Complex Model Simulation

Package Loading

Before loading the package, we should allocate enough memory for Java. Here we allocate 10GB of memory for Java.

```
set.seed(123)
# Allocate 10GB of memory for Java. Must be called before library(iBART)
options(java.parameters = "-Xmx10g")
library(iBART)
```

Complex Model

In this vignette, we will run iBART on the complex model described in Section 3.4 of the paper, i.e. the data-generating model is

$$y = 15\{\exp(x_1) - \exp(x_2)\}^2 + 20\sin(\pi x_3 x_4) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}_n(0, \sigma^2 I).$$

The primary features are $X = (x_1, ..., x_p)$, where $x_1, ..., x_p \overset{\text{iid}}{\sim} \text{Unif}_n(-1, 1)$. We will use the following setting: $n = 250$, $p = 10$, and $\sigma = 0.5$. The goal in OIS is to identify the 2 true descriptors: $f_1(X) = \{\exp(x_1) - \exp(x_2)\}^2$ and $f_2(X) = \sin(\pi x_3 x_4)$ using only (y, X) as input.

```
#### Simulation Parameters ####
n <- 250 # Change n to 100 here to reproduce result in Supplementary Materials A.2.3
p <- 10  # Number of primary features


#### Generate Data ####
X <- matrix(runif(n * p, min = -1, max = 1), nrow = n, ncol = p)
colnames(X) <- paste("x.", seq(from = 1, to = p, by = 1), sep = "")
y <- 15 * (exp(X[, 1]) - exp(X[, 2]))^2 + 20 * sin(pi * X[, 3] * X[, 4]) + rnorm(n, mean = 0, sd = 0.5)


#### iBART ####
iBART_results <- iBART(X = X, y = y,
                       head = colnames(X),
                       opt = c("unary", "binary", "unary"), # unary operator first
                       sin_cos = TRUE,
                       apply_pos_opt_on_neg_x = FALSE,
                       Lzero = TRUE,
                       K = 4,
                       aic = TRUE,
                       standardize = FALSE,
                       seed = 123)
#> Start iBART descriptor generation and selection...
#> Iteration 1
#> iBART descriptor selection...
#> avg.........null.....................................................
#> Constructing descriptors using unary operators...
```

```
#> Iteration 2
#> iBART descriptor selection...
#> avg..........null...............................................
#> Constructing descriptors using binary operators...
#> Iteration 3
#> iBART descriptor selection...
#> avg..........null...............................................
#> Constructing descriptors using unary operators...
#> BART iteration done!
#> LASSO descriptor selection...
#> L-zero regression...
#> Total time: 266.82212305069 secs
```

