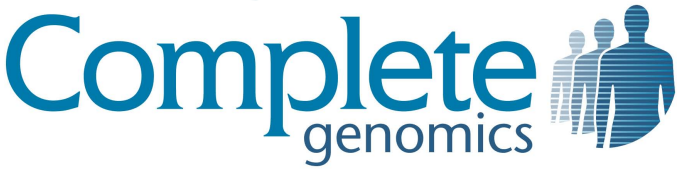

cgatools Installation Guide

Version 1.4.0

Table Of Contents

Preface

This document contains the installation instructions for **cgatools** from Complete Genomics, Inc (CGI).

Conventions

This document uses the following notational conventions:

Notation	Description
42$ `echo hello world`	Indicates that the reader type "echo hello world" (without quotes) at the command prompt. The steps are numbered (bold number before the prompt) sequentially through the document.

cgatools Documents

This document is part of a set that describe aspects of **cgatools** and the Complete Genomics human genome data:

- *cgatools Methods* – Describes the motivation and design decisions for **cgatools**, the open source project to provide tools for downstream analysis of Complete Genomics data.
- *Complete Genomics Data File Formats* – Describes the organization and content of the format used to deliver Complete Genomics human genome data.
- *Variation Data Frequently Asked Questions* – Provides detailed answers to questions about Complete Genomics variation and evidence files.
- *Getting Started Frequently Asked Questions* – Describes how to prepare for receiving Complete Genomics data files.

For more information about **cgatools**, see the Complete Genomics website:

http://www.completegenomics.com/sequence-data/cgatools/

References

cgatools requires users to be familiar with Unix-like commands. Here are some basic resources to get you started or enhance your skills:

- A Brief UNIX Shell Comparison: http://www.thewellroundedgeek.com/2007/04/brief-unix-shell-comparison.html
- An alphabetical list of basic Unix/Linux commands with thorough examples: http://www.math.utah.edu/lab/unix/unix-commands.html
- A complete list of Linux commands and their options: http://www.oreillynet.com/linux/cmd/

Installing **cgatools** requires users to have installed several Unix utilities. See the resources below for downloads and more information. This installation guide provides step-by-step instructions for using the utilities to install **cgatools**.

- **CMake**: An extensible, open-source system that manages the build process in an operating system and in a compiler-independent manner. Here's an overview of the utility and its operation: http://www.cmake.org/Wiki/CMake.
- **Boost** C++ libraries: This freely downloaded, peer-reviewed set of C++ libraries provides the core C++ functionality on which the **cgatools** source code is built. You will not need to

interact with Boost other than to make sure it is installed and available during the **cgatools** installation process. Here's where to find out more: http://www.boost.org/users/faq.html.

For detailed documentation and installation instructions, see: http://www.boost.org/doc/libs/1_42_0/more/getting_started/unix-variants.html.

- **Python** interpreter: This open source programming language gives users of CGI data a powerful analysis platform. You won't need to interact with Python for this installation, but you may find that you want to use it to expand your data analysis capabilities. Here's where to get started with Python: http://docs.python.org/tutorial/.

Overview and Requirements

cgatools is an open source project to provide tools for downstream analysis of Complete Genomics data. **cgatools** is distributed as source files that can easily be compiled and run on a Linux or Unix desktop or server or as compiled binaries for Linux or Mac OS X.

Before you install **cgatools**, make sure you have the following requirements:

Install Process in a Nutshell

You can install the **cgatools** software on 64-bit Linux or Mac OS X systems from pre-compiled binary distributions provided by Complete Genomics. This document provides detailed instructions for these high-level steps:

1. Prepare your environment.
2. Install **cgatools** from the binary distribution.
3. Create a copy of the reference human genome formatted for use with **cgatools.**

Alternatively, you can build **cgatools** from source code. This document provides detailed instructions for these high-level steps:

1. Prepare your environment.
2. Verify that you have the required third-party software.
3. Build and install CMake.
4. Build and install Boost.
5. Use CMake to build and install **cgatools**.
6. Create a copy of the reference human genome formatted for use with **cgatools.**

User Requirements

- Command-line Unix skills

System Requirements

- CentOS 5.1 (64-bit) Linux or equivalent.

  This package works on 32-bit or 64-bit Linux and Unix-like systems, including Mac OS X (via the terminal program). However, CGI has formally tested only the CentOS 64-bit environment. It has not been tested on Microsoft Windows.
- 2GB RAM available
- Disk usage proportional to the data.

