Notes on joint p-values

Federico Nutarelli,

federico.nutarelli@imtlucca.it

Introduction

The subsequent remarks are founded upon the Sorted Effects method as expounded in the work of Chernozhukov et al. (2018)[1]. With respect to Chernozhukov et al. (2018) they offer the possibility of acting also in contexts where counterfactuals are estimated via machine learning and heterogeneity analysis is conducted without a contemporaneous control group. See Duenas M. (2022) for further details. As far as I am aware, no publicly available code exists for computing joint p-values in Stata. Furthermore, it is worth noting that currently, there is no publicly available code that effectively calculates the joint p-values in situations where an additional layer of uncertainty arises due to the absence of a contemporaneous control group.
Given the close relationship between the notion of joint p-values and the research conducted by Chernozhukov et al. (2018), I assume that the reader is familiar with the content of the aforementioned paper, particularly with regards to the representations of Classification Analysis (CA) and CADiff. However, for the sake of clarity, a concise reminder will be provided. In the ensuing discussion, the variable T denotes the treatment applied to an outcome variable Y, while simultaneously controlling for certain variables denoted as W. I also provide an application in the particular case where counterfactuals are estimated via machine learning and heterogeneity analysis is conducted without a contemporaneous control group. The latter application is drawn from our paper Duenas M. (2022).
Following Chen et al. (2019), *"The Sorted Partial Effects (SPEs) are also useful to conduct classification analysis (CA). This analysis consists of two steps. First, classify the observational units with Partial Effects (PEs) above or below some thresholds defined by tail SPEs in the most or least affected groups. Second, report and compare the moments or distributions of the outcome and covariates in the two groups"*.
These notes aim to provide a straightforward explanation of the rationale behind joint p-values in the context of CADiff sorted effects and the algorithm implemented to construct them with connections to the theoretical paper of Chernozhukov et al. (2018). In this context, it is assumed that CA estimates are computed for all values of t within the set $\mathscr{T}$. It should be noted that this assumption is not of significant concern since, in principle, the CA could be constructed without p-values, and subsequently, the results of the CA could be used to construct the CADiff with p-values. The CADiff, in essence, represents the estimated CA differences for each variable.
For the sake of clarity, I will now elaborate on the meanings of the key symbols employed in this discussion:

1. SPEs are percentiles of the of the PE of T on Y, holding W fixed at w, defined as $\Delta(X) = g(t_1, w) - g(t_0, w)$ in the population of interest, i.e. $\Delta^*_\mu(u) = u^{th} - quantile\ of\ \Delta(X), \quad X \sim \mu$

2. $\Lambda^u_{\Delta,\mu}(t) = [\Lambda^{-u}_{\Delta,\mu}(t), \Lambda^{+u}_{\Delta,\mu}(t)]$ represents the effects of interest along the $t \in \mathscr{T}$ variables. Specifically, $\Lambda^{-u}_{\Delta,\mu}(t), \Lambda^{+u}_{\Delta,\mu}(t)$ are generic objects indexed by t in the least and most affected subpopulations, respectively. Following the example of Chernozhukov et al. (2018), for instance $\Lambda^{-u}_{\Delta,\mu}(t)$ may represent the the t-moment of Z (e.g. the average if $t = 1$) in the u-least affected subpopulation, i.e. $E_\mu[Z^t | \Delta(X) \leq \Delta^*_\mu(u)]$

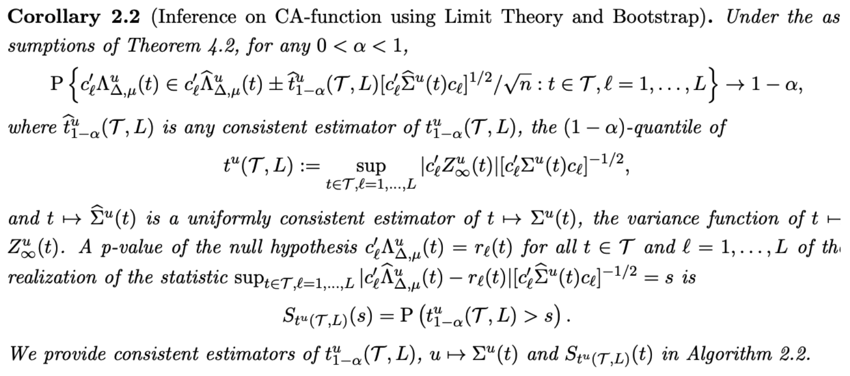
3. c'_l represents linear combinations of a random variable. r_l is a vector of conditions. In general, we have that a condition on $\Lambda^u_{\Delta,\mu}(t)$ looks like: $c'_l \Lambda^u_{\Delta,\mu}(t) = r_l(t)$. In our case, this simplifies to $c'_l \Lambda^u_{\Delta,\mu}(t) = \mathbf{0}$, being $c'_l = [-1, 1]$, i.e. $\Lambda^{+u}_{\Delta,\mu}(t) = \Lambda^{-u}_{\Delta,\mu}(t)$

