Linux for Beginners – Part 1

3CPG Workshop

Robert Bukowski
Computational Biology Service Unit

BioHPC infrastructure at CBSU

BioHPC Web Computing Resources

- Compute clusters
- Data storage
- Web server

CBSU/3CPG Lab

- cbsuwrkst1 (Windows)
- cbsuwrkst2,3,4 (Linux)
- cbsum1c1b00n (Linux)
- cbsum1c2b00n (Linux)
- 31 “remote” machines
- cbsulm01,cbsulm02 (Linux, 64 and 500 GB RAM)

Sequencing reads from Cornell sequencing facility

http://www.cbsuapps.tc.cornell.edu/

Your client machine

Job submission (via web pages)

File transfer (via web pages)

File transfer (via sftp client)

Commands, scripts, programs (via ssh client)

Topic of today’s workshop
BioHPC infrastructure at CBSU

BioHPC Web Computing Resources

- Jobs submitted to clusters by filling out a web form and hitting “Submit” button
- It is a batch system (submit and wait for email with links to results – not much interactivity)
- Coupled to the BioHPC Data Store – easy access to large files
- Simple to use – no knowledge of OS necessary – all details taken care of by programs running “behind the scenes”
- Limited number of Next-Gen applications available; web submission interfaces nontrivial to create – often not flexible enough

3CPG Lab workstations

- Fully interactive work through terminal window (remotely – through ssh and sftp client programs)
- A lot of applications available (or easy to install) – custom pipelines may be created for any project
- Need to know Linux to do anything at all
- Useful to know scripting to do more complicated things
Part 1: (today)
- Reserving time on 3CPG Lab workstations
- Logging in to a Linux workstation
- Terminal window and tricks
- Linux directory structure
- Working with files
- Working with text files
- Trying it (machines reserved for you until end of Wednesday, Nov. 9)

Part 2: (Nov 14, 2011)
- Transferring files to/from workstations
- Running applications
  - Note: this will only cover the Linux aspect of running applications; the functionality and the biological aspect will be covered in workshop Using BioHPC Lab Software on Nov 28, 2011.
- Basics of scripting (shell and Perl)
  - Note: this will not teach you scripting – just get you started. We are planning a series of workshops on Perl in the fall – stay tuned. In the meantime - use multiple resources online (google “Perl tutorial”, for example).
Mouse over the CBSU/3CPG BioHPC Lab tab, then click BioHPC Lab Login and log in with your lab user ID (typically – the same as your Cornell NetID) and lab password

- Initial lab password is sent to you in a “welcome e-mail” upon account creation

Check out other links under the CBSU/3CPG BioHPC Lab tab (some do not require a login)

- Software: summary info page about the software programs installed on the workstations
- Reservations: reserve time slot on any of the lab workstations
- User’s Guide: learn more about the lab and how to use its resources
- Forum: post a question or suggestion to the community of users and admins of the 3CPG Lab (requires registration)
- Office hours: need specific advice about your project? Talk to an expert on duty – here you will find out when they are available.
To change your lab password:

- From any computer on the network, go to http://cbsu.tc.cornell.edu.
- If not yet logged in, click on CBSU/3CPG BioHPC Lab tab, then BioHPC Lab Login, and log in with your old password.
- Click on CBSU/3CPG BioHPC Lab tab, then on User <userID> password (where <userID> will be your Lab user ID) – this will take you to the password change dialog.

Forgot your lab password? Reset it:

- From any computer on the network, go to http://cbsu.tc.cornell.edu.
- Click on CBSU/3CPG BioHPC Lab tab, then on Lab User Password Reset – this will take you to the password reset dialog: provide your lab user ID, and the new password will be e-mailed to you.

Password change will be effective immediately both on the Linux and Windows machines as well as on the Lab web page (including Reservation page).

To update your Lab user profile:

- From any computer on the network, go to http://cbsu.tc.cornell.edu and log in
- Click on CBSU/3CPG BioHPC Lab tab, then on Lab <userID> profile – this will take you to the profile update dialog. When done making changes, click “Submit”.
Reserving time on a workstation
(no need to do it for the workshop!)

