

Plain text file processing in UNIX

Libor Mořkovský, Václav Janoušek

What do we learn?

- Search a pattern
- Word and line count
- Retrieve & count unique records
- String extraction & replacement
- Join & paste data

Regular expressions

Matching string patterns according to certain rules

<code>^A</code>	Match A at the beginning of line
<code>A\$</code>	Match A at the end of line
<code>[0-9]</code>	Match numerical character
<code>[A-Z]</code>	Match alphabetical character
<code>[ATGC]</code>	Match A, T, C or G
<code>[^A]</code>	Match any character but A
<code>.</code>	Match any character
<code>A*</code>	Match A 0 or more times
<code>A{2}</code>	Match A exactly two times
<code>A{1,}</code> or <code>A+</code>	Match A one or more times
<code>A{1,3}</code>	Match A 1 to 3 times
<code>AATT TTAA</code>	Match AATT or TTAA
<code>\s</code>	Match whitespace

Regular expressions

Matching string patterns according to certain rules

<code>^ATG\$</code>	a) AATG	b) ATGGC	c) ATG
<code>ATG</code>	a) AATG	b) ATGGC	c) ATG
<code>[ATGC]{6}</code>	a) AAAAAA	b) NATCGGCN	c) GGCT
<code>^[ATGC]{6}\$</code>	a) AAAAAA	b) NATCGGCN	c) GGCT
<code>[^A]{3,5}</code>	a) AAA	b) GGG	c) CTACG
<code>^A*GCT</code>	a) GCT	b) GGCT	c) AAAGCT

Regular expressions

Matching string patterns according to certain rules

<code>^ATG\$</code>	a) AATG	b) ATGGC	c) ATG
<code>ATG</code>	a) AATG	b) ATGGC	c) ATG
<code>[ATGC]{6}</code>	a) AAAAAA	b) NATCGGCN	c) GGCT
<code>^[ATGC]{6}\$</code>	a) AAAAAA	b) NATCGGCN	c) GGCT
<code>[^A]{3,5}</code>	a) AAA	b) GGG	c) CTACG
<code>^A*GCT</code>	a) GCT	b) GGCT	c) AAAGCT

Pattern Search: `grep`

```
grep pattern file.txt # Returns lines matching a pattern
```

```
grep -v pattern file.txt # Returns lines not matching a pattern
```

```
grep -E regex file.txt # Returns lines not matching a regex
```

```
grep -c pattern file.txt # Returns number of lines matching a pattern
```

```
grep -B pattern file.txt # Returns number of lines before a line matching a pattern
```

```
grep -o pattern file.txt # Returns only matching part of lines
```

```
man grep # For other options
```

Word & line count: wc

```
wc file.txt # Returns number of bytes, words and lines
```

```
wc -c file.txt # Returns number of bytes (i.e. number of characters incl. \n)
```

```
wc -w file.txt # Returns number of words in a file
```

```
wc -l file.txt # Returns number of lines in a file
```

```
wc -l *.txt # Returns number of lines in all TXT files by file
```

Exercises

What is the number of SNPs in the VCF file?

What is the number of SNPs that passed or failed the filter?

What is the number of SNPs on chromosome Z that passed?

Retrieve & count unique records

Select columns

```
cut -f1-3 file.txt  
cut -d $'\t' -f1-3 file.txt  
cut --complement -f4 file.txt
```

Sorting data based on selected column

```
sort -k1,1 file.txt  
sort -k1,1 -k2,2nr file.txt  
sort -k1,3 file.txt
```

Retrieve unique records

```
sort -u file.txt  
< file.txt sort | uniq -c
```

Exercise

What is the number of SNPs per chromosome in the VCF file??

```
FILE=/data-shared/vcf_examples/luscinia_vars_flags.vcf.gz

< $FILE zcat |
grep -v '^#' |
cut -f1 |
sort |
uniq -c |
sort -k1,1n
```

