数学函数库的使用

李会民
hmli@ustc.edu.cn

中国科学技术大学 超级计算中心

2014年4月10日

Intel MKL

- Intel核心数学库(Math Kernel Library, MKL)，用户可以直接调用，以提高性能、加快开发。
- 安装后的路径类似下面，具有Intel 64(EM64T、AMD64、X86_64)和IA-32版本：
  - */opt/ intel /composer_xe_2013.2.146/mkl*
  - */opt/ intel /Compiler/11.1/073/mkl*
  - */opt/ intel /composerxe−2011.3.174/mkl*
  - */opt/ intel /mkl/10.0.4.023*

环境设置

设置针对em64t的MKL所需的INCLUDE、LD_LIBRARY_PATH和MANPATH等环境变量：

- bash下可在*~/.bashrc*之类文件中添加类似以下代码之一（与具体版本对应）

```
../opt/intel/composer_xe_2013.2.146/bin/compilervars.sh intel64
../opt/intel/mkl/10.0.4.023/tools/environment/mklvarsem64t.sh
../opt/intel/Compiler/11.1/073/mkl/tools/environment/mklvarsem64t.sh
../opt/intel/composerxe/mkl/bin/intel64/mklvars_intel64.sh
```

- csh下可在*~/.login*之类文件中添加以下代码之一（与具体版本对应）

```
source /opt/intel/composer_xe_2013.2.146/bin/compilervars.csh intel64
source /opt/intel/mkl/10.0.4.023/tools/environment/mklvarsem64t.csh
source /opt/intel/Compiler/11.1/073/mkl/tools/environment/mklvarsem64t.c
source /opt/intel/composerxe/mkl/bin/intel64/mklvars_intel64.csh
```

MKL主要内容

- BLAS(level 1, 2, and 3)和LAPACK线性代数程序：支持向量、向量-矩阵、矩阵-矩阵操作。
- PARDISO*直接离散算子：一种迭代稀疏矩阵解算器，支持用于求解方程的离散系统的离散BLAS(level 1, 2, and 3)程序。
- ScaLAPACK分布式的线性代数程序：含有基本线性代数通信子程序（Basic Linear Algebra Communications Subprograms, BLACS)和并行基本线性代数子程序（Parallel Basic Linear Algebra Subprograms, PBLAS）。
- 快速傅立叶变换子程序（Fast Fourier transform, FFT）：支持1、2、3维和混合基数（不局限于2的次方），并具有分布式并行的版本。
- 向量数学库（Vector Math Library, VML）：优化后的针对向量的数学操作程序。
- 向量统计库（Vector Statistical Library, VSL）：提供高性能的向量化随机数产生子(RNG)，主要针对几率分布、卷积和相关性程序、摘要统计功能。
- 数据拟合库（Data Fitting Library）：提供基于样条函数逼近能力，导数和积分，和搜索。

MKL目录内容

MKL主目录，比如/opt/intel/composer_xe_2013.2.146/mkl

主目录下的子目录	内容
bin	存储设置环境变量的脚本
bin/ia3	针对IA-32架构
bin/intel64	针对Intel 64架构
benchmarks/linpack	包含OpenMP版的LINPACK的基准程序
benchmarks/mp_linpack	包含MPI版的LINPACK的基准程序
examples	一些例子，建议用户参考学习
include	含有INCLUDE文件
include/ia32	含有针对IA-32架构的Fortran 95 .mod文件
include/intel64/lp64	含有针对Intel 64架构及LP64接口的Fortran 95 .mod文件
include/intel64/ilp64	含有针对Intel 64架构及ILP64接口的Fortran 95 .mod文件
include/fftw	FFTW2和FFTW3接口的头文件
interfaces/blas95	包含BLAS的Fortran 90封装及用于编译成库的makefile
interfaces/fftw2xc	包含2.x版FFTW(C接口)封装及用于编译成库的makefile
interfaces/fftw2xf	包含2.x版FFTW(Fortran接口)封装及用于编译成库的makefile
interfaces/fftw3xc	包含3.x版FFTW(C接口)封装及用于编译成库的makefile
interfaces/fftw3xf	包含3.x版FFTW(Fortran接口)封装及用于编译成库的makefile
interfaces/fftw2x_cdft	包含2.x版MPI FFTW(集群FFT)封装及用于编译成库的makefile
interfaces/lapack95	包含LAPACK的Fortran 90封装及用于编译成库的makefile
lib/32	包含IA-32架构的静态库和共享目标文件
lib/intel64	包含Intel 64架构的静态库和共享目标文件
tests	一些测试文件
tools/builder	包含用于生成定制动态可链接库的工具

