Subject: Fast matrix filling in IDL
Posted by weitkamp on Fri, 11 Dec 1998 08:00:00 GMT

View Forum Message <> Reply to Message

Dear all,

What is the fastest way of filling a matrix with identical
column vectors in IDL?

More precisely, if A is a float matrix (m x n) and V is
a vector with m elements, is there a faster way than

```
  FOR i=0,n-1 DO A[*,i]=V ?
```

Thanks,
  Timm

--
Timm Weitkamp
ESRF, Grenoble, France

-----------== Posted via Deja News, The Discussion Network ==----------
http://www.dejanews.com/   Search, Read, Discuss, or Start Your Own

Subject: Re: Fast matrix filling in IDL
Posted by David Foster on Mon, 14 Dec 1998 08:00:00 GMT

View Forum Message <> Reply to Message

```
Phillip & Suzanne David wrote:
>
>  David Fanning wrote:
>>
>>  Stein Vidar Hagfors Haugan (steinhh@ulrik.uio.no) writes:
>>>
>>>  A slight modification of David's program, and adding
>>>  my favourite speedup method:
>>>
>>>    time = systime(1)
>>>    array = rebin(reform(v,m,1,/overwrite),m,n,/sample)
>>>    print, 'Time for Rebin Operations: ', systime(1) - time
>>>
>>>  On { alpha OSF unix 5.2 Oct 30 1998}, this gives:
>>>
>>>  Time for Loop:                   0.27343702
>>>  Time for Matrix Operations:      0.093750000
>>>  Time for Rebin Operations:       0.067382932
>>>
```

>>> Note that the relative speeds can vary quite a lot on
>>> different architectures.
>>
>> I guess. Here is what I get with Stein Vidar's modifications
>> on my Windows NT machine:
>>
>>   IDL> Print, !Version
>>   { x86 Win32 Windows 5.2 Oct 30 1998}
>>   IDL> test
>>   Time for Loop: 0.10000002
>>   Time for Matrix Operations: 0.019999981
>>   Time for Rebin Operations: 0.039999962
>>
>> The Rebin operations are twice as slow as the matric operations.
>> Hummm. Why!?
>
> Here's another result from IDL 5.0.2 on the Mac:
> Time for Loop: 0.30000007
> Time for Matrix Operations: 0.13333333
> Time for Rebin Operations: 0.50000000
>
> Phillip

Hopefully everyone is running this a few times and noting when
the times have become "stable". Here are three sequential runs
on a Sun Sparc 2 (IDL 5.1):

```
IDL> test
Time for Loop:        2.7740721
Time for Matrix Operations:        5.2337180     ; OUT OF WHACK!
Time for Rebin Operations:       0.16969705
IDL> test
Time for Loop:        2.6756930
Time for Matrix Operations:       0.27333605
Time for Rebin Operations:       0.16920698
IDL> test
Time for Loop:        2.7509290
Time for Matrix Operations:       0.27153599
Time for Rebin Operations:       0.16843796
```

Dave

--

```
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~
    David S. Foster         Univ. of California, San Diego
     Programmer/Analyst     Brain Image Analysis Laboratory
     foster@bial1.ucsd.edu  Department of Psychiatry
```

(619) 622-5892   8950 Via La Jolla Drive, Suite 2240
                      La Jolla, CA 92037
~~~~~~ ~~~~~~~

Subject: Re: Fast matrix filling in IDL
Posted by Vapuser on Tue, 15 Dec 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Phillip & Suzanne David <pdavid@earthling.net> writes:

> David Fanning wrote:
>>
>> Stein Vidar Hagfors Haugan (steinhh@ulrik.uio.no) writes:
>>>
>>> A slight modification of David's program, and adding
>>> my favourite speedup method:
>>>
>>>   time = systime(1)
>>>   array = rebin(reform(v,m,1,/overwrite),m,n,/sample)
>>>   print, 'Time for Rebin Operations: ', systime(1) - time
>>>
>>> On { alpha OSF unix 5.2 Oct 30 1998}, this gives:
>>>
>>> Time for Loop: 0.27343702
>>> Time for Matrix Operations: 0.093750000
>>> Time for Rebin Operations: 0.067382932
>>>
>>> Note that the relative speeds can vary quite a lot on
>>> different architectures.
>>
>> I guess. Here is what I get with Stein Vidar's modifications
>> on my Windows NT machine:
>>
>>   IDL> Print, !Version
>>   { x86 Win32 Windows 5.2 Oct 30 1998}
>>   IDL> test
>>   Time for Loop: 0.10000002
>>   Time for Matrix Operations: 0.019999981
>>   Time for Rebin Operations: 0.039999962
>>
>> The Rebin operations are twice as slow as the matric operations.
>> Hummm. Why!?
>
>
> Here's another result from IDL 5.0.2 on the Mac:
> Time for Loop: 0.30000007
> Time for Matrix Operations: 0.13333333
~~~~~~

