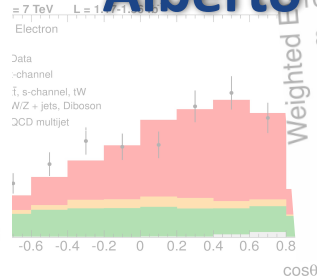
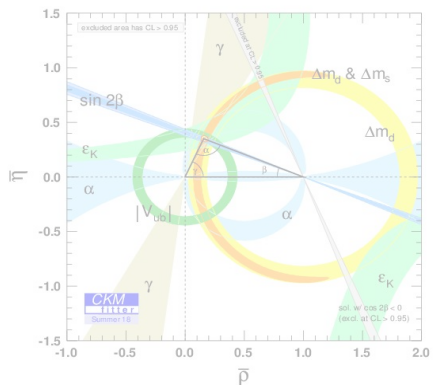
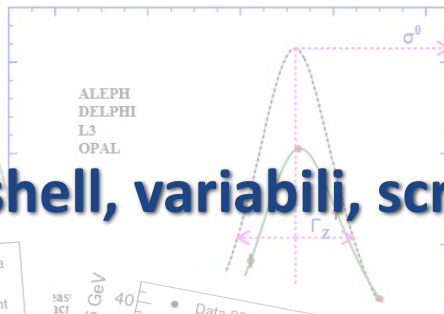
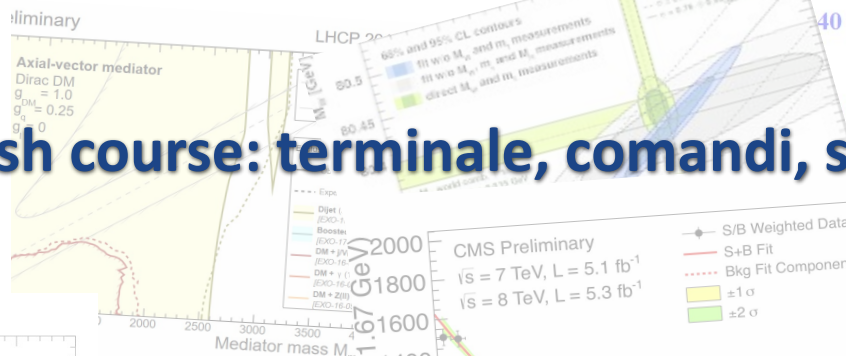
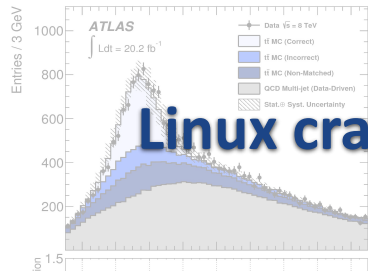
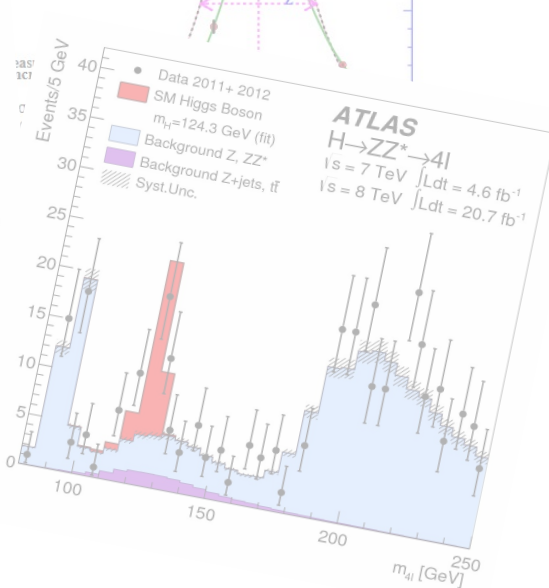




UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II



Alberto Orso Maria Iorio



Il terminale

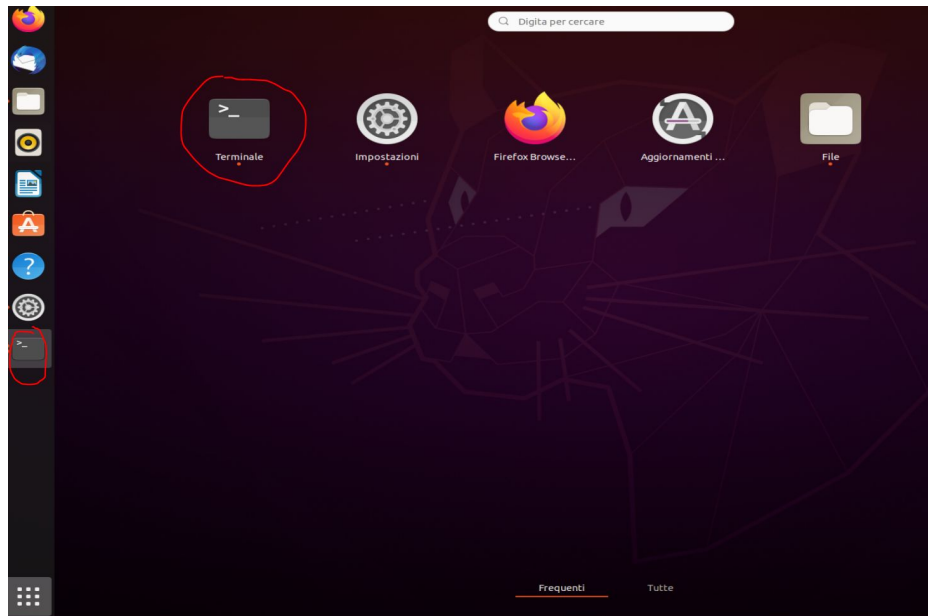
Un sistema operativo tipicamente “parte” dal formato di un terminale

- Prima di Windows c’era “DOS”
- Windows stesso ha un prompt di linea di comando da cui è possibile modificare le cose senza l’interfaccia grafica

Per Linux, tipicamente si può cercare nelle applicazioni il terminale

Oppure si apre con

`ctrl+alt+t`



Linux: comandi di base

Per vedere il path completo in cui mi trovo uso il comando

pwd

```
(base) orso@orso-VirtualBox:~$ pwd  
/home/orso
```

Questo si chiama “percorso completo” - **full path** - del sistema di file (**filesystem**)

Per vedere il contenuto della directory in cui mi trovo

ls

```
(base) orso@orso-VirtualBox:~$ ls  
anaconda3      Documenti  macros      Musica      Pubblici  
condainstall.sh Immagini   Modelli     notebooks   root  
(base) orso@orso-VirtualBox:~$
```

→ la directory attuale è /home/orso

→ ci sono un certo numero di directory (in blue) e files (in bianco)

→ Il full path delle sottocartelle è quella della cartella dove mi trovo + cartella, ad esempio:

/home/orso/Scrivania

Linux: comandi di base

Per vedere il path completo in cui mi trovo uso il comando

pwd

```
(base) orso@orso-VirtualBox:~$ pwd  
/home/orso
```

Questo si chiama “percorso completo” - **full path** - del sistema di file (**filesystem**)

Per vedere il contenuto della directory in cui mi trovo

ls

```
(base) orso@orso-VirtualBox:~$ ls  
anaconda3      Documenti  macros      Musica      Pubblici  
condainstall.sh Immagini   Modelli     notebooks   root  
(base) orso@orso-VirtualBox:~$
```

Nome utente

Nome macchina

→ la directory attuale è /home/orso

→ ci sono un certo numero di directory (in blue) e files (in bianco)

→ Il full path delle sottocartelle è quella della cartella dove mi trovo + cartella, ad esempio:

/home/orso/Scrivania

Linux: comandi di base

Esistono files “nascosti” che iniziano tipicamente col punto. Si possono vedere con l’opzione aggiuntiva “-a”. Altre opzioni, come “-l”, permettono di vedere più informazioni :

```
ls -al
```

```
(base) orso@orso-VirtualBox:~$ ls -la
totale 260700
drwxr-xr-x 27 orso orso      4096 ott 30 12:20 .
drwxr-xr-x  3 root root      4096 ott 24 20:30 ..
drwxrwxr-x  3 orso orso      4096 ott 24 21:37 .anaconda
drwxrwxr-x 28 orso orso      4096 ott 24 21:46 anaconda3
-rw-r--r--  1 orso orso       868 ott 29 15:09 .bash_history
-rw-r--r--  1 orso orso       220 ott 24 18:49 .bash_logout
-rw-r--r--  1 orso orso     4281 ott 29 15:09 .bashrc
```