文档在上层目录的Documentation/en_US/mkl子目录下，如：/opt/intel/composer_xe_2013.2.146/Documentation/en_US/mkl。

动态库与静态库

- 静态函数库
  - 库名一般是libxxx.a
  - 整个函数库的所有数据都会被整合进目标代码中
  - 优点：
    - 编译后的执行程序不需要外部的函数库支持
    - 执行速度快
  - 缺点：
    - 编译成的文件比较大
    - 如静态函数库改变了，那么程序必须重新编译
    - 如多个应用程序使用的话，会被装载多次，浪费内存
- 动态函数库
  - 库名一般是libxxx.so
  - 编译时没有被编译进目标代码中，执行到相关函数时才调用该函数库里的相应函数
  - 优点
    - 产生的可执行文件比较小
    - 动态函数库的改变并不影响程序，动态函数库的升级比较方便
    - 共享：多个应用程序可以使用同一个动态库，启动多个应用程序的时候，只需要将动态库加载到内存一次即可
  - 缺点：1、程序的运行环境中必须提供相应的库；2、运行速度慢

利用-mkl编译器参数

Intel Composer XE编译器支持采用-mkl[1]参数链接Intel MKL：

- -mkl或-mkl=parallel：采用标准线程（OpenMP）Intel MKL库链接
- -mkl=sequential：采用串行Intel MKL库链接
- -mkl=cluster：采用Intel MPI和串行MKL库链接
- 对Intel 64架构的系统，默认使用LP64接口链接程序

[1]是-mkl，不是-lmkl，其它编译器未必支持此-mkl选项。

使用单一动态库

- 可以通过使用Intel MKL Single Dynamic Library(SDL)来简化链接行。
- 为了使用SDL，请在链接行上添加libmkl_rt.so，如
  icc 」application .c」−lmkl_rt
- SDL使得可以在运行时选择Intel MKL的接口和线程。默认使用SDL链接时提供:
  - 对Intel 64架构的系统，使用LP64接口链接程序
  - Intel线程
- 如需要使用其它接口或改变线程性质，含使用串行版本Intel MKL等，需要使用函数或环境变量来指定选择，参见动态选择接口和线程层部分。

选择所需库进行链接

- 选择所需库进行链接，一般需要：
  - 从接口层(Interface layer)和线程层(Threading layer)各选择一个库
  - 从计算层(Computational layer)和运行时库(run-time libraries, RTL)添加仅需的库
- 链接应用程序时的对应库参见下表

