BTW, I thought that my 42S did square-matrix exponentials. However, doing e^x on any matrix simply takes the natural antilog of each element.

-- KS

Edited: 2 Dec 2004, 10:55 p.m.

Re: e^A where A is a Matrix

Message #8 Posted by ***Harry*** *on 3 Dec 2004, 12:15 a.m.,*
in response to message #7 by Karl Schneider

BTW, I thought that my 42S did square-matrix exponentials. However, doing e^x on any matrix simply takes the natural antilog of each element.

This is exactly what the HP42 does. But it is not what e^A really is by the definition I gave in a previous post. An easier example for this is the following. If you got a matrix in the stack of the HP42 and then press 1/x, it doesn't calculate the invers of that matrix, but it it calculates the invers of each element of the matrix. So if you have a 0 in your matrix you get a "division by zero". Its the same with e^x, it also works element by element.

This is a rather funny behavior an quite confusing, as the HP48 for example calculates the invers of the matrix after pressing 1/x (wich i think the HP42s should do as well insted of calculating the invers of each element)

Best Regards, Harry

Re: e^A where A is a Matrix

Message #9 Posted by ***Imad Eddine Srairi*** *on 3 Dec 2004, 2:03 a.m.,*
in response to message #8 by Harry

Hi,

Notice that Harry's calculations applies to any matrix which may be diagonalized, that is which may be written as A=P^-1 . D . P , where P is inversible and D only has diagonal terms (the eigenvalues). In this case, A^n=P^-1 . D^n . P^and D^n is easy to compute; exp(A) is therefore easy to compute too since its elements are simple the exponential of each eagenvalue i.e. denoting by li the eigenvalues

exp(A) = P^-1 . Diag (exp(l1), ..., exp(ln)) P.

I think that some HP calculators can compute the eigenvalues and anyway there are programs to achieve that. P is a basis of linearly independant eigenvectors. The only hard problem is finding whether it is possible to diagonalize A or not. Sometimes we easily know we can, for instance when A is symetric and positive (that is (tX).A.X is positive for any X - in this case all eigenvalues are positive).

Hope it helps.

Best regards,

Imad

Re: e^A where A is a Matrix

Message #10 Posted by ***Axel*** *on 3 Dec 2004, 3:01 a.m.,*
in response to message #9 by Imad Eddine Srairi

MAP applies a function on each element of a matrix. Hence, MAP([[a,0],[0,b]],EXP[x]) would give [[e^a,1],[1,e^b]] and not [[e^a,0],[0,e^b]].

Greetings Axel

Re: e^A where A is a Matrix

Message #11 Posted by ***Harry*** *on 3 Dec 2004, 6:16 a.m.,*
in response to message #10 by Axel

So that means it is no improvement over the 42s. Thanks for that info.

Harry

Re: e^A where A is a Matrix

Message #12 Posted by ***Harry*** *on 3 Dec 2004, 6:31 a.m.,*
in response to message #9 by Imad Eddine Srairi

Hi,

Notice that Harry's calculations applies to any matrix which may be diagonalized, that is which may be written as A=P^-1 . D . P , where P is inversible and D only has diagonal terms (the eigenvalues). In this case, A^n=P^-1 . D^n . P^and D^n is easy to compute; exp(A) is therefore easy to compute too since its elements are simple the exponential of each eagenvalue i.e. denoting by li the eigenvalues

exp(A) = P^-1 . Diag (exp(l1), ..., exp(ln)) P.

I think that some HP calculators can compute the eigenvalues and anyway there are programs to achieve that. P is a basis of linearly independant eigenvectors. The only hard problem is finding whether it is possible to diagonalize A or not. Sometimes we easily know we can, for instance when A is symetric and positive (that is (tX).A.X is positive for any X - in this case all eigenvalues are positive).

Hope it helps.

Yes, it does help!

I didn't think about diagonalizing the matrix. It is a good way if it can be diagonalized. But that does not work if the matrix is not of full rank, then P can't be calculated as it consists of the eigenvektors of A.

