

# Utilizzo di librerie scientifiche

Fabio Affinito ([f.affinito@cenea.it](mailto:f.affinito@cenea.it))



## Outline

- Un'introduzione alle librerie
- Un case-study: le BLAS
- LAPACK e SCALAPACK
- Librerie “proprietarie”: MKL, ACML e ESSL
- Linking, name mangling e compatibilità C/Fortran
- Esempi: librerie MASS, librerie BLAS, librerie ESSL
- Cenni su altre librerie (FFTW, HDF5...)

## Che cos'è una libreria?

E' una collezione di funzioni, subroutines orientate alla soluzione di un determinato problema (algebra lineare, I/O, etc.)

### Vantaggi:

- modularità
- standardizzazione
- portabilità
- efficienza
- è una blackbox

### Svantaggi:

- è una blackbox
- non sapere cosa si sta usando
- eccesso di fiducia

# Mai fidarsi ciecamente di una libreria....

## LAPACK BUG LIST

---

### Table of Contents

[bug0079 :: ILA\[SDCZ\]LR out of bound access in the DO LOOP for array A.](#)  
[bug0078 :: DBDSDC problem with orthogonality of U in certain cases](#)  
[bug0077 :: \[DS\]GESVD Minimum Worksize comments need clarification](#)  
[bug0076 :: Problem in ?\(sy/he\)tri2 when nbmax greater than n](#)  
[bug0075 :: not orthogonal: eigenvectors of a symmetric matrix with DSYEV](#)  
[bug0074 :: ?sysv and ?hesv](#)  
[bug0073 :: secondst.f and dsecnd.f](#)  
[bug0072 :: cchkhs.f and zchkhs.f](#)  
[bug0071 :: slarre and dlarre](#)  
[bug0070 :: IWORK dimension is incorrect](#)  
[bug0069 :: sytri2x is writing on other part of A](#)  
[bug0068 :: Need to add LDA in \\*sytrs2](#)  
[bug0067 :: Need to add LDA in \\*syswapr](#)  
[bug0066 :: bugs in {s,d}gejsv](#)  
[bug0065 :: workspace queries not being used](#)  
[bug0064 :: got a bug in divide and conquer \(non convergence\)](#)  
[bug0063 :: Division by 0 in \\*GESVJ & \\*GEVSJ with zero columns](#)  
[bug0062 :: Array out-of-bounds reference in xLAQR5](#)  
[bug0061 :: zgehrd.f is overflowing](#)  
[bug0060 :: Bug in dtgsy2, missing IWORK call](#)  
[bug0060 :: Typo in TESTING/EIG/dchkee.f and TESTING/EIG/zchkee.f](#)  
[bug0059 :: ZGELSD crashes on large matrices \(eq M=N=NHRS>=88\), if real workspace allocated exactly as reported on workspace query.](#)  
[bug0058 :: variable N0 in xLASQ3 is INPUT/OUTPUT. \(The comments said INPUT.\)](#)  
[bug0057 :: Non-eigenvectors from DGEEV, problem in balancing](#)  
[bug0056 :: Non-orthogonal eigenvectors returned from DSYEVR](#)  
[bug0055 :: Missing arguments descriptions of work and uplo in \[csz\]syequb](#)  
[bug0054 :: Fix comment some of the argument descriptions have the wrong type.](#)  
[bug0053 :: Fix external declaration issue in testing suite file: cebchvxx.f](#)  
[bug0052 :: Incorrect NRHS parameter in call to ?SYTRS from ?la\\_{po,sy,he}rfsx\\_extended.f.](#)  
[bug0051 :: INFO=N+2 in xDGEPS due to the wrong threshold in xDGEPS](#)

In alcuni linguaggi, come il C, usare le librerie è un'operazione naturale e necessaria, fin dalle prime righe!

```
#include<stdio.h>  
#include<math.h>
```

```
gcc myprogram.c -lm
```

```
include 'mpif.h'
```

```
mpif77 myprogram.f
```

