Using the *tmle.npvi* R package

Antoine Chambaz   Pierre Neuvial

Package version 0.8.1
Date 2014-02-08

Contents

The caution symbol ☡ marks important details.

1 Citing *tmle.npvi*

If you use the *tmle.npvi* package, please cite [Chambaz et al., 2012].

2 The non-parametric variable importance parameter

Consider the following statistical problem. We observe the data structure $O = (W, X, Y)$ on an experimental unit of interest, where $W \in \mathcal{W}$ stands for a vector of baseline covariates, $X \in \mathbb{R}$ and $Y \in \mathbb{R}$ respectively quantify an exposure and a response, and we wish to investigate the relationship between X on Y, accounting for W. Taking W into account is desirable because we know (or cannot rule out the possibility) that it contains confounding factors, *i.e.*, common factors upon which the exposure X and the response Y may simultaneously depend. Furthermore, the exposure features a reference level x_0 with positive mass (there is a positive probability that $X = x_0$) and a *continuum* of other levels.

This motivates the definition of the non-parametric variable importance (NPVI) parameter introduced in [Chambaz et al., 2012]: for all distributions P of O compatible with the above description of O, for f a user-supplied function such that $f(0) = 0$,

$$\Psi_f(P) = \frac{E_P\{f(X - x_0)[E_P(Y|X,W)) - E_P(Y|X = x_0, W)]\}}{E_P\{f(X - x_0)^2\}}.$$

In contrast, the parameter Φ_f characterized by

$$\Phi_f(P) = \frac{E_P\{f(X - x_0)Y\}}{E_P\{f(X - x_0)^2\}}$$

neglects the information conveyed by W.

The *tmle.npvi* R package implements the inference of $\Psi_f(P)$ (and $\Phi_f(P)$) based on independent draws from P based on the targeted minimum loss estimation (TMLE) principle, as described and studied in [Chambaz et al., 2012].

3 Using the *tmle.npvi* R package on simulated data

3.1 Set up

We first set the verbosity parameter and random seed.

```
> library("tmle.npvi")
> library("R.utils")
> log <- Arguments$getVerbose(-8, timestamp=TRUE)
> set.seed(12345)
```

3.2 Generating a simulated data set

The package includes a function, `getSample`, to generate independent copies of $O = (W, X, Y) \in [0, 1] \times \mathbb{R} \times \mathbb{R}$ from the distribution P^s characterized in Section 6.4 of [Chambaz et al., 2012]. The distribution P^s is inspired by real data from The Cancer Genome Atlas (TCGA) project [The Cancer Genome Atlas (TGCA) research Network, 2008], a collaborative initiative to better understand several types of cancers using existing large-scale whole-genome technologies. Given the `EGFR` gene, known to be altered in glioblastoma multiforme (GBM) cancers, the random variables W, X and Y can be interpreted as follows:

- W: a measure of DNA methylation of `EGFR`, the proportion of "methylated" signal at a CpG locus in the promoter region of `EGFR`,
- X: a measure of DNA copy number of `EGFR`, a locally smoothed total copy number relative to a set of reference samples,
- Y: a measure of the expression of `EGFR`, a "unified" gene expression level across three microarray platforms,

all evaluated in GBM cancer cells of a patient. The simulation strategy implements three constraints:

- there are generally up to three copy number classes: normal regions, and regions of copy number gains and losses;
- in normal regions, expression is negatively correlated with methylation;
- in regions of copy number alteration, copy number and expression are positively correlated.

We set parameters for the simulation.

```
> O <- cbind(W=c(0.05218652, 0.01113460),
+            X=c(2.722713, 9.362432),
+            Y=c(-0.4569579, 1.2470822))
> O <- rbind(NA, O)
> lambda0 <- function(W) {-W}
> p <- c(0, 1/2, 1/2)
> omega <- c(0, 3, 3)
> S <- matrix(c(10, 1, 1, 0.5), 2 ,2)
> n <- 200
```

