BIOHDF – GETTING STARTED

VERSION 0.4 ALPHA

DECEMBER 2011

Hello everyone!

This is the BioHDF 0.4 alpha distribution. The main improvements have been to the build system, error checking and testing. It is still largely un-optimized.

BUILDING AND INSTALLING FROM SOURCE

The BioHDF source tarball can be obtained from the downloads page at http://www.biohdf.org.

INSTALLING HDF5

If a zlib-linked HDF5 distribution is not present on your system, you will need to go get it. First of all…

***** DO NOT OBTAIN HDF5 FROM YOUR PACKAGE MANAGER *****

The HDF5 Group does not provide RPMs, DEBs, etc. for the various Linux distribution package managers so these binaries are created by third parties and they are typically either out of date[1] or built with settings that are incompatible with BioHDF.

Linux, Mac OS X and other Unix-like POSIX systems (including Cygwin)

To get HDF5 for your system, I recommend building from source.

Go to http://www.hdfgroup.org/HDF5/release/obtain5.html and download the source tarball. The only pre-requisites you might need aside from a compiler are the development headers, etc. for zlib. Those are often installed separately from the main zlib library.

On most systems, you can just use:

[1] For example, Ubuntu's package is based on HDF5 1.8.4, which is two years (four releases) old.

```
./configure

make

make check

make install
```

to build the HDF5 C library. By default, make install will put the HDF5 targets in a subdirectory of the build directory named hdf5/ so you won't need to be root.

Windows

On Windows, we recommend downloading HDF5 binaries. Use the ones that do **not** say 'CMake' in the description (they don't use a Windows installer for installation). For building BioHDF, the contents of the distribution need to be copied into a specific subfolder of the BioHDF source tree so that Visual Studio can find them[2].

<biohdf_folder>\windows\external_lib\<platform>\hdf5

<platform> will be either win32 or x64, depending on whether you are building 32- or 64-bit BioHDF.

If you have installed HDF5 to some other location, you'll have to adjust the link and include paths in the Visual Studio projects.

BUILDING BIOHDF

Linux, Mac OS X and other Unix-like POSIX systems (including Cygwin)

Use:

```
./configure <options>

make

make check

make install
```

to build the BioHDF library and tools. The tests in `make check` are not fully implemented but you should not get any failures.

Some key configure options are:

`--enable-silent-rules` BioHDF supports less verbose builds that make it easier to spot warnings.

[2] In the future, we hope to move to CMake for generating the Visual Studio projects which will make this a lot easier.

`--with-hdf5=DIR`	The path to the HDF5 binaries, if they are not in the system locations.
`--prefix=DIR`	Where to install BioHDF.
`--help`	Complete list of build options.

By default, BioHDF installs to a subdirectory of the build folder so you don't need to be root to run `make install`.

Windows

Open up the BioHDF.sln file in Visual Studio and build the software. The library and tools will be in the windows/bin directory. Note that the HDF5 binaries will be assumed to be in the correct windows/external_lib subfolder in the distribution (see HDF5 installation, above).

TESTING BIOHDF

Linux, Mac OS X and other Unix-like POSIX systems (including Cygwin)

Run `'make check'` to test some of the fundamental API functionality and run the tools through their paces. Some of the tests are not fully implemented at this time, but you should not get failures (reports of incomplete tests are ok).

Windows

On Windows, you can run the test_vectors, test_reads and test_alignments programs individually. The tools tests are unavailable on Windows since they use a Bourne shell script.

WHAT YOU GET

Building BioHDF will create several tools in addition to the libraries:

- bioh5g_import_reads
- bioh5g_export_reads
- bioh5g_import_alignments
- bioh5g_export_alignments
- bioh5g_preparse_sam

LIBBIOHDF

This is the core BioHDF library. The header files that do not contain "_internal" describe the API. You should only need to include two files – biohdf.h and bioh5g.h – when building software that targets the library. The sub-files (biohdf_file.h, etc.) are included via those two master files. Any function which begins with BIOHDF_API is an official API function. Internal functions begin with an underscore and do not include the BIOHDF_API declaration. The /doc/api directory includes HTML API documentation created using Doxygen. All BioHDF documentation is also available on the web.

