How to seep-up VSURF ?

Robin Genuer[a], Jean-Michel Poggi[b], Christine Tuleau-Malot[c]

[a]Université de Bordeaux, ISPED, Inserm U1219, Inria
[b]Université Paris Descartes et Laboratoire de mathématiques d'Orsay
[c]Laboratoire Jean-Alexandre Dieudonné, Université de Nice

July 12
UseR 2019, Toulouse

Random Forests

- introduced by Breiman (2001)
- ensemble methods family Dietterich (1999, 2000)
- very efficient algorithm of statistical learning, for both classification and regression problems

Random Forests

- introduced by Breiman (2001)
- ensemble methods family Dietterich (1999, 2000)
- very efficient algorithm of statistical learning, for both classification and regression problems

$\mathcal{L}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ i.i.d. r.v. with the same distribution as (X, Y).

$X = (X^1, ..., X^p) \in \mathbb{R}^p$ (input variables)
$Y \in \mathcal{Y}$ (output variable)

- $\mathcal{Y} = \mathbb{R}$: regression
- $\mathcal{Y} = \{1, \ldots, L\}$: classification

Goal : build a predictor $\widehat{h} : \mathbb{R}^p \to \mathcal{Y}$ and select the most relevant variables

Definition : Random Forests (Breiman 2001)

$\left\{\widehat{h}(.,\Theta_\ell), 1 \leq \ell \leq q\right\}$ randomized tree-predictor collection,
$(\Theta_\ell)_{1\leq\ell\leq q}$ i.i.d. r.v. independent with $\mathcal{L}_n$

RF predictor $\widehat{h}_{\text{RF}}$ obtained by agreggating the collection of trees

Agreggation :

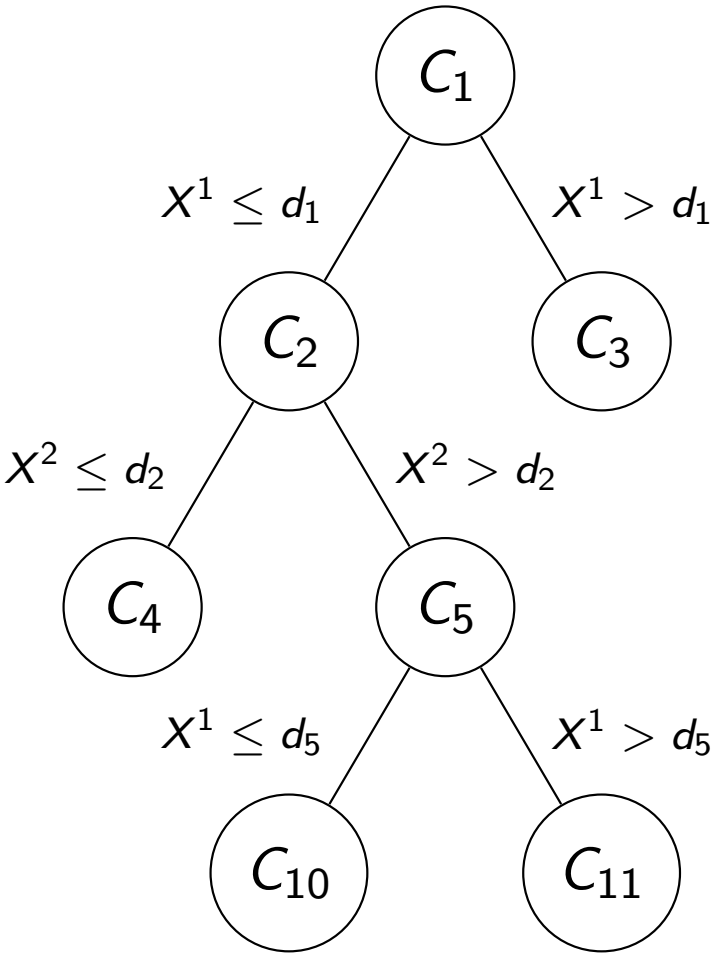
- $\widehat{h}_{\text{RF}}(x) = \dfrac{1}{q}\sum_{\ell=1}^{q}\widehat{h}(x,\Theta_\ell)$ regression
- $\widehat{h}_{\text{RF}}(x) = \underset{1\leq c\leq L}{\text{argmax}}\sum_{\ell=1}^{q}\mathbb{1}_{\widehat{h}(x,\Theta_\ell)=c}$ classification

