



地球与空间科学系

DEPARTMENT OF EARTH AND SPACE SCIENCES

地空系并行集群使用说明

TCHE LIU* GONGHENG ZHANG†

2021 年 12 月 21 日

Version: 0.7

*tcheliu@mail.ustc.edu.cn

†11749276@mail.sustc.edu.cn

目录

1	注意事项	3
2	LSF 作业调度系统	3
2.1	集群队列设置	3
2.2	查询系统运行情况	4
2.3	LSF 作业提交	4
2.3.1	作业提交脚本的书写	4
2.3.2	依赖作业提交	5
2.4	LSF 作业管理	6
2.4.1	LSF 作业状态	6
2.4.2	LSF 作业调度	6
2.4.3	动态优先级策略	7
2.5	参考阅读	8
3	Environment Modules 的使用	8
3.1	系统软件环境	8
3.1.1	系统软件安装目录	8
3.1.2	编译器及解释器	8
3.1.3	运行库	9
3.1.4	工具	10
3.2	module 的使用	11
3.2.1	搜索路径 MODULEPATH	11
3.2.2	自定义 modulefile 文件	11
3.2.3	module 模块命令	12
3.2.4	参考阅读	12
4	其他服务	13
4.1	公共数据	13
4.1.1	SRTM-DEM 全球地形数据	13
4.1.2	earth_relief 全球地形起伏数据	13
4.1.3	CENC 2013-2016 连续波形数据	13
4.1.4	美国南加州部分台阵连续波形数据	14
5	常见问题解决	15
5.1	作业运行问题	15

一、注意事项

- (1) 严禁在任何节点直接运行并行程序，所有多核任务必须经由作业系统提交运行；
- (2) 提交作业时务必注意节点的内存限制，切勿提交超负荷任务；
- (3) 提交作业时烦请注意队列的提交限制，以防提交无效作业；
- (4) 用户 HOME 目录所在磁盘 share 为全节点共享，该磁盘所有用户的用户硬限制配额为 1.5 TB，当用户的总数据量超过该限制时，该用户将无法创建新文件；
- (5) 请注意及时备份集群上的重要数据，本集群不保证用户数据的安全性；
- (6) 为规范使用，仅允许用户在其作业运行期间，ssh 登录到相应的计算节点查看作业运行情况，其他时间已设置为禁止登录；
- (7) 本集群虽安装有 matlab 等绘图软件，但不推荐在集群上直接成图，建议将数据拷贝到本地机器上，再利用本地机器作图。

二、LSF 作业调度系统

2.1 集群队列设置

表 1: 集群队列设置

节点	队列	核数	内存	作业时长限制	作业核数限制	优先级	交互式提交
c002-c005	ser	28	100 G	120:00	5 核以下	40	允许
c004-c028	short	28	100 G	48:00	420 核以下	40	允许
c029-c042	large	24	170 G	36:00	96 核以上	30	禁止
c043-c047	long	32	170 G	168:00	无限制	20	禁止
s001-s002	smp	72	480 G	24:00	无限制	20	禁止

注意：

- 节点 **c001** 为专用节点，供用户运行 matlab 等占用资源较多的大型软件，无需提交作业可直接运行单核的简单数据处理或绘图程序；**禁止在登录节点 mn01 上运行此类软件。**
- 队列 **ser** 为 **串行队列**，仅允许向该队列提交单核串行任务，任何并行作业都会被强制中止运行；其它队列均为 **并行队列**，原则上不允许提交串行任务，影响到并行作业运行的串行作业会被强制中止运行。

- 队列 `short` 限制单用户最大作业运行核数为 420，即在该队列某一用户正在运行的所有作业总核数不超过 420，超过该限制的更多作业将由系统自动挂起排队，直至满足总核数要求后再由系统自动派发运行；其他队列无此限制。
- 在设置程序使用每节点核数时，尽量设置为节点总核数的整除数。

