Python 及 Pymatgen 软件的安装

Pymatgen 的安装需要 Python2.7 以上版本，需要调用 Tck/Tk(要求版本 8.6 以上)，因此手动安装时配置如下:

Tcl/Tk 的安装配置

./configure --prefix=/share/home/jiangjun/Softs/tcl8.6.5 --enable-shared --enable-64bit --enable-symbols
make && make install
./configure --prefix=/share/home/jiangjun/Softs/tk8.6.5 --enable-shared --enable-64bit --enable-symbols
make && make install

lapack 和 atlas 的安装参见文档。建议**:** 用 **OpenBlas** 代替 **atlas** 更方便
安装 OpenBlas 非常简单: make CC=gcc FC=gfortran

请注意: 为了让 lapack 和 atlas 最后生成动态库函数，编译 lapack 库时，在 make.inc 中每个编译选项都加上-fPIC -m64
如
FORTRAN = gfortran
OPTS = -O3 -std=legacy -m64 -fno-second-underscore -fPIC -c
DRVOPTS = $(OPTS)
NOOPT = -O0 -frecursive -fPIC -m64
LOADER = gfortran
LOADOPTS = -fPIC -m64

CC = gcc
CFLAGS = -O3 -fPIC -m64

在非 root 权限下，atlas 推荐的编译选项:

../configure -b 64 -C ic gcc -C if gfortran -Fa alg -fPIC --prefix= 指定安装目录 --with-netlib-lapack= 静态库函数 liblapack.a 的绝对路径

手工将静态库编译为动态库
gfortran -fPIC -m64 -shared liblapack.a -o liblapack.so
gfortran -fPIC -m64 -shared libblas.a -o libblas.so

参见文档ATLAS_Numpy_Scipy_Theano 的环境搭建.pdf

python 的源码包安装配置

./configure CC=gcc --prefix=/share/home/jiangjun/Softs/Python-2.7.11 --enable-universalsdk --with-universal-archs="64-bit" --with-cxx-main=g++ --with-tcltk-includes='-I/share/home/jiangjun/lib/tcltk/include' --with-tcltk-libs='/share/home/jiangjun/lib/tcltk/lib/libtcl8.6.so /share/home/jiangjun/lib/tcltk/lib/libtk8.6.so'
make && make install

setuptools 安装 (参见文档 Python 的安装过程 (含 setuptools).pdf)
python setup.py build
python setup.py install
采用 setuptools，在可以实现在普通用户账号下安装模块 (ez_setup.py 是 python 官方给出的一个安装 setuptools 的工具)，如:
/share/software/python-2.7.10/bin/python ez_setup.py --user jiangjun
easy_install 会被安装在 ~/.local/bin/ 目录下)

在此基础上，通过选项 -d 可将各种模块安装到普通用户账号下的指定目录:
~/.local/bin/easy_install -d ~/.local/lib/python2.7/site-package 安装模块
在 ~/.bashrc 中加入下列变量 (即 PYTHONPATH 包含新增模块目录):
export PYTHONPATH=$PATHONPATH:~/.local/lib/python2.7/site-package

有此PYTHONPATH也可以用命令行，将有关模块直接安装到指定环境下
python setup.py install --prefix=~/.local
参见文档
Python 环境变量 PYTHONPATH 设置和 easy_install 简单使用.pdf
可以在配置文件 ~/.pydistutils.cfg 中指定下载镜像:
[easy_install]
index_url = http://pypi.douban.com/simple/
index_url = http://e.pypi.python.org/simple

如果采用 pip 安装模块，建议通过配置文件 ~/.pip/pip.conf 中指定有关参数:
[global]
timeout = 6000
index-url = http://pypi.douban.com/simple/
[install]
use-mirrors = true
mirrors = http://pypi.douban.com/simple/
trusted-host = pypi.douban.com
install-option=--prefix=~/.local # 指定安装路径选项

在无法联网的服务器上用 pip 安装 Python 模块的一些处理:
下载安装模块: https://pypi.python.org/simple/模块.tar.gz (无需解压)
pip install 模块. 版本.tar.gz

Python 运行 pip 时,如果提示Can't connect to HTTPS URL because the SSL module is not available.-skipping，就需要安装 openssl(1.1 或 1.02 版) 或 libressl 的 2.6.4 版本:

- wget http://www.openssl.org/source/openssl-xxx.tar.gz
- tar xvzf openssl-xxx.tar.gz
- ./config -prefix=/usr/local/openssl-xxx –openssldir=/usr/local/openssl-xxx/openssl no-zlib # 建议加上 no-zlib 否则会出现 undefined symbol: SSL_CTX_get0_param 错误
- make && make install
- echo "/usr/local/openssl-xxx/lib" » /etc/ld.so.conf

- ldconfig -v

重新编译和安装 Python，记得修改 Module/Setup 文件:

- vim /root/Python-XXXX/Modules/Setup
- 修改结果如下:

```
# Socket module helper for socket(2)
_socket socketmodule.c timemodule.c
# Socket module helper for SSL support; you must comment out the other
# socket line above, and possibly edit the SSL variable:
SSL=/usr/local/openssl-xxx
_ssl _ssl.c
-DUSE_SSL  -I$(SSL)/include  -I$(SSL)/include/openssl
-L$(SSL)/lib -lssl -lcrypto
```