> Time for Rebin Operations: 0.50000000
>
> Phillip

Almost no differences between the matrix/rebin for an SGI.

```
testspeed
Time for Loop:      0.28050208
Time for Matrix Operations:     0.047688007
Time for Rebin Operations:     0.043171048
IDL> print,!version
{ mipseb IRIX unix 5.1.1 Jul 20 1998}
IDL>
```

Subject: Re: Fast matrix filling in IDL
Posted by J.D. Smith on Wed, 16 Dec 1998 08:00:00 GMT

View Forum Message <> Reply to Message

Kevin Ivory wrote:
>
>> David F. wrote:
>>
>>> You must have some hot machine, Kevin. Add more zeros!
>>>
>>>   Print, systime(1) - time, Format='(F20.18)'
>
> Stein Vidar Hagfors Haugan wrote:
>>
>> ..or maybe he's just got a very "slow" clock (coarse
>> granularity). Try repeating the operations a few (10)
>> times in one-line loops.
>
> I just sent in a bug report this morning. As usual, it was rejected
> by RSI and forwarded to our local distributor, so it might take a few
> weeks before I hear about this again.
>
> At least on my machine & configuration (SuSE 5.3 + glibc extentions),
> systime() only returns full seconds. The output of the IDL time test
> is real fun, so I'll include it here:

This behavior was introduced on Linux with IDL v5.2. I submitted a bug report which is still pending. How about that geometric mean! Please do not attempt to submit this result for inclusion in the IDLSpec database.

By the way, a new version of idlspec which is more careful with IDL version, machine type collection, and graphics format (24bit vs 8bit),

is on its way. I'm targetting it at IDL v5.2, but it looks like this
will have to wait until RSI resolves the systime() issue. Beta testers
will be welcomed.

JD

J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|

Subject: Re: Fast matrix filling in IDL
Posted by David Ritscher on Thu, 17 Dec 1998 08:00:00 GMT

View Forum Message <> Reply to Message

```
>  Hopefully everyone is running this a few times and noting when
>  the times have become "stable". Here are three sequential runs
>  on a Sun Sparc 2 (IDL 5.1):
>
>  IDL> test
>  Time for Loop:        2.7740721
>  Time for Matrix Operations:        5.2337180     ; OUT OF WHACK!
>  Time for Rebin Operations:       0.16969705
>  IDL> test
>  Time for Loop:        2.6756930
>  Time for Matrix Operations:       0.27333605
>  Time for Rebin Operations:       0.16920698
>  IDL> test
>  Time for Loop:        2.7509290
>  Time for Matrix Operations:       0.27153599
>  Time for Rebin Operations:       0.16843796
```

Perhaps the first one is not out of whack, but rather, the accurate
time.
When the test is being performed a second time, the operating system has
information that is cached, thus avoiding doing part of the work
associated with the task. Particularly in big tests, I often notice
speed-ups each time, for the first few times I test something. I like
to test by completely exiting my environment, doing other things, and
coming back and repeating the test. If I'm feeling hard-core about it,
I do a reboot in between.

David Ritscher

--

Cardiac Rhythm Management Laboratory
Department of Medicine
University of Alabama at Birmingham
B168 Volker Hall - 1670 University Boulevard
Birmingham AL 35294-0019
Tel: (205) 975-2122 Fax: (205) 975-4720
Email: david.ritscher@bigfoot.com

Subject: Re: Fast matrix filling in IDL
Posted by Martin Schultz on Thu, 17 Dec 1998 08:00:00 GMT
View Forum Message <> Reply to Message

David Ritscher wrote:

```
>>  Hopefully everyone is running this a few times and noting when
>>  the times have become "stable". Here are three sequential runs
>>  on a Sun Sparc 2 (IDL 5.1):
>>
>>  IDL> test
>>  Time for Loop:       2.7740721
>>  Time for Matrix Operations:       5.2337180     ; OUT OF WHACK!
>>  Time for Rebin Operations:      0.16969705
>>  IDL> test
>  [...]
```

> Perhaps the first one is not out of whack, but rather, the accurate
> time.
> When the test is being performed a second time, the operating system has
> information that is cached, thus avoiding doing part of the work
> associated with the task. Particularly in big tests, I often notice
> speed-ups each time, for the first few times I test something. I like
> to test by completely exiting my environment, doing other things, and
> coming back and repeating the test. If I'm feeling hard-core about it,
> I do a reboot in between.
>
> David Ritscher
>

Although I am definitively not a system expert, this sounds very reasonable and corresponds to my experience. E.g., when I read a binary data file, it usually takes much longer the first time than at subsequent times - often it is not even too much faster than sequential reading of the same file in ASCII(!) (but on subsequent attempts, the binary read is 1-2 orders of magnitude faster). I would assume that this behaviour is somehow associated with memory allocation/swapping, together with caching as David noted. The speed difference should therefore depend on the array size. It would be nice to devise some "objective" tests on this matter, except if some C malloc guru out there already knows the exact answer.