We simulate a data set of 200 independent and identically distributed observations.

```
> sim <- getSample(n, O, lambda0, p=p, omega=omega, sigma2=1, Sigma3=S)
> obs <- sim$obs
> head(obs)
```

```
                W         X           Y
[1,] 0.0215683362 13.048392  2.52935398
[2,] 0.0003507203 10.524890  2.92867989
[3,] 0.0384478781  6.870260  0.68096600
[4,] 0.0002115650  7.726289  1.67007558
[5,] 0.0775521257  2.722713 -0.01222786
[6,] 0.0109059868  2.722713 -0.45807013
```

At this stage, the baseline covariate W takes its values in $[0, 1]$. The *tmle.npvi* `R` package can deal with multi-dimensional W, so we may add other baseline covariates for the sake of the presentation. Note that this alters the definition of P^s. However, the value of the NPVI parameter is preserved.

```
> V <- matrix(runif(3*nrow(obs)), ncol=3)
> colnames(V) <- paste("V", 1:3, sep="")
> obs <- cbind(V, obs)
> head(obs)
```

```
             V1         V2         V3            W         X           Y
[1,] 0.08129589 0.30313148 0.36805931 0.0215683362 13.048392  2.52935398
[2,] 0.59804261 0.13819697 0.32411013 0.0003507203 10.524890  2.92867989
[3,] 0.82233342 0.46616611 0.82844535 0.0384478781  6.870260  0.68096600
[4,] 0.25047667 0.07210956 0.08028252 0.0002115650  7.726289  1.67007558
[5,] 0.52639709 0.66855746 0.25451450 0.0775521257  2.722713 -0.01222786
[6,] 0.05044493 0.33037193 0.38912254 0.0109059868  2.722713 -0.45807013
```

Baseline covariates are identified in the matrix of observations as those numbers stored in the columns which are not labelled `"X"` nor `"Y"`. The `"X"` and `"Y"` columns respectively correspond to the exposure and response.

At this stage, the reference value for X is `O[2, "X"]`.

3.3 True value of the NPVI parameter

The function `getSample` also computes an approximation to the true value of the NPVI parameter $\Psi_f(P^s)$, as well as an approximation to the true variance of the efficient influence curve of Ψ_f at P^s. The approximated variance can be used together with the user-supplied sample size `nrow(sim$obs)` to assert how accurate is the approximation. Used together with the number of observations in `obs`, we obtain an interval (approximately) centered at $\Psi_f(P^s)$ whose length is (approximately) the smallest possible length of a confidence interval based on `nrow(obs)` observations.

```
> sim <- getSample(1e4, O, lambda0, p=p, omega=omega,
+                  sigma2=1, Sigma3=S, verbose=log)
> truePsi <- sim$psi
> confInt0 <- truePsi + c(-1, 1)*qnorm(.975)*sqrt(sim$varIC/nrow(sim$obs))
> confInt <- truePsi + c(-1, 1)*qnorm(.975)*sqrt(sim$varIC/nrow(obs))
> msg <- "\nCase f=identity:\n"
> msg <- c(msg, "\ttrue psi is: ", paste(signif(truePsi, 3)), "\n")
> msg <- c(msg, "\t95%-confidence interval for the approximation is: ",
+          paste(signif(confInt0, 3)), "\n")
> msg <- c(msg, "\toptimal 95%-confidence interval is: ",
+          paste(signif(confInt, 3)), "\n")
> cat(msg)
```

```
Case f=identity:
         true psi is:  0.232
         95%-confidence interval for the approximation is:  0.228 0.237
         optimal 95%-confidence interval is:  0.199 0.266
```

By default, f is taken equal to the identity. We could choose any function f such that $f(0) = 0$, say, for example, $f = \arctan$.

```
> sim2 <- getSample(1e4, O, lambda0, p=p, omega=omega,
+                   sigma2=1, Sigma3=S, f=atan, verbose=log)
> truePsi2 <- sim2$psi
> confInt02 <- truePsi2 + c(-1, 1)*qnorm(.975)*sqrt(sim2$varIC/nrow(sim2$obs))
> confInt2 <- truePsi2 + c(-1, 1)*qnorm(.975)*sqrt(sim2$varIC/nrow(obs))
> msg <- "\nCase f=atan:\n"
> msg <- c(msg, "\ttrue psi is: ", paste(signif(truePsi2, 3)), "\n")
> msg <- c(msg, "\t95%-confidence interval for the approximation is: ",
+          paste(signif(confInt02, 3)), "\n")
> msg <- c(msg, "\toptimal 95%-confidence interval is: ",
+          paste(signif(confInt2, 3)), "\n")
> cat(msg)
```

```
Case f=atan:
         true psi is:  1.3
         95%-confidence interval for the approximation is:  1.27 1.33
         optimal 95%-confidence interval is:  1.1 1.5
```

3.4 TMLE procedure

The function `tmle.npvi` implements the inference of $\Psi_f(P)$ (and of $\Phi_f(P)$) based on independent draws from P based on the TMLE principle.