## SP6

```

autoload/0.1/quiet          deisa/1.0(default)
----- /cineca/prod/modulefiles/base/libraries -----
PETSc/3.0.0-p7--xl--10.1(default)
PETSc/3.0.0-p7-single--xl--10.1
ParMetis/3.1.1--xl--10.1
arpack/96--xl--10.1
atlas/3.9.23--gcc--4.2.0-ibm--binary(default)
blacs/1.1--xl--10.1
boost/1.34.1--xl--10.1
cfitsio/3.21--xl--10.1
fftw/2.1.5--xl--10.1
fftw/3.2.2--xl--10.1
gc/7.1--xl--10.1
gmp/4.3.1--xl--10.1
gsl/1.9--xl--10.1(default)
guile/1.8.7--xl--10.1
hdf/4.2r4--xl--10.1(default)
hdf4/4.2r4--xl--10.1
hdf5/1.8.4_gpfs--xl--10.1
hdf5/1.8.4_par--xl--10.1
hdf5/1.8.4_ser--xl--10.1(default)
iconv/1.13--xl--10.1
icu/4.2.1--xl--10.1
jasper/1.900.1--xl--10.1
lapack/3.1.1--xl--10.1(default)
lapack/3.2.1--xl--10.1
libctl/3.1--xl--10.1
libjpeg/7--xl--10.1(default)
libxml/2.2.5.11--xl--10.1
metis/4.0--xl--10.1
mpfr/2.4.1--xl--10.1
nag/22--binary(default)
netcdf/4.0.1--xl--10.1
netcdf/4.0.1_gpfs--xl--10.1
netcdf/4.0.1_ser--xl--10.1(default)
numpy/1.3.0--xl--10.1
q5cost/1.0--xl--10.1
qt/4.5.2--xl--10.1
scalapack/1.8.0--xl--10.1
sprng/4.0--xl--10.1
szlib/2.1--xl--10.1
udunits/1.12.9--xl--10.1
udunits2/2.1.8--xl--10.1
wsmp/10.1.16--xl--10.1(default)
zlib/1.2.3--xl--10.1
zlib/1.2.5--xl--10.1(default)
----- /cineca/prod/modulefiles/base/compilers -----

```

## PLX

```

----- /cineca/prod/modulefiles/base/libraries -----
ParMetis/3.1.1--openmpi--1.3.3-gnu--4.1.2
ParMetis/3.1.1--openmpi--1.3.3--intel--11.1--binary
Qt/4.6(default)
Qt/4.7.1
cineca-libs/1.3--openmpi--1.3.3-gnu--4.1.2
cineca-libs/1.3--openmpi--1.3.3--intel--11.1--binary
curl/7.21.4--gnu--4.5.2
fftw/2.1.5--openmpi--1.3.3-gnu--4.1.2
fftw/2.1.5--openmpi--1.3.3--intel--11.1--binary
fftw/3.2.2--gnu--4.1.2
graphviz/2.26.3--gnu--4.1.2
gsl/1.14--gnu--4.1.2
gsl/1.14--intel--11.1--binary
hdf5/1.8.5_par--intel--11.1--binary
hdf5/1.8.5_ser--intel--11.1--binary
libevent/1.3b--intel--11.1--binary
lua/5.1.4--gnu--4.5.2
lzo/2.03--gnu--4.1.2
lzo/2.03--intel--11.1--binary
mkl/10.2.2--binary
netcdf/3.6.2--gnu--4.5.2
parfe/0.2--openmpi--1.3.3--intel--11.1--binary
seplib/7.0.3--intel--11.1--binary
sqlite/3.6.22--gnu--4.1.2
szip/2.1--intel--11.1--binary
trilinos/8.0.8--openmpi--1.3.3--intel--11.1--binary
zlib/1.2.5--intel--11.1--binary
----- /cineca/prod/modulefiles/base/compilers -----

```

... ma ogni libreria è personalizzabile a piacere  
(... e secondo esigenze...)



`./configure --help`



Dove andare a guardare quando serve aiuto...