- From any computer on the network, go to http://cbsu.tc.cornell.edu/lab/lab.aspx
- Click on CBSU/3CPG BioHPC Lab tab -> Reservations
- If not yet logged in, you will be asked for your lab user ID (typically same as NetID) and the lab password
- Make a reservation
  - Note that only cbsuwrkst1 is a Windows machine, all others are running Linux (CentOS 5.5)
  - If your reservation ends while you are still logged in, you may continue working. However, you will be disconnected once another user with valid reservation attempts to log in. To prevent this, go to the reservation page and extend the reservation, if possible.
  - Cancel reservations you no longer need (click on “X” near your reservation)
Logging in to a Linux workstation
(lab machines only)

Remotely through VNC connection:
- In web browser, navigate to [http://cbsu.tc.cornell.edu/lab/lab.aspx](http://cbsu.tc.cornell.edu/lab/lab.aspx), log in (if not yet logged in), click on CBSU/3CPG BioHPC Lab, select tab My Reservations

<table>
<thead>
<tr>
<th>Hour</th>
<th>Start</th>
<th>End</th>
<th>Computer</th>
<th>LCC System info</th>
<th>Other users</th>
<th>Action</th>
<th>VNC port #</th>
</tr>
</thead>
<tbody>
<tr>
<td>1020</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
<tr>
<td>1042</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
<tr>
<td>1045</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
<tr>
<td>1044</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
<tr>
<td>1045</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
<tr>
<td>1044</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
<tr>
<td>1047</td>
<td>11/2/2011 11:00:00 AM</td>
<td>11/2/2011 11:59:59 PM</td>
<td>dell1</td>
<td>PowerEdge M900</td>
<td>8 cores, 16GB RAM, 16TB HDD</td>
<td>login</td>
<td>Connect VNC</td>
</tr>
</tbody>
</table>

Click “Connect VNC”, allow Java applet to open, enter password when prompted
CBSU / 3CPG BioHPC Laboratory (625 Rhodes Hall)

VNC Connection

Everything is OK! Launching VNC connection

Pop-up windows cannot be opened!

Please disable popup blocker in your browser or add this site to trusted sites.

Otherwise you can use the two links below to open a new browser window with VNC applet (1st link) and navigate back to 'My Reservations' page (2nd link)

[open VNC applet window]
[go back to 'My Reservations']
Enter your Lab password, hit ENTER
Logging in to a Linux workstation
(lab machines only)
Logging in to a Linux machine
(applies to any Linux machine, not only Lab)

- Remotely from a PC via ssh (secure shell)
  - Install and configure remote access software (PuTTY, Xming). For details, consult http://cbsu.tc.cornell.edu/lab/doc/Remote_access.pdf
  - Use PuTTY to open a terminal window on the reserved workstation using ssh protocol;
  - You may open several terminal windows, if needed.
  - start Xming if you intend to run any graphical applications on a workstation (e.g., gsAssembler, Firefox, IGV,...)

- Remotely from Mac via ssh (secure shell)
  - Launch the Mac’s terminal window. Type ssh -Y bukowski@cbsuwrkstX.tc.cornell.edu (replace the “X” with the workstation that you just reserved, and “bukowski” with your own user ID). Enter the lab password when prompted.
  - You may open several terminal windows, if needed, and log in to the workstation from each of them.

- Directly from the console (consoles from cbsuwrkst2, 3, and 4 are in 625 Rhodes)
  - Turn on the monitor, then log in using the login screen
  - Right-click anywhere on the desktop, then select “Open terminal” – a terminal window will open
  - You can open additional terminals window the same way
Terminal window

- User communicates with the machine via **commands** typed in the terminal window
  - Typically, each command is typed in one line and “**entered**” by hitting the “**Enter**” key on the keyboard.
  - Commands may request information (e.g., list user’s files), launch a simple task (e.g., rename a file), or start an application (e.g., Firefox web browser, BWA aligner, IGV viewer, ...)

- Helpful tricks to avoid excessive command typing
  - Use **copy/paste**. Any text “mouse-selected” while holding the left mouse button is copied to clipboard. It may then be pasted, e.g., into a command, by clicking the **right** mouse button (PuTTy) or the **middle** button (when working through the console in 625 Rhodes).
  - Use **Up/Down arrow keys** – this will cycle through recently executed commands.
  - Use the **TAB key** – this will often present you with a list of choices after typing a part of a command – no need to remember everything.
Example of directory structure

Referring to files:
Full path: 
/home/bukowski/tst5/transcripts.gtf

Relative path (i.e., relative to /home/bukowski) 
tst5/transcripts.gtf

Relative path (i.e., relative to /home/bukowski/tst5) 
transcripts.gtf
Working with Directories

Right after logging in or opening a terminal window, you are in your **home directory** (e.g., /home/bukowski).

