COMP-40730 HPC

Report for Assignment 4

Author:	Paula Dwan
Due date:	July-2014
Lecturer:	Alexey Lastovetsky
Subject:	COMP-40730 High Performance Computing
College:	University College Dublin

CONTENTS

EXERCISE

Write a parallel MPI program computing the norm of the product of two n×n dense matrices on a p-processor SMP so that :

- p processors are involved in the computations.
- The 1-dimensional parallel algorithm of matrix multiplication is employed:
    - the matrices are identically and equally partitioned in one dimension into p horizontal slices
    - there is one-to-one mapping between the partitions and the processors
    - each processor is responsible for computation of the corresponding slice of the resulting matrix

You can use BLAS or ATLAS for local computations.

Experiment with the program and build:

- The dependence of the execution time of the program on the matrix size n.
- The speedup over a serial counterpart of the program.

Explain the results.

Variants of the assignment:

1. Granularity of the program:
    (a) Two successive steps:
        i. Parallel matrix multiplication
        ii. Parallel computation of the norm of the resulting matrix
    (b) One-step algorithm. No intermediate resulting matrix.

2. Partitioning scheme:
    (a) Left matrix is horizontally partitioned
    (b) Right matrix is vertically partitioned

3. Matrix norm to be computed:
    (a) The maximum absolute column sum norm (aka one-norm):

$$||A||_1 = \max_{0 \le j < n} \sum_{i=0}^{n-1} |a_{ij}|$$

    (b) The maximum absolute row sum norm (aka infinity-norm):

$$||A||_\infty = \max_{0 \le i < n} \sum_{j=0}^{n-1} |a_{ij}|$$

Overview of Computations Obtained and How

Assignment 4 basically involved (for me) writing two programmes,

- **A4-mpi-manual.c**
  one which contained the code for manual straight-forward IJK computation and dgemm to calculate matrix |C| and the infinity norm of |C|.
- **A4-mpi-solo.c**
  second which contained the code for manual straight-forward IJK computation and MPI to calculate matrix |C| and the infinity norm of |C|.

I then compared the time taken to calculate matrix |C| for each method using [N]x[N] matrices |A| and |B| of different sizes, allocating values for the |A| and |B| using random numbers (1 to 10 inclusive) or the column value + 1 (so that column 1000 would have a value of 1001).

1. **manual straight-forward IJK computation**

   Implementation of a straight forward matrix nxn multiplication.

   *Present in **A4-mpi-manual.c** and in **A4-mpi-solo.c***

   Code
   ```
   for (ni=0 ; ni<rows ; ni++)
   {
      for (nj=0 ; nj<cols ; nj++)
      {
         double sum = 0.0 ;
         for (nk=0 ; nk<rows ; nk++)
         {
            sum+= (A[(ni*rows)+nk]) * (B[(nk*rows)+nj]) ;
         }
         C[(ni*rows)+nj] = sum ;
      }
   }
   ```

2. **Dgemm for straight-forward IJK computation**

   I used dgemm compiled for atlas or for cblas.

   The matrix |C| was calculated using cblas by default otherwise the user could decide to re-build and execute using cblas or atlas).

   *Present in **A4-mpi-manual.c** only*

   Code
   ```
      int ni, nj ;
   // m, n, k :    local integers indicating the size of the matrices for
   // rows x columns :: A :  m x  k, B :  k x n, C:  m x n
   // Here, m = n = k = rows = columns = <nx> = <ny> as supplied
      int lm = rows, ln = rows ;
   // la_offset, lb_offset, lc_offset :
   // Leading dimension of matrix A, B or C respectively, or the number of elements
   // between successive rows for row-major storage or columns for column-major
   // storage.
      int la_offset = rows, lb_offset = cols, lc_offset = rows ;
      int ALPHA=1.0 ;
      int BETA=0.0 ;

      cblas_dgemm( CblasRowMajor, CblasNoTrans, CblasNoTrans, lm, ln, ln, ALPHA, \
         A, la_offset, B, lb_offset, BETA, C, lc_offset) ;
   ```

3. **MPI calculation**

   This was completed using a data structure, creating a new process to correspond to the each segment.

   *Present in **A4-mpi-solo.c** only*

   Code
   ```
   MPI_Barrier ()
   MPI_Bcast ()
   MPI_Comm_rank ()
   MPI_Comm_size ()
   MPI_Finalize ()
   MPI_Gather ()
   MPI_Init ()
   MPI_Scatter ()
   MPI_Wtime()
   ```

   Note: *Please see section for detailed description of code used : Overview of MPI functions used.*

Thus results were obtained for each option 1., 2. and 3. using the same source |A| and |B| for both -i (increment) and also using -r (random) .

Multiple implementation of each .c program was enabled using ***./runAssignment4.sh,*** for example :

```
$ ./runAssignment4.sh -1 -i -v
```

This runs the c. program using cblas and manual straight-forward IJK for incremental column values, using predefined matrix sizes for each implementation.

Single implementation is completed using

- ./**A4-mpi-manual.c.**

  ```
  $ ./A4-mpi-manual-cblas -i 10 file.txt file.dat
  ```

  This is compiled using *gcc* for atlas and cblas, as follows :

  ```
  gcc -I/home/cs/khasanov/libs/CBLAS/src A4-mpi-manual.c -o A4-mpi-manual-cblas       \
      /home/cs/khasanov/libs/cblas_LINUX.a  /usr/lib/libblas.a -lgfortran

  gcc -o A4-mpi-manual-atlas A4-mpi-manual.c -I/home/cs/khasanov/libs/ATLAS/include/  \
      -L/home/cs/khasanov/libs/ATLAS/lib/Linux_UNKNOWNSSE2_4/ -lcblas -latlas -lm -O3
  ```

- ./**A4-mpi-solo.c.**

  ```
  $ mpirun -np <number of processors> --hostfile hostfile A4-mpi-solo -i <matrix size> <matrix file>.txt <summary timing file>.dat
  mpirun -np 5 --hostfile hostfile A4-mpi-solo -i 10 10.txt 10.dat
  ```

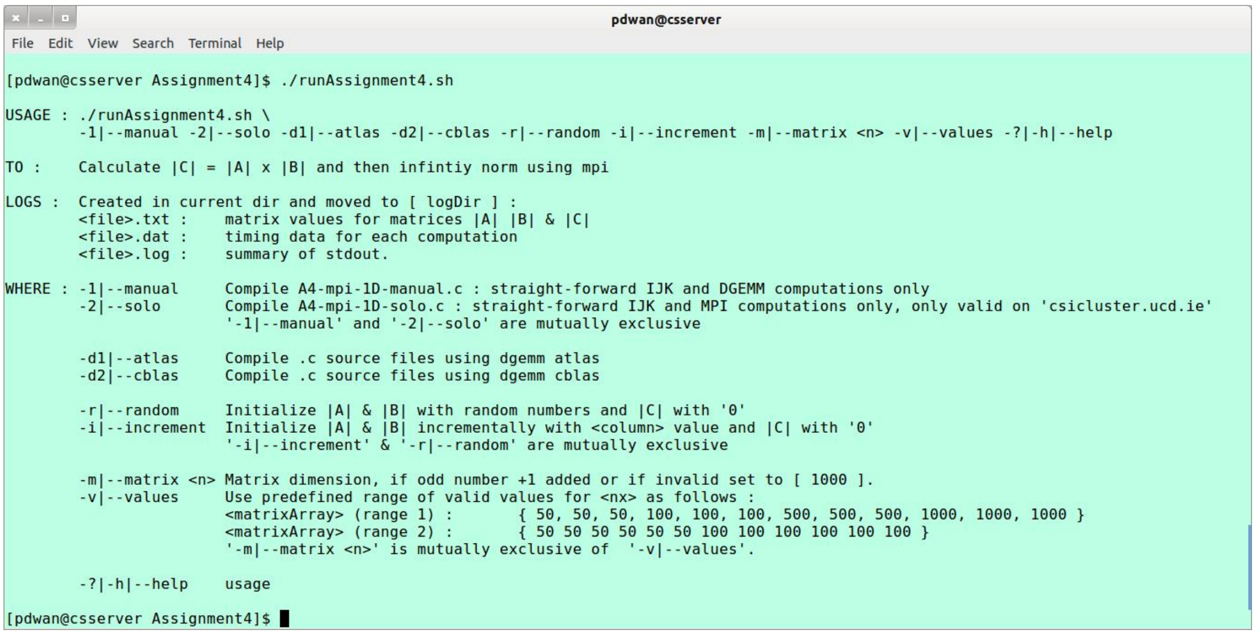
  This is compiled using *mpicc* , as follows :

  ```
  mpicc -Wall A4-mpi-solo.c -o A4-mpi-solo
  ```

ASSIGNMENT EXECUTION

The compiled .c program ./A4-mpi-manual-<cblas|atlas> was executed multiple times standalone or using the script *./runAssignment4.sh* to obtain as wide a range of time taken to calculate |C| using each algorithm.

This has multiple options and the syntax and usage follows :

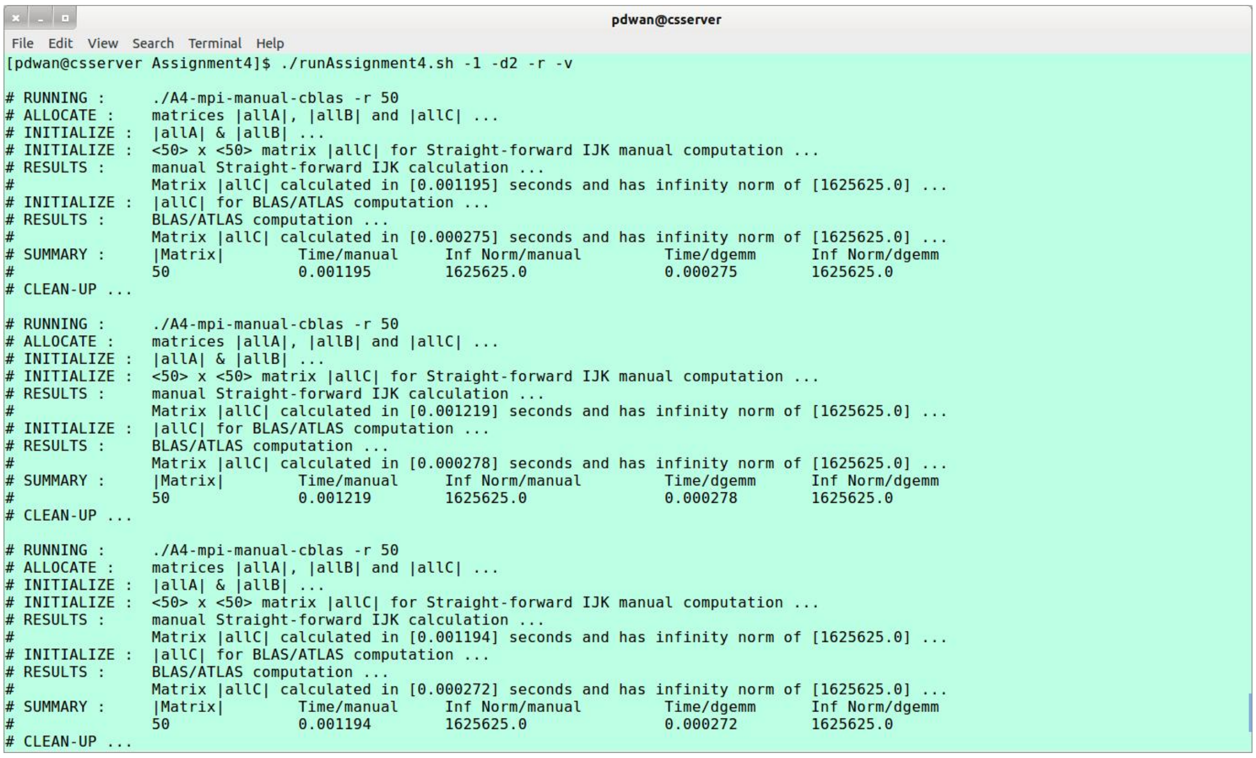


```
[pdwan@csserver Assignment4]$ ./runAssignment4.sh

USAGE : ./runAssignment4.sh \
        -1|--manual -2|--solo -d1|--atlas -d2|--cblas -r|--random -i|--increment -m|--matrix <n> -v|--values -?|-h|--help

TO :    Calculate |C| = |A| x |B| and then infintiy norm using mpi

LOGS :  Created in current dir and moved to [ logDir ] :
        <file>.txt :    matrix values for matrices |A| |B| & |C|
        <file>.dat :    timing data for each computation
        <file>.log :    summary of stdout.

WHERE : -1|--manual     Compile A4-mpi-1D-manual.c : straight-forward IJK and DGEMM computations only
        -2|--solo       Compile A4-mpi-1D-solo.c : straight-forward IJK and MPI computations only, only valid on 'csicluster.ucd.ie'
                        '-1|--manual' and '-2|--solo' are mutually exclusive

        -d1|--atlas     Compile .c source files using dgemm atlas
        -d2|--cblas     Compile .c source files using dgemm cblas

        -r|--random     Initialize |A| & |B| with random numbers and |C| with '0'
        -i|--increment  Initialize |A| & |B| incrementally with <column> value and |C| with '0'
                        '-i|--increment' & '-r|--random' are mutually exclusive

        -m|--matrix <n> Matrix dimension, if odd number +1 added or if invalid set to [ 1000 ].
        -v|--values     Use predefined range of valid values for <nx> as follows :
                        <matrixArray> (range 1) :       { 50, 50, 50, 100, 100, 100, 500, 500, 500, 1000, 1000, 1000 }
                        <matrixArray> (range 2) :       { 50 50 50 50 50 50 100 100 100 100 100 100 }
                        '-m|--matrix <n>' is mutually exclusive of  '-v|--values'.

        -?|-h|--help    usage

[pdwan@csserver Assignment4]$ █
```

Execute this script in the home directory of Assignment 4.

Sample execution follows for matrixArray range 2 :

```
$ ./runAssignment4.sh -1 -d2 -r -v
```



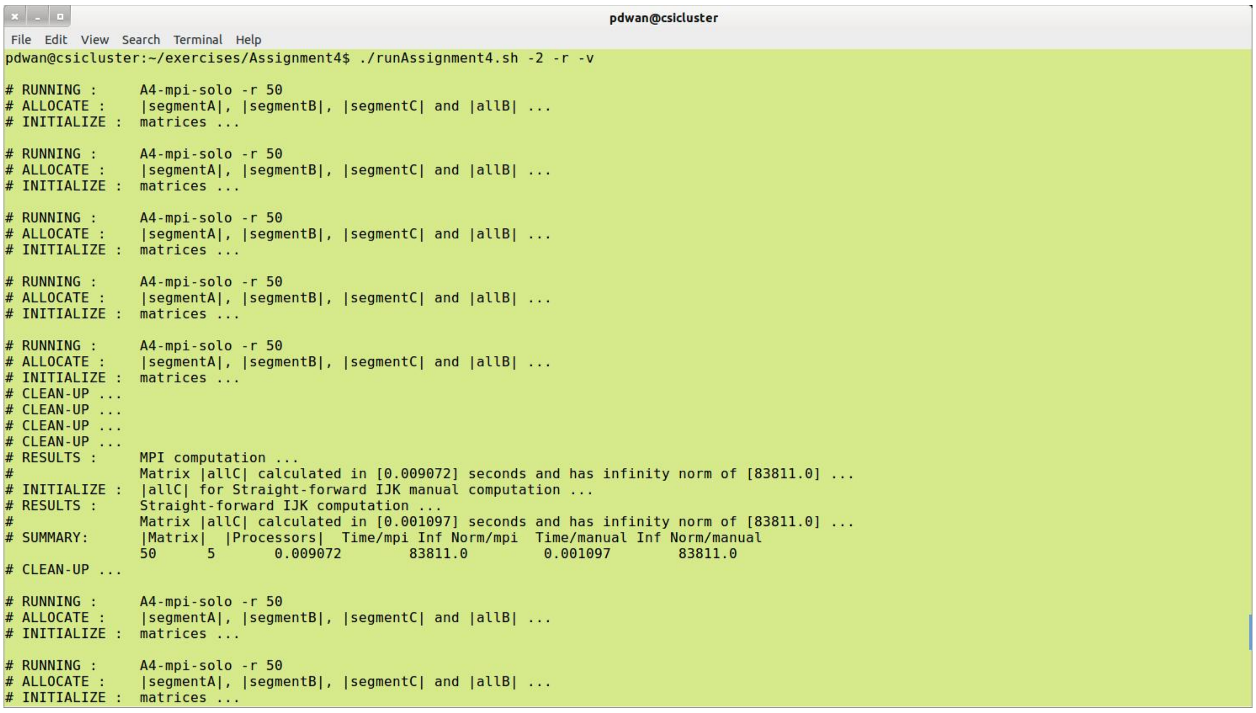
pdwan@csserver

File Edit View Search Terminal Help

```
[pdwan@csserver Assignment4]$ ./runAssignment4.sh -1 -d2 -r -v

# RUNNING :     ./A4-mpi-manual-cblas -r 50
# ALLOCATE :    matrices |allA|, |allB| and |allC| ...
# INITIALIZE :  |allA| & |allB| ...
# INITIALIZE :  <50> x <50> matrix |allC| for Straight-forward IJK manual computation ...
# RESULTS :     manual Straight-forward IJK calculation ...
#               Matrix |allC| calculated in [0.001195] seconds and has infinity norm of [1625625.0] ...
# INITIALIZE :  |allC| for BLAS/ATLAS computation ...
# RESULTS :     BLAS/ATLAS computation ...
#               Matrix |allC| calculated in [0.000275] seconds and has infinity norm of [1625625.0] ...
# SUMMARY :     |Matrix|        Time/manual     Inf Norm/manual         Time/dgemm      Inf Norm/dgemm
#               50              0.001195        1625625.0               0.000275        1625625.0
# CLEAN-UP ...

# RUNNING :     ./A4-mpi-manual-cblas -r 50
# ALLOCATE :    matrices |allA|, |allB| and |allC| ...
# INITIALIZE :  |allA| & |allB| ...
# INITIALIZE :  <50> x <50> matrix |allC| for Straight-forward IJK manual computation ...
# RESULTS :     manual Straight-forward IJK calculation ...
#               Matrix |allC| calculated in [0.001219] seconds and has infinity norm of [1625625.0] ...
# INITIALIZE :  |allC| for BLAS/ATLAS computation ...
# RESULTS :     BLAS/ATLAS computation ...
#               Matrix |allC| calculated in [0.000278] seconds and has infinity norm of [1625625.0] ...
# SUMMARY :     |Matrix|        Time/manual     Inf Norm/manual         Time/dgemm      Inf Norm/dgemm
#               50              0.001219        1625625.0               0.000278        1625625.0
# CLEAN-UP ...

# RUNNING :     ./A4-mpi-manual-cblas -r 50
# ALLOCATE :    matrices |allA|, |allB| and |allC| ...
# INITIALIZE :  |allA| & |allB| ...
# INITIALIZE :  <50> x <50> matrix |allC| for Straight-forward IJK manual computation ...
# RESULTS :     manual Straight-forward IJK calculation ...
#               Matrix |allC| calculated in [0.001194] seconds and has infinity norm of [1625625.0] ...
# INITIALIZE :  |allC| for BLAS/ATLAS computation ...
# RESULTS :     BLAS/ATLAS computation ...
#               Matrix |allC| calculated in [0.000272] seconds and has infinity norm of [1625625.0] ...
# SUMMARY :     |Matrix|        Time/manual     Inf Norm/manual         Time/dgemm      Inf Norm/dgemm
#               50              0.001194        1625625.0               0.000272        1625625.0
# CLEAN-UP ...
```

