Linux command line basics II and introduction to Project I

Yanbin Yin

Spring 2013

Homework #7

Create a folder under your home called homework7

Check NCBI website to learn about the Gene database and the Taxonomy database (what is the tax id of homo sapiens?)

Go to NCBI ftp site, find and download the correct file and use shell command line to find out: (wget or lftp)

- How many homo sapiens (using human taxid) genes are there (awk, cut, sort, wc)
- The top 10 genes with the largest number of pubmed papers (awk, cut, sort, uniq -c, sort)
- Search the top 5 genes at NCBI website using the Gene id as query to see what are these genes
- Save your command history as a file called hist.hw7

Write a report (in word or ppt) to include all the operations/commands and screen shots.

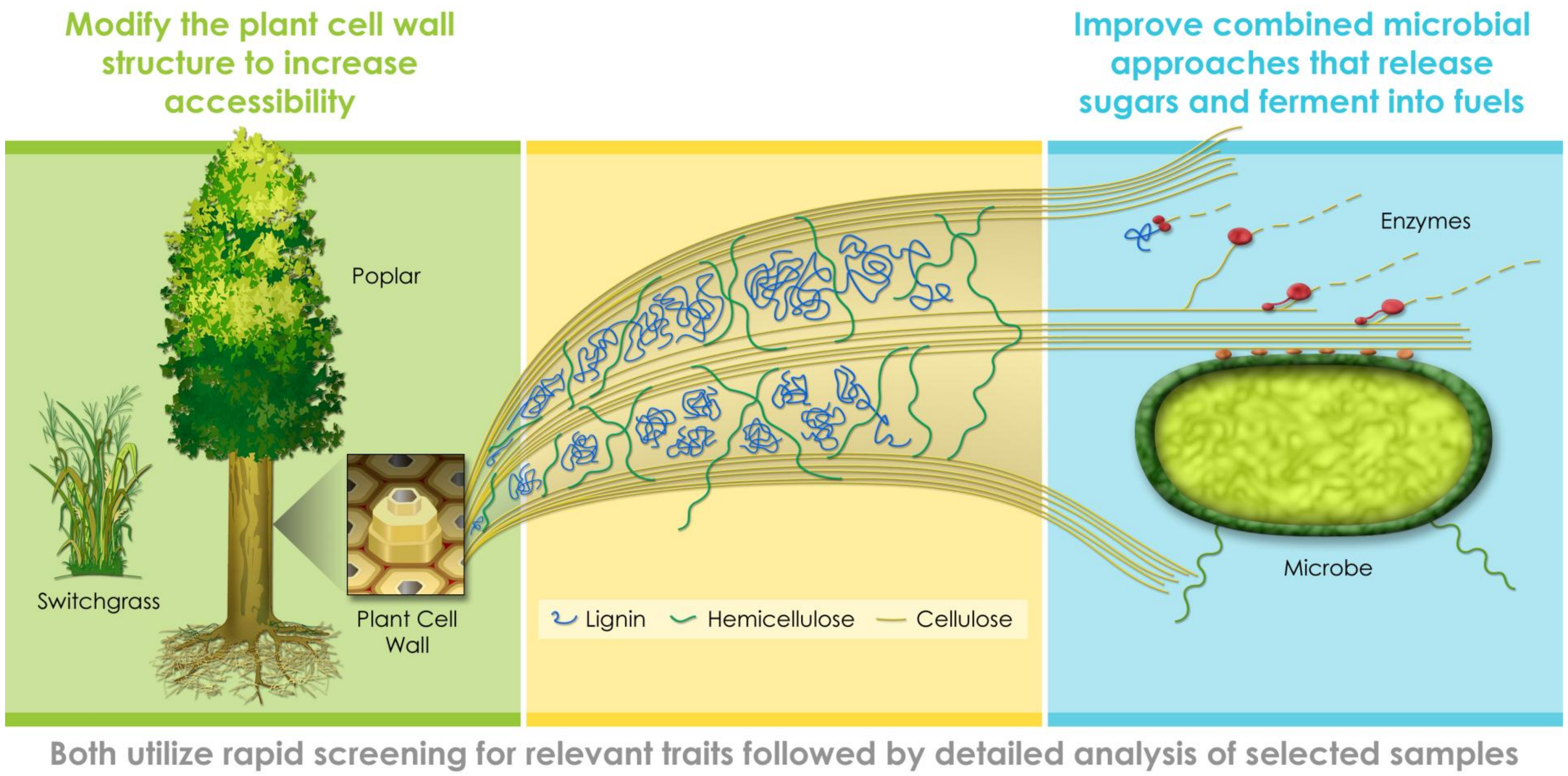
Due on April 02 (send by email)

