

# Foreign Language Interfaces

Valerie Obenchain

Fred Hutchinson Cancer Research Center

17-18 February, 2011

# Overview

- ▶ Motivation
- ▶ Interface functions
- ▶ Compile and load dynamic libraries
- ▶ Using `.C()`
- ▶ Register native routines

# Motivation

- ▶ Areas where the R implementation is suboptimal :
  - ▶ sliding window algorithms
  - ▶ calculations that are difficult to vectorize
- ▶ Implement third party algorithms or libraries (e.g., GSL, BOOST, BGL, SAMtools, affxparser)

# Interface Functions

The following functions provide a standard interface to compiled code that has been linked into *R*:

- ▶ .C
- ▶ .Call
- ▶ .Fortran

# Advantages of .Call() vs .C()

- ▶ Less copying
- ▶ Memory allocation in C
- ▶ Pass and receive R objects instead of primitive types
- ▶ Access to the attributes of the vectors (i.e., names)
- ▶ Ability to handle missing values easily

## C code

- ▶ Compiled code should not return anything except through the arguments
- ▶ C functions should be of type void
- ▶ Include Rdefines.h and R\_ext/Rdynload.h

```
/* composite_linkage_disequilibrium.c */  
void  
composite_linkage_disequilibrium(  
    unsigned char *snp, /* matrix indiv x snp */  
    int *n_ind,         /* # individuals */  
    int *n_snp,         /* # snps */  
    int *width,         /* adjacent snp window */  
    double *delta)      /* result */  
{  
    ...  
}
```

# Compile and load dynamic libraries : *R* Session

- ▶ The shared library can be created with R CMD SHLIB `composite_linkage_disequilibrium.c`.
- ▶ From within an *R* session the compiled library can be loaded with `dyn.load`. The functions in the compiled code are now available for use in the *R* session.  
> `dyn.load("composite_linkage_disequilibrium.so")`

# Compile and loading dynamic libraries : *R* Package

- ▶ Load with `useDynLib(mypkg)` in the `NAMESPACE`
- ▶ Other instructions can be put in `.onLoad` and `.onUnload` functions in a `zzz.R` file.



## Using .C()

- ▶ The first argument to .C is a character string of the C function name. The remainder of the arguments are *R* objects to be passed to the C function.
- ▶ All arguments should be coerced to the correct *R* storage mode

```
snps <- matrix(sample((0:2), replace=TRUE),  
               nrow=10, ncol=4)  
nsnp <- ncol(snps)  
nsub <- nrow(snps)  
width <- 3  
delta <- rep.int(0, (nsnp-width)*width)  
out <- .C("composite_linkage_disequilibrium",  
          snp = as.integer(snp), n_ind = as.integer(nind),  
          n_snp = as.integer(nsnp), width = as.integer(width),  
          delta = as.double(delta))
```

# Register Native Routines

Motivation :

- ▶ Platform-independent mechanism for finding routines in shared objects
- ▶ Information about a native routine made available within *R*

Steps :

- ▶ Create an initialization file called `R_init_mypkg.c`
- ▶ Create an array describing the function with `R_CmethodDef`
- ▶ Register the function with `R_registerRoutines`

# Register Native Routines

- ▶ Create an array describing the C routine using `R_CMethodDef` using the following *R* types and corresponding type identifiers :

<code>`numeric'</code>	<code>`REALSXP'</code>
<code>`integer'</code>	<code>`INTSXP'</code>
<code>`logical'</code>	<code>`LGLSXP'</code>
<code>`character'</code>	<code>`STRSXP'</code>
<code>`list'</code>	<code>`VECSXP'</code>

# Register Native Routines

- ▶ Given the C function declared as

```
void composite_linkage_disequilibrium(  
    unsigned char *snp,  
    int *n_ind,  
    int *n_snp,  
    int *width,  
    double *delta)
```

- ▶ We would create the R\_CMethodDef array

```
R_CMethodDef cMethods[] = {  
    {"composite_linkage_disequilibrium",  
     (DL_FUNC) &composite_linkage_disequilibrium, 5,  
     { INTSXP, INTSXP, INTSXP, INTSXP, REALSXP }  
    },  
    {NULL, NULL, 0}  
};
```

# Initialization Function

- The initialization file contains the R\_CMethodDef array and the call to R\_registerRoutines in the R\_init\_STudentGWAS function.

```
void R_init_StudentGWAS(DllInfo *info)
{
    /* Create the R_CMethodDef array */
    R_CMethodDef cMethods[] = {
        {"composite_linkage_disequilibrium",
         (DL_FUNC) &composite_linkage_disequilibrium, 5,
         { INTSXP, INTSXP, INTSXP, INTSXP, REALSXP }
        },
        {NULL, NULL, 0}
    };

    /* Register the routine */
    R_registerRoutines(info, cMethods, NULL, NULL, NULL)
}
```

# Resources

- ▶ Writing *R* Extensions Manual <http://www.r-project.org/>