Sample execution follows for matrixArray range 1 for MPI:

```
$ ./runAssignment4.sh -2 -r -v
```



pdwan@csicluster

File Edit View Search Terminal Help

```
pdwan@csicluster:~/exercises/Assignment4$ ./runAssignment4.sh -2 -r -v

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...
# CLEAN-UP ...
# CLEAN-UP ...
# CLEAN-UP ...
# CLEAN-UP ...
# RESULTS :     MPI computation ...
#               Matrix |allC| calculated in [0.009072] seconds and has infinity norm of [83811.0] ...
# INITIALIZE :  |allC| for Straight-forward IJK manual computation ...
# RESULTS :     Straight-forward IJK computation ...
#               Matrix |allC| calculated in [0.001097] seconds and has infinity norm of [83811.0] ...
# SUMMARY:      |Matrix|  |Processors|  Time/mpi Inf Norm/mpi  Time/manual Inf Norm/manual
                50      5       0.009072        83811.0         0.001097        83811.0
# CLEAN-UP ...

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...

# RUNNING :     A4-mpi-solo -r 50
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...
```

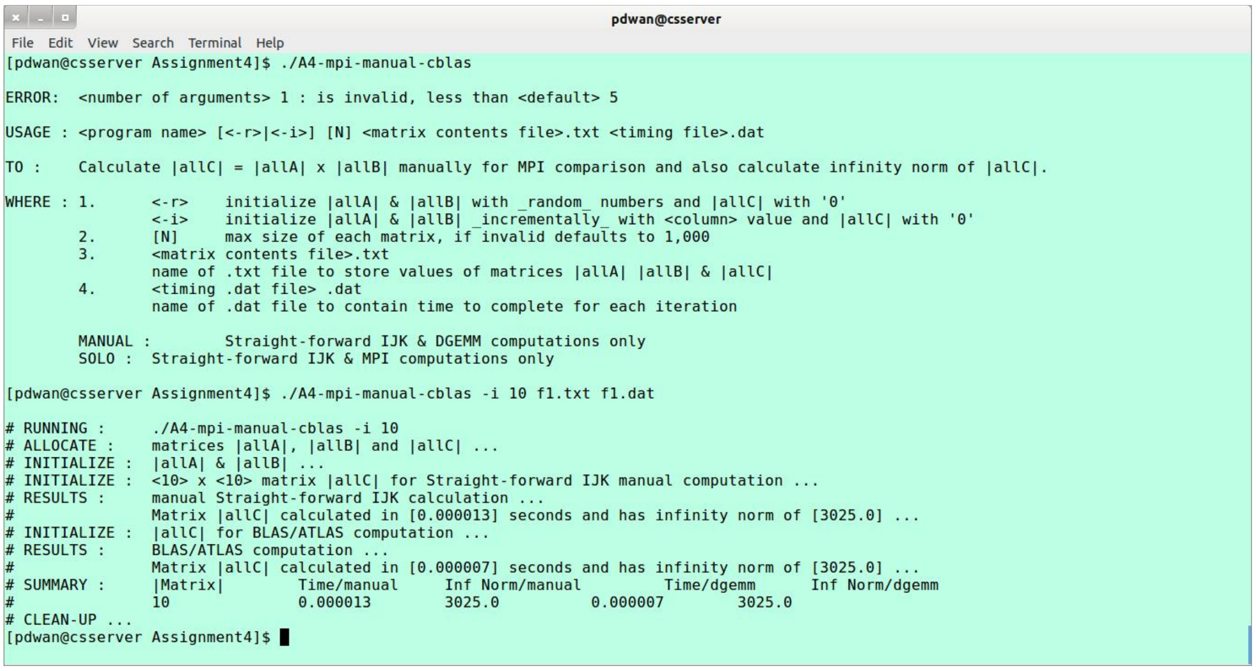
Please retain the overall directory structure when unzipping.

Note that the script `./runAssignment4.sh` allows two types of implementation

- Multiple iteration : use the switch <-v|--values>, when a predefined range applies for [N] : matrix size.
- Single iteration : use the switch <-m|--matrix> [N] where the user specifies value for [N] : matrix size.

RUNNING A4-MPI-MANUAL-<ATLAS | CBLAS> : STANDALONE

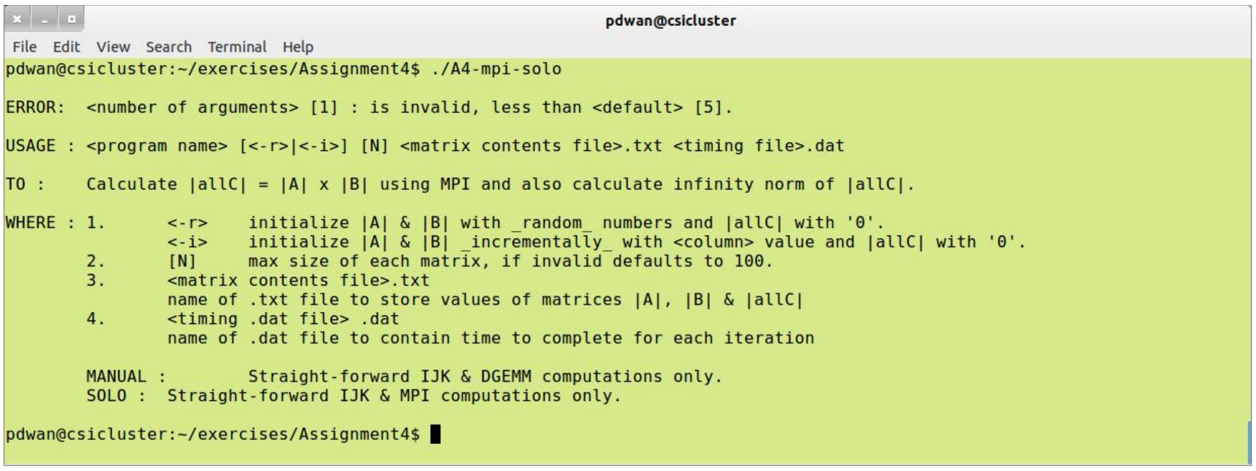
The compiled .c program may also be run standalone. Usage and sample execution follows :



```
[pdwan@csserver Assignment4]$ ./A4-mpi-manual-cblas

ERROR:  <number of arguments> 1 : is invalid, less than <default> 5

USAGE : <program name> [<-r>|<-i>] [N] <matrix contents file>.txt <timing file>.dat

TO :    Calculate |allC| = |allA| x |allB| manually for MPI comparison and also calculate infinity norm of |allC|.

WHERE : 1.      <-r>    initialize |allA| & |allB| with _random_ numbers and |allC| with '0'
                <-i>    initialize |allA| & |allB| _incrementally_ with <column> value and |allC| with '0'
        2.      [N]     max size of each matrix, if invalid defaults to 1,000
        3.      <matrix contents file>.txt
                name of .txt file to store values of matrices |allA| |allB| & |allC|
        4.      <timing .dat file> .dat
                name of .dat file to contain time to complete for each iteration

        MANUAL :        Straight-forward IJK & DGEMM computations only
        SOLO :  Straight-forward IJK & MPI computations only

[pdwan@csserver Assignment4]$ ./A4-mpi-manual-cblas -i 10 f1.txt f1.dat

# RUNNING :     ./A4-mpi-manual-cblas -i 10
# ALLOCATE :    matrices |allA|, |allB| and |allC| ...
# INITIALIZE :  |allA| & |allB| ...
# INITIALIZE :  <10> x <10> matrix |allC| for Straight-forward IJK manual computation ...
# RESULTS :     manual Straight-forward IJK calculation ...
#               Matrix |allC| calculated in [0.000013] seconds and has infinity norm of [3025.0] ...
# INITIALIZE :  |allC| for BLAS/ATLAS computation ...
# RESULTS :     BLAS/ATLAS computation ...
#               Matrix |allC| calculated in [0.000007] seconds and has infinity norm of [3025.0] ...
# SUMMARY :     |Matrix|        Time/manual     Inf Norm/manual         Time/dgemm      Inf Norm/dgemm
#               10              0.000013        3025.0          0.000007        3025.0
# CLEAN-UP ...
[pdwan@csserver Assignment4]$ █
```

RUNNING A4-MPI-SOLO : STANDALONE

The compiled .c program may also be run standalone. Usage and sample execution follows :



```
pdwan@csicluster:~/exercises/Assignment4$ ./A4-mpi-solo

ERROR:  <number of arguments> [1] : is invalid, less than <default> [5].

USAGE : <program name> [<-r>|<-i>] [N] <matrix contents file>.txt <timing file>.dat

TO :    Calculate |allC| = |A| x |B| using MPI and also calculate infinity norm of |allC|.

WHERE : 1.      <-r>    initialize |A| & |B| with _random_ numbers and |allC| with '0'.
                <-i>    initialize |A| & |B| _incrementally_ with <column> value and |allC| with '0'.
        2.      [N]     max size of each matrix, if invalid defaults to 100.
        3.      <matrix contents file>.txt
                name of .txt file to store values of matrices |A|, |B| & |allC|
        4.      <timing .dat file> .dat
                name of .dat file to contain time to complete for each iteration

        MANUAL :        Straight-forward IJK & DGEMM computations only.
        SOLO :  Straight-forward IJK & MPI computations only.

pdwan@csicluster:~/exercises/Assignment4$ █
```

LOG FILES OBTAINED

Data text files suitable containing the values of the computation used for matrices |A| and |B| and the results stored in |C| are saved in the appropriate log files. File naming convention via the script is :

<data log file name>	`Values-<time>-A4-mpi-<iteration>.txt`
example:	`Values-20140715.170928-A4-mpi-0.txt` `f1.txt`

Single iteration also applies where the user enters arbitrary, valid values for matrix size and does not use the scripts and the other required parameters. Each new matrices |A| and |B| and the results in |C| were saved to the data file, thus simple validation using *LibreOffice Calc.*

A summary file containing processing time for each computation (manual & DGEMM and manual & MPI) is also saved. This is in a format suitable for us with GNUplot.

<timing log file name>	`Data-<time>-A4-mpi-1D.dat`

example:	`Data-20140715.171337-A4-mpi-1D.dat`
	`Data-20140715.172124-A4-mpi-1D.dat`
	`f1.dat`

I did not save a separate .dat file for each run of the script for each algorithm. Instead each .dat file contains the time taken for each matrix size for the preset range of values. **./runAssignment4.sh** may be updated with more if needed but the following are those in use at the moment.

```
#   Matrix - range 1
    declare -a NXArray=( 50 50 50 50 50 50 100 100 100 100 100 100 )
#   Matrix size - range 2
    declare -a NXArray=( 50 50 50 100 100 100 500 500 500 500 1000 1000 1000 1000 )
```

For compilation using the script, a suffix of ***-atlas*** indicates compilation for atlas and a suffix of ***-cblas*** indicates that the c program was compiled via cblas. No suffix indicates compilation using mpicc for MPI.

Finally a log file containing a listing of each algorithm used for that iteration.

After each run, all .log, .txt, .dat and .bup files are copied to the directory *logDir/.*

If either compiled file is used without the script then .dat and .txt files may be named whatever the user wishes and no .log file applies.

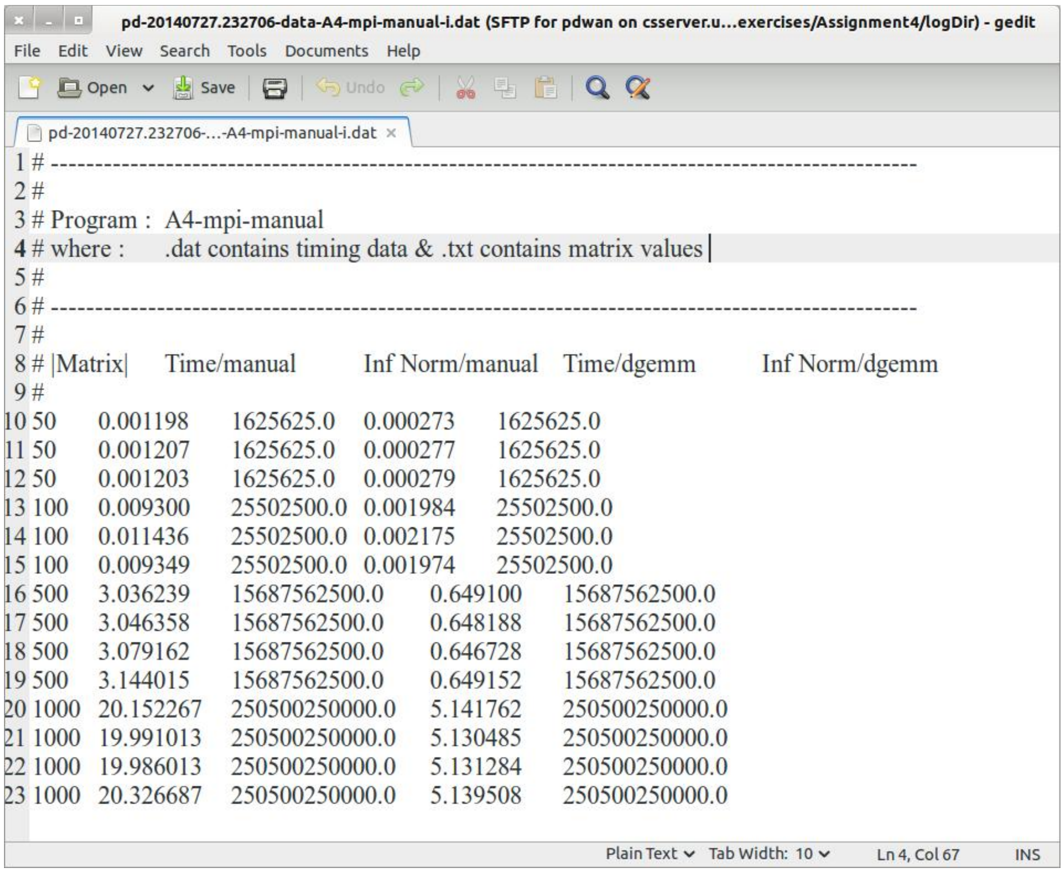
I wished to keep each .c program as clean as possible and so all production setup was completed in the script for each assignment. Thus file creation and validation for each iteration was completed before the .c program was even called. Simple validation of the arguments passed to each .c program is also completed if ran standalone.

I also spot-checked the results as practical. Results spot-check are detailed in Appendix I – Validate Results.