Office hour:
Tue, Thu and Fri 2-4pm, MO325A
Or email: yyin@niu.edu

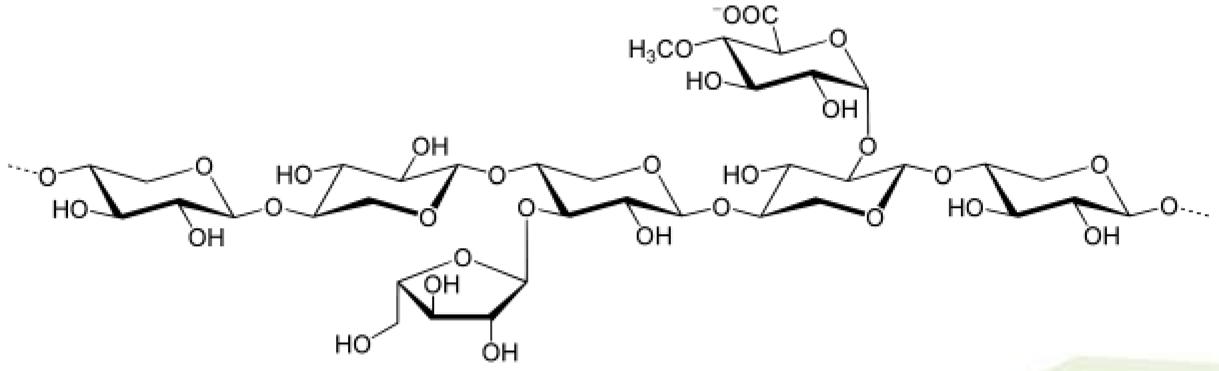
Course Project 1: discover novel GH enzymes in metagenomes

- Glycoside hydrolases: the most critical enzymes to break down polysaccharides
- Metagenomes: the gold mine for enzymes
- Homology search: use known GH proteins to search for homologs in metagenomes
- Phylogeny study: cluster known GHs and metagenome homologs

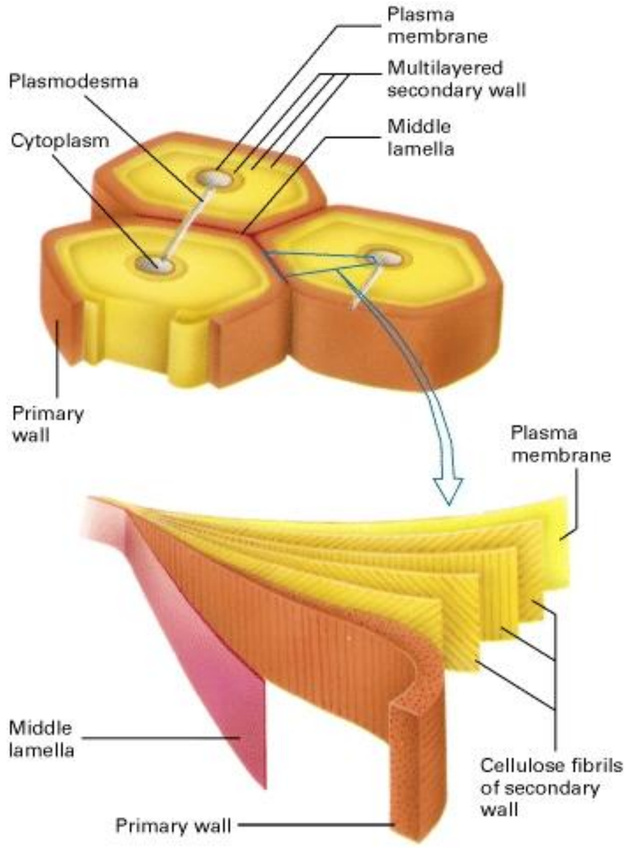
Cellulosic biomass-based bioenergy research: problems and solutions