The function `tmle.npvi` assumes that the reference value $x_0 = 0$. So it is necessary here to shift the values of X.

```
> X0 <- O[2,2]
> obsC <- obs
> obsC[, "X"] <- obsC[, "X"] - X0
> obs <- obsC
> head(obs)
```

```
             V1         V2         V3            W         X           Y
[1,] 0.08129589 0.30313148 0.36805931 0.0215683362 10.325679  2.52935398
[2,] 0.59804261 0.13819697 0.32411013 0.0003507203  7.802177  2.92867989
[3,] 0.82233342 0.46616611 0.82844535 0.0384478781  4.147547  0.68096600
[4,] 0.25047667 0.07210956 0.08028252 0.0002115650  5.003576  1.67007558
[5,] 0.52639709 0.66855746 0.25451450 0.0775521257  0.000000 -0.01222786
[6,] 0.05044493 0.33037193 0.38912254 0.0109059868  0.000000 -0.45807013
```

We now run the TMLE procedure, with f = identity (default value) and relying on parametric models to estimate some relevant infinite-dimensional features of P^s. Alternatively, we could have chosen to rely on the Super Learning methodology [van der Laan et al., 2007, Polley and van der Laan, 2011] (by setting `flavor="superLearning"`), in which case we could have parallelized the computations using several nodes (by tuning the `nodes` argument). All options are given in the documentation of the `tmle.npvi` function.

```
> npvi <- tmle.npvi(obs, f=identity, flavor="learning")
```

```
[1] 1
[1] 2
```

```
> npvi
```

```
NPVI object:

Sample size: 200

Estimator of psi:               0.217
Estimated standard error:       0.278
```

```
Convergence criteria:
- scaled empirical mean of estimating function                < 0.01
- TV distance between P_n^k and P_n^{k+1}                 < 0.01
- change between successive values of "psi"                 < 0.1

Convergence reached after 2 iteration(s) because:
        TV distance between P_n^k and P_n^{k+1} is within 0.01-tolerance

0.95-confidence interval:         [0.178, 0.255]
Test of "psi(P_0)=0":                 p-value = 0
Test of "psi(P_0)=phi(P_0)":        p-value = 0.00887
 (estimated phi(P_0):        0.171)
```

The basic summary of a NPVI object like npvi, as shown above, presents:

- the sample size;
- the value of the TMLE estimator of $\Psi_f(P)$ and its estimated standard deviation;
- the fine-tuning of the convergence criteria, number of iterations and reasons for stopping the iterative procedure;
- a confidence interval for $\Psi_f(P)$ (default 95%-confidence level), and the *p*-values of the two-sided tests of "$\Psi_f(P) = 0$" and "$\Psi_f(P) = \Phi_f(P)$" (the estimate of $\Phi_f(P)$ is also provided).

It is possible to specify another confidence level.

```
> setConfLevel(npvi, 0.9)
> npvi
```

```
NPVI object:

Sample size: 200

Estimator of psi:                 0.217
Estimated standard error:         0.278

Convergence criteria:
- scaled empirical mean of estimating function                < 0.01
- TV distance between P_n^k and P_n^{k+1}                 < 0.01
- change between successive values of "psi"                 < 0.1

Convergence reached after 2 iteration(s) because:
        TV distance between P_n^k and P_n^{k+1} is within 0.01-tolerance
```

```
0.9-confidence interval:        [0.184, 0.249]
Test of "psi(P_0)=0":                p-value = 0
Test of "psi(P_0)=phi(P_0)":        p-value = 0.00887
 (estimated phi(P_0):        0.171)
```

