wqsreg - A Stata command for Weighted Quantile Sum regression

Feb 2026

Marta Ponzano[1], Stefano Renzetti[2], Andrea Bellavia[3]

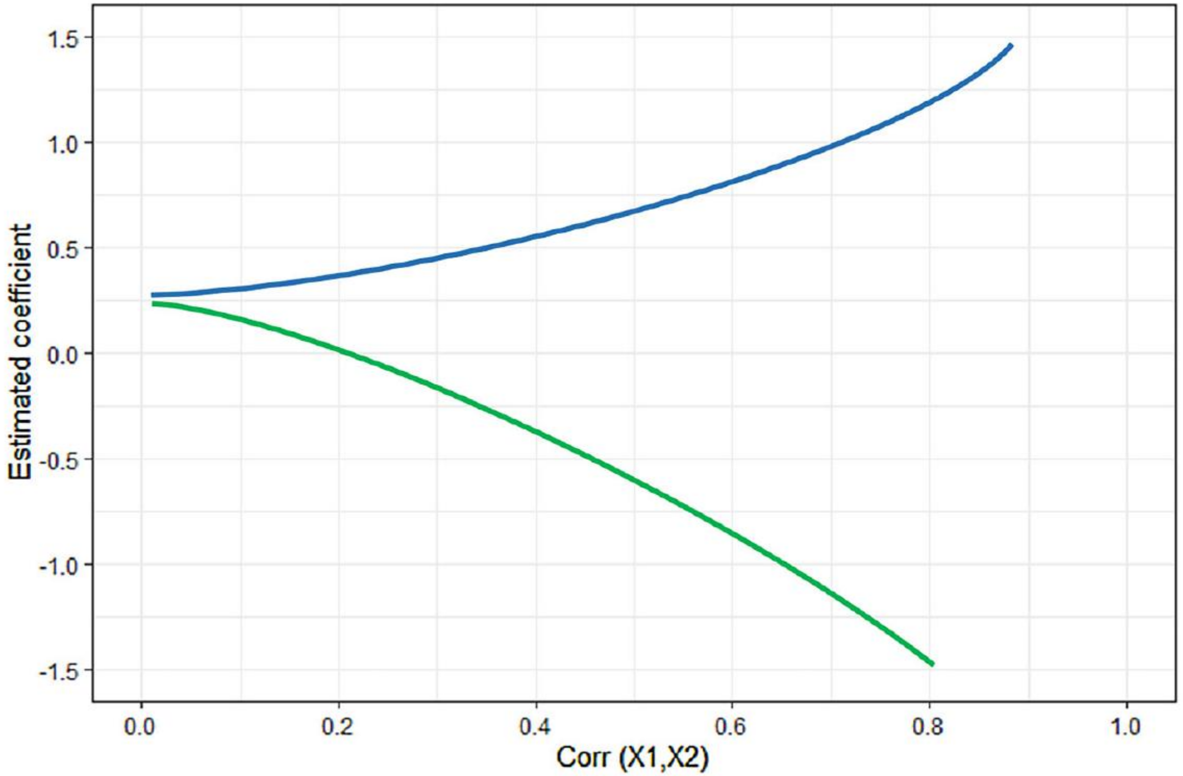
[1] Department of Life Sciences, Health and Health Professions, Link Campus University, Rome, Italy/ Department of Health Sciences, University of Genoa, Genoa, Italy.

[2] Department of Medicine and Surgery, University of Parma, Parma, Italy.

[3] Department of Environmental Health, Harvard T.H. Chan School of Public Health, Boston, Massachusetts, USA / TIMI Study Group, Brigham and Women's Hospital, Harvard Medical School, Boston, Massachusetts, USA

Correlated predictors

- The presence of highly correlated predictors is a well-known issue in numerous domains
- Including highly correlated covariates in the same regression model increases the risk of multicollinearity:
  - ⬇ accuracy
  - ⬆ standard errors



The reversal paradox.

Correlated predictors – environmental epidemiology

- This issue is particularly common in environmental epidemiology, when studying the health effects of a combined group of multiple factors, known as environmental mixtures:

1. Mixture Bad Actors (specific role of each factor)
2. Joint Mixture Effect (overall)

1.

2.