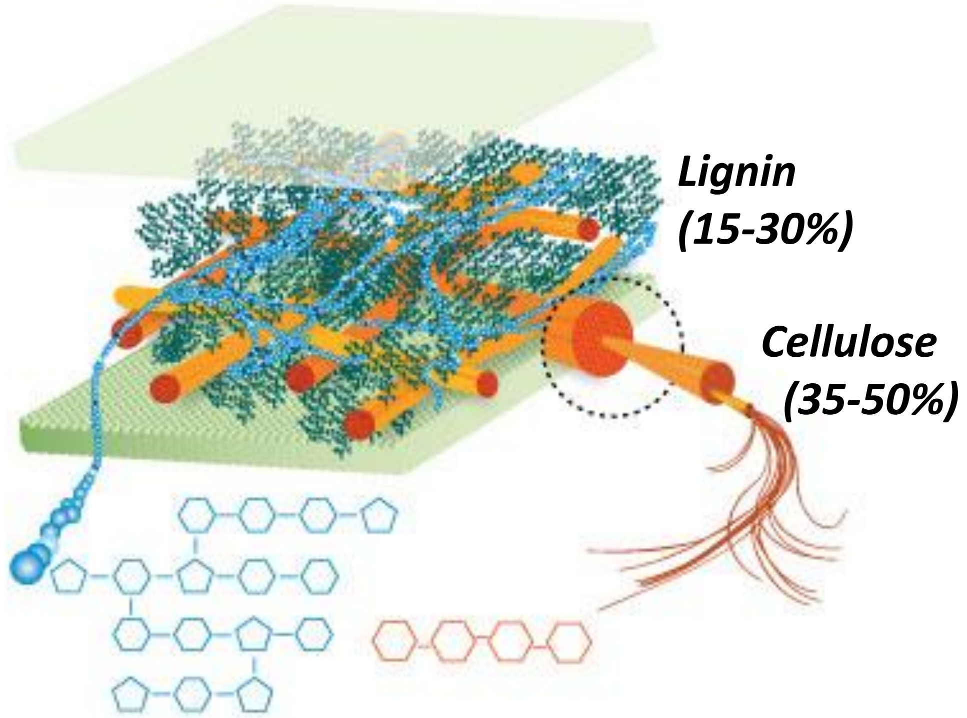

Modify the plant cell wall structure to increase accessibility
Improve combined microbial approaches that release sugars and ferment into fuels
Poplar
Switchgrass
Plant Cell Wall
Enzymes
Microbe
Lignin
Hemicellulose
Cellulose
Both utilize rapid screening for relevant traits followed by detailed analysis of selected samples

Recalcitrance: natural resistance of plant biomass to microbial and enzymatic deconstruction



Top three biopolymers:
Cellulose, xylan, lignin





Hemicelluloses (25-30%)
(xylan, xyloglucan, mannan, MLG)

Metagenomes represent a huge gene pool where people can mine for any treasures they want

- Mine **metagenomes** for novel biomass degrading enzymes

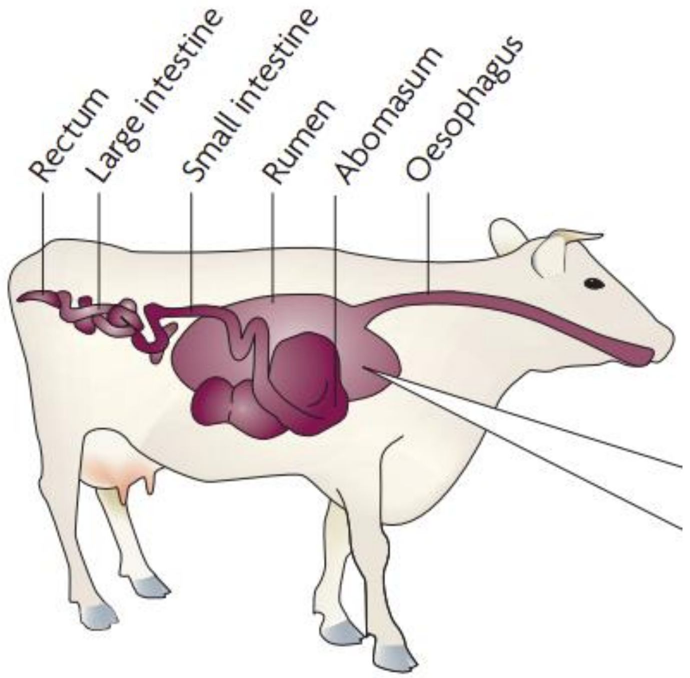
<u>Biomass-degrading microbial communities:</u>

- Animal gut/rumen
- Decomposed biomass

<u>Field soils, oceans, hot springs, whale falls etc.</u>

Public metagenome databases	**# (million) of peptides**
JGI	153
Meta-HIT-BGI	3.3
Meta-HIT-Europe	4.4
Cow-rumen	2.5
CAMERA	67.6
NCBI-env-nr	6.1