  Typically, you will need 30-50 GB disk space per genome when working with CGI variation and evidence data, reference data, and **cgatools** output from one or more tools. However, for the ST001RM product option (genomes with reads and initial mappings included), your CGI data may be 400 GB or more per genome. Depending on what you intend to do with the data, you will need even more disk space than this. For example, if you intend to convert the initial mapping data to SAM and BAM format, you need more than 2 TB of space, some of which is taken up by intermediate files (the SAM files) that can be deleted after the conversion.

Software Requirements to Build cgatools from Source Code

- GCC C++ compiler, version 4.1.2 or later
- CMake version 2.8.1 or later
- Boost version 1.42
- Python version 2.4.3 or later (only required if running test cases)
- Doxygen version 1.6.2 or later (only required if generating API docs)

Note: You may have older versions of some of these software programs installed, or they may be installed in a manner inconvenient for the **cgatools** build process. If you encounter error messages during the **cgatools** build process, we recommend you download and install fresh versions of CMake and Boost. Consider installing them in a private directory separate from the other version on your computer.

Preparing the Environment

1. Create a directory structure.

   We recommend the following directory structure to support the installation, build, and use of **cgatools**. This structure assumes your username is "cgi" and all CGI data and applications are installed by user "cgi".

   `/home/cgi/src`

   Tar files and other intermediate files that are not needed after the **cgatools** installation is complete.

   `/home/cgi/local/bin`

   The location of the **cgatools** executable as well as commands used by **cgatools** build process, including CMake.

   `/home/cgi/local/bin/cgatools`

   The final directory for the **cgatools** executable.

   `/home/cgi/local/share/cgatools-X.X.X/doc`

   The final location of **cgatools** documentation. For example, `/home/cgi/local/share/cgatools-1.1.0/doc`.

   `/home/cgi/ref`

   Reference human genome sequence files.

   `/home/cgi/data`

   Your CGI human genome data files.

   The installation instructions assume this directory structure. If you choose to create a variation on this structure, make sure to adapt the installation instructions below to your structure.

2. Be sure that the **cgatools** `/bin` directory (here, `/home/cgi/local/bin`) is in your $PATH.

   Consult your shell documentation for how to set the $PATH in your `.tcshrc` or `.bashrc` file.

Installing cgatools from a Binary Distribution

cgatools is available as a pre-compiled binary distribution for 64-bit Linux or for Mac OS X:

1. Download the **cgatools** binary distribution into `/home/cgi/src`.

   The **cgatools** binary distribution is named as follows, where the X's are replaced by the current version number:

   Linux: `cgatools-X.X.X.X-linux-x86_64.tar.gz`

   Mac OS X: `cgatools-X.X.X.X-darwin-x86_64.tar.gz`

   They are available here:

   http://sourceforge.net/projects/cgatools/files/

2. Change to the download directory.

   ```
   B1$ cd /home/cgi/src
   ```

3. Untar the tarball:

   Linux:
   ```
   B2$ tar xzf cgatools-X.X.X.X-linux-x86_64.tar.gz
   ```

   Mac OS X:
   ```
   B2$ tar xzf cgatools-X.X.X.X-darwin-x86_64.tar.gz
   ```

   This command creates a directory including **cgatools** executable and documents:

   - Executable: `/bin/cgatools`
   - Documentation: `/share/cgatools-X.X.X/doc/index.html`

4. Copy the executable and documentation into your binary directory:

   Linux:
   ```
   B3$ cp cgatools-X.X.X.X-linux-x86_64/bin/cgatools /home/cgi/bin
   ```

   Mac OS X:
   ```
   B3$ cp cgatools-X.X.X.X-darwin-x86_64/bin/cgatools /home/cgi/bin
   ```

   Alternatively, if you are installing **cgatools** into a Linux system directory which requires sudo (that is, root privileges) you will need to type:

   ```
   B3$ sudo cp cgatools-X.X.X.X-linux-x86_64/bin/cgatools /usr/local/bin
   ```

5. Make sure the new commands are available to the shell.

   For C-shell (`csh, tcsh`):

   ```
   B4$ rehash
   ```

   For Bash or Bourne shell:

   ```
   B4$ hash –r
   ```

6. Test the install:

   ```
   B5$ cgatools
   ```

   If **cgatools** returns with the version number (for example "`1.1.0.0`") and a page of help notes, you have successfully installed **cgatools**.

