Evolution

```
evol = function (mod, gen, f, pop, N, wAA, wAa, waa, p = FALSE) #1 Cria
funcao "evol" com os argumentos "mod", "gen", "f", "pop", "N", "wAA", "wAa",
"waa" e "p".
{
  #VERIFICANDO OS PARAMETROS COMUNS
  if (mod != "d" && mod != "s") #2 Verifica se "mod" foi inserido
corretamente.
  {
    stop ("mod só pode ser d ou s.") #3 Senao, interrompe a funcao e exibe
mensagem para o usuario.
  }
  if (gen != round (gen) | gen <= 0) #4 Verifica se "gen" eh um numero
inteiro e maior que zero.
  {
    stop ("gen deve ser um numero inteiro e > 0.") #5 Senao, interrompe a
funcao e exibe mensagem para o usuario.
  }
  if (pop != round (pop) | pop <= 0) #6 Verifica se "pop" eh um numero
inteiro e maior que zero.
  {
    stop ("pop deve ser um numero inteiro e > 0.") #7 Senao, interrompe a
funcao e exibe mensagem para o usuario.
  }
  if (p != FALSE) #8 Verifica se o usuario inseriu algum valor para "p".
  {
    if (p < 0 | p > 1 ) #9 Se sim, verifica se "p" esta no intervalo
adequado para a funcao.
    {
      stop ("p deve estar no intervalo 0 <= p <= 1.") #10 Senao, interrompe
a funcao e exibe mensagem para o usuario.
    }
  }
  #FUNCAO DE DERIVA GENETICA
  if (mod == "d") #11 Se o argumento "mod" for igual a "d", roda o modelo de
deriva genetica.
  {
    #VERIFICANDO OS PARAMETROS ESPECIFICOS
    if (f < 0 | f > 1 ) #12 Verifica se "f" esta no intervalo adequado para
a funcao.
    {
      stop ("f deve estar no intervalo 0 <= f <= 1.") #13 Senao, interrompe
a funcao e exibe mensagem para o usuario.
    }
    if (N != round (N) | N <= 0) #14 Verifica se "N" eh um numero inteiro e
maior que zero.
    {
      stop ("N deve ser um numero inteiro e > 0.") #15 Senao, interrompe a
```

```
funcao e exibe mensagem para o usuario.
    }
    #CRIANDO OS OBJETOS DA FUNCAO
    simulacao = rep (NA, N) #16 Cria vetor "simulacao" com "N" NAs para
guardar os resultados da simulacao.
    populacao = c ((rep ("A", f * N)), rep ("a", N - (f * N))) #17 Cria
vetor "populacao" com as frequencias alelicas estabelecidas por "f".
    final = rep (NA, pop) #18 Cria vetor "final" com "pop" NAs para guardar
as frequencias alelicas finais de cada populacao.
    #PREPARANDO A AREA DOS GRAFICOS
    x11 () #17 Abre dispositivo de tela.
    par (mfrow = c (1, 2)) #19 Divide o dispositivo de tela em duas colunas.
    plot (x = NULL, y = NULL, #20 Plota area do grafico vazia.
          xlim = c (0, gen), ylim = c (0, 1), #21 Determina valores minimos
e maximos dos eixos X e Y.
          bty = "l", #22 Coloca margens nos lados 1 e 2.
          xlab = "Geracoes", ylab = "f(A)") #23 Coloca os titulos dos eixos
do grafico.
    #SIMULANDO AS TRAJETORIAS DAS POPULACOES
    for (j in 1:pop) #24 Entra em um fluxo "for" com contador "j" de 1 ate
"pop".
    {
      points (0, f, col = j, pch = 16, cex = 0.5) #25 Plota a primeira
frequencia alelica no grafico.
      f.old = f #26 Guarda o valor de "f" em "f.old".
      for (i in 1:gen) #27 Entra em um fluxo "for" com contador "i" de 1 ate
"gen".
      {
        simulacao = sample (populacao, replace = TRUE) #28 Amostra com
reposicao os alelos de "populacao" e guarda em "simulacao".
        f.new = (length (simulacao [simulacao == "A"])) / N #29 Calcula em
"f.new" a nova frequencia alelica.
        points (i, f.new, col = j, pch = 16, cex = 0.5) #30 Plota a nova
frequencia alelica no grafico.
        segments (i - 1, f.old, i, f.new, col = j) #31 Une os pontos das
frequencias alelicas.
        f.old = f.new #32 A nova frequencia alelica passa a ser a antiga
frequencia alelica para o proximo sorteio.
        populacao = simulacao #33 A populacao amostrada ("simulacao") passa
a ser a "populacao" para o proximo sorteio.
      }
      populacao = c ((rep ("A", f * N)), rep ("a", N - (f * N))) #34 Retorna
"populacao" para a frequencia alelica inicial.
      final [j] = f.new #35 Guarda a frequencia alelica final na posicao "j"
de "final".
    }
    #CRIANDO O HISTOGRAMA DAS FREQUENCIAS ALELICAS FINAIS
    hist (final, xlab = "f(A) final", ylab = "Frequencia", main = NULL) #36
Plota histograma com as frequencias alelicas finais.
    #CALCULANDO O VALOR DE P
```

```
    if (p != FALSE) #37 Verifica se o usuario inseriu algum valor para "p".
