Translation of R to C++

The R package ast2ast

Konrad Krämer

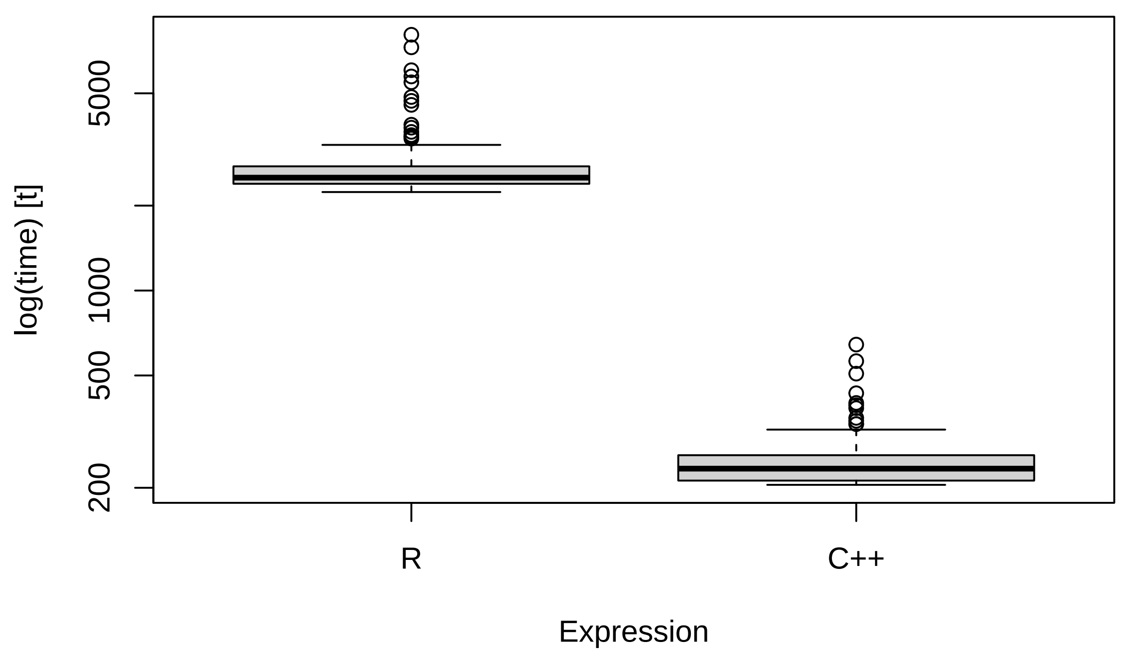
2022-06-21

Motivation and aim

Problem: Using R functions which:

- are called very often
- are called by C or C++:
    - expensive copying of memory from R to C/C++ and *vice versa*

```
Loading required package: rmumps
```



Motivation and aim

How to solve the problem?

Motivation and aim

Solution: Translating an R function using *ast2ast*

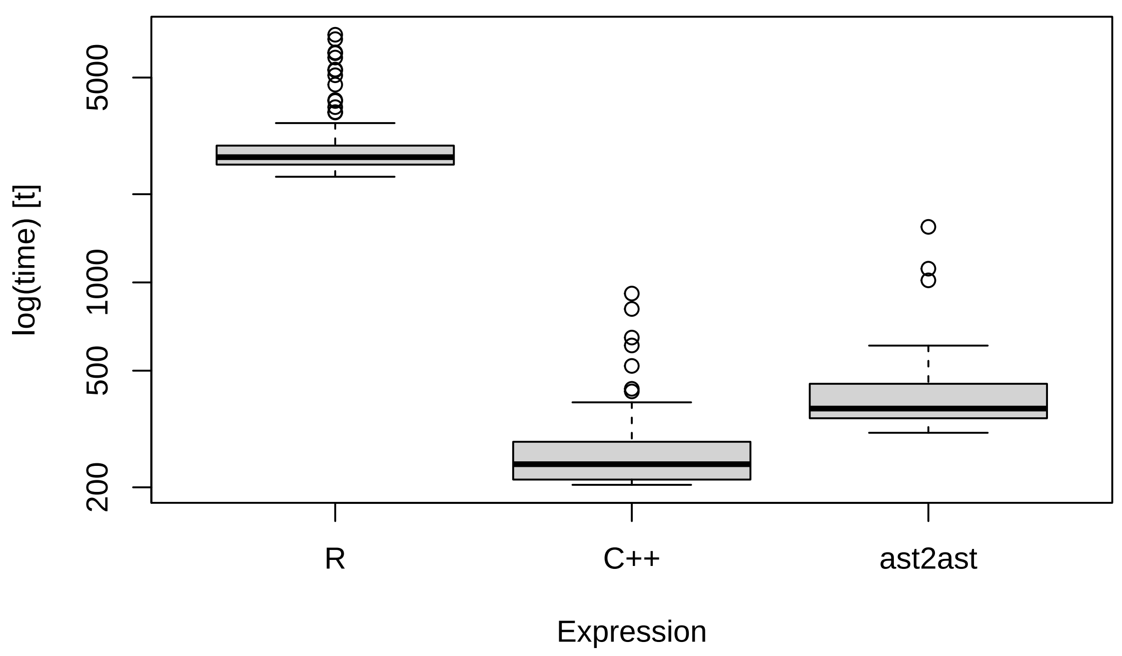
- Optimization
- ODE-functions

```
# ast2ast version
ti <- seq(0, 5, length.out=101)
y0 <- 0

library(ast2ast)
ode <- function(y, ydot) {
  nu <- 2
  a <- 1
  ydot[1] <- -nu*(y[1] - a)
}
pointer_to_ode <- translate(ode,
                            reference = TRUE)
res_exp3 <- solve_ode(pointer_to_ode,
                      ti, y0)
attributes(res_exp3) <- NULL

stopifnot(identical(res_exp,
                res_exp2,
                res_exp3))
out <- microbenchmark(
  r2cvodes(y0, ti,
          frhs, param = p),
  r2cvodes(y0, ti,
          ptr_exp, param = pv),
  solve_ode(pointer_to_ode,
          ti, y0))

boxplot(out, names=c("R", "C++", "ast2ast"))
```



