Practical Bioinformatics

Regular Expressions

Stefan Wyder
stefan.wyder@uzh.ch
URPP Evolution
www.evolution.uzh.ch

Universität Zürich UZH

URPP

Example

```
>gi|16127999|ref|NP_414546.1|
MKKMQSIVLALSLVLVAPMAAQAAEITLVPSVKLQIGDRDNRGYYWDGGHWRDHGWWKQHYEWRGNRWHLHGPPPPPR
HHKKAPHDHHGGHGPGKHHR
>gi|16128000|ref|NP_414547.1|
MLILISPAKTLDYQSPLTTTRYTLPELLDNSQQLIHEARKLTPPQISTLMRISDKLAGINAARFHDWQPDFTPANARQAILAFKGD
VYTGLQAETFSEDDFDFAQQHLR
>gi|16128007|ref|NP_414554.1|
MKSVFTISASLAISLMLCCTAQANDHKLLGAIAMPRNETNDLALKLPVCRIVKRIQLSADHGDLQLSGASVYFKAARSASQSLNIP
SEIKEGQTTDWININSDNDNKRCVSKI
```

```
grep -c ">" proteins.fa
```

```
grep -c "^>" proteins.fa
```

Regular Expressions (regex/regexp)

- a simple language to describe a *set of strings*

- Powerful search and replace function
- Extract information (format conversion)
- Format checking

- available in
  - many Linux tools (g**re**p, sed, awk)
  - most programming languages
  - many text editors/OpenOffice

Literal characters

- the simplest regexps are literal characters:

  h matches the character 'h'

  bat matches 'bat'

- case-sensitive per default

Simple regexps

. matches any character

    .at matches any three-character string ending with at ('hat','cat','bat',..)

[hc] matches exactly one of the enclosed characters

    [hc]at matches 'hat' and 'cat'

    [^h]at matches all strings matched by .at except 'hat'

    [^hc]at matches all strings matched by .at except 'hat' and 'cat'

Character classes []

	Matches
[abcde]	exactly one of the characters listed
[a-e]	exactly one character in the given range
[^abcde]	any character not listed
[^a-e]	any character that is not in the given range
{URPP,evolution}	exactly one entire word from the options given

Range limits are defined according to the ASCI values

Wildcards

	means
.	any character [-.?+%$A-Za-z0-9...]
\d	digit [0-9]
\w	word character (alphanumerics or underscore)
\s	white space (space, tab, end-of-line)
\t	tab
\S	complement of \s: any non-whitespace character
\D \W	

Wildcards

5th
3rd
2nd
4th

A wildcard is a special character that represents a specific set of character

\d\w\w

\d matches any digit (0-9)

\w matches any letter (A-z) or digit (0-9) or underscore (_) [A-z0-9_]

Regexps are **non-overlapping**
(\w\w would match 5t 3r 2n)

Some examples

Regex	chr.	chr\d	chr[1-5]	AAF12\.[1-3]	AT[1,5]G\d+\.[1,2]
	chr1	chr1	chr1	AAF12.1	AT5G08160.1
	chr2	chr2	chr2	AAF12.2	AT5G08160.2
	chr3	chr3	chr3	AAF12.3	AT5G10245.1
	chr4	chr4	chr4		AT1G14525.1
	chr5	chr5	chr5		
	chr6	chr6			
	chrM				

Capturing text with ()

5th
3rd
2nd
4th

Search:	(\d)\w\w	Capture portions of the search with ()
Replace:	\1	Reuse captured text with \1

5
3
2
4

Matching once or more

\w+ matches until the next non-word character (e.g. space, punctuation, end of line)

Agalma elegans
Frillagalma vitiazi
Mus musculus

Search: (\w)\w+ (\w+)
Replace: \1. \2

A. elegans
F. vitiazi
M. musculus

Quantifiers

	means
*	zero or more times
+	one or more times
{n}	exactly n times
{m,n}	at least m times but no more than n times

[Nn]ick matches 'Nick', 'nick'
[Nn]*ick matches 'ick', 'Nick', 'nick', 'NNick', 'Nnick', 'Nnick, 'nnick', ...
[Nn]+ick matches 'Nick', 'nick', 'NNick', 'Nnick', 'nnick', 'NNNick', ...
[Nn]{2}ick matches 'NNick', 'nnick', 'Nnick', 'nNick'

Regexps match the first instance

```
Agalma,A. elegans,hydrozoan,316164
Frillagalma,F. vitiazi,hydrozoan,645341
Mus,M. musculus,rodent,10088
```

```
([^,]+),([^,]+),[^,]+,([^,]+)
\3\t\1\t\2
```

```
hydrozoan  Agalma       316164
hydrozoan  Frillagalma  645341
rodent     Mus          10088
```

4 columns: this regexp will match all 4

5 columns: leaves 5th column untouched

<4 columns: no match

8 columns: this regexp will match twice

* and + are greedy

They match the maximum number of characters they can (from left to right)

abcdefgabc

Search: (a.*c)
Replace: \1

abcdefgabc **NOT abc !!**

Use the lazy quantifier '?' so that the expression tries the minimal match first

Search: (a.+?c)
Replace: \1

abc

Anchoring

^[hc]at matches 'hat' and 'cat', but only at the line beginning

[hc]at$ matches 'hat' and 'cat', but only at the line end

Several regexp dialects

POSIX Basic (BRE)	POSIX Extended (ERE)	Perl-compatible (PCRE)
	adds ? + |	extension of ERE
[:word:] [:space:] [:digit:]	[:word:] [:space:] [:digit:]	\w \s \d
Operators {} () must be escaped with \ sub("abc", "\(a.*.c\)", "\1")	Literal chars {} () + | ? must be escaped with \ sub("abc", "(a.*.c)", "\1")	Literal chars {} () + | ? must be escaped with \ sub("abc", "(a.*.c)", "\1")
grep (default) sed (default)	egrep / grep -E sed -r in R: default	python re grep -P in R: sub(..., perl=TRUE)

Sources & Links

General (incl Linux, Python, regexps, databases)

- Haddock&Dublin. Practical Computing for Biologists. Sinauer Associates 2011.

regular expressions

- online tool to build&learn http://www.regexr.com/
- Cheatsheet practicalcomputing.org/files/PCfB_Appendices.pd
- http://stackoverflow.com/questions/4736/learning-regular-expressions