`pwd` *(print working directory)* – show the current directory

`cd`  
*Change (current) directory; without additional arguments, this command will take you to your home directory*

`cd ./`  
*Change (current) directory to the same one (i.e., do nothing). Note: ./ refers to the current directory.*

`cd /workdir/bukowski/indexes`  
*Change (current) directory from wherever to /workdir/bukowski/indexes.*

`cd indexes`  
*Change (current) directory to indexes (will work if the current directory contains “indexes”)*

`cd ../`  
*Change (current) directory one level back (closer to the root)*

`cd ../../../`  
*Change (current) directory three levels back (closer to the root)*

`mkdir /home/bukowski/my_new_dir`  
*Make a new directory called “my_new_dir” in /home/bukowski*

`mkdir my_new_dir`  
*Make a new directory called “my_new_dir” in the current directory*

`rm -Rf /home/bukowski/my_new_dir`  
*Remove directory called “my_new_dir” in /home/bukowski with all its content (i.e. all files and subdirectories will be gone)*

`rm -Rf my_new_dir`  
*Remove directory called “my_new_dir” in current directory with all its content (i.e. all files and subdirectories will be gone)*
Listing content (files and subdirectories) of a directory

`ls`

*(list)*

**`ls -al`**

*List all files and directories in current directory in long format*

**`ls -al /home/bukowski/tst`**

*List content of /home/bukowski/tst (which does not have to be the current directory)*

**`ls -alt`**

*Lists content of the current directory sorted according to modification time (use `ls -altr` to sort in reverse)*

**`ls -alS`**

*Lists content of the current directory sorted according to size (use `ls -alSr` to sort in reverse)*

**`ls -al | more`**

*Lists content of the current directory using pagination – useful if the file list is long (SPACE bar will take you to the next page)*

LOTS more options for `ls` – try `man ls` to see them all (may be intimidating).
Listing content of a directory

```bash
ls -al
```

<table>
<thead>
<tr>
<th>Owner and group</th>
<th>File name (directories in blue, executable files in green)</th>
<th>Size (in bytes)</th>
<th>Last modification time</th>
</tr>
</thead>
<tbody>
<tr>
<td>drwxr-xr-x 5</td>
<td>bukowski bukowski 4096 Dec 3 11:58 454</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxr-xr-x 5</td>
<td>bukowski bukowski 4096 Jan 6 11:30 454_2.5.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 3</td>
<td>bukowski bukowski 4096 Jan 6 11:30 bin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxr-xr-x 2</td>
<td>bukowski bukowski 4096 Nov 22 15:55 Desktop</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 4</td>
<td>bukowski bukowski 4096 Jan 26 13:49 ecoli_tst</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 2</td>
<td>bukowski bukowski 4096 Feb 18 17:25 GATK_tst</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 3</td>
<td>bukowski bukowski 4096 Dec 15 11:35 igv</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-rw-r--r-- 1</td>
<td>bukowski bukowski 3595 Jan 21 13:47 igv.log</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-rw-r--r-- 1</td>
<td>bukowski bukowski 401 Nov 24 11:07 perl_test.txt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 19</td>
<td>bukowski bukowski 4096 Feb 18 17:23 programs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-rw-r--r-- 1</td>
<td>bukowski bukowski 231 Dec 1 10:16 schedfile</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 2</td>
<td>bukowski bukowski 4096 Feb 1 11:26 stacks_tst</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 2</td>
<td>bukowski bukowski 4096 Dec 1 11:27 tst</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 6</td>
<td>bukowski bukowski 4096 Nov 29 14:22 tst2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 8</td>
<td>bukowski bukowski 4096 Nov 29 15:52 tst3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 6</td>
<td>bukowski bukowski 4096 Nov 29 15:41 tst4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 2</td>
<td>bukowski bukowski 4096 Dec 21 15:17 tst5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 2</td>
<td>bukowski bukowski 4096 Jan 17 17:14 tst_blat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drwxrwxr-x 3</td>
<td>bukowski bukowski 4096 Dec 22 10:56 tst_toxedo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-rw-rr--r-- 1</td>
<td>bukowski bukowski 106 Feb 2 10:08 ttt.pl</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