GNUplot Execution

I followed the same structure for each .dat file as produced, an example follows :

Sample .dat file


pd-20140727.232706-data-A4-mpi-manual-i.dat (SFTP for pdwan on csserver.u...exercises/Assignment4/logDir) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
pd-20140727.232706-...-A4-mpi-manual-i.dat
Plain Text
Tab Width: 10
Ln 4, Col 67
INS

```
# ----------------------------------------------------------------------------------------------
#
# Program :  A4-mpi-manual
# where :      .dat contains timing data & .txt contains matrix values
#
# ----------------------------------------------------------------------------------------------
#
# |Matrix|    Time/manual       Inf Norm/manual   Time/dgemm        Inf Norm/dgemm
#
50      0.001198      1625625.0     0.000273      1625625.0
50      0.001207      1625625.0     0.000277      1625625.0
50      0.001203      1625625.0     0.000279      1625625.0
100     0.009300      25502500.0    0.001984      25502500.0
100     0.011436      25502500.0    0.002175      25502500.0
100     0.009349      25502500.0    0.001974      25502500.0
500     3.036239      15687562500.0     0.649100      15687562500.0
500     3.046358      15687562500.0     0.648188      15687562500.0
500     3.079162      15687562500.0     0.646728      15687562500.0
500     3.144015      15687562500.0     0.649152      15687562500.0
1000    20.152267     250500250000.0    5.141762      250500250000.0
1000    19.991013     250500250000.0    5.130485      250500250000.0
1000    19.986013     250500250000.0    5.131284      250500250000.0
1000    20.326687     250500250000.0    5.139508      250500250000.0
```

If wished, the .txt file contains the matrices |A| and |B| used to calculate |C| and the type of computation applicable and the time taken to complete. The .dat file is just a summary of the matrix sizes (when the later is applicable) as well as time taken for each type of computation.

The contents of each .dat was then presented in graphical format using GNUplot, comparing times taken for manual and for BLAS/ATLAS computations.

Sample GNUplot program execution	
	```
	# To execute, launch GNUplot and run :
	# gnuplot> load <filename.gp>
	# making sure that the data file name used is updated if needed.
	# -------------------------------------------------------------------------

	# Paula Dwan : Assignment 4
	reset
	set xtic auto
	set ytic auto
	set size 1,1
	set grid
	set key outside
	#
	set title 'mpi : Matrix size -v- Time taken'
	set ylabel 'Time taken / seconds'
	set xlabel 'Matrix size'
	set xrange [0:1000]
	set yrange [0:25]
	set xtics (0,100,200,300,400,500,600,700,800,900,1000)
	set ytics (0,5,10,15,20,25)
	set origin 0,0
	set key outside
	plot 'logDir/Data-mpi.dat' u 1:2 t 'manual' w l lw 0.8 lc rgb 'blue',
	'logDir/Data-mpi.dat' u 1:4 t 'dgemm' w l lw 0.8 lc rgb 'red'
	#
	pause -1
	```

Thankfully for Linux (Ubuntu) – I could install and run GNUplot locally.

Screen shots of each were taken and added to the section Summary Results.

DGEMM Computation on CSICLUSTER – Why Not?

Basic Linear Algebra Subprograms (BLAS) are a specified set of low-level subroutines that are used in this exercise for matrix multiplication (Level 3 BLAS). Compilation may be completed using ATLAS (Automatically Tuned Linear Algebra Software) or cBLAS.

So in this exercise, we calculate :

`|C| += |A| x |B|`

Attempting to compile for cblas or for atlas on csisluster gave the following error.

```
pdwan@csicluster:~/exercises/Assignment4$ mpicc -I/home/cs/khasanov/libs/CBLAS/src
A4-mpi.c -o A4-mpi  /home/cs/khasanov/libs/cblas_LINUX.a  /usr/lib/libblas.a -lgfortran
  /usr/bin/ld: i386 architecture of input file
`/home/cs/khasanov/libs/cblas_LINUX.a(cblas_dgemm.o)' is incompatible with i386:x86-64
output
  /usr/bin/ld: i386 architecture of input file
`/home/cs/khasanov/libs/cblas_LINUX.a(cblas_globals.o)' is incompatible with i386:x86-64
output
  /usr/bin/ld: i386 architecture of input file
`/home/cs/khasanov/libs/cblas_LINUX.a(cblas_xerbla.o)' is incompatible with i386:x86-64
output
```

Hence, I compiled A4-mpi-manual.c on ***csserver*** and A4-mpi-solo.c on ***csicluster***. I also compiled manual Straight-forward IJK for each to ensure better validation.

Overview of MPI functions Used

Message Passing Interface (MPI) supports communication between processors, i.e.: parallel programming. MPI is a standardized implementation, in this case written in C, and easily calculated the segment of the matrix A as sent to that processor for computation.

In this .c program implementing MPI, we have a fixed number of processors [P] using a [N]x[N] sized matrix for |A|, |B|, and |C|. Each processor can communicate via calls to MPI communication primitives. Each will use collective communication which involves a group of processors.

The MPI functions used are *(not necessarily in order of use)* :

- Ø MPI_Init (&argc, &argv)

  When the program starts only one process is in use (the root process), MPI_Init initializes the run time environment creating the child processes.

  Syntax :

  ```
  int MPI_Init(int *argc, char ***argv)
  ```

  where :

&argc	*argc*	*C/C++ only: Pointer to the number of arguments.* *Number of arguments in program.*
&argv	*argv*	*C/C++ only: Argument vector.* *Listing of arguments applicable to program.*

- Ø MPI_Comm_rank (MPI_COMM_WORLD, &rank)

  Returns the rank of the current process.

  Syntax :

  ```
  int MPI_Comm_rank(MPI_Comm comm, int *rank)
  ```

  where :

MPI_COMM_WORLD	*comm*	*Communicator (handle).* *Default communicator* *communicator of all processes making up the MPI program*
&rank	*rank*	*Rank of the calling process in group of comm (integer).* *Returns the rank of all the processes in the group.*

- Ø MPI_Comm_size (MPI_COMM_WORLD, &size)

  Returns the number of processes requested for the the job

  Syntax :

  ```
  int MPI_Comm_size(MPI_Comm comm, int *size)
  ```

  where :

MPI_COMM_WORLD	*comm*	*Communicator (handle).* *Default communicator* *communicator of all processes making up the MPI program*
&size	*size*	*Number of processes in the group of comm (integer).* *Number of processes in the group*

- MPI_Bcast (B, nx*ny, MPI_DOUBLE, 0, MPI_COMM_WORLD)
  Broadcasts a message from the root process to all other processes in the group.
  Syntax :
  ```
  int MPI_Bcast(void *buffer, int count, MPI_Datatype datatype, int root, MPI_Comm comm)
  ```
  where :

B	*buffer*	*Starting address of buffer (choice).* *Matrix |B|*
*nx*ny*	*count*	*Number of entries in buffer (integer).* *[N] x [N] matrix size of [row] x [column]*
MPI_DOUBLE	*datatype*	*Data type of buffer (handle).* *Data item is an integer : size of matrix [row] x [column]*
0	*root*	*Rank of broadcast root (integer).* *Root process reference*
MPI_COMM_WORLD	*comm*	*Communicator (handle).* *Default communicator* *communicator of all processes making up the MPI program*

- MPI_Scatter (*A, nx*ny/np, MPI_DOUBLE, A + offset, nx*ny/np, MPI_DOUBLE, 0, MPI_COMM_WORLD*)
  Sends data from one task to all tasks in a group.
  Syntax :
  ```
  int MPI_Scatter(const void *sendbuf, int sendcount, MPI_Datatype sendcount,
      void *recvbuf, int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm)
  ```
  where :

A	*sendbuf*	*Address of send buffer (choice, significant only at root).* *matrix |A| → full*
*nx*ny/np*	*sendcount root).*	*Number of elements in send buffer (integer, significant only at* *Size of buffer – so [row] x [column] / [no. of processors]*
MPI_DOUBLE	*sendtype*	*Datatype of send buffer elements (handle, significant only at root).* *→ here, double.*
A + offset	*recvbuf*	*Address of receive buffer (choice).* *matrix |A| → subset only*
*nx*ny/np*	*recvcount*	*Number of elements in receive buffer (integer).* *size of buffer – so [row] x [column] / [no. of processors]*
MPI_DOUBLE	*recvtype*	*Datatype of receive buffer elements (handle).* *→ here, double.*
0	*root*	*Rank of broadcast root (integer).* *Root process reference*
MPI_COMM_WORLD	*comm*	*Communicator (handle).* *Default communicator* *communicator of all processes making up the MPI program*

- MPI_Finalize ()
  Closes the run-time environment. No more MPI calls may be made once this function is executed, not even MPI_Init().
  Syntax :
  ```
  int MPI_Finalize(void)
  ```

Ø MPI_Wtime ()

MPI_Wtime returns a floating-point number of seconds, representing elapsed wall-clock time since some time in the past.

Syntax :

```
double MPI_Wtime()
```

Ø MPI_Barrier

Blocks until all processes in the communicator have reached this routine.

Syntax :

```
int MPI_Barrier(MPI_Comm comm)
```

where :

MPI_COMM_WORLD	*comm*	*Communicator (handle).* *Default communicator* *communicator of all processes making up the MPI program*

Ø MPI_Gather (C+offset, nx*ny/np, MPI_DOUBLE, C, nx*ny/np, MPI_DOUBLE, 0, MPI_COMM_WORLD)

Gather / collect the distributed blocks and return them to the root process.

Syntax :

```
int MPI_Gather(const void *sendbuf, int sendcount, MPI_Datatype sendtype,
    void *recvbuf, int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm)
```

where :

C + offset	*sendbuf*	*Starting address of send buffer (choice).* *matrix |C| → subset only*
*nx*ny/np*	*sendcount*	*Number of elements in send buffer (integer).* *size of buffer – so [row] x [column] / [no. of processors]*
MPI_DOUBLE	*sendtype*	*Datatype of send buffer elements (handle).* *→ here, double.*
C	*recvbuf*	*Address of receive buffer (choice, significant only at root).* *matrix |C| → full*
*nx*ny/np*	*recvcount*	*Number of elements for any single receive (integer, significant only at root).* *As previously,* *size of buffer – so [row] x [column] / [no. of processors]*
MPI_DOUBLE	*recvtype r*	*Datatype of recvbuffer elements (handle, significant only at root).* *→ here, double.*
0	*root*	*Rank of receiving process (integer).*
MPI_COMM_WORLD	comm	Communicator (handle). *Default communicator* *communicator of all processes making up the MPI program*

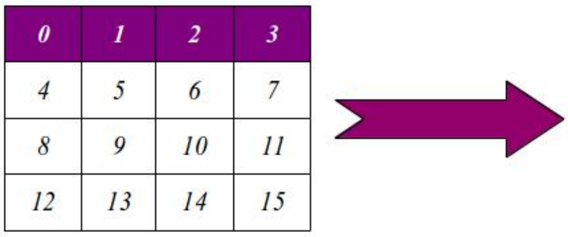
Matrix Sizes Evaluated

When applying the three options I used matrices |A| and |B| of varying sizes from [10x10] to [1000x1000]. The values in each matrix were dependent on the switch

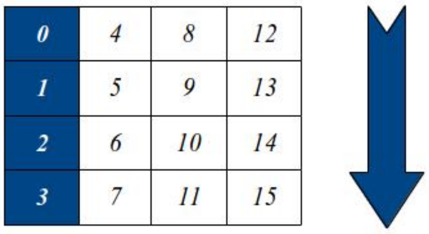
- **-r**
  random from 1 to 10, so each cell regardless of matrix size had a value of 1 to 10 inclusive. This reduced computation time based on large cell values.
- **-i**
  increment based on column index + 1, so all cells in column 1000 had a value of 1001. This could increase computation time of larger matrices as the cell would have comparatively larger values also.

Row-Major as applied

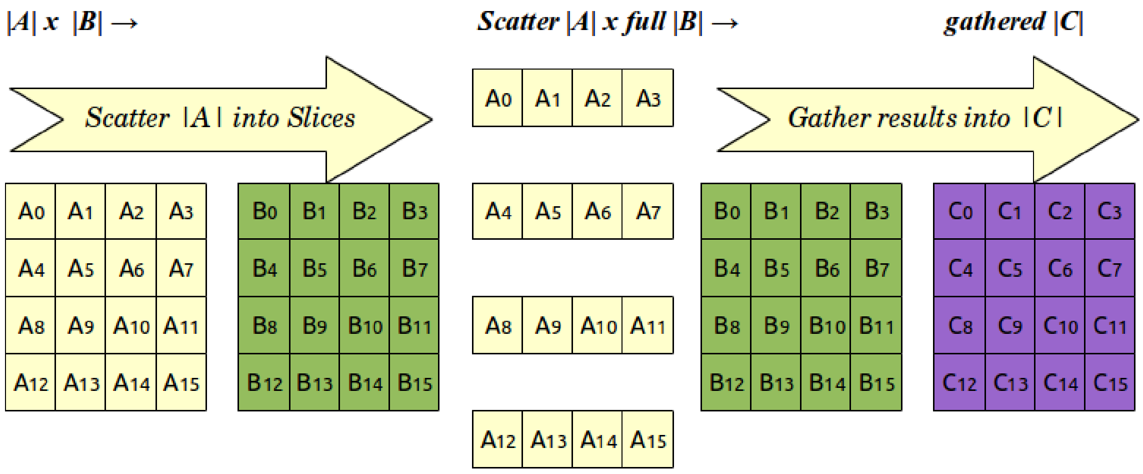
For the straight-forward algorithm of $|C|_{[ij]} \mathrel{+}= |A|_{[ik]} * |B|_{[kj]}$ uses row major as follows (e.g.: [4x4] matrix:



Whereas column-major is as follows :



So for this assignment (variant applied) we use :



Summary Results

Build/plot:	• The dependence of the execution time of the program on the matrix size n. • The speedup over a serial counterpart of the program.
Variant :	• *One-step algorithm. No intermediate resulting matrix.* • *Left matrix is horizontally partitioned* • *The maximum absolute row sum norm (aka infinity-norm):* $$\|A\|_\infty = \max_{0\leq i<n} \sum_{j=0}^{n-1} \|a_{ij}\|$$
Infinity norm	*Sum the absolute values along each row and then take the largest value as the answer.* *Example:* *A =* $\begin{vmatrix} 1 & -7 \\ -2 & -3 \end{vmatrix}$ *then matrix norm of A = max (1 + |-7|, |-2| + |-3|) = max (8, 5) =* ***8***

Comparison MPI, Open MP & pthreads

As this is the fourth assignment in a series using pthreads, open MP and MPI, it is probably worth comparing each at this stage, at a very high level.

OpenMP implementation utilities thread spawning and thread-to-processor allocation. This means less memory usage and direct thread to thread communication. However, threads must be synchronized and joined, thus increasing the time taken to execute the program particularly for larger matrix sizes. Open MP optimizes program performance which runs on a single machine using multiple CPU's.

PThreads is essentially OpenMP on a lower level. It uses #pragmas (basically, code comments) to indicate to the compiler that this is where threading applies.

Message Passing Interface (MPI), basically divides a complex problem into smaller parts which can be solved on individual machines. MPI uses direct inter-process communication via bus messages. Each process (program segment) is standalone and no memory is thus shared.

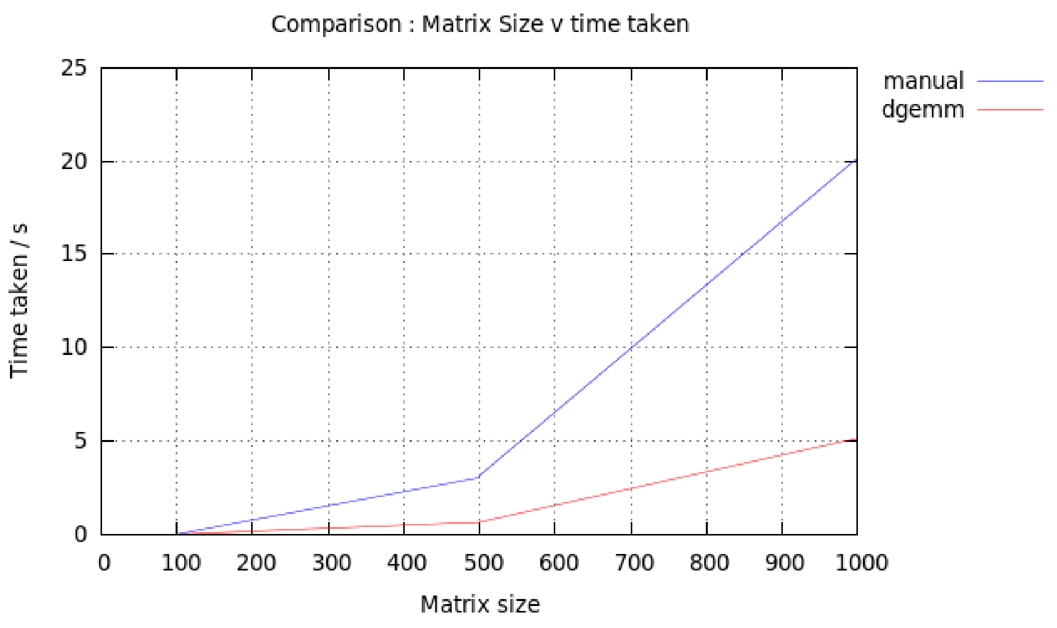
GNUPLOT GRAPHS – MATRIX SIZES 50 → 1000

$ ./runAssignment4.sh -i -v : manual – sample set

Data

|Matrix|	|Processors|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
50		0.001198	1,625,625.0	0.000273	1,625,625.0
50		0.001207	1,625,625.0	0.000277	1,625,625.0
50		0.001203	1,625,625.0	0.000279	1,625,625.0
100		0.009300	25,502,500.0	0.001984	25,502,500.0
100		0.011436	25,502,500.0	0.002175	25,502,500.0
100		0.009349	25,502,500.0	0.001974	25,502,500.0
500		3.036239	15,687,562,500.0	0.649100	15,687,562,500.0
500		3.046358	15,687,562,500.0	0.648188	15,687,562,500.0
500		3.079162	15,687,562,500.0	0.646728	15,687,562,500.0
500		3.144015	15,687,562,500.0	0.649152	15,687,562,500.0
1000		20.152267	250,500,250,000.0	5.141762	250,500,250,000.0
1000		19.991013	250,500,250,000.0	5.130485	250,500,250,000.0
1000		19.986013	250,500,250,000.0	5.131284	250,500,250,000.0
1000		20.326687	250,500,250,000.0	5.139508	250,500,250,000.0

