

Advanced Linux Usage

2018-11-13

Martin Dahlö martin.dahlo@scilifelab.uu.se

Enabler for Life Science











Same program, many files

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep   1 16:42 sample_9.bam
$ my_prog sample_1.bam
```



Same program, many files

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep    1 16:42 sample_9.bam
$ my_prog sample_1.bam
$ my_prog sample_2.bam
```



Same program, many files

```
s ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 16:42 sample 9.bam
$ my proq sample 1.bam
$ my prog sample 2.bam
$ my prog sample 3.bam
$ my prog sample 4.bam
$ my prog sample 5.bam
$ my prog sample 6.bam
$ my prog sample 7.bam
$ my prog sample 8.bam
$ my prog sample 9.bam
```



- Same program, many files
 - 10 files? Ok
 - 1000 files? Not ok



- Same program, many files
 - 10 files? Ok
 - 1000 files? Not ok
- Reproducibility
 - Self and others



- Same program, many files
 - 10 files? Ok
 - 1000 files? Not ok
- Reproducibility
 - Self and others

A solution - write a script!



```
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep 1 17:18 sample_9.bam
s nano analysis.sh
```



GNU nano 2.0.9 File: analysis.sh Modified

^G Get Help ^X Exit













GNU nano 2.0.9

File: analysis.sh

Modified

my_prog sample_1.bam











GNU nano 2.0.9

File: analysis.sh

Modified

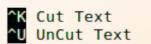
my_prog sample_1.bam
my_prog sample_2.bam

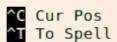














GNU nano 2.0.9 File: analysis.sh Modified

my_prog sample_1.bam my_prog sample_2.bam my_prog sample_3.bam my_prog sample_4.bam my_prog sample_5.bam my_prog sample_6.bam my_prog sample_7.bam my_prog sample_8.bam my_prog sample_9.bam















```
s l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep 7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 9.bam
$
```



```
s l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep 7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 9.bam
$ bash analysis.sh
```



```
s l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep 7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 1.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 2.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 3.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 4.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 5.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 6.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 7.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 8.bam
-rw-rw-r-- 1 dahlo dahlo
                           0 Sep 1 17:18 sample 9.bam
$ bash analysis.sh
```

Still not OK for 1000 or more files!

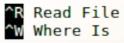


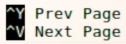
GNU nano 2.0.9 File: analysis.sh Modified

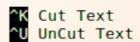
my_prog sample_1.bam my_prog sample_2.bam my_prog sample_3.bam my_prog sample_4.bam my_prog sample_5.bam my_prog sample_6.bam my_prog sample_7.bam my_prog sample_8.bam my_prog sample_9.bam

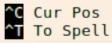
^G Get Help ^X Exit













GNU nano 2.5.3

File: analysis.sh

```
my_prog -r references/human_genome.fa sample_1.bam
my_prog -r references/human_genome.fa sample_2.bam
my_prog -r references/human_genome.fa sample_3.bam
my_prog -r references/human_genome.fa sample_4.bam
my_prog -r references/human_genome.fa sample_5.bam
my_prog -r references/human_genome.fa sample_6.bam
my_prog -r references/human_genome.fa sample_7.bam
my_prog -r references/human_genome.fa sample_8.bam
my_prog -r references/human_genome.fa sample_9.bam
```

















Assigning

```
my_variable=5
my_variable="nice text"
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

```
$my_variable
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

\$my variable

```
$ my_variable="Dave"
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

\$my variable

```
$ my_variable="Dave"
$ echo "Hello, $my variable."
```



Assigning

```
my_variable=5
my_variable="nice text"
```

Using

\$my variable

```
$ my_variable="Dave"
$ echo "Hello, $my_variable."
Hello, Dave.
```



GNU nano 2.5.3

File: analysis.sh