File permissions ("d" means this is a directory)
Local vs. network directories
(3CPG LAB – specific)

Network directories
/home, /programs, /shared_data
(with all subdirectories)
• Physically located on the file server
• Visible from all workstations
• Relatively SLOW access – DO NOT run any calculations there, avoid transferring large files there

Local directories:
/workdir (with all subdirectories), all other directories
• Physically attached to “its own” workstation
• Not visible from other workstations
• Fast access – all calculations should be run in /workdir
Working with files

File names
- Use only letters (upper- and lower-case), numbers from 0 to 9, a dot (.), and an underscore (_).
- Avoid other characters, as they may have special meaning to either Linux, or to the application you are trying to run. Do not use “space” in file names.
- Use of special characters in file names is possible if absolutely necessary, but will lead to problems if done incorrectly.
- Extensions (like “.zip”, “.gz”, “.ps”,...) are commonly used to denote the type of file, but are not necessary to “open” a file. While working in command line terminal you always explicitly specify a program which is supposed to work with this file.
- The dot (.) does not have any special meaning in Linux file names.
There are many types of files. Here are the most important:

- **Binary files** (cannot be viewed using a text editor)
  - Executables (e.g., samtools, bwa, bowtie)
  - Data in binary format (e.g., BAM files, index files for BWA or Bowtie, formatted BLAST databases)
  - Compressed files (usually *.

- **Text files** (can be viewed and modified using a text editor)
  - Text documents (e.g., README files)
  - Data in text format (e.g., FASTA, FASTQ, VCF, ...)
  - Scripts:
    - Shell scripts (usually *.
    - Perl scripts (usually *.
    - Python scripts (usually *.
    - ...

- **Symbolic links**: pointers to other files or directories.
  - In the example below, file /programs/bin/samtools/samtools is a symbolic link to /programs/samtools-0.1.11/samtools. Note the “l” character in the first column of output from “ls –al”.

```bash
$ cd /programs/bin/samtools
$ ls –al samtools
lrwxrwxrwx 1 root root 30 Nov 23 13:44 samtools -> ..../samtools-0.1.11/samtools
```
Working with files

Copying a file

cp  <source_file>  <destination_file>

Examples:

• cp sample_data.fa /workdir/bukowski/sample.fa (copy file sample_data.fa from the current directory to /workdir/bukowski and give the copy a name sample.fa; destination directory must exist)

• cp /workdir/bukowski/my_script.sh .  (copy file myscript.sh from /workdir/bukowski to the current directory under the same file name)

• cp /home/bukowski/*.*fastq /workdir/bukowski  (copy all files with file names ending with “.fastq” from /home/bukowski to /workdir/bukowski; destination directory must exist)

• cp –R /workdir/bukowski/tst5 /home/bukowski  (if tst5 is a directory, it will be copied with all its files and subdirectories to directory /home/bukowski/tst5; if /home/bukowski/tst5 did not exist, it will be created).

• Try man cp for all options to the cp command.
Working with files

Moving and renaming files

`mv <source_file> <destination_file>`

Examples:

- `mv my_file_one my_file_two` *(change the name of the file my_file_one in the current directory)*
- `mv my_file_one /workdir/bukowski` *(move the file my_file_one from the current directory to /workdir/bukowski without changing file name; the file will be removed from the current directory)*
- `mv /workdir/bukowski/my_file_two ./my_file_three` *(move the file my_file_two from /workdir/bukowski to the current directory changing the name to my_file_three; the file will be removed from /workdir/bukowski)*
- Try `man mv` for all options to the `mv` command....

Removing (deleting) files

`rm <file_name>`

Examples:

- `rm my_file_one` *(delete file my_file_one from the current directory)*
- `rm /workdir/bukowski/my_file_two` *(delete file my_file_two from directory /workdir/bukowski)*
- `rm -R ./tst5` *(if tst5 is a subdirectory in the current directory, it will be removed with all its files and directories)*
- Try `man rm` for all options to the `rm` command....
Working with files

Since there are no strict naming conventions for various file types, it is not always clear what kind of file we deal with. When in doubt, use the `file` command:

```bash
cd /programs/samtools-0.1.11
defile samtools
samtools: ELF 64-bit LSB executable, AMD x86-64, version 1 (SYSV), for GNU/Linux 2.6.9, dynamically linked (uses shared libs), for GNU/Linux 2.6.9, not stripped
```
Working with files

To save disk space, we can compress large files if we do not intend to use them for a while. A lot of files downloaded from the web are compressed and need to be uncompressed before any processing can take place.