Best regards, Harry

Re: e^A where A is a Matrix

Message #13 Posted by ***Veli-Pekka Nousiainen*** *on 3 Dec 2004, 6:48 a.m.,*
in response to message #12 by Harry

```
MAP applies a function on each element of a matrix.
Hence, MAP([[a,0],[0,b]],EXP[x]) would give
[[e^a,1],[1,e^b]] and not [[e^a,0],[0,e^b]].


Greetings Axel
```

```
=====================================================
A=P^-1 . D . P , where P is inversible
and D only has diagonal terms (the eigenvalues).
In this case, A^n=P^-1 . D^n .
<snip>
exp(A) = P^-1 . Diag (exp(l1), ..., exp(ln)) P.
=====================================================
The 49g+ will show it's power using SYLVESTER:
A=P^-1 . D . P (actually A => D P)
Then use MAP on the D
:-D
[VPN]
```

Re: e^A where A is a Matrix

*Message #14 Posted by **Veli-Pekka Nousiainen** on 10 Dec 2004, 9:40 p.m.,*
in response to message #13 by Veli-Pekka Nousiainen

I also found this program at www.hpcalc.org
Matrix Function MFUN
http://www.hpcalc.org/details.php?id=6019
[VPN]

Re: e^A where A is a Matrix

*Message #15 Posted by **Valentin Albillo** on 3 Dec 2004, 4:39 a.m.,*
in response to message #1 by Harry

Have a look at this

It even gives the *exact* (not an infinite Taylor series expansion applied to matrices) expression for e^A, where A is a general 2x2 square matrix:

```
        A = |  a   b  |
            |  c   d  |
```

As you may see, the resulting expression is quite complicated, even for such a simple case. Don't want to think what it may look like for, say, a 5x5 matrix.

However, this explicit formula for the 2x2 case, when programmed, runs rings around any iterative series expansion, so if your matrices are 2x2, that's the way.

Best regards from V.

A program to compute e^A

Message #16 Posted by Valentin Albillo on 3 Dec 2004, 7:52 a.m.,
in response to message #1 by Harry

Hi, Harry:

To further show just how easy is it to compute **e^A** for arbitrary numeric square matrices, here's a subprogram for the HP-71B I've just written right now to prove the point. If you don't have a real 71B and/or its Math ROM to test this program, hold on: you can download an excellent emulator for free (Math ROM included!) from the homepage of Emu71.

For greatest convenience, the computation of e^A is performed by calling a subprogram, **MATEXP**, which accepts two parameters, the original matrix **A**, passed *by value*, and the resulting matrix **B = e^A**, passed *by reference*, which will be filled up by the subprogram and returned to the caller.

The subprogram is just 3 lines long (not counting the END SUB instruction which can be omitted), and it's completely general, dealing with arbitrary NxN matrices. Notice that you aren't required to pass N as a further parameter.

```
    100 SUB MATEXP(A(,),B(,)) @ DIM T(UBND(A,1),UBND(A,2)),P,I
    110 MAT B=IDN @ MAT T=IDN @ I=1 @ P=1.E-12
    120 MAT T=A*T @ MAT T=(1/I)*T @ IF RNORM(T)>P THEN MAT B=B+T @ I=I+1 @ GOTO 120
    130 END SUB
```

As an example, let's find e^A, where A is the 2x2 matrix:

```
    A = |  1/3   1/6  |
        | -1/3   5/6  |
```

Here's a sample calling program, though you could call this subprogram right from the keyboard, as we'll see later:

```
    10 DESTROY ALL @ OPTION BASE 1 @ DIM A(2,2),B(2,2) @ READ A(,)
    20 DATA 1/3,1/6,-1/3,5/6
    30 CALL MATEXP(A,B) @ MAT DISP B @ END
```

Now, simply RUN it:

```
    >RUN [ENTER]


          1.34970850034        .299012770355
         -.598025540708        2.2467468114
```

Let's test the result versus the exact values:

```
    >EXP(1/2)*2-EXP(2/3),EXP(2/3)-EXP(1/2)
```

```
        1.34970850034        .29901277036


    >EXP(1/2)*2-2*EXP(2/3),2*EXP(2/3)-EXP(1/2)


      -.59802554072         2.24674681142
```