**Main Index of Software Libraries**

---

[Http://www.netlib.org](http://www.netlib.org)

Repository di circa 160 librerie



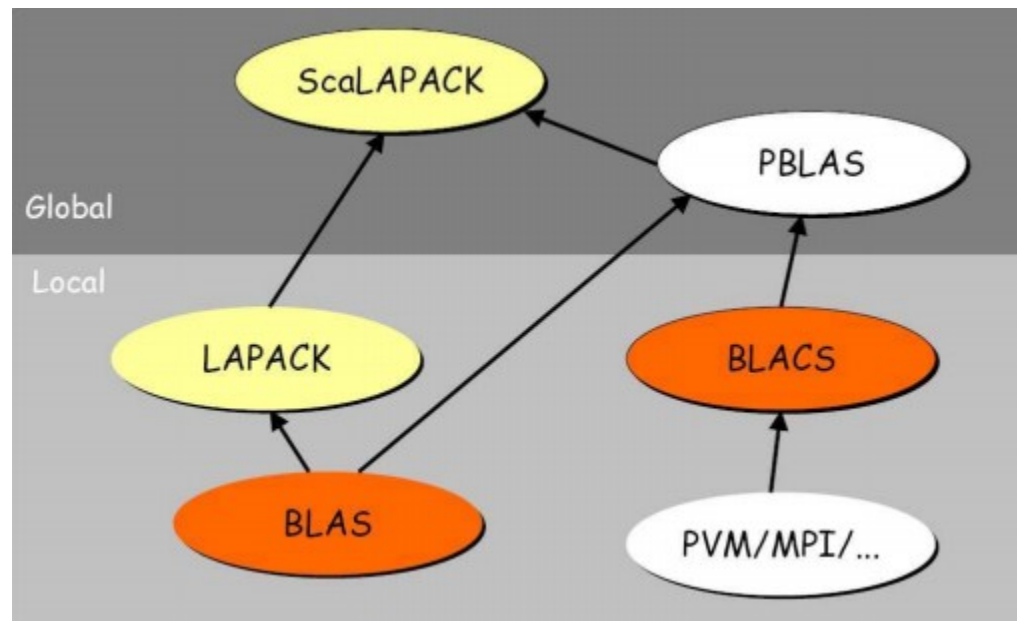
## Un case-study: **BLAS** (Basic Linear Algebra Subprogram)

- building blocks per operazioni tra vettori e matrici
- scritte in Fortran
- tre livelli:
  - Level 1: Operazioni tra vettori (scalano come  $O(n)$ )
  - Level 2: Operazioni tra vettori e matrici (scalano come  $O(n^2)$ )
  - Level 3: Operazioni fra matrici (scalano come  $O(n^3)$ )

PBLAS: Parallel Basic Linear Algebra Subprogram

BLACS: Basic Linear Algebra Communication Subprograms

Sulle BLAS si appoggiano librerie di algebra lineare di più alto livello, come le LAPACK, o di parallelizzazione come le BLACS (o loro combinazioni come ScaLAPACK)



Le BLAS sono realizzate puntando soprattutto a **robustezza e correttezza.**

- Su tutte le piattaforme HPC è disponibile una versione proprietaria che ne sfrutta al massimo l'efficienza (MKL, ESSL, ACML)
- ATLAS: librerie open-source ottimizzate per le prestazioni (multi-piattaforma)

## Le funzioni BLAS:

- Blas level 1

- \*SWAP    --> scambia vettori
  - \*COPY    --> copia vettori
  - \*SCAL    --> scala vettori
  - \*NRM2    --> norma in L2
  - \*AXPY    --> somma di vettori
  - ...

- Blas level 2

- \*GEMV    --> prodotto vettore - matrice generica
  - \*HEMV    --> prodotto vettore - matrice hermitiana
  - \*SYMV    --> prodotto vettore - matrice simmetrica
  - ...