4. The $\hat{}$ (hat) variables are the estimated ones

[1]The notation of the present notes is consistent with the notation used in the Arxiv paper of 2018 in order to adhere to the notation of the R code article Chen et al. (2019).

5. (Needed for algorithm 2.2). The weights $(\omega_1, \dots, \omega_n)$ primarily serve as a formalism to designate nonnegative random variables that are independent of the data, while adhering to the conditions stipulated in the asymptotic literature. For example, $(\omega_1, \dots, \omega_n)$ is a multinomial vector with dimension n and probabilities $(1/n, ..., 1/n)$ in the empirical bootstrap. As we will see, Algorithm 2.2 (as well as all the bootstrap involving algorithms) begins with "drawing a realization of $(\omega_1, \dots, \omega_n)$", meaning (broadly), assigning a resampling weight to all the data, where the probability of being assigned weights is $1/n$ for all observations in principle. Therefore, it can be interpreted, to the scope of the present notes, as performing a "bootstrap repetition."

After considering these clarifications, the crucial aspect (pertaining to p-value computation) emphasized in the theoretical paper of Chernozhukov et al. (2018), is as follows:



Corollary 2.2 (Inference on CA-function using Limit Theory and Bootstrap). *Under the as sumptions of Theorem 4.2, for any* $0 < \alpha < 1$,

$$\mathrm{P}\left\{c'_\ell \Lambda^u_{\Delta,\mu}(t) \in c'_\ell \widehat{\Lambda}^u_{\Delta,\mu}(t) \pm \widehat{t}^u_{1-\alpha}(\mathcal{T}, L)[c'_\ell \widehat{\Sigma}^u(t) c_\ell]^{1/2}/\sqrt{n} : t \in \mathcal{T}, \ell = 1, \dots, L\right\} \to 1 - \alpha,$$

where $\widehat{t}^u_{1-\alpha}(\mathcal{T}, L)$ *is any consistent estimator of* $t^u_{1-\alpha}(\mathcal{T}, L)$, *the* $(1-\alpha)$*-quantile of*

$$t^u(\mathcal{T}, L) := \sup_{t \in \mathcal{T}, \ell = 1, \dots, L} |c'_\ell Z^u_\infty(t)| [c'_\ell \Sigma^u(t) c_\ell]^{-1/2},$$

and $t \mapsto \widehat{\Sigma}^u(t)$ *is a uniformly consistent estimator of* $t \mapsto \Sigma^u(t)$, *the variance function of* $t \vdash$ $Z^u_\infty(t)$. *A p-value of the null hypothesis* $c'_\ell \Lambda^u_{\Delta,\mu}(t) = r_\ell(t)$ *for all* $t \in \mathcal{T}$ *and* $\ell = 1, \dots, L$ *of th* *realization of the statistic* $\sup_{t \in \mathcal{T}, \ell = 1, \dots, L} |c'_\ell \widehat{\Lambda}^u_{\Delta,\mu}(t) - r_\ell(t)| [c'_\ell \widehat{\Sigma}^u(t) c_\ell]^{-1/2} = s$ *is*

$$S_{t^u(\mathcal{T}, L)}(s) = \mathrm{P}\left(t^u_{1-\alpha}(\mathcal{T}, L) > s\right).$$

We provide consistent estimators of $t^u_{1-\alpha}(\mathcal{T}, L)$, $u \mapsto \Sigma^u(t)$ *and* $S_{t^u(\mathcal{T}, L)}(t)$ *in Algorithm 2.2.*

