gWQS: An R Package for Linear and Generalized Weighted Quantile Sum (WQS) Regression

Stefano Renzetti
University of Brescia

Chris Gennings
Icahn School of Medicine at Mount Sinai

Paul C. Curtin
Icahn School of Medicine at Mount Sinai

Abstract

Weighted Quantile Sum (WQS) regression is a statistical model for multivariate regression in high-dimensional datasets commonly encountered in environmental exposures. The model constructs a weighted index estimating the mixture effect associated with all predictor variables on an outcome. The package **gWQS** extends WQS regression to applications with continuous, categorical and count outcomes. We provide two examples to illustrate the usage of the package.

Keywords: WQS, Weighted Quantile Sum, regression, mixture.

1. The gWQS package

The R package **gWQS** extends WQS regression to applications with continuous, categorical and count outcomes. In particular, this package uses the `solnp()` function from the **Rsolnp** package as optimization algorithm to estimates the weights. This function solves general nonlinear programming problems through the augmented Lagrangian multiplier method (Ye 1987; Ghalanos and Theussl 2015).
We list two examples to illustrate the usage of the package.

1.1. Example 1

The main function of the **gWQS** package is `gwqs()`, which allows the implementation of WQS regression for linear, logistic, multinomial, Poisson, quasi-Poisson and negative binomial regression. For Poisson and negative binomial regression a zero inflated option is also implemented. We created the `wqs_data` dataset (available once the package is installed and loaded) to demonstrate the use of this function. These data reflect 34 exposure concentrations simulated from a distribution of PCB exposures measured in subjects participating in the NHANES study (2001-2002). Additionally, an end-point measure, simulated from a distribution of leukocyte telomere length (LTL), a biomarker of chronic disease, is provided as well (variable name: `y`), along with simulated dichotomous (variable name: `y_bin`), multinomial (variable name: `y_multinom`) and count (variable name: `y_count`) outcome variables and

covariates, e.g. `sex`. This dataset can thus be used to test the **gWQS** package by analyzing the mixture effect of the 34 simulated PCBs on the outcomes, with adjustments for covariates. The following script calls a WQS model for a continuous outcome using the function `gwqs()`; we also show the script to reproduce the plots and tables that are automatically generated when setting the options `plots = TRUE, tables = TRUE`:

```
R> # we save the names of the mixture variables in the variable "toxic_chems"
R> toxic_chems <- names(wqs_data)[1:34]
R> # we run the model and save the results in the variable "results"
R> results <- gwqs(y ~ wqs, mix_name = toxic_chems,
+                  data = wqs_data, q = 4, validation = 0.6, b = 2,
+                  b1_pos = TRUE, b1_constr = FALSE, family = "gaussian",
+                  seed = 2016, plots = TRUE, tables = TRUE)
R> #
R> # bar plot
R> w_ord <- order(results$final_weights$mean_weight)
R> mean_weight <- results$final_weights$mean_weight[w_ord]
R> mix_name <- factor(results$final_weights$mix_name[w_ord],
+                     levels = results$final_weights$mix_name[w_ord])
R> data_plot <- data.frame(mean_weight, mix_name)
R> ggplot(data_plot, aes(x = mix_name, y = mean_weight, fill = mix_name)) +
+    geom_bar(stat = "identity", color = "black") + theme_bw() +
+    theme(axis.ticks = element_blank(),
+          axis.title = element_blank(),
+          axis.text.x = element_text(color='black'),
+          legend.position = "none") + coord_flip()
R> #
R> # scatter plot y vs wqs
R> ggplot(results$y_wqs_df, aes(wqs, y_adj)) + geom_point() +
+    stat_smooth(method = "loess", se = FALSE, size = 1.5) + theme_bw()
R> #
R> # scatter plot residuals vs fitted values
R> fit_df <- broom::augment(results$fit)
R> ggplot(fit_df, aes(x = .fitted, y = .resid)) + geom_point() +
+    theme_bw() + xlab("Fitted values") + ylab("Residuals")
R>
```