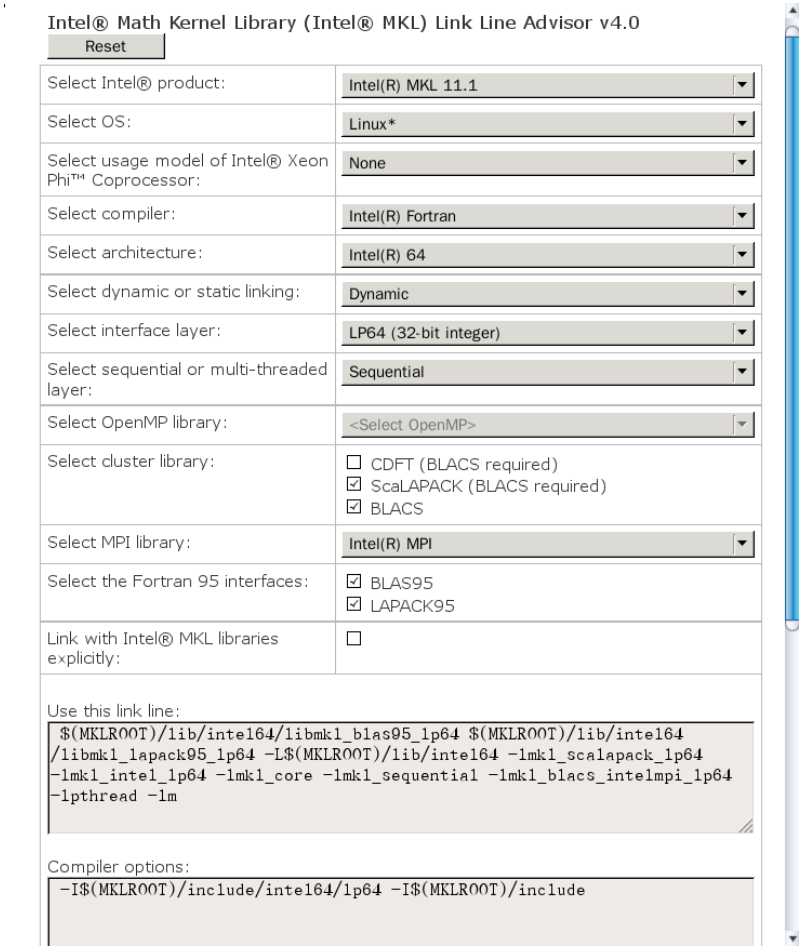
	接口层	线程层	计算层	运行库
IA-32架构，静态链接	libmkl_intel.a	libmkl_intel_thread.a	libmkl_core.a	libiomp5.so
IA-32架构，动态链接	libmkl_intel.so	libmkl_intel_thread.so	libmkl_core.so	libiomp5.so
Intel 64架构，静态链接	libmkl_intel_lp64.a	libmkl_intel_thread.a	libmkl_core.a	libiomp5.so
Intel 64架构，动态链接	libmkl_intel_lp64.so	libmkl_intel_thread.so	libmkl_core.so	libiomp5.so

- SDL会自动链接接口、线程和计算库，简化了链接处理。下表列出的是采用SDL动态链接时的Intel MKL库

	单一动态库	运行时库
IA-32和Intel 64架构	libmkl_rt.so	libiomp5.so

使用链接行顾问

- Intel提供了网页方式的链接行顾问帮用户设置所需的MKL链接参数
- 访问http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor
- 按照提示输入所需要信息即可获取链接Intel MKL时所需要的参数



使用命令行链接工具

- 使用Intel MKL的命令行链接工具可简化使用Intel MKL编译程序
- 本工具不仅可以给出所需的选项、库和环境变量，还可执行编译和生成可执行程序。
- *mkl_link_tool* 命令安装在*<mkl directory>/tools*，主要有三种模式：
  - 查询模式：返回所需的编译器参数、库或环境变量等：
    - 获取Intel MKL库：*mkl_link_tool␣−libs␣[Intel␣MKL␣Link␣Tool␣options]*
    - 获取编译参数：*mkl_link_tool␣−opts␣[Intel␣MKL␣Link␣Tool␣options]*
    - 获取编译环境变量：*mkl_link_tool␣−env␣[Intel␣MKL␣Link␣Tool␣options]*
  - 编译模式：可编译程序：
    - *mkl_link_tool␣[options]␣<compiler>␣[options2]␣file1␣[file2␣...]*
  - 交互模式：采用交互式获取所需要的参数等：
    - *mkl_link_tool␣− interactive*
- 参见：
  http://software.intel.com/en-us/articles/mkl-command-line-link-tool

在Intel 64架构上链接

- 在这些例子中：
  - MKLPATH=$MKLROOT/lib/intel64
  - MKLINCLUDE=$MKLROOT/include
- 如已设置好环境变量，那么：
  - 在所有例子中可以略去-I$MKLINCLUDE
  - 在所有动态链接的例子中可以略去-L$MKLPATH