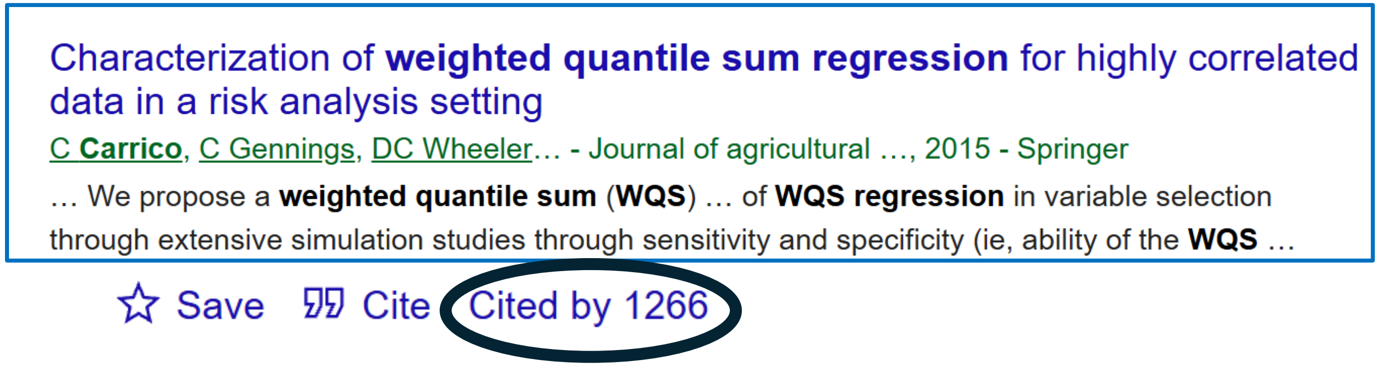
Statistical methods

- To address this complexity, some methodologies have been considered (e.g., penalized regression) and *ad hoc* statistical methods have been developed and then used also in different fields outside of environmental epidemiology:

  - Mixture Indexing approaches

    Weighted Quantile Sum (WQS) Regression, Quantile G-computation, etc.
  - Flexible Approaches for complex settings

    Bayesian Kernel Machine Regression (BKMR), Machine Learning approaches, etc.



Characterization of **weighted quantile sum regression** for highly correlated data in a risk analysis setting

C **Carrico**, C Gennings, DC Wheeler… - Journal of agricultural …, 2015 - Springer

… We propose a **weighted quantile sum** (**WQS**) … of **WQS regression** in variable selection through extensive simulation studies through sensitivity and specificity (ie, ability of the **WQS** …

WQS Regression

- WQS regression is a supervised statistical approach where exposure aggregation via indexing is conducted in two steps:

1. Creating a summary index by weighting each mixture component
2. Including the index in a regression model

1.

2.

1. Creating the summary index - weights estimation

- Split the dataset into training and validation (40/60)
- Score all exposure factors into categories based on quantiles (quartiles)
- Generate *b* bootstrap samples of the training dataset (100)
- For each sample, estimate parameters (weights and βs):

$$f(y) = \beta_0 + \beta_1 \left(\sum_{i=1}^{c} w_i q_i \right) + \boldsymbol{\beta} \cdot \mathbf{C}'$$

$0 \leq w_i \leq 1$; $\sum_{i=1}^{c} w_i = 1$

Assumption of unidirectionality (pre-specified direction): only average results over bootstrap samples with either positive or negative estimates

- Derive $\overline{w}_i$ for each component based on estimates from bootstrap samples
- Derive the final estimate of the WQS as weighted sum of the components

$$WQS = \sum_{i=1}^{c} \bar{w}_i q_i$$

2. Including the index in a regression model

- Fit a regression model in the validation set to estimate the βs

$$f(y) = \beta_0 + \beta_1 \cdot WQS + \boldsymbol{\beta} \cdot \mathbf{C}'$$

WQS Regression in Stata

A Stata command has not been implemented yet. To address this gap, we have developed a new command – **wqsreg**

- ***wqsreg*** fits WQS regression for continuous, binary and count outcomes, while allowing for several flexible components of this framework. It returns estimates from WQS regression, generates plots of the estimated weights, and saves additional related information.

- https://github.com/PonzanoMarta/wqsreg
  net install wqsreg, from("https://raw.githubusercontent.com/PonzanoMarta/wqsreg/master/") replace

wqsreg -Syntax

outcome variable

list of exposure variables

Syntax:

```
wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

list of confounders

wqsreg -Syntax

Number of bootstrap samples (>0):

- boot = 1: no bootstrapping
- default: boot = 100

Validation set percentage ([0, 100)):

- validation = 0: no split
- default: validation = 60

```
Syntax:


wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

wqsreg -Syntax

Definition of quantiles:

- default: q = 4

Uni-directionality constraint:

- b1_neg = 1 : negative
- default: b1_neg = 0

Syntax:

```
wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

random seed:

- default: seed = 0

wqsreg -Syntax

Maximum number of iterations:
- default: conv_maxiter=2000

Tolerance:
- default: conv_vtol= 0.000000001

```
Syntax:


wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

Optimization method:
- technique = 'nr’: Modified Newton–Raphson
- default: technique = 'bfgs' (Broyden–Fletcher–Goldfarb–Shanno)

wqsreg -Syntax

```
Syntax:

wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

Type of model:

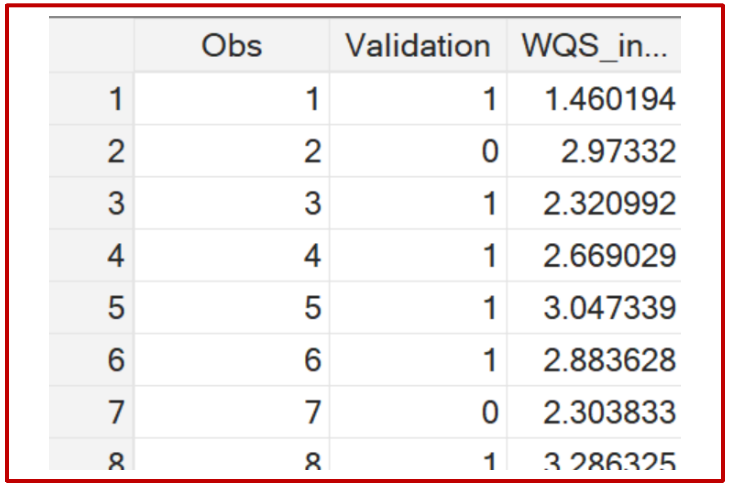
- model_fam='Poisson’
- model_fam ='Logistic’
- default: model_fam='Linear'

wqsreg -Syntax

Name of the dataset that will include the WQS index, only if you want to save it.