```
my_prog -r references/human_genome.fa sample_1.bam
my_prog -r references/human_genome.fa sample_2.bam
my_prog -r references/human_genome.fa sample_3.bam
my_prog -r references/human_genome.fa sample_4.bam
my_prog -r references/human_genome.fa sample_5.bam
my_prog -r references/human_genome.fa sample_6.bam
my_prog -r references/human_genome.fa sample_7.bam
my_prog -r references/human_genome.fa sample_8.bam
my_prog -r references/human_genome.fa sample_9.bam
```

















GNU nano 2.5.3

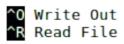
File: analysis.sh

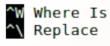
```
ref=references/human_genome.fa
```

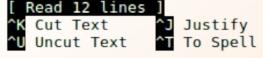
```
my_prog -r $ref sample_2.bam
my_prog -r $ref sample_3.bam
my_prog -r $ref sample_4.bam
my_prog -r $ref sample_5.bam
my_prog -r $ref sample_6.bam
my_prog -r $ref sample_7.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample 9.bam
```

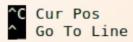
my prog -r \$ref sample 1.bam

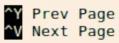














GNU nano 2.5.3

File: analysis.sh

ref=references/goat_genome_version4.1.fa

```
my_prog -r $ref sample_2.bam
my_prog -r $ref sample_3.bam
my_prog -r $ref sample_4.bam
my_prog -r $ref sample_5.bam
my_prog -r $ref sample_6.bam
my_prog -r $ref sample_7.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample 9.bam
```

my prog -r \$ref sample 1.bam

















```
for var in 1 2 3;
do
    echo $var
done
```

```
$ bash loop_test.sh
1
2
3
```



```
for var in text works too;
do
    echo $var
done
```

```
$ bash loop_test.sh
text
works
too
$
```



```
for var in mix them 5;
do
    echo $var
done
```

```
$ bash loop_test.sh
mix
them
5
```



```
for var in *.txt;
do
    echo $var
done
```

```
$ bash loop_test.sh
all.txt
examples.txt
readme.txt
```



GNU nano 2.5.3

File: analysis.sh

ref=references/goat_genome_version4.1.fa

```
my_prog -r $ref sample_2.bam
my_prog -r $ref sample_3.bam
my_prog -r $ref sample_4.bam
my_prog -r $ref sample_5.bam
my_prog -r $ref sample_6.bam
my_prog -r $ref sample_7.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample 9.bam
```

my prog -r \$ref sample 1.bam

















GNU nano 2.5.3

File: analysis.sh

ref=references/goat genome version4.1.fa for file in *.bam ; my_prog -r \$ref \$file done















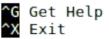


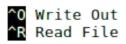
GNU nano 2.5.3

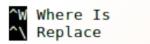
File: analysis.sh

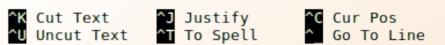
```
ref=references/goat genome version4.1.fa
for file in *.bam ;
   echo my prog -r $ref $file
done
```

Debugging!



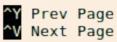














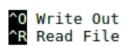
GNU nano 2.5.3

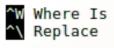
File: analysis.sh

```
ref=references/goat_genome_version4.1.fa
for file in *.bam ;
do
    echo my_prog -r $ref $file
done
```

```
$ bash analysis.sh
my_prog -r references/goat_genome_version4.1.fa sample_1.bam
my_prog -r references/goat_genome_version4.1.fa sample_2.bam
my_prog -r references/goat_genome_version4.1.fa sample_3.bam
my_prog -r references/goat_genome_version4.1.fa sample_4.bam
my_prog -r references/goat_genome_version4.1.fa sample_5.bam
my_prog -r references/goat_genome_version4.1.fa sample_6.bam
my_prog -r references/goat_genome_version4.1.fa sample_7.bam
my_prog -r references/goat_genome_version4.1.fa sample_8.bam
my_prog -r references/goat_genome_version4.1.fa sample_9.bam
```