So, as you can see, accuracy is good to a few ulps. As stated, you don't need to write a calling program, you could do it as easily right from the keyboard, like this:

```
    >DESTROYALL @ OPTION BASE 1 @ DIM A(2,2),B(2,2) @ MAT INPUT A [ENTER]


    A(1,1)? 1/3   [ENTER]
    A(1,2)? 1/6   [ENTER]
    A(2,1)? -1/3  [ENTER]
    A(2,2)? -5/6  [ENTER]


    >CALL MATEXP(A,B) @ MAT DISP B  [ENTER]


       1.36849513056        .136047911528
      -.272095823052        .416159749881
```

There's no problem at all in adapting this simple subprogram for any other HP model which has built-ín matrix operations, such as the HP-15C, HP-41C, and HP42S, to name a few. All you need to know is that, in the previous BASIC code:

- **UBND(A,1)** gives the Upper Bound for the index of the first dimension in matrix A. In our example, this returns 2. Likewise for **UBND(A,2)** which returns the upper bound for the second dimension (2, as well; it's a square matrix).

  This is used just to dimension auxiliar matrix T (term) to have the same dimensions as A (and B), thus no need to pass N (i.e: they are NxN matrices) as an extra parameter to the subprogram.

- **MAT B=IDN** just makes matrix B equal to the identity matrix corresponding to its dimensions.

- **P** is the precision we want. You may use 1E-10 in an HP-15C or HP-41C instead. Also, the smaller the precision, the faster the running time. You might be perfectly happy with 1E-5, say.

- **MAT T=A*T** is a true matrix multiplication, while **MAT T=(1/I)*T** simply divides each element of matrix T by the scalar I.

- **RNORM(T)** return the Row Norm of matrix T, **MAT B=B+T** is a simple matrix addition

And that's all there's to it. Writing the corresponding HP-15C/HP-41C/HP42S program is now quite affordable, and is left as an exercise to the reader ! :-)

Best regards from V.

Thanks, Valentin...

Message #17 Posted by ***Vieira, Luiz C. (Brazil)*** *on 3 Dec 2004, 9:35 a.m.,*
in response to message #16 by Valentin Albillo

... for the (as always) elegant and thoughtfull contribution.

Best regards.

Luiz (Brazil)

Re: Thanks, Valentin...

Message #18 Posted by ***Valentin Albillo*** *on 3 Dec 2004, 9:37 a.m.,*
in response to message #17 by Vieira, Luiz C. (Brazil)

You're welcome, Luiz. You're always so kind :-) :-)

(... by the way, your "thoughtfull" has one "l" too many ... :-)

Best regards from V.

I lost my spell checker... Thanks again! (N.T.)

Message #19 Posted by ***Vieira, Luiz C. (Brazil)*** *on 3 Dec 2004, 11:33 a.m.,*
in response to message #18 by Valentin Albillo

...

When do you use e^A

Message #20 Posted by ***Thibaut.be*** *on 3 Dec 2004, 9:53 a.m.,*
in response to message #16 by Valentin Albillo

Hi Valentin (and everybody else of course), I find it very instructing to learn a matrix functions that I didn't know.

All my apologizes for being such a pragmatic, but in what applications one uses e^A ?

Re: When do you use e^A

*Message #21 Posted by **Valentin Albillo** on 3 Dec 2004, 10:11 a.m.,*
in response to message #20 by Thibaut.be

Hi, Thibaut:

Thibaut posted:

"I find it very instructing to learn a matrix functions that I didn't know.

All my apologizes for being such a pragmatic, but in what applications one uses e^A ?"

Many, in fact, but the one that instantly comes to my mind is as one of the most elegant (as well as practical) ways of solving a linear system of Ordinary Differential Equations (O.D.E.'s, for short), and of course, such systems do appear absolutely all the time in real-life applications in all fields of science, industry, and economics.

Have a look at this excellent link, for instance.

Best regards from V.

Re: When do you use e^A

*Message #22 Posted by **Patrick** on 3 Dec 2004, 9:56 p.m.,*
in response to message #21 by Valentin Albillo

The same principle which leads to exp(A) leads to sin(A), cos(A), etc. In general, if you've got a convergent Taylor series, you can get a corresponding convergent series when the argument, x, is replaced by a matrix, A.