Exercise

Get the first six base pairs from every read and calculate prevalence of the these kmers

```
cat *.fastq |  
grep -E "^[ACGT]+$" |  
cut -c1-6  
sort |  
uniq -c |  
sort -k1,1nr |  
less
```

String extraction and replacement

`tr` (TRansliterate)

- *Replaces or deletes **individual characters***
- *ideal for changing delimiters, removing line endings, uppercase to lowercase conversion*

`sed` (text Stream Editor)

- *Matches, replaces and extracts **complex patterns***
- *Useful for extraction of a value according a specific tag from a gff3 of vcf file*

`grep -o`

- *Returns only matching parts of the text*
- *Useful for extraction of repeating patterns (e.g. microsatellites)*

tr

```
tr ";" "\t" file.txt # Replace delimiter
```

```
tr -d "\n" file.txt # Remove line ending character
```

```
tr "[ATGCN]" "[atgcn]" file.txt # Uppercase to lowercase
```

sed

```
sed 's/pattern/replacement/'
```

Remove anything that is not ACGT at the beginning of line

```
sed 's/^[ACGTN]\{6\}/NNNNNN/'
```

The same thing using extended regular expressions

```
sed -r 's/^[ACGTN]{6}/NNNNNN/'
```

```
echo 'AAATTTCCCGGG' | sed -r 's/A+(T+)C+(G+)/\1\2/'
```

The result would be 'TTTGGG'

grep -o

Match AT di-nucleotide twice or more times

```
grep -o -E "(AT){2,}"
```

Match GTC tri-nucleotide twice or more times

```
grep -o -E "(GTC){2,}"
```

Match any repeating pattern

```
grep -o -E "([ATGC]{1,})\1+"
```

Exercise

What is the number of SNPs per chromosome in the VCF file??

...without using cut command:

```
FILE=/data-shared/vcf_examples/luscinia_vars_flags.vcf.gz

< $FILE zcat |
grep -o -E '^chr[Z1-9]+' |
sort |
uniq -c |
sort -k1,1nr
```


Exercise

Microsatellites statistics: Extract all AT dinucleotides repeating at least twice and calculate their frequency distribution in the whole dataset.

```
cat *.fastq |  
grep -E "^[ACGT]+$" |  
grep -o -E "(AT){2,}" |  
sort |  
uniq -c |  
less
```

Join and paste data

join

- *Joining two files based on a specific key column*
- *Corresponds to JOIN in SQL language*

paste

- *Simply aligns files by column*
- *No key column is needed*
- *Assumes one to one correspondence between the two datasets*

join

Join file1.txt and file2.txt based on 2nd and 3rd column

```
sort -k2,2 file1.txt > file1.tmp
```

```
sort -k3,3 file2.txt > file2.tmp
```

```
join -12 -23 file1.tmp file2.tmp > joined-file.txt
```

paste

```
# Merge vertically two files
```

```
paste file1.txt file2.txt > file-merged.txt
```

file1.txt

ID1

ID2

ID3

ID4

+

file2.txt

AATG

CAAG

ATCG

GTTG

=

file-merged.txt

ID1 AATG

ID2 CAAG

ID3 ATCG

ID4 GTTG

paste

```
# Transpose file  
< filte.txt paste - -
```

file1.txt

item-line1
item-line2
item-line3
item-line4

=

item-line1 item-line2
item-line3 item-line4

Exercise

Convert FASTQ file to TAB separated file with each read on one line

```
cat *.fastq |  
paste - - - - |  
cut --complement -f3 \  
> reads.tab
```

What did we learn today?

- Search a pattern (`grep` and regular expressions)
- Word and line count (`wc`)
- Retrieve & count unique records (`cut`, `sort`, `uniq`)
- String extraction & replacement (`tr`, `sed`)
- Join & paste data (`join`, `paste`)

Lunch, Lunch!!