1. Implemention of *ETR*

- *Expression template library R (ETR)*

```
// [[Rcpp::depends(RcppArmadillo)]]
// [[Rcpp::depends(ast2ast)]]
#include "etr.hpp"
// [[Rcpp::plugins("cpp17")]]

// [[Rcpp::export]]
void example() {
  sexp pi = 3.14;
  sexp vec = colon(1, 4);
  sexp mat = matrix(5, 2, 2);
  print(pi*vec);
  print();
  print(vec + vec + mat);
}
```

```r
example()
```

```
3.14
6.28
9.42
12.56

7    11
9    13
```

1. Comparison of *ETR* and R

```cpp
// [[Rcpp::depends(RcppArmadillo)]]
// [[Rcpp::depends(ast2ast)]]
#include "etr.hpp"
// [[Rcpp::plugins("cpp17")]]

// [[Rcpp::export]]
void example() {
  sexp pi = 3.14;
  sexp vec = colon(1, 4);
  sexp mat = matrix(5, 2, 2);
  print(pi*vec);
  print();
  print(vec + vec + mat);
}
```

```r
example <- function() {
  pi = 3.14
  vec = 1:4
  mat = matrix(5, 2, 2)
  print(pi*vec)
  cat("\n")
  print(vec + vec + mat)
}
example()
```

```
[1]  3.14  6.28  9.42 12.56
```

```
     [,1] [,2]
[1,]    7   11
[2,]    9   13
```

- C++ and R code are very similar
- R code is translated to C++

2. Translation of R code

```r
1:4 + a[1]
```

2. Translation of R code

```r
library(lobstr)
ast(1:4 + a[1])
```

```
  `+`
   `:`
    1
    4
   `[`
    a
    1
```

```r
library(lobstr)
ast(colon(1, 4) + subset(a, 1))
```

```
  `+`
   colon
    1
    4
   subset
    a
    1
```

Example

```r
library(ast2ast)
fibonacci <- function() {
  v <- vector(6)
  v[1] <- 1
  v[2] <- 1

  for(i in 3:length(v)) {
    v[i] <- v[i - 1] + v[i - 2]
    print(v[i])
  }

}
sourceCpp_out <-
  translate(fibonacci,
            R_fct = TRUE)
f()
```

```
2
3
5
8
```

```cpp
void f() {
sexp v;
v = vector(i2d(6));
subassign(v, 1) = i2d(1);
subassign(v, 2) = i2d(1);

for(auto&i:  colon(i2d(3), length(v))) {
subassign(v, i) =
  subset(v, i - i2d(1))
  + subset(v, i - i2d(2));
etr::print(subset(v, i));
}
```

Interface with Rcpp

```cpp
// Rcpp to sexp
NumericVector a{1, 2};
sexp a_; // sexp a_ = a; Error!
a_ = a;
print(a_);

// sexp to Rcpp
sexp b_ = coca(3, 4);
```

```
NumericVector b = b_;
Rcpp::Rcout << b << std::endl;
```

Pointer interface

Copy memory

```
double* ptr;
ptr = new double[size];
int cob = 0;
sexp a(size, ptr, 0);
delete [] ptr;
a = vector(3.14, 5);
```

Take Ownership

```
double* ptr;
ptr = new double[size];
sexp b(size, ptr, 1);
b = vector(5, 3);
```

Pointer interface

borrow ownership

```
double* ptr;
ptr = new double[size];
sexp c(size, ptr, 2);
//c = vector(5, size + 1); //error calls resize
c = vector(4, size);
delete[] ptr;
```

Conclusion

Thank you very much for your attention

Get in contact:

- Github: https://github.com/Konrad1991
- Twitter: https://twitter.com/kraemer_konrad