As described in Chernozhukov et al., 2018, the above corollary *provides uniform bands that cover L linear combinations of the 2-dimensional vector $\Lambda^u_{\Delta,\mu}(t)$ with coefficients $c_1, \cdots, c_L$ simultaneously over $t \in \mathscr{T}$ with pre-specified probability in large samples*. Notice that Z^u_∞ is a generic centered Gaussian process in t. Remember that a Gaussian process is a family of random variables, indexed by some index set, and such that each finite linear combination of such random variables is Gaussian. In our case, the CA process converges in distribution to a centered bivariate Gaussian process, namely $\sqrt{\hat{\Lambda}^u_{\Delta,\mu}(t) - \Lambda^u_{\Delta,\mu}(t)} \sim Z^u_\infty$. In Corollary 2.2, therefore, the L linear combinations of the 2-dimensional vector $\Lambda^u_{\Delta,\mu}(t)$ – i.e. $c'_l \Lambda^u_{\Delta,\mu}(t)$– converge also to Z^u_∞. As a consequence, the **true** quantity $t^u(\mathscr{T}, \mathscr{L})$ in the corollary [2], will depend on the asymptotic values of $\Lambda^u_{\Delta,\mu}(t)$, i.e. Z^u_∞ and Σ^u and will, hence, converge to the asymptotic process Z^u_∞ [3].

How to compute the joint p-values in general

The main effort lies in recovering $S_{t^u(\mathscr{T}, \mathscr{L})}(s)$. Remember that, at the end of the day, this is a p-value, hence, we will have an estimate (that is $t^u_{(1-\alpha)}(\mathscr{T}, \mathscr{L})$) and an estimator (that is s) in parentheses.
The idea of how to compute these joint p-values is given in Algorithm 2.2 of the paper. Its objective is to compute the joint p-value: $P\left(t^u_{1-\alpha}(\mathscr{T}) > sup_{t \in \mathscr{T}} \frac{|c'_l \hat{\Lambda}^u_{\Delta,\mu}(t)|}{[c'_l \hat{\Sigma}^u c_l]^{\frac{1}{2}}}\right)$:

[2]These are nothing else but the t-values computed as by their definition. They are useful because Confidence Intervals (C.I.) are computed via t-values. these provide more robust C.I. under certain circumstances, see e.g. Knezevic (2008) for an example.

[3]The R paper (Chen et al., 2019) may contain an oversight in its suggestion for computing the p-value. Specifically, the R paper just presents, on page 140, the equation of $S_{t^u(\mathscr{T}, \mathscr{L})}(s)$. In this context, the numerator of s is wrongly reported as a quadratic form o the type: $|c' \Lambda^u_{\Delta,\mu}(t) c|$ which is not possible not being $\Lambda^u_{\Delta,\mu}(t)$ a square matrix (at least I cannot see it anywhere). As a further confirmation of this, above, the part of the numerator in absolute value is $c'_l \hat{\Lambda}^u_{\Delta,\mu}(t) - r_l(t)$

