Psana Build Instructions

There are two environments in this directory. Psana_environment.yml is suitable for general use and contains the usual CCTBX dependencies plus psana and its dependencies. Psana_lcls_environment.yml should only be used at LCLS; it adds a local build of mpi4py that is required for running jobs on multiple psana nodes.

The build steps below were tested on Oct 28, 2020. They should be done in a clean environment (no cctbx installation has been activated, etc.)

Creating the conda environment (step `base`) may take up to ~20 min with no obvious progress occurring.

General build

These steps were tested on a CentOS 7.4.1708 machine with 64 cores. In the bootstrap.py step you should adjust nproc to suit your environment.

```
$ mkdir cctbx; cd cctbx
$ wget
https://raw.githubusercontent.com/cctbx/cctbx_project/master/libtbx/auto_buil

$ wget
```

```
https://raw.githubusercontent.com/cctbx/cctbx_project/master/xfel/conda_envs/

$ python bootstrap.py --builder=xfel --use-conda=psana_environment.yml --
nproc=64 --python=37 --no-boost-src hot update base
$ conda activate `pwd`/conda_base # if no conda is availble, first source
mc3/etc/profile.d/conda.sh
$ python bootstrap.py --builder=xfel --use-conda=psana_environment.yml --
nproc=64 --python=37 build
$ source build/conda_setpaths.sh
$ libtbx.python -c "import psana" # Should exit with no output
```

LCLS build

These steps were tested on an ssh connection to pslogin.slac.stanford.edu. You will have to remember the location of $INSTALL because a new ssh connection is opened before the build step.

```
$ ssh psexport
$ cd $INSTALL; mkdir cctbx; cd cctbx
$ wget
https://raw.githubusercontent.com/cctbx/cctbx_project/master/libtbx/auto_buil

$ wget
https://raw.githubusercontent.com/cctbx/cctbx_project/master/xfel/conda_envs/

$ python bootstrap.py --builder=xfel --use-
conda=psana_lcls_environment.yml \
  --no-boost-src --python=37 hot update base
$ mkdir `pwd`/conda_base/lib/hdf5
$ ln -s `pwd`/conda_base/lib/plugins `pwd`/conda_base/lib/hdf5/plugin #
needed until dials 3.4 is released
$ exit  # logout of psexport
$ ssh psana
$ cd $INSTALL/cctbx
$ conda  # If you get a usage message, skip the next step.
$ source mc3/etc/profile.d/conda.sh # Activate conda. It could also be
found at
                                    # ~/miniconda3/etc/profile.d/conda.sh
$ conda activate `pwd`/conda_base
$ python bootstrap.py --builder=xfel --use-
conda=psana_lcls_environment.yml \
  --config-flags="--compiler=conda" --config-flags="--
use_environment_flags" \
  --config-flags="enable_cxx11" --config-flags="--no_bin_python"
\
  --no-boost-src --python=37 --nproc=10 build
```

```
$ source build/conda_setpaths.sh
$ source modules/cctbx_project/xfel/conda_envs/test_psana_lcls.sh
```

Alternative LCLS build

The `bootstrap.py base` step above takes >1 hr, mainly to create the conda environment. This can be improved to ~15 min using Mamba, a C++ implementation of Conda. You need a base Conda environment with Mamba installed (`conda install mamba -c conda-forge`).

```
$ ssh psexport
$ cd $INSTALL; mkdir cctbx; cd cctbx
$ wget
https://raw.githubusercontent.com/cctbx/cctbx_project/master/libtbx/auto_buil

$ wget
https://raw.githubusercontent.com/cctbx/cctbx_project/master/xfel/conda_envs/

$ python bootstrap.py --builder=xfel --use-
conda=psana_lcls_environment.yml \
  --no-boost-src --python=37 hot update
$ source ~/miniconda3/etc/profile.d/conda.sh # modify as needed
$ conda activate base
$ mamba env create -f psana_lcls_environment.yml -p `pwd`/conda_base
$ exit  # logout of psexport
$ ssh psana
$ cd $INSTALL/cctbx
$ conda  # If you get a usage message, skip the next step.
$ source mc3/etc/profile.d/conda.sh # Activate conda. It could also be
found at
                                    # ~/miniconda3/etc/profile.d/conda.sh
```

129 lines (110 sloc) | 5.62 KB

```
conda=psana_lcls_environment.yml \
  --config-flags="--compiler=conda" --config-flags="--
use_environment_flags" \
  --config-flags="enable_cxx11" --config-flags="--no_bin_python"
\
  --no-boost-src --python=37 --nproc=10 build
$ source build/conda_setpaths.sh
$ source modules/cctbx_project/xfel/conda_envs/test_psana_lcls.sh
```

cctbx.xfel tests

The cctbx.xfel regression tests include tests from several repositories. The below instructions reproduce what we do nightly. If psana is configured, it will be tested as well.

```
$ cd modules
$ conda install -c conda-forge git-lfs
$ git clone https://gitlab.com/cctbx/xfel_regression.git
$ git clone https://github.com/nksauter/LS49.git
$ git clone https://gitlab.com/cctbx/ls49_big_data.git
$ cd xfel_regression
$ git lfs install --local
$ git lfs pull
$ cd ../uc_metrics
$ git lfs install --local
$ git lfs pull
$ cd ../ls49_big_data
$ git lfs install --local
$ git lfs pull
$ cd ../../
$ mkdir test; cd test
$ libtbx.configure xfel_regression LS49 ls49_big_data
$ export OMP_NUM_THREADS=4
$ libtbx.run_tests_parallel module=uc_metrics module=simtbx
module=xfel_regression module=LS49 nproc=64
```

Note, bootstrap.py has several 'builders' available that set up which packages are cloned and configured. The xfel builder will clone uc_metrics for you, but for reference, here's how to get it standalone if needed:

```
$ git clone https://gitlab.com/cctbx/uc_metrics.git
$ libtbx.configure uc_metrics
$ cd `libtbx.show_build_path`; make
```

Note, when running these tests at LCLS itself on its psana computing cluster, the following environment variable needs to be exported to suppress a warning:

```
export OMPI_MCA_mca_base_component_show_load_errors=0
```