Tree: piece-wise constant predictor from a recursive binary partitioning of $\mathbb{R}^p$

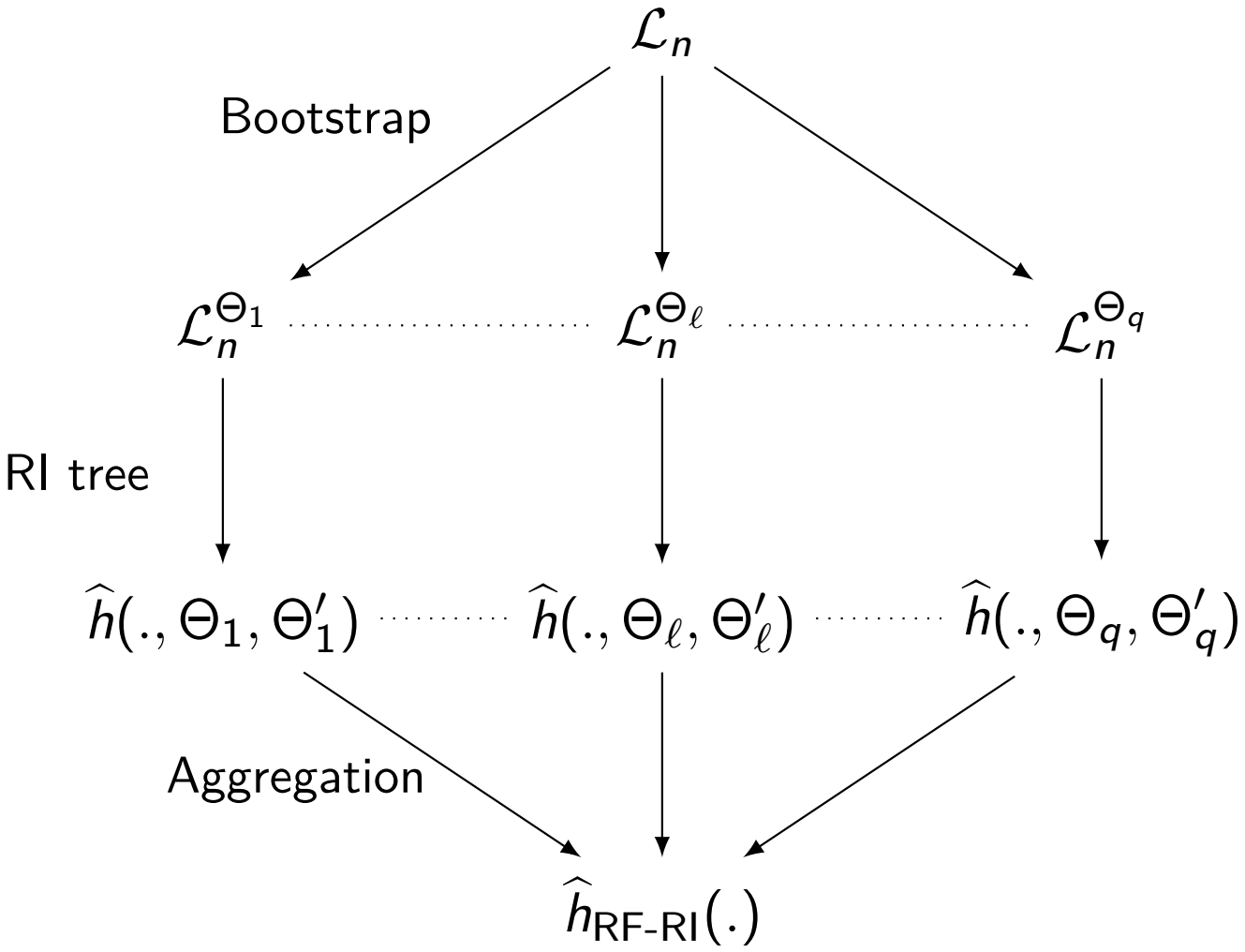
Splits are parallel to axes

Typically, at each node, the "best" split is seek (heterogeneity criterion based on the Y's)

Example: CART (maximal tree building + pruning) Breiman et.al. (1984)