Algorithm 2.2 (Bootstrap law of $t(\mathcal{T}, L)$, quantiles and p-values)**.** ***1)*** *Draw a realization of the bootstrap weights $(\omega_1, \ldots, \omega_n)$.* ***2)*** *For each $t \in \mathcal{T}$, compute $\widetilde{\Lambda}^{-u}_{\Delta,\mu}(t) = \Lambda^{-u}_{\widetilde{\Delta},\widetilde{\mu}}(t)$ and $\widetilde{\Lambda}^{+u}_{\Delta,\mu}(t) = \Lambda^{+u}_{\widetilde{\Delta},\widetilde{\mu}}(t)$, a bootstrap draw of $\widehat{\Lambda}^{-u}_{\Delta,\mu}(t) = \Lambda^{-u}_{\widehat{\Delta},\widehat{\mu}}(t)$ and $\widehat{\Lambda}^{+u}_{\Delta,\mu}(t) = \Lambda^{+u}_{\widehat{\Delta},\widehat{\mu}}(t)$, where $\widetilde{\Delta}$ and $\widetilde{\mu}$ are the bootstrap versions of $\widehat{\Delta}$ and $\widehat{\mu}$ that use $(\omega_1, \ldots, \omega_n)$ as sampling weights in the computation of the estimators. Construct a bootstrap draw of $Z^u_\infty(t)$ as $\widetilde{Z}^u_\infty(t) = \sqrt{n}(\widetilde{\Lambda}^u_{\Delta,\mu}(t) - \widehat{\Lambda}^u_{\Delta,\mu}(t))$, where $\widetilde{\Lambda}^u_{\Delta,\mu}(t) = [\widetilde{\Lambda}^{-u}_{\Delta,\mu}(t), \widetilde{\Lambda}^{+u}_{\Delta,\mu}(t)]$.* ***3)*** *Repeat steps (1)-(2) B times.* ***4)*** *For each $t \in \mathcal{T}$ and $\ell = 1, \ldots, L$, compute a bootstrap estimator of $[c'_\ell \Sigma^u(t) c_\ell]^{1/2}$ such as the bootstrap interquartile range rescaled with the normal distribution $[c'_\ell \widehat{\Sigma}^u(t) c_\ell]^{1/2} = (q^u_{0.75}(t,\ell) - q^u_{0.25}(t,\ell))/(z_{0.75} - z_{0.25})$, where $q^u_p(t,\ell)$ is the pth sample quantile of $c'_\ell \widetilde{Z}^u_\infty(t)$ in the B draws and z_p is the pth quantile of $N(0,1)$.* ***5)*** *Use the empirical distribution of $\widetilde{t}(\mathcal{T}, L) = \sup_{t\in\mathcal{T},\ell=1,\ldots,L} |c'_\ell \widetilde{Z}^u_\infty(t)| [c'_\ell \widehat{\Sigma}^u(t) c_\ell]^{-1/2}$ across the B draws to approximate the distribution of $t(\mathcal{T}, L) = \sup_{t\in\mathcal{T},\ell=1,\ldots,L} |c'_\ell Z^u_\infty(t)| [c'_\ell \Sigma^u(t) c_\ell]^{-1/2}$. In particular, construct $\widehat{t}_{1-\alpha}(\mathcal{T}, L)$, an estimator of $t_{1-\alpha}(\mathcal{T}, L)$, as the $(1-\alpha)$-quantile of the B draws of $\widetilde{t}(\mathcal{T}, L)$,*

and an estimation of the p-value $S_{t^u(\mathcal{T},L)}(s)$ as the proportion of the B draws of $\widetilde{t}(\mathcal{T}, L)$ that are greater than s.

For a full understanding of the algorithm, remember that $\widehat{\Delta}(X)$ represents the estimate of the PE. Therefore, if we define $\Lambda^{-u}_{\Delta,\mu}(t)$ as $E_\mu[Z^t|\Delta(X) \leq \Delta^*_\mu(u)]$ (see above), we will have simply that $\widehat{\Lambda}^{-u}_{\Delta,\mu}(t) = \Lambda^{-u}_{\widehat{\Delta},\widehat{\mu}}(t) = E_{\widehat{\mu}}[Z^t|\widehat{\Delta}(X) \leq \widehat{\Delta}^*_{\widehat{\mu}}(u)]$. In plain English this is the expected value of a certain moment of Z (say the average), for the observations whose PE lies below the u^{th} *− quantile of the estimated PE, i.e.* $\widehat{\Delta}(X)$. The quantity $\widehat{\Delta}(X)$ functions as a "true" value (as demonstrated in its asymptotic validity in Chernozhukov et al., 2018) for the purpose of comparison with bootstrap estimates[4]. As a consequence, all the quantities depending on $\widehat{\Delta}(X)$ can also be computed via bootstrap and their bootstrapped value can be compared to the "true" estimated value denoted with an hat. Following this logic one can also simulate a draw from $Z^u_\infty(t)$ as $\tilde{Z}^u_\infty(t) = \sqrt{n}(\tilde{\Lambda}^u_{\Delta,\mu}(t) - \widehat{\Lambda}^u_{\Delta,\mu}(t))$. In other words, a bootstrapped version of $Z^u_\infty(t)$ is computed so that we can draw from it simulating a draw from the actual $Z^u_\infty(t)$ assuming that B is sufficiently high. This is useful to construct the first block of $t^u_{1-\alpha}(\mathscr{T}, \mathscr{L})$ of Corollary 2.2 above, i.e. $[c'_l Z^u_\infty(t)]$.
Specifically Algorithm 2.2 does the following:

1. As anticipated, the reference to weights $(\omega_1, \cdots, \omega_n)$ serves as a reference for bootstrap sampling (i.e. they assign a resampling weight to all the data, where the probability of being assigned weights is $1/n$ for all observations). To make joint bootstrap standard errors we should take, from the SPE the most and least affected groups and construct bootstrap versions $\tilde{\Delta}, \tilde{\mu}$ of $\hat{\Delta}, \hat{\mu}$ in order to compute the bootstrap $\tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu}}(t)$. This must be done $\forall t \in \mathscr{T}$.

