

Libraries



- IBM libraries
 - ESSL (and BLAS)
 - PESSL (and BLACS)
 - MASS
- Third Party
 - LAPACK
 - ScaLAPCK
 - FFTW
 - HSL
- Conclusions
- References

- Engineering and Scientific Subroutine Library
 - provides a wide range of numerical routines
 - highly optimised for SP systems
- Includes
 - Basic linear Algebra Subprograms (BLAS)
 - Linear equation solving, eigensolvers, random number generation, numerical quadrature, fourier transforms...
- Two different versions
 - serial version
 - shared-memory parallel version

- Example

```
mpxlf90_r -qsuffix=f=f90 -o example example.f90 -lessl
```

- The -lessl library is thread-safe
- can also link with -lesslsmpl for a multi-threaded version
 - Number of threads taken from OMP_NUM_THREADS
- works with both 32- and 64-bit option (-q32 and -q64)

- ESSL and LAPACK

- contains a small subset of the LAPACK library

- Further information

- IBM ESSL documentation:

http://www-1.ibm.com/servers/eserver/pseries/library/sp_books/essl.html

- A BLAS library is also provided
 - link with `-lblas`
- Compiled version of public domain source
 - low performance
 - recommend you use ESSL, which contains BLAS routines optimised for AIX
- Example
 - matrix matrix multiply using DGEMM

<code>-lessl</code>	<code>-lblas</code>
0.88s	1.75s

- On the Cray
 - default real 64 bits - BLAS/LAPACK routines used the 'S' form e.g. SGEMM, SAXPY
- On the IBM
 - Cray 64 bit real corresponds to double precision
 - convert routines to use the 'D' form e.g. DGEMM, DAXPY
 - similarly convert complex routines from 'C' to 'Z'
- Example
 - On Cray: `call saxpy(n,1.0,a,1,b,1)`
 - On IBM: `call daxpy(n,1.0d0,a,1,b,1)`

- **Parallel ESSL**
 - scalable mathematical subroutine library
 - distributed data version (SMP version already discussed)
 - uses BLACS for communication
 - see later
 - -lpesslsmp
 - Thread safe, both 32- and 64-bit
 - Not mixed-mode, despite name
 - -lpessl
 - Not thread safe, only 32-bit.
- **Contains a subset of the ESSL routines**
 - subset of both Parallel BLAS (PBLAS) and ScaLAPACK
 - routines for fourier transforms and random number generation

- Performance is usually better than ScaLAPACK
 - not always! (e.g. PDGEMM)
 - gains are not huge (~10%)
- Example

```
mpxlf90_r -qsuffix=f=f90 -o example example.f90
-lpesslsmp -lblacssmp -lessl
```

 - BLACS is discussed in the following slide
- Further information
 - IBM ESSL documentation:

http://www-1.ibm.com/servers/eserver/pseries/library/sp_books/essl.html

- **Basic Linear Algebra Communications Subroutines**
 - message passing communications library, designed specifically for linear algebra routines.
 - just as fast as MPI but not as versatile.
- **Two versions available**
 - IBM BLACS
 - -lblacssmp: 32- and 64-bit thread-safe
 - -lblacs: 32-bit only, not thread-safe
 - Public domain BLACS
 - -lblacs: 32- and 64-bit thread safe
 - located in /usr/local/lib
 - -lblacsF77init or -lblacsCinit is also required.
- **BLACS is required if Parallel ESSL is employed.**

- The Mathematical Acceleration SubSystem
 - library of tuned mathematical intrinsic functions
- MASS currently contains
 - a scalar library
 - a general vector library
 - a vector library tuned for the POWER4
- Example

```
xlf90_r -qsuffix=f=f90 -o example example.f90 -lmass
xlc_r -o example example.c -lmass -lm
```
- See:
<http://www.rs6000.ibm.com/resource/technology/MASS>

- Math function performance*
 - (cycles per call, 1000 loop)

Function	Range	libxlf90.a	libmass.a	ratio
sqrt	A	125	72	1.74
exp	D	89	42	2.12
log	C	167	84	1.99
sin	B	54	24	2.25
sin	D	73	69	1.06
cos	B	53	23	2.30
cos	D	76	69	1.10
tan	D	178	73	2.44