- Blas level 3

- \*GEMM    --> prodotto matrice - matrice generica
  - \*SYMM    --> prodotto matrice - matrice simmetrica
  - ...
  - \*GE\*     --> Matrici generiche
  - \*SY\*     --> Matrici simmetriche
  - \*HE\*     --> Matrici hermitiane
  - \*TR\*     --> Matrici triangolari

## La sintassi delle BLAS:

- Prodotto matrice per matrice
  - DGEMM: prodotto in doppia precisione
  - SGEMM: prodotto in singola precisione
- Prodotto scalare vettore vettore
  - DDOT: prodotto scalare di numeri reali in doppia precisione
  - SDOT: prodotto scalare di numeri reali in singola precisione
  - ZDOT: prodotto di numeri complessi in doppia precisione
  - CDOT: prodotto di numeri complessi in singola precisione
  - ....

# Lapack and Scalapack

Linear Algebra Package and Scalable Lapack ([www.netlib.org](http://www.netlib.org))

- Matrix decomposition.
- Solution of Linear Systems.
- Eigenvalues and Eigenvectors
- Linear Least Square solutions  
for dense, banded, sparse, real and complex matrices.

# INTEL MKL (Math Kernel Library)

- Basic Linear Algebra Subprograms (BLAS):
- Sparse BLAS (basic vector operations on sparse vectors)
- Fast Fourier transform routines (with Fortran and C interfaces)
- LAPACK routines for solving systems of linear equations
- Auxiliary LAPACK routines
- Vector Mathematical Library (VML) functions for computing core mathematical functions on vector arguments (with Fortran and C interfaces)
- Vector Statistical Library (VSL) functions for generating vectors of pseudorandom numbers with different types of statistical distributions



# ACML (AMD Core Math Library)

- is a set of numerical routines tuned specifically for AMD64 platform processors (including Opteron(TM) and Athlon(TM) 64 ).
  - The routines are available via both FORTRAN 77 and C interfaces
  - BLAS - Basic Linear Algebra Subprograms (including Sparse Level 1 BLAS);
  - LAPACK - A comprehensive package of higher level linear algebra routines;
  - FFT - a set of Fast Fourier Transform routines for real and complex data.

# ESSL and PESSL

Engineering and Scientific Subroutine Library

- Level 1,2 and 3 BLAS ([www.netlib.org](http://www.netlib.org))
- Level 2 and 3 PBLAS
- Eigensystem Analysis
- Fourier Transforms (scalar & parallel)
- Random Number Generation
- Sorting, Searching, Interpolation, Quadrature

# ATLAS

## Automatically Tuned Linear Algebra Software

<http://sourceforge.net/>  
<http://math-atlas.sourceforge.net/devel/>

- E' una libreria mirata a ottenere alte prestazioni in base al processore su cui viene compilata;
- E' open source
- Contiene una propria versione delle BLAS e funzioni di algebra linear (LU decomposition, Cholesky dec.)

# Linking

*compiler ... -L/library\_directory -llibrary*

Spesso il codice e' scritto in un linguaggio e le librerie sono scritte in un altro, cosa fare?

name mangling, ovvero gli underscores

passaggio argomenti per indirizzo (Fortran)

passaggio argomenti per valore (C)

# Librerie statiche vs librerie dinamiche

- **statiche**: il codice linkato diventa parte dell'applicazione al momento della compilazione (.a)
- **dinamiche**: il codice della libreria viene caricato run-time (.so)

# Name Mangling

alcuni compilatori appendono uno o piu' underscore al nome della subroutine (tipicamente i compilatori Fortran), si possono usare i comandi:

```
nm libreria  
strings libreria
```

Sorgente Fortran

```
...  
CALL mysub(...)  
...
```

Sorgente C

```
...  
void mysub_(...)  
...
```

Nei compilatori vi sono flags per aggiungere o togliere underscore

# Richiamare una routine Fortran da C

passare tutti gli argomenti per indirizzo, aggiungere un underscore al nome le stringhe sono un problema!

```
...  
void fortsub_( int * n, double * a);  
int n;  
double a[20]; /* a e' un puntatore */  
...  
fortsub_ ( &n, a )
```

Qualche esempio...

[http://www.cae.tnitech.edu/help/programming/mixed\\_languages](http://www.cae.tnitech.edu/help/programming/mixed_languages)



In Fortran il passaggio di argomenti è sempre per riferimento...

