Ana Claudia Mancusi Valeije

Bacharel em Ciências Moleculares pela Universidade de São Paulo e mestranda em Bioinformática, também pela Universidade de São Paulo. Desenvolve pesquisa na área de predição de sítios de ligação de fatores de transcrição, utilizando como modelo alguns receptores nucleares (COUP-TF e ER). A pesquisa tem por objetivo desenvolver um método computacional capaz de analisar a probabilidade de uma sequência de DNA ser um sítio de ligação de um determinado fator de transcrição através do estudo de variáveis como energia de ligação do complexo DNA-fator de transcrição. Orientador: Paulo Sergio Lopes de Olievira, LNBio (Campinas -SP).

Meus Exercícios

Linque para Exercícios da Ana

Propostas de Trabalho Final

Plano A

Uma função que varre uma sequência de caracteres (ou números) com uma janela de tamanho especificado e atribui a cada subsequência lida um valor. Este valor pode ser calculado através de uma função já existente no R ou através de um dataframe passado para a função, que atribua valores a todas as sequências possíveis com o tamanho da janela. Os argumentos da função seriam: um vetor com a sequência, um número com o tamanho da janela, um número com o tamanho do passo da leitura, um booleano dizendo se deve plotar ou não o vetor de resultados, um dataframe com a relação sequência-valor, uma função a ser aplicada na janela. A função deve receber todos os parâmetros, com exceção do dataframe e da função a ser aplicada: a função deve receber apenas um destes. Ela retorna um vetor com os resultados.

Exemplos de aplicação:

→ Em gráficos que apresentam muito ruído, como os de simulação computacional de moléculas, é interessante calcular a média dos pontos dentro de uma janela, o que diminui o ruído e facilita encontrar padrões nos dados.

→ Com um dataframe que relacione códons e anti-códons ou códons e aminoácidos, seria possível por exemplo transcrever uma sequência de DNA ou traduzir uma de RNA.

→ No caso do meu projeto, tenho um arquivo que relaciona cada sequência de DNA possível com 6 pares de base (4^6 = 4096 sequências) a uma energia (energia de ligação entre um fator de transcrição específico e aquela sequência de DNA). Seria possível passar uma sequência grande de DNA para essa função, junto deste arquivo de energias, para varrer a sequência em janelas de 6

nucleotídeos e criar um gráfico com o resultado. Este gráfico pode revelar padrões para sítios de ligação deste fator de transcrição.

Plano B

Fazer a mesma função acima, mas que só receba o dataframe de comparação, e não uma função a ser aplicada nas sequências varridas.

Comentários

Minha dica é começar com o plano e B e expandir caso haja tempo. Parece que a implementação não é tão simples, mas imagino que já tenha experiencia com programação já que vem da Moleculares. Pelo que entendi vc. no exemplo do seu trabalho terá uma janela de 6 pares de base... e cada uma deve ser relacionada a 4096. Imagino que queira fazer todas as possibilidades de janelas de 6, o que e o mesmo que quebrar de seis em seis todas a sequencia e depois, pular a primeira base e fazer o mesmo.. até a quinta! Essa quebra pode ser feita pelas funções de separar strings de caracteres como o substring(), em seguida pode usar match() para associar a sequencia observada com as sequencias do objeto onde há o valor da ligação... — *Alexandre Adalardo de Oliveira 2012/04/03 15:28*