2.2 查询系统运行情况

通过命令 `lnodes`，可查看系统各节点连接情况。

通过命令 `lcores [队列名]`，可查看系统中某队列各节点计算核心当前使用情况。

通过命令 `lshosts`，可查看系统各节点硬件配置情况。

通过命令 `lsload`，可查看系统各节点当前负载。

通过命令 `lsmon`，可动态查看系统各节点当前负载。

通过命令 `bhosts`，可查看系统各节点作业运行情况。

通过命令 `bqueues`，可查看系统中各队列作业运行情况。

通过命令 `busers all`，可按用户查看所有用户的作业运行整体情况。

通过命令 `bjobs -u all`，可按作业查看所有用户的作业运行详细情况。

通过命令 `bhist -t -T <起始时刻, 终止时刻>`，可按时间顺序查看某一时间段内作业系统的调度历史。

通过命令 `mmlsquota --block-size=auto`，可查看用户数据占用磁盘空间大小及磁盘配额情况。

2.3 LSF 作业提交

2.3.1 作业提交脚本的书写

```
#!/bin/bash
#BSUB -J MPIJob          ### set the job name
#BSUB -q short            ### specify queue
#BSUB -n 40               ### ask for number of cores (default:1)
5 #BSUB -R "span[ptile=20]" ### ask for 20 cores per node
#BSUB -R "hname!=c004"    ### exclude the host c004
#BSUB -W 10:00            ### set walltime limit: hh:mm
#BSUB -oo std_%J.out      ### specify the output and error file. %J is the job-id
#BSUB -eo std_%J.err      ### -o and -e mean append, -oo and -eo mean overwrite
10
# here follow the commands you want to execute

# load the necessary modules
# NOTE: this is just an example, check with the available modules
15 module load intel/2018.4
module load mpi/intel/2018.4

### This uses the LSB_DJOB_NUMPROC to assign all the cores reserved
### This is a very basic syntax.
20 mpirun -np $LSB_DJOB_NUMPROC ./Your_MPI_Program
```

Listing 1: MPI 并行作业提交脚本示例

```

#!/bin/bash
#BSUB -J OpenMPJob          ### set the job name
#BSUB -q short              ### specify queue
#BSUB -n 20                 ### ask for number of cores (default: 1)
5 #BSUB -R "span[hosts=1]"   ### specify that the cores MUST BE on a single host!
#BSUB -m "c001"            ### specify the host that the job will run on
#BSUB -W 10:00              ### set walltime limit: hh:mm
#BSUB -o std_%J.out         ### specify the output and error file. %J is the job-id
#BSUB -e std_%J.err         ### -o and -e mean append, -oo and -eo mean overwrite
10
# set OMP_NUM_THREADS and export!
export OMP_NUM_THREADS=$LSB_DJOB_NUMPROC

# run your program
15 ./Your_OpenMP_Program [options]

```

Listing 2: OpenMP 并行作业提交脚本示例

在书写作业提交脚本时，LSF 作业系统主要的解析制导指令有：

- (1) `#BSUB -J <作业名>`：指定作业名；
- (2) `#BSUB -q <队列名>`：指定排队队列，此项不指定则调用默认队列；
- (3) `#BSUB -n <申请总核数>`：指定总核数，此项不指定则总核数默认为 1；
- (4) `#BSUB -R "<描述信息>"`：指定资源限制，这里的描述信息可以为：
`span[ptile=<每个节点的核数>]` 指定每节点使用核数、
`span[hosts=1]` 指定使用一个节点、
`select[hname!='c001']` 强制不使用 c001 节点、
`rusage[mem=<每个节点的内存限制>]` 指定每核可用内存最低限制、
 等内容；
- (5) `#BSUB -m "<节点列表>"`：指定作业运行节点，多个节点以空格分隔；
- (6) `#BSUB -W <挂钟时间>`：指定作业最长运行时间，时间格式为 hh:mm；
- (7) `#BSUB -o <标准输出文件名>`：指定标准输出文件；
- (8) `#BSUB -e <错误输出文件名>`：指定标准错误输出文件；

