Reproducing Results for 112018-00062

This document walks through the necessary steps to reproduce the results reported in paper #112018-00062 submitted for possible publication in Mathematical Programming Computation. Hence, the below `bash` commands are written for their servers, and should be modified accordingly for other *nix platforms.

NOTE. We assume that the host machine has `configure`, `make` and proper compiler toolchains installed and configured properly. We also assume that the host has `curl`, `bunzip2` and `pdflatex` (with `mathtools` and `pgfplots` packages) installed.

Initial Setup

Before anything else, we should make a `local` directory under `$HOME` that contains binaries, libraries and configuration files of local installations of the programs:

```
mkdir -p $HOME/local/{bin,etc,include,lib,share}
ln -s lib $HOME/local/lib64
```

Then, we should modify `$HOME/.bash_profile` so that the environment variables `PATH` and `LD_LIBRARY_PATH` point to the correct locations:

```
# $HOME/.bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
  . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$HOME/local/bin:$PATH
export PATH

LD_LIBRARY_PATH=$HOME/local/lib
export LD_LIBRARY_PATH
```

After saving the file, we need to `source $HOME/.bash_profile` to make the changes valid for the current session.

Now that we have setup the paths, we can proceed with the installation procedure.

Installation

POLO requires CMake (at least `v3.9.0`) to install its headers and C-API while managing its dependencies. It is also recommended to install the dependencies

via CMake itself. For this reason, we first install CMake from source:

```
# Build and install CMake from source
curl --output /tmp/cmake.tar.gz   \
  https://gitlab.kitware.com/cmake/cmake/-/archive/v3.9.0/cmake-v3.9.0.tar.gz
tar xzf /tmp/cmake.tar.gz -C /tmp
cd /tmp/cmake-v3.9.0
./configure --prefix=$HOME/local  \
            --datadir=share/cmake \
            --docdir=doc/cmake    \
            --no-qt-gui
make
make install
```

NOTE. At this point, we *might* need to logoff and login back to make environment changes valid so that `which cmake` points to the local installation with `cmake --version` reporting `3.9.0`.

Then, we install the dependencies, i.e., 0MQ (`v4.2.5`), OpenBLAS (`v0.3.3`), cereal (`v1.2.2`) and Google Test (`v1.8.1`, for unit testing), one by one:

```
# Build and install 0MQ from source
git clone https://github.com/zeromq/libzmq /tmp/libzmq
cd /tmp/libzmq
git checkout -b install v4.2.5
mkdir build
cd build
cmake -D CMAKE_INSTALL_PREFIX=$HOME/local \
      -D CMAKE_BUILD_TYPE=Release         \
      -D ENABLE_DRAFTS=OFF                \
      -D ENABLE_CURVE=OFF                 \
      -D BUILD_TESTS=OFF                  \
      -D BUILD_SHARED=ON                  \
      -D BUILD_STATIC=ON                  \
      -D WITH_OPENPGM=OFF                 \
      -D WITH_DOC=OFF                     \
      -D LIBZMQ_WERROR=OFF                \
      -D LIBZMQ_PEDANTIC=OFF              \
      ../
cmake --build .
cmake --build . --target install

# Build and install OpenBLAS from source
git clone https://github.com/xianyi/OpenBLAS /tmp/OpenBLAS
cd /tmp/OpenBLAS
git checkout -b install v0.3.3
mkdir build
cd build
```

```
cmake -D CMAKE_INSTALL_PREFIX=$HOME/local \
      -D CMAKE_BUILD_TYPE=Release         \
      -D BUILD_SHARED_LIBS=ON             \
      -D BUILD_WITHOUT_LAPACK=OFF         \
      -D BUILD_WITHOUT_CBLAS=ON           \
      -D DYNAMIC_ARCH=OFF                 \
      ../
cmake --build .
cmake --build . --target install

# Build and install cereal from source
git clone https://github.com/USCiLab/cereal /tmp/cereal
cd /tmp/cereal
git checkout -b install v1.2.2
mkdir build
cd build
cmake -D CMAKE_INSTALL_PREFIX=$HOME/local \
      -D JUST_INSTALL_CEREAL=ON           \
      ../
cmake --build .
cmake --build . --target install

# Build and install Google Test from source
git clone https://github.com/google/googletest /tmp/googletest
cd /tmp/googletest
git checkout -b install release-1.8.1
mkdir build
cd build
cmake -D CMAKE_INSTALL_PREFIX=$HOME/local \
      -D CMAKE_BUILD_TYPE=Release         \
      -D BUILD_SHARED_LIBS=ON             \
      ../
cmake --build .
cmake --build . --target install
```

Finally, we are ready to install POLO from source:

```
git clone https://github.com/pologrp/polo /tmp/polo
mkdir /tmp/polo/build
cd /tmp/polo/build
cmake -D CMAKE_INSTALL_PREFIX=$HOME/local \
      -D CMAKE_PREFIX_PATH=$HOME/local    \
      -D CMAKE_BUILD_TYPE=Release         \
      -D BUILD_SHARED_LIBS=ON             \
      ../
cmake --build .
cmake --build . --target test
```

```
cmake --build . --target install
```

Experiments

Having installed POLO and its dependencies successfully, we clone the repository and generate the figures reported in the paper by issuing the following:

```
git clone https://github.com/pologrp/experiments $HOME/experiments
mkdir $HOME/experiments/build
cd $HOME/experiments/build
cmake -D CMAKE_PREFIX_PATH=$HOME/local    \
      -D CMAKE_BUILD_TYPE=Release         \
      ../
cmake --build .
cmake --build . --target figures
```

The above snippet will build the example scripts, run the resulting binaries with both generated and actual test data, and finally create a `figures.pdf` file under `$HOME/experiments/build`.