Fábio Henrique Kuriki Mendes

Sou bacharel em Biologia pela USP e anseio pela prova do mestrado da genética. Também anseio por finalizar a licenciatura antes que o inverso se concretize.

O Prof. Dr. Diogo Meyer me orientará em meu mestrado, no qual estudarei os sinais de seleção natural incidente em genes do HLA e regiões adjacentes, procurando compreender como o desequilíbrio de ligação entre eles afeta a evolução desse complexo e sua associação a doenças.

Meus Exercícios

Resposta da lista de exercicios 1

Resposta da lista de exercicios 2

Resposta da lista de exercicios 3

Resposta da lista de exercicios 4

Resposta da lista de exercicios 5

Resposta da lista de exercicios 6

Resposta da lista de exercicios 7

Resposta da lista de exercicios 8

Resposta da lista de exercicios 9

Proposta de Trabalho Final

Principal

Informações iniciais:

O HapMap é um projeto internacional que busca encontrar variantes genéticas associadas a doenças humanas. Eles têm como dados brutos dados populacionais humanos (populações africanas, européias, etc) constituídos por dataframes, por cromossomo, com todos os *Single Nucleotide Polymorphisms* (nucleotídeos polimórficos dentro daquela população) detectados através de alinhamento de sequências).

É possível utilizar as frequências alélicas de cada SNP para calcular as taxas de heterozigose populacional e total para aquele polimorfismo específico. Usando essas taxas de heterozigose se compõem o índice Fst, que basicamente nos diz o quanto as populações diferem para aquele polimorfismo. Quanto maior o Fst, mais diferentes são as populações. Teremos então um Fst para

cada SNP e, a partir de todos os SNPs, temos uma distribuição de Fst's.

Podemos então comparar a distribuição de Fst's de diferentes grupos de SNPs. Exemplo: SNPs em regiões gênicas específicas e SNPs em regiões não-gênicas neutras. Se esperamos que um gene está evoluindo por seleção natural positiva, espera-se que a distribuição do Fst dos SNPs dentro de seus limites seja diferente (com valores maiores) da distribuição do Fst dos SNPs em regiões não-gênicas neutras. Por outro lado, se imaginamos que um gene evolui por seleção balanceadora, espera-se que os Fsts de seus SNPs sejam significativamente mais baixos do que os Fsts de regiões não-gênicas neutras.

Minha função:

O **objetivo primário** é calcular o índice Fst através da razão Ht-Hs/Ht (explicada abaixo) para um grupo de SNPs de interesse em diferentes populações humanas.

Para atingir o objetivo a função deve:

1. Reorganizar os dados pré-organizados* da base de dados HapMap (esses dados consistem em uma lista contendo a tabela de cada população comparada somente com os dados necessários) em uma única tabela; para ver como são os dados *bulk* do HapMap, (clique aqui);
2. Usando as colunas certas, calcular o índice Fst;
3. Avisar se as tabelas têm as mesmas dimensões;
4. Verificar se o arquivo de entrada foi corretamente preparado (o arquivo de entrada será uma lista com os dataframes das diferentes populações).
5. Fazer um histograma com os valores do Fst de todos os SNPs para cada população.

Coluna1: rsid do SNP (identificação daquele SNP específico)

Coluna2(*²): refallelecount (contagem bruta de um dos alelos existentes para aquele SNP - aqui é o alelo referência)

Coluna3(*²): otherallelecount (contagem bruta do outro alelo para aquele SNP)

Coluna4: totalcount (número de alelos amostrados)

Coluna5: taxa de heterozigose daquela população por SNP (Hs)

cálculo: 2 * (contagem bruta do alelo 1/numero total de alelos amostrados para esse SNP) * (contagem bruta do alelo 2/numero total de alelos amostrados para esse SNP); é basicamente o cálculo da taxa de heterozigose sob a premissa de equilíbrio de Hardy-Weinberg (2pq)

Coluna6: taxa de heterozigose total, envolvendo todas as populações escolhidas (Ht)

cálculo: mesma conta de Hs, só que usando os dados de todas as populações juntos

Para preencher as colunas 5 e 6 a função precisa antes juntar as tabelas de todas as populações em uma só de modo que ela consiga calcular o Ht.