Common compressed formats:

- **gzip (gz)**
  - `gzip my_file` *(compresses file my_file, producing its compressed version, my_file.gz)*
  - `gzip -d my_file.gz` *(decompress my_file.gz, producing its original version my_file)*

- **bzip2**
  - `bzip2 my_file` *(compresses file my_file, producing its compressed version, my_file.bz2)*
  - `bunzip2 my_file.bz2` *(decompress my_file.bz2, producing its original version my_file)*
Working with files

Common compressed formats (continued):

- **zip**
  - `zip my_file.zip my_file1 my_file2 my_file3` *(create a compressed archive called my_files.zip, containing three files: my_file1, my_file2, my_file3)*
  - `zip -r my_file.zip my_file1 my_dir` *(if my_dir is a directory, create an archive my_file.zip containing the file my_file1 and the directory my_dir with all its content)*
  - `zip -l my_file.zip` *(list contents of the zip archive my_file.zip)*
  - `unzip my_files.zip` *(decompress the archive into the constituent files and directories)*

- **tar**
  - `tar -cvf my_file.tar my_file1 my_file2 my_dir` *(create a compressed archive called my_files.tar, containing files my_file1, my_file2 and the directory my_dir with all its content)*
  - `tar -tvf my_file.tar` *(list contents of the tar archive my_file.tar)*
  - `tar -xvf my_files.tar` *(decompress the archive into the constituent files and directories)*
Working with files

Common compressed formats (continued):

- **tgz** (also, **tar.gz** – essentially a combo of “tar” and “gzip”)
  - `tar -czvf my_file.tgz my_file1 my_file2 my_dir` (create a compressed archive called my_files.tgz, containing files my_file1, my_file2 and the directory my_dir with all its content)
  - `tar –tzvf my_file.tgz` (list contents of the tar archive my_file.tar)
  - `tar -xzvf my_files.tgz` (decompress the archive into the constituent files and directories)

Compression works best (i.e., saves most disk space) for text files (e.g., large FASTQ files).

Getting help about compression tools:

- `gzip -h`, `bzip2 --help`, `zip`, `tar --help`
- `man gzip`, `man bzip2`, `man zip`, `man tar` (may be intimidating...)
File permissions

```
drwxrwxr-x 19 bukowski bukowski 4096 Feb 18 17:23 programs
-rw-rw-r-- 1 bukowski bukowski 231 Dec 1 10:16 schedfile
```

“d”: directory (or “-” if file); “r”: read permission; “w”: write permission; “x”: execute permission (or permission to “cd” if it is a directory); “-”: no permission

```
d r w x r w x r w x:    User (owner),  Group,  Others
```

Examples:

programs:
- is a directory ("d" in the first column)
- everybody can read and “cd” to it, but not write (“r-x” in the last three
- owner and everybody in his group can also write to it

schedfile:
- is a file readable by everybody and writable by owner and his group
- the file is not executable by anyone

ttt.pl:
- is a file readable by everybody, but writable and executable only by owner
Changing file permissions

**chmod** command – some examples

- `chmod o-rwx /home/bukowski` *make my home directory inaccessible to others*
- `chmod ug+x my_script.pl` *make the file my_script.pl (in the current directory) executable by the owner (“u”) and the members of the group (“g”).*
- `chmod a-w /workdir/bukowski/my_file` *deny the right to write to the file my_file (in /workdir/bukowski) – will prevent accidental deletion*
- Try **man chmod** for more information (may be somewhat intimidating!)