At this point you can skip ahead to "Obtaining a Reference Human Genome for Use with cgatools."

Installing CMake

If you already have a correct version of CMake installed on your system, make sure the command is in your $PATH and skip to the next section "Installing Boost."

Note: Precompiled binaries are available for CMake. If you choose to use a CMake binary as opposed to build from source, make sure its machine architecture is the same as used for Boost. For example, 64-bit CMake should only be used with 64-bit Boost.

To build and install CMake from source:

1. Open a Lunix/Unix command shell.
2. Download the CMake distribution into `/home/cgi/src`.

   http://www.cmake.org/cmake/resources/software.html
3. Change to the download directory.

   ```
   1$ cd /home/cgi/src
   ```
4. Unpack the tarball:

   ```
   2$ tar -xvf cmake-2.8.1.tar.gz
   ```

   This creates a `cmake-2.8.1` subdirectory.
5. Change to the CMake directory.

   ```
   3$ cd cmake-2.8.1
   ```
6. Configure the software, specifying the final installation target:

   ```
   4$ ./bootstrap --prefix=/home/cgi/local
   ```

   Resolve any errors, such as not having GNU C Compiler Collection installed. See "Software Requirements."
7. Build the software.

   ```
   5$ gmake
   ```

   This takes a few minutes. Resolve any errors.
8. Install the software.

   ```
   6$ make install
   ```

   Alternatively, if you are installing CMake into a system directory which requires sudo (that is, root privileges) you will need to type:

   ```
   6$ sudo make install
   ```
9. Make sure the new commands are available to the shell.

   For C-shell (`csh, tcsh`):

   ```
   7$ rehash
   ```

   For Bash or Bourne shell:

   ```
   7$ hash -r
   ```
10. Test the installation.

    ```
    8$ cmake –-help
    ```

    If the `cmake` command prints the correct version number and a page of help notes, you have successfully installed CMake and your path variable is set correctly.

Installing Boost

To build and install Boost from source:

1. Download the Boost distribution into `/home/cgi/src`.

   http://sourceforge.net/projects/boost/files/boost
2. Change to the download directory.

   ```
   9$ cd /home/cgi/src
   ```
3. Unpack the tarball:

   ```
   10$ tar -xvf boost_1_42_0.tgz
   ```

   This will take several minutes. This creates a `boost_1_42_0` subdirectory.
4. Change to the Boost directory:

   ```
   11$ cd boost_1_42_0
   ```
5. Configure the software, specifying the final installation target:

   ```
   12$ ./bootstrap.sh --prefix=/home/cgi/local
   ```
6. Now build the software:

   ```
   13$ ./bjam install
   ```

   Alternatively, if you are installing CMake into a system directory which requires sudo (that is, root privileges) you will need to type:

   ```
   13$ sudo ./bjam install
   ```

   This step will take a long time (perhaps an hour). You will see many `libboost*` files appear in `/home/cgi/local/lib` and many `*.hpp` files in `/home/cgi/local/include`.

   It is not uncommon to see some errors in the Boost install.

Installing cgatools from Source Code

To build and install **cgatools**:

1. Download the **cgatools** distribution into `/home/cgi/src`.

   The **cgatools** distribution has a name such as `cgatools-X.X.X.X-source.tar.gz` where the X's are replaced by the current version number. It is available here:

   http://sourceforge.net/projects/cgatools/files/

2. Change to the download directory.

   ```
   14$ cd /home/cgi/src
   ```

3. Unpack the tarball:

   ```
   15$ tar -xvf cgatools-X.X.X.X-source.tar.gz
   ```

4. Change to the `cgatools-X.X.X.X-source` directory:

   ```
   16$ cd cgatools-X.X.X.X-source
   ```

5. Create an empty directory in which to build **cgatools** (`build` for example).

   ```
   17$ mkdir build
   ```

6. Change to the `build` directory:

   ```
   18$ cd build
   ```

7. Configure **cgatools**.

   You can type this next command on one line or use \ (backslash) to split it onto multiple lines, as shown here.

   ```
   19$ cmake -DBOOST_ROOT=/home/cgi/local \
   -DCMAKE_INSTALL_PREFIX=/home/cgi/local/ \
   -DCMAKE_BUILD_TYPE=Release \
   /home/cgi/src/cgatools-X.X.X.X-source
   ```

   where the `BOOST_ROOT` flag points to the final installation target for Boost.