Random Forests-Random Inputs (Breiman 2001)



Randomness + aggregation $\Rightarrow$ increase of efficiency

RF-RI

Definition : RI-tree

An RI-tree is a variant of CART which selects at random, at each node, `mtry` variables, and splits only using the selected variables
An RI-tree is also not pruned

`mtry` is THE method parameter and is same for all nodes of all trees in the forest:

- from $\texttt{mtry} = 1$ (splitting variable choice completely random)
- to $\texttt{mtry} = p$ (Bagging Breiman 1996)

Main R packages

Reference R package `randomForest` Liaw, Wiener (since 2002)
$\rightarrow$ based on initial code of Breiman, Cutler (2000)

Other implementations in R:

- `ranger` Wright, Ziegler (since 2015): "high-dimension"
- `Rborist` Seligman (since 2015): "big data"

Two main parameters considered here:

- `ntree/num.trees/nTree` : number of trees in the forest (default = 500)
- `mtry/mtry/predFixed` : number of variables randomly selected at each node (default = $\sqrt{p}$ in classification)

Variable importance

Breiman (2001), Strobl *et al.* (2007, 2008), Ishwaran (2007), Archer *et al.* (2008), Louppe *et al.* (2013)

OOB = Out Of Bag (≈ "Out Of Bootstrap")

Definition: Variable importance (VI)

Let $j \in \{1, \ldots, p\}$. For each OOB sample, randomly permute the j-th variable values of the data

$\text{VI}(X^j)$ = mean increase of a tree error after permutation

The more the error increases, the more important is the variable.

Variable Selection

Genuer, Poggi, Tuleau (2010)

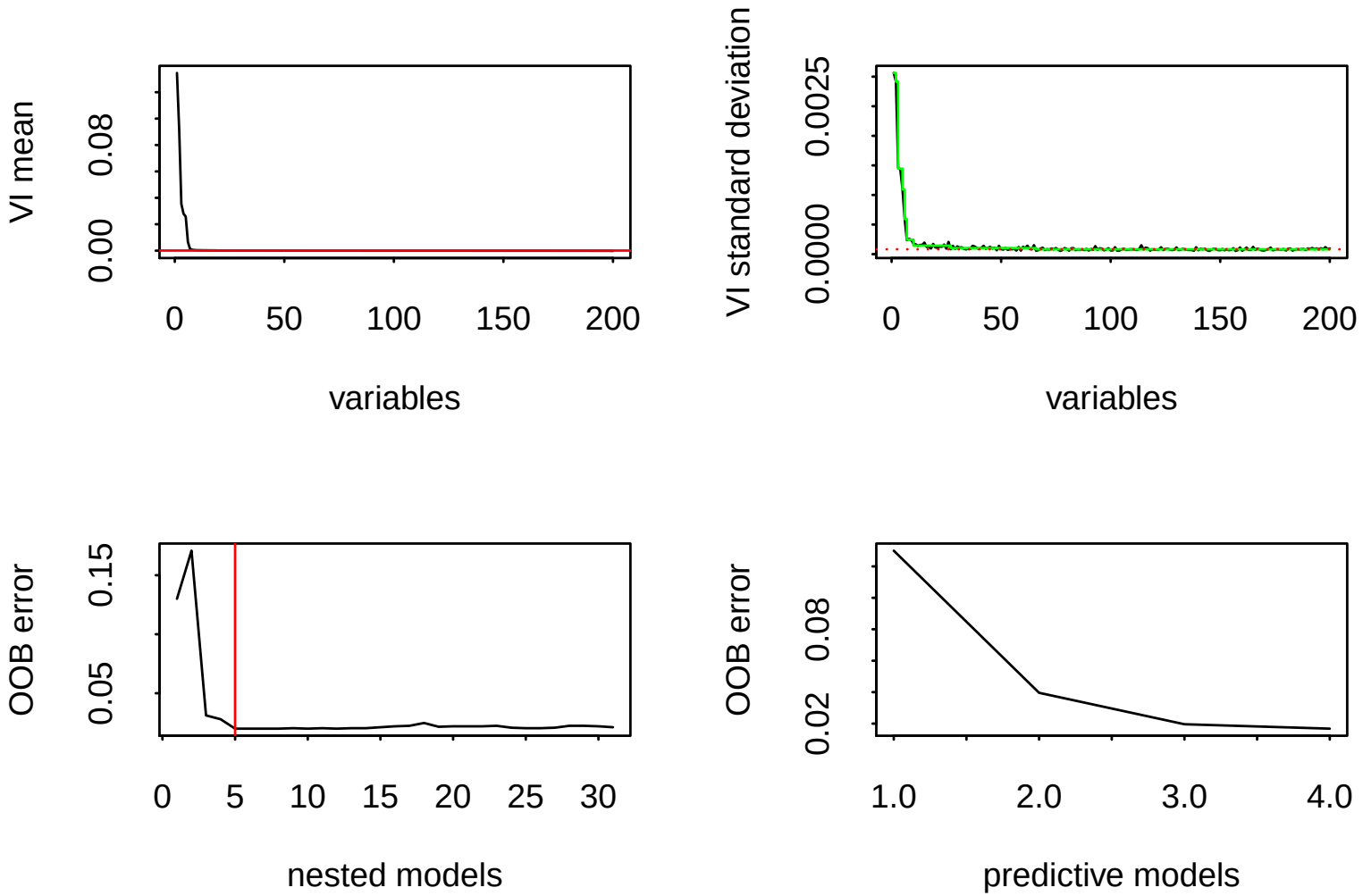
We distinguish two different objectives:

