Algoritmos para Análise de Sequências Biológicas

Ficha 8

Objetivo

- PWMs e PSSMs
- Alinhamentos progressivos

PWMs e PSSMs

- Crie uma função chamada `pwm(alinhamento, pseudo = 0)` que recebe uma lista que representa um alinhamento entre várias sequências e devolve a PWM na forma de uma lista de dicionários em que as chaves são as bases e os valores são as probabilidades. O valor por omissão da pseudocontagem deve ser de zero;
- Crie uma função chamada `prob_seq(seq, pwm)` que recebe uma sequência do mesmo tamanho que a PWM e que devolve a probabilidade desta ter sido gerada pela PWM;
- Crie uma função chamada `seq_mais_provavel(seq, pwm)` que recebe uma sequência e uma PWM e que devolve a subsequência mais provável de ter sido gerada pela PWM.
- Crie uma função chamada `pssm(alinhamento, pseudo = 1)` que recebe uma lista que representa um alinhamento entre várias sequências e devolve a PSSM correspondente. O valor por omisão da pseudocontagem deve ser de um;

Código providenciado

```
def print_profile(perfil):
    """Imprime um perfil probabilístico de forma legível

        @param perfil O perfil probabilístico
    """
    bases = sorted(perfil[0].keys())
    tab = [[f"{p[b]:-5.2f}" for b in bases] for p in perfil]
    for p in zip(*([bases] + tab)):
        print(*p)
```

Exemplo

```
>>> P = pwm(['ATTG','ATCG','ATTC','ACTC'], pseudocount = 0.5)
>>> print_profile(P)
A  0.75  0.08  0.08  0.08
C  0.08  0.25  0.25  0.42
G  0.08  0.08  0.08  0.42
T  0.08  0.58  0.58  0.08
>>> prob_seq("ACCG", P)
0.01953125
>>> seq_mais_provavel("TACCGTGCA", P)
'ACCG'
>>> print_profile(pssm(['ATTG','ATCG','ATTC','ACTC'], pseudocount = 0.5))
A  1.58 -1.58 -1.58 -1.58
C -1.58  0.00  0.00  0.74
G -1.58 -1.58 -1.58  0.74
T -1.58  1.22  1.22 -1.58
```

Algoritmo de Alinhamento Progressivo

1. Alinhar as duas primeiras sequências utilizando Needleman/Wunshc
2. Adicionar as sequências alinhadas ao alinhamento múltiplo
3. Enquanto existirem mais sequências para alinhar
    1. Calcular o consenso do alinhamento
    2. Alinhar a sequência de consenso com a nova sequência utilizando Needleman/Wunsch
    3. Adicionar ao alinhamento múltiplo
4. Percorrer as sequências e corrigir os possíveis alinhamentos tendo em conta o consenso final

Alinhamento Progressivo

- Implemente a função `alinhamento(s1, s2)` que recebe duas sequências e devolve um tuplo com as duas sequências alinhadas;
- Implemente a função `consenso(s1, s2)` que recebe duas sequências alinhadas e devolve a sequência de consenso;
- Implemente a função `alinhamento_progressivo(seqs)` que recebe uma lista de sequências e que as alinhe progressivamente utilizando a ordem das sequências na lista.

Exemplo

Nestes exemplos utilizou-se o score de 4 para carateres iguais de -1 para substituições e de -4 para espaçamentos.

```
>>> alinhamento_progressivo("TGACTA TACGTA TGGTA GAT".split())
TGAC-TA
T-ACGTA
TG--GTA
-GA--T-

>>> alinhamento_progressivo("GTTGCACCA GTCAGCA TTCCCA GCAGA".split())
GTTGCACCA
G-T-CAGCA
-TT-C-CCA
---GCA-GA

>>> alinhamento_progressivo("ACTCAT AGTCAT ACGTCCT".split())
A-CTCAT
A-GTCAT
ACGTCCT
```