Subject: Re: Can I do this without using loops?
Posted by steinhh on Sun, 16 Jun 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Here are some timing results for an Alphaserver 2100 with 4 CPUs installed and about 1 GB internal memory (which means that the large test didn't swap, I think. However, the .SIZE command in IDL only takes unsigned integers for the number of Kilobytes to reserve, so some tests bombed because of internal memory limits in IDL). I made sure that only 2 other jobs were executing concurrently, so one CPU was more or less dedicated to running this test. (With the fourth doing housekeeping).

I'm not sure about the cache size or processor speed, though.

I added methods 10 and 11, (included at the end of the article) to test the /sample keywords, and to try a combination of the fast column insertion (method 8) with a transpose.

I don't have pv-wave, but just for fun I tested both Idl 3.6.1c and Idl 4.0, and I got a surprise! Unless I've made some strange mistakes, the comparison is quite counterintuitive.

Rows and columns: 3000 3000

	IDL 3.6.1c	4.0		3.6.1c	4.0
1.	4.64	7.16		8.16	10.10
2.	6.92	7.60		0.71	0.74
3.	0.96	1.39		0.92	1.36
4.	2.25	2.90		2.20	2.98
5.	17.58	21.84		17.56	21.83
6.	32.78	38.85		32.22	38.94
7.	18.98	21.38		18.96	21.67
8.	0.73	0.75		3.64	3.69
9.	9.40	11.51		5.49	7.22
10.	0.69	0.69		3.65	3.70
11.	0.69	0.69		4.75	5.38

Rows and columns: 1000 30000

	3.6.1c	4.0		3.6.1c	4.0
1.	15.88	24.00		17.98	24.87
2.	14.21	16.70		2.77	2.81
3.	2.97	4.52		3.02	4.52
4.	*******	*******		*******	*******
5.	58.70	72.90		58.67	72.88

```
 6. 109.13  129.85  |    107.16   129.93
 7.  60.89   69.90  |     60.94    71.47
 8.   2.75    2.78  |      5.35     5.71
 9.******* *******  |     19.64    24.75
10.   2.71    2.72  |      5.42     5.48
11.   2.70    2.79  |   *******  *******
```

Note that version 4.0 is actually *always* slower than version 3.6.1c. Is the "upgrade" to 4.0 coincident with the switch from fortran to C source code? In that case, maybe RSI should consider switching back (at least for some core routines), if these numbers are correct.

Here are the 3.6.1c results relative to the best 3.6.1c results, and 4.0 results relative to the corresponding 3.6.1c results:

Rows and columns: 3000 3000

```
   3.6.1c    4.0   |    3.6.1c    4.0

 1.   6.72    1.54  |     11.49    1.24
 2.  10.03    1.10  |      1.00    1.04
 3.   1.39    1.45  |      1.30    1.47
 4.   3.26    1.29  |      3.10    1.35
 5.  25.48    1.24  |     24.73    1.24
 6.  47.51    1.19  |     45.38    1.21
 7.  27.51    1.13  |     26.70    1.14
 8.   1.06    1.03  |      5.13    1.01
 9.  13.62    1.22  |      7.73    1.32
10.   1.00    1.00  |      5.14    1.01
11.   1.00    1.00  |      6.69    1.13
```

Rows and columns: 1000 30000

```
   3.6.1c    4.0   |    3.6.1c    4.0

 1.   5.88    1.50  |      6.49    1.38
 2.   5.26    1.17  |      1.00    1.01
 3.   1.10    1.52  |      1.09    1.50
 4.******* *******  |   *******  *******
 5.  21.74    1.24  |     21.18    1.24
 6.  40.42    1.19  |     38.69    1.21
 7.  22.55    1.14  |     22.00    1.17
 8.   1.02    1.01  |      1.93    1.07
 9.******* *******  |      7.09    1.26
10.   1.00    1.00  |      1.95    1.01
11.   1.00    1.03  |   *******  *******
```

For a slower alpha machine (front cover says DEC 3000) I only bothered to time the small test. Here the differences between version 3.6 and 4.0 are slightly less systematic.