---

* Para pré-organizar os dados *bulk* é preciso utilizar uma função simples (cujo código está lá embaixo junto com o código da função do trabalho final) para os dados de cada população e criar uma lista contendo os objetos resultantes dessa função.

*[2]SNPs na grandíssima maioria dos casos são bialélicos pelo simples motivo de que não é trivial termos uma mutação diferente das já existentes ocorrendo naquela exata posição. Não só isso, mas essa terceira mutação (terceiro alelo) deveria se tornar frequente o suficiente para ser detectada na pequena amostragem feita pelo HapMap (aproximadamente 20 indivíduos por população).

Comentários PI

Uma das melhores propostas que li. Super clara, precisa e detalhada. Vc sabe o que quer e como chegar lá, manda bala! Nem vou reclamar da falta de Plano B :)

Página de Ajuda

```
fst                package:nenhum                R Documentation


Distribuição dos Fst's para um conjunto de SNPs


Description:

     Calcula o Fst de cada um dos SNPs de um determinado conjunto para um
dado número de populações. Cria um histograma para a distribuição desses
Fst's.

Usage:

     fst(x, pops=3, l=10)

Arguments:

 x: Lista cujos elementos são os dataframes das populações comparadas
contendo as informações de frequência dos SNPs (ver detalhes).

pops: Numérico. Número de populações comparadas

l: Numérico. Número de SNPs para os quais se calculará o Fst.


Details:

     Os SNPs devem ser os mesmos para cada população e o número de SNPs por
população também deve ser o mesmo.
```

```
Warning:

     A função 'fst' não funcionará se:
     1) O arquivo de entrada não for uma lista
     2) Se a dimensão dos dataframes do arquivo de entrada não for a mesma
ou se o número de SNPs especificado estiver incorreto
     3) Se o número de populações especificado estiver incorreto

Author(s):

     Fábio Henrique Kuriki Mendes
References:

     Barreiro, L.B., Laval, G., Quach, H., Patin, E. & QuintanaMurci, L.
Natural selection has driven population differentiation in modern humans.
Nat. Genet 40, 340-345(2008).

See Also:

     Função 'dfo' para criar os dataframes do arquivo de entrada a partir do
bulk do HapMap e função 'list' para criar o arquivo de entrada da função
'fst' descrita aqui.

Examples:

     rs<-seq(1:10)
     refallele_count<-c(4, 3, 2, 4, 1, 3, 5, 8, 8, 9)
     refallele_freq<-refallele_count/10
     otherallele_count<-c(6, 7, 8, 6, 9, 7, 5, 2, 2, 1)
     otherallele_freq<-otherallele_count/10
     totalcount<-rep(10, 10)
     Hpop<-c(0.48, 0.42, 0.32, 0.48, 0.18, 0.42, 0.5, 0.32, 0.32, 0.18)
     tab1<-data.frame(rs, refallele_count, refallele_freq,
otherallele_count, otherallele_freq, totalcount, Hpop)
     refallele_count<-c(4, 1, 1, 4, 2, 6, 7, 5, 3, 2)
     otherallele_count<-c(6, 9, 9, 6, 8, 4, 3, 5, 7, 8)
     refallele_freq<-refallele_count/10
     otherallele_freq<-otherallele_count/10
     Hpop<-c(0.48, 0.18, 0.18, 0.48, 0.32, 0.48, 0.42, 0.5, 0.42, 0.32)
     tab2<-data.frame(rs, refallele_count, refallele_freq,
otherallele_count, otherallele_freq, totalcount, Hpop)
     dadosteste<-list(tab1, tab2)
     fst(dadosteste, pops=2, l=10)
```