NCBI-nr: 20M



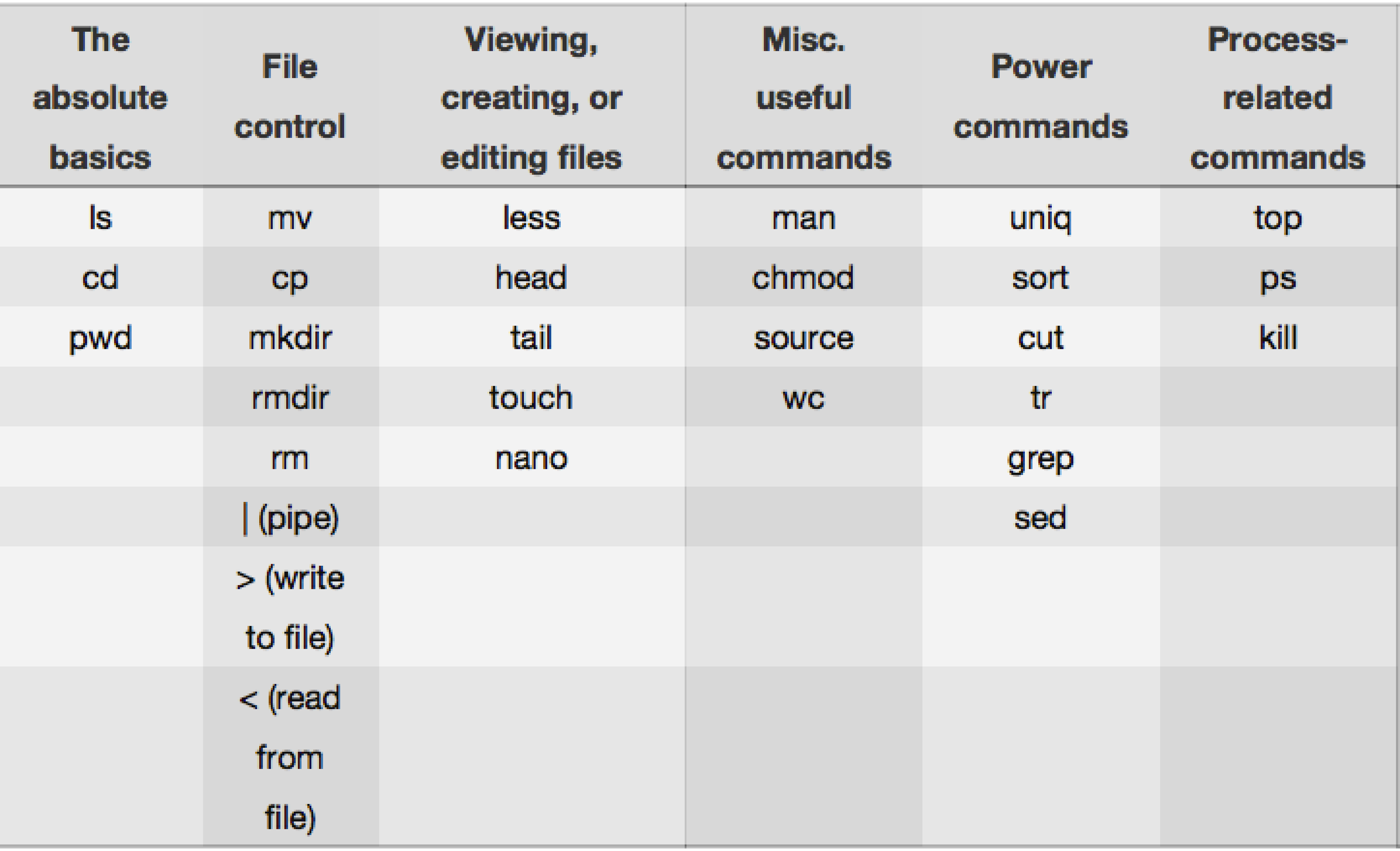
What we learned last class:

file system,
relative/absolute paths,
working folder, home folder,

ssh, pwd, ls
cd, mkdir, rmdir, rm, man
less, more, head, tail
cp, mv

If things go wrong, try:

Ctrl+c (sometimes multiple times)
Esc
q
Ctrl+z



The absolute basics	File control	Viewing, creating, or editing files	Misc. useful commands	Power commands	Process-related commands
ls	mv	less	man	uniq	top
cd	cp	head	chmod	sort	ps
pwd	mkdir	tail	source	cut	kill
	rmdir	touch	wc	tr	
	rm	nano		grep	
	| (pipe)			sed	
	> (write to file)				
	< (read from file)				

http://korflab.ucdavis.edu/Unix_and_Perl/unix_and_perl_v3.1.1.pdf

Create or edit files

Text editors:

nano

pico

vi

Suppose you are at your home:

1. Copy a file to your home/bioinfo

```
cp /home/yyin/work/SRR043594/454Isotigs.txt bioinfo
```

2. Try nano (Intuitive user interface)

```
nano 454Isotigs.txt
```

3. Try vi (command-driven interface, but much more power)

```
vi 454Isotigs.txt
```

Create a file from scratch using vi.

