Siesta 5.0.0 MPI (Open MPI)

ウェブページ

https://gitlab.com/siesta-project/siesta

バージョン

5.0.0 (+ELPA 2024.03.001, ELSI 2.9.1, NetCDF 4.9.2, NetCDF Fortran 4.6.1, libxc 6.2.2)

ビルド環境

- GCC 13.1.1 (gcc-toolset-13)
- Intel MKL 2024.1
- Open MPI 4.1.6
- autoconf 2.72 (for ELPA)
- Python 3.9
    - rumael.yaml (pip3.9 install ruamel.yaml --user)

ビルドに必要なファイル

- siesta-5.0.0.tar.gz
- netcdf-c-4.9.2.tar.gz
- netcdf-fortran-4.6.1.tar.gz
- libxc-6.2.2.tar.bz2
- elsi_interface-v2.9.1.tar.gz
- elpa-2024.03.001.tar.gz
- wannier90-3.1.0.tar.gz
- gnu_rccs_ompi.cmake (for ELSI 2.9.1)

```
### GCC ###

SET(CMAKE_Fortran_COMPILER "mpif90" CACHE STRING "MPI Fortran compiler")
SET(CMAKE_C_COMPILER "mpicc" CACHE STRING "MPI C compiler")
SET(CMAKE_CXX_COMPILER "mpicxx" CACHE STRING "MPI C++ compiler")

SET(CMAKE_Fortran_FLAGS "-O3 -fallow-argument-mismatch" CACHE STRING "Fortran flags")
SET(CMAKE_C_FLAGS "-O3 -std=c99" CACHE STRING "C flags")
SET(CMAKE_CXX_FLAGS "-O3 -std=c++11" CACHE STRING "C++ flags")

SET(ENABLE_PEXSI ON CACHE BOOL "Enable PEXSI")
SET(ENABLE_TESTS ON CACHE BOOL "Enable Fortran tests")
SET(ENABLE_C_TESTS ON CACHE BOOL "Enable C tests")

SET(LIB_PATHS "$ENV{MKLROOT}/lib/intel64" CACHE STRING "External library paths")
SET(LIBS "mkl_scalapack_lp64 mkl_gf_lp64 mkl_sequential mkl_core mkl_blacs_openmpi_lp64 m dl" CACHE
```

```
STRING "External libraries")
```

ビルド手順

ELSI (for PEXSI, not for ELPA)

```
#!/bin/sh

VERSION=2.9.1
INSTDIR=/apl/siesta/5.0.0/exts

BASEDIR=/home/users/${USER}/Software/ELSI/${VERSION}
TARBALL=${BASEDIR}/elsi_interface-v${VERSION}.tar.gz

WORKDIR=/gwork/users/${USER}

MYTC_NAME=gnu_rccs_ompi.cmake
MYTC=${BASEDIR}/${MYTC_NAME}

PARALLEL=24
export LANG=C
export LC_ALL=C

# -----------------------------------------
umask 0022
ulimit -s unlimited

cd ${WORKDIR}
if [ -d elsi_interface-v${VERSION} ]; then
 mv elsi_interface-v${VERSION} elsi-erase
 rm -rf elsi-erase &
fi

module -s purge
module -s load gcc-toolset/13
module -s load mkl/2024.1
module -s load openmpi/4.1.6/gcc13

tar xf ${TARBALL}
cd elsi_interface-v${VERSION}
cp ${MYTC} toolchains

mkdir build && cd build
cmake .. \
    -DCMAKE_INSTALL_PREFIX=${INSTDIR} \
    -DCMAKE_TOOLCHAIN_FILE=../toolchains/${MYTC_NAME} \
    -DBUILD_SHARED_LIBS=ON

make -j ${PARALLEL}
make test
make install
```