**main.f90**

```
integer :: n  
real*8 :: a(10)  
Call fwrap( n, a)
```

**wrapper.c**

```
void fwrap_( int * n, double * a){  
    void csub( int n, double * a);  
    csub( *n, a );  
}
```

**lib.c**

```
void csub( int n, double * a){  
...  
}
```

## Un esempio di ottimizzazione delle operazioni matematiche con le librerie **MASS** (Mathematical Acceleration SubSystem)

- MASS: Scalare, versione accelerata di SQRT, SIN, COS, EXP, LOG  
non bisogna cambiare il codice
- VMASS: Vettoriale, subroutine vettoriali di SQRT, SIN, COS, EXP, LOG, divisioni ed inversioni che agiscono su array di valori  
bisogna cambiare leggermente il codice

# Libreria MASS

La versione scalare contiene:

sqrt, rsqrt, exp, log, sin, cos, tan, atan, atan2,  
sinh, cosh, tanh, dnint, x\*\*y

per utilizzarle basta specificare sulla linea di linking:

**-L /cineca/lib/mass -lmass** (sia da C che da Fortran)

La versione scalare contiene:

per visualizzare le operazioni compiute dalle mass vettoriali  
utilizzare il comando **more /cineca/lib/mass/libmassv.f**

per utilizzarle bisogna cambiare il codice, in C bisogna  
anche includere il file **"/cineca/lib/mass/massv.h"**

per fare il link con queste librerie bisogna specificare  
**-L /cineca/lib/mass -lmassv** (sia da C che da Fortran)

Table 10. MASS scalar functions

Double-precision function	Single-precision function	Description	Double-precision function prototype	Single-precision function prototype
acos	acosf	Returns the arccosine of x	double acos (double x);	float acosf (float x);
acosh	acoshf	Returns the hyperbolic arccosine of x	double acosh (double x);	float acoshf (float x);
	anint	Returns the rounded integer value of x		float anint (float x);
asin	asinf	Returns the arcsine of x	double asin (double x);	float asinf (float x);
asinh	asinhf	Returns the hyperbolic arcsine of x	double asinh (double x);	float asinhf (float x);
atan2	atan2f	Returns the arctangent of x/y	double atan2 (double x, double y);	float atan2f (float x, float y);
atan	atanf	Returns the arctangent of x	double atan (double x);	float atanf (float x);
atanh	atanhf	Returns the hyperbolic arctangent of x	double atanh (double x);	float atanhf (float x);
cbrt	cbrtf	Returns the cube root of x	double cbrt (double x);	float cbrtf (float x);
copysign	copysignf	Returns x with the sign of y	double copysign (double x, double y);	float copysignf (float x);
cos	cosf	Returns the cosine of x	double cos (double x);	float cosf (float x);
cosh	coshf	Returns the hyperbolic cosine of x	double cosh (double x);	float coshf (float x);
cosisin		Returns a complex number with the real part the cosine of x and the imaginary part the sine of x.	double _Complex cosisin (double);	
dnint		Returns the nearest integer to x (as a double)	double dnint (double x);	
erf	erff	Returns the error function of x	double erf (double x);	float erff (float x);
erfc	erfcf	Returns the complementary error function of x	double erfc (double x);	float erfcf (float x);
exp	expf	Returns the exponential function of x	double exp (double x);	float expf (float x);
expm1	expm1f	Returns (the exponential function of x) - 1	double expm1 (double x);	float expm1f (float x);
hypot	hypotf	Returns the square root of $x^2 + y^2$	double hypot (double x, double y);	float hypotf (float x, float y);
lgamma	lgammaf	Returns the natural logarithm of the absolute value of the Gamma function of x	double lgamma (double x);	float lgammaf (float x);
log	logf	Returns the natural logarithm of x	double log (double x);	float logf (float x);
log10	log10f	Returns the base 10 logarithm of x	double log10 (double x);	float log10f (float x);
log1p	log1pf	Returns the natural logarithm of (x + 1)	double log1p (double x);	float log1pf (float x);
pow	powf	Returns x raised to the power y	double pow (double x, double y);	float powf (float x);
XL Fortran version: x**y		Returns x raised to the power y	N/A	
rsqrt		Returns the reciprocal of the square root of x	double rsqrt (double x);	
sin	sinf	Returns the sine of x	double sin (double x);	float sinf (float x);
sincos		Sets *s to the sine of x and *c to the cosine of x	void sincos (double x, double* s, double* c);	
sinh	sinhf	Returns the hyperbolic sine of x	double sinh (double x);	float sinhf (float x);
sqrt		Returns the square root of x	double sqrt (double x);	
tan	tanf	Returns the tangent of x	double tan (double x);	float tanf (float x);
tanh	tanhf	Returns the hyperbolic tangent of x	double tanh (double x);	float tanhf (float x);