A more comprehensive history of the TMLE procedure can be easily obtained from a `NPVI` object like `npvi`. The content of each column is given in the documentation of the `getHistory` function.

```
> history <- getHistory(npvi)
> print(round(history, 4))
```

```
         eps    lli   mic1 epsT lliT    mic2    psi psi.sd  psiPn psiPn.sd
step0     NA     NA 0.0044   NA   NA  0.0215 0.1893 0.2815 0.1937   0.2640
step1 0.3493 0.8932 0.0053   NA   NA  0.0015 0.2100 0.3049 0.2153   0.2924
step2 0.0896 0.0610 0.0039   NA   NA -0.0036 0.2168 0.3157 0.2207   0.2996
         mic    div    sic   phi sicAlt
step0 0.0258     NA 0.2811 0.171 0.2506
step1 0.0068 0.0149 0.2772 0.171 0.2464
step2 0.0003 0.0044 0.2781 0.171 0.2473
```

The following lines produce the plot shown in Figure 1.

```
> hp <- history[, "psi"]
> hs <- history[, "sic"]
> hs[1] <- NA
> ics <-  c(-1,1) %*% t(qnorm(0.975)*hs/sqrt(nrow(getObs(npvi))))
> pch <- 20
> ylim <- range(c(confInt, hp, ics+hp), na.rm=TRUE)
> xs <- (1:length(hs))-1
> plot(xs, hp, ylim=ylim, pch=pch, xlab="Iteration", ylab=expression(psi[n]),
+      xaxp=c(0, length(hs)-1, length(hs)-1))
> dummy <- sapply(seq(along=xs), function(x) lines(c(xs[x],xs[x]), hp[x]+ics[, x]))
> abline(h=confInt, col="blue")
> abline(h=confInt0, col="red")
```

4 Session information

```
> sessionInfo()
```

```
R Under development (unstable) (2014-02-06 r64933)
Platform: x86_64-apple-darwin10.8.0 (64-bit)
```

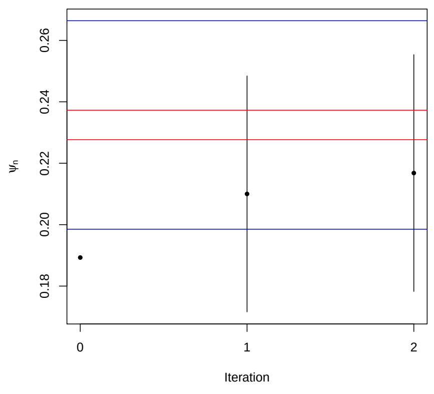

Figure 1: A visual summary of the TMLE procedure contained in `npvi`. The black dots represent the successive values of $\Psi_f(P_n^k)$ for $k = 0, 1, 2$, where $\Psi_f(P_n^2)$ is the TMLE of $\Psi_f(P^s)$ based on $n = 200$ observations (here, f = identity). The black vertical lines represent 95%-confidence intervals for $\Psi_f(P^s)$. The red (blue, respectively) lines represent the interval approximately centered at the truth $\Psi_f(P^s)$ with a length approximately equal to the smallest possible length of confidence interval based on 10000 (200, respectively) observations.

```
locale:
[1] C/fr_FR.UTF-8/fr_FR.UTF-8/C/fr_FR.UTF-8/fr_FR.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] tmle.npvi_0.8.1   sgeostat_1.0-25   MASS_7.3-29       R.utils_1.28.4
[5] R.oo_1.17.0       R.methodsS3_1.6.1

loaded via a namespace (and not attached):
[1] SuperLearner_2.0-10 nnls_1.4            tools_3.1.0
```

References

[Chambaz et al., 2012] Chambaz, A., Neuvial, P., and van der Laan, M. J. (2012). Estimation of a non-parametric variable importance measure of a continuous exposure. *Electronic Journal of Statistics*, 6:1059–1099.

[Polley and van der Laan, 2011] Polley, E. and van der Laan, M. J. (2011). *SuperLearner*. R package version 2.0-4.

[The Cancer Genome Atlas (TGCA) research Network, 2008] The Cancer Genome Atlas (TGCA) research Network (2008). Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455:1061–1068.

[van der Laan et al., 2007] van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). Super learner. *Stat. Appl. Genet. Mol. Biol.*, 6:Article 25.