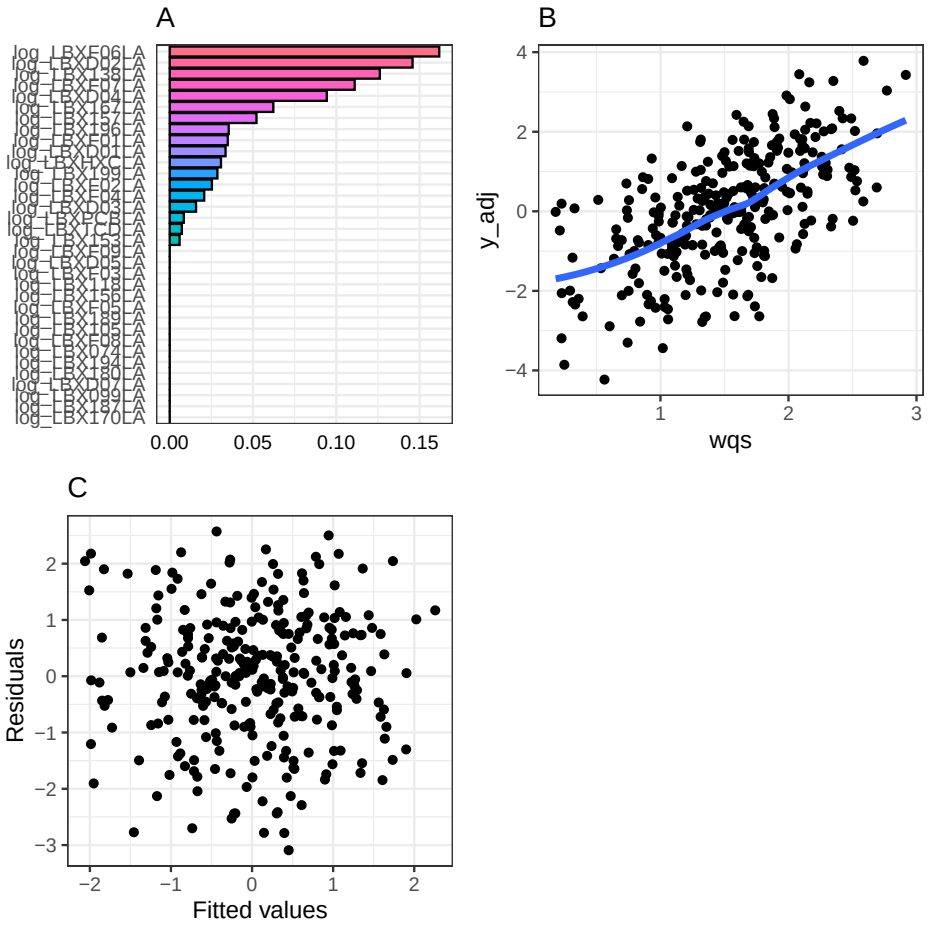

Figure 1: Plots displayed for linear outcomes when `plots = TRUE`

This WQS model tests the relationship between our dependent variable, `y`, and a WQS index estimated from ranking exposure concentrations in quartiles (`q = 4`); the `wqs` term must be included in the `formula`). It also divided the data for training and validation, with 40% of the dataset for training and 60% for validation (`validation = 0.6`), and assigned 2 bootstrap samples (`b = 2`) for parameter estimation (in practical applications we suggest at least 100 bootstrap samples to be used). Because WQS provides a unidirectional evaluation of mixture effects, we first examined weights derived from bootstrap models where β_1 was positive (`b1_pos = TRUE`); we could test for negative associations by setting that parameter to be false (`b1_pos = FALSE`). We can also choose to constrain the β_1 to be positive (`b1_pos = TRUE` and `b1_constr = TRUE`) or negative (`b1_pos = FALSE` and `b1_constr = TRUE`) when we estimate the weights; in the case of example 1 we are not applying a constraint to β_1. We linked our model to a gaussian distribution to test for relationships between the continuous outcome and exposures (`family = "gaussian"`), and fixed the seed to 2016 for reproducible results (`seed = 2016`). We plotted a summary model with loess fit, and a summary of each variables' relative weight, and the residuals vs fitted values plot (`plots = TRUE`). The command `tables = TRUE` automatically generates in the Viewer pane the tables of the weight ranked list and the model summary.