Graph

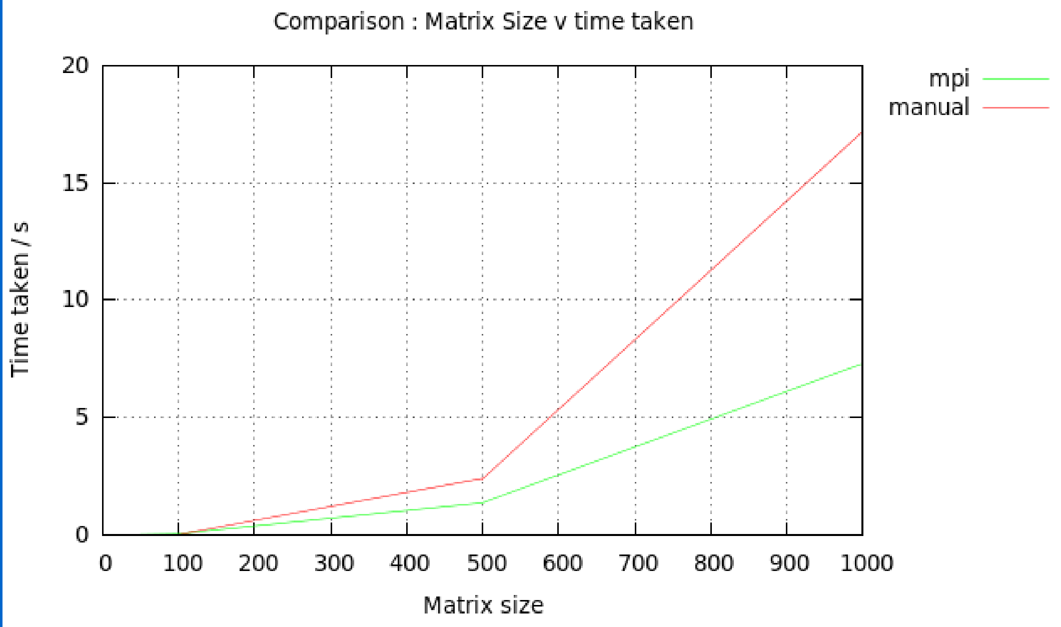


$ ./runAssignment4.sh -i -v : solo | MPI – sample set

Data

|Matrix|	|Processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
50	5	0.009033	75,625.0	0.001095	75,625.0
50	5	0.008928	75,625.0	0.001123	75,625.0
50	5	0.008975	75,625.0	0.001087	75,625.0
100	5	0.038091	1,102,500.0	0.008466	1,102,500.0
100	5	0.038131	1,102,500.0	0.008528	1,102,500.0
100	5	0.038197	1,102,500.0	0.008449	1,102,500.0
500	5	1.358874	637,562,500.0	2.393659	637,562,500.0
500	5	1.367390	637,562,500.0	2.380557	637,562,500.0
500	5	1.362464	637,562,500.0	2.319429	637,562,500.0
500	5	1.358222	637,562,500.0	2.378163	637,562,500.0
1000	5	7.291051	10,100,250,000.0	17.179521	10,100,250,000.0
1000	5	7.340551	10,100,250,000.0	17.143703	10,100,250,000.0
1000	5	7.283727	10,100,250,000.0	17.212086	10,100,250,000.0
1000	5	7.353827	10,100,250,000.0	17.172394	10,100,250,000.0

Graph



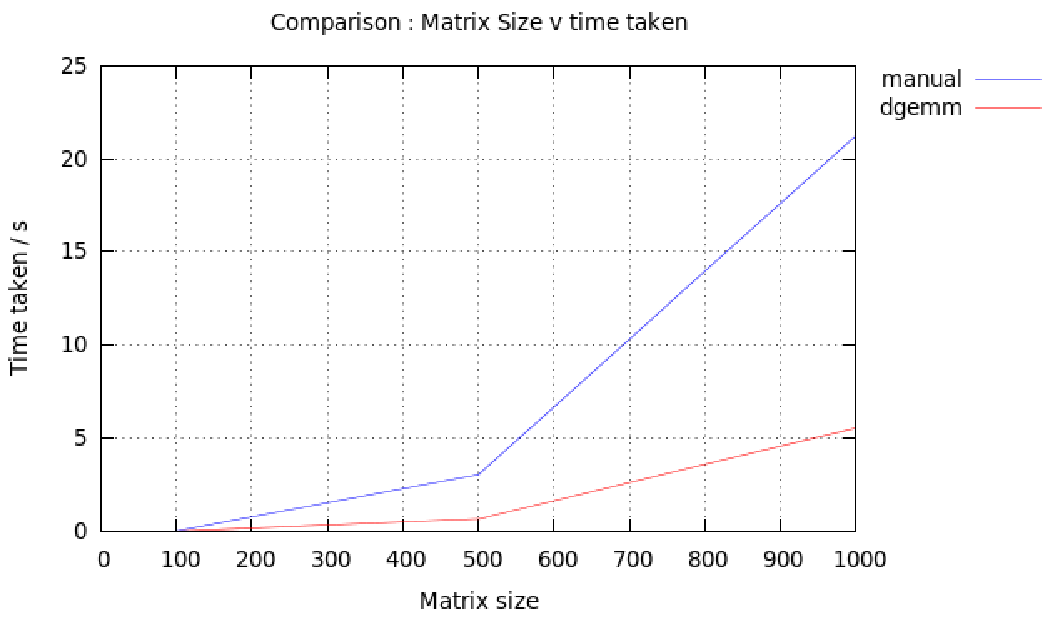
GNUPLOT GRAPHS – MATRIX SIZES 50 → 1000

$ ./runAssignment4.sh -r -v : manual – sample set

Data

|Matrix|	|Processors|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
50		0.001202	89,502.0	0.000280	89,502.0
50		0.001196	89,502.0	0.000276	89,502.0
50		0.001191	89,502.0	0.000278	89,502.0
100		0.009326	353,266.0	0.001976	353,266.0
100		0.009313	353,266.0	0.001984	353,266.0
100		0.009330	353,266.0	0.001985	353,266.0
500		3.039778	8,117,937.0	0.660509	8,117,937.0
500		3.030908	8,117,937.0	0.644442	8,117,937.0
500		3.052243	8,117,937.0	0.645495	8,117,937.0
500		3.028897	8,117,937.0	0.649714	8,117,937.0
1000		21.223333	31,896,067.0	5.537096	31,896,067.0
1000		20.835579	31,896,067.0	5.389093	31,896,067.0
1000		20.613177	31,896,067.0	5.260293	31,896,067.0
1000		20.402660	31,896,067.0	5.315662	31,896,067.0

Graph

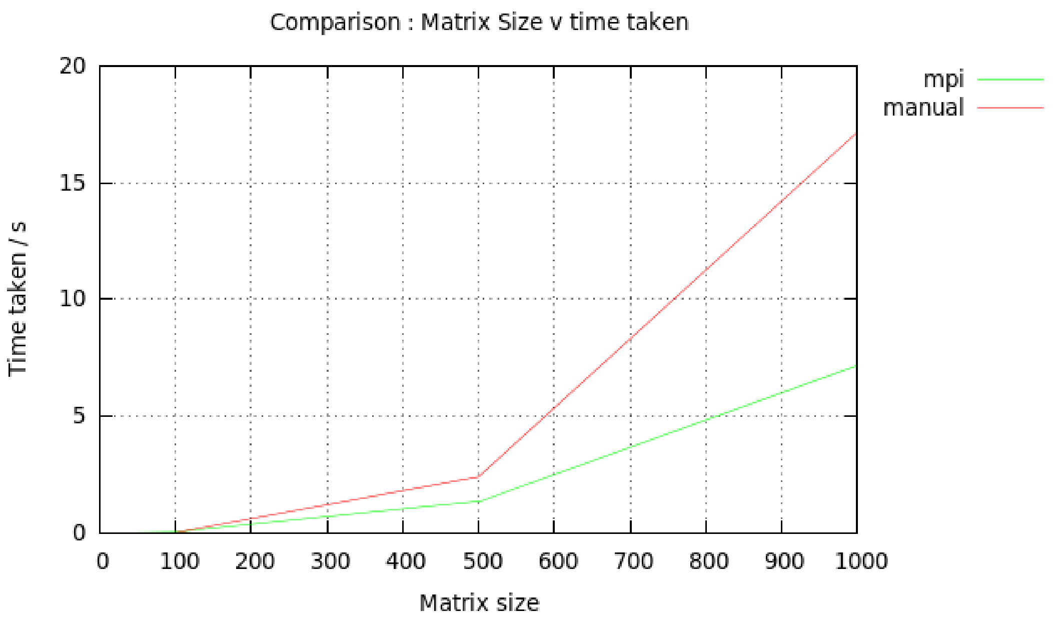


$ ./runAssignment4.sh -r -v : solo | MPI – sample set

Data

|Matrix|	|Processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
50	5	0.009035	83,811.0	0.001208	83,811.0
50	5	0.008932	83,811.0	0.001103	83,811.0
50	5	0.008888	83,811.0	0.001088	83,811.0
100	5	0.037540	338,563.0	0.008523	338,563.0
100	5	0.037636	338,563.0	0.008522	338,563.0
100	5	0.037552	338,563.0	0.008451	338,563.0
500	5	1.350460	8,078,358.0	2.395111	8,078,358.0
500	5	1.328560	8,078,358.0	2.397888	8,078,358.0
500	5	1.359001	8,078,358.0	2.391396	8,078,358.0
500	5	1.323249	8,078,358.0	2.394876	8,078,358.0
1000	5	7.143116	31,600,720.0	17.156587	31,600,720.0
1000	5	7.045932	31,600,720.0	17.116244	31,600,720.0
1000	5	7.149100	31,600,720.0	17.177210	31,600,720.0
1000	5	7.089481	31,600,720.0	17.191204	31,600,720.0

Graph



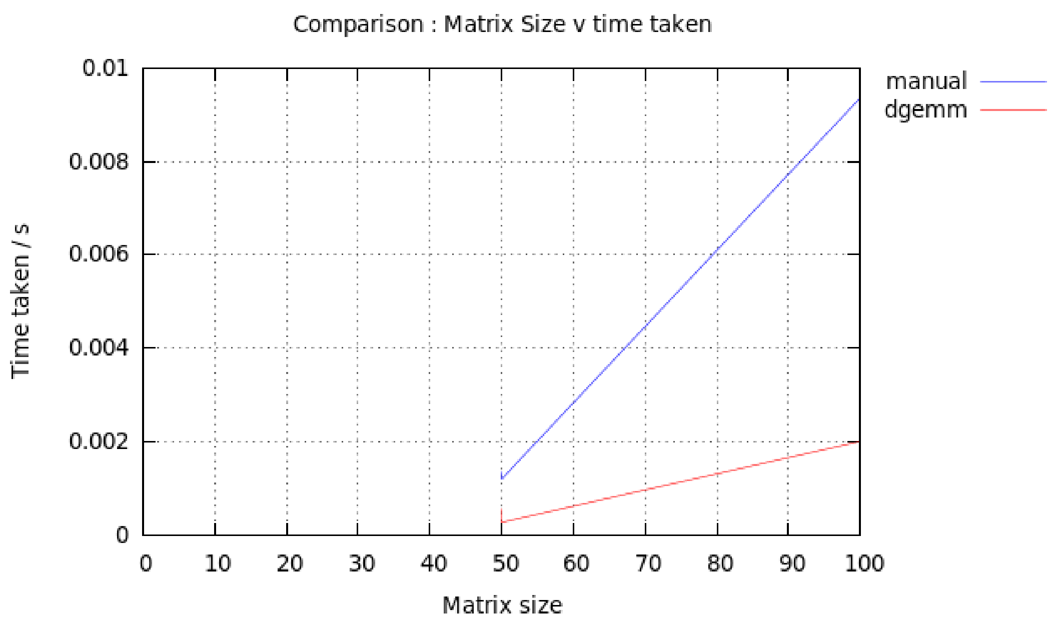
GNUPLOT GRAPHS – MATRIX SIZES 50 → 100

$ ./runAssignment4.sh -r -v : manual – sample set

Data

|Matrix|	|Processors|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
50		0.001339	1,625,625.0	0.000280	1,625,625.0
50		0.001196	1,625,625.0	0.000549	1,625,625.0
50		0.001221	1,625,625.0	0.000327	1,625,625.0
50		0.001198	1,625,625.0	0.000275	1,625,625.0
50		0.001192	1,625,625.0	0.000275	1,625,625.0
50		0.001198	1,625,625.0	0.000273	1,625,625.0
100		0.009352	25,502,500.0	0.001989	25,502,500.0
100		0.010935	25,502,500.0	0.002078	25,502,500.0
100		0.009532	25,502,500.0	0.002036	25,502,500.0
100		0.009452	25,502,500.0	0.003692	25,502,500.0
100		0.009453	25,502,500.0	0.002046	25,502,500.0
100		0.009330	25,502,500.0	0.002878	25,502,500.0

Graph

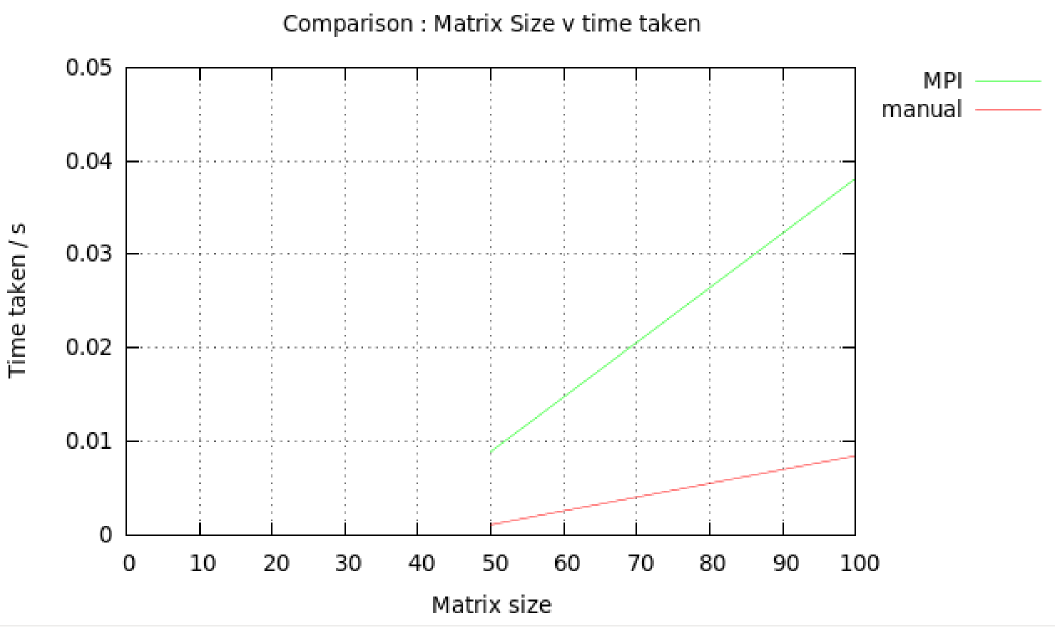


$ ./runAssignment4.sh -i -v : solo | MPI – sample set

Data

|Matrix|	|Processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
50	5	0.008958	75,625.0	0.001088	75,625.0
50	5	0.008924	75,625.0	0.001087	75,625.0
50	5	0.008960	75,625.0	0.001089	75,625.0
50	5	0.008933	75,625.0	0.001089	75,625.0
50	5	0.008925	75,625.0	0.001088	75,625.0
50	5	0.008914	75,625.0	0.001088	75,625.0
100	5	0.038111	1,102,500.0	0.008443	1,102,500.0
100	5	0.038198	1,102,500.0	0.008453	1,102,500.0
100	5	0.038174	1,102,500.0	0.008453	1,102,500.0
100	5	0.038084	1,102,500.0	0.008454	1,102,500.0
100	5	0.038198	1,102,500.0	0.008459	1,102,500.0
100	5	0.038247	1,102,500.0	0.008617	1,102,500.0

Graph



Conclusions

Manual straight-forward IJK computation seems to be consistent in time taken for one or multiple processors with a slight increase when using multiple processors. However this could be perceived to be due to the fact that the cell values were not fully incremental (e.g.: matrix column ha1000 having value if 1001, but instead based on the segment value also → [N] / [P]). This reduced the cell value somewhat but the time taken overall was consistent averaging about 18 to 19 seconds for multiple processors (A4-mpi-solo.c) and 20 to 22 seconds for single processors (A4-mpi-manual.c). Dgemm (cblas only) was computed on single processor only in order to complete the results obtained for large and smaller matrix sizes with the time taken for straight-forward IJK computations. It seemed to remain reasonably consistent for smaller matrices of time taken for dgemm : manual of 1 : 5 and larger matrices of 1 : 4.