```
^G Get Help
^X Exit
```









GNU nano 2.5.3

File: analysis.sh

ref=references/goat genome version4.1.fa for file in *.bam ; my_prog -r \$ref \$file done

















\$ bash analysis.sh



\$ bash analysis.sh data/



\$ bash analysis.sh data/

\$1



\$ bash analysis.sh data/ second_argument

\$1 \$2



\$ bash analysis.sh data/ second_argument third

\$1

\$2

\$3



\$ bash analysis.sh data/ second_argument third "fourth argument"
\$1 \$2 \$3 \$4



GNU nano 2.5.3

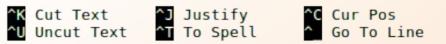
File: analysis.sh

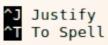
```
ref=references/goat genome version4.1.fa
for file in *.bam ;
   my_prog -r $ref $file
done
```















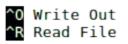


GNU nano 2.5.3

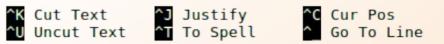
File: analysis.sh

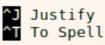
```
ref=references/goat genome version4.1.fa
for file in $1/*.bam ;
   my_prog -r $ref $file
done
```

















```
$ cat file.list
sample_1.bam
sample_3.bam
smaple_9.bam
```

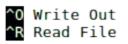


GNU nano 2.5.3

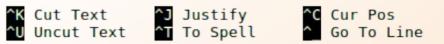
File: analysis.sh

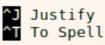
```
ref=references/goat genome version4.1.fa
for file in $1/*.bam ;
   my_prog -r $ref $file
done
```















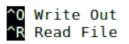


GNU nano 2.5.3

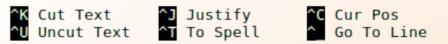
File: analysis.sh

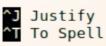
ref=references/goat genome version4.1.fa for file in \$(cat \$1); my_prog -r \$ref \$file done



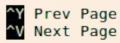














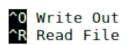
GNU nano 2.5.3

File: analysis.sh

```
ref=references/goat_genome_version4.1.fa
for file in $( cat $1 ) ;
do
    my_prog -r $ref $file
done
```

```
$ cat file.list
sample_1.bam
sample_3.bam
sample_9.bam
$ bash analysis.sh file.list
```

```
^G Get Help
^X Exit
```



```
^W Where Is
^\ Replace
```

```
^K Cut Text
^U Uncut Text
```

```
if condition; then
  action
fi
```



```
if true; then
  echo "This is true"
fi
```

result:

This is true

```
if false; then
  echo "This is true"
fi
```

result:



```
if [[ 5 < 9 ]]; then
  echo "This is true"
fi</pre>
```

result:

This is true

```
if [[ 5 > 9 ]]; then
  echo "This is true"
fi
```

result:

```
if [[ 5 == 9 ]]; then
  echo "This is true"
fi
```

result:



```
if [[ "Hello" == "Hello" ]]; then
  echo "This is true"
fi
```

result:

This is true

```
if [[ "Hello" == "Hi" ]]; then
  echo "This is true"
fi
```

result:



```
if [[ "Hello" == "Hel"* ]]; then
  echo "This is true"
fi
```

result:

This is true



```
for file in $1/*.bam ;
do
     echo my_prog $file
done
```



```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then
        echo my_prog $file
    fi
done
```



```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam



```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file



```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then
       echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file

dog_1.bam



```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then
        echo my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file

dog_1.bam



```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then
        my_prog $file
    fi
done
```

Ex: \$file is /path/to/dog_1.bam

basename \$file

dog_1.bam



- Programming is programming
 - Perl, Python, Bash, and more



- Programming is programming
 - Perl, Python, Bash, and more

```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then
        my_prog $file
    fi
done
```



- Programming is programming
 - Perl, Python, Bash, and more



- Programming is programming
 - Perl, Python, Bash, and more



- Programming is programming
 - Perl, Python, Bash, and more
- Start with one, git gud, (learn another)



- Programming is programming
 - Perl, Python, Bash, and more
- Start with one, git gud, (learn another)

PYTHON



Laboratory time! (yet again)