1) you type `vi filename` and hit *enter*
2) after you are in *vi*, type *i* to get into edit mode and copy & paste content in *vi*
3) hit *Esc* to exit edit mode and then *:x* to save the file and exit *vi*.

vi basics

$$\begin{tikzcd}\text{command mode} \arrow[r, shift left, "i"] & \text{edit mode} \arrow[l, shift left, "\text{Esc}"]\end{tikzcd}$$

The following commands **operate in command mode (hit Esc before using them)**

x	delete one character at cursor position
u	undo
dd	delete the current line
G	go to end of file
1G	go to beginning of file
10G	go to line 10
$	go to end of line
1	go to beginning of line
:q!	exit without saving
:w	save (but not exit)
:wq or :x	save and exit
Arrow keys:	move cursor around (in both modes)

http://cbsu.tc.cornell.edu/ww/1/Default.aspx?wid=36

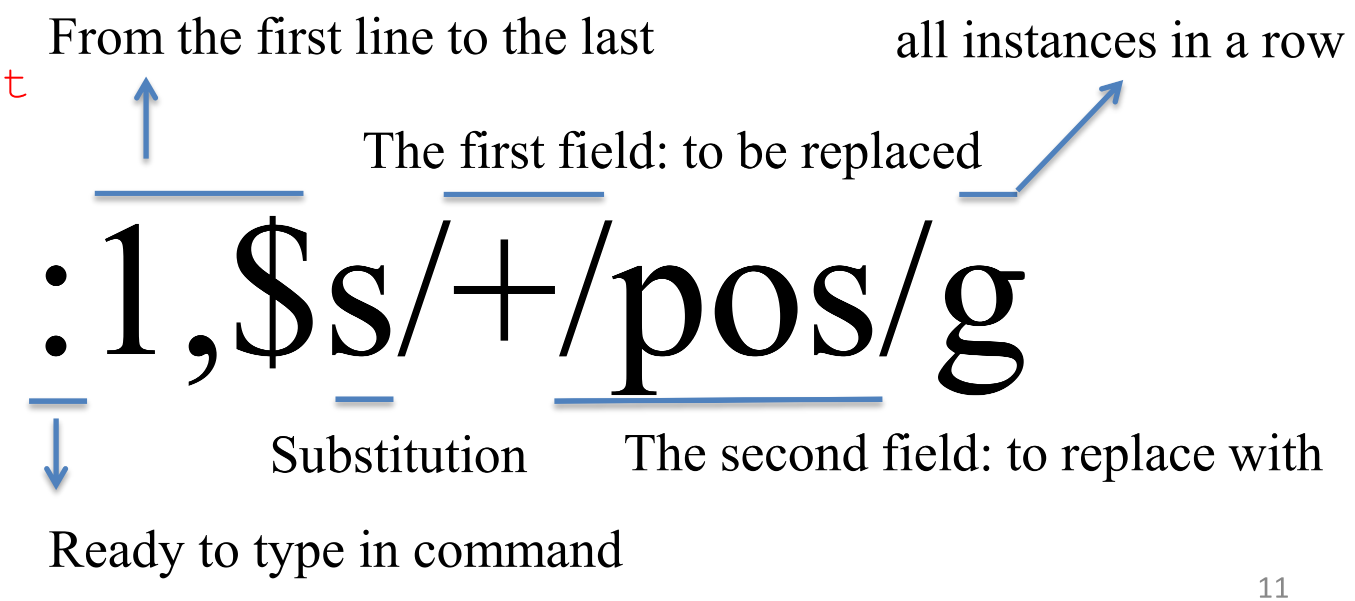
Search and substitution in vi

In command mode, you can do a number of fancy things. The most useful are:

- Search: hit slash ("/") to get the cursor to the left-bottom corner; you can type any word or letter to search it; type *n* to go to the next instance

- Replace: hit *Esc* (at any time, hitting *Esc* to get back to the default status is the safest thing to do) and type "*:1,$s/+/pos/g*" and then enter will replace all "+" to "pos".