```
Syntax:

wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

Variable that identifies the observations



	Obs	Validation	WQS_in...
1	1	1	1.460194
2	2	0	2.97332
3	3	1	2.320992
4	4	1	2.669029
5	5	1	3.047339
6	6	1	2.883628
7	7	0	2.303833
8	8	1	3.286325

wqsreg -Syntax

```
Syntax:

wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

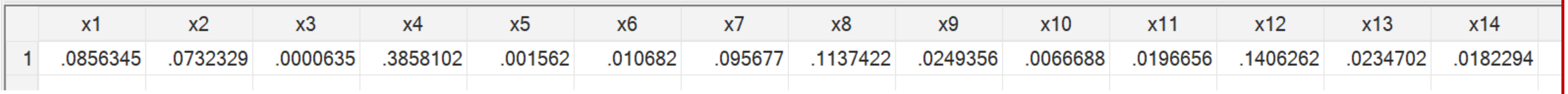
Number of repetitions for repeated holdout validation:

- default: rh_rep= 1 (no repeated holdout validation)

wqsreg -Syntax

Name of the dataset that will include the weights, only if you want to save it

```
Syntax:

wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string 'bfgs')
model_fam(string 'Linear')
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

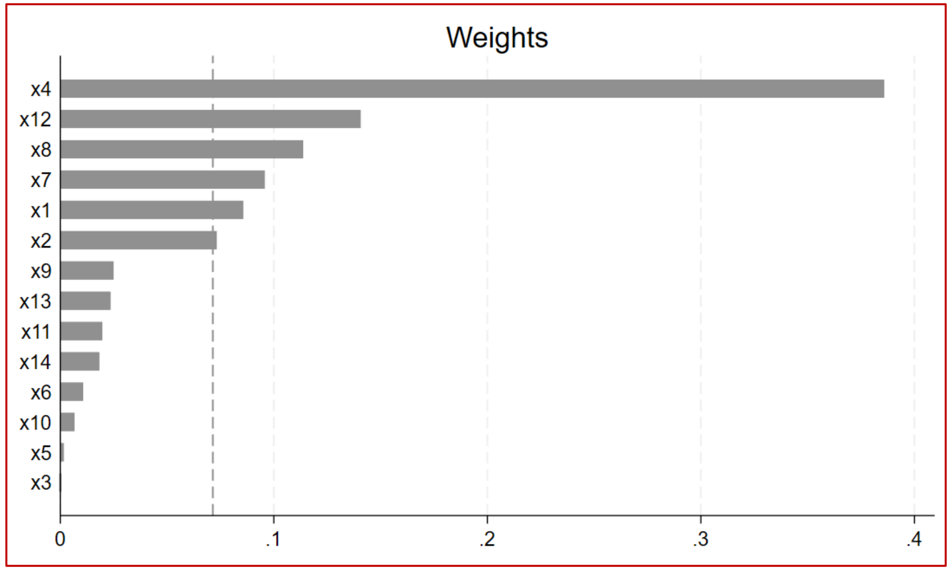


	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14
1	.0856345	.0732329	.0000635	.3858102	.001562	.010682	.095677	.1137422	.0249356	.0066688	.0196656	.1406262	.0234702	.0182294