Rows and columns: 3000 3000

IDL	3.6.1c	4.0	3.6.1c	4.0
1.	11.96	11.63	14.27	14.62
2.	7.83	9.88	0.85	0.88
3.	1.71	2.83	1.71	2.83
4.	19.18	21.98	20.64	17.82
5.	39.89	43.95	40.09	43.98
6.	81.61	98.88	80.17	101.32
7.	42.68	50.47	41.92	48.96
8.	0.92	1.43	3.91	3.88
9.	41.61	48.65	9.87	13.07
10.	0.85	0.85	3.92	3.85
11.	0.91	1.39	42.91	29.58

Relative results:

1.	14.07	0.97	16.79	1.02
2.	9.21	1.26	1.00	1.04
3.	2.01	1.65	2.01	1.65
4.	22.56	1.15	24.28	0.86
5.	46.93	1.10	47.16	1.10
6.	96.01	1.21	94.32	1.26
7.	50.21	1.18	49.32	1.17
8.	1.08	1.55	4.60	0.99
9.	48.95	1.17	11.61	1.32
10.	1.00	1.00	4.61	0.98
11.	1.07	1.53	50.48	0.69

For a DECstation 5000/240 I only ran an even smaller test (1000,1000), with the following results:

Rows and columns: 1000 1000

IDL	3.6.1c	4.0	3.6.1c	4.0
1.	4.66	12.90	4.84	11.67
2.	2.60	2.80	0.11	0.09
3.	0.91	0.93	0.90	0.98
4.	1.72	1.66	1.22	1.25
5.	13.10	12.53	13.34	12.40
6.	28.32	26.28	21.76	25.91
7.	13.82	12.93	13.98	12.93
8.	0.40	0.33	0.70	0.71
9.	4.02	4.07	3.11	3.11

```
10.   0.19    0.24  |      0.58     0.66
11.   0.32    0.33  |      1.11     1.12
```

3.6.1c results relative to the best 3.6.1c results,
and 4.0 results relative to the corresponding 3.6.1c results:

```
 1.  24.53   2.77  |      44.00        2.41
 2.  13.68   1.08  |       1.00        0.82
 3.   4.79   1.02  |       8.18        1.09
 4.   9.05   0.97  |      11.09        1.02
 5.  68.95   0.96  |     121.27        0.93
 6. 149.05   0.93  |     197.82        1.19
 7.  72.74   0.94  |     127.09        0.92
 8.   2.11   0.83  |       6.36        1.01
 9.  21.16   1.01  |      28.27        1.00
10.   1.00   1.26  |       5.27        1.14
11.   1.68   1.03  |      10.09        1.01
```

In other words, even less systematic differences. I'm still a bit surprised that *any* of these numbers go the "wrong" way.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

PRO TEST10,rows,columns
  strt = systime(1)
;
  array = rebin(FINDGEN(1, columns), rows, columns,/sample)
;
  time = systime(1)-strt
  PRINT, 'elapsed time: ',time
  RETURN
;
END


;; Identical to number 8 (just filling in for completeness)
;; The row version uses this plus a transpose
PRO test11,rows,columns
  strt = systime(1)
;
  array = rebin(FINDGEN(1, columns), rows, columns)
;
  time = systime(1)-strt
  PRINT, 'elapsed time: ',time
  RETURN
END
```

```
PRO test10_rows,rows,columns
  strt = systime(1)
;
  array = rebin(FINDGEN(rows,1), rows, columns,/sample)
;
  time = systime(1)-strt
  PRINT, 'elapsed time: ',time
  RETURN
;
END

PRO test11_rows,rows,columns
  strt = systime(1)
;
  array = TRANSPOSE(rebin(FINDGEN(1,rows), columns, rows,/sample))
;
  time = systime(1)-strt
  PRINT, 'elapsed time: ',time
  RETURN
;
END
```