Página de Ajuda

```
scan.seq        package:none        R Documentation

Varre uma sequência de caracteres ou números, atribuindo valores às janelas
de leitura

Descrição:

    Varre uma sequência de caracteres ou um vetor de números utilizando uma
janela de leitura e um passo de varredura dados. Atribue valores às janelas
a partir de um dataframe ou de uma função fornecidos.

Uso:

    scan.seq(sequence, scores = NA, FUN = NA, step = 1, window = 1)

Argumentos:

    sequence: Um vetor de caracteres ou de números

    scores: Um data frame atribuindo valores (de quaisquer tipos) às
subsequências de sequence de tamanho window. Utilizado apenas quando
"sequence" é um vetor de caracteres. Ou "scores" ou "FUN" deve ser
```

```
fornecido.

    FUN: A função a ser aplicada às janelas. Pode ser usada tanto para
vetores de caracteres quanto para vetores de números. Ou "scores" ou "FUN"
deve ser fornecido.

    step: O passo (inteiro) com o qual a janela será movida na varredura da
sequência.

    window: O tamanho da janela (inteiro) usada para varrer a sequência.

    ...: quaisquer argumentos a repassar para a função FUN

Valor:
    Se "sequence" é um vetor de caracteres, retorna um data frame de duas
variáveis: as janelas lidas e seus respectivos valores, calculados a partir
de "scores" ou de "FUN".
    Se "sequence" é um vetor de números, retorna um vetor de números com os
resultados das varreduras.

Autor(es):
    Ana Claudia Mancusi Valeije
    ana.valeije@gmail.com

Ver também:
    "sweep"

Exemplos:
    # Tradução de uma sequência de mRNA nos 3 frames de leitura:
    sequencia.mrna = "AAUGGUAGCAAUCGAUCGUUGACU" # Sequência de mRNA com 24
bases
    bases = c("A", "C", "G", "U")
    codons = paste(rep(bases,each=4*length(bases)), rep(bases, each =
length(bases)),bases,sep="")
    aminoacidos = c("K", "N", "K", "N", rep("T", 4), "R", "S", "R", "S",
"I", "I", "M", "I", "Q", "H", "Q", "H", rep("P", 4), rep("R", 4), rep("L",
4), "E", "D", "E", "D", rep("A", 4), rep("G", 4), rep("V", 4), "Z", "Y",
"Z", "Y", rep("S", 4), "Z", "C", "W", "C", "L", "F", "L", "F")
    tabela = data.frame(codons, aminoacidos, stringsAsFactors = FALSE)
    colnames(tabela) = c("codons", "aminoacidos")
    frame1 = scan.seq(sequencia.mrna, scores = tabela, step = 3, window = 3)
    frame1
    frame2 = scan.seq(substr(sequencia.mrna, 2, 24), scores = tabela, step =
3, window = 3)
    frame2
    frame3 = scan.seq(substr(sequencia.mrna, 3, 24), scores = tabela, step =
3, window = 3)
    frame3

    # Retirada de ruído
    dados = 100:199
```

```
    dados = jitter(dados, 50) # adicionando ruído
    par(mfrow = c(1,2))
    plot(dados)
    dados.scan = scan.seq(dados, FUN=mean, step = 1, window = 10)
    plot(dados.scan)
```

Código da Função

```
scan.seq = function(sequence, scores = NA, FUN = NA, step = 1, window = 1,
...) {
  if (class(scores) == "logical" & class(FUN) == "logical") {
    stop("scores or FUN must be given")
  }
  if (class(sequence) == "character") {
    scan = vector()
    values = vector()
    len = 0
    len.seq = nchar(sequence)
    for (i in seq(1, len.seq - (window - 1), step)) {
      len = len + 1
      scan[len] = substr(sequence, i, i + window - 1)
    }
    len = 0
    if (class(scores) == "logical") {
      for (i in 1:length(scan)) {
        len = len + 1
        values[len] = FUN(scan[i], ...)
      }
      results = data.frame(scan, values)
    }
    else {
      for (i in 1:length(scan)) {
        len = len + 1
        values[len] = scores[scores[,1] == scan[i], 2]
      }
      results = data.frame(scan, values)
      colnames(results) = colnames(scores)
    }
    return(results)
  }
  else if (class(sequence) == "numeric" | class(sequence) == "integer") {
    len.seq = length(sequence)
    scan = list()
    values = vector()
    len = 0
    for (i in seq(1, len.seq - (window - 1), step)) {
      len = len + 1
      scan[[len]] = sequence[i:(i + window - 1)]
```

```
    }
    len = 0
    if (class(scores) == "logical") {
      for (i in 1:length(scan)) {
        len = len + 1
        values[len] = FUN(scan[[i]], ...)
      }
    }
    return(values)
  }
}
```

Arquivo da Função

scan.seq.r