wqsreg -Syntax

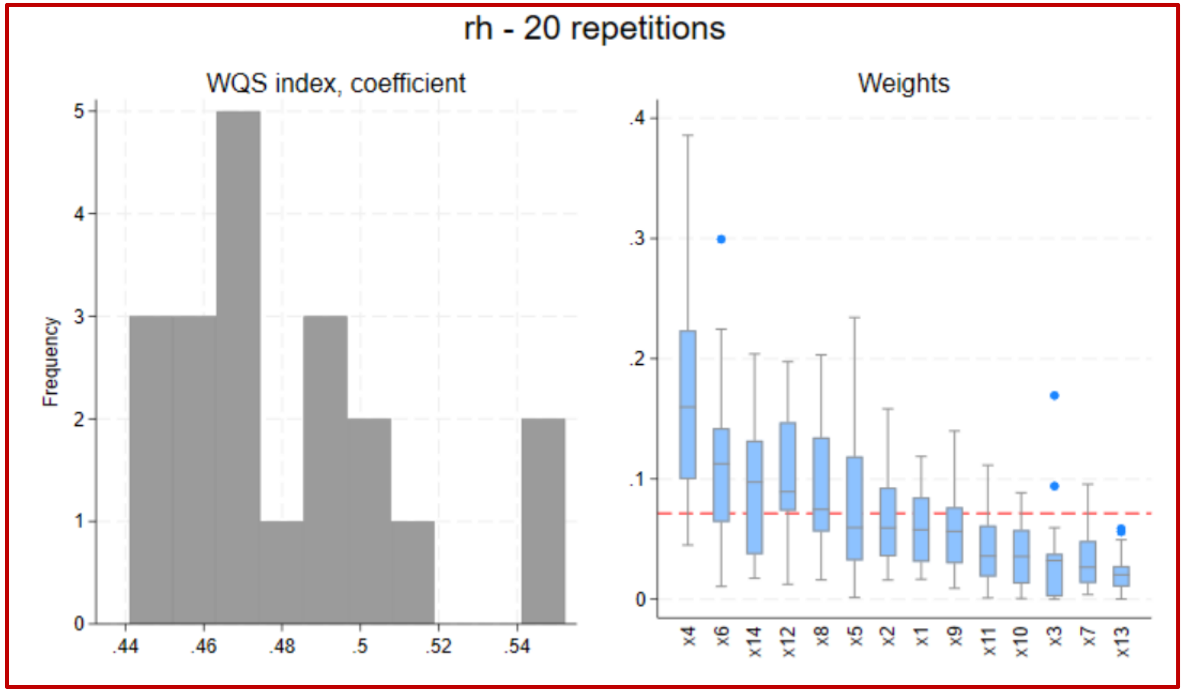
Filename for the plot of weights, only if you want to save it

```
Syntax:


wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```



wqsreg -Syntax