2. Construct a bootstrap draw of $Z^u_\infty(t)$. How can we do that? Remember that $Z^u_\infty(t)$ is nothing else but the distribution of $\hat{\Lambda}^{-u}_{\hat{\Delta},\hat{\mu}}(t) - \Lambda^{-u}_{\Delta,\mu}(t)$. So, we can use the bootstrap equivalent to construct an estimate of the latter. In particular, at this point we have the estimated value of $\Lambda^u_{\Delta,\mu}$ (i.e. the estimate of the SPE obtained in the sample where we are drawing), $\hat{\Lambda}^{-u}_{\hat{\Delta},\hat{\mu}}(t)$, and its bootstrap equivalent. So $\tilde{Z}_\infty(t) = \sqrt{n}(\tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu}}(t) - \hat{\Lambda}^{-u}_{\hat{\Delta},\hat{\mu}}(t))$.

3. Repeat (1) and (2), B times so that a good approximation of $Z^u_\infty(t)$ drawings are constructed.

4. The fourth step constructs an estimate of the second building block of $t^u_{1-\alpha}(\mathscr{T}, \mathscr{L})$ of Corollary 2.2, i.e. $[c'_l \Sigma^u(t) c_l]^{-1/2}$. This is done, as suggested by the algorithm, exploiting the bootstrap distribution $\tilde{Z}^u_\infty(t)$ and the standard normal distribution $N(0,1)$. Specifically, $[c'_l \Sigma^u(t) c_l]^{-1/2}$ is estimated via $[c'_l \widehat{\Sigma}^u(t) c_l]^{-1/2}$ which is a function of the pth quantile of $c'_l \tilde{Z}_\infty(t)$ and the pth quantile of $N(0,1)$.

5. The fifth step concludes by exploiting the building block constructed in order to provide an estimate $\hat{t}_{1-\alpha}(\mathscr{T}, \mathscr{L})$ of the t-value at $1-\alpha$ confidence.

Correspondences with Git-hub algorithm

Algorithm 2.2 can be readily reproduced through code. In an effort to provide a comprehensive understanding of the original code presented by Chen et al. (2019) in the R language, I present below a pseudo-code. Given our primary

[4]These bootstrap estimates are obtained by averaging over the partial effects (PE) estimated in a quantity denoted as B of bootstrap samples.

focus on inference, I will solely describe the "inference" segment of the Git-hub code by Chen et al. (2019), using a pseudo-code format and making references to Algorithm 2.2. The pseudo-code is structured into three main parts: the first part outlines the computation of the joint p-values; the second part details the process of constructing reliable inference on categories, and finally, the third part provides a conclusion. Comments are in blue. In red I highlighted the important code parts.