For matrices of [100] x [100], MPI was slower than a manual straight-forward IJK computation. However, not surprisingly for large matrices of [1000] x [1000], MPI was usually quicker to compute results by a ratio of MPI : manual of 1 : 2.5. This was the case for both random and incremental cell value initialization for |A| and |B|.

I was restricted to a maximum of five processor on csicluster (csicluster02 still appears to be inoperable). It would be interesting to calculate the results for the large matrices using more processors and confirm if the trend continues, whereby computation time is reduced over that on a single processor : csserver. Or if the amalgamation of the results over a larger number of processors for a similarly large matrix would negate the speed of computing the individual segments.

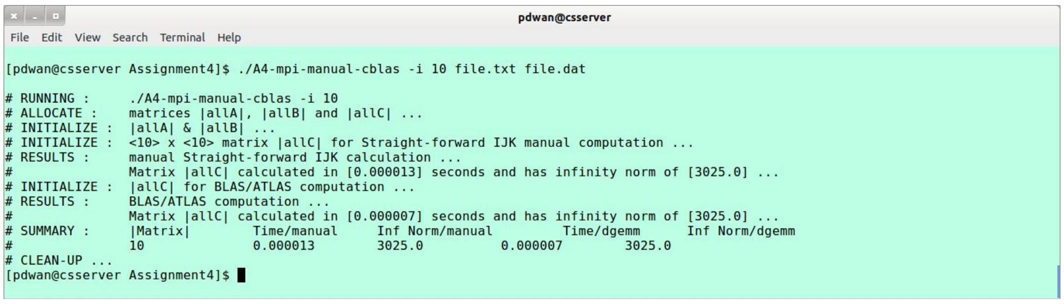
Appendices

Appendix I – Validate Results

Spot check only using 10x10 matrices, initializing matrices |A| and |B| using successive column values.

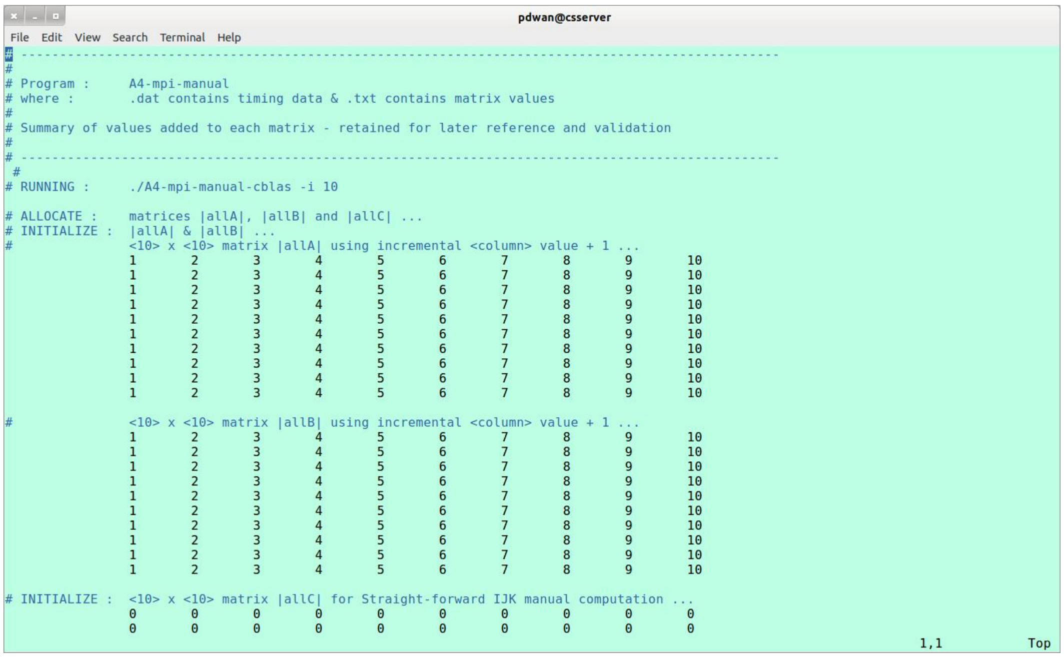
A4-mpi-manual-cblas

Executed using :

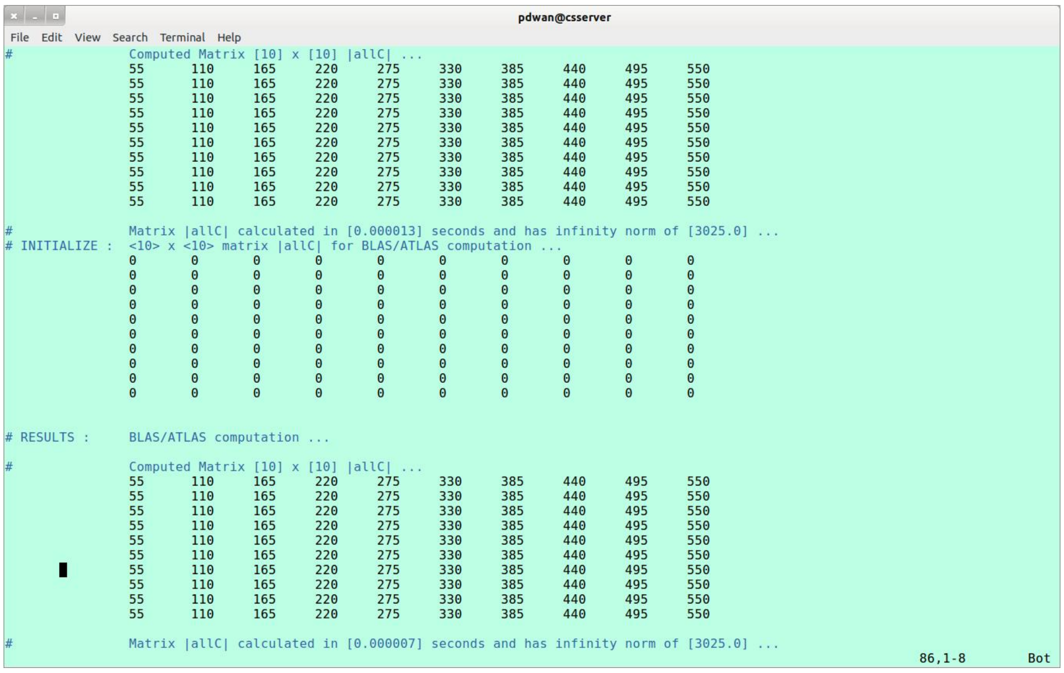


```
[pdwan@csserver Assignment4]$ ./A4-mpi-manual-cblas -i 10 file.txt file.dat

# RUNNING :     ./A4-mpi-manual-cblas -i 10
# ALLOCATE :    matrices |allA|, |allB| and |allC| ...
# INITIALIZE :  |allA| & |allB| ...
# INITIALIZE :  <10> x <10> matrix |allC| for Straight-forward IJK manual computation ...
# RESULTS :     manual Straight-forward IJK calculation ...
#               Matrix |allC| calculated in [0.000013] seconds and has infinity norm of [3025.0] ...
# INITIALIZE :  |allC| for BLAS/ATLAS computation ...
# RESULTS :     BLAS/ATLAS computation ...
#               Matrix |allC| calculated in [0.000007] seconds and has infinity norm of [3025.0] ...
# SUMMARY :     |Matrix|        Time/manual     Inf Norm/manual         Time/dgemm      Inf Norm/dgemm
#               10              0.000013        3025.0          0.000007        3025.0
# CLEAN-UP ...
[pdwan@csserver Assignment4]$
```

Resulting sample Matrix .txt file contains :

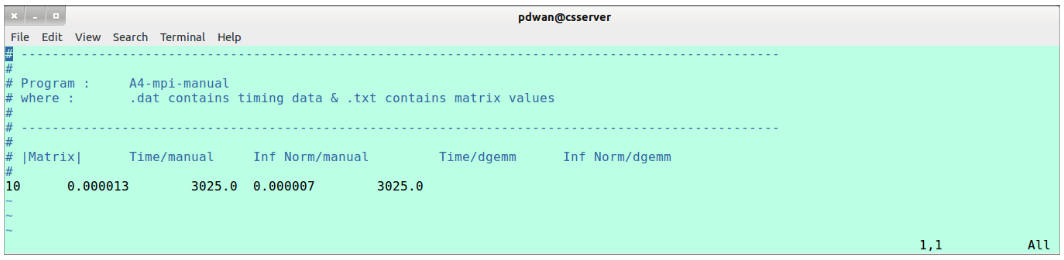


```
# ----------------------------------------------------------------------------------------------------
#
# Program :     A4-mpi-manual
# where :       .dat contains timing data & .txt contains matrix values
#
# Summary of values added to each matrix - retained for later reference and validation
#
# ----------------------------------------------------------------------------------------------------
 #
# RUNNING :     ./A4-mpi-manual-cblas -i 10

# ALLOCATE :    matrices |allA|, |allB| and |allC| ...
# INITIALIZE :  |allA| & |allB| ...
#               <10> x <10> matrix |allA| using incremental <column> value + 1 ...
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10

#               <10> x <10> matrix |allB| using incremental <column> value + 1 ...
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10
                1       2       3       4       5       6       7       8       9       10

# INITIALIZE :  <10> x <10> matrix |allC| for Straight-forward IJK manual computation ...
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                                                                                          1,1           Top
```



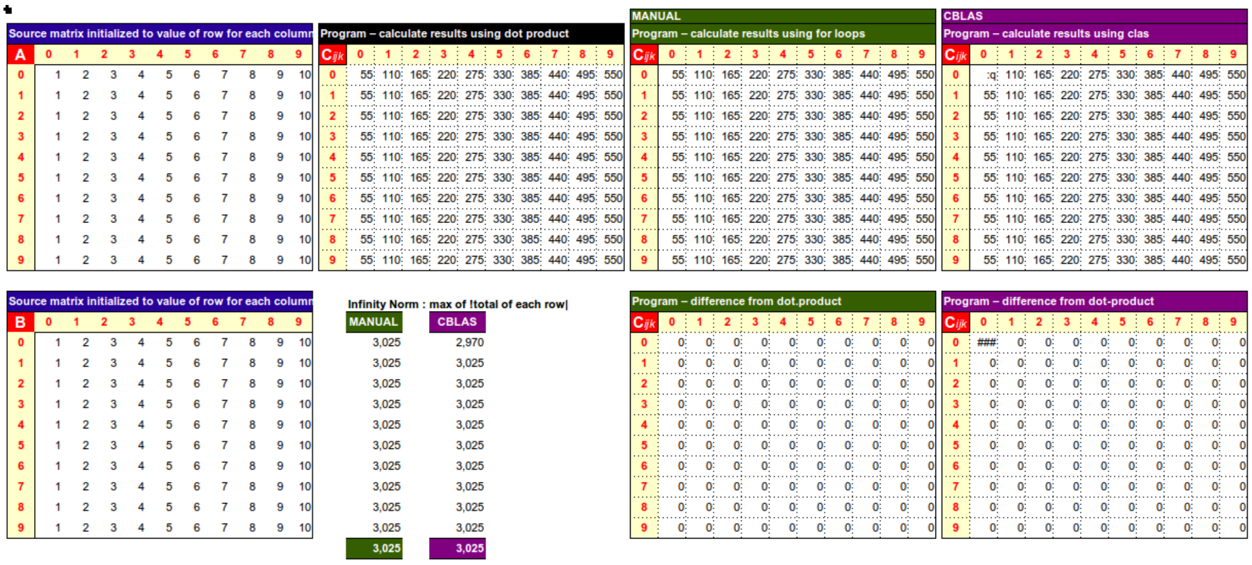
```
#               Computed Matrix [10] x [10] |allC| ...
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550

#               Matrix |allC| calculated in [0.000013] seconds and has infinity norm of [3025.0] ...
# INITIALIZE :  <10> x <10> matrix |allC| for BLAS/ATLAS computation ...
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0
                0       0       0       0       0       0       0       0       0       0


# RESULTS :     BLAS/ATLAS computation ...

#               Computed Matrix [10] x [10] |allC| ...
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550
                55      110     165     220     275     330     385     440     495     550

#               Matrix |allC| calculated in [0.000007] seconds and has infinity norm of [3025.0] ...
                                                                                          86,1-8        Bot
```

Resulting sample summary timing data file contains :



```
pdwan@csserver
File  Edit  View  Search  Terminal  Help
# -----------------------------------------------------------------------------------------
#
# Program :      A4-mpi-manual
# where :        .dat contains timing data & .txt contains matrix values
#
#
# -----------------------------------------------------------------------------------------
#
# |Matrix|       Time/manual      Inf Norm/manual          Time/dgemm       Inf Norm/dgemm
#
10       0.000013         3025.0  0.000007         3025.0
~
~
~
                                                                              1,1           All
```

Validating results for manual computation gives :



Source matrix initialized to value of row for each column										
A	0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10
4	1	2	3	4	5	6	7	8	9	10
5	1	2	3	4	5	6	7	8	9	10
6	1	2	3	4	5	6	7	8	9	10
7	1	2	3	4	5	6	7	8	9	10
8	1	2	3	4	5	6	7	8	9	10
9	1	2	3	4	5	6	7	8	9	10

Program – calculate results using dot product										
C*ijk*	0	1	2	3	4	5	6	7	8	9
0	55	110	165	220	275	330	385	440	495	550
1	55	110	165	220	275	330	385	440	495	550
2	55	110	165	220	275	330	385	440	495	550
3	55	110	165	220	275	330	385	440	495	550
4	55	110	165	220	275	330	385	440	495	550
5	55	110	165	220	275	330	385	440	495	550
6	55	110	165	220	275	330	385	440	495	550
7	55	110	165	220	275	330	385	440	495	550
8	55	110	165	220	275	330	385	440	495	550
9	55	110	165	220	275	330	385	440	495	550

MANUAL										
Program – calculate results using for loops										
C*ijk*	0	1	2	3	4	5	6	7	8	9
0	55	110	165	220	275	330	385	440	495	550
1	55	110	165	220	275	330	385	440	495	550
2	55	110	165	220	275	330	385	440	495	550
3	55	110	165	220	275	330	385	440	495	550
4	55	110	165	220	275	330	385	440	495	550
5	55	110	165	220	275	330	385	440	495	550
6	55	110	165	220	275	330	385	440	495	550
7	55	110	165	220	275	330	385	440	495	550
8	55	110	165	220	275	330	385	440	495	550
9	55	110	165	220	275	330	385	440	495	550

CBLAS										
Program – calculate results using clas										
C*ijk*	0	1	2	3	4	5	6	7	8	9
0	:q	110	165	220	275	330	385	440	495	550
1	55	110	165	220	275	330	385	440	495	550
2	55	110	165	220	275	330	385	440	495	550
3	55	110	165	220	275	330	385	440	495	550
4	55	110	165	220	275	330	385	440	495	550
5	55	110	165	220	275	330	385	440	495	550
6	55	110	165	220	275	330	385	440	495	550
7	55	110	165	220	275	330	385	440	495	550
8	55	110	165	220	275	330	385	440	495	550
9	55	110	165	220	275	330	385	440	495	550

Source matrix initialized to value of row for each column										
B	0	1	2	3	4	5	6	7	8	9
0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10
4	1	2	3	4	5	6	7	8	9	10
5	1	2	3	4	5	6	7	8	9	10
6	1	2	3	4	5	6	7	8	9	10
7	1	2	3	4	5	6	7	8	9	10
8	1	2	3	4	5	6	7	8	9	10
9	1	2	3	4	5	6	7	8	9	10

Infinity Norm : max of !total of each row|

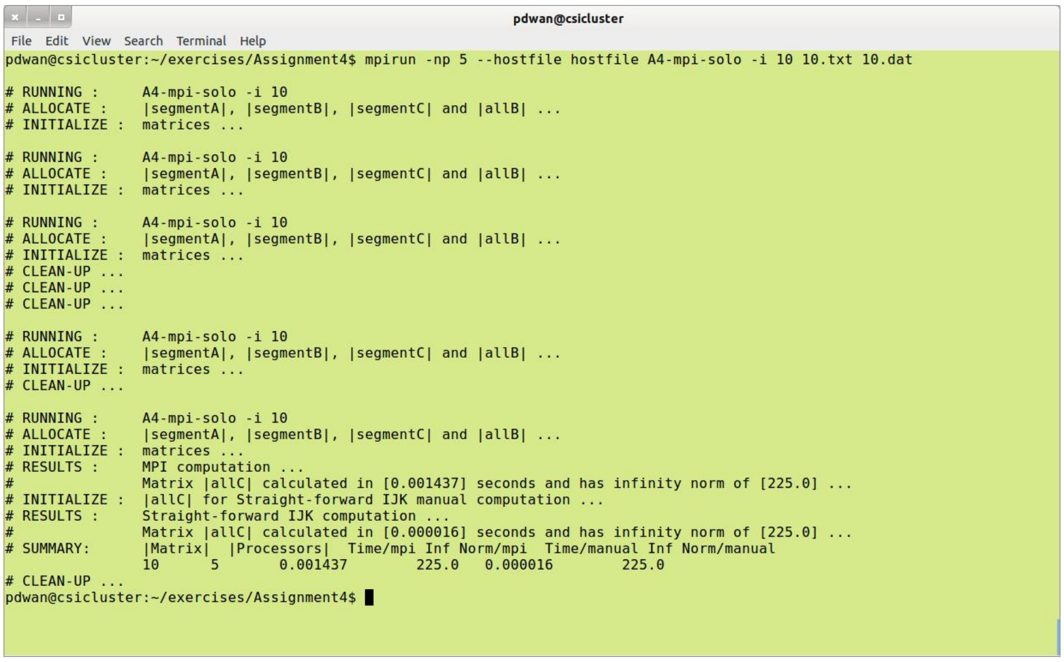
MANUAL	CBLAS
3,025	2,970
3,025	3,025
3,025	3,025
3,025	3,025
3,025	3,025
3,025	3,025
3,025	3,025
3,025	3,025
3,025	3,025
3,025	3,025
3,025	**3,025**

