HP Forum Archive 16

[Return to Index | Top of Index]

HP-15C: Quicker identity-matrix program

*Message #1 Posted by **Karl Schneider** on 29 Apr 2006, 2:31 a.m.*

Linear-algebra users --

The HP-15C was HP's first calculator with built-in support for linear-algebra functions, namely:

1. Matrix arithmetic
2. Solution of exactly-determined systems
3. Matrix inversion
4. Transpose and transpose-multiplication
5. Residual
6. Determinant
7. Euclidian/Frobenius and row norms
8. Utilities to support some operations for complex-valued matrices

However, the HP-15C was not designed as a full-featured tool for linear algebra; no built-in function to create an identity matrix is provided. This is not unreasonable, as keyboard space for functionality and RAM space for matrices is rather limited. Few linear-algebra problems that are convenient to solve on the HP-15C would require identity matrices.

(Interestingly enough, the HP-41 Advantage Pac and the HP-42S also lack built-in functions for creating identity matrices, although the HP-48 has the "IDN" function for that purpose.)

The HP-15C Advanced Functions Handbook (page 119) includes a 12-line program for creating an identity matrix:

```
LBL 8           42,21, 8
MATRIX 1        42,16, 1
LBL 9           42,21, 9
RCL 0           45  0
RCL 1           45  1
TEST 6          43,30, 6    (checks if off-diagonal element)
```

```
CLx             43 35
TEST 5          43,30, 5    (checks if on-diagonal element)
EEX             26
USER STO (i)  u 44 24       (store 0 or 1 and increment element)
USER            (no code)   (exits USER mode)
GTO 9           22  9       (skipped when last element reached)
RTN             43 32
```

This program, although compact, requires the user to manually dimension the identity matrix and store its descriptor to the indirect register. Also, it runs slowly, because it determines and stores each matrix element individually.

Example:

```
5
ENTER
DIM E
RCL MATRIX E
STO I
GSB 8
```

(5x5 identity matrix E is produced in 28 seconds)

Here's my 25-line program that is easier to use and runs faster (but requires almost two more registers of storage space):

```
LBL .9      42,21,.9
x<>y        34
STO I       44 25      (stores matrix descriptor to indirect)
CLx         43 35
ENTER       36
DIM I       42,23,25   (zeroes out the matrix)
Rdn         33
Rdn         33
ABS         43 16
INT         43 44      ("normalizes" the matrix order)
ENTER       36
DIM I       42,23,25   (redimensions as zero-filled matrix)
STO 0       44  0
STO 1       44  1      (positions pointer at end)
/           10         (creates 1, except if order is zero)
LBL .8      42,21,.8
STO (i)     44 24      (place 1 on diagonal element)
DSE 0       42, 5, 1   (decrement row pointer)
ABS         43 16      (a necessary "no op")
DSE 1       42, 5, 1   (decrement column pointer)
GTO .8      22 .8
```

```
CLx         43 35      (removes "1" from stack
RCL I       45 25      (column pointer reached 0; display result)
MATRIX 1    42,16, 1
RTN         43 32
```

This program requires that the user only recall the descriptor of the desired identity matrix (empty or not) to the stack, enter its desired order from 1 to 8 (space permitting), then execute the program. The positive integer portion of negative and fractional entered orders will be used. Only ones are stored to matrix elements, as the remainder of the matrix is already zero-filled.

Example:

```
RCL MATRIX E
5
GSB .9
```

(5x5 identity matrix E is produced in 7 seconds)

A sample application:

How close is the calculated inverse of matrix "A" (up to 4x4)?

1. (Create matrix A)
2. RCL MATRIX A
3. RESULT C
4. 1/x
5. (Create identity matrix in B using program)
6. RESULT B
7. RCL MATRIX A
8. RCL MATRIX C
9. MATRIX 6 (residual calculation)
10. (Read element-by-element error from matrix B)
11. (Compute Frobenius/Euclidian or row norms of B if desired)

-- KS

Edited: 29 Apr 2006, 5:03 p.m.

Re: HP-15C: Quicker identity-matrix program

*Message #2 Posted by **Valentin Albillo** on 29 Apr 2006, 11:54 p.m.,*
in response to message #1 by Karl Schneider

Hi, Karl:

I asked for such a program in one of my very first HP-15C mini-challenges many months ago. My own solution was:

```
        LBL A
        CLX
        STO MATRIX A
        MATRIX 1
        X<> 0
        LBL 0
        STO+ 0
    "u" STO A
        GTO 0
        RTN
```

which is just 10 steps long (including LBL and RTN), and only takes a maximum of 3 seconds for a 7x7 matrix, which is the largest possible size. You can substitute MATRIX A and STO A for whatever desired matrix, A to E.