# Temporizzazione Codici

```
#include<stdio.h>
#include<time.h>
#include<ctype.h>
#include<sys/types.h>
#include<sys/time.h>
```

Attenzione all'underscore!

```
double cclock_()
{
```

```
    /* Restituisce il valore del CLOCK di sistema in secondi */
```

```
    struct timeval tmp;
    double sec;
    gettimeofday( &tmp, (struct timezone *)0 );
    sec = tmp.tv_sec + ((double)tmp.tv_usec)/1000000.0;
    return sec;
```

```
}
```

## program mass\_test1

```
integer, parameter :: dim = 1000000
real*8 :: sigma = 2.0d0
real*8 :: xm = 0.0d0
real*8 :: pi = 3.1425d0
real*8 :: di = 10.0d0 / DBLE( dim )
real*8 :: x, z, t1, t2
real*8, allocatable :: y(:)
integer :: i
```

```
ALLOCATE( y( dim ) )
```

```
! Calculate Gauss function
```

```
CALL cpu_time( t1 )
```

```
do i = 1, dim
  x = - 5.0d0 + 10.0d0 * (i-1) / dble(i)
  z = - ( x - xm ) ** 2 / ( 4.0 * pi * sigma )
  y(i) = exp( z )
end do
```

```
CALL cpu_time( t2 )
```

```
write(*,*) ' vector sum = ', sum(y)
write(*,*) ' tempo (secondi) = ', t2-t1
```

```
DEALLOCATE( y )
end program
```

## program mass\_test3

```
integer, parameter :: dim = 1000000
real*8 :: sigma = 2.0d0
real*8 :: xm = 0.0d0
real*8 :: pi = 3.1425d0
real*8 :: di = 10.0d0 / DBLE( dim )
real*8, allocatable :: x(:), y(:), z(:)
real*8 :: t1, t2
integer :: i
ALLOCATE( y( dim ), x( dim ), z( dim ) )
! Calculate Gauss function
call cpu_time( t1 )
do i = 1, dim
  y(i) = 10.0d0 * (i-1)
end do
do i = 1, dim
  z(i) = dble(i)
end do
call vdiv( x, y, z, dim )
do i = 1, dim
  z(i) = - ( ( x(i) - 5.0d0 ) - xm ) ** 2 / ( 4.0 * pi * sigma )
end do
call vexp( y, z, dim )
call cpu_time( t2 )
```

```
write(*,*) ' vector sum = ', sum(y)
write(*,*) ' tempo (secondi) = ', t2-t1
```

```
DEALLOCATE( y, x, z )
end program
```

È sempre vero che le librerie vettoriali sono  
più veloci di quelle scalari?