Program – difference from dot.product										
C*ijk*	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

Program – difference from dot-product										
C*ijk*	0	1	2	3	4	5	6	7	8	9
0	###	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

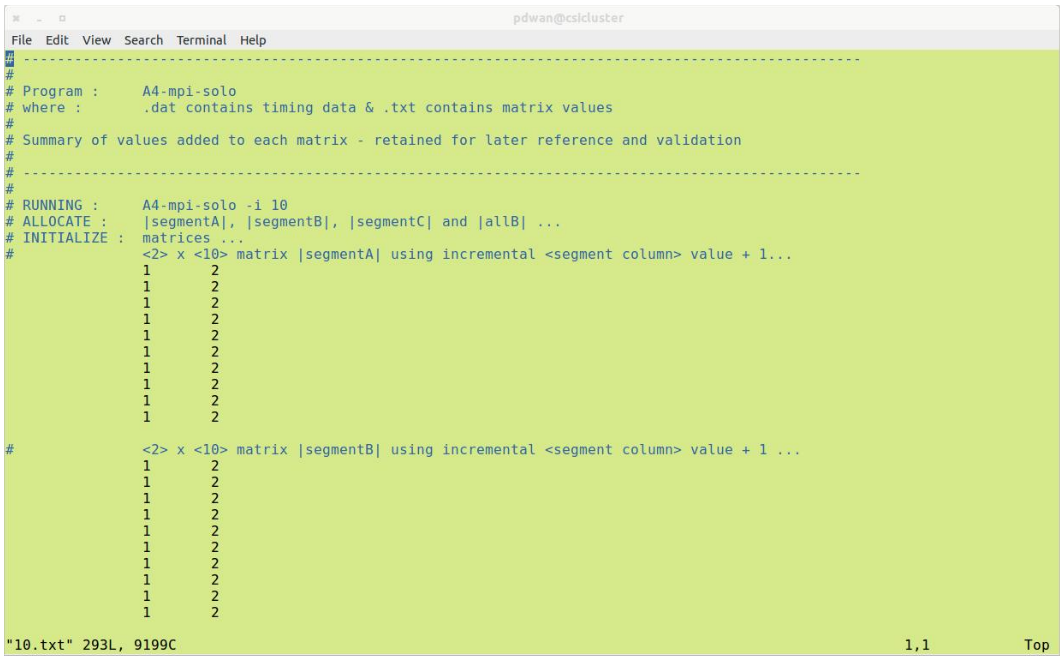
A4-mpi-solo

Executed using :

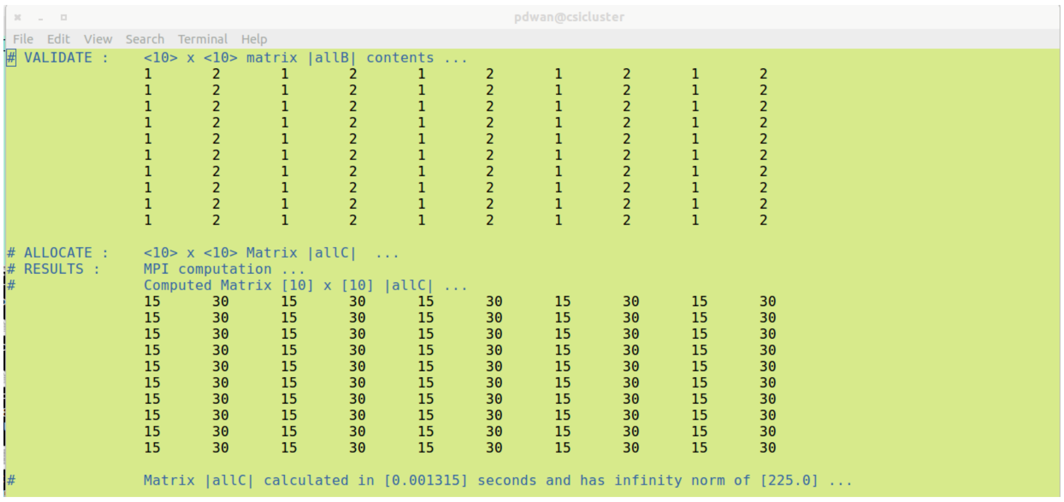


```
pdwan@csicluster
File  Edit  View  Search  Terminal  Help
pdwan@csicluster:~/exercises/Assignment4$ mpirun -np 5 --hostfile hostfile A4-mpi-solo -i 10 10.txt 10.dat

# RUNNING :      A4-mpi-solo -i 10
# ALLOCATE :     |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :   matrices ...

# RUNNING :      A4-mpi-solo -i 10
# ALLOCATE :     |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :   matrices ...

# RUNNING :      A4-mpi-solo -i 10
# ALLOCATE :     |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :   matrices ...
# CLEAN-UP ...
# CLEAN-UP ...
# CLEAN-UP ...

# RUNNING :      A4-mpi-solo -i 10
# ALLOCATE :     |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :   matrices ...
# CLEAN-UP ...

# RUNNING :      A4-mpi-solo -i 10
# ALLOCATE :     |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :   matrices ...
# RESULTS :      MPI computation ...
#                Matrix |allC| calculated in [0.001437] seconds and has infinity norm of [225.0] ...
# INITIALIZE :   |allC| for Straight-forward IJK manual computation ...
# RESULTS :      Straight-forward IJK computation ...
#                Matrix |allC| calculated in [0.000016] seconds and has infinity norm of [225.0] ...
# SUMMARY:       |Matrix|  |Processors|  Time/mpi Inf Norm/mpi  Time/manual Inf Norm/manual
                 10      5       0.001437        225.0   0.000016        225.0
# CLEAN-UP ...
pdwan@csicluster:~/exercises/Assignment4$ █
```

Resulting matrix file contains :

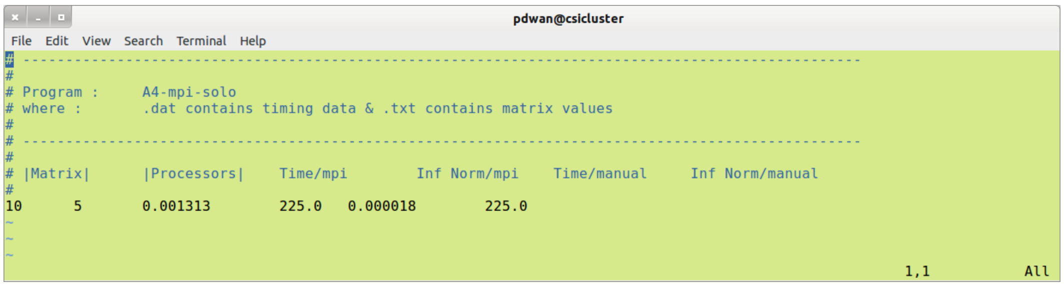


```
# -----------------------------------------------------------------------------------------------------
#
# Program :     A4-mpi-solo
# where :       .dat contains timing data & .txt contains matrix values
#
# Summary of values added to each matrix - retained for later reference and validation
#
# -----------------------------------------------------------------------------------------------------
#
# RUNNING :     A4-mpi-solo -i 10
# ALLOCATE :    |segmentA|, |segmentB|, |segmentC| and |allB| ...
# INITIALIZE :  matrices ...
#               <2> x <10> matrix |segmentA| using incremental <segment column> value + 1...
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2

#               <2> x <10> matrix |segmentB| using incremental <segment column> value + 1 ...
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2
                1       2

"10.txt" 293L, 9199C                                                                1,1           Top
```



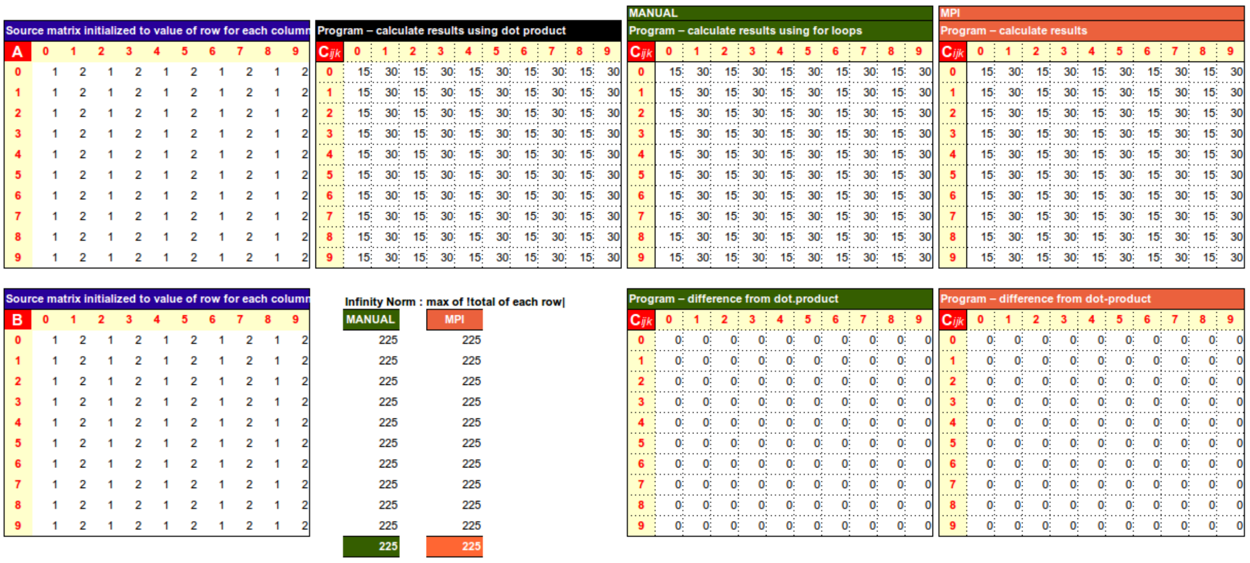
```
# VALIDATE :    <10> x <10> matrix |allB| contents ...
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2
                1       2       1       2       1       2       1       2       1       2

# ALLOCATE :    <10> x <10> Matrix |allC|  ...
# RESULTS :     MPI computation ...
#               Computed Matrix [10] x [10] |allC| ...
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30
                15      30      15      30      15      30      15      30      15      30

#               Matrix |allC| calculated in [0.001315] seconds and has infinity norm of [225.0] ...
```

Resulting sample summary timing data file contains :



```
# -----------------------------------------------------------------------------------------------------
#
# Program :     A4-mpi-solo
# where :       .dat contains timing data & .txt contains matrix values
#
# -----------------------------------------------------------------------------------------------------
#
# |Matrix|      |Processors|    Time/mpi        Inf Norm/mpi    Time/manual     Inf Norm/manual
#
10      5       0.001313        225.0   0.000018        225.0
~
~
~
                                                                                    1,1           All
```

Validating results for mpi computation gives :



Source matrix initialized to value of row for each column

A	0	1	2	3	4	5	6	7	8	9
0	1	2	1	2	1	2	1	2	1	2
1	1	2	1	2	1	2	1	2	1	2
2	1	2	1	2	1	2	1	2	1	2
3	1	2	1	2	1	2	1	2	1	2
4	1	2	1	2	1	2	1	2	1	2
5	1	2	1	2	1	2	1	2	1	2
6	1	2	1	2	1	2	1	2	1	2
7	1	2	1	2	1	2	1	2	1	2
8	1	2	1	2	1	2	1	2	1	2
9	1	2	1	2	1	2	1	2	1	2

Program – calculate results using dot product

C_{ijk}	0	1	2	3	4	5	6	7	8	9
0	15	30	15	30	15	30	15	30	15	30
1	15	30	15	30	15	30	15	30	15	30
2	15	30	15	30	15	30	15	30	15	30
3	15	30	15	30	15	30	15	30	15	30
4	15	30	15	30	15	30	15	30	15	30
5	15	30	15	30	15	30	15	30	15	30
6	15	30	15	30	15	30	15	30	15	30
7	15	30	15	30	15	30	15	30	15	30
8	15	30	15	30	15	30	15	30	15	30
9	15	30	15	30	15	30	15	30	15	30

MANUAL

Program – calculate results using for loops

C_{ijk}	0	1	2	3	4	5	6	7	8	9
0	15	30	15	30	15	30	15	30	15	30
1	15	30	15	30	15	30	15	30	15	30
2	15	30	15	30	15	30	15	30	15	30
3	15	30	15	30	15	30	15	30	15	30
4	15	30	15	30	15	30	15	30	15	30
5	15	30	15	30	15	30	15	30	15	30
6	15	30	15	30	15	30	15	30	15	30
7	15	30	15	30	15	30	15	30	15	30
8	15	30	15	30	15	30	15	30	15	30
9	15	30	15	30	15	30	15	30	15	30

MPI

Program – calculate results

C_{ijk}	0	1	2	3	4	5	6	7	8	9
0	15	30	15	30	15	30	15	30	15	30
1	15	30	15	30	15	30	15	30	15	30
2	15	30	15	30	15	30	15	30	15	30
3	15	30	15	30	15	30	15	30	15	30
4	15	30	15	30	15	30	15	30	15	30
5	15	30	15	30	15	30	15	30	15	30
6	15	30	15	30	15	30	15	30	15	30
7	15	30	15	30	15	30	15	30	15	30
8	15	30	15	30	15	30	15	30	15	30
9	15	30	15	30	15	30	15	30	15	30

Source matrix initialized to value of row for each column

B	0	1	2	3	4	5	6	7	8	9
0	1	2	1	2	1	2	1	2	1	2
1	1	2	1	2	1	2	1	2	1	2
2	1	2	1	2	1	2	1	2	1	2
3	1	2	1	2	1	2	1	2	1	2
4	1	2	1	2	1	2	1	2	1	2
5	1	2	1	2	1	2	1	2	1	2
6	1	2	1	2	1	2	1	2	1	2
7	1	2	1	2	1	2	1	2	1	2
8	1	2	1	2	1	2	1	2	1	2
9	1	2	1	2	1	2	1	2	1	2

Infinity Norm : max of !total of each row|

MANUAL	MPI
225	225
225	225
225	225
225	225
225	225
225	225
225	225
225	225
225	225
225	225
225	225

Program – difference from dot.product

C_{ijk}	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

Program – difference from dot-product

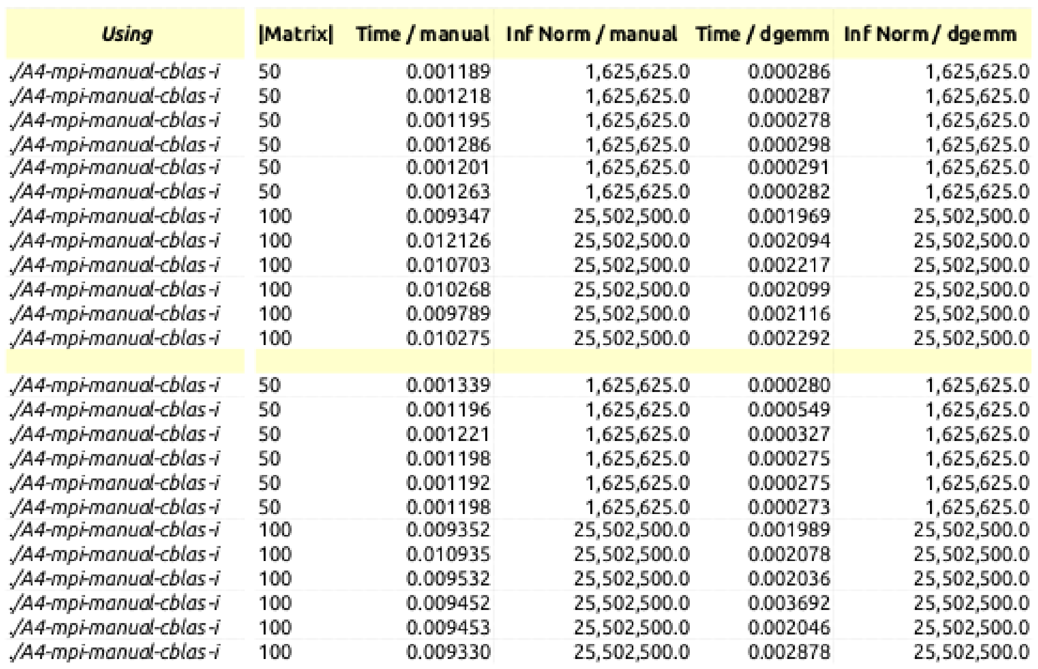
C_{ijk}	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