```
Syntax:
wqsreg yvar, mixture(varlist) ///
[boot(integer 100) validation(integer 60) q(integer 4) b1_neg(integer 0)
cvar(varlist) seed(integer 0)
conv_maxiter(integer 2000) conv_vtol(real 0.000000001) technique(string ‘bfgs’)
model_fam(string 'Linear’)
savingWQSindex(string) figureName(string) savingWeights(string)
id(string) rh_rep(integer 1)]
```

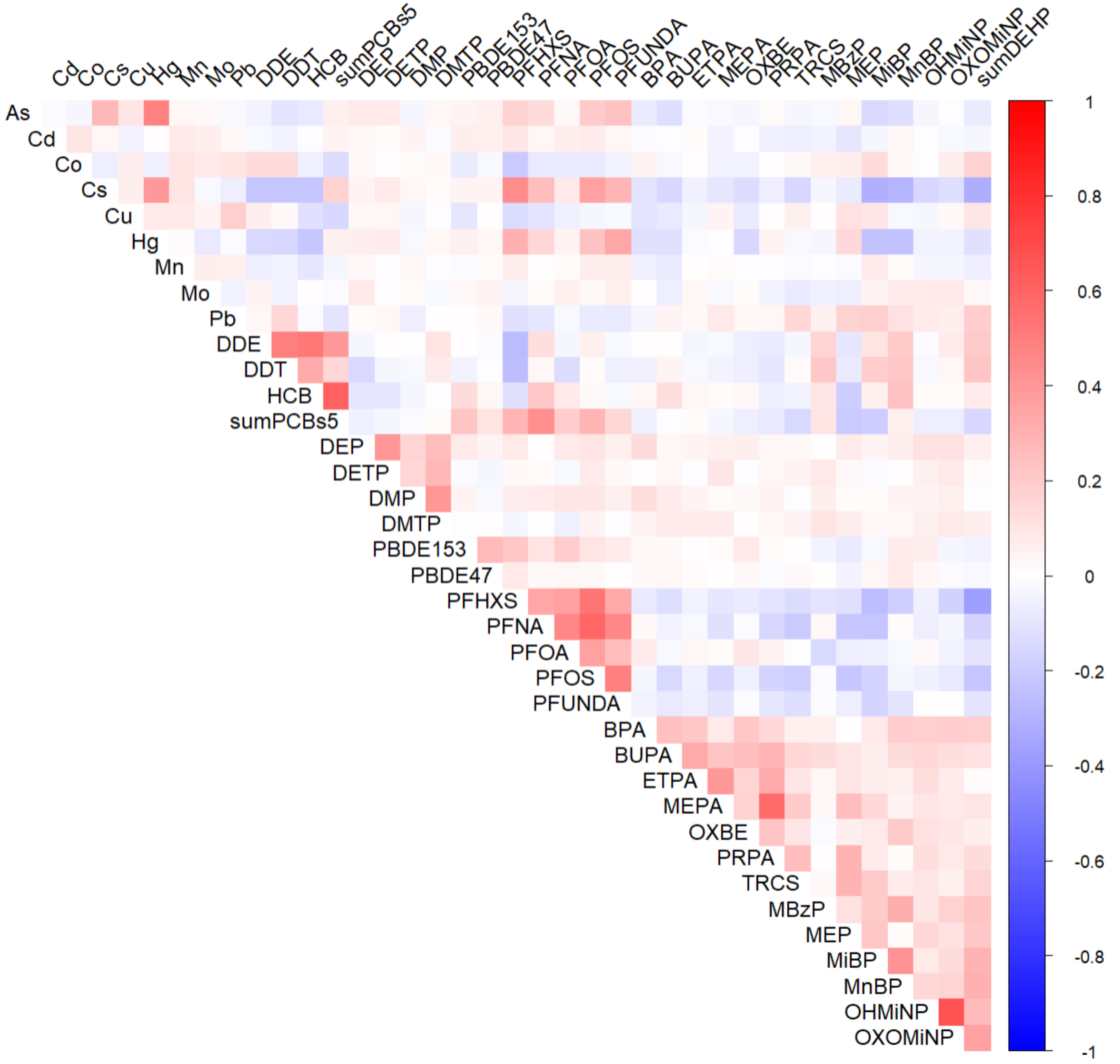


Exposome analysis example

- We present the application of the `wqsreg` command using partially simulated data derived from the early exposome HELIX study.
- This dataset consisted of 1301 mother–child pairs and included several environmental exposure variables, some of which highly correlated.
- We used `wqsreg` to explore the association between postnatal biomonitoring concentrations of 38 chemicals/metabolites and Body Mass Index (BMI) z-score at 6-11 years old while adjusting for a set of covariates.

Exposome analysis example

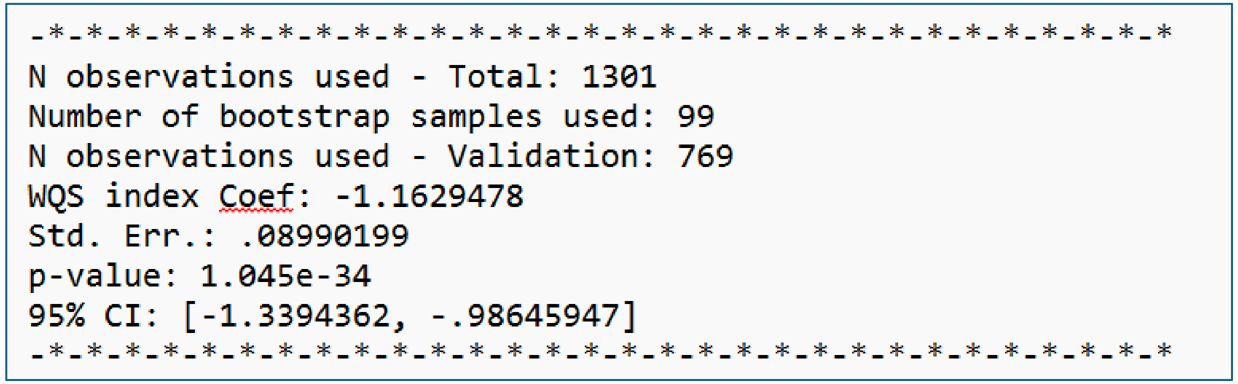
- Postnatal exposures examined included a mixture of metals, organochlorines, organophosphate pesticides, polybrominated diphenyl ethers, per- and polyfluoroalkyl substances, phenols and phthalates.



Exposome analysis example

Model 1. Negative direction- quartiles for exposure categorization, 100 bootstrap samples, 40/60 training-validation split.

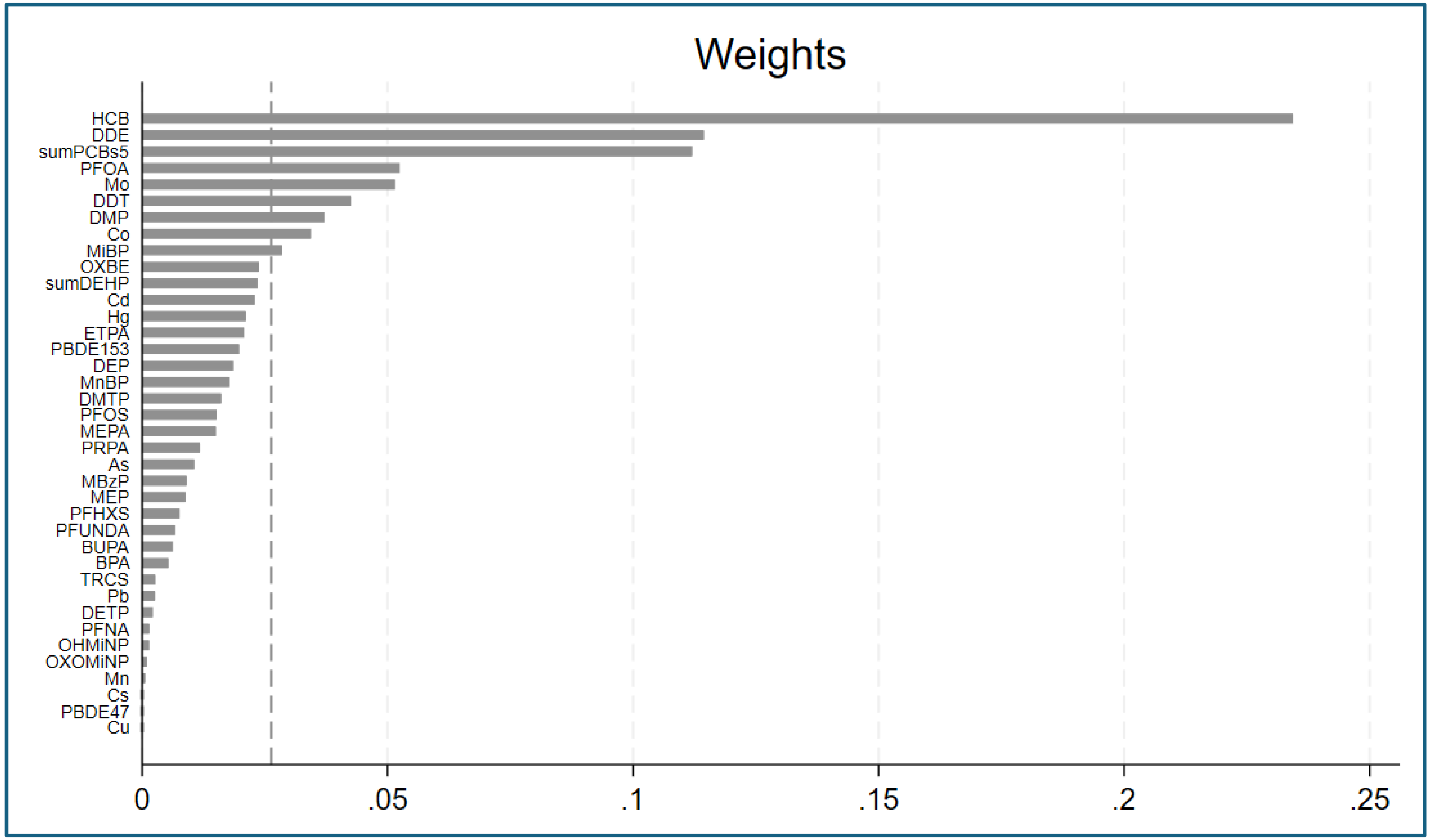
```
wqsreg hs_zbmi_who    , ///
       mixture(`mixture') cvar(`confounders') ///
       b1_neg(1) figureName("Neg")
```