`iBART()` returns many interesting outputs. For example, `iBART_results$descriptor_names` returns the descriptors selected by iBART, and `iBART_results$iBART_model` returns the selected model—a `cv.glmnet` object. We can use the iBART model the same way we use a `glmnet` model. For instance, we can print out the coefficients using `coef()`.

```
# iBART selected descriptors
iBART_results$descriptor_names
#> [1] "(exp(x.1)-exp(x.2))^2" "sin(pi*(x.3*x.4))"

# iBART model
class(iBART_results$iBART_model)
#> [1] "cv.glmnet"

coef(iBART_results$iBART_model, s = "lambda.min")
#> 146 x 1 sparse Matrix of class "dgCMatrix"
#>                                  1
#> (Intercept)              0.1928037
#> x.1                      .
#> x.2                      .
#> x.3                      .
#> x.4                      .
#> exp(x.1)                 .
#> exp(x.2)                 .
#> exp(x.3)                 .
#> exp(x.4)                 .
#> (x.2+exp(x.2))           .
#> (x.1-exp(x.2))           .
#> (x.2-exp(x.1))           .
#> (exp(x.1)-exp(x.2))      .
#> (x.1*x.2)                .
#> (x.2*exp(x.1))           .
#> (x.3*x.4)                .
#> (exp(x.1)*exp(x.2))      .
#> (x.2/exp(x.1))           .
#> (exp(x.2)/exp(x.1))      .
#> |x.1-x.2|                .
#> |x.3-x.4|                .
#> |exp(x.1)-exp(x.2)|      .
#> exp(x.1)^0.5             .
#> exp(x.2)^0.5             .
#> exp(x.3)^0.5             .
```

```
#> exp(x.4)^0.5                  .
#> (exp(x.1)*exp(x.2))^0.5       .
#> (exp(x.2)/exp(x.1))^0.5       .
#> |x.1-x.2|^0.5                 .
#> |x.3-x.4|^0.5                 .
#> |exp(x.1)-exp(x.2)|^0.5       .
#> x.1^2                         .
#> x.2^2                         .
#> x.3^2                         .
#> x.4^2                         .
#> exp(x.1)^2                    .
#> exp(x.2)^2                    .
#> exp(x.3)^2                    .
#> exp(x.4)^2                    .
#> (x.2+exp(x.2))^2              .
#> (x.1-exp(x.2))^2              .
#> (x.2-exp(x.1))^2              .
#> (exp(x.1)-exp(x.2))^2       14.7643022
#> (x.1*x.2)^2                   .
#> (x.2*exp(x.1))^2              .
#> (x.3*x.4)^2                   .
#> (exp(x.1)*exp(x.2))^2         .
#> (x.2/exp(x.1))^2              .
#> (exp(x.2)/exp(x.1))^2         .
#> |x.1-x.2|^2                   .
#> |x.3-x.4|^2                   .
#> log((exp(x.1)*exp(x.2)))      .
#> log((exp(x.2)/exp(x.1)))      .
#> log(|x.1-x.2|)                .
#> log(|x.3-x.4|)                .
#> log(|exp(x.1)-exp(x.2)|)      .
#> exp(exp(x.1))                 .
#> exp(exp(x.2))                 .
#> exp(exp(x.3))                 .
#> exp(exp(x.4))                 .
#> exp((x.2+exp(x.2)))           .
#> exp((x.1-exp(x.2)))           .
#> exp((x.2-exp(x.1)))           .
#> exp((exp(x.1)-exp(x.2)))      .
#> exp((x.1*x.2))                .
#> exp((x.2*exp(x.1)))           .
#> exp((x.3*x.4))                .
#> exp((exp(x.1)*exp(x.2)))      .
#> exp((x.2/exp(x.1)))           .
#> exp((exp(x.2)/exp(x.1)))      .
#> exp(|x.1-x.2|)                .
#> exp(|x.3-x.4|)                .
#> exp(|exp(x.1)-exp(x.2)|)      .
#> sin(pi*x.1)                   .
#> sin(pi*x.2)                   .
#> sin(pi*x.3)                   .
#> sin(pi*x.4)                   .
#> sin(pi*exp(x.1))              .
```

```
#> sin(pi*exp(x.2))                .
#> sin(pi*exp(x.3))                .
#> sin(pi*exp(x.4))                .
#> sin(pi*(x.2+exp(x.2)))          .
#> sin(pi*(x.1-exp(x.2)))          .
#> sin(pi*(x.2-exp(x.1)))          .
#> sin(pi*(exp(x.1)-exp(x.2)))     .
#> sin(pi*(x.1*x.2))               .
#> sin(pi*(x.2*exp(x.1)))          .
#> sin(pi*(x.3*x.4))              19.5876303
#> sin(pi*(exp(x.1)*exp(x.2)))     .
#> sin(pi*(x.2/exp(x.1)))          .
#> sin(pi*(exp(x.2)/exp(x.1)))     .
#> sin(pi*|x.1-x.2|)               .
#> sin(pi*|x.3-x.4|)               .
#> sin(pi*|exp(x.1)-exp(x.2)|)     .
#> cos(pi*x.1)                     .
#> cos(pi*x.2)                     .
#> cos(pi*x.3)                     .
#> cos(pi*x.4)                     .
#> cos(pi*exp(x.1))                .
#> cos(pi*exp(x.2))                .
#> cos(pi*exp(x.3))                .
#> cos(pi*exp(x.4))                .
#> cos(pi*(x.2+exp(x.2)))          .
#> cos(pi*(x.1-exp(x.2)))          .
#> cos(pi*(x.2-exp(x.1)))          .
#> cos(pi*(exp(x.1)-exp(x.2)))     .
#> cos(pi*(x.1*x.2))               .
#> cos(pi*(x.2*exp(x.1)))          .
#> cos(pi*(x.3*x.4))               .
#> cos(pi*(exp(x.1)*exp(x.2)))     .
#> cos(pi*(x.2/exp(x.1)))          .
#> cos(pi*(exp(x.2)/exp(x.1)))     .
#> cos(pi*|x.1-x.2|)               .
#> cos(pi*|x.3-x.4|)               .
#> x.1^(-1)                        .
#> x.2^(-1)                        .
#> x.3^(-1)                        .
#> x.4^(-1)                        .
#> exp(x.1)^(-1)                   .
#> exp(x.2)^(-1)                   .
#> exp(x.3)^(-1)                   .
#> exp(x.4)^(-1)                   .
#> (x.2+exp(x.2))^(-1)             .
#> (x.1-exp(x.2))^(-1)             .
#> (x.2-exp(x.1))^(-1)             .
#> (exp(x.1)-exp(x.2))^(-1)        .
#> (x.1*x.2)^(-1)                  .
#> (x.2*exp(x.1))^(-1)             .
#> (x.3*x.4)^(-1)                  .
#> (exp(x.1)*exp(x.2))^(-1)        .
#> (x.2/exp(x.1))^(-1)             .
```