使用LP64接口的并行Intel MKL库链接

- 静态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-Wl,--start-group $MKLPATH/libmkl_intel_lp64.a \
$MKLPATH/libmkl_intel_thread.a $MKLPATH/libmkl_core.a \
-Wl,--end-group -liomp5 -lpthread -lm
```

- 动态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5 -lpthread -lm
```

使用LP64接口的串行Intel MKL库链接

- 静态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-Wl,--start-group $MKLPATH/libmkl_intel_lp64.a \
$MKLPATH/libmkl_sequential.a $MKLPATH/libmkl_core.a \
-Wl,--end-group -lpthread -lm
```

- 动态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-lmkl_intel_lp64 -lmkl_sequential -lmkl_core -lpthread -lm
```

使用ILP64接口的并行Intel MKL库链接

- 动态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-Wl,--start-group $MKLPATH/libmkl_intel_ilp64.a \
$MKLPATH/libmkl_intel_thread.a $MKLPATH/libmkl_core.a \
-Wl,--end-group -liomp5 -lpthread -lm
```

- 动态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-lmkl_intel_ilp64 -lmkl_intel_thread -lmkl_core -liomp5 -lpthread -lm
```

使用串行或并行（调用函数或设置环境变量选择线程串行模式，并设置接口）Intel MKL库链接

- 动态链接myprog.f:

```
ifort myprog.f –lmkl_rt
```

使用Fortran 95 LAPACK接口和LP64接口的并行Intel MKL库链接

- 静态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-I$MKLINCLUDE/intel64/lp64 -lmkl_lapack95_lp64 \
-Wl,--start-group $MKLPATH/libmkl_intel_lp64.a \
$MKLPATH/libmkl_intel_thread.a $MKLPATH/libmkl_core.a \
-Wl,--end-group -liomp5 -lpthread -lm
```

使用Fortran 95 BLAS接口和LP64接口的并行Intel MKL库链接

- 静态链接myprog.f:

```
ifort myprog.f -L$MKLPATH -I$MKLINCLUDE \
-I$MKLINCLUDE/intel64/lp64 -lmkl_blas95_lp64 \
-Wl,--start-group $MKLPATH/libmkl_intel_lp64.a \
$MKLPATH/libmkl_intel_thread.a $MKLPATH/libmkl_core.a \
-Wl,--end-group -liomp5 -lpthread -lm
```

在命令行上列出所需库链接