BIOH5G_IMPORT_READS

This tool imports reads from a file into BioHDF.

- FASTA and FASTQ supported
- Input from a file or stdin

NOTE: It is not necessary to import the reads separately from the alignments.

BIOH5G_EXPORT_READS

This tool exports reads from a BioHDF file.

- FASTQ and FASTA supported. Can also write out just the sequences or just the quality values, one line at a time.
- Output to a file or stdout
- Can just write the first or last *n* reads (like the unix head and tail commands)

BIOH5G_IMPORT_ALIGNMENTS

This tool imports alignment data from a SAM file into a BioHDF file.

- Input from a file or stdin

BIOH5G_EXPORT_ALIGNMENTS

This tools exports alignment data from a BioHDF file.

- SAM format only at this time
- Output to a file or stdout
- First or last *n* alignments in the named storage location (like the unix head and tail commands). NOTE: using first or last *n* alignments overrides the range filters and MAPQ/FLAGS filters.
- Ranges can be specified with <reference name>:<start>:<end> triplets on the command line. More than one range can be specified at a time.
- Can specify a minimum MAPQ value and SAM FLAGS mask.

BIOH5G_PREPARSE_SAM

This tool scans a SAM file to determine the widths of the string fields. These widths can be passed to bioh5g_import_alignments on the command line for more efficient data storage.

BASIC USE

IMPORTING SAM DATA INTO BIOHDF

1) Decide on a data organization scheme.

BioHDF uses "groups" to organize data in a BioHDF file. They work like directories in a filesystem. See the users guide for a more in-depth discussion of this.

If you don't care, name your reads "/reads/" and your alignments "/alignments/".

2) Determine the lengths of your input strings.

Storing variable-length data efficiently has always been a problem for data storage and BioHDF is no different. When dealing with variable-length data such as CIGAR strings and SAM tags, there are two ways to store the data.

1. Tell BioHDF the maximum possible length of the string. This is very efficient, even for highly variable data, but you have to be careful not to specify a string length that is too short or you can truncate data. BioHDF includes a bioh5g_preparse_sam tool that can scan a SAM file and give you the maximum string lengths.
2. Let BioHDF use a wildly inefficient, yet very safe, method to store any length string. We're not kidding when we say that it's slow, either. It's reeeeaaaaallllyyy slow. And big, too. Did I mention big?
3. (FUTURE) In the future, we plan to have a hybrid of 1 and 2 that combines the advantages of both so you can not care AND get good performance.

If you don't care, don't specify any lengths and the tools will default to the slow method. I wouldn't use this on a file larger than a few tens of MB, however, unless you have a lot of spare time.

Ideally, you want to specify as many lengths as you can. If you are accepting data from a stream it might be hard to do this, but you can probably at least determine the lengths of the FASTQ data.

3) Import the SAM data using bioh5g_import_alignments

You can import from either a file or the standard input stream. There is no need to use the bioh5g_import_reads tool as the reads will be imported and stored as the SAM file/stream is parsed.

Here is the shortest possible command to import some alignments from a SAM file:

```
./bioh5g_import_alignments –f my_biohdf_file.h5 –i some_sam_data.sam –A
"/some_aligns/" –R "/some_reads/"
```

Again, please note that this does not specify any string data lengths so it will make a large file and take a lot longer than you might like.

EXPORTING SAM DATA FROM BIOHDF

1) Export the SAM data using bioh5g_import_alignments

This tool emits SAM data to a file or stream. We support several data filters (reference regions, MAPQ, SAM flag mask). See the users guide for details.

Here is the shortest possible command to export some alignments to stdout:

```
./bioh5g_export_alignments –f my_biohdf_file.h5 –A "/some_aligns/"
```

FOR MORE INFORMATION…

Hopefully you now have working BioHDF software. The users guide includes more information about BioHDF in general and a guide to the tools. The API reference and internals document are still under construction, but there is a /doc/api directory which includes documentation for the BioHDF API calls. SWIG wrappers should be available in the future so you can explore the API via your favorite non-C programming language.

PLEASE COMPLAIN!

If *anything* we are doing is broken, hard to understand, doesn't perform well or awkward to use, please let me know! We need your feedback to make the best software possible for the NGS community and your complaints and suggestions will be invaluable.