Appendix II – Summary of Results Obtained

./A4-mpi-manual-cblas -r 50 → 100

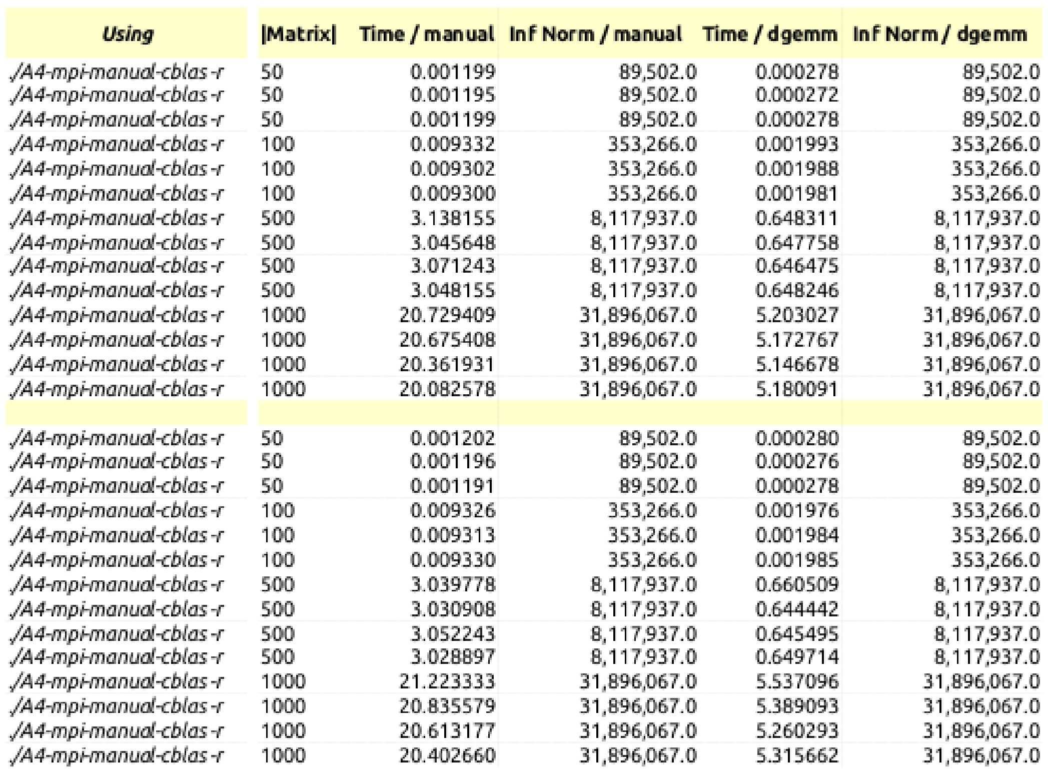
Using	|Matrix|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
./A4-mpi-manual-cblas -r	50	0.001191	89,502.0	0.000292	89,502.0
./A4-mpi-manual-cblas -r	50	0.001198	89,502.0	0.000277	89,502.0
./A4-mpi-manual-cblas -r	50	0.002133	89,502.0	0.000281	89,502.0
./A4-mpi-manual-cblas -r	50	0.001196	89,502.0	0.000276	89,502.0
./A4-mpi-manual-cblas -r	50	0.001186	89,502.0	0.000279	89,502.0
./A4-mpi-manual-cblas -r	50	0.001208	89,502.0	0.000279	89,502.0
./A4-mpi-manual-cblas -r	100	0.009325	353,266.0	0.001976	353,266.0
./A4-mpi-manual-cblas -r	100	0.009433	353,266.0	0.001986	353,266.0
./A4-mpi-manual-cblas -r	100	0.009892	353,266.0	0.002030	353,266.0
./A4-mpi-manual-cblas -r	100	0.009310	353,266.0	0.001984	353,266.0
./A4-mpi-manual-cblas -r	100	0.009311	353,266.0	0.003972	353,266.0
./A4-mpi-manual-cblas -r	100	0.009313	353,266.0	0.002515	353,266.0
./A4-mpi-manual-cblas -r	50	0.001187	89,502.0	0.000280	89,502.0
./A4-mpi-manual-cblas -r	50	0.001201	89,502.0	0.000281	89,502.0
./A4-mpi-manual-cblas -r	50	0.001240	89,502.0	0.000280	89,502.0
./A4-mpi-manual-cblas -r	50	0.001190	89,502.0	0.000279	89,502.0
./A4-mpi-manual-cblas -r	50	0.001583	89,502.0	0.000276	89,502.0
./A4-mpi-manual-cblas -r	50	0.001195	89,502.0	0.000277	89,502.0
./A4-mpi-manual-cblas -r	100	0.009364	353,266.0	0.001984	353,266.0
./A4-mpi-manual-cblas -r	100	0.009343	353,266.0	0.001987	353,266.0
./A4-mpi-manual-cblas -r	100	0.009307	353,266.0	0.001966	353,266.0
./A4-mpi-manual-cblas -r	100	0.009316	353,266.0	0.001985	353,266.0
./A4-mpi-manual-cblas -r	100	0.009366	353,266.0	0.001989	353,266.0
./A4-mpi-manual-cblas -r	100	0.009341	353,266.0	0.002004	353,266.0

./A4-mpi-manual-cblas -i 50 → 100



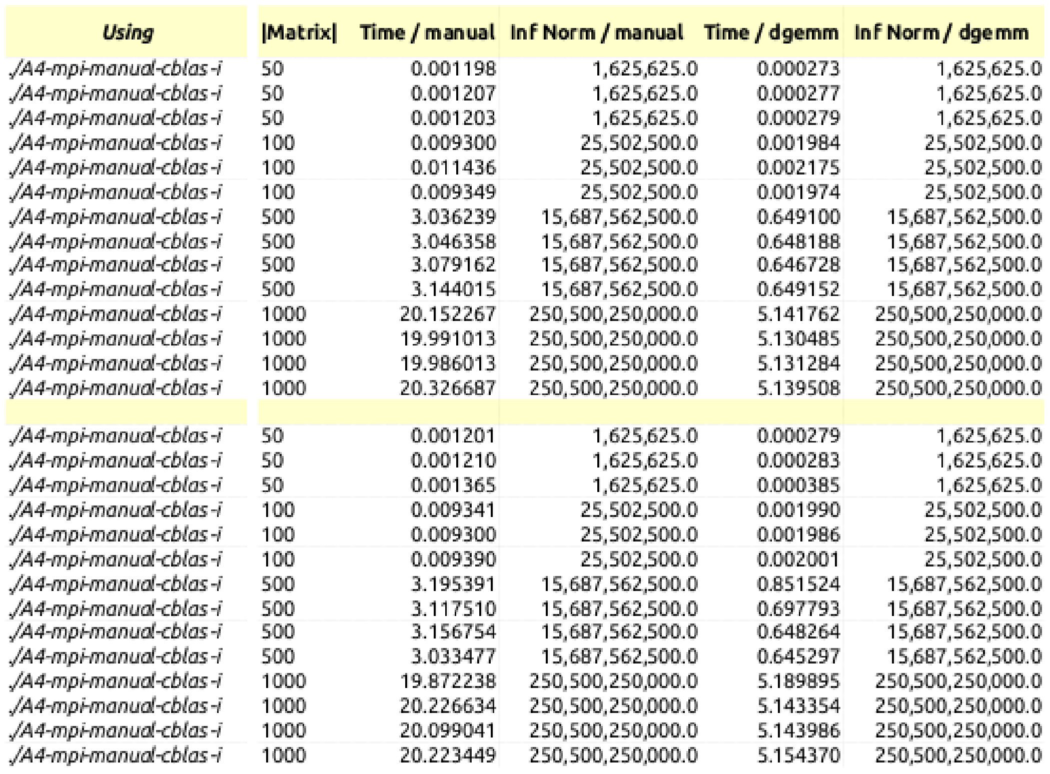
Using	|Matrix|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
./A4-mpi-manual-cblas -i	50	0.001189	1,625,625.0	0.000286	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001218	1,625,625.0	0.000287	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001195	1,625,625.0	0.000278	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001286	1,625,625.0	0.000298	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001201	1,625,625.0	0.000291	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001263	1,625,625.0	0.000282	1,625,625.0
./A4-mpi-manual-cblas -i	100	0.009347	25,502,500.0	0.001969	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.012126	25,502,500.0	0.002094	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.010703	25,502,500.0	0.002217	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.010268	25,502,500.0	0.002099	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009789	25,502,500.0	0.002116	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.010275	25,502,500.0	0.002292	25,502,500.0
./A4-mpi-manual-cblas -i	50	0.001339	1,625,625.0	0.000280	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001196	1,625,625.0	0.000549	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001221	1,625,625.0	0.000327	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001198	1,625,625.0	0.000275	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001192	1,625,625.0	0.000275	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001198	1,625,625.0	0.000273	1,625,625.0
./A4-mpi-manual-cblas -i	100	0.009352	25,502,500.0	0.001989	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.010935	25,502,500.0	0.002078	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009532	25,502,500.0	0.002036	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009452	25,502,500.0	0.003692	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009453	25,502,500.0	0.002046	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009330	25,502,500.0	0.002878	25,502,500.0

./A4-mpi-manual-cblas -r 50 → 1000



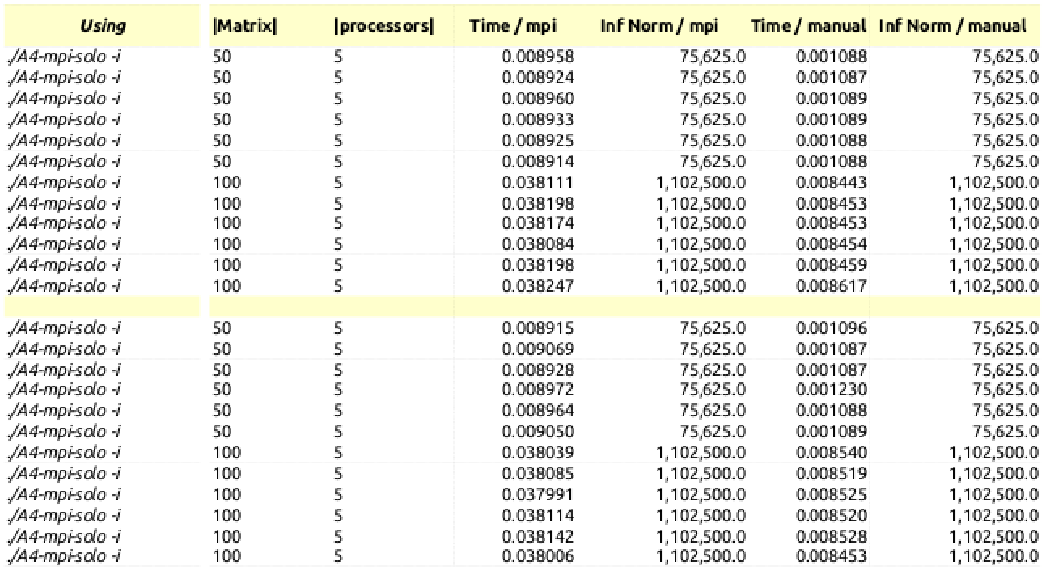
Using	|Matrix|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
./A4-mpi-manual-cblas -r	50	0.001199	89,502.0	0.000278	89,502.0
./A4-mpi-manual-cblas -r	50	0.001195	89,502.0	0.000272	89,502.0
./A4-mpi-manual-cblas -r	50	0.001199	89,502.0	0.000278	89,502.0
./A4-mpi-manual-cblas -r	100	0.009332	353,266.0	0.001993	353,266.0
./A4-mpi-manual-cblas -r	100	0.009302	353,266.0	0.001988	353,266.0
./A4-mpi-manual-cblas -r	100	0.009300	353,266.0	0.001981	353,266.0
./A4-mpi-manual-cblas -r	500	3.138155	8,117,937.0	0.648311	8,117,937.0
./A4-mpi-manual-cblas -r	500	3.045648	8,117,937.0	0.647758	8,117,937.0
./A4-mpi-manual-cblas -r	500	3.071243	8,117,937.0	0.646475	8,117,937.0
./A4-mpi-manual-cblas -r	500	3.048155	8,117,937.0	0.648246	8,117,937.0
./A4-mpi-manual-cblas -r	1000	20.729409	31,896,067.0	5.203027	31,896,067.0
./A4-mpi-manual-cblas -r	1000	20.675408	31,896,067.0	5.172767	31,896,067.0
./A4-mpi-manual-cblas -r	1000	20.361931	31,896,067.0	5.146678	31,896,067.0
./A4-mpi-manual-cblas -r	1000	20.082578	31,896,067.0	5.180091	31,896,067.0
./A4-mpi-manual-cblas -r	50	0.001202	89,502.0	0.000280	89,502.0
./A4-mpi-manual-cblas -r	50	0.001196	89,502.0	0.000276	89,502.0
./A4-mpi-manual-cblas -r	50	0.001191	89,502.0	0.000278	89,502.0
./A4-mpi-manual-cblas -r	100	0.009326	353,266.0	0.001976	353,266.0
./A4-mpi-manual-cblas -r	100	0.009313	353,266.0	0.001984	353,266.0
./A4-mpi-manual-cblas -r	100	0.009330	353,266.0	0.001985	353,266.0
./A4-mpi-manual-cblas -r	500	3.039778	8,117,937.0	0.660509	8,117,937.0
./A4-mpi-manual-cblas -r	500	3.030908	8,117,937.0	0.644442	8,117,937.0
./A4-mpi-manual-cblas -r	500	3.052243	8,117,937.0	0.645495	8,117,937.0
./A4-mpi-manual-cblas -r	500	3.028897	8,117,937.0	0.649714	8,117,937.0
./A4-mpi-manual-cblas -r	1000	21.223333	31,896,067.0	5.537096	31,896,067.0
./A4-mpi-manual-cblas -r	1000	20.835579	31,896,067.0	5.389093	31,896,067.0
./A4-mpi-manual-cblas -r	1000	20.613177	31,896,067.0	5.260293	31,896,067.0
./A4-mpi-manual-cblas -r	1000	20.402660	31,896,067.0	5.315662	31,896,067.0

./A4-mpi-manual-cblas -i 50 → 1000



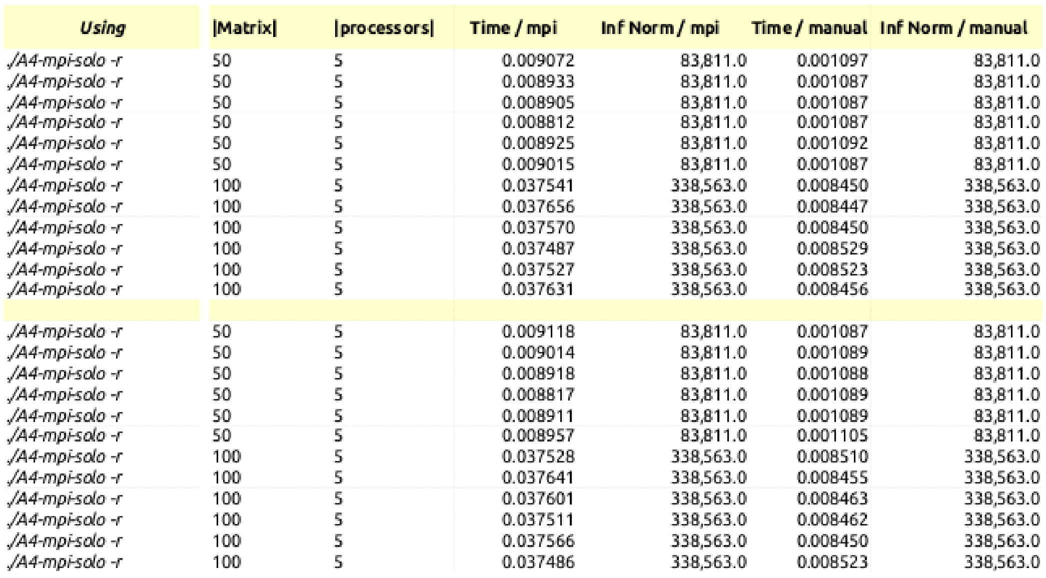
Using	|Matrix|	Time / manual	Inf Norm / manual	Time / dgemm	Inf Norm / dgemm
./A4-mpi-manual-cblas -i	50	0.001198	1,625,625.0	0.000273	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001207	1,625,625.0	0.000277	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001203	1,625,625.0	0.000279	1,625,625.0
./A4-mpi-manual-cblas -i	100	0.009300	25,502,500.0	0.001984	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.011436	25,502,500.0	0.002175	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009349	25,502,500.0	0.001974	25,502,500.0
./A4-mpi-manual-cblas -i	500	3.036239	15,687,562,500.0	0.649100	15,687,562,500.0
./A4-mpi-manual-cblas -i	500	3.046358	15,687,562,500.0	0.648188	15,687,562,500.0
./A4-mpi-manual-cblas -i	500	3.079162	15,687,562,500.0	0.646728	15,687,562,500.0
./A4-mpi-manual-cblas -i	500	3.144015	15,687,562,500.0	0.649152	15,687,562,500.0
./A4-mpi-manual-cblas -i	1000	20.152267	250,500,250,000.0	5.141762	250,500,250,000.0
./A4-mpi-manual-cblas -i	1000	19.991013	250,500,250,000.0	5.130485	250,500,250,000.0
./A4-mpi-manual-cblas -i	1000	19.986013	250,500,250,000.0	5.131284	250,500,250,000.0
./A4-mpi-manual-cblas -i	1000	20.326687	250,500,250,000.0	5.139508	250,500,250,000.0
./A4-mpi-manual-cblas -i	50	0.001201	1,625,625.0	0.000279	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001210	1,625,625.0	0.000283	1,625,625.0
./A4-mpi-manual-cblas -i	50	0.001365	1,625,625.0	0.000385	1,625,625.0
./A4-mpi-manual-cblas -i	100	0.009341	25,502,500.0	0.001990	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009300	25,502,500.0	0.001986	25,502,500.0
./A4-mpi-manual-cblas -i	100	0.009390	25,502,500.0	0.002001	25,502,500.0
./A4-mpi-manual-cblas -i	500	3.195391	15,687,562,500.0	0.851524	15,687,562,500.0
./A4-mpi-manual-cblas -i	500	3.117510	15,687,562,500.0	0.697793	15,687,562,500.0
./A4-mpi-manual-cblas -i	500	3.156754	15,687,562,500.0	0.648264	15,687,562,500.0
./A4-mpi-manual-cblas -i	500	3.033477	15,687,562,500.0	0.645297	15,687,562,500.0
./A4-mpi-manual-cblas -i	1000	19.872238	250,500,250,000.0	5.189895	250,500,250,000.0
./A4-mpi-manual-cblas -i	1000	20.226634	250,500,250,000.0	5.143354	250,500,250,000.0
./A4-mpi-manual-cblas -i	1000	20.099041	250,500,250,000.0	5.143986	250,500,250,000.0
./A4-mpi-manual-cblas -i	1000	20.223449	250,500,250,000.0	5.154370	250,500,250,000.0