Algorithm 1 Inference algorithm pt.1

```
if (interest == "moment") then
    # draws.H is B * mL
    draws_H <- result_boot$t        ▷ In our case, draws_H represents the $B$ observations describing the CA statistic from the bootstrap. In other words this is $\tilde{\Lambda}^u_{\tilde{\Delta},\tilde{\mu}}(t)$
    colnames(draws_H) <- rep(varname, dim(cl)[2])
    Zc <- draws_H - matrix(H, nrow = b, ncol = length(H), byrow = TRUE)        ▷ The above part performs the second step of the algorithm, i.e. $\tilde{Z}_\infty(t) = \sqrt{n}\tilde{\Lambda}^u_{\tilde{\Delta},\tilde{\mu}}(t) - \hat{\Lambda}^u_{\hat{\Delta},\hat{\mu}}(t)$
    sig <- (apply(Zc, 2, quantile, 0.75, na.rm = TRUE) - apply(Zc, 2, quantile, 0.25, na.rm = TRUE)) / (qnorm(0.75)
- qnorm(0.25))        ▷ Here is computed $[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}} = \frac{(q^u_{0.75}(t,l)-q^u_{0.25}(t,l))}{(z_{0.75}-z_{0.25})}$
    # t.tilde is B * mL
    t_tilde <- apply(abs(Zc)/matrix(sig, nrow = b, ncol = length(sig), byrow = TRUE), 1, max, na.rm = TRUE)        ▷ Computed here $\tilde{t}^u(\mathscr{T}) = sup_{t\in T}\frac{c_l'\tilde{Z}_\infty(t)c_l}{[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}}$
     Bias correction
     H_bc <- 2 * H - apply(draws_H, 2, mean, na.rm = TRUE) # 1 * mL
     Calculate the pointwise p-value (non and bc corrected)
    pw_p <- 1 - pnorm(abs(H/sig))
    pw_p_bc <- 1 - pnorm(abs(H_bc/sig))        ▷ They are pointwise
     # Joint p-values        ▷ This is the important part!

     stat <- abs(H_bc)/sig        ▷ Remember that, under the null, we want, $c_l'\hat{\Lambda}^u_{\Delta,\mu}(t) - r_l(t)$. Hence, being, the latter distributed as $Z_\infty$ by construction, we can substitute it in the definition of the t-stat that we want to evaluate to end up with the statistic of interest, i.e. $sup_{t\in T}\frac{c_l'\hat{\Lambda}^u(t)-r_l(t)}{[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}}$, see pag. 13 of Chernozhukov et al. (2018).
    # Proportions of the B draws of t.tilde that are greater than s
    # 1 * mL. First m columns represent joint p-vals for the m vars for the first hypothesis.

     j_pvals <- sapply(stat, epvals, zs = t_tilde)        ▷ "epvals", is a function to obtain empirical p-values with arguments "z" and "zs". In particular it returns: *return(mean(zs > z))*. So basically they are constructing the empirical p-values using as z scores the bootstrapped $\tilde{t}(\mathscr{T},\mathscr{L})$.

    # Not bias-corrected joint p-values
    stat_un <- abs(H)/sig
    j_pvals_un <- sapply(stat_un, epvals, zs = t_tilde)
```

To summarize the last steps (after "# Joint p-values"), Chen et al. (2019) construct $sup_{t\in T}\frac{c_l'\hat{\Lambda}^u(t)-r_l(t)}{[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}}$ as being the statistic for hypohesis testing. They then employ $\tilde{t}(\mathscr{T},\mathscr{L})$ "as if it is a z-score" of the probability distribution (constructed through the bootstrap). In other words, they are collecting the average number of times in which $\tilde{t}(\mathscr{T},\mathscr{L})$ is greater than a realization, s of the statistic of interest $sup_{t\in T}\frac{c_l'\hat{\Lambda}^u(t)-r_l(t)}{[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}}$ (under the null).

Algorithm 2 Inference algorithm pt.2 (categories)

```
 Now work on p-vals accounting for testing all vars within one category
if (!is.null(cat)) then
    # first tease out characteristics that are not factors
    noncat <- vars[-match(cat, vars)]
    # for noncat subset pointwise p-values from previous results
    if if (dim(cl)[2] == 1) then
        noncat_p <- pw_p[, noncat]
        noncat_p_bc <- pw_p_bc[, noncat]
    end if
else if (dim(cl)[2] == 2) then
    pw_p_most <- matrix(pw_p[, 1:length(varname)], nrow = 1)
    pw_p_least <- matrix(pw_p[, -(1:length(varname))], nrow = 1)
    colnames(pw_p_most) <- colnames(pw_p_least) <- varname
    noncat_p_most <- pw_p_most[, noncat]
    noncat_p_least <- pw_p_least[, noncat]
    pw_p_most_bc <- matrix(pw_p_bc[, 1:length(varname)], nrow = 1)
    pw_p_least_bc <- matrix(pw_p_bc[, -(1:length(varname))], nrow = 1)
    colnames(pw_p_most_bc) <- colnames(pw_p_least_bc) <- varname
    noncat_p_most_bc <- pw_p_most_bc[, noncat]
    noncat_p_least_bc <- pw_p_least_bc[, noncat]
end if
Now we calculate joint p-values within each factor category for each factor, get the dummy names, subset
from the bootstrap matrix and conduct inference
newfunc <- function(x, bootstrap_mat, H, b, cl, varname) {
# get the dummy name for each factor
name <- colnames(subsetting(data, x))
if  (dim(cl)[2] == 1) then
     # get bootstrap matrix and estimates within each category
    boot_mat <- bootstrap_mat[, name]
    H_mat <- H[, name]
end if
(dim(cl)[2] == 2)
# subset only gets the first group, need to split the matrix and get the second group, too
most <- bootstrap_mat[, 1:length(varname)]
least <- bootstrap_mat[, -(1:length(varname))]
most_H <- matrix(H[1:length(varname)], nrow = 1)
     l
east_H <- matrix(H[-(1:length(varname))], nrow = 1)
colnames(most_H) <- colnames(least_H) <- varname
mat_most <- most[, name]
mat_least <- least[, name]
H_most <- most_H[, name]
H_least <- least_H[, name]
 }
     =0
```