ELPA

```
#!/bin/sh

ELPA_VERSION=2024.03.001
INSTDIR=/apl/siesta/5.0.0/elpa
WORKDIR=/gwork/users/${USER}

BASEDIR=/home/users/${USER}/Software/ELPA/${ELPA_VERSION}
TARBALL=${BASEDIR}/elpa-${ELPA_VERSION}.tar.gz

PARALLEL=12

#-----------------------------------------------------------------------
umask 0022
ulimit -s unlimited

module -s purge
module -s load gcc-toolset/13
module -s load mkl/2024.1
module -s load openmpi/4.1.6/gcc13
module -s load autoconf/2.72

export LANG=C
export LC_ALL=C

export FC=mpif90
export CC=mpicc
export CXX=mpicxx
export CFLAGS="-march=znver3"
# mkl_link_tool -opts -c gnu_f -p no --cluster_library=scalapack -m openmpi
export FCFLAGS="-m64 -I${MKLROOT}/include"
# mkl_link_tool -libs -c gnu_f -p no --cluster_library=scalapack -m openmpi
export LDFLAGS="-L${MKLROOT}/lib -lmkl_scalapack_lp64 -Wl,--no-as-needed -lmkl_gf_lp64 -lmkl_sequential -
lmkl_core -lmkl_blacs_openmpi_lp64 -lpthread -lm -ldl"

cd ${WORKDIR}
if [ -d elpa-${ELPA_VERSION} ]; then
 mv elpa-${ELPA_VERSION} elpa-erase
 rm -rf elpa-erase &
fi
tar zxf ${TARBALL}
cd elpa-${ELPA_VERSION}

./configure --prefix=${INSTDIR} \
        --disable-avx512-kernels
make -j ${PARALLEL}
make check
make install
```