- 运行 python
- import ssl 测试正常即可

在此基础上，准备手动安装 numpy、scipy 等模块，建议指定环境变量 ATLAS(版本 3.8.4)、LAPACK(版本 3.6.0)、BLAS
注意: 环境变量要具体到绝对路径 (包括文件名)，如:
export ATLAS=/share/home/jiangjun/lib/atlas3.8.4/lib/libatlas.so
export LAPACK=/share/home/jiangjun/lib/atlas3.8.4/lib/liblapack.so
export BLAS=/share/home/jiangjun/lib/atlas3.8.4/lib/libblas.so
分别进入 numpy 和 scipy 目录
在文件 site.cfg 中指定有关库函数的参数 (注意: 写绝对路径)

```
[ALL]
library_dirs =/share/home/jiangjun/lib/atlas3.8.4/lib
include_dirs =/share/home/jiangjun/lib/atlas3.8.4/include
src_dir=/share/home/jiangjun/lib/atlas3.8
search_static_first=0
#
# Atlas
# -----
# Atlas is an open source optimized implementation of the BLAS and Lapack
# routines. Numpy will try to build against Atlas by default when available in
# the system library dirs. To build numpy against a custom installation of
# Atlas you can add an explicit section such as the following. Here we assume
# that Atlas was configured with “prefix=/opt/atlas“.
#
[atlas]
library_dirs = /share/home/jiangjun/lib/atlas3.8.4/lib
include_dirs = /share/home/jiangjun/lib/atlas3.8.4/include
atlas_libs=lapack, f77blas, cblas, atlas


[blas_opt]
library_dirs = /share/home/jiangjun/lib/atlas3.8.4/lib
```

include_dirs = /share/home/jiangjun/lib/atlas3.8.4/include
blas_libs=f77blas, cblas, atlas

[lapack_opt]
library_dirs = /share/home/jiangjun/lib/atlas3.8.4/lib
include_dirs = /share/home/jiangjun/lib/atlas3.8.4/include
lapack_libs=lapack, atlas

[amd]
amd_libs = amd
#
[umfpack]
umfpack_libs = umfpack

用 **Intel_mkl** 库支持 **numpy** 和 **scipy**
[mkl]
library_dirs = $intel 编译器安装目录/mkl/lib/intel64/
include_dirs = $intel 编译器安装目录/mkl/include
mkl_libs = mkl_rt mkl_blas95_lp64 mkl_core mkl_intel_lp64 mkl_intel_thread
lapack_libs =mkl_lapack95_lp64

修改 $numpy-目录/numpy/distutils/intelccompiler.py:
将self.cc_exe(class intel 或 class intelem 里的)为:
self.cc_exe = 'icc -O3 -g -fPIC -fp-model strict -fomit-frame-pointer -openmp -xhost'
numpy 编译:
python setup.py config --compiler=intel build_clib --compiler=intel build_ext --compiler=intel install
64 位将intel改为intelem
scipy 编译:
python setup.py config --compiler=intel --fcompiler=intel build_clib --compiler=intel --fcompiler=intel build_ext --compiler=intel --fcompiler=intel install
64 位同 numpy 修改命令行

python setup.py build
sudo python setup.py install

在 Ubuntu 系统中，如果有超级用户权限，则可以简单处理为:
sudo apt-get install libopenblas-dev liblapack-dev
export BLAS=/usr/lib/libblas.so
export LAPACK=/usr/lib/liblapack.so

在此基础上完成安装:
pip install numpy
pip install scipy

1 关于 MongoDB 的启动

MongoDB 可以通过控制文件 (如文件名为 mongodb.config) 来控制启动，mongodb.config 中指定各种参数:

```
fork = true
dbpath = /home/jun_jiang/WORKS/TEST/TEST_mongo/Data_Base
logpath = /home/jun_jiang/WORKS/TEST/TEST_mongo/mongo.log
logappend =true
journal = true
# repair = true
bind_ip = 127.0.0.1,192.168.113.42# 这里的 IP(192.168.113.42) 为数据库服务器的内部 IP(即 ifconfig 命令查询到的本机 IP)
port = 27017
```

如果需要在同一地点启动多个数据库，只需要修改 dbpath 即可

如 mongodb_atomate.config:

```
fork = true
dbpath = /home/jun_jiang/WORKS/TEST/TEST_mongo/Data_Base_Atomate
logpath = /home/jun_jiang/WORKS/TEST/TEST_mongo/mongo.log
logappend =true
journal = true
# repair = true
bind_ip = 127.0.0.1,192.168.113.42# 这里的 IP(192.168.113.42) 为数据库服务器的内部 IP(即 ifconfig 命令查询到的本机 IP)
port = 27017
```

如 mongodb_firework.config:

```
fork = true
dbpath = /home/jun_jiang/WORKS/TEST/TEST_mongo/Data_Base_Firework
logpath = /home/jun_jiang/WORKS/TEST/TEST_mongo/mongo.log
logappend =true
journal = true
# repair = true
bind_ip = 127.0.0.1,192.168.113.42# 这里的 IP(192.168.113.42) 为数据库服务器的内部 IP(即 ifconfig 命令查询到的本机 IP)
port = 27017
```

启动数据库的命令为: mongod -f mongodb_XXXX.config

进入数据库的命令为: mongo，然后用 help 检查 MongoDB 的各种命令

2 Pymatgen

采用 Pymatgen 生成 VASP 的 POTCAR 库函数时，用的命令是
pmg config -p Dir-source Dir-object
然后用 pmg config add PMG_VASP_PSP_DIR Dir-object 生效即可

pmg potcar 命令是用于根据 POTCAR 库产生对应元素的 POTCAR