另外注意，在编译程序过程中若使用了特别版本的编译器或运行库，需在作业提交脚本中通过命令 `module load <模块名>` 加载这些编译器或运行库环境模块。

示例脚本见集群上 `/share/system/reference/` 目录下。

2.3.2 依赖作业提交

通过在作业提交脚本头部中加入 `#BSUB -w "<复合作业状态>(<作业号>)"` 的内容，可以实现相互依赖的多个作业任务的提交。当指定作业达到指定状态时，当前作业才会被派发。

这里的复合作业状态可以设置为：

- (1) done: 基本作业状态 DONE;
- (2) ended: 基本作业状态 EXIT 或 DONE;
- (3) exit: 基本作业状态 EXIT, 可通过 `exit(<作业号>,[操作符]<退出码>)` 指定特殊的退出码, 这里的操作符可以设为 >、>=、<、<=、== 或 !=;
- (4) started: 基本作业状态 DONE、EXIT、USUSP、SSUSP 或 RUN。

基本作业状态参见下文内容。

通过命令 `bjdepinfo <作业号>`, 可查看已提交作业的依赖关系。

2.4 LSF 作业管理

2.4.1 LSF 作业状态

LSF 调度系统的基本作业状态有:

- (1) PEND: 在系统中排队, 等待派发;
- (2) RUN: 已经被派发, 在运行中;
- (3) DONE: 正常运行完毕, 程序退出码为 0;
- (4) PSUSP: 被用户自己或管理员在派发前挂起;
- (5) USUSP: 被用户自己或管理员在派发后挂起;
- (6) SSUSP: 被 LSF 系统在派发后挂起;
- (7) EXIT: 被异常中止, 或程序退出码非 0。

被用户自己或管理员挂起的作业不会被系统自动派发; 被系统挂起的作业待满足运行条件后系统会自动派发。

作业被系统挂起的原因可能有: 该作业依赖于其他待完成作业、该作业在队列中的优先级较低、没有足够的计算资源等。

2.4.2 LSF 作业调度

参照前述示例正确书写作业提交脚本 `job.lsf` 后, 即可通过命令 `bsub <job.lsf>` 提交作业。正确提交作业后, 该命令会返回系统分配的作业号。区别于 PBS 作业系统, 注意这里的“<”是必需的。

作业提交后, 通过命令 `bjobs -l [作业号]`, 可以查看作业状态。

作业提交后, 通过命令 `bstop <作业号>`, 用户可以将自己的正在排队 (PEND 状态) 的作业挂起 (PSUSP 状态); 此时, 可以通过命令 `bresume <作业号>`, 将挂起 (PSUSP 状态) 的作业重新参与系统排队 (PEND 状态), 等待系统派发。