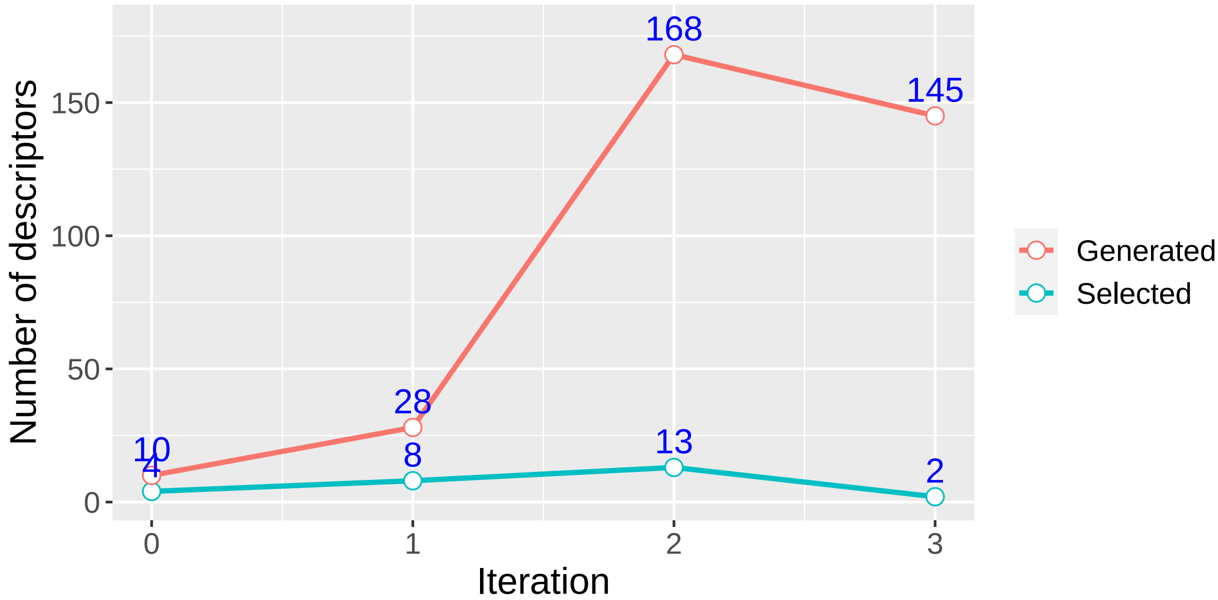
```
#> (exp(x.2)/exp(x.1))^(-1)        .
#> |x.1-x.2|^(-1)                  .
#> |x.3-x.4|^(-1)                  .
#> |exp(x.1)-exp(x.2)|^(-1)        .
#> abs(x.1)                        .
#> abs(x.2)                        .
#> abs(x.3)                        .
#> abs(x.4)                        .
#> abs((x.2+exp(x.2)))             .
#> abs((x.1-exp(x.2)))             .
#> abs((x.2-exp(x.1)))             .
#> abs((x.1*x.2))                  .
#> abs((x.2*exp(x.1)))             .
#> abs((x.3*x.4))                  .
#> abs((x.2/exp(x.1)))             .
```