- la directory attuale si indica sempre con il punto, “.”
- la directory superiore si indica con “..”
- **anaconda3** è una cartella, **.anaconda** una cartella nascosta
- **.bashrc** è un **file nascosto molto importante: contiene le operazioni effettuate all’avvio del terminale**
- **.bash_history** raccoglie gli ultimi comandi usati
- **.bash_logout** dice cosa fare in chiusura di terminale

Linux: comandi di base

Esistono files “nascosti” che iniziano tipicamente col punto. Si possono vedere con l’opzione aggiuntiva “-a”. Altre opzioni, come “-l”, permettono di vedere più informazioni :

```
ls -al
```

```
(base) orso@orso-VirtualBox:~$ ls -la
totale 260700
drwxr-xr-x 27 orso orso      4096 ott 30 12:20 .
drwxr-xr-x  3 root root      4096 ott 24 20:30 ..
drwxrwxr-x  3 orso orso      4096 ott 24 21:37 .anaconda
drwxrwxr-x 28 orso orso      4096 ott 24 21:46 anaconda3
-rw-----  1 orso orso       868 ott 29 15:09 .bash_history
-rw-r--r--  1 orso orso       220 ott 24 18:49 .bash_logout
-rw-r--r--  1 orso orso     4281 ott 29 15:09 .bashrc
```

→ i primi 10 caratteri (ad es. drwxr-xr-x) indicano se è una directory (**d**) e i permessi:

lettura **r**, scrittura **w**, esecuzione **x**

Sono divisi in tre sottogruppi:

utente : in questo caso orso

group : un gruppo di utenti che contiene molteplici utenti, ad es. I membri di una collaborazione

all: qualunque utente che entra su questa macchina

Linux: comandi di base

Esistono files “nascosti” che iniziano tipicamente col punto. Si possono vedere con l’opzione aggiuntiva “-a”. Altre opzioni, come “-l”, permettono di vedere più informazioni :

```
ls -al
```

```
(base) orso@orso-VirtualBox:~$ ls -la
totale 260700
drwxr-xr-x 27 orso orso      4096 ott 30 12:20 .
drwxr-xr-x  3 root root      4096 ott 24 20:30 ..
drwxrwxr-x  3 orso orso      4096 ott 24 21:37 .anaconda
drwxrwxr-x 28 orso orso      4096 ott 24 21:46 anaconda3
-rw-----  1 orso orso       868 ott 29 15:09 .bash_history
-rw-r--r--  1 orso orso       220 ott 24 18:49 .bash_logout
-rw-r--r--  1 orso orso     4281 ott 29 15:09 .bashrc
```

I campi successivi sono

tipo : tipo di file (file 1, directory 2+)

proprietario: l’utente proprietario del proprietario del file . PS ma chi è sto root? See next slides...

gruppo: il gruppo a cui appartiene l’utente proprietario.

dimensione: nota che le cartelle hanno dimensione piccola: **NON è la dimensione del contenuto della cartella!1!!11!**

data e ora ultima modifica

Linux: comandi di base

Per vedere le sottocartelle

```
ls nomedirectory
```

Esempio:

```
(base) orso@orso-VirtualBox:~$ ls anaconda3/  
bin          condabin    doc         etc         info  
compiler_compat  conda-meta  envs       include    lib  
(base) orso@orso-VirtualBox:~$
```

Un'altra combinazione di comandi utili è

`ls -lthra` → oltre a quello che fa `-la` ordina i risultati in ordine cronologico (opzione `-t`) inverso (`-r`) e mostra le dimensioni in formato leggibile (kilobytes, megabytes, gigabytes etc)

Linux: la directory “root” e l’utente “root”

Esiste un utente che agisce da “root” che è l’amministratore (o “superuser”) del computer. Se l’utente “root” coincide con uno degli utenti