1. to select all important variables, even with high redundancy, for interpretation purpose
2. to find a sufficient parsimonious set of important variables for prediction

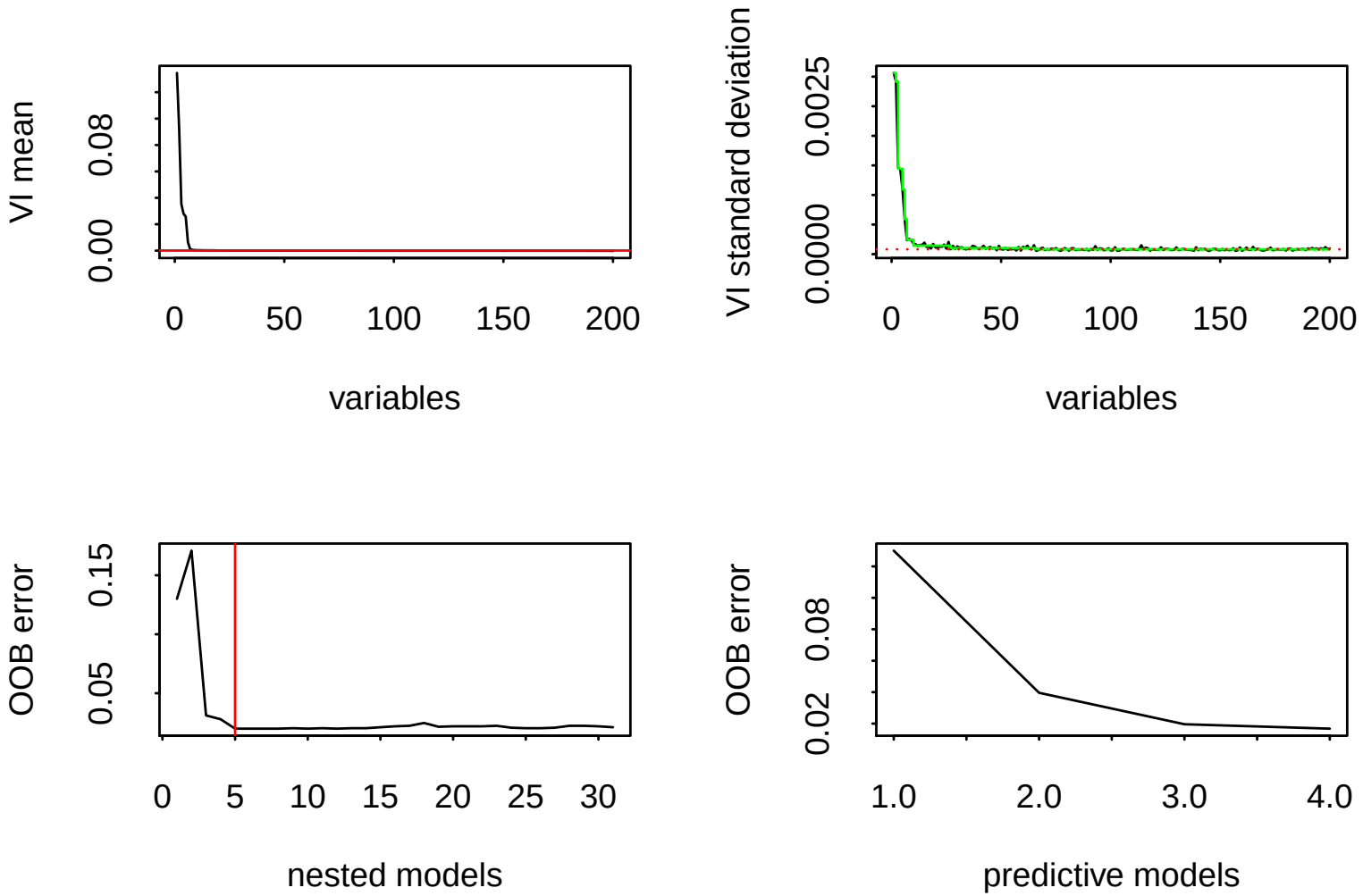
Our aim is to build an automatic procedure, which fulfills these two objectives