# BLAS un esempio

Moltiplicazione di matrici Reali Generiche

**DGEMM** (*transa*, *transb*, 1, *n*, *m*, *alpha*, *a*, *lda*, *b*, *ldb*, *beta*, *c*, *ldc*)

**c = alpha op( a ) \* op( b ) + beta c**  
**real\*8 a(lda,\*), b(ldb,\*), c(ldc,\*)**

$$C_{lm} = \alpha \sum_n A_{ln} B_{nm} + \beta C_{lm}$$

$$C_{lm} = \alpha \sum_n A_{ln}^T B_{nm} + \beta C_{lm}$$

$$C_{lm} = \alpha \sum_n A_{ln} B_{nm}^T + \beta C_{lm}$$

$$C_{lm} = \alpha \sum_n A_{ln}^T B_{nm}^T + \beta C_{lm}$$

# Programmi test

```
PROGRAM test_dgemm
```

```
IMPLICIT NONE
```

```
INTEGER, PARAMETER :: dim = 1000
REAL*8, ALLOCATABLE :: x(:, :), y(:, :), z(:, :)
INTEGER :: i, j, k
REAL*8 :: t1, t2
REAL*8 :: cclock
EXTERNAL :: cclock
ALLOCATE( x( dim, dim ), y( dim, dim ) )
ALLOCATE( z( dim, dim ) )
y = 1.0d0
z = 1.0d0 / DBLE( dim )
x = 0.0d0
t1 = cclock( )
do j = 1, dim
    do i = 1, dim
        do k = 1, dim
            x(i, j) = x(i, j) + y(i, k) * z(k, j)
        end do
    end do
end do
t2 = cclock()
write(*,*) ' Matrix sum = ', sum(x)
write(*,*) ' tempo (secondi) ', t2-t1
DEALLOCATE( x, y, z )
```

```
END PROGRAM
```

```
PROGRAM test_dgemm
```

```
IMPLICIT NONE
```

```
INTEGER, PARAMETER :: dim = 1000
REAL*8, ALLOCATABLE :: x(:, :), y(:, :), z(:, :)
INTEGER :: i, j, k
REAL*8 :: t1, t2
REAL*8 :: cclock
EXTERNAL :: cclock
ALLOCATE( x( dim, dim ), y( dim, dim ), z( dim, dim ) )

y = 1.0d0
z = 1.0d0 / DBLE( dim )
x = 0.0d0
t1 = cclock()

! x = matmul( y, z )
call dgemm('N', 'N', dim, dim, dim, 1.0d0, y,
c          dim, z, dim, 0.0d0, x, dim)

t2 = cclock()
write(*,*) ' Matrix sum = ', sum(x)
write(*,*) ' tempo (secondi) ', t2-t1
DEALLOCATE( x, y, z )
```

```
END PROGRAM
```

Procedimento a mano...

Linkando le BLAS....

Linkando (su SP6) le ESSL....

