

Algoritmos para Análise de Sequências Biológicas

Expressões regulares e motifs

Sumário

- Expressões regulares
- Procura de padrões fixos ou através de expressões regulares

Procura de padrões fixos

- `in` operador que verifica se uma substring ocorre dentro de outra
- `find` método das strings que devolve o primeiro índice de ocorrência da substring na string
- `rfind` método das strings que devolve o último índice de ocorrência da substring na string
- `count` método das strings que conta o nº de ocorrências de uma substring numa string

Expressões regulares

- Linguagem que permite especificar um conjunto de strings
- Permite especificar um padrão que pode ser utilizado para procurar em strings
- Pode-se utilizar uma expressão regular para:
 - ▶ verificar se um padrão ocorre numa string
 - ▶ obter a sua posição e a substring correspondente
 - ▶ obter todas as ocorrências do padrão na string
- Expressões regulares são úteis para:
 - ▶ tratamento de textos
 - ▶ parsing de resultados de ferramentas
- Consulte o [manual](#) para aprofundar a utilização de expressões regulares

Expressões regulares

Padrão	Significado
.	Qualquer caractere
()	Usado para agrupar a expressão regular que se encontra dentro e marca um grupo
\número	n-ésimo grupo
\d	Algarismo
\s	Whitespace
\w	Caracteres contidos em palavras
\D	Oposto de \d
\S	Oposto de \s
\W	Oposto de \w
\b	Início ou fim de uma palavra
\	Remove o significado especial de um caractere
^	Início de uma string
\$	Fim de uma string
	Disjunção de expressões regulares

Metacarateres

Padrão	Significado
*	0 ou mais ocorrências do padrão anterior
+	1 ou mais ocorrências do padrão anterior
?	0 ou 1 ocorrências do padrão anterior
?	Torna o metacaratere anterior não <i>greedy</i>
{n}	n ocorrências do padrão anterior
{m, n}	Entre m e n ocorrências do padrão anterior. A omissão do m equivale a 0 e a omissão do n equivale a infinito

Classes de caracteres

- Delimitados por []
- Se o primeiro caractere for ^ indica o oposto
- Se o caractere - existir no meio, indica uma gama de caracteres
- As classes de caracteres \d \s \w e as suas complementares podem ser utilizadas

Significados especiais

- (?:...) Versão que não captura
- (?=...) Versão que faz match com ... mas não consome caracteres
- (?!...) Versão que não faz match com ... mas não consome
- (?<=...) Verifica se ... ocorre **antes**
- (?<!...) Verifica se ... **não** ocorre antes

Exemplos de classes de caracteres

[A-Z] maiúsculas

[0-9] o mesmo que \d

[^0-9] o mesmo que \D

[ACTGactg] base de DNA

[ACDEFGHIKLMNPQRSTVWY] aminoácido

Métodos

- `match` só no início da string
- `search` procura expressão regular na string
- `findall` todas as ocorrências
- `finditer` devolve iterador que procura em todas as ocorrências
 - `sub` substitui
 - `split` separa por expressão regular
- `compile` compila a expressão regular

Exemplos

```
>>> import re
>>> re.match(r'\d+', 'a23b67')
>>> re.search(r'\d+', 'a23b67')
<re.Match object; span=(1, 3), match='23'>
>>> re.findall(r'\d+', 'a23b67')
['23', '67']
>>> for m in re.finditer(r'[0-9]+', 'a23b67'): print(m)
<re.Match object; span=(1, 3), match='23'>
<re.Match object; span=(4, 6), match='67'>
>>> [m.group() for m in re.finditer(r'\d+', 'a23b67')]
['23', '67']
>>> [m.span() for m in re.finditer(r'\d+', 'a23b67')]
[(1, 3), (4, 6)]
>>> re.search(r'a{3,5}', 'aaaaaa')
<re.Match object; span=(0, 5), match='aaaaa'>
>>> re.search(r'a{3,5}?', 'aaaaaa')
<re.Match object; span=(0, 3), match='aaa'>
>>> re.findall('A[AGC]TT', 'AATTTAGTTGACTTG')
['AATT', 'AGTT', 'ACTT']
```

Exemplos

```
>>> re.split(r'\d+', 'xyz123abc456')
['xyz', 'abc', '']
>>> re.split(r'(\d+)', 'xyz123abc456ttt')
['xyz', '123', 'abc', '456', 'ttt']
>>> re.findall('(ACC)|(G[AC]C)', 'AACCGGACT')
[('ACC', ''), ('', 'GAC')]
>>> re.findall('(?:ACC)|(?:G[AC]C)', 'AACCGGACT')
['ACC', 'GAC']
>>> re.findall(r'...', 'AATTTAGTTGAC')
['AAT', 'TTA', 'GTT', 'GAC']
>>> re.findall(r'(?=(...))', 'AATTTAGTTGAC')
['AAT', 'ATT', 'TTT', 'TTA', 'TAG', 'AGT', 'GTT', 'TTG', 'TGA', 'GAC']
>>> re.findall('M.*?_', 'AMT_A_MAMA_T')
['MT_', 'MAMA_']
>>> er = re.compile('(TATA..)((GC){3})')
>>> er.search('ATATAAGGCGCGCGCTTATGCGC')
<re.Match object; span=(1, 13), match='TATAAGGCGCGC'>
```