```
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
N observations used - Total: 1301
Number of bootstrap samples used: 99
N observations used - Validation: 769
WQS index Coef: -1.1629478
Std. Err.: .08990199
p-value: 1.045e-34
95% CI: [-1.3394362, -.98645947]
-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

Exposome analysis example

Model 1. Negative direction- quartiles for exposure categorization, 100 bootstrap samples, 40/60 training-validation split.



Exposome analysis example

Model 2. Positive direction- quartiles for exposure categorization, 100 bootstrap samples, 40/60 training-validation split.

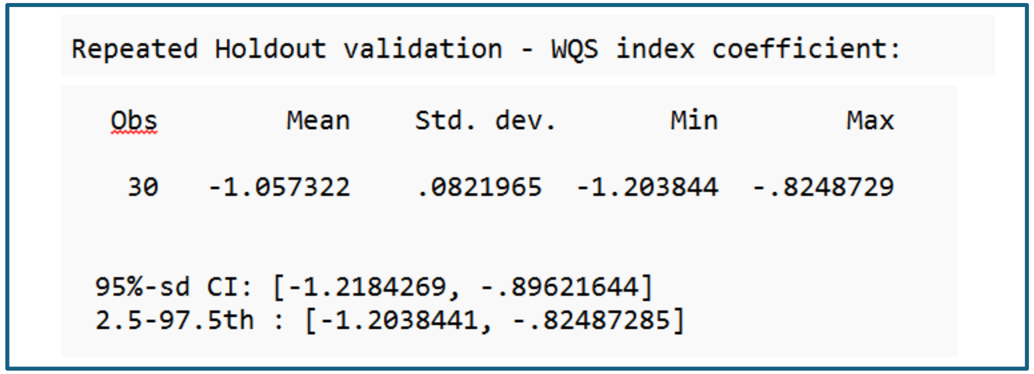
```
wqsreg hs_zbmi_who   , ///
       mixture(`mixture') cvar(`confounders') ///
       b1_neg(0) figureName("Pos")
```

```
Error: There are no positive b1
```

Exposome analysis example

Model 3. Negative direction- quartiles for exposure categorization, 100 bootstrap samples, 40/60 training-validation split, repeated holdout validation with 30 repetitions