# Performance tuning, an example : intrinsic libraries

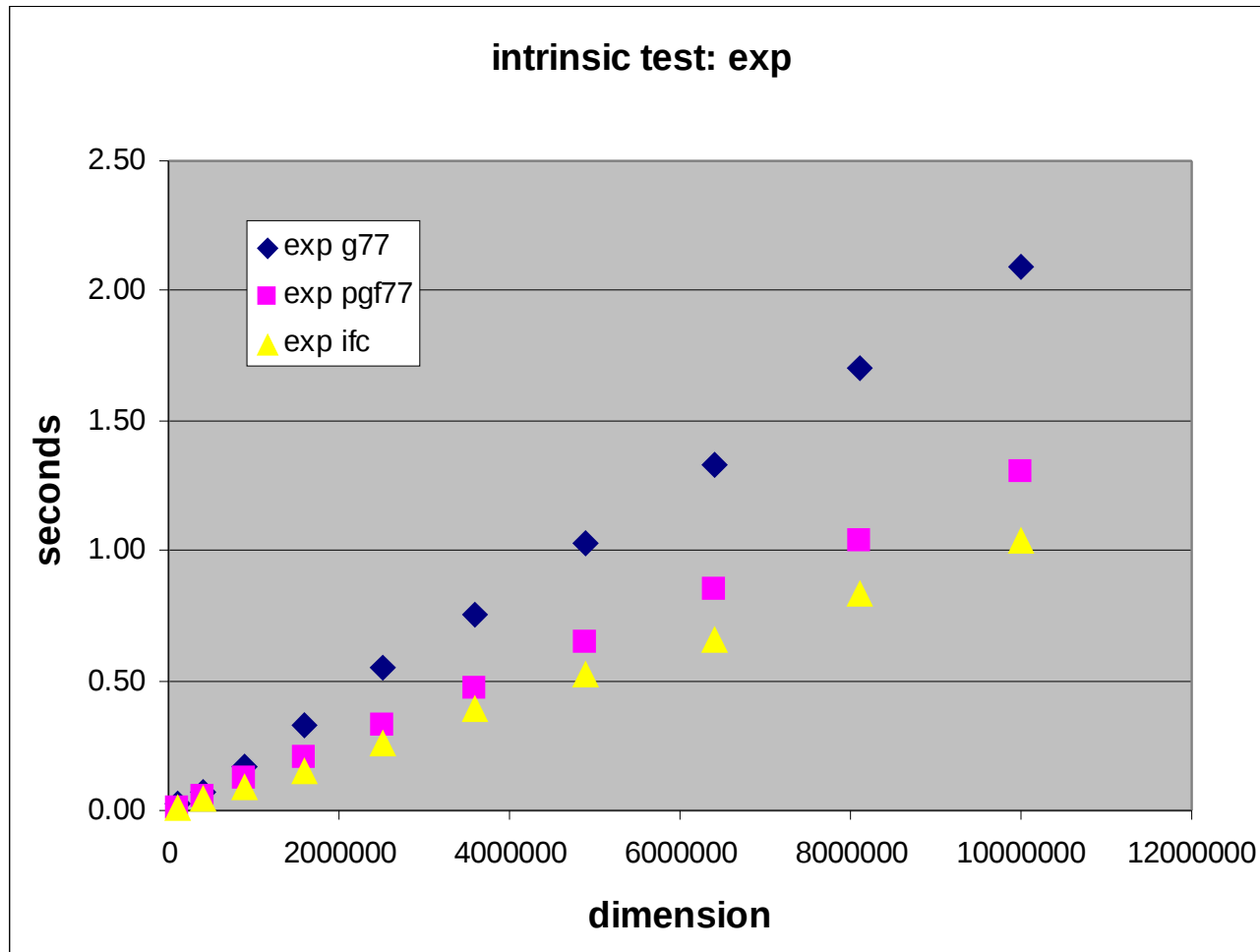
different compilers use different intrinsic libraries,  
these libraries could have a severe  
impact on the execution time

## Test Cases:

```
do i = 1, m  
  b(i) = exp( a(i) )  
end do
```

```
do i = 1, m  
  c(i) = sin( a(i) )  
end do
```

# Performance tuning: intrinsic libraries





<http://www.fftw.org>

Libreria di Fast Fourier Trasform public domain piu' completa e piu' diffusa

FFT complex to complex

FFT complex to real

Parallel FFT

Multi-thread FFT



LINKS

- What is HDF5?
- Downloads
- Documentation
- Software using HDF5
- HDF5 Users
- Sample HDF5 Files
- Acknowledgments
- Licenses

[HOME](#) > [HDF5](#)

## WHAT IS HDF5?

HDF5 is a unique technology suite that makes possible the management of extremely large and complex data collections.

### The HDF5 technology suite includes:

- A versatile data model that can represent very complex data objects and a wide variety of metadata.
- A completely portable file format with no limit on the number or size of data objects in the collection.
- A software library that runs on a range of computational platforms, from laptops to massively parallel systems, and implements a high-level API with C, C++, Fortran 90, and Java interfaces.
- A rich set of integrated performance features that allow for access time and storage space optimizations.
- Tools and applications for managing, manipulating, viewing, and analyzing the data in the collection.

The HDF5 data model, file format, API, library, and tools are open and distributed without charge.

Building on its 20-year history, The HDF Group offers personalized consulting, training, design, software development, and support services to help clients take full advantage of HDF5 capabilities in addressing their unique data management challenges.

## Testi e siti di riferimento:

- [www.netlib.org](http://www.netlib.org)
- <http://www-03.ibm.com/systems/software/essl/>
- <http://software.intel.com/en-us/articles/intel-math-kernel-library-documentation/>
- <http://www.fftw.org>
- [www.hdf5.org](http://www.hdf5.org)
- Autori vari, Numerical Recipes in Fortran90 / C / C++