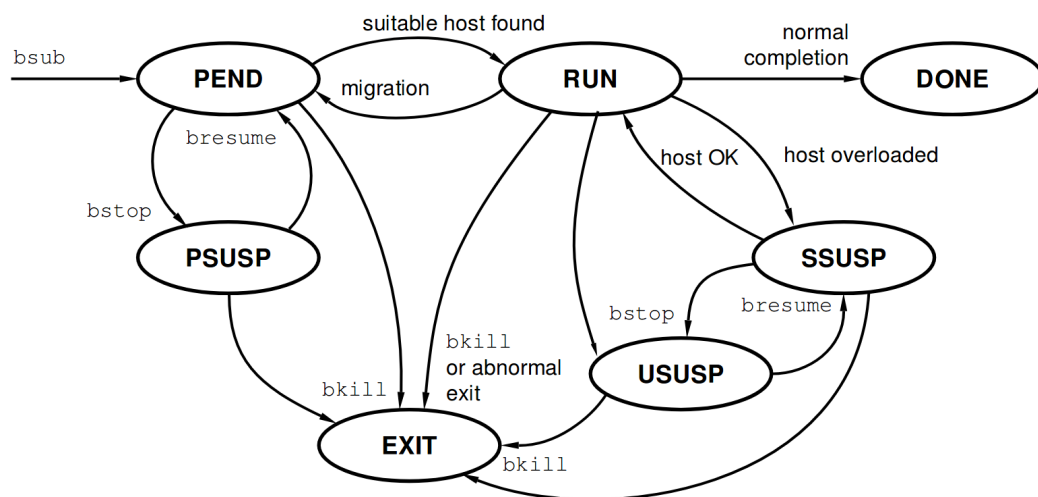


图 1: LSF 基本作业状态

作业开始运行后，同样可以通过 `bstop` 命令，用户可以将自己的正在运行（RUN 状态）的作业挂起（USUSP 状态）；同样地，通过 `bresume` 命令，再次将挂起（USUSP 状态）的作业重新参与系统排队（SSUSP 状态），等待系统派发。

通过命令 `bkill <作业号>`，用户可以将自己待运行或正在运行的作业取消（EXIT 状态）。

通过命令 `bkill -s <系统消息代码> <作业号>`，用户可以向自己的正在运行的作业发送系统消息。

通过命令 `bhist -l <作业号>`，用户可以查看指定作业的系统调度历史。

通过命令 `bpeek [作业号]`，用户可以查看自己的未完成作业的标准输出和错误输出。

在作业完成后，LSF 系统会保留作业记录 1 小时，此时间段内，还可以通过作业号查看作业的运行记录。

2.4.3 动态优先级策略

LSF 作业系统采用队列用户 **动态优先级** 策略，即系统会根据用户作业的作业槽数、运行时间和累积核时等资源占用情况，动态地实时调整用户的队列优先级。

用户在某一队列占用的系统资源越多，则在该队列的优先级越低。系统会优先派发高优先级用户提交的作业。

每个队列单独进行统计计算，同一用户在不同队列上拥有不同的动态优先级。通过命令 `bqueues -l [队列名]`，用户可以查看所有使用过该队列的用户在该队列的动态优先级。

目前，系统收集用户资源消耗历史的时间长度设置为 5 小时，即当前系统会根据用户 5 小时内的资源消耗情况来调整用户的动态优先级。

2.5 参考阅读

LSF 用户手册: http://hpc.sustech.edu.cn/ref/lsf_users_guide_v10.1.pdf

太乙用户手册: http://hpc.sustech.edu.cn/ref/TaiYi_User_Manual_v0.1.pdf

IBM Spectrum LSF V10.1 documentation: https://www.ibm.com/support/knowledgecenter/en/SSWRJV_10.1.0/lfs_welcome/lfs_welcome.html

LSF Batch User's Guide: <https://ls11-www.cs.tu-dortmund.de/people/hermes/manuals/LSF/users.pdf>

三、Environment Modules 的使用

3.1 系统软件环境

3.1.1 系统软件安装目录

系统软件的安装目录主要有:

/share/apps: 安装了 python 和 java 的解释器;

/share/base: 安装了大部分的运行库和一部分工具;

/share/intel: 安装了 intel 编译器及附带工具;

/share/tools: 安装了一些工具。

3.1.2 编译器及解释器

系统安装了多个版本的 C 和 Fortran 语言编译器:

系统默认的 gcc/4.8.5 (系统标准安装目录)、

gcc/8.2.0 (/share/base/gcc/8.2.0 目录)、

gcc/10.2.0 (/share/base/gcc/10.2.0 目录)、

intel/2017.8 (/share/intel/2017u8 目录)、

intel/2018.3 (/share/intel/2018u3 目录)、

intel/2018.4 (/share/intel/2018u4 目录)、

intel/2019.5 (/share/intel/2019u5 目录)、

intel/oneapi-2021.1 (/share/intel/oneapi-2021.1 目录)。

系统安装了多个版本的 Python 语言解释器:

python/2.7.15 (/share/apps/python/2.7.15 目录)、

python/3.7.0 (/share/apps/python/3.7.0 目录)、

python/anaconda2/5.2.0 (/share/apps/anaconda2/5.2.0 目录)、

python/anaconda3/5.2.0 (/share/apps/anaconda3/5.2.0 目录)、

python/intelpython2/2018.3.039 (/share/apps/ipython/2018.3.039/intelpython2 目录)、

python/intelpython2/2019.9.047 (/share/apps/ipython/2019.9.047/intelpython2 目录)、

python/intelpython3/2018.3.039 (/share/apps/ipython/2018.3.039/intelpython3 目录)、
python/intelpython3/2019.9.047 (/share/apps/ipython/2019.9.047/intelpython3 目录)。

系统安装了两个版本的 matlab 语言解释器：
matlab/R2016b (/share/apps/matlab/R2016b 目录)、
matlab/R2018b (/share/apps/matlab/Matlab 目录)。

系统安装了一个版本的 go 语言编译器：
go/1.11.1 (/share/base/go/1.11.1 目录)。

系统安装了两个版本的 java 语言解释器：
java/1.8.0_181 (/share/apps/java/1.8.0_181 目录)、
java/10.0.2 (/share/apps/java/10.0.2 目录)。

3.1.3 运行库

系统安装的运行库大部分都同时有 gcc/4.8.5 和 intel/2018.4 两个编译器的对应版本。

系统安装的 MPI 并行库有：
mpi/intel (在各自对应版本的 intel 编译器安装目录下)、
mpi/mpich (/share/base/mpich 目录)、
mpi/openmpi (系统标准安装目录或 /share/base/openmpi 目录)。

系统安装的其他第三方运行时函数库主要有：
zlib (/share/base/zlib 目录)、
fftw (/share/base/fftw 目录)、
hdf5 (/share/base/hdf5 目录)、
netcdf-c (/share/base/netcdf-c 目录)、
netcdf-fortran (/share/base/netcdf-fortran 目录)、
netcdf-cxx (/share/base/netcdf-cxx 目录)、
BLAS (/share/base/netlib/BLAS 目录)、
LAPACK (/share/base/netlib/LAPACK 目录)、
petsc (/share/base/petsc 目录)、
SuperLU (/share/base/SuperLU 目录)、
SuiteSparse (/share/base/SuiteSparse 目录)、
eigen (/share/base/eigen 目录)、
ICU (/share/base/ICU 目录)、
boost (/share/base/boost 目录)。

3.1.4 工具

所有编译安装的工具软件都是调用 gcc/4.8.5 编译器完成安装的。

系统安装的专业工具有：

SAC/101.6a (/share/tools/SAC/101.6a 目录)、
mseed2sac/2.2 (/share/tools/mseed2sac/2.2 目录)。

系统安装的绘图工具有：

GMT/4.5.18 (/share/tools/GMT/4.5.18 目录)、
GMT/6.0.0 (/share/tools/GMT/6.0.0 目录)。

系统安装的软件编译安装工具有：

系统默认的 make/3.82 (系统标准安装目录)、
make/4.2 (/share/tools/make/4.2 目录)、
系统默认的 cmake/2.8.12.2 (系统标准安装目录)、
cmake/3.12.2 (/share/base/cmake/3.12.2 目录)、
meson/0.50.0 (/share/tools/meson/0.50.0 目录)、
ninja/1.9.0 (/share/tools/ninja/1.9.0 目录)。