```
<files to link>

-L<MKL path> -I<MKL include>
[-I<MKL include>/{ia32|intel64|{ilp64|lp64}}]
[-lmkl_blas{95|95_ilp64|95_lp64}]
[-lmkl_lapack{95|95_ilp64|95_lp64}]
[ <cluster components> ]
-lmkl_{intel|intel_ilp64|intel_lp64|intel_sp2dp|gf|gf_ilp64|gf_lp64}
-lmkl_{intel_thread|gnu_thread|pgi_thread|sequential}
-lmkl_core
-liomp5 [-lpthread] [-lm] [-ldl]
```

- *[]*内的表示可选，*|*表示其中之一、*{}*表示含有。
- 在静态连接时，在分组符号（如，*-Wl,--start-group␣$MKLPATH/libmkl_cdft_core.a␣$MKLPATH/libmkl_blacs_intelmpi_ilp64 .a␣$MKLPATH/libmkl_intel_ilp64.a␣$MKLPATH/libmkl_intel_thread.a $MKLPATH/libmkl_core.a␣-Wl,--end-group*）封装集群组件、接口、线程和计算库。

动态选择接口和线程层链接

SDL接口使得用户可以动态选择Intel MKL的接口和线程层。

设置接口层

- 可用的接口与系统架构有关，对于Intel 64架构，可使用LP64和ILP64接口
- 在运行时设置接口，可调用mkl_set_interface_layer函数或设置*MKL_INTERFACE_LAYER*环境变量
- 可用的接口层的值

接口层	*MKL_INTERFACE_LAYER*的值	mkl_set_interface_layer的参数值
LP64	LP64	MKL_INTERFACE_LP64
ILP64	ILP64	MKL_INTERFACE_ILP64

- 如果调用了mkl_set_interface_layer函数，那么环境变量*MKL_INTERFACE_LAYER*的值将被忽略
- 默认使用LP64接口

设置线程层

- 在运行时设置线程层，可以调用mkl_set_threading_layer函数或者设置环境变量*MKL_THREADING_LAYER*
- 可用的线程层的值

线程层	*MKL_INTERFACE_LAYER*的值	mkl_set_interface_layer的参数值
Intel线程	INTEL	MKL_THREADING_INTEL
串行线程	SEQUENTIAL	MKL_THREADING_SEQUENTIAL
GNU线程	GNU	MKL_THREADING_GNU
PGI线程	PGI	MKL_THREADING_PGI

- 如果调用了mkl_set_threading_layer函数，那么环境变量MKL_THREADING_LAYER的值被忽略
- 默认使用Intel线程

使用ILP64接口 vs. LP64接口

- Intel MKL ILP64库采用64-bit整数（索引超过含有$2^{31}-1$个元素的大数组时使用），而LP64库采用32-bit整数索引数组
- LP64和ILP64接口在接口层实现，分别采用下面接口层链接：
  - 静态链接：libmkl_intel_lp64.a或libmkl_intel_ilp64.a
  - 动态链接：libmkl_intel_lp64.so或libmkl_intel_ilp64.so
- ILP64接口提供以下功能：
  - 支持大数据数组（具有超过$2^{31}-1$个元素）
  - 添加-i8编译器参数编译Fortran程序
- LP64接口提供与以前Intel MKL版本的兼容，因为LP64对于仅提供一种接口的版本低于9.1的Intel MKL来说是一个新名字
- 如果用户的应用采用Intel MKL计算大数据数组或此库也许在将来会用到时请选择使用ILP64接口
- Intel MKL提供的ILP64和LP64头文件路径是相同的

采用LP64/ILP64编译

- 采用ILP64和LP64接口进行编译：
  - Fortran：
    - ILP64： *ifort 」–i8」–I<mkl」directory>/include」 ...*
    - LP64： *ifort 」–I<mkl」directory>/include」 ...*
  - C/C++：
    - ILP64： *icc 」–DMKL_ILP64」–I<mkl」directory>/include」...*
    - LP64： *icc 」–I<mkl」directory>/include」 ...*
- 注意： 采用-i8或-DMKL_ILP64选项链接LP64接口库时也许将会产生预想不到的错误。

编写代码

- 如果不使用ILP64接口，无需修改代码。
- 为了移植或者新写代码使用ILP64接口，需要使用正确的Intel MKL函数和子程序的参数类型：

整数类型	Fortran	C/C++
32-bit整数	INTEGER*4或INTEGER(KIND=4)	int
针对ILP64/ LP64的通用整数（ILP64使用64-bit，其余32-bit）	INTEGER，不指明KIND	MKL_INT
针对ILP64/ LP64的通用整数（64-bit整数）	INTEGER*8或INTEGER(KIND=8)	MKL_INT64
针对ILP64/LP64的FFT接口	INTEGER，不指明KIND	MKL_LONG

局限性