Here iBART generated 145 descriptors in the last iteration, and it correctly identified the true descriptors $f_1(X)$ and $f_2(X)$ without selecting any false positive. This is very reassuring especially when some of these descriptors are highly correlated with $f_1(X)$ or $f_2(X)$. For instance, $\tilde{f}_1(X) = |\exp(x_1) - \exp(x_2)|$ in the descriptor space is highly correlated with $f_1(X)$.

```
f1_true <- (exp(X[,1]) - exp(X[,2]))^2
f1_cor <- abs(exp(X[,1]) - exp(X[,2]))
cor(f1_true, f1_cor)
#> [1] 0.9517217
```

`iBART()` also returns other useful and interesting outputs, such as `iBART_results$iBART_gen_size` and `iBART_results$iBART_sel_size`. They store the dimension of the newly generated / selected descriptor space for each iteration. Let's plot them and see how **iBART** use nonparametric variable selection for dimension reduction. In each iteration, we keep the dimension of intermediate descriptor space under $\mathcal{O}(p^2)$, leading to a progressive dimension reduction.

```
library(ggplot2)
df_dim <- data.frame(dim = c(iBART_results$iBART_sel_size, iBART_results$iBART_gen_size),
                     iter = rep(0:3, 2),
                     type = rep(c("Selected", "Generated"), each = 4))
ggplot(df_dim, aes(x = iter, y = dim, colour = type, group = type)) +
  theme(text = element_text(size = 15), legend.title = element_blank()) +
  geom_line(size = 1) +
  geom_point(size = 3, shape = 21, fill = "white") +
  geom_text(data = df_dim, aes(label = dim, y = dim + 10, group = type),
            position = position_dodge(0), size = 5, colour = "blue") +
  labs(x = "Iteration", y = "Number of descriptors") +
  scale_x_continuous(breaks = c(0, 1, 2, 3))
#> Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
#> i Please use `linewidth` instead.
#> This warning is displayed once every 8 hours.
#> Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
#> generated.
```



R Session Info

```
sessionInfo()
#> R version 4.0.5 (2021-03-31)
#> Platform: x86_64-w64-mingw32/x64 (64-bit)
#> Running under: Windows 10 x64 (build 22621)
#>
#> 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] ggplot2_3.4.4 iBART_0.0.3.3
#>
#> loaded via a namespace (and not attached):
#>  [1] compiler_4.0.5      pillar_1.9.0        iterators_1.0.13
#>  [4] tools_4.0.5         digest_0.6.33       missForest_1.4
#>  [7] evaluate_0.22       lifecycle_1.0.3     tibble_3.2.1
#> [10] gtable_0.3.4        lattice_0.20-44     pkgconfig_2.0.3
#> [13] rlang_1.1.1         Matrix_1.3-4        foreach_1.5.1
#> [16] cli_3.6.1           rstudioapi_0.15.0   yaml_2.3.5
#> [19] parallel_4.0.5      xfun_0.40           fastmap_1.1.1
#> [22] rJava_1.0-4         withr_2.5.1         dplyr_1.1.3
#> [25] knitr_1.44          generics_0.1.3      bartMachineJARs_1.1
#> [28] vctrs_0.6.4         tidyselect_1.2.0    glmnet_4.1-1
#> [31] grid_4.0.5          glue_1.6.2          bartMachine_1.2.6
#> [34] R6_2.5.1            fansi_1.0.3         survival_3.2-11
```

```
#> [37] rmarkdown_2.25      farver_2.1.0        magrittr_2.0.3
#> [40] scales_1.2.1        codetools_0.2-18    htmltools_0.5.6.1
#> [43] itertools_0.1-3     splines_4.0.5       randomForest_4.6-14
#> [46] shape_1.4.6         colorspace_2.0-3    labeling_0.4.3
#> [49] utf8_1.2.2          munsell_0.5.0       crayon_1.5.2
```