These functions satisfy certain multivariate differential equations in the same manner as their scalar cousins. Valentin pointed out a link which shows how exp(At) satisfies f' = Af, just like in the scalar case. Pretty neat, if you ask me.

Ah, enlightenment!

*Message #23 Posted by **Karl Schneider** on 3 Dec 2004, 11:36 p.m.,* *in response to message #21 by Valentin Albillo*

Thanks again, Valentin, for providing insight and informative links.

I was familiar with the Taylor-series definition of the matrix exponential from a graduate-level linear algebra course, but was unaware until now of any closed-form expression.

Recently, I'd considered writing a program to calculate it on the 42S, but hadn't determined what test to use for convergence.

With no real need to compute e^A on a handheld calculator, I'm content to allow Matlab to handle the chore, if it ever arises.

-- KS

Edited: 3 Dec 2004, 11:39 p.m.

Calculation with Vandermonde matrix

Message #24 Posted by ***Tizedes Csaba [Hungary]*** *on 9 Dec 2004, 6:39 a.m.,*
in response to message #20 by Thibaut.be

Hello, I used it in controlling of systems... The real case this was an element of solution differential equations. Here is the Maple version of calculating e^(A*t) simbolically, with Vandermonde-matrix: (I'm not an Maple-guru, sorry...)

If the eigenvalues are not equals, this method works perfectly. If they are same, you must to make new e() "lines": make derivative one of that equation, what contain one of the same eigenvalue...!

Sorry for my terrible english, I cant to say what I want...

Csaba

```
----------------------------------------------------
> restart;
> with(LinearAlgebra):with(linalg):
Warning, the previous binding of the name GramSchmidt has been removed and it now has an assigned value


Warning, the protected names norm and trace have been redefined and unprotected


> n:=2;
```

```
                              n := 2


> A:=matrix(n,n,[1,3,0,2]);


                                 [1    3]
                            A := [      ]
                                 [0    2]


> lambda:=eigenvals(A);


                            lambda := 1, 2


> V:=vandermonde([lambda]);


                                 [1    1]
                            V := [      ]
                                 [1    2]


> e:=array(1..n);


                        e := array(1 .. 2, [])


> for i from 1 to n do
> e[i]:=exp(lambda[i]*t);od;


                            e[1] := exp(t)



                           e[2] := exp(2 t)


> ga:=linsolve(V,e);


         ga := [2 exp(t) - exp(2 t), -exp(t) + exp(2 t)]
```

```
> unassign('i');
> e_ad_At:=evalm(ga[1]*IdentityMatrix(n)+sum(ga[i]*A^(i-1),i=2..n));


                      [exp(t)    -3 exp(t) + 3 exp(2 t)]
            e_ad_At := [                               ]
                      [  0             exp(2 t)        ]


>
```

Re: A program to compute e^A

*Message #25 Posted by **Harry** on 4 Dec 2004, 9:26 a.m.,*
in response to message #16 by Valentin Albillo

Thank you very much Valentin!

Especially for explaning the HP71B syntax afterwards, without that, I would not have been able to understand the Program you wrote.

What I like most about this forum is that most of the people here do not only know a lot about calculators, but also they are very good at mathmatics. Every time I ask a question about how to repair a calculator or how to solve a mathmatical problem with it, I get very good answers. Thanks to all of you!

Best regards, Harry

My Program

*Message #26 Posted by **Harry** on 4 Dec 2004, 3:38 p.m.,*
in response to message #16 by Valentin Albillo

I wrote a program for the HP48 now. What I din'd do so far is to program a stop criteria. I don not quite understand yours, valentin, as I do not know what the "row norm" of a matrix is. Could you or anybody else please explain that?

```
<< DUP IND DUP -> A E S
  << 1 25 FOR I
          A I / 'S' STO* 'E' STO+
        NEXT
     E
  >>
>>
```

Regards, Harry

Sorry, its IDN, not IND of course.

Message #27 Posted by ***Harry*** *on 4 Dec 2004, 4:55 p.m.,*
in response to message #26 by Harry

.