系统安装的其他辅助工具有：

系统默认的 git/1.8.3.1 (系统标准安装目录)、
git/2.18.0 (/share/base/git/2.18 目录)、
git/2.31.0 (/share/base/git/2.31 目录)、
htop/2.2.0 (/share/base/htop/2.2.0 目录)、
cURL/7.64.0 (/share/tools/curl 目录)、
autojump/22.5.3 (/share/tools/autojump/22.5.4 目录, 在加载此模块后, 还需再执行命令 *source \$AUTOJUMPROOT/.autojump/etc/profile.d/autojump.sh*, 方可调用 j 命令快速跳转目录)、
parallel/20180922 (即 GNU Parallel, /share/base/parallel 目录)、
ghostscript/9.26 (/share/tools/ghostscript/9.26 目录)、
feh/3.1.3 (/share/tools/feh/3.1.3 目录)、
valgrind/3.14.0 (/share/tools/valgrind/3.14.0 目录)、
ImageMagick/7.0.8-39 (/share/tools/ImageMagick/7.0.8-39 目录)、
FFmpeg/4.2.1 (/share/tools/FFmpeg/4.2.1 目录)、
GraphicsMagick/1.3.33 (/share/tools/GraphicsMagick/1.3.33 目录)、
GDAL/3.0.2 (/share/tools/GDAL/3.0.2 目录)、
PROJ/6.2.1 (/share/tools/PROJ/6.2.1 目录)、
SQLite/3.30.1 (/share/tools/SQLite/3.30.1 目录)、
Tcl/8.6.11 (/share/tools/Tcl/8.6.11 目录)。

3.2 module 的使用

当在 Linux 下存在多个版本的同名编译器或运行库时，如果每次编译都写上绝对路径就很麻烦，使用 environment-modules（以下简称 module）来管理环境变量则相比方便很多。

例如，在我们服务器中安装了多个版本的 intel 编译器，包括：intel/2017.8、intel/2018.3 和 intel/2018.4。在调用 intel/2017.8 时，一般先要修改 PATH 和 LD_LIBRARY_PATH 等系统环境变量。此时若要改为调用 intel/2018.4，又要再次修改这些环境变量，容易造成混乱。使用 module 则可以方便地通过 modulefile 进行系统环境变量的配置。

3.2.1 搜索路径 MODULEPATH

module 的搜索路径由系统环境变量 MODULEPATH 定义，在 MODULEPATH 包含的路径下的 modulefile 文件，可以自动被 module 识别。非 root 用户可以通过修改 MODULEPATH 加入自己的 modulefile 目录，从而实现通过 module 来管理自己在 home 目录下单独安装的软件。

3.2.2 自定义 modulefile 文件

在自定义 modulefile 文件时需注意，文件必须以 `#%Module` 开头，这样才会被 module 识别为 modulefile 文件。

常用的 modulefile 命令有：

- (1) `module-whatis "模块描述"`：添加对该 module 的描述信息；
- (2) `conflict <模块名>`：如果这里指定的模块已被加载，当前定义的模块将不会被加载；
- (3) `set <变量名> <值>`：设置变量，这里定义的变量仅在该 modulefile 文件中生效；
- (4) `setenv <系统环境变量名> <值>`：设置系统环境变量，将系统环境变量重置。这里定义的变量将在加载该模块后对整个系统生效；
- (5) `prepend-path <系统环境变量名> <新添值>`：设置系统环境变量，在原变量值前添加新的内容。

```
#%Module
conflict mpi
5 set version 3.1.1rc1
set OPENMPIROOT "/usr/mpi/gcc/openmpi-${version}"
module-whatis "Enable usage for openmpi version ${version}"
10 setenv MPI_ROOT "${OPENMPIROOT}"
```

```
#use 'normal' compact core binding as a default
setenv OMPI_MCA_rmaps_base_mapping_policy core
15 setenv OMPI_MCA_hwloc_base_binding_policy core

prepend-path PATH           "${OPENMPIROOT}/bin"
prepend-path INCLUDE        "${OPENMPIROOT}/include"
prepend-path LD_LIBRARY_PATH "${OPENMPIROOT}/lib"
20 prepend-path MANPATH       "${OPENMPIROOT}/share/man"
```