One earlier work must be cited: Díaz-Uriarte, Alvarez de Andrés (2006).

A simulated dataset ($L = 2, n = 100, p = 200$, 6 true var.)



A simulated dataset ($L = 2, n = 100, p = 200, 6$ true var.)



50 RF with VI + 31 × 25 RF + 5 × 25 RF

VSURF() with randomForest

```
library(VSURF)
data(toys)
toysDef <- VSURF(toys$x, toys$y)
plot(toysDef)
summary(toysDef)
```

```
##
##  VSURF computation time: 1.1 mins
##
##  VSURF selected:
##  31 variables at thresholding step (in 45 secs)
##  5 variables at interpretation step (in 21.4 secs)
##  4 variables at prediction step (in 1.5 secs)
```

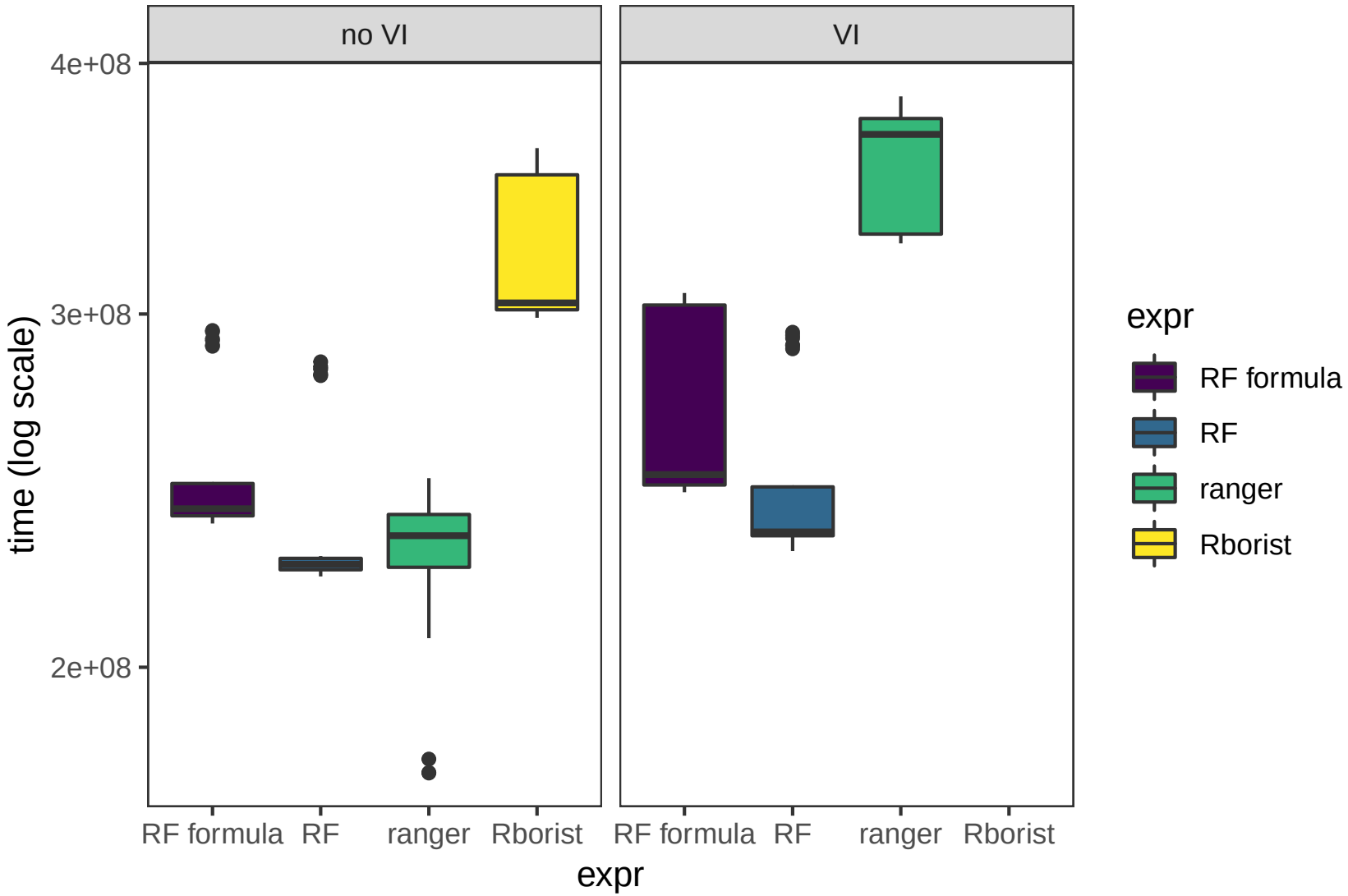
VSURF() with ranger

```
toysRanger <- VSURF(toys$x, toys$y, RFimplem = "ranger")
summary(toysRanger)
```