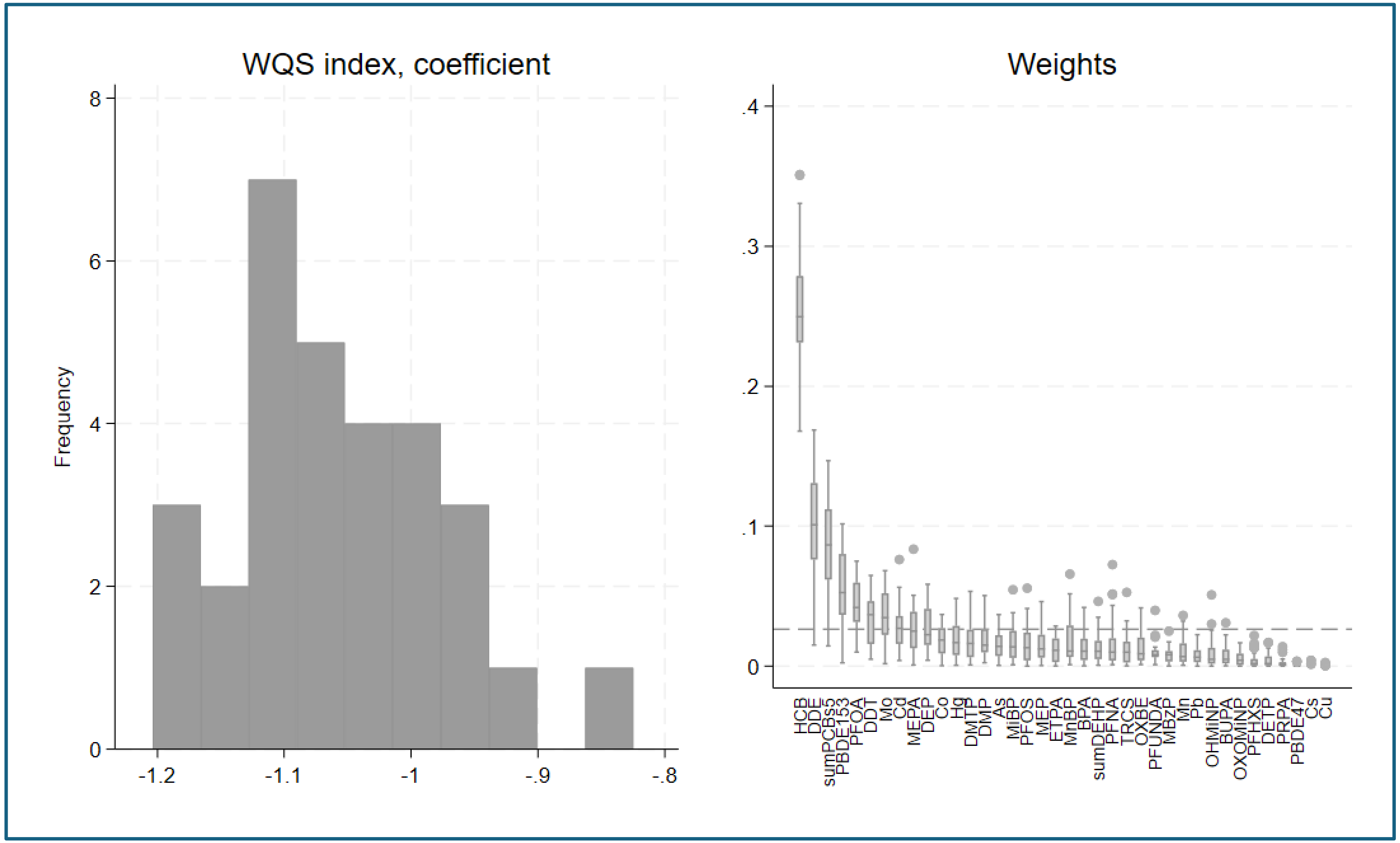
```
wqsreg hs_zbmi_who    , ///
       mixture(`mixture') cvar(`confounders') ///
       b1_neg(1) figureName("Neg_RH") rh_rep(30)
```



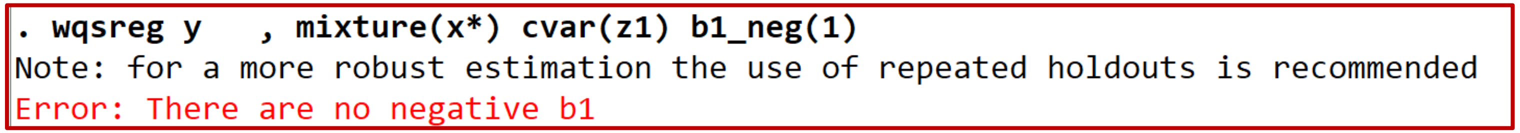
```
Repeated Holdout validation - WQS index coefficient:

  Obs        Mean    Std. dev.       Min        Max

   30   -1.057322    .0821965  -1.203844  -.8248729


 95%-sd CI: [-1.2184269, -.89621644]
 2.5-97.5th : [-1.2038441, -.82487285]
```