Try this in
```
vi 454Isotigs.txt
```


From the first line to the last
all instances in a row
The first field: to be replaced
:1,$s/+/pos/g
Substitution
The second field: to replace with
Ready to type in command

Wild cards and regular expression

Regular expression (regex or regexp) is a very powerful tool for text processing and widely used in text editors (e.g. vi) and programming languages (e.g. Shell commands: sed, awk, grep and perl, python, PHP) to automatically edit (match and replace strings) texts.

Finding and replacing exact words or characters are simple, e.g. the vi example shown above

However, if you want to match multiple words or characters, you will need wildcards or patterns.

a list of commonly used wildcards and patterns:

*	any numbers of letters, numbers and characters except for spaces and special characters, e.g. ()[]+\/$@#%;,?
.	any single letter, number and character including special characters
^	start of a line caret
$	end of a line
^$	an empty line, i.e. nothing between ^ and $
[]	create your own pattern, e.g. [ATGC] matches one of the four letters only, [ATGC]{2} matches two such letters; [0-9]: any numbers
\w	any letter (a-z and A-Z)
\d	any number (0-9)
+	previous items at least one times, e.g. \w+ matches words of any sizes
{n}	previous items n times, e.g. \w{5} matches words with exactly five letters
\s	space
\t	tabular space
\n	new line

Curly brackets

http://www.bsd.org/regexintro.html

Get data from remote ftp/http website

ftp
sftp
ncftp
lftp

`lftp addr`	command to connect to a remote ftp server
`cd dir`	change to the directory
`cd ..`	change to the upper folder (..)
`ls`	list files and folders in the current directory at once
`ls dir`	list files and folders in dir at once
`ls | less`	list page by page (good if the list is too long)
`get file`	get a file
`mirror dir`	get a folder
`zmore file`	view the file content
`by` or `bye`	exit lftp

Transfer files between two Linux machines

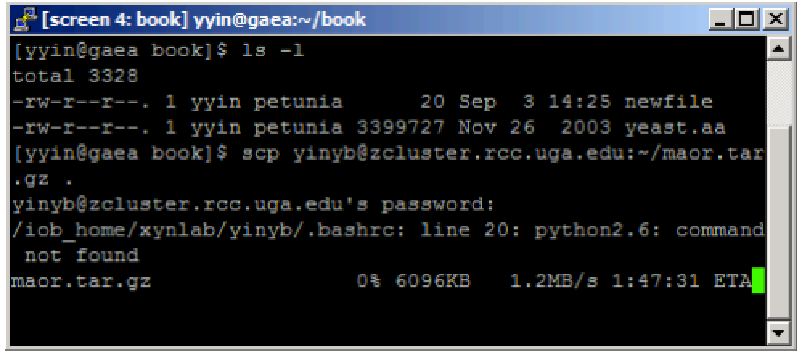
scp: secure copy files/folders between hosts on a network

You are at a Linux machine, e.g. your laptop with Ubuntu installed
You must have access to the remote machine, say glu server

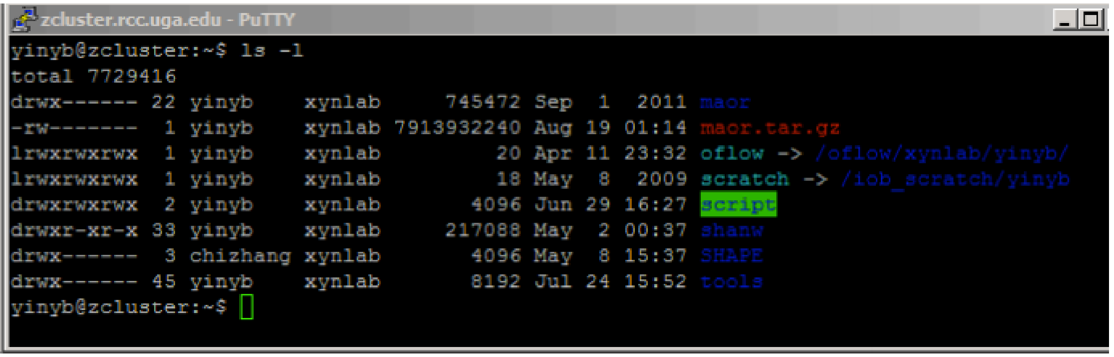
Try

```
scp 454Isotigs.txt username@131.156.41.196:~/
scp -r username@131.156.41.196:~/bioinfo .
```