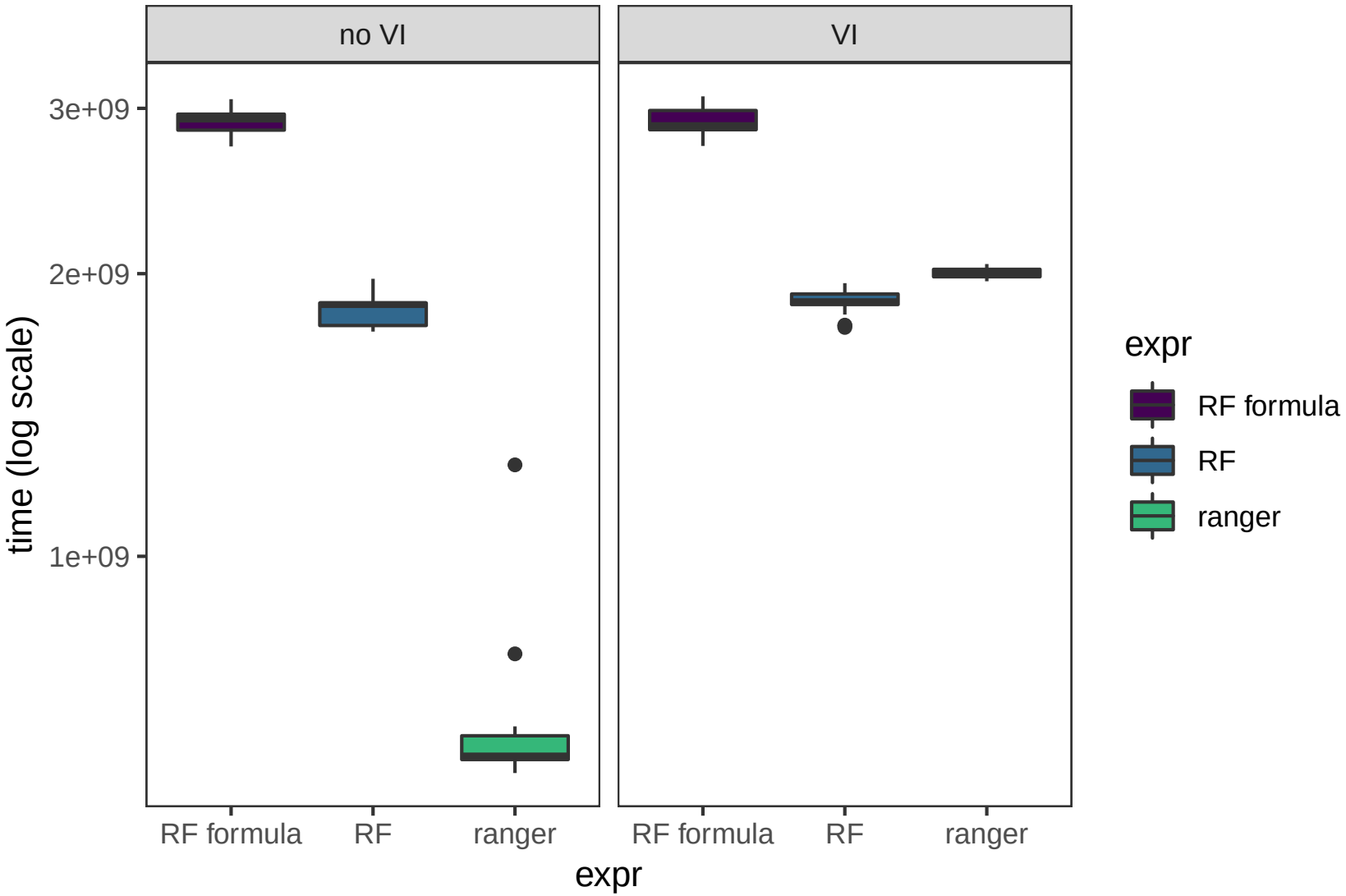
```
##
##  VSURF computation time: 1 mins
##
##  VSURF selected:
##  25 variables at thresholding step (in 44.5 secs)
##  5 variables at interpretation step (in 15.1 secs)
##  4 variables at prediction step (in 1.6 secs)
```