**from <http://techsupport.services.ibm.com/server/mass?fetch=home.html>*

- **Compilation**

```
xlf90_r -qsuffix=f=f90 -o example example.f90 -lmassv  
xlc_r -o example example.c -lmassv -lm
```

- **Example**

```
.....
```

```
DIMENSION X(500), Y(500)
```

```
.....
```

```
CALL VEXP(Y,X,500)
```

- returns a vector Y of length 500 whose elements are $\exp(X(I))$; $I=1,500$

- **See**

<http://techsupport.services.ibm.com/server/mass?fetch=home.html>

- Set of routines for solving
 - systems of linear equations, least-squares solutions of linear systems of equations, eigenvalue problems and singular value problems
- Highly portable
- Uses BLAS routines as much as possible
 - BLAS routines often highly optimised
- Example

```
xlf90_r -qsuffix=f=f90 -o example example.f90  
-lessl -llapack
```
- LAPACK is a 32- and 64-bit thread-safe library
- See: <http://www.netlib.org/lapack/>

- Some LAPACK routines are also in ESSL
 - linking with `-lessl -llapack` will pick up the ESSL version, which will probably be faster.
- However, some of the ESSL versions of the routines have a different interface
 - picking up the ESSL version when you are expecting the LAPACK interface may give wrong results
 - if this is the case, you need to link with `-llapack -lessl` instead.

- **Distributed memory version of LAPACK**
 - subset included in PESSL
 - public domain version provided to aid porting
 - ScaLAPACK is both 32- and 64-bit and thread-safe
- **Example**

```
xlc_r -o example example.c -lessl -lblacssmp  
-lscalapack
```

-lblacssmp is required for 32- and 64-bit thread-safe codes
- **See:**
www.netlib.org/scalapack/scalapack_home.html

- **Fastest Fourier Transform in the West**
 - self-optimising Fourier transform routines
 - can be faster than those provided by ESSL/PESSL
 - Portable
- **See**
 - www.fftw.org
- **Multiple versions of FFTW are available**
 - 32- and 64-bit,
 - single- and double-precision,
 - serial, threaded and MPI

- Header, library and information files
 - `/usr/local/packages/fftw/include`
 - `/usr/local/packages/fftw/lib`
 - `/usr/local/packages/fftw/info`
- Examples
 - Double-precision serial 32-bit FFTW in serial code
 - `xlf90_r code.f`
 - I`/usr/local/packages/fftw/include`
 - L`/usr/local/packages/fftw/lib` -ldfftw
 - Single-precision, serial 64-bit FFTW in MPI code
 - `mpxlf90_r -q64 code.f`
 - I`/usr/local/packages/fftw/include`
 - L`/usr/local/packages/fftw/lib` -lsfftw**64**

- Latest version of FFTW
 - has a different interface: simply relinking won't work.
 - better performance than version 2 (?)
- Header, library and information files are in
 - `/usr/local/packages/fftw/fftw3_32` (32 bit)
 - `/usr/local/packages/fftw/fftw3_64` (64 bit)
- Link with `-lfftw3` for C or `-lfftw3f` for Fortran

- HSL (formerly Harwell Subroutine Library)
 - collection of ISO Fortran codes for large scale scientific computation
 - particularly useful for sparse equation solving
 - threadsafe
- See
 - `www2.cse.dl.ac.uk/Activity/HSL`
- To link in HSL
 - `-L/usr/local/lib -lhsl`

- Introduced IBM libraries
 - ESSL
 - PESSL
 - BLACS
 - MASS
- Summarised 3rd party libraries
 - LAPACK
 - ScaLAPACK
 - FFTW
 - HSL

- Specific library references given on the slides
 - The HPCx Service User Guide
<http://www.hpcx.ac.uk/support/documentation/>
 - Redbooks
 - The Power4 Processor Introduction and Tuning Guide (some ESSL and PESSL information)
<http://www.redbooks.ibm.com/redbooks/SG247041.html>
 - RS/6000 SP:Practical MPI Programming (some practical help on using PESSL)
<http://www.redbooks.ibm.com/redbooks/SG245380.html>
-