

# BIOHDF – GETTING STARTED

VERSION 0.3 ALPHA

*MARCH 2011*

Hello everyone!

This is the BioHDF 0.3 alpha distribution. It includes many bugfixes and enhancements over version 0.2a. It is still largely unoptimized but we now have a fairly solid framework so the next versions of the software should be faster.

## BUILDING AND INSTALLING ON LINUX

The code builds and runs properly on 64-bit Linux (tested on Fedora 13 and Ubuntu 10.04 and 11.04). BioHDF has not been tested by us on 32-bit operating systems though I know of no reason why it wouldn't work. BioHDF is only distributed in source form, so in order to build BioHDF, you will obviously need a standard GNU build system, including gcc, etc.

I have Visual Studio 2008 solutions and projects for the BioHDF library and tools which are available on request. They will probably be released in the next version of the software.

---

## INSTALLING HDF5

If HDF5 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. In particular, the Ubuntu distribution is built with certain backwards-compatibility modes turned on that make it awkward to use those binaries with BioHDF.

The first thing you will need is HDF5. Go to <http://www.hdfgroup.org/HDF5/release/obtain5.html> and download the appropriate STATIC library binaries for your system. Using static libraries will produce larger executables than using shared libraries would but it's much easier to build the software. The easiest way to install the libraries is to pick a spot in your home directory for the library and just copy them to that location. For example, I created a directory named HDF5 in my home directory and then moved all the distribution files into a subdirectory named static/.

---

## TWEAKING YOUR MAKEFILE

We don't use a configure script, so you'll have to tweak the makefile yourself. There should just be two things you need to change, both of which are right at the top - HDF5\_INC\_PATH and HDF5\_LIB\_PATH need to point to the location of your header files and library files. If you used my scheme above, you should just have to replace /path/to in the paths with /home/<username>. *e.g.* /home/bobsmith/HDF5/static/include.

---

## BUILDING BIOHDF

Once you have your makefile configured properly, building BioHDF should be as easy as typing 'make' in the source directory. The makefile is not particularly sophisticated as we plan to transition to CMake in the near future.

'make all' will build a few extra test programs that we use to test some of the BioHDF parts. To run these programs, run the test\_biohdf.sh script located in the /test directory. Any errors or file differences will appear on the screen and in the .out output files (also deposited in the /test directory).

## WHAT YOU GET

Building BioHDF will create several tools and a static library:

- libbiohdf.a
- bioh5g\_import\_reads
- bioh5g\_export\_reads
- bioh5g\_import\_alignment\_hits
- bioh5g\_export\_alignment\_hits
- bioh5g\_preparse\_sam

The tools are 100% statically linked and can be freely moved around.

---

## LIBBIOHDF.A

This is our static 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.

---

## 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 (unlike the prototype).*

---

## 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.

## 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 reeeeeaaaaaalllllyyy 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  
"/alignments/" -R "/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 "/alignments/"
```

## 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 shortly 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.