8. Compile **cgatools**.

   ```
   20$ make -j8
   ```

9. Tests **cgatools**.

   This step uses Python to run the 75 **cgatools** test cases.

   ```
   21$ ctest -j8
   ```

   If any of the tests fail, investigate the failure before proceeding to the next step.

10. Install **cgatools**.

    ```
    22$ make -j8 install
    ```

    Alternatively, if you are installing **cgatools** into a system directory which requires sudo (that is, root privileges) you will need to type:

    ```
    22$ sudo make -j8 install
    ```

11. Make sure the new commands are available to the shell.

    For C-shell (`csh, tcsh`):

    ```
    23$ rehash
    ```

    For Bash or Bourne shell:

    ```
    23$ hash -r
    ```


12. Test the install:
    ```
    24$ cgatools
    ```
    If **cgatools** returns with the version number (for example "`1.1.0.0`") and a page of help notes, you have successfully installed **cgatools**.

    If you are ultimately unable to install **cgatools** and are warned that there are problems with the Boost libraries, you may need to consult a system administrator for help.
13. Locate the documentation for **cgatools**:
    `/home/cgi/local/share/cgatools-X.X.X/doc/index.html`

Obtaining a Reference Human Genome for Use with cgatools

Complete Genomics supports two references. The first, which we refer to as "build 36," consists of the assembled nuclear chromosomes from NCBI build 36 (not unplaced or alternate loci) plus Yoruban mitochondrion NC_001807.4. This assembly is also known as UCSC hg18. The second reference, which we refer to as "build 37," consists of the assembled nuclear chromosomes from GRCh37 (not unplaced or alternate loci), plus the Cambridge Reference Sequence for the mitochondrion (NC_012920.1). This assembly (though with an alternate mitochondrial sequence) is also known as UCSC hg19.

cgatools uses a compact representation of the human genome in a specialized CRR (Compact Randomly accessible Reference) format. There are two ways to obtain the CRR file associated with your build.

1. Downloading the CRR File from the Complete Genomics FTP site.
2. Building the CRR File from FASTA Files that you download from the Complete Genomics FTP site.

In each case, you will want to verify the content of the file, as described in "Verifying CRR File Content."

Downloading the CRR File

1. Download the CRR files into `/home/cgi/src`:

   The CRR files are available here:

   ftp://ftp.completegenomics.com/ReferenceFiles/build36.crr

   ftp://ftp.completegenomics.com/ReferenceFiles/build37.crr
2. Unzip the CRR file.

   ```
   $ bunzip2 build36.crr.bz2
   ```

Next, verify the CCR file content, as described in "Verifying CRR File Content."

Building the CRR File from FASTA Files

To build the reference human genome:

1. Download the FASTA files into `/home/cgi/src`:

   The FASTA sequences are available here:

   ftp://ftp.completegenomics.com/ReferenceFiles/build36.fa.bz2

   ftp://ftp.completegenomics.com/ReferenceFiles/build37.fa.bz2
2. Change to the download directory.

   ```
   $ cd /home/cgi/src
   ```
3. Perform the format conversion, making the CRR file. For example:

   ```
   $ cgatools fasta2crr --input build36.fa.bz2 --output build36.crr
   ```

Next, verify the CCR file content, as described in "Verifying CRR File Content."