(Rborist implementation in progress...)

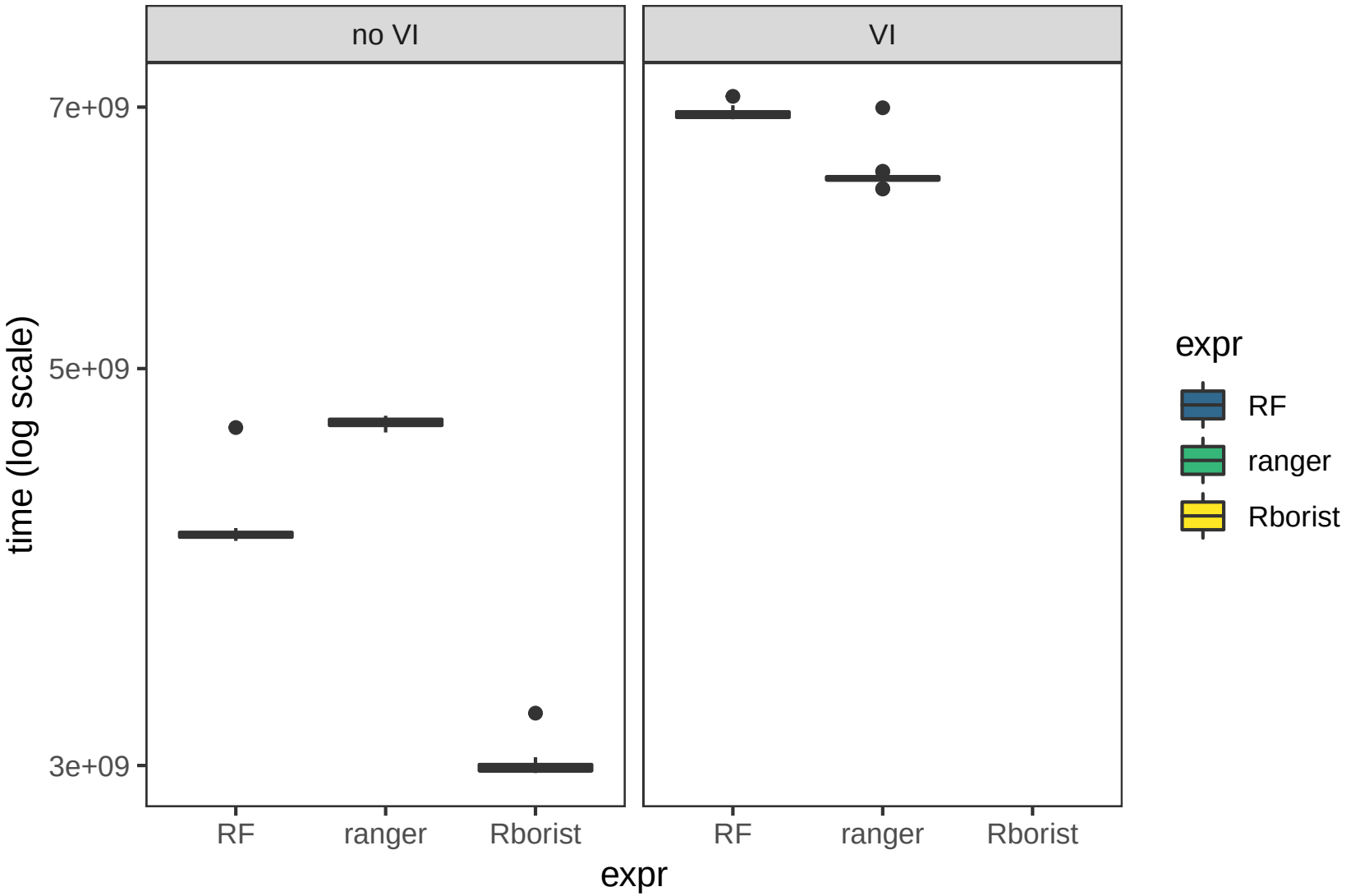
microbenchmark (25 runs) only on RF runs