Inoltre, la cartella madre di tutte, è indicata con “/” senza nulla davanti
Solo l’utente root può modificare questa cartella e i suoi contenuti

```
ls / -la
```

```
(base) orso@orso-VirtualBox:~$ ls / -la
totale 1190472
drwxr-xr-x 20 root root    4096 ott 24 18:48 .
drwxr-xr-x 20 root root    4096 ott 24 18:48 ..
lrwxrwxrwx  1 root root      7 ott 24 18:46 bin -> usr/bin
drwxr-xr-x  4 root root    4096 ott 24 21:50 boot
drwxrwxr-x  2 root root    4096 ott 24 18:48 cdrom
drwxr-xr-x 20 root root    4140 ott 25 13:36 dev
drwxr-xr-x 132 root root   12288 ott 29 23:07 etc
```

Sono cartelle che appartengono a root e solo il superuser le può modificare

Si possono eseguire comandi come amministratore facendo “sudo + comando”, ad es.:
sudo ls ; sudo pwd e così via. N.B.: ovviamente richiede la password da amministratore!

Linux: le cartelle di base

Le sottocartelle della root danno l'ossatura al filesystem:

```
ls /
```

```
(base) orso@orso-VirtualBox:~$ ls /  
bin boot cdrom dev etc home lib lib32 lib64 libx32 lost+found media mnt opt proc
```

Alcune comunemente usate:

/home

→ Dove sono gli *utenti* e tutte le sottocartelle annesse (ad es. Desktop)

/bin

→ Dove stanno *i comandi eseguibili che in genere linux conosce*, come “*ls, pwd, cd*” e così via

/lib

→ Dove sono solitamente ubicate le “*librerie comuni*”

/mnt o /mount

→ I dischi esterni: penne, ma anche filesystem che non sono quello attuale

/etc

→ Dove vengono spesso salvate informazioni aggiuntive sul setup che non hanno un posto specifico

Linux: spostarsi di cartella in cartella

Per spostarmi in una cartella diversa da quella attuale uso il comando

```
cd cartella
```

Posso spostarmi in due modi: 1) mettendo il full path partire dalla cartella root ("/")

```
(base) orso@orso-VirtualBox:~$ pwd
/home/orso
(base) orso@orso-VirtualBox:~$ cd /
(base) orso@orso-VirtualBox:/$ pwd
/
(base) orso@orso-VirtualBox:/$ cd home/orso/
(base) orso@orso-VirtualBox:~$ pwd
/home/orso
(base) orso@orso-VirtualBox:~$
```

2) Mi sposto a partire dal percorso relativo dove mi trovo, ad esempio_

```
cd scrivania
```

Dove si sottintende il path di partenza da cui mi trovo

Linux: altri comandi utili

Per creare una cartella nuova si può fare:

```
mkdir cartella
```

Per rimuovere un file si può fare

```
rm file
```

Per cambiare i permessi di un file:

```
chmod (chi) (+ o -) (cosa)
```

Esempi:

`chmod u+x file` → aggiunge all'utente il permesso di eseguire (oltre a quelli che ha)

`chmod g-w file` → toglie al gruppo i permessi di scrivere sul file (se ce l'ha)

`chmod o-w file` → toglie a tutti gli altri i permessi di scrivere sul file (se ce l'ha)

`chmod a+xwr file` → aggiunge a tutti (u, g, o) il permesso di leggere, scrivere o eseguire




Linux: la shell

Il terminale ha un **modo di comunicare con il filesystem** / con il nucleo del diverso da macchina a macchina.

L'interfaccia dei comandi col terminale è la **“shell”**

Ci sono diversi tipi di shell, trovate la vostra con il comando

```
echo $SHELL
```



```
(base) orso@orso-VirtualBox:~$ echo $SHELL
/bin/bash
```

`SHELL` è una **“variabile d’ambiente”**: sono definite quando apri il terminale e sono usate per indicare i percorsi importanti o le quantità cruciali

→ Esistono diversi tipi di shell, le più frequenti sono

`bash` : la shell “standard” o meglio una sua evoluzione (“sh” è quella originaria).