所有Intel MKL函数都支持ILP64编程，但是针对Intel MKL的FFTW接口：

- FFTW 2.x封装不支持ILP64
- FFTW 3.2封装通过专用功能函数plan_guru64支持ILP64

使用Fortran 95接口库

- libmkl_blas95*.a和libmkl_lapack95*.a库：
  - 分别含有BLAS和LAPACK所需的Fortran 95接口
  - 并且是与编译器无关
- 在Intel MKL包中：
  - 已经为Intel Fortran编译器预编译了
  - 如果使用其它编译器，请在使用前先编译

使用线程库链接 I

- 串行库模式
  - 采用Intel MKL串行（非线程化）模式时，Intel MKL运行非线程化代码。它是线程安全的（除了LAPACK已过时的子程序?lacon），即可以在用户程序的OpenMP代码部分使用。串行模式不要求与OpenMP运行时库的兼容，环境变量*OMP_NUM_THREADS*或其Intel MKL等价变量对其也无影响。
  - 只有在不需要使用Intel MKL线程时才应使用串行模式。当使用一些非Intel编译器线程化程序或在需要非线程化库（比如使用MPI的一些情况时）的情形使用Intel MKL时，串行模式也许有用。为了使用串行模式，请选择*sequential.*库。
  - 对于串行模式，由于*sequential.*依赖于pthread，请在链接行添加POSIX线程库(pthread)。
- 选择线程库层

  一些Intel MKL支持的编译器使用OpenMP线程技术。Intel MKL支持这些编译器提供OpenMP技术实现，为了使用这些支持，需要采用正确的线程层和编译器支持运行库进行链接。

使用线程库链接 II

- 线程层：每个Intel MKL线程库包含针对同样的代码采用不同编译器（Intel、GNU和PGI编译器）分别编译的库
- 运行时库：
  - 此层包含Intel编译器兼容的OpenMP运行时库libiomp
  - 在Intel编译器之外，libiomp提供在Linux操作系统上对更多线程编译器的支持
  - 采用GNU编译器线程化的程序可以安全地采用intel MKL和libiomp链接
- 不同情形使用Intel MKL时选择线程库和运行时库（仅静态链接情形）

编译器	应用是否线程化	线程层	推荐的运行时库	备注
Intel	无所谓	libmkl_intel_thread.a	libiomp5.so	
PGI	Yes	libmkl_pgi_thread.a 或libmkl_sequential.a	由PGI*提供	使 用libmkl_sequential.a从Intel MKL调用中去除线程化
PGI	No	libmkl_intel_thread.a	libiomp5.so	
PGI	No	libmkl_pgi_thread.a	由PGI*提供	
PGI	No	libmkl_sequential.a	None	
GNU	Yes	libmkl_gnu_thread.a	libiomp5.so或GNU OpenMP运行时库	libiomp5提供监控缩放性能
GNU	Yes	libmkl_sequential.a	None	
GNU	No	libmkl_intel_thread.a	libiomp5.so	
other	Yes	libmkl_sequential.a	None	
other	No	libmkl_intel_thread.a	libiomp5.so	

使用计算库链接

- 如不使用Intel MKL集群软件在链接应用程序时只需要一个计算库即可，其依赖于链接方式：
  - 静态链接：libmkl_core.a
  - 动态链接：libmkl_core.so
- 采用Intel MKL集群软件的计算库
  - ScaLAPACK和集群Fourier变换函数(Cluster FFTs)要求更多的计算库，其也许依赖于架构
  - 针对Intel 64架构的使用ScaLAPACK或集群FFTs的计算库：

函数域	静态链接	动态链接
ScaLAPACK，LP64接口	libmkl_scalapack_lp64.a和 libmkl_core.a	libmkl_scalapack_lp64.so和 libmkl_core.so
ScaLAPACK，ILP64接口	libmkl_scalapack_ilp64.a和 libmkl_core.a	libmkl_scalapack_ilp64.so和 libmkl_core.so
集群FFTs	libmkl_cdft_core.a和 libmkl_core.a	libmkl_cdft_core.so和 libmkl_core.so

- 针对IA-32架构的使用 ScaLAPACK或集群FFTs的计算库：

函数域	静态链接	动态链接
ScaLAPACK	libmkl_scalapack_core.a和libmkl_core.a	libmkl_scalapack_core.so和libmkl_core.so
集群FFTs	libmkl_cdft_core.a和libmkl_core.a	libmkl_cdft_core.so和libmkl_core.so

- 对于ScaLAPACK和集群FFTs，当在MPI程序中使用时，还需要添加BLACS库

使用编译器运行库链接

- 在静态链接其它库时，也可动态链接libiomp5、兼容的OpenMP运行时库
- 静态链接libiomp5也许会存在问题：
  - 因为由于操作环境或应用越复杂，将会包含更多多余的库的复本
  - 不仅会导致性能问题，甚至导致不正确的结果
- 动态链接libiomp5时，需确保*LD_LIBRARY_PATH*环境变量设置正确

使用系统库链接