High-dimensional ($L = 2, n = 100$, **p=2000**, **66** true var.)



Big data ($L = 2$, **n=10000**, **p=10**, 6 true var.)



VSURF on vac18 data ($L = 4, n = 42, p = 1000$)

randomForest

```
##
##  VSURF computation time: 1.5 mins
##
##  VSURF selected:
##  97 variables at thresholding step (in 45.7 secs)
##  23 variables at interpretation step (in 35.9 secs)
##  13 variables at prediction step (in 6.2 secs)
##
##  VSURF ran in parallel on a FORK cluster and used 3 cores
```

ranger

```
##
##  VSURF computation time: 2.8 mins
##
##  VSURF selected:
##  97 variables at thresholding step (in 51.7 secs)
##  23 variables at interpretation step (in 1.7 mins)
##  13 variables at prediction step (in 16.1 secs)
##
##  VSURF ran in parallel on a FORK cluster and used 3 cores
```

VSURF on spam data ($L = 2, n = 2300, p = 57$)

randomForest

```
##
##  VSURF computation time: 11.9 mins
##
##  VSURF selected:
##  97 variables at thresholding step (in 1.2 mins)
##  23 variables at interpretation step (in 2.4 mins)
##  13 variables at prediction step (in 8.3 mins)
##
##  VSURF ran in parallel on a FORK cluster and used 32 cores
```

ranger

```
##
##  VSURF computation time: 25.5 mins
##
##  VSURF selected:
##  97 variables at thresholding step (in 40.9 secs)
##  23 variables at interpretation step (in 2.9 mins)
##  13 variables at prediction step (in 22 mins)
##
##  VSURF ran in parallel on a FORK cluster and used 32 cores
```

Concluding Remarks

- Speed-up `VSURF` by changing RF implementation: not so simple !
  - `ranger` improve RF run time, but not always with VI
  - `Rborist` improve in the big data case, but do not compute VI natively
- `ranger` and `Rborist` allow parallel computing (`randomForest` not)
- Development version include `RFimplem` parameter:

```
remotes::installgithub("robingenuer/VSURF")
```

Short bibliography


Breiman, L., Friedman J., Olshen R., Stone C. *Classification And Regression Trees*. Chapman & Hall (1984)


Breiman, L. *Random Forests*. Machine Learning (2001)


Díaz-Uriarte R., Alvarez de Andrés S. *Gene Selection and classification of microarray data using random forest*. BMC Bioinformatics (2006)


Genuer R., Poggi J.-M. and Tuleau-Malot C. *VSURF: An R Package for Variable Selection Using Random Forests*. Rjournal (2015)


Seligman. *Rborist: Extensible, Parallelizable Implementation of the Random Forest Algorithm*. R package version 0.1-17 (2019)


Wright and Ziegler. *ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R*. JSS (2017)