You will be asked for password



```
[yyin@gaea book]$ ls -l
total 3328
-rw-r--r--. 1 yyin petunia      20 Sep  3 14:25 newfile
-rw-r--r--. 1 yyin petunia 3399727 Nov 26  2003 yeast.aa
[yyin@gaea book]$ scp yinyb@zcluster.rcc.uga.edu:~/maor.tar
.gz .
yinyb@zcluster.rcc.uga.edu's password:
/iob_home/xynlab/yinyb/.bashrc: line 20: python2.6: command
 not found
maor.tar.gz                0% 6096KB   1.2MB/s 1:47:31 ETA
```



```
yinyb@zcluster:~$ ls -l
total 7729416
drwx------ 22 yinyb    xynlab     745472 Sep  1  2011 maor
-rw-------  1 yinyb    xynlab 7913932240 Aug 19 01:14 maor.tar.gz
lrwxrwxrwx  1 yinyb    xynlab         20 Apr 11 23:32 oflow -> /oflow/xynlab/yinyb/
lrwxrwxrwx  1 yinyb    xynlab         18 May  8  2009 scratch -> /iob_scratch/yinyb
drwxrwxrwx  2 yinyb    xynlab       4096 Jun 29 16:27 script
drwxr-xr-x 33 yinyb    xynlab     217088 May  2 00:37 shanw
drwx------  3 chizhang xynlab       4096 May  8 15:37 SHAPE
drwx------ 45 yinyb    xynlab       8192 Jul 24 15:52 tools
yinyb@zcluster:~$
```

wget

wget is a program useful for downloading files from both FTP and HTTP sites.

wget is non-interactive: you simply enter the necessary options and arguments on the command line and the file is downloaded for you.

You must identify the links first: browse a http webpage or a ftp site and locate the remote files/folders you want to download and then go to the terminal and type

-q quiet
-r recursive (for folders)

For example,

```
wget http://cys.bios.niu.edu/yyin/teach/PBB/cesa-pr.fa
wget -q ftp.ncbi.nih.gov/blast/db/FASTA/yeast.aa.gz
wget -r -q
ftp://ftp.ncbi.nih.gov/genomes/Bacteria/Escherichia_coli_K_12_
substr__MG1655_uid57779
wget -q ftp.ncbi.nih.gov:/blast/executables/LATEST/ncbi-blast-
2.2.27+-x64-linux.tar.gz
wget ftp://emboss.open-bio.org/pub/EMBOSS/emboss-latest.tar.gz
```

Input and output redirection: **the greater-than sign**

Unix has a special way to **direct input and output** from commands or programs.

By default, the input is from keyboard (called standard input, *stdin*): you type in a command and Shell takes the command and executes it.

The standard output by default is to the terminal screen (*stdout*);

if the command or program failed, you will also have standard errors dumped to the terminal screen (*stderr*).

However, if you do not want the output dumped to the screen, you can use ">" to redirect/write the output into a file. For example, try

```
head -50 454Isotigs.txt
head -50 454Isotigs.txt > 454Isotigs.txt.head50
head -50 454Isotigs.tx
head -50 454Isotigs.tx 2> 454Isotigs.txt.head50
```

"*2>*" to dump the error message
No space here!

Archive and compress files/folders

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:

zless to view zipped files

- 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*)*

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)*

Use man tar to learn more

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)*

Try to unpack and uncompress files downloaded in slide #14

Check disk usage

Disk space is a limited resource, and you want to frequently monitor how much disk space you have used. To check the disk space usage, use the *du* (disk usage) command

```
yyin@glu:~$ du -hs .
22G
```

To check how much space left on the entire storage file system, use the *df* command

```
yyin@glu:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       894G  665G  184G  79% /
udev             12G  8.0K   12G   1% /dev
tmpfs           4.8G  848K  4.8G   1% /run
none            5.0M     0  5.0M   0% /run/lock
none             12G  304K   12G   1% /run/shm
/dev/sdb1       1.8T  196M  1.7T   1% /media/DATAPART5
/dev/sdf1       917G  3.0G  868G   1% /media/DATAPART4
/dev/sde1       917G  2.7G  868G   1% /media/DATAPART3
/dev/sdd1       917G  4.5G  866G   1% /media/DATAPART2
/dev/sdc1       917G  4.3G  867G   1% /media/DATAPART1
```

job monitor and control

top: similar to windows task manager (space to refresh, q to exit)

w: who is there

ps: all running processes, PID, status, type
ps -ef

bg: move current process to background

fg: move current process to foreground

jobs: list running and suspended processes

kill: kill processes
kill pid (could find out using top or ps)

sort, cut, uniq, join, paste, sed, grep, awk, wc, diff, comm, cat

All types of bioinformatics sequence analyses are essentially text processing.