- 使用Intel MKL的FFT、Trigonometric Transforn或Poisson、Laplace和Helmholtz求解程序时，需要通过在链接行添加-lm参数链接数学支持系统库
- 在Linux系统上，由于多线程libiomp5库依赖于原生的pthread库，因此，在任何时候，libiomp5要求在链接行随后添加-lpthread参数（列出的库的顺序非常重要）

ACML： AMD核心数学库

AMD Core Math Library(ACML)是一些数值函数的组合，并且特别针对AMD64平台处理器（如Opteron）等做了优化。ACML函数库具有FORTRAN 77和C语言接口，主要包含：

- BLAS - 基本线性代数子系统库
- LAPACK - 线性代数库
- FFT - 傅立叶变换程序
- RNG - 随机数发生器和统计分布函数

网址：http://developer.amd.com/tools-and-sdks/cpu-development/amd-core-math-library-acml/

Rogue Wave IMSL：数学与统计数值函数库

- IMSL是Rogue Wave公司的一套完整的数学与统计数值函数库，能够让使用者嵌入至他们的应用系统中
- IMSL提供高效能的计算能力并提供所有专家所需要开发与建置的精密数学分析应用程序。这些程序库能够让用户直接使用已经写好的数学与统计算法，而免去自我撰写程序的麻烦，并能够让用户轻易的嵌入至用户所使用的C与Fortran语言程序中
- IMSL与MKL相比主要多了积分、随机数及统计等方面的函数
- 网址：www.roguewave.com/products/imsl-numerical-libraries.aspx

Netlib: 数学软件、论文及数据库的宝库

- 免费开源
- http://www.netlib.org

Netlib Libraries:

a
access
aicm
alliant
amos
ampl
anl-reports
apollo
arpack
atlas
benchmark
bib
bibnet
bihar
blacs
blas
blast
bmp
c
c++
cephes
chammp
cheney-kincaid
clapack
commercial
confdb
conformal
contin
control
crc
cumulvs
ddsv
dierckx
diffpack
domino
eispack
elefunt
env
etemplates
f2c
fdlibm
fftpack
fishpack
fitpack
floppy
fmm
fn
fortran
fortran-m
fp
gcv
gmat
gnu
go
graphics
harwell
hence
hompack
hpf
hypercube
ieeecss
ijsa
image
intercom
itpack
jakef
java
kincaid-cheney
la-net
lanczos
lanz
lapack
lapack++
lapack3e
lapack90
laso
lawson-hanson
linalg
linpack
list
lp
lyapack
machines
magic
maspar
mds
microscope
minpack
misc
mpfun
mpi
mpicl
na-digest-html
napack
netsolve
news
numeralgo
ode
odepack
odrpack
opt
p4
paragraph
paranoia
parkbench
parmacs
pascal
pdes
performance
photo
picl
pltmg
poly2
polyhedra
popi
port
posix
pppack
presto
problem-set
pvm3
quadpack
random
research
rib
scalapack
sched
scilib
seispack
sequent
sfmm
slap
slatec
sminpack
sodepack
sparse
sparse-blas
sparspak
specfun
spin
srwm
stoeplitz
stringsearch
svdpack
templates
tennessee
textbook
toeplitz
toms
tomspdf
transform
typesetting
uncon
vanhuffel
vfftpack
vfnlib
voronoi
xblas
xmagic
xmetlib
y12m

GotoBLAS2、Open BLAS及ATLAS

- GotoBLAS2性能非常高的一种BLAS实现，Open BLAS是其后续版本
  - GotoBLAS2：https://www.tacc.utexas.edu/tacc-projects/gotoblas2/
  - Open BLAS：http://xianyi.github.io/OpenBLAS/
- ATLAS(Automatically Tuned Linear Algebra Software)：
  - 性能非常高另一种BLAS实现
  - 网址：http://math-atlas.sourceforge.net/

联系信息

- 中国科大超算中心：
  - 电话：0551-63602248
  - 信箱：sccadmin@ustc.edu.cn
  - 主页：http://scc.ustc.edu.cn
  - 办公室：中国科大东区新图书馆一楼东侧126室
- 李会民：
  - 电话：0551-63600316
  - 信箱：hmli@ustc.edu.cn
  - 主页：http://hmli.ustc.edu.cn