Algorithm 3 Inference algorithm pt.3

```
 # conduct inference
 inference <- function(h, bt, rownum) {
Zc <- bt - matrix(h, nrow = rownum, ncol = length(h), byrow = TRUE)
sig <- (apply(Zc, 2, quantile, 0.75, na.rm = TRUE) -
    apply(Zc, 2, quantile, 0.25, na.rm = TRUE)) / (qnorm(0.75) - qnorm(0.25))
t_tilde <- apply(abs(Zc)/matrix(sig, nrow = b, ncol = length(sig),
    byrow = TRUE), 1, max, na.rm = TRUE)
H_bc <- 2 * h - apply(bt, 2, mean, na.rm = TRUE)
stat <- abs(H_bc)/sig
# bias corrected categorical pvalues
cat_pvals <- sapply(stat, epvals, zs = t_tilde)
stat_un <- abs(h)/sig
# non bias corrected categorical pvalues
cat_pvals_un <- sapply(stat_un, epvals, zs = t_tilde)
out <- cbind(cat_pvals, cat_pvals_un)
if (dim(cl)[2] == 1) then
    pvalues <- inference(h = H_mat, bt = boot_mat, rownum = b)
   }
else if (dim(cl)[2] == 2) then
   pvalues_most <- inference(h = H_most, bt = mat_most, rownum = b)
   pvalues_least <- inference(h = H_least, bt = mat_least, rownum = b)
   pvalues <- cbind(pvalues_most, pvalues_least)
   return(pvalues)
   result <- sapply(cat, newfunc, bootstrap_mat = draws_H, H = H, b = b,
    cl = cl, varname = varname)

Now we tabulate the results
```

How to compute the joint p-values in the case of Duenas et al. (2022) (an idea)

I try to sketch here a step by step procedure in our case (see Duenas et al. (2022) for further details). Here we assume that the results of CA analysis are available. Basically, we should repeat the CA analysis B times (by drawing data from the original sample where the CA has been done) to obtain B estimates of the least and most affected units for a variable t.

- Take the observations of a single bootstrap iteration;
- Perform the CA $\forall t \in \mathscr{T}$ and obtain the values of the variables of interest for least and most affected units, namely $\tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu}}(t)$ and $\tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu}}(t)$;
- Compute $\sqrt{n}(\tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu}}(t) - \hat{\Lambda}^{-u}_{\hat{\Delta},\hat{\mu}}(t))$, where $\hat{\Lambda}^{-u}_{\hat{\Delta},\hat{\mu}}(t) - \Lambda^{-u}_{\Delta,\mu}(t)$ and $\hat{\Lambda}^{+u}_{\hat{\Delta},\hat{\mu}}(t) - \Lambda^{-u}_{\Delta,\mu}(t)$ are the original estimates of the CA for the variable t (i.e. the CA effects of variable t estimated from the original sample from which we made the bootstrap);
- Of course we have to repeat the previous steps for all the B iterations. Hence, we end up with B bootstrap estimates $\tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu}}(t)$ and $\tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu}}(t)$ for each variable t. For the sake of simplicity, let's focus on a **single** variable, say,

t_1. In such a case, we end up with a $B \times 2$ (sub) database of the form:

$$\left(\begin{bmatrix} \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},1}(t_1) & \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},1}(t_1) \\ \vdots & \\ \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},k}(t_1) & \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},k}(t_1) \\ \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},k+1}(t_1) & \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},k+1}(t_1) \\ \vdots & \\ \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},B}(t) & \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},B}(t_1) \end{bmatrix}\right) \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} B/2 \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} B/2 \end{matrix}$$

The same for the differences among the two.