Exposome analysis example

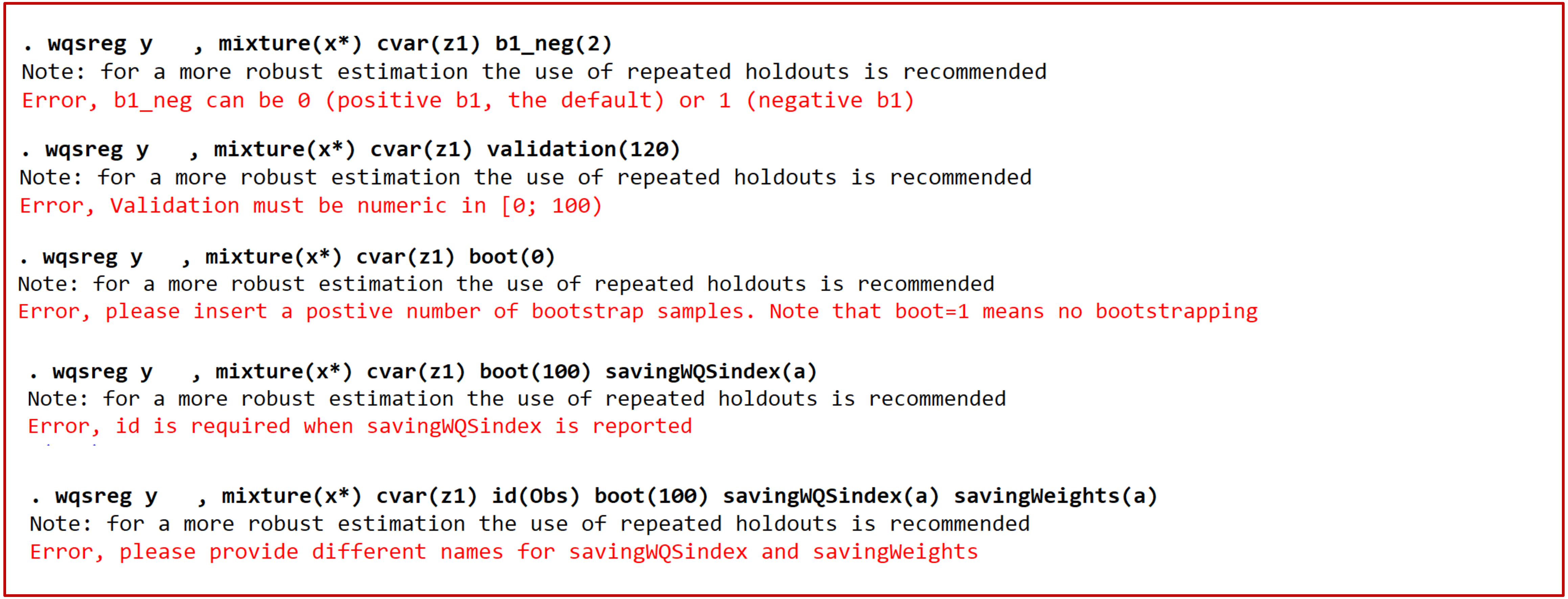
Model 3. Negative direction- quartiles for exposure categorization, 100 bootstrap samples, 40/60 training-validation split, repeated holdout validation with 30 repetitions



Possible errors generated by ***wqsreg*** include:



```
. wqsreg y   , mixture(x*) cvar(z1) b1_neg(1)
Note: for a more robust estimation the use of repeated holdouts is recommended
Error: There are no negative b1
```



```
. wqsreg y   , mixture(x*) cvar(z1) b1_neg(2)
Note: for a more robust estimation the use of repeated holdouts is recommended
Error, b1_neg can be 0 (positive b1, the default) or 1 (negative b1)

. wqsreg y   , mixture(x*) cvar(z1) validation(120)
Note: for a more robust estimation the use of repeated holdouts is recommended
Error, Validation must be numeric in [0; 100)

. wqsreg y   , mixture(x*) cvar(z1) boot(0)
Note: for a more robust estimation the use of repeated holdouts is recommended
Error, please insert a postive number of bootstrap samples. Note that boot=1 means no bootstrapping

. wqsreg y   , mixture(x*) cvar(z1) boot(100) savingWQSindex(a)
Note: for a more robust estimation the use of repeated holdouts is recommended
Error, id is required when savingWQSindex is reported

. wqsreg y   , mixture(x*) cvar(z1) id(Obs) boot(100) savingWQSindex(a) savingWeights(a)
Note: for a more robust estimation the use of repeated holdouts is recommended
Error, please provide different names for savingWQSindex and savingWeights
```

Discussion

- The importance of appropriately exploring complex multidimensional exposures, such as environmental mixtures, is increasingly recognized.
- **`wqsreg`** is the first command to apply WQS regression in Stata.

Selected references

- Bellavia A., Statistical Methods for Environmental Mixtures, Springer, 2025
- Carrico, C., Gennings, C., Wheeler, D. C., & Factor-Litvak, P. (2015). Characterization of weighted quantile sum regression for highly correlated data in a risk analysis setting. Journal of agricultural, biological, and environmental statistics, 20(1), 100-120.
- Czarnota, J., Gennings, C., & Wheeler, D. C. (2015). Assessment of weighted quantile sum regression for modeling chemical mixtures and cancer risk. Cancer informatics, 14, CIN-S17295.
- Renzetti, S., Curtin, P., Just, A. C., Bello, G., Gennings, C., Renzetti, M. S., & Rsolnp, I. (2021). Package 'gWQS'.

Selected references

- Vrijheid M, Slama R, Robinson O, Chatzi L, Coen M, Van Den Hazel P, et al. The Human Early-Life Exposome (HELIX): Project Rationale and Design. Environ Health Perspect. 2014;122:535–44.
- Maitre L, De Bont J, Casas M, Robinson O, Aasvang GM, Agier L, et al. Human Early Life Exposome (HELIX) study: a European population-based exposome cohort. BMJ Open. 2018;8:e021311.
- Tanner EM, Bornehag C-G, Gennings C. Repeated holdout validation for weighted quantile sum regression. MethodsX. 2019;6:2855–60.

ponzano.marta@gmail.com
m.ponzano@unilink.it