Exemplos

```
>>> re.search(r'[ACGT]+', 'agtca', flags = re.I)
<re.Match object; span=(0, 5), match='agtca'>
>>> re.search(r'(?i)[ACGT]+', 'agtca')
<re.Match object; span=(0, 5), match='agtca'>
>>> re.sub(r'\d+', 'XXX', 'a123b76')
'aXXXbXXX'
>>> re.sub(r'\d+', lambda m: m.group(), 'a123b76')
'a123b76'
>>> re.sub(r'\d+', lambda m: m.group() * 3, 'a123b76')
'a123123123b767676'
>>> re.sub(r'\d+', lambda m: eval(f"str({m.group()} * 3)"), 'a123b76')
'a369b228'
re.sub(r'(\w+)(\d+)', r'\2\1', 'qw123rt4')
'4qw123rt'
```

Exemplos

```
>>> texto = """
>gi|1322285|gb|AAC03525.1| eukaryotic initiation factor 4E-I
MQSDFHRMKNFANPKSMFKTSAPSTEQGRPEPPTSAAAPAEAKDVKPKEDPQETGEPAGN
TATTTAPAGDDAVRTEHLYKHPLMNVTWLWYLENDRSKSWEDMQNEITSFDTVEDFWSLY
NHIKPPSEIKLGSYSLFKKNIRPMWEDAAN"""
>>> m = re.search('^>(.*?)$(.*)', texto, flags = re.DOTALL | re.MULTILINE)
>>> print(m.group(0))
>gi|1322285|gb|AAC03525.1| eukaryotic initiation factor 4E-I
MQSDFHRMKNFANPKSMFKTSAPSTEQGRPEPPTSAAAPAEAKDVKPKEDPQETGEPAGN
TATTTAPAGDDAVRTEHLYKHPLMNVTWLWYLENDRSKSWEDMQNEITSFDTVEDFWSLY
NHIKPPSEIKLGSYSLFKKNIRPMWEDAAN
>>> print(m.group(1))
gi|1322285|gb|AAC03525.1| eukaryotic initiation factor 4E-I
>>> print(m.group(2))

MQSDFHRMKNFANPKSMFKTSAPSTEQGRPEPPTSAAAPAEAKDVKPKEDPQETGEPAGN
TATTTAPAGDDAVRTEHLYKHPLMNVTWLWYLENDRSKSWEDMQNEITSFDTVEDFWSLY
NHIKPPSEIKLGSYSLFKKNIRPMWEDAAN
```

Motifs

- padrão existente numa sequência biológica e que é partilhado por várias sequências dada a função biológica associada
- Exemplos:
 - DNA : locais de ligação de proteínas (ou outros elementos) regulatórios ao DNA, controlando a transcrição
 - RNA : representação de padrões de microRNAs
 - Proteínas : representação de domínios conservados de proteínas com funções biológicas determinadas (e.g. locais de ligação de enzimas a substratos ou outras moléculas)

Padrões

Perfis estocásticos definem probabilidades por posição

Padrões com espaçamentos, representados por expressões regulares

Blocos/Motifs sem espaçamentos, representados por strings

Padrões determinísticos do PROSITE

Padrão	Significado
x	Qualquer aminoácido
[]	Qualquer destes aminoácidos
{}	Todos os aminoácidos menos um destes
(m)	Exatamente <i>m</i> cópias
(m, n)	Entre <i>m</i> e <i>n</i> cópias
-	Usado para separar padrões

Exemplos

Padrão

Expressão regular

F-[GSTV]-P-R-L-G

F[GSTV]PRLG

[AC]-x-V-x(4)-{ED}

[AC].V.{4}[^ED]

A-x-[ST](2)-x(0,1)-V

A.[ST]{2}.?V

C-x-H-x-[LIVMFY]-C-x(2)-C-[LIVMYA]

C.H.[LIVMFY]C.{2}C[LIVMYA]

Enzimas de restrição

- São proteínas que cortam o DNA em zonas que contenham sub-sequências específicas
- Exemplo: EcoRI – corta DNA em zonas que tenham o padrão GAATTC (cortando entre o G e o primeiro A)
- Estas enzimas cortam ambas as cadeias (uma vez que são normalmente “palíndromos biológicos” – iguais aos seus complementos inversos). Assim, deixam um “overhang” (bases não complementadas) em cada cadeia (tornando possível reconstruir os fragmentos por exemplo para clonagem ou sequenciação)
- Assim, os mapas de restrição para sequências de DNA (localizações onde enzimas específicas cortam) são ferramentas úteis em biologia molecular

Representação

- Base de dados REBASE inclui lista de enzimas de restrição e seus padrões de corte
- O caractere ^ representa o ponto de corte
- Eis os IUB ambiguity codes:

R G ou A

Y C ou T

M A ou C

K G ou T

S G ou C

W A ou T

B não A

D não C

H não G

V não T

N A ou C ou G ou T

Exemplo

Enzima

G[^]AMTV

Sequência

TAAGACTGAAT

Corte

TAAG ACTGAAT