Listing 3: modulefile 文件书写示例

更多示例参见服务器上 `/share/base/modulefiles/` 目录。

3.2.3 module 模块命令

module 模块命令有：

- (1) `module avail [模块名]`：查看系统可用的模块；
- (2) `module list`：查看已经加载的模块；
- (3) `module whatis [模块名]`：查看模块描述信息；
- (4) `module load <模块名>`：加载指定模块；
- (5) `module unload <模块名>`：卸载指定模块；
- (6) `module purge`：卸载所有已加载的模块。

例如，我们要调用 `intel/2017.8` 来编译程序，可先由命令 `module load intel/2017.8` 加载该软件模块环境后，即可直接通过命令 `icc/ifort` 调用该模块的编译器来编译程序。

此时，若想换用 `intel/2018.4` 模块，可先由命令 `module unload intel/2017.8` 卸载 `intel/2017.8` 模块后，再由命令 `module load intel/2018.4` 加载 `intel/2018.4` 模块后即可。

另外，在调用 `mpi/intel` 模块时请注意，`mpicc/mpif90` 实际最终调用的还是 GNU 编译器；若想调用 *intel* 编译器，请使用 *mpiicc/mpiiifort* 命令，此时还需额外加载相应的 *intel* 编译器模块。

3.2.4 参考阅读

module 命令手册：<https://modules.readthedocs.io/en/stable/module.html>

modulefile 配置手册：<https://modules.readthedocs.io/en/stable/modulefile.html>

module 简单使用：<https://blog.csdn.net/jslove1997/article/details/80338370>

四、其他服务

4.1 公共数据

所有数据均有来源，烦请大家在使用这些数据时，正确合理地 **致谢或引用相应的数据来源**。

4.1.1 SRTM-DEM 全球地形数据

SRTM (即 Shuttle Radar Topography Mission) 项目根据获取的雷达影像处理制成了 DEM (即 Digital Elevation Model), [此处](#)有更多详细介绍。

在服务器 /share/data/SRTM-DEM/ 目录下, 分别存放有 SRTM30+(topo30.grd)和 SRTM15+(topo15.grd 和 SRTM15+V2.nc) 全球地形起伏数据, 引用分别见:

[srtm30+ 主页](https://topex.ucsd.edu/WWW_html/srtm30_plus.html): https://topex.ucsd.edu/WWW_html/srtm30_plus.html

[srtm15+ 主页](https://topex.ucsd.edu/WWW_html/srtm15_plus.html): https://topex.ucsd.edu/WWW_html/srtm15_plus.html

4.1.2 earth_relief 全球地形起伏数据

earth_relief 地形数据是由 [GMT](#) 采用其他公开地形数据制作, 并以 netCDF 格式文件发布的一组包含多种空间精度的覆盖全球的地形起伏数据。

在服务器 /share/data/earth_relief/ 目录下, 存放有该数据集中 1° 至 15'' 多种精度的原始数据。更多详细介绍及数据引用请参考:

[earth_relief 中文介绍](https://docs.gmt-china.org/latest/dataset/earth-relief): <https://docs.gmt-china.org/latest/dataset/earth-relief>

[earth_relief 英文主页](https://docs.generic-mapping-tools.org/6.0/datasets/earth_relief.html): https://docs.generic-mapping-tools.org/6.0/datasets/earth_relief.html

4.1.3 CENC 2013-2016 连续波形数据

在服务器 /share/data/CENC/ 目录下, 按月存放有 CENC (即中国地震台网中心) 全国范围部分台站自 2013 年 1 月起至 2016 年 12 月止 (除部分敏感事件日期外) 的原始连续波形数据。由于涉密的原因, 仅向保密协议范围内的使用者开放。如有需要, 在向管理员提出申请, 签署保密协议后, 即可使用。

