Integração entre R e C

Elias Teixeira Krainski

26 de janeiro de 2009

Sumário

Resumo

Neste documento mostramos como usar bibliotecas escritas em C e compiladas no R. Iniciamos mostrando exemplos de operações simples, como soma e produto, para mostrar como gerar uma biblioteca dinâmica e usá-la no R. Também mostramos exemplos com uso da biblioteca de geradores de números aleatórios, cálculo de probabilidades, etc., escritas em C e usada no R, e uso de funções da biblioteca LAPACK escrita em Fortran e usadas no R.

1 Introdução

Os procedimentos iterativos (for, while, repeat) são mais eficientes em C que em R, devido ao Rser uma linguagem interpretada. O objetivo básico é fazer os cálculos iterativos em C, gerar uma biblioteca dinamica e usa-la a partir do R. Eventualmente o procedimento que vamos programar em C requer uso de funções com cálculos mais avançados, tais como inversão de matrizes, geração de números aleatórios, optimização, etc. Neste ponto entra uma vantagem de programar integrando o Re o C. Essa vantagem é a facilidade de utilizar as bibliotecas R escritas em C, que fazem esses cálculos, dentro do nossas funções escritas em C.

A integração entre Re C pode ocorrer em dois sentidos, ou seja, podemos chamar funções em C no Re chamar funções em Rdentro do C. Neste material iniciamos com a

chamada de funções escritas em C dentro do R. Esse procedimento é feito via função de interface `.C()` do R. No primeiro argumento dessa função deve ser passado a *string* como o nome da função em C. A função em C, neste caso, deve ser uma função que apenas faz um processamento e não retorna nada explicitamente. Além disso, deve-se incluir a biblioteca R.h. Por exemplo, criamos o arquivo "helloc.c" com o seguinte conteúdo:

```
#include<R.h>
void helloc() {
  Rprintf("Hello!\n");
}
```

Note que a função "helloc" apenas imprime uma palavra e nada é retornado após sua execução.

Esse arquivo deve ser compilado com o comando

```
R CMD SHLIB helloc.c
```

gerando o arquivo "helloc.so" no Linux ou "helloc.dll" no Windows. Esse arquivo deve ser carregado no R para que as funções em C definidas nele possam ser utilizadas no R. Carregamos com

```
> dyn.load("helloc.so")
```

E a partir desse momento podemos utilizar a função "helloc" escrita em C dentro do R, usando a função de interface `.C`:

```
> .C("helloc")

Hello!
list()
```

Quase sempre nosso objetivo é fazer uma análise de dados. Portanto, a função em C deveria receber os dados e fazer uma análise e retornar o resultado. Com a função de interface `.C` nós podemos passar vetores para as funções em C que os recebem como ponteiros. Dessa forma, podemos passar um vetor de dados e um objeto para receber o resultado.

Suponha que queremos somar os elementos de um vetor. Então podemos fazer:

```
#include<R.h>
void somac(int *n, double *x, int *res) {
  int i;
  *res = 0.0;
  for (i=0; i<*n; i++)
    *res += x[i];
}
```

Note que temos que passar também o tamanho do vetor. Após compilar e carregar a biblioteca em R, podemos utilizar a função "somac"com `.C()`. Porém agora, devemos também passar os argumentos que devem ser da mesma classe!

```
> dyn.load("somac.so")
> .C("somac", as.integer(5), as.double(1:5), double(1))

[[1]]
[1] 5

[[2]]
[1] 1 2 3 4 5

[[3]]
[1] 15
```

Note que `.C()` retorna uma lista com os argumentos passados para a função. Note também que o terceiro argumento retorna o resultado!

2 Números aleatórios, densidades, etc.

As funções em C podem ser mais complexas e envolver uso de geradores de números aleatórios, optimização, etc. ou inversão de matrizes, cálculo de determinante, etc. No primeiro grupo de funções, o R

to do ...

3 Análise de regressão

4 Densidade da distribuição normal multivariada

5 Exemplo de MCMC

6 Chamando uma função R do C