Un’evoluzione è la korn shell (`ksh`)

`csh` : c-shell con alcune funzionalità di c, inclusi alias e operazioni aritmetiche

`zsh`: usata per la customizzazione

Linux: variabili d'ambiente

A seconda del tipo di shell, posso definire comandi facendo

```
VARIABILE=hello
```

E ne ricavo il contenuto con il comando echo e \$ davanti alla variabile il contenuto

```
echo $VARIABILE
```

```
(base) orso@orso-VirtualBox:~$ VARIABILE=hello  
(base) orso@orso-VirtualBox:~$ echo $VARIABILE  
hello
```

Un esempio è la variabile "USER" che mi dice il nome dell'utente attuale:

```
echo $USER
```

Le variabili si possono comporre con del testo:

```
VARIABILE2= VARIABILE" world"
```

```
(base) orso@orso-VirtualBox:~$ VARIABILE2=$VARIABILE" world"  
(base) orso@orso-VirtualBox:~$ echo $VARIABILE2  
hello world
```

O con l'output di un comando usando il "reverse single quotation mark" .

```
VARIABILE3='pwd'
```

```
(base) orso@orso-VirtualBox:~$ VARIABILE3=`pwd`  
(base) orso@orso-VirtualBox:~$ echo $VARIABILE3  
/home/orso
```


Linux: variabili d'ambiente

Una variabile importante è il PATH

```
echo $PATH
```

```
(base) orso@orso-VirtualBox:~$ echo $PATH  
/home/orso/root/bin:/home/orso/anaconda3/bin:/home/orso/anaconda3/condabin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

Questa variabile indica la **directory da cui la shell prende i comandi <-> binaries!**

Nota bene: qui ci sono **molteplici cartelle separate da due punti!**

Si può estendere ad esempio componendola con il nuovo path:

```
PATH=$PATH:root/bin
```

Che aggiunge i comandi di root a quelli disponibili

Linux: script di shell

Una sequenza di comandi può essere scritto in uno “script di shell”, tipo quello [messo su git](#)

Si esegue con il comando:

```
source shellexercise.sh
```

Questo contiene una serie di comandi di esempio che eseguirà in sequenza

```
echo "Hello world!"
#Il comando echo ripete qualsiasi cosa tu scriva dopo, quindi espande le variabili
#il path è"
echo $PATH
echo "un comando interessante è 'more' che fa leggere un file di testo dall'inizio alla fine, ad es.
more shellexercise.sh. Adesso lancerò il comando:"
#
#
more shellexercise.sh
#
#I commenti sono con il cancelletto. Questo commento compare perchè lo script usa il "more"
echo "altri comandi utili:
history : fa la storia dei comandi usati
tail : legge un file dalla fine invece che dall'inizio
ps : mostra i processi aperti sulla macchina
"
```

Script bashrc / tcshrc

Esistono script “default” che vengono avviati quando si inizializza la shell.

Si trovano nella directory “home”, ovvero `/home/username`

Sono files nascosti e si chiamano tipicamente, per shell bash e tcsh:

`.bashrc`

`.tcshrc`

Hanno diversi scopi, ad esempio:

- inizializzare le variabili d’ambiente
 - avviare eventuali processi
 - montare eventuali altri supporti (altri filesystem / macchine di stoccaggio dati)
 - spostarsi in una cartella di lavoro
- etc...

La cosa più importante: **si possono personalizzare!**

Esempio: la variabile `PATH` dei comandi si può aggiornare aggiungendo ulteriori path dove prendere i comandi, tipo ... root!

Script bashrc / tcshrc: personalizzazione

Apriamo uno script bash, ad esempio `.bashrc`:

```
#!/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
    *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth
```

Alla fine c'è una parte aggiunta durante l'installazione di anaconda. che non è presente nel linux natio. Io qui ho aggiunto il "source" del file `thisroot.sh` che a sua volta fa il setup di alcune variabili d'ambiente:

```
unset __conda_setup
# <<< conda initialize <<<

source /home/orso/root/bin/thisroot.sh
```

Questo ci porta alla nostra prossima meta: parlare di ROOT (of all evil)

Esercizio: scriviamo un bash script!

E1) Creiamo un file di shell nuovo:

```
worksetup.sh
```

- Questo file deve fare le seguenti cose:

- 1.1 creare spostarsi nella propria cartella home usando il path completo
- 1.2 creare una cartella di lavoro "username_workspace"
- 1.3 andare lì dentro
- 1.4 da lì fare il source di thisroot.sh che si trova all'interno della cartella root in cui è stato installato