Siesta

```
#!/bin/sh

SIESTA_VERSION=5.0.0
INSTDIR=/apl/siesta/5.0.0

WORKDIR=/gwork/users/${USER}
BASEDIR=/home/users/${USER}/Software/Siesta/${SIESTA_VERSION}
TARBALL=${BASEDIR}/siesta-${SIESTA_VERSION}.tar.gz

NETCDF_C_VERSION=4.9.2
NETCDF_F_VERSION=4.6.1
BASEDIR_NETCDF=/home/users/${USER}/Software/NETCDF
TARBALL_NETCDF_C=${BASEDIR_NETCDF}/c${NETCDF_C_VERSION}/netcdf-c-
${NETCDF_C_VERSION}.tar.gz
TARBALL_NETCDF_F=${BASEDIR_NETCDF}/f${NETCDF_F_VERSION}/netcdf-fortran-
${NETCDF_F_VERSION}.tar.gz
WANNIER90_VERSION=3.1.0
BASEDIR_WANNIER90=/home/users/${USER}/Software/wannier90/${WANNIER90_VERSION}
TARBALL_WANNIER90=${BASEDIR_WANNIER90}/wannier90-${WANNIER90_VERSION}.tar.gz

LIBXC_VERSION=6.2.2
BASEDIR_LIBXC=/home/users/${USER}/Software/libxc
TARBALL_LIBXC=${BASEDIR_LIBXC}/${LIBXC_VERSION}/libxc-${LIBXC_VERSION}.tar.bz2

PARALLEL=12

#-----------------------------------------------------------------------------
umask 0022
ulimit -s unlimited

module -s purge
module -s load gcc-toolset/13
module -s load mkl/2024.1
module -s load openmpi/4.1.6/gcc13

export LANG=C
export LC_ALL=C
export OMP_NUM_THREADS=1

# netcdf-c

cd ${WORKDIR}
if [ -d netcdf-c-${NETCDF_C_VERSION} ]; then
 mv netcdf-c-${NETCDF_C_VERSION} netcdf-c-erase
 rm -rf netcdf-c-erase &
fi
tar zxf ${TARBALL_NETCDF_C}
cd netcdf-c-${NETCDF_C_VERSION}
```

```
./configure --prefix=${INSTDIR}/exts
make -j${PARALLEL}
make -j${PARALLEL} check
make install

export PATH="${INSTDIR}/exts/bin:${PATH}"
export CPATH="${INSTDIR}/exts/include:${CPATH}"
export LD_LIBRARY_PATH="${INSTDIR}/exts/lib:${LD_LIBRARY_PATH}"
export LIBRARY_PATH="${INSTDIR}/exts/lib:${LIBRARY_PATH}"

# netcdf-f

cd ${WORKDIR}
if [ -d netcdf-fortran-${NETCDF_F_VERSION} ]; then
 mv netcdf-fortran-${NETCDF_F_VERSION} netcdf-fortran-erase
 rm -rf netcdf-fortran-erase &
fi
tar zxf ${TARBALL_NETCDF_F}
cd netcdf-fortran-${NETCDF_F_VERSION}

LDFLAGS="-L${INSTDIR}/exts/lib" \
   ./configure --prefix=${INSTDIR}/exts
make -j${PARALLEL}
make -j${PARALLEL} check
make install

# libxc

cd ${WORKDIR}
if [ -d libxc-${LIBXC_VERSION} ]; then
 mv libxc-${LIBXC_VERSION} libxc-erase
 rm -rf libxc-erase &
fi
tar xf ${TARBALL_LIBXC}
cd libxc-${LIBXC_VERSION}
autoreconf -i
./configure --prefix=${INSTDIR}/exts
make -j24
make -j24 check
make install

# siesta

cd ${WORKDIR}
rm -rf netcdf-fortran-${NETCDF_F_VERSION} \
      netcdf-c-${NETCDF_C_VERSION} \
      libxc-${LIBXC_VERSION} &

if [ -d siesta-${SIESTA_VERSION} ]; then
 mv siesta-${SIESTA_VERSION} siesta-erase
```

```
 rm -rf siesta-erase
fi

tar zxf ${TARBALL}
cd siesta-${SIESTA_VERSION}

unset CC
unset FC
export WANNIER90_PACKAGE=${TARBALL_WANNIER90}

mkdir build && cd build
cmake .. \
 -DCMAKE_INSTALL_PREFIX="${INSTDIR}" \
 -DCMAKE_PREFIX_PATH="${INSTDIR}/elpa;${INSTDIR}/exts" \
 -DCMAKE_C_COMPILER=mpicc \
 -DCMAKE_Fortran_COMPILER=mpif90 \
 -DPython3_EXECUTABLE=/usr/bin/python3.9 \
 -DSIESTA_WITH_MPI=ON \
 -DNetCDF_ROOT="${INSTDIR}/exts" \
 -DLAPACK_LIBRARY="-m64 -L${MKLROOT}/lib -Wl,--no-as-needed -lmkl_gf_lp64 -lmkl_sequential -lmkl_core -
lpthread -lm -ldl" \
 -DBLAS_LIBRARY="-m64 -L${MKLROOT}/lib -Wl,--no-as-needed -lmkl_gf_lp64 -lmkl_sequential -lmkl_core -
lpthread -lm -ldl" \
 -DSCALAPACK_LIBRARY="-lmkl_scalapack_lp64 -lmkl_gf_lp64 -lmkl_sequential -lmkl_core -
lmkl_blacs_openmpi_lp64 -lpthread -lm -ldl" \
 -DSIESTA_WITH_WANNIER90=ON \
 -DSIESTA_WITH_ELPA=ON \
 -DSIESTA_WITH_PEXSI=ON

make -j ${PARALLEL}
SIESTA_TESTS_VERIFY=1 ctest
make install

cd ../
cp -r Examples ${INSTDIR}
```

テスト

ELSI

すべてパス

ELPA

以下のテストでエラー

- FAIL: validate_c_version_complex_double_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_c_version_real_double_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_c_version_complex_single_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_c_version_real_single_eigenvalues_2stage_default_kernel_analytic_default.sh