- Now we have to compute an estimator for the variance (i.e., the denominator of the statistic of interest). Chernuzokov et al. (2018) provide a simple estimator for that as we saw in the explanation of Algorithm 2.2 point 4 [5]: $[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}} = \frac{(q^u_{0.75}(t,l) - q^u_{0.25}(t,l))}{(z_{0.75} - z_{0.25})}$. The denominator can be found on the standard normal tables (z_p is nothing else but the p^{th} quantile of the standard normal distribution $N(0,1)$ as aforementioned). How can we find the numerator of $[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}$ then? The numerator is composed by $q^u_p(t,l)$ being the p^{th} sample quantile of $\tilde{Z}_\infty$ in the B draws. So, since we have the distribution of $\tilde{Z}_\infty$ for t_1 (for sake of simplicity stay with a single variable t_1) as:

$$\left(\begin{bmatrix} \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},1}(t_1) - \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},1}(t_1) \\ \vdots \\ \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},k}(t_1) - \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},k}(t_1) \\ \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},k+1}(t_1) - \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},k+1}(t_1) \\ \vdots \\ \tilde{\Lambda}^{-u}_{\tilde{\Delta},\tilde{\mu},B}(t) - \tilde{\Lambda}^{+u}_{\tilde{\Delta},\tilde{\mu},B}(t_1) \end{bmatrix}\right) \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} B/2 \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} B/2 \end{matrix}$$

  From this distribution we can easily extrapolate the 0.25 and 0.75 quantile (just sort the above vector for all $t \in \mathscr{T}$ and extract the quantiles). Now we have $[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}$;

- Once we have $[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}$, we will use $\tilde{t}^u(\mathscr{T}) = sup_{t \in T} \frac{c_l'\tilde{Z}_\infty(t)c_l}{[c_l'\hat{\Sigma}c_l]^{\frac{1}{2}}}$ across the B draws to approximate the distribution of $t(\mathscr{T},\mathscr{L})$. Specifically, since the numerator of $\tilde{t}^u(\mathscr{T})$ depends on the bootstrap samples (due to $\tilde{Z}^u_\infty$), we will have B $\tilde{t}^u(\mathscr{T})$ (i.e. a distribution). We can, therefore, 1) approximate $\hat{t}_{(1-\alpha)}(\mathscr{T},\mathscr{L})$ as the $(1-\alpha)$ quantile of the B draws of $\tilde{t}^u(\mathscr{T})$:

$$\tilde{t}^u = \left(\begin{bmatrix} \tilde{t}^u_1 \\ \vdots \\ \tilde{t}^u_k \\ \tilde{t}^u_{k+1} \\ \vdots \\ \tilde{t}^u_B \end{bmatrix}\right) \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} B/2 \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} B/2 \end{matrix}$$

  The p-value of interest can be computed as the proportion of the B draws of $\tilde{t}^u(\mathscr{T})$ that are greater than s. How to compute s? See the Algorithm 2.2. Its quantities can be computed following the above instructions as well.

The STATA code provided basically follows the steps described in this last section.

[5]Remember that the z scores of the normal can be easily found in statistic tables or online.

References

Chen, Shuowen, Victor Chernozhukov, Iván Fernández-Val, and Ye Luo (2019). “SortedEffects: sorted causal effects in R”. In: *arXiv preprint arXiv:1909.00836*.

Chernozhukov, Victor, Iván Fernández-Val, and Ye Luo (2018). “The sorted effects method: discovering heterogeneous effects beyond their averages”. In: *Econometrica* 86.6, pp. 1911–1938.

Duenas M. Nutarelli F., Ortiz-Gimenez V. et al. (2022). “Assessing the Heterogeneous Impact of Global Trade Shocks”. In: *Under Review*.

Knezevic, Andrea (2008). “Overlapping confidence intervals and statistical significance”. In: *StatNews: Cornell University Statistical Consulting Unit* 73.1.