Introduction to the paglm package

Benjamin Christoffersen

2018-11-16

The motivation for the `parglm` package is a parallel version of the `glm` function. It solves the iteratively re-weighted least Squares using a QR decomposition with column pivoting with `DGEQP3` function from LAPACK. The computation is done in parallel as in the `bam` function `mgcv`. The cost is an additional $O(Mp^2 + p^3)$ where p is the number of coefficients and M is the number chunks to be computed in parallel. More one the latter shortly. The advantage is that you do not need an optimized BLAS or LAPACK which support multithreading to perform the estimation.

Example of computation time

Below, we perform estimate a logistic regression with 100000 observations and 50 covariates. We vary the number of cores being used with the `nthreads` argument to `parglm.control`.

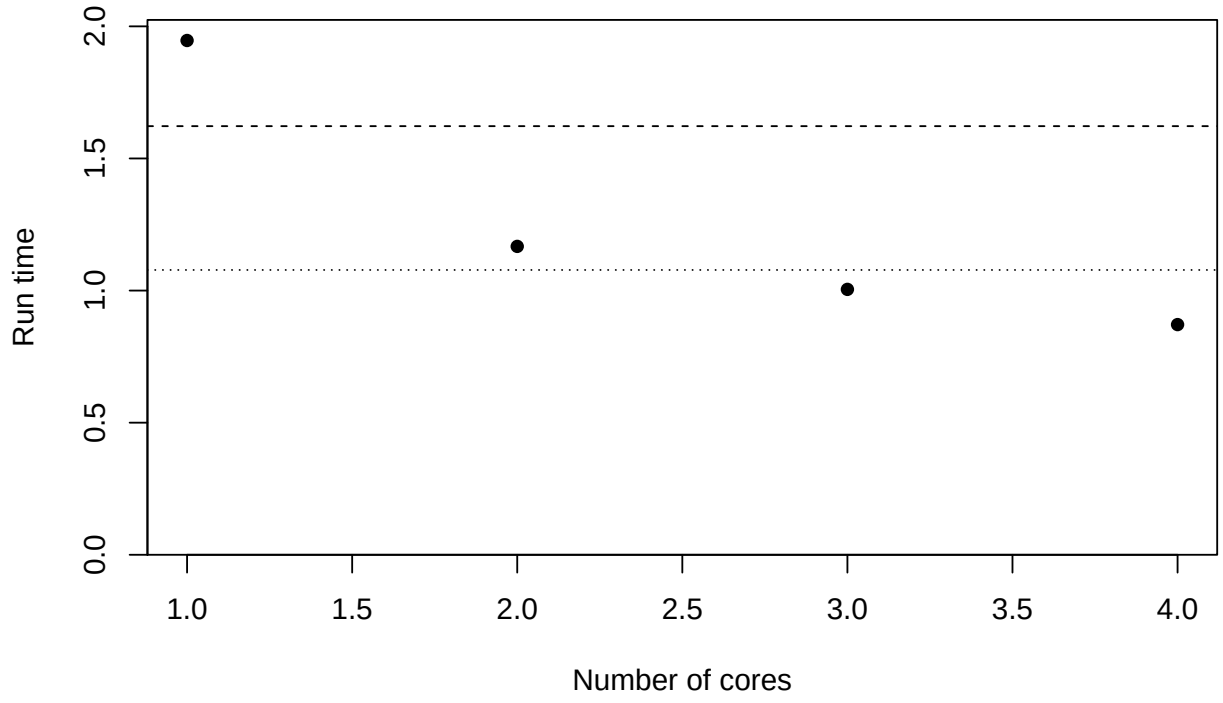
```
#####
# simulate
n # number of observations
#> [1] 100000
p # number of covariates
#> [1] 50
X <- matrix(rnorm(n * p, 1/p, 1/sqrt(p)), n, ncol = p)
set.seed(68024947)
df <- data.frame(y = 1/(1 + exp(-(rowSums(X) - 1))) > runif(n), X)

#####
# compute and measure time. Setup call to make
library(microbenchmark)
library(speedglm)
#> Warning: package 'speedglm' was built under R version 3.5.1
#> Loading required package: Matrix
#> Loading required package: MASS
library(parglm)
cl <- list(
  quote(microbenchmark),
  glm      = quote(glm     (y ~ ., binomial(), df)),
  speedglm = quote(speedglm(y ~ ., family = binomial(), data = df)),
  times = 5L)
cl <- c(
  cl, lapply(1:n_threads, function(i) bquote(parglm(
      y ~ ., binomial(), df, control = parglm.control(nthreads = .(i))))))
names(cl)[5:(5L + n_threads - 1L)] <- paste0("parglm.", 1:n_threads)
cl <- as.call(cl)
cl # the call we make
#> microbenchmark(glm = glm(y ~ ., binomial(), df), speedglm = speedglm(y ~
#>     ., family = binomial(), data = df), times = 5L, parglm.1 = parglm(y ~
#>     ., binomial(), df, control = parglm.control(nthreads = 1L)),
#>     parglm.2 = parglm(y ~ ., binomial(), df, control = parglm.control(nthreads = 2L)),
#>     parglm.3 = parglm(y ~ ., binomial(), df, control = parglm.control(nthreads = 3L)),
#>     parglm.4 = parglm(y ~ ., binomial(), df, control = parglm.control(nthreads = 4L)))
```

```
out <- eval(cl)
out # result from `microbenchmark`
#> Unit: milliseconds
#>       expr       min        lq      mean    median        uq       max neval
#>        glm 1563.0610 1576.1956 1605.8570 1622.3456 1631.4565 1636.226     5
#>   speedglm 1012.3342 1012.4784 1074.3872 1077.9995 1133.6775 1135.446     5
#>   parglm.1 1879.7253 1899.2323 1940.7508 1946.5177 1979.5457 1998.733     5
#>   parglm.2 1148.7156 1160.1325 1191.3742 1167.1859 1178.0284 1302.809     5
#>   parglm.3  948.3287  976.0063 1038.0595 1004.4800 1100.3825 1161.100     5
#>   parglm.4  855.4105  857.7500  911.5601  871.2761  885.2696 1088.094     5
```

