PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
Departamento de Engenharia Industrial
Rua Marquês de São Vicente, 225
22453-900 – Rio de Janeiro
Brasil

ENG1536 – Inferência Estatística

Laboratório – Guia de Estudo 6

Prosseguimos com o tema do estudo anterior, onde você aprendeu a criar funções novas no R. Agora, vamos estudar as estruturas de programação disponíveis, especificamente **laços de repetição** (*loops*) e **tomadas de decisão**. A sua criação é muito simples em R e, com elas, você estará apto a escrever programas sofisticados, quando for necessário.

Laços de repetição são estruturas que forçam o computador a repetir um conjunto de operações. Elas podem ser repetidas um número fixo de vezes ou podem ser repetidas até que alguma condição estipulada pelo programador seja atingida. O primeiro caso é implementado pela função `for`. Digite o exemplo abaixo:

```
for(i in 1:7) {print(paste("dia",i))}
```

O argumento da função `for` especifica o nome de um objeto local (no exemplo acima, `i`) que é o contador do laço de repetição, seguido da palavra `in` e depois por um vetor de valores que o contador assume sequencialmente. Para cada valor que o contador assume, a expressão agrupada entre chaves é executada uma vez. Observe que o valor do contador pode ser usado pelas expressões agrupadas. O exemplo traz duas funções novas como bônus para você: `print` e `paste`. A função `print` apenas imprime na tela o seu argumento; se este for o nome de um objeto, a função imprime o conteúdo do objeto. Já a função `paste` serve para colar os elementos correspondentes de dois vetores de caracteres. Experimente criar algumas expressões com essas duas funções para entendê-las bem.

No próximo exemplo, veja que os valores que o contador da função `for` assume não precisam ser numéricos. (Qual é o papel do parâmetro `sep` nessa segunda chamada da função `paste`?)

```
diasdetrabalho <- c("segunda","terça","quarta","quinta","sexta")
for(dia in diasdetrabalho) {print(paste(dia,"-feira",sep=""))}
```

Você pode criar laços **aninhados** (*nested loops*). Um laço de repetição está "aninhado" se ele aparece dentre as expressões agrupadas de outro laço. Nesse caso, em cada repetição do laço externo acontecem todas as repetições do laço aninhado. O exemplo a seguir usa dois laços, um aninhado dentro de outro, para criar uma tabuada multiplicativa completa numa matriz 10x10. De brinde, você aprende mais uma função em R: `matrix`.

```
tabuada <- matrix(ncol=10,nrow=10)
for(m in 1:10) {for(n in 1:10) {tabuada[m,n] <- m*n}}; tabuada
```

Como dissemos no início deste guia, podemos criar também laços sem um número predeterminado de repetições, que só terminam quando uma condição é atingida. Um laço desse segundo tipo pode ser criado pela função `while`. Veja neste exemplo:

```
z <- 0; while(abs(z) <= 1.645) {z <- rnorm(1); print(z)}
```

O argumento da função `while` especifica uma expressão lógica que é o teste para interrupção do laço. As expressões agrupadas depois da função `while` continuam sendo repetidas enquanto a expressão lógica for verdadeira. Só quando ela se torna falsa, o laço é finalizado. No exemplo acima, sorteios normais são impressos no console até que seja sorteado um número maior em módulo do que 1.645 (qual é a probabilidade de isso acontecer?). Repita o exemplo algumas vezes e veja que o número de execuções do laço é variável.

Para concluir este estudo, aprendemos a forma básica de tomada de decisão em R. A função, neste caso, chama-se `if`. A sua estrutura é parecida com a da função `while`, mas ela não determina um laço de repetição. O argumento da função `if` é uma expressão lógica e as expressões agrupadas depois da função só são executadas (uma vez) se a expressão for verdadeira. Neste exemplo sorteamos e imprimimos um número normal padrão. Caso ele seja menor do que um desvio padrão, imprime-se também o comentário em texto:

```
z <- rnorm(1); print(z); if(abs(z) < 1) {print("Nada extraordinário")}
```

É possível enriquecer a tomada de decisão especificando uma expressão agrupada para ser executada caso o teste lógico resulte em falsidade. Usamos a palavra `else` para esse tipo de estrutura. Exemplo:

```
z <- rnorm(1); print(z); if(abs(z) < 1) {print("Nada extraordinário")} else
{print("Atenção!)}
```

Repita esse último exemplo algumas vezes para constatar como o R adapta corretamente o comentário textual ao número sorteado.

Em conclusão a este estudo, é importante frisar uma coisa. Na utilização cotidiana do R, dificilmente as estruturas que você acaba de aprender serão imprescindíveis para alguma análise de dados. A natureza vetorial das expressões em R permite realizar muita coisa sem termos de recorrer a laços de repetição ou tomadas de decisão. Por exemplo, abaixo sorteamos dez números normais e, através de um laço e de uma decisão, mostramos no console os números positivos dentre os sorteados:

```
z <- rnorm(10); for(i in 1:10) {if(z[i]>0) {print(z[i])}}
```

Exatamente a mesma coisa pode ser feita assim:

```
z <- rnorm(10); z[z>0]
```

Acostume-se a pensar vetorialmente. Sempre que você achar que precisa de um laço de repetição, pare e pense se a linguagem R não permite fazer o que você quer sem o laço. Num estudo futuro, aprenderemos sobre uma família de funções chamada `apply` que também ajuda a contornar a necessidade de laços de repetição em diversas aplicações.

Lista de exercícios 6

(1) Suponha que um vetor chamado `vendas` contenha os 60 volumes financeiros mensais das vendas de uma loja nos últimos cinco anos, em R$. Escreva uma expressão que mostre na tela a média móvel trimestral das vendas: o primeiro número mostrado é a média dos meses 1 a 3, o segundo número é a média dos meses de 2 a 4, etc.

(2) DESAFIO: Estude (ou relembre) o método de Newton-Raphson para estimar as raízes de um polinômio. Depois, escreva uma expressão em R para estimar uma raiz da função $f(x)=x^2-x-2$ a partir de dois valores `x0` e `x1` entre os quais a raiz esteja. Estabeleça como condição de convergência que a diferença entre os valores finais de `x0` e `x1` seja menor do que 0,01. A estimativa final da raiz estará em `x1`. Dica: você precisará de um laço de repetição, mas o número de iterações até a convergência é desconhecido de antemão.

(3) Existe uma maneira de interromper um laço de repetição antes que a sua condição de término seja atingida. Descubra qual palavra reservada em R faz isso.

(4) Quando você usa a função `while` para criar um laço condicional, a expressão agrupada pode não ser executada nem uma vez sequer, caso a expressão lógica de teste seja falsa. (Para entender isso, faça igual a 2, ao invés de 0, o valor inicial de `z` no primeiro exemplo do guia sobre essa função.) Descubra se existe em R outra função programação que permita criar um laço condicional no qual sempre haverá pelo menos uma repetição da expressão agrupada.

(5) Neste guia você aprendeu a função `print`, que permite a um programa comunicar-se com o usuário escrevendo mensagens na tela. Existe uma função irmã dessa que permite ao R receber interativamente uma mensagem do usuário. Descubra o nome da função e estude-a. Depois, escreva uma expressão em R que pergunte ao usuário a idade dele e, dependendo da idade digitada, imprime-se o texto "`menor de idade`" ou "`adulto`".

* * *