HP48/49 RNRM (aka: Row Norm)

Message #28 Posted by ***Vieira, Luiz C. (Brazil)*** *on 4 Dec 2004, 8:34 p.m.,*
in response to message #26 by Harry

Hi, Harry;

the "row norm" of a matrix is the highest value amongst the sum of the elements of each row. See this:

```
[[1 2 3]
 [7 8 9]
 [4 5 6]]
```

The column with the sum of each row is:

```
[[6]  (=1+2+3)
 [24] (=7+8+9)
 [15]](=4+5+6)
```

And in this case, the row norm is 24. The function that returns the row norm of a matrix is RNRM.

Hope this helps.

Luiz (Brazil)

Edited: 4 Dec 2004, 11:30 p.m.

Re: HP48/49 RNRM (aka: Row Norm)

Message #29 Posted by ***Harry*** *on 5 Dec 2004, 12:36 a.m.,*
in response to message #28 by Vieira, Luiz C. (Brazil)

Hi Luiz,

yes, that does help! Thanks a lot!

Harry

Re: e^A where A is a Matrix

Message #30 Posted by ***will*** *on 3 Dec 2004, 4:18 p.m.,*
in response to message #1 by Harry

You will be better served using a CAS software..mathematica 5 does this problem quite well.

Re: e^A where A is a Matrix

Message #31 Posted by ***Scott Guth*** *on 4 Dec 2004, 1:15 a.m.,*
in response to message #1 by Harry

Hi all!

I hate to say this, and please forgive this devoted HP fan for speaking of the "dark side", but my TI85 could do e^A back in 1992.

My TI89 Titanium can do e^A, ln(A), sin(A), cos(A), arcsin(A), arccos(A), sinh(A), cosh(A), arcsinh(A), arccosh(A) ...

The TI85, TI86, TI89 series, TI92, and TI Voyage are all capable of these.

The only requirement is that the matrix is diagonalizable. The main application for such operations is in solving linear systems of differential equations.

Regards, Scott

Re: e^A where A is a Matrix

Message #32 Posted by ***Rodger Rosenbaum*** *on 4 Dec 2004, 8:37 a.m.,*
in response to message #31 by Scott Guth

For diagonalizable matrices, the following HP48G program will work well. And in fact it can compute all the things Scott mentioned and MatrixSqrt, too (for those matrices that have a square root, that is). Just put the appropriate function in place of EXP.
```
<< EGV OBJ-> 0BJ-> DROP ->LIST EXP OBJ-> 1 ->LIST ->ARRY DUP SIZE OBJ-> DROP DIAG-> OVER INV * * >>
```

I made no attempt to optimize this; I just left everything inline to make it easier to understand. It works on complex matrices, too. And, don't forget, if you use trig functions to do, for example, MatrixSine, put the calculator in radians mode. The HP48G could do this in 1992, or whenever it came out; I forget. It's fun to play with.

Try [[1 2][3 4]] MatrixSqrt, then square the result. Try [[0 1][1 0]] MatrixSqrt.

Re: e^A where A is a Matrix

Message #33 Posted by ***Eddie Shore*** *on 4 Dec 2004, 9:48 a.m.,*
in response to message #32 by Rodger Rosenbaum

Thank you, Rodger (and everyone else in this wonderful forum). I am surprised to find that the HP does not have functions for square matrices. Ironically the TIs do.

Re: e^A where A is a Matrix

Message #34 Posted by ***Ren*** *on 6 Dec 2004, 6:05 p.m.,*
in response to message #32 by Rodger Rosenbaum

My College Algebra class (a "100 level" course) just finished a section on Matrices and Determinants. I was going to ask the same question Thibault.be asked regarding "real-world applications of e^A matrices". Unfortunately, the only answer that has been posted is over my head.

If some kind soul would enlighten this HPcalc wannabe as to what is meant by eigenvalue (not mentioned in my textbook) and the term "diagonalize" (does that relate to Identity Matrix?) I would appreciate it.

Thanks

Ren dona nobis pacem

Re: e^A where A is a Matrix

Message #35 Posted by ***Jeff O.*** *on 7 Dec 2004, 7:25 a.m.,*
in response to message #34 by Ren

Perhaps these may help:
Eigenvalue
Diagonalizable Matrix

[Return to Index | Top of Index]