Verifying CRR File Content

1. List the contents of the CRR file. For example:

   ```
   $ cgatools listcrr --reference build36.crr
   ```

   For build 36 CRR files, the output should be identical to the following:

```
ChromosomeId Chromosome    Length Circular Md5
           0       chr1 247249719    false 9ebc6df9496613f373e73396d5b3b6b6
           1       chr2 242951149    false b12c7373e3882120332983be99aeb18d
           2       chr3 199501827    false 0e48ed7f305877f66e6fd4addbae2b9a
           3       chr4 191273063    false cf37020337904229dca8401907b626c2
           4       chr5 180857866    false 031c851664e31b2c17337fd6f9004858
           5       chr6 170899992    false bfe8005c536131276d448ead33f1b583
           6       chr7 158821424    false 74239c5ceee3b28f0038123d958114cb
           7       chr8 146274826    false 1eb00fe1ce26ce6701d2cd75c35b5ccb
           8       chr9 140273252    false ea244473e525dde0393d353ef94f974b
           9      chr10 135374737    false 4ca41bf2d7d33578d2cd7ee9411e1533
          10      chr11 134452384    false 425ba5eb6c95b60bafbf2874493a56c3
          11      chr12 132349534    false d17d70060c56b4578fa570117bf19716
          12      chr13 114142980    false c4f3084a20380a373bbbdb9ae30da587
          13      chr14 106368585    false c1ff5d44683831e9c7c1db23f93fbb45
          14      chr15 100338915    false 5cd9622c459fe0a276b27f6ac06116d8
          15      chr16  88827254    false 3e81884229e8dc6b7f258169ec8da246
          16      chr17  78774742    false 2a5c95ed99c5298bb107f313c7044588
          17      chr18  76117153    false 3d11df432bcdc1407835d5ef2ce62634
          18      chr19  63811651    false 2f1a59077cfad51df907ac25723bff28
          19      chr20  62435964    false f126cdf8a6e0c7f379d618ff66beb2da
          20      chr21  46944323    false f1b74b7f9f4cdbaeb6832ee86cb426c6
          21      chr22  49691432    false 2041e6a0c914b48dd537922cca63acb8
          22       chrX 154913754    false d7e626c80ad172a4d7c95aadb94d9040
          23       chrY  57772954    false 62f69d0e82a12af74bad85e2e4a8bd91
          24       chrM     16571     true d2ed829b8a1628d16cbeee88e88e39eb
```

   For build 37 CRR files, the output should be identical to the following:

```
ChromosomeId Chromosome    Length Circular Md5
           0       chr1 249250621    false 1b22b98cdeb4a9304cb5d48026a85128
           1       chr2 243199373    false a0d9851da00400dec1098a9255ac712e
           2       chr3 198022430    false 641e4338fa8d52a5b781bd2a2c08d3c3
           3       chr4 191154276    false 23dccd106897542ad87d2765d28a19a1
           4       chr5 180915260    false 0740173db9ffd264d728f32784845cd7
           5       chr6 171115067    false 1d3a93a248d92a729ee764823acbbc6b
           6       chr7 159138663    false 618366e953d6aaad97dbe4777c29375e
           7       chr8 146364022    false 96f514a9929e410c6651697bded59aec
           8       chr9 141213431    false 3e273117f15e0a400f01055d9f393768
           9      chr10 135534747    false 988c28e000e84c26d552359af1ea2e1d
          10      chr11 135006516    false 98c59049a2df285c76ffb1c6db8f8b96
          11      chr12 133851895    false 51851ac0e1a115847ad36449b0015864
          12      chr13 115169878    false 283f8d7892baa81b510a015719ca7b0b
          13      chr14 107349540    false 98f3cae32b2a2e9524bc19813927542e
          14      chr15 102531392    false e5645a794a8238215b2cd77acb95a078
          15      chr16  90354753    false fc9b1a7b42b97a864f56b348b06095e6
          16      chr17  81195210    false 351f64d4f4f9ddd45b35336ad97aa6de
          17      chr18  78077248    false b15d4b2d29dde9d3e4f93d1d0f2cbc9c
          18      chr19  59128983    false 1aacd71f30db8e561810913e0b72636d
          19      chr20  63025520    false 0dec9660ec1efaaf33281c0d5ea2560f
          20      chr21  48129895    false 2979a6085bfe28e3ad6f552f361ed74d
          21      chr22  51304566    false a718acaa6135fdca8357d5bfe94211dd
          22       chrX 155270560    false 7e0e2e580297b7764e31dbc80c2540dd
          23       chrY  59373566    false 1e86411d73e6f00a10590f976be01623
          24       chrM     16569     true c68f52674c9fb33aef52dcf399755519
```

2. Copy the new reference to a more useful location. For example:

```
$ mkdir /home/cgi/ref
$ mv build36.crr /home/cgi/ref
```

   If the `cgi` directories require root privileges, you need to use super-user privileges to perform the move:

```
$ sudo mkdir /home/cgi/ref
$ sudo mv build36.crr /home/cgi/ref
```

3. Test your reference file by extracting a sequence from the reference genome based on user-defined coordinates. For example:

```
$ cgatools decodecrr \
--reference /home/cgi/ref/build36.crr \
--range chr16:10000000-10000050
```

Congratulations. You are now ready to use **cgatools**!