此外, /share/data/CENC_SAC/ 目录下存放有由 /share/data/CENC/ 目录原始数据经预处理得到的 sac 格式数据。预处理的目的是将原始 miniSEED 格式数据转存为更易于后续处理的 sac 格式, 同时对原始数据做降采样处理。原始数据中大部分台站的采样频率为 100 Hz, 降采样后对应的采样频率为 10 Hz。由于数据质量和软件缺陷的原因, 部分原始 miniSEED 文件未能成功生成 sac 文件。在 /share/data/CENC_SAC/runlog/ 目录下的各月份目录中存有以下文件:

- (1) 预处理过程中使用的作业提交脚本 `m2s.lsf`, 它调用 `ms2spm.par.sh` 脚本 (内部调用 `process.sh` 脚本) 并行执行预处理操作;

- (2) 预处理过程产生的日志文件 `run.log`，它记录了原始 miniSEED 格式文件转存成功与否；
- (3) 预处理结果统计检查脚本 `check.sh`，它可以统计给出当前月份目录数据预处理转存的失败率；
- (4) 预处理转存成功的原始数据文件列表 `finish.list`，它记录了所有成功得到 sac 格式文件的当前月份原始数据文件；
- (5) 预处理转存失败的原始数据文件列表 `fail.list`，它记录了所有未能得到 sac 格式文件的当前月份原始数据文件。

对于单个原始 miniSEED 文件的预处理流程及参数如下例所示：

```
#!/bin/bash

module load mseed2sac/2.2
module load SAC/101.6a

# convert miniSEED to SAC
mseed2sac /share/data/CENC/201301/20130101/AH.ANQ.00.BHE.20130101000005.mseed

# merge and subsample
sac << END
  r *.SAC
  rmean
  rtr
  bp co 0.0077 5 n 2 p 2
  merge
  decimate 5 filter on
  decimate 2 filter on
  w merge.sac
  q
END

mv merge.sac AH.ANQ.00.BHE.20130101000005.sac
rm -f *.SAC
```

Listing 4: CENC 波形数据预处理示例

4.1.4 美国南加州部分台阵连续波形数据

在服务器 `/share/data/California/` 目录下，按台阵存放有南加州地区部分台阵（包括固定台网和流动台阵）不同时间区间的连续波形数据。所有数据均由孟浩然老师从 [IRIS 网站](#) 下载并上传至服务器，如有需要，可联系管理员获取访问权限。更多详细介绍及 [数据引用](#) 请参考：

RA: [IRIS-gmap](#), [FDSN-YN_2010](#)
Ridgecrest_nodal: [IRIS-gmap](#), [FDSN-3J_2019](#)
Ridgecrest_Permanent: [IRIS-gmap](#), [FDSN-CI](#)
SJFZ_nodal: [IRIS-gmap](#), [FDSN-ZG_2014](#)

五、常见问题解决

5.1 作业运行问题

- 1、若使用 `mpi/intel/*` 模块编译运行并行程序, 程序异常退出并报错 “unexpected DAPL connection event 0x4008” 时, 可尝试使用其他版本的 MPI 库来编译运行。
- 2、使用 `zsh` 的用户, 在提交作业时, 若出现 `module` 命令找不到的问题, 可以在提交脚本头部加入指令 `#BSUB -L /bin/bash`, 以指定作业系统采用 `bash` 解析脚本。另需注意, 如果使用 `chsh` 命令修改了用户默认登录 SHELL, 则可能导致, 即使有已提交的作业在某计算节点持续运行, 也无法通过 `ssh` 命令登录到该节点查看作业任务运行情况。
- 3、如有特殊需求, 如测试程序计算效率, 需排外地独占节点运行程序, 在提交脚本头部加入指令 `#BSUB -x` 即可。另需注意, 目前仅 `short` 队列支持独占节点作业的提交, 还请勿恶意提交独占节点作业, 以免资源浪费。