- FAIL: validate_c_version_complex_double_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_c_version_complex_double_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_c_version_real_double_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_c_version_real_double_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_c_version_complex_single_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_c_version_complex_single_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_c_version_real_single_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_c_version_real_single_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_cpp_version_complex_double_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_cpp_version_real_double_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_cpp_version_complex_single_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_cpp_version_real_single_eigenvalues_2stage_default_kernel_analytic_default.sh
- FAIL: validate_cpp_version_complex_double_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_cpp_version_complex_double_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_cpp_version_real_double_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_cpp_version_real_double_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_cpp_version_complex_single_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_cpp_version_real_single_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_cpp_version_real_single_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_analytic_default.sh
- FAIL: validate_real_single_eigenvectors_2stage_default_kernel_analytic_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_frank_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_random_split_comm_myself_explicit_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_random_split_comm_myself_default.sh
- FAIL: validate_real_single_eigenvectors_2stage_default_kernel_random_explicit_default.sh
- FAIL: validate_real_single_eigenvectors_2stage_default_kernel_random_default.sh
- FAIL: validate_real_double_eigenvectors_2stage_default_kernel_toeplitz_default.sh
- FAIL: validate_real_single_eigenvectors_2stage_default_kernel_toeplitz_default.sh
- FAIL: validate_complex_2stage_banded_default.sh
- FAIL: validate_single_complex_2stage_banded_default.sh

Siesta

(netcdf, libxc についてはすべてパス)
siesta
については以下のテストで数値エラー。ほとんどは軽微なエラー。一部で少し大きめな誤差が発生しているものの、コンパイ

- 61 - siesta-02.SpinPolarization-fe_spin_mpi4[verify] (Failed)
- 63 - siesta-02.SpinPolarization-fe_spin_directphi_mpi4[verify] (Failed)
- 67 - siesta-02.SpinPolarization-fe_noncol_gga_mpi4[verify] (Failed)
- 71 - siesta-02.SpinPolarization-fe_noncol_sp_mpi4[verify] (Failed)

- 73 - siesta-03.SpinOrbit-FePt-X-X_mpi4[verify] (Failed)
- 87 - siesta-04.SCFMixing-chargemix_mpi4[verify] (Failed)
- 127 - siesta-08.GeometryOptimization-broyden_vc_mpi4[verify] (Failed)

テストログは /apl/siesta/5.0.0/test_results 以下のコピーを保存しています。

メモ

- 今回は openmp 版は省略
- インテルコンパイラを使った時には siesta のエラーが増える(数値エラー)。速度的にも大きなメリットは確認できなかったため、今回は gcc13 を採用
  - AMD CPU でインテルコンパイラを使う場合、-xHost 指定が原因でエラーが発生することがある。Config/cmake/toolchains/intel.cmake にある -xHost を -march=core-avx2 に変更することで回避可能
  - icx+ifort の組み合わせでのビルドは可能。icx+ifx ではビルドできず。
- Intel MPI を使った場合、並列数をあげた時に Bad DM normalization: Qtot, Tr[D*S] = 548.00000000 548.00983099 のようなメッセージでエラー終了する場合があるため openmpi をメインで使用。
  - Examples/Carbon_Nanoscroll を 64 並列で実行した時に確認。高確率で再現する。Open MPI 版では発生せず。
  - 小規模並列時(4 or 8)には問題を確認できていない。
  - Intel MPI 版は一応別途作成。gcc13 が使えないため gcc11 を利用。
  - Intel MPI 版の複数バージョン(2021.5.1 - 2021.12)で試すも改善せず
  - (2025/2/13 更新) I_MPI_HYDRA_TOPOLIB 環境変数の値を ipl に設定することで問題が解消することを確認。module 内で定義するようにしたため、今後は影響を受けないと思われます。
- PEXSI を有効にするとログ中で &m -- Max memory after compute_DM 166.26 のようなメッセージが表示されるようになる。
  - SCF の最中にも徐々に数字が増えているようにも見える
- ELSI 内の ELPA を使うことは難しそうであったため、ELPA だけ別に用意している。
- siesta のテストで数値チェックをするためには Python 環境が必要
  - ruamel.yaml が必要であるため事前に pip3.9 install ruamel.yaml --user を実行して導入
  - テストコードで Python 3.8 を期待する部分があるため、システム標準の python (バージョン 3.6) は利用できず。
    - Pathlib の unlink メソッドで missing_ok のフラグを指定する部分の問題。(フラグ部分だけを削除すれば python 3.6 でも動作したかもしれないが未検証。)
  - CMake が python3 を探す場合、システム標準ではないものを優先的に見つける場合がある。今回はシステム標準の 3.6 では上記の問題があったことも有り、明示的に 3.9 を指定。