Here another little piece of info on this: on my SGI, I get different
behaviour for two slightly different routines:
#1 is the same as Stein Vidar listed it - on the first call, the matrix
operations take longer than on subsequent calls
#2 I changed array to array1, array2, and array3, "declaring" all three of
them in the beginning: This time I get more pronounced changes in the rebin
version! (both tests done after exiting the shell and new login)

Interestingly, when I call test#1 first, then #2 (they are both in one pro
file), the times reported for #2 are stable from the beginning and vice
versa. There must be some overhead processing going on when IDL executes a
routine for the first time.

Just as a side: here are the numbers that I get on my SGI Origin 2000:

```
Time for Loop:       0.32725203
Time for Matrix Operations:      0.057738066
Time for Rebin Operations:      0.034290910
```

Martin.

 -- -------
Dr. Martin Schultz
Department for Engineering&Applied Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
 -- -------

Subject: Re: Fast matrix filling in IDL
Posted by weitkamp on Wed, 06 Jan 1999 08:00:00 GMT

View Forum Message <> Reply to Message

In article <36798167.7F63C1CE@io.harvard.edu>,
 Martin Schultz <mgs@io.harvard.edu> wrote:
> David Ritscher wrote:
>
>>> Hopefully everyone is running this a few times and noting when
>>> the times have become "stable". Here are three sequential runs
>>> on a Sun Sparc 2 (IDL 5.1):
>>>
>>> IDL> test

>>> Time for Loop: 2.7740721
>>> Time for Matrix Operations: 5.2337180 ; OUT OF WHACK!
>>> Time for Rebin Operations: 0.16969705
>>> IDL> test
>> [...]
>
>> Perhaps the first one is not out of whack, but rather, the accurate
>> time.
>> When the test is being performed a second time, the operating system has
>> information that is cached, thus avoiding doing part of the work
>> associated with the task. Particularly in big tests, I often notice
>> speed-ups each time, for the first few times I test something. I like
>> to test by completely exiting my environment, doing other things, and
>> coming back and repeating the test. If I'm feeling hard-core about it,
>> I do a reboot in between.
>>
>> David Ritscher
>>
>
> Although I am definitively not a system expert, this sounds very reasonable
> and corresponds to my experience. E.g., when I read a binary data file, it
> usually takes much longer the first time than at subsequent times - often
> it is not even too much faster than sequential reading of the same file in
> ASCII(!) (but on subsequent attempts, the binary read is 1-2 orders of
> magnitude faster). I would assume that this behaviour is somehow associated
> with memory allocation/swapping, together with caching as David noted. The
> speed difference should therefore depend on the array size. It would be
> nice to devise some "objective" tests on this matter, except if some C
> malloc guru out there already knows the exact answer.
>
> Here another little piece of info on this: on my SGI, I get different
> behaviour for two slightly different routines:
> #1 is the same as Stein Vidar listed it - on the first call, the matrix
> operations take longer than on subsequent calls
> #2 I changed array to array1, array2, and array3, "declaring" all three of
> them in the beginning: This time I get more pronounced changes in the rebin
> version! (both tests done after exiting the shell and new login)
>
> Interestingly, when I call test#1 first, then #2 (they are both in one pro
> file), the times reported for #2 are stable from the beginning and vice
> versa. There must be some overhead processing going on when IDL executes a
> routine for the first time.
>
> [...]

This means that the "correct" way of determining computing time for this
piece of code strongly depends its later use within a program. My original
question, "what is the fastest way of filling a matrix with identical column

vectors", may therefore not have a definite answer even for a given machine, except "it depends on how many times this will be done and what happens in between".

Anyway, some interesting questions have been raised here, and David's and Stein Vidar's solutions to my problem have helped me speed up my program enormously.

Thanks,

Timm

P.S. Even on the different HP9000's I use, the relative speeds
for the different operations depend strongly on slight
differences in architecture:

```
HP-UX B.10.20 9000/871 (IDL 5.1)
 Time for Loop:       0.72376597
 Time for Matrix Operations:      0.095106006
 Time for Rebin Operations:      0.15038002

HP-UX B.10.20 9000/755 (IDL 5.1)
 Time for Loop:       0.79779303
 Time for Matrix Operations:      0.083228946
 Time for Rebin Operations:      0.57194102
```

--
Timm Weitkamp
European Synchrotron Radiation Facility (ESRF)
B.P. 220
F-38043 Grenoble Cedex
France

E-mail: weitkamp@esrf.fr