The plot below shows median run time versus the number of cores. The dashed line is the median run time of `glm` and the dotted line is median run time of `speedglm`.

```
par(mar = c(4.5, 4.5, .5, .5))
o <- aggregate(time ~ expr, out, median)[, 2] / 10^9
ylim <- range(o, 0); ylim[2] <- ylim[2] + .04 * diff(ylim)
plot(1:n_threads, o[-(1:2)], xlab = "Number of cores", yaxs = "i",
     ylim = ylim, ylab = "Run time", pch = 16)
abline(h = o[1], lty = 2)
abline(h = o[2], lty = 3)
```



It is worth mentioning that `speedglm` computes the cross product of the weighted design matrix. This is advantages in terms of computation cost but may lead to unstable solutions.

`parglm` does not at the moment handle close to singular problems as "neatly" as `glm` where `glm` forces some elements to be excluded.

Session info

```
sessionInfo()
#> R version 3.5.0 (2018-04-23)
#> Platform: x86_64-w64-mingw32/x64 (64-bit)
#> Running under: Windows >= 8 x64 (build 9200)
#>
#> Matrix products: default
#>
#> locale:
#> [1] LC_COLLATE=English_United States.1252
#> [2] LC_CTYPE=English_United States.1252
#> [3] LC_MONETARY=English_United States.1252
#> [4] LC_NUMERIC=C
#> [5] LC_TIME=English_United States.1252
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#> [1] microbenchmark_1.4-4 speedglm_0.3-2       MASS_7.3-49
#> [4] Matrix_1.2-14        parglm_0.1.0
#>
#> loaded via a namespace (and not attached):
#>  [1] Rcpp_1.0.0       codetools_0.2-15 lattice_0.20-35  digest_0.6.15
#>  [5] rprojroot_1.3-2  grid_3.5.0       backports_1.1.2  magrittr_1.5
#>  [9] evaluate_0.10.1  stringi_1.1.7    rmarkdown_1.9    tools_3.5.0
#> [13] stringr_1.3.0    yaml_2.1.18      compiler_3.5.0   htmltools_0.3.6
#> [17] knitr_1.20
```