./A4-mpi-solo -i 50 → 100



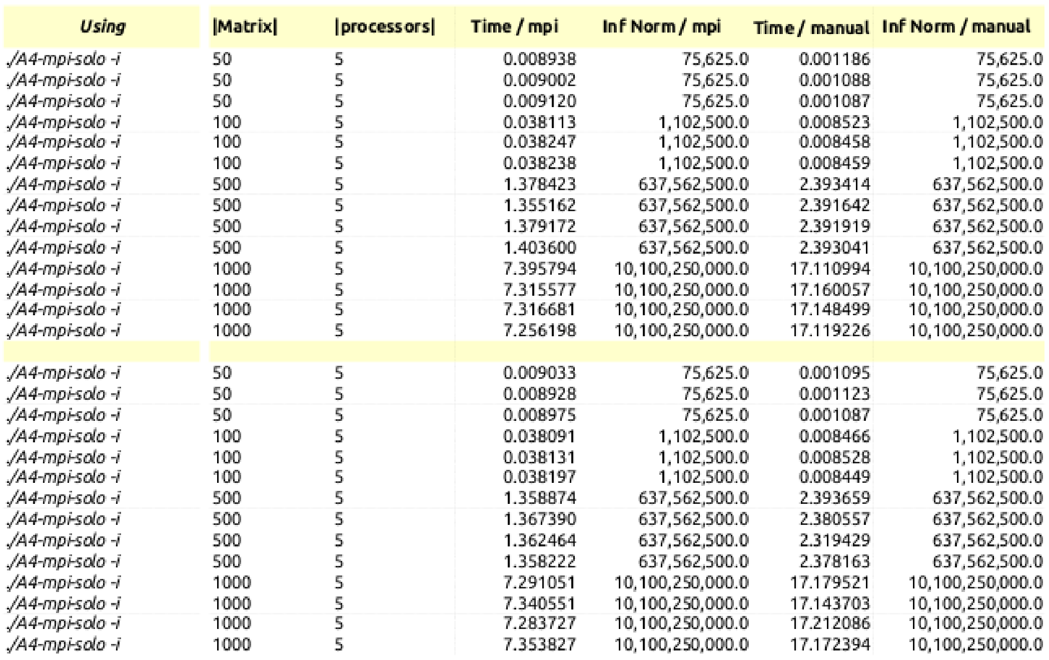
Using	|Matrix|	|processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
./A4-mpi-solo -i	50	5	0.008958	75,625.0	0.001088	75,625.0
./A4-mpi-solo -i	50	5	0.008924	75,625.0	0.001087	75,625.0
./A4-mpi-solo -i	50	5	0.008960	75,625.0	0.001089	75,625.0
./A4-mpi-solo -i	50	5	0.008933	75,625.0	0.001089	75,625.0
./A4-mpi-solo -i	50	5	0.008925	75,625.0	0.001088	75,625.0
./A4-mpi-solo -i	50	5	0.008914	75,625.0	0.001088	75,625.0
./A4-mpi-solo -i	100	5	0.038111	1,102,500.0	0.008443	1,102,500.0
./A4-mpi-solo -i	100	5	0.038198	1,102,500.0	0.008453	1,102,500.0
./A4-mpi-solo -i	100	5	0.038174	1,102,500.0	0.008453	1,102,500.0
./A4-mpi-solo -i	100	5	0.038084	1,102,500.0	0.008454	1,102,500.0
./A4-mpi-solo -i	100	5	0.038198	1,102,500.0	0.008459	1,102,500.0
./A4-mpi-solo -i	100	5	0.038247	1,102,500.0	0.008617	1,102,500.0
./A4-mpi-solo -i	50	5	0.008915	75,625.0	0.001096	75,625.0
./A4-mpi-solo -i	50	5	0.009069	75,625.0	0.001087	75,625.0
./A4-mpi-solo -i	50	5	0.008928	75,625.0	0.001087	75,625.0
./A4-mpi-solo -i	50	5	0.008972	75,625.0	0.001230	75,625.0
./A4-mpi-solo -i	50	5	0.008964	75,625.0	0.001088	75,625.0
./A4-mpi-solo -i	50	5	0.009050	75,625.0	0.001089	75,625.0
./A4-mpi-solo -i	100	5	0.038039	1,102,500.0	0.008540	1,102,500.0
./A4-mpi-solo -i	100	5	0.038085	1,102,500.0	0.008519	1,102,500.0
./A4-mpi-solo -i	100	5	0.037991	1,102,500.0	0.008525	1,102,500.0
./A4-mpi-solo -i	100	5	0.038114	1,102,500.0	0.008520	1,102,500.0
./A4-mpi-solo -i	100	5	0.038142	1,102,500.0	0.008528	1,102,500.0
./A4-mpi-solo -i	100	5	0.038006	1,102,500.0	0.008453	1,102,500.0

./A4-mpi-solo -r 50 → 100



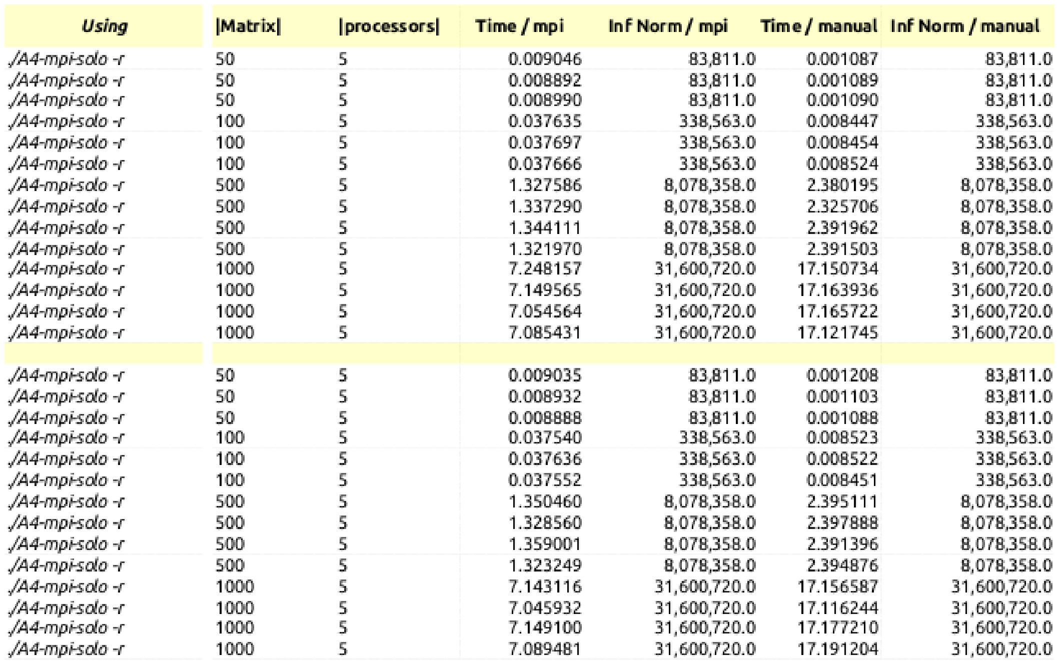
Using	|Matrix|	|processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
./A4-mpi-solo -r	50	5	0.009072	83,811.0	0.001097	83,811.0
./A4-mpi-solo -r	50	5	0.008933	83,811.0	0.001087	83,811.0
./A4-mpi-solo -r	50	5	0.008905	83,811.0	0.001087	83,811.0
./A4-mpi-solo -r	50	5	0.008812	83,811.0	0.001087	83,811.0
./A4-mpi-solo -r	50	5	0.008925	83,811.0	0.001092	83,811.0
./A4-mpi-solo -r	50	5	0.009015	83,811.0	0.001087	83,811.0
./A4-mpi-solo -r	100	5	0.037541	338,563.0	0.008450	338,563.0
./A4-mpi-solo -r	100	5	0.037656	338,563.0	0.008447	338,563.0
./A4-mpi-solo -r	100	5	0.037570	338,563.0	0.008450	338,563.0
./A4-mpi-solo -r	100	5	0.037487	338,563.0	0.008529	338,563.0
./A4-mpi-solo -r	100	5	0.037527	338,563.0	0.008523	338,563.0
./A4-mpi-solo -r	100	5	0.037631	338,563.0	0.008456	338,563.0
./A4-mpi-solo -r	50	5	0.009118	83,811.0	0.001087	83,811.0
./A4-mpi-solo -r	50	5	0.009014	83,811.0	0.001089	83,811.0
./A4-mpi-solo -r	50	5	0.008918	83,811.0	0.001088	83,811.0
./A4-mpi-solo -r	50	5	0.008817	83,811.0	0.001089	83,811.0
./A4-mpi-solo -r	50	5	0.008911	83,811.0	0.001089	83,811.0
./A4-mpi-solo -r	50	5	0.008957	83,811.0	0.001105	83,811.0
./A4-mpi-solo -r	100	5	0.037528	338,563.0	0.008510	338,563.0
./A4-mpi-solo -r	100	5	0.037641	338,563.0	0.008455	338,563.0
./A4-mpi-solo -r	100	5	0.037601	338,563.0	0.008463	338,563.0
./A4-mpi-solo -r	100	5	0.037511	338,563.0	0.008462	338,563.0
./A4-mpi-solo -r	100	5	0.037566	338,563.0	0.008450	338,563.0
./A4-mpi-solo -r	100	5	0.037486	338,563.0	0.008523	338,563.0

./A4-mpi-solo -i 50 → 1000



Using	|Matrix|	|processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
./A4-mpi-solo -i	50	5	0.008938	75,625.0	0.001186	75,625.0
./A4-mpi-solo -i	50	5	0.009002	75,625.0	0.001088	75,625.0
./A4-mpi-solo -i	50	5	0.009120	75,625.0	0.001087	75,625.0
./A4-mpi-solo -i	100	5	0.038113	1,102,500.0	0.008523	1,102,500.0
./A4-mpi-solo -i	100	5	0.038247	1,102,500.0	0.008458	1,102,500.0
./A4-mpi-solo -i	100	5	0.038238	1,102,500.0	0.008459	1,102,500.0
./A4-mpi-solo -i	500	5	1.378423	637,562,500.0	2.393414	637,562,500.0
./A4-mpi-solo -i	500	5	1.355162	637,562,500.0	2.391642	637,562,500.0
./A4-mpi-solo -i	500	5	1.379172	637,562,500.0	2.391919	637,562,500.0
./A4-mpi-solo -i	500	5	1.403600	637,562,500.0	2.393041	637,562,500.0
./A4-mpi-solo -i	1000	5	7.395794	10,100,250,000.0	17.110994	10,100,250,000.0
./A4-mpi-solo -i	1000	5	7.315577	10,100,250,000.0	17.160057	10,100,250,000.0
./A4-mpi-solo -i	1000	5	7.316681	10,100,250,000.0	17.148499	10,100,250,000.0
./A4-mpi-solo -i	1000	5	7.256198	10,100,250,000.0	17.119226	10,100,250,000.0
./A4-mpi-solo -i	50	5	0.009033	75,625.0	0.001095	75,625.0
./A4-mpi-solo -i	50	5	0.008928	75,625.0	0.001123	75,625.0
./A4-mpi-solo -i	50	5	0.008975	75,625.0	0.001087	75,625.0
./A4-mpi-solo -i	100	5	0.038091	1,102,500.0	0.008466	1,102,500.0
./A4-mpi-solo -i	100	5	0.038131	1,102,500.0	0.008528	1,102,500.0
./A4-mpi-solo -i	100	5	0.038197	1,102,500.0	0.008449	1,102,500.0
./A4-mpi-solo -i	500	5	1.358874	637,562,500.0	2.393659	637,562,500.0
./A4-mpi-solo -i	500	5	1.367390	637,562,500.0	2.380557	637,562,500.0
./A4-mpi-solo -i	500	5	1.362464	637,562,500.0	2.319429	637,562,500.0
./A4-mpi-solo -i	500	5	1.358222	637,562,500.0	2.378163	637,562,500.0
./A4-mpi-solo -i	1000	5	7.291051	10,100,250,000.0	17.179521	10,100,250,000.0
./A4-mpi-solo -i	1000	5	7.340551	10,100,250,000.0	17.143703	10,100,250,000.0
./A4-mpi-solo -i	1000	5	7.283727	10,100,250,000.0	17.212086	10,100,250,000.0
./A4-mpi-solo -i	1000	5	7.353827	10,100,250,000.0	17.172394	10,100,250,000.0

./A4-mpi-solo -r 50 → 1000



Using	|Matrix|	|processors|	Time / mpi	Inf Norm / mpi	Time / manual	Inf Norm / manual
./A4-mpi-solo -r	50	5	0.009046	83,811.0	0.001087	83,811.0
./A4-mpi-solo -r	50	5	0.008892	83,811.0	0.001089	83,811.0
./A4-mpi-solo -r	50	5	0.008990	83,811.0	0.001090	83,811.0
./A4-mpi-solo -r	100	5	0.037635	338,563.0	0.008447	338,563.0
./A4-mpi-solo -r	100	5	0.037697	338,563.0	0.008454	338,563.0
./A4-mpi-solo -r	100	5	0.037666	338,563.0	0.008524	338,563.0
./A4-mpi-solo -r	500	5	1.327586	8,078,358.0	2.380195	8,078,358.0
./A4-mpi-solo -r	500	5	1.337290	8,078,358.0	2.325706	8,078,358.0
./A4-mpi-solo -r	500	5	1.344111	8,078,358.0	2.391962	8,078,358.0
./A4-mpi-solo -r	500	5	1.321970	8,078,358.0	2.391503	8,078,358.0
./A4-mpi-solo -r	1000	5	7.248157	31,600,720.0	17.150734	31,600,720.0
./A4-mpi-solo -r	1000	5	7.149565	31,600,720.0	17.163936	31,600,720.0
./A4-mpi-solo -r	1000	5	7.054564	31,600,720.0	17.165722	31,600,720.0
./A4-mpi-solo -r	1000	5	7.085431	31,600,720.0	17.121745	31,600,720.0
./A4-mpi-solo -r	50	5	0.009035	83,811.0	0.001208	83,811.0
./A4-mpi-solo -r	50	5	0.008932	83,811.0	0.001103	83,811.0
./A4-mpi-solo -r	50	5	0.008888	83,811.0	0.001088	83,811.0
./A4-mpi-solo -r	100	5	0.037540	338,563.0	0.008523	338,563.0
./A4-mpi-solo -r	100	5	0.037636	338,563.0	0.008522	338,563.0
./A4-mpi-solo -r	100	5	0.037552	338,563.0	0.008451	338,563.0
./A4-mpi-solo -r	500	5	1.350460	8,078,358.0	2.395111	8,078,358.0
./A4-mpi-solo -r	500	5	1.328560	8,078,358.0	2.397888	8,078,358.0
./A4-mpi-solo -r	500	5	1.359001	8,078,358.0	2.391396	8,078,358.0
./A4-mpi-solo -r	500	5	1.323249	8,078,358.0	2.394876	8,078,358.0
./A4-mpi-solo -r	1000	5	7.143116	31,600,720.0	17.156587	31,600,720.0
./A4-mpi-solo -r	1000	5	7.045932	31,600,720.0	17.116244	31,600,720.0
./A4-mpi-solo -r	1000	5	7.149100	31,600,720.0	17.177210	31,600,720.0
./A4-mpi-solo -r	1000	5	7.089481	31,600,720.0	17.191204	31,600,720.0

Appendix III – References / Acknowledgements

- www.stackoverflow.com : general queries on MPI function not working and possible workaround
- http://en.wikipedia.org/wiki/Basic_Linear_Algebra_Subprograms
- http://en.wikipedia.org/wiki/Message_Passing_Interface
- www.emsl.pnl.gov : Pacific Northwest National Laboratory / Batelle : "Introduction to MPI"
- www.ucd.ie : COMP40700 High Performance Computing – Notes 2014 "Message Passing Libraries"