Best regards from V.

Re: HP-15C: Quicker identity-matrix program

*Message #3 Posted by **Karl Schneider** on 30 Apr 2006, 6:17 p.m.,*
in response to message #2 by Valentin Albillo

Hi, Valentin --

Looking through the Archives, I couldn't find that particular challenge. Most were in Archive 8 and 14-15. One of the earlier ones dealt with matrices, but not creating identity matrices. If I've overlooked it, please let me know.

Your program is indeed clever, and probably runs a bit faster than mine (considering that mine takes more actions). It makes use of "0 STO MATRIX n" and the loop-exiting feature when at the end of a matrix.

My program does not require the user to pre-dimension the identity matrix to be created, and does allow any matrix A-E to be created without modifying the program. (Unfortunately, "STO MATRIX I" is not a valid instruction, which precludes indirect intialization.)

Both programs are much more efficient that the one in the HP-15C AFH, which stores *each individual element(!)*, when there are much better ways of creating a zero-filled matrix.

I remember your once stating (it may or may not have been in MoHPC Forum) that an "identity matrix creator function" would have been preferable to the present "MATRIX 1" function that sets R0 = R1 = 1. Even though to manually create an identity matrix can indeed be cumbersome, I'd have to disagree on that point:

- Positioning indices to the beginning of a matrix is a frequent operation when working with matrices. Calculating with sparse, space-consuming identity matrices would not be a common task on the HP-15C, given its limited resources.
- "f MATRIX 1" is three keystrokes, versus five for "1 STO 0 STO 1".
- "f MATRIX 1" does not disturb the stack.
- "f MATRIX 1" does not require the user to know that R0 and R1 contain the matrix indices (although s/he certainly *should*.) "MATRIX 1" and USER mode allow the user to navigate any matrix in its order of elements.
- More microcode would have been required to implement any capable and user-friendly "matrix generator" function, such as "eye" in Matlab.

Yup, those talented HP calculator engineers of yesteryear did the right thing, as usual. Those who would second-guess them do so at their own risk... :-)

Best regards,

-- KS

Edited: 30 Apr 2006, 10:02 p.m.

Re: HP-15C: Quicker identity-matrix program

*Message #4 Posted by **Valentin Albillo** on 1 May 2006, 10:12 p.m.,*
in response to message #3 by Karl Schneider

Hi again, Karl:

Karl posted:

"Looking through the Archives, I couldn't find that particular challenge. Most were in Archive 8 and 14-15. One of the earlier ones dealt with matrices, but not creating identity matrices. If I've overlooked it, please let me know."

> Unfortunately, I did not keep a copy of the thread, but I remember it probably was my very first HP-15C mini-challenge, and it gathered some interesting replies. Later on I issued some other matrix-related HP-15C mini-challenges, dealing with fast and short ways of performing certain very common operations which weren't built-in. They all must be somewhere in the archives.

"My program does not require the user to pre-dimension the identity matrix to be created, and does allow any matrix A-E to be created without modifying the program."

> My program is exactly the one which was given as a solution for the challenge, which asked to assign the identity matrix to a user-provided square matrix A of arbitrary dimension, and I've refrained from making any changes to it. I'm sure you can change my program to suit your uses, if desired and deemed worthwhile.
>
> The main difference is probably that you developed yours as a general purpose routine, capable of dealing with arbitrary matrices and such, while given the scarce RAM available in the HP-15C, I considered mine as some very short and fast code that would assign the identity matrix to some matrix A-E, to be inserted inline in your own code (just 8 steps and one label), as part of a larger, more important program, and modified accordingly to deal with your particular matrix A-E. Using 25 steps instead of 8 would mean that 3 data registers would be lost, thus further decreasing the size of matrices that could be worked upon.

"Both programs are much more efficient that the one in the HP-15C AFH, which stores each individual element(!), when there are much better ways of creating a zero-filled matrix."

> Yes, I consider the program in the AFH as totally unfit and undeserving of being featured in such an advanced book. Seems to me the people writing it were not in the mood to think that day because it's absolutely substandard for such a wonderful and valuable reference book. Shame on them.

"I remember your once stating (it may or may not have been in MoHPC Forum) that an "identity matrix creator function" would have been preferable to the present "MATRIX 1" function that sets R0 = R1 = 1. Even though to manually create an identity matrix can indeed be cumbersome, I'd have to disagree on that point:"

> Well, then we agree to disagree because I rest my case, I'll assign MATRIX 1 to provide the identity matrix instead, anytime. Given that MATRIX 1 actually does what it does, this means that we have to live with that and make the very best use of it, and it certainly comes useful now and then, and not only for matrix operations.

Thanks and best regards from V.

[Return to Index | Top of Index]

GTO Go back to the main exhibit hall