Want to make your files accessible to some (but not all) other users? **Contact us!**

- we would need to make sure that you and those other users are in the same user groups
Working with text files

Viewing text files
Examples:

- **more README.txt** *(display the content of the file README.txt in the current directory dividing the file into pages; press SPACE bar to go to the next page)*
- **head -100 my_reads.fastq** *(display first 100 lines of the file my_reads.fastq in the current directory)*
- **tail -100 my_reads.fastq** *(display last 100 lines of the file my_reads.fastq in the current directory)*
- **tail -1000 my_reads.fastq | more** *(extract the last 1000 lines of the file my_reads.fastq and display them page by page)*
- **head -1000 my_reads.fastq | tail -100** *(display lines 901 through 1000 of the file my_reads.fastq)*
- **cat my_reads.fastq** *(print the file on screen – good for small files)*
- **wc my_reads.fastq** *(display the number of lines, words, and characters in a file)*

Looking for a string in a group of text files:

- **grep “Error: lane” *.out** *(display all files *.out from the current directory which contain the string “Error: lane”; also display the lines containing that string)*
Editing text files

vi
- Available on all UNIX-like systems (Linux included), i.e., also on lab workstations (type vi or vi file_name)
- Free Windows implementation available (once you learn vi, you can just use one editor everywhere)
- Runs locally on Linux machine (no network transfers)
- User interface rather peculiar (no nice buttons to click, need to remember quite a few keyboard commands instead)
- Some love it, some hate it

nano
- Available on most Linux machines (our workstations included; type nano or nano file_name)
- Intuitive user interface. Keyboard commands-driven, but help always displayed on bottom bar (unlike in vi).
- Runs locally on Linux machine (no network transfers during editing)

geedit (installed on lab workstations; just type gedit or gedit file_name to invoke)
- X-windows application – need to have X-ming running on client PC.
- May be slow on slow networks...

edit+ (http://www.editplus.com/)
- Commercial product
- Runs on a local machine (laptop) and transfers data to/from Linux workstation as needed
- Can browse Linux directories in a Windows-like file explorer
- May be slow on slow networks
- Some people swear by it
vi basics

Opening a file:
vi my_reads.fastq (open the file my_reads.fastq in the current directory for editing; if the file does not exist, it will be created)

Command mode: typing will issue commands to the editor (rather than change text itself)
Edit mode: typing will enter/change text in the document

<Esc> exit edit mode and enter command mode (this is the most important key – use it whenever you are lost)

The following commands will take you to edit mode:
i enter insert mode
r single replace
R multiple replace
a move one character right and enter insert mode
o start a new line under current line
O start a new line above the current line

The following commands operate in command mode (hit <Esc> before using them)
x delete one character at cursor position
dd delete the current line
G go to end of file
1G go to beginning of file
154G go to line 154
$ go to end of line
1 go to beginning of line
:q! exit without saving
:w save (but not exit)
:wq! save and exit

Arrow keys: move cursor around (in both modes)
Disk usage guidelines
(3CPG lab specific)

Your home directory (e.g., /home/bukowski)
• Is network-mounted and therefore access to it is slow
• Visible from each workstation – no matter which one you log in to
• 200 GB quota will be imposed (may change depending on conditions)
• Use it to store files which you use frequently (reference genomes, index files) or which are small and hard to replace (scripts and executables)
• Never run any disk intensive applications (all Next-Gen tools are disk intensive) with your home directory (or any of its subdirectories) as the “current directory”. Work on /workdir instead.

The /workdir directory
• Is local to its workstation (located on disks physically attached to the machine’s controller)
• Not visible from other workstations
• Temporary – the content of /workdir may be erased after you log out. When you log in again, your files may be no longer there
• After you log in, create your own subdirectory in /workdir (if not already there)
• All the files to be used in processing have to be moved to that subdirectory
• Applications have to be started in that subdirectory
• Important output files have to be copied back to the home directory or (better yet) out of the machine.
Checking disk space

How much disk space is taken by my files?

**du –hs.** *(displays combined size of all files in the current directory and recursively in all its subdirectories)*

**du –h --max-depth=1.** *(as above, but sizes of each subdirectory are also displayed)*

How much disk space is available?

**df -h**

```
Filesystem   Size  Used  Avail  Use%  Mounted on
/dev/mapper/VolGroup00-LogVol000  3.6T  236G  3.1T    7%  /
/dev/sda1    99M   19M   75M    21%  /boot
tmpfs        12G    0  12G     0%  /dev/shm
cbsuwrkstfsrv1.tc.cornell.edu:/bigdisk/home  6.9T  246G  6.3T    4%  /home
cbsuwrkstfsrv1.tc.cornell.edu:/bigdisk/programs  6.9T  246G  6.3T    4%  /programs
cbsuwrkstfsrv1.tc.cornell.edu:/bigdisk/shared_data  6.9T  246G  6.3T    4%  /shared_data
```

/workdir is a part of it