    {
      p = round (p, digits = 1) #38 Se sim, estabelece o valor de "p" com
uma casa decimal.
      abline (v = p, col = "red") #39 Coloca linha vertical no histograma
com o valor de "p".
      mtext ("p", side = 3, at = p, col = "red") #40 Escreve "p" sobre a
linha criada em #39.
      final = c (round (final, digits = 1)) #41 Arredonda todos os elementos
de "final" para uma casa decimal.
      prob = (length (final [final == p])) / pop #42 Calcula a frequencia
que o valor de "p" aparece em "final".
      cat (paste ("Probabilidade de f(A) =", p, "em", pop, "populacoes com
tamanho", N, "apos", gen, "geracoes:", prob)) #43 Imprime mensagem na tela
com o valor de "p".
    }
  }
  #FUNCAO DE SELECAO NATURAL
  if (mod == "s") #44 Se o argumento "mod" for igual a "s", roda o modelo de
selecao natural.
  {
    #VERIFICANDO OS PARAMETROS ESPECIFICOS
    if (any (f < 0 | f > 1)) #45 Verifica se os elementos de "f" estao no
intervalo adequado para a funcao.
    {
      stop ("f deve ser um vetor com elementos no intervalo 0 <= f <= 1.")
#46 Senao, interrompe a funcao e exibe mensagem para o usuario.
    }
    if (wAA < 0 | wAA > 1 ) #47 Verifica se "wAA" esta no intervalo adequado
para a funcao.
    {
      stop ("wAA deve estar no intervalo 0 <= wAA <= 1.") #48 Senao,
interrompe a funcao e exibe mensagem para o usuario.
    }
    if (wAa < 0 | wAa > 1 ) #49 Verifica se "wAa" esta no intervalo adequado
para a funcao.
    {
      stop ("wAa deve estar no intervalo 0 <= wAa <= 1.") #50 Senao,
interrompe a funcao e exibe mensagem para o usuario.
    }
    if (waa < 0 | waa > 1 ) #51 Verifica se "waa" esta no intervalo adequado
para a funcao.
    {
      stop ("waa deve estar no intervalo 0 <= waa <= 1.") #52 Senao,
interrompe a funcao e exibe mensagem para o usuario.
    }
    #CRIANDO OS OBJETOS DA FUNCAO
    semi.final = rep (NA, length (f)) #53 Cria vetor "semi.final" com o
comprimento de "f" preenchido com NAs para gaurdar as frequencias alelicas
finais.
    #PREPARANDO A AREA DOS GRAFICOS
```

```
    x11() #54 Abre dispositivo de tela.
    par (mfrow = c (1, 2)) #55 Divide o dispositivo de tela em duas colunas.
    plot (x = NULL, y = NULL, #56 Plota area do grafico vazia.
          xlim = c (0, gen), ylim = c (0, 1), #57 Determina valores minimos
e maximos dos eixos X e Y.
          bty = "l", #58 Coloca margens nos lados 1 e 2.
          xlab = "Geracoes", ylab = "f(A)") #59 Coloca os titulos dos eixos
do grafico.
    #SIMULANDO AS TRAJETORIAS DAS POPULACOES
    for (j in 1:length (f)) #60 Entra em um fluxo "for" com contador "j" de
1 ate o comprimento de "f".
    {
      points (0, f[j], col = j, pch = 16, cex = 0.5) #61 Plota a frequencia
alelica inicial no grafico.
      f.old = f[j] #62 Guarda o valor da frequencia alelica inicial em
"f.old".
      for (i in 1:gen) #63 Entra em um fluxo "for" com contador "i" de 1 ate
"gen".
      {
        f.new = (((f.old * f.old) * wAA) + (f.old * (1 - f.old) * wAa)) /
#64 Calcula em "f.new" a nova frequencia alelica.
                (((f.old * f.old) * wAA) + (2 * f.old * (1 - f.old) * wAa) +
(((1 - f.old) * (1 - f.old)) * waa))
        points (i, f.new, col = j, pch = 16, cex = 0.5) #65 Plota a nova
frequencia alelica no grafico.
        segments (i - 1, f.old, i, f.new, col = j) #66 Une os pontos das
frequencias alelicas.
        f.old = f.new #67 A nova frequencia alelica passa a ser a antiga
frequencia alelica para o proximo sorteio.
      }
      semi.final [j] = f.new #68 Guarda a frequencia alelica final na
posicao "j" de "semi.final".
    }
    #CRIANDO O HISTOGRAMA DAS FREQUENCIAS ALELICAS FINAIS
    final = rep (semi.final, each = pop) #69 Como o modelo eh
deterministico, nao ha necessidade de repetir as simulacoes para as mesmas
frequencias alelicas iniciais.
                                          #Assim, no vetor "final" os valores
das simulacoes sao repetidos para as simulacoes com as mesmas frequencias
iniciais.
    hist (final, xlab = "f(A) final", ylab = "Frequecia", main = NULL) #70
Plota histograma com as frequencias alelicas finais.
    #CALCULANDO O VALOR DE P
    if (p != FALSE) #71 Verifica se o usuario inseriu algum valor para "p".
    {
      p = round (p, digits = 1) #72 Se sim, estabelece o valor de "p" com
uma casa decimal.
      abline (v = p, col = "red") #73 Coloca linha vertical no histograma
com o valor de "p".
      mtext ("p", side = 3, at = p, col = "red")  #74 Escreve "p" sobre a
```

```
linha criada em #73.
      final = c (round (final, digits = 1)) #75 Arredonda todos os elementos
de "final" para uma casa decimal.
      prob = round ((length (final [final == p])) / (pop * length (f)),
digits = 2) #76 Calcula a frequencia que o valor de "p" aparece em "final".
      cat (paste ("Probabilidade de f(A) =", p, "após", gen, "geracoes:",
prob)) #77 Imprime mensagem na tela com o valor de "p".
    }
  }
}
```