Unix Shell has the above commands that are very useful for processing texts and also allows **the output from one command to be passed to another command as input using pipes** ("|").

This makes the processing of files using Shell very convenient and very powerful: you do not need to write output to intermediate files orload all data into the memory.

For example, combining different Unix commands for text processing is like passing an item through a manufacturing pipeline when you only care about the final product

Hands on example 1: cosmic mutation data

- Go to UCSC genome browser website: http://genome.ucsc.edu/
- On the left, find the Downloads link
- Click on Human
- Click on Annotation database
- Ctrl+f and then search "cosmic"
- On "cosmic.txt.gz" right-click -> copy link address
- Go the terminal and wget the above link (middle click or Shift+Insert to paste what you copied)
- Similarly, download the "cosmicRaw.txt.gz" file

- Under your home, create a folder called class (mkdir)
- Under home/class, create a folder called mar19 (mkdir)
- Move the above downloaded files to home/class/mar19 (mv)
- Change to that directory (cd)
- Use gzip to uncompress the two files (gzip -d)

```
zless cosmic.txt.gz (q to exit)
zless cosmicRaw.txt.gz
gzip -d *.gz
less cosmicRaw.txt

less cosmicRaw.txt | cut -f2
(ctrl+c to stop, or ctrl+z to suspend and then jobs, then kill -9 %1)

less cosmicRaw.txt | cut -f2 | less
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | less

(http://hgdownload.soe.ucsc.edu/goldenPath/hg19/database/cosmicRaw.sql)

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | less
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | cut -f1 | sort -u | wc
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | awk '$6=="liver"`
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | less
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | sort | uniq -c
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | sort | uniq -c | sort -k
1,1nr

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | sort | uniq -c | sort -k
2,2n

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | cut -f6 | sort | uniq
-c | sort -k 1,1nr

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | cut -f2 | sort | uniq
-c | sort -k 1,1nr | less
```

Hands on example 2: process fasta sequence data

```
- Download genome data of multiple e.coli k-12 strains
wget -q -r ftp://ftp.ncbi.nih.gov/genomes/Bacteria/Escherichia_coli_K_12* &
- List
ls -l ftp.ncbi.nih.gov/genomes/Bacteria/
- List
ls -l
ftp.ncbi.nih.gov/genomes/Bacteria/Escherichia_coli_K_12_substr__MG1655_uid57779/
- Find all .faa files
find ftp.ncbi.nih.gov/ -name *faa
- Count
find ftp.ncbi.nih.gov/ -name *faa | wc
- Cat all protein sequences into one large file
find ftp.ncbi.nih.gov/ -name *faa| xargs cat | less
find ftp.ncbi.nih.gov/ -name *faa| xargs cat > ecoli-all.faa
- Count how many proteins
find ftp.ncbi.nih.gov/ -name *faa| xargs cat | grep '>' | wc -l
- View the protein description lines
find ftp.ncbi.nih.gov/ -name *faa |xargs cat | grep '>' | less
find ftp.ncbi.nih.gov/ -name *faa |xargs cat | grep '>' | cut -f1 -d ' ' | less

find ftp.ncbi.nih.gov/ -name *faa |xargs cat | grep '>' | cut -f1 -d ' ' | sed
's/>//' | head

find ftp.ncbi.nih.gov/ -name *faa |xargs cat | grep '>' | cut -f1 -d ' ' | sed
's/>//' | cut -f4 -d'|' | head
```

put the process to background

for loop on command line

```
for variable in `command`
do
        command 1
        command 1
done
```

The symbol on the tilde key (~)

`

http://en.wikipedia.org/wiki/Grave_accent

Or backtick

```
for x in `find ftp.ncbi.nih.gov/ -name *faa`
do
        echo $x
        cat $ix | grep '>' | wc -l
done
```

```
for x in `find ftp.ncbi.nih.gov/ -name *faa`
do
        cat $x >> ecoli-all.faa.2
done
```

find ftp.ncbi.nih.gov/ -name *faa| xargs cat > ecoli-all.faa

- Save history of your commands:

```
history | less
history > hist1
```

- Send message to other online users

```
write username (ctrl+c to exit)
```

- Change your password

```
passwd
```

Ctrl+c to tell Shell to stop current process
Ctrl+z to suspend
bg to send to background
Ctrl+d to exit the terminal (logout)

More example:

Find a cazy family, vi to save the protein id

Save as excel file and upload to glu

Practice cut, sort uniq etc