Código da Função

```
## FUNÇÃO DE CÁLCULO DE Fst
```

```
## TRABALHO FINAL: FABIO HENRIQUE KURIKI MENDES



fst<-function(x, pops, l)

{

    if(class(x)=="list")

        {

        if((length(x)!=pops)==TRUE)

            {

            cat("\n #ERRO# Nada foi feito: o número de populações
especificado está incorreto \n")

            }

        else

            {

            c(l, 7)->dimensao

            vetor.dim<-rep(NA, pops)

            for(i in 1:pops)

                {

                sum(dim(as.data.frame(x[i])==dimensao))->vetor.dim[i]

                }

            if((sum(vetor.dim)<(2*pops))==TRUE)

                {

                cat("\n #ERRO# Nada foi feito: um dos três erros a seguir
podem ter ocorrido: \n 1- Suas tabelas nao tem as mesmas dimensoes \n 2- Uma
de suas tabelas é na verdade um vetor \n 3- Você especificou um número
errado de SNPs \n")

                }

            else

                {
```

```
                as.data.frame(x[1])->pop1

                pop1[,1]<-as.character(pop1[,1])

                data.frame(pop1[-1], row.names=pop1[,1])->df.final

                Hspops<-data.frame(rep(NA, l))

                Hspops[1]<-df.final[,6]

                for(i in 2:pops)

                    {

                    df.final<-cbind(df.final, as.data.frame(x[i])[-1])

                    Hspops[i]<-df.final[,(6*i)]

                    }

                Hs<-rep(NA, l)

                for(i in 1:l)

                    {

                    Hs[i]<-mean(mean(Hspops[i,]))

                    }

                df.final[,((pops*6)+1)]<-Hs

                pt.vetor<-rep(NA, l)

                pt.vetor<-df.final[,1]

                qt.vetor<-rep(NA, l)

                qt.vetor<-df.final[,3]

                tc.vetor<-rep(NA, l)

                tc.vetor<-df.final[,5]

                for(i in 2:pops)

                    {

                    pt.vetor<-df.final[,((i-1)*6+1)]+pt.vetor
```

```
                    qt.vetor<-df.final[,((i-1)*6+3)]+qt.vetor

                    tc.vetor<-df.final[,((i-1)*6+5)]+tc.vetor

                    }

                df.final[,((pops*6)+2)]<-pt.vetor

                df.final[,((pops*6)+3)]<-qt.vetor

                df.final[,((pops*6)+4)]<-tc.vetor

                Ht<-2 * (pt.vetor/tc.vetor) * (qt.vetor/tc.vetor)

                df.final[,((pops*6)+5)]<-Ht

                Fst<-(df.final[,((pops*6)+5)] - df.final[,((pops*6)+1)]) /
df.final[,((pops*6)+5)]

                df.final[,((pops*6)+6)]<-Fst

                names(df.final)[((pops*6)+1)]<-"Hs"

                names(df.final)[((pops*6)+2)]<-"refalelle_count"

                names(df.final)[((pops*6)+3)]<-"otheralelle_count"

                names(df.final)[((pops*6)+4)]<-"totalcount_total"

                names(df.final)[((pops*6)+5)]<-"Ht"

                names(df.final)[((pops*6)+6)]<-"Fst"

                print(df.final)

                x11()

                hist(Fst, nclass=l, xlim=range(Fst))

                }

            }

        }

    else

        {

        cat("\n #ERRO# Nada foi feito: seus dados não são uma lista \n")
```

```
        }

}
```

Arquivo da Função

Arquivo .R com a função dfo() para organizar os dados de entrada da função fst()

Arquivo .R com a função fst() (função do trabalho final)

From:
http://ecor.ib.usp.br/ - **ecoR**

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2010:alunos:trabalho_final:binhologia:start

Last update: **2020/08/12 06:04**