Figure 1 A is a barplot showing the weights assigned to each variable ordered from the highest weight to the lowest. These results indicate that the variables `log_LBXF06LA` and `log_LBXD02LA` are the largest contributors to this mixture effect, with the first 6 chemicals explaining more than the 70% of the total weights.
In plot B of figure 1 we have a representation of the wqs index vs the outcome (adjusted for the model residual when covariates are included in the model) that shows the direction and the shape of the association between the exposure and the outcome. For example, in this case we can observe a linear and positive relationship between the mixture and the `y` variable.
In plot C a diagnostic graph of the residuals vs the fitted values is shown to check if they are randomly spread around zero or if there is a trend.
To test the statistical significance of the association between the variables in the model, the following code has to be run:

```
R> summary(results$fit)

Call:
glm(formula = formula, family = family, data = bdtf)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.09131  -0.71326   0.06459   0.78517   2.57249

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -2.3453     0.1909  -12.29   <2e-16 ***
wqs           1.5785     0.1187   13.30   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 1.304685)

    Null deviance: 619.48  on 299  degrees of freedom
Residual deviance: 388.80  on 298  degrees of freedom
AIC: 935.14

Number of Fisher Scoring iterations: 2
```

This result tells us that the association is positive and statistically significant (`p=0.025`).
To have the exact values of the estimated weights we can apply the command `results$final_weights`. The following code shows the first six highest weights; the full list of weights can be called by omitting the head function:

```
R> head(results$final_weights)

                mix_name mean_weight
log_LBXF06LA log_LBXF06LA  0.16181177
log_LBXD02LA log_LBXD02LA  0.14579100
```

```
log_LBX138LA log_LBX138LA  0.12612569
log_LBXF07LA log_LBXF07LA  0.11104863
log_LBXD04LA log_LBXD04LA  0.09430793
log_LBX167LA log_LBX167LA  0.06219165
```

These tables are also shown in the Viewer window when we set `tables = TRUE`.
The `gwqs()` function gives back other outputs like the vector of the values that indicate whether the solver has converged (0) or not (1 or 2) (`results$conv`), the matrix with all the estimated weights and the associated β_1, standard errors, statistics and p-values for each bootstrap sample (`results$bres`), the vector of the estimated `wqs` index (`results$wqs`), the vector containing the cutoffs used to determine the quantiles (`results$q_i`), the list of vectors containing the rows of the subjects included in each bootstrap dataset (`results$bindex`), the rows identifying the subjects used to estimate the weights in each bootstrap (`results$tindex`) and the rows identifying the subjects used to estimate the parameters of the final model (`results$vindex`).

1.2. Example 2

In the following code we run a logistic regression (`family = binomial`) to test the association between the exposure to the mixture and the outcome `y_bin` and we also add the covariate `sex`. Since the mixture concentrations in this example are already standardized we can also run a model without categorizing for quantiles (`q = NULL`) after checking that there were no skewed distributions. Furthermore we examined the ability of our model to predict the outcome on a third part of the dataset (`pred = 0.3`). As we see from the script below `validation = 0.4`; that means that the 30% of the data are used as test dataset, 40% for validation and the last 30% for prediction; the script to generate the additional plot is reported:

```
R> # we run the logistic model and save the results in the variable
R> # "results2"
R> results2 <- gwqs(y_bin ~ wqs + sex, mix_name = toxic_chems,
+                   data = wqs_data, q = NULL, validation = 0.4, b = 2,
+                   b1_pos = TRUE, b1_constr = FALSE, family = binomial,
+                   seed = 2018, plots = TRUE, tables = FALSE, pred = 0.3)
R> #
R> # plot ROC curve
R> gg_roc <- ggplot(results2$df_pred, aes(d=y, m=p_y)) + geom_roc(n.cuts = 0) +
+    style_roc(xlab = "1 - Specificity", ylab = "Sensitivity")
R> auc_est <- plotROC::calc_auc(gg_roc)
R> gg_roc + annotate("text", x=0.75, y=0.25,
+                     label=paste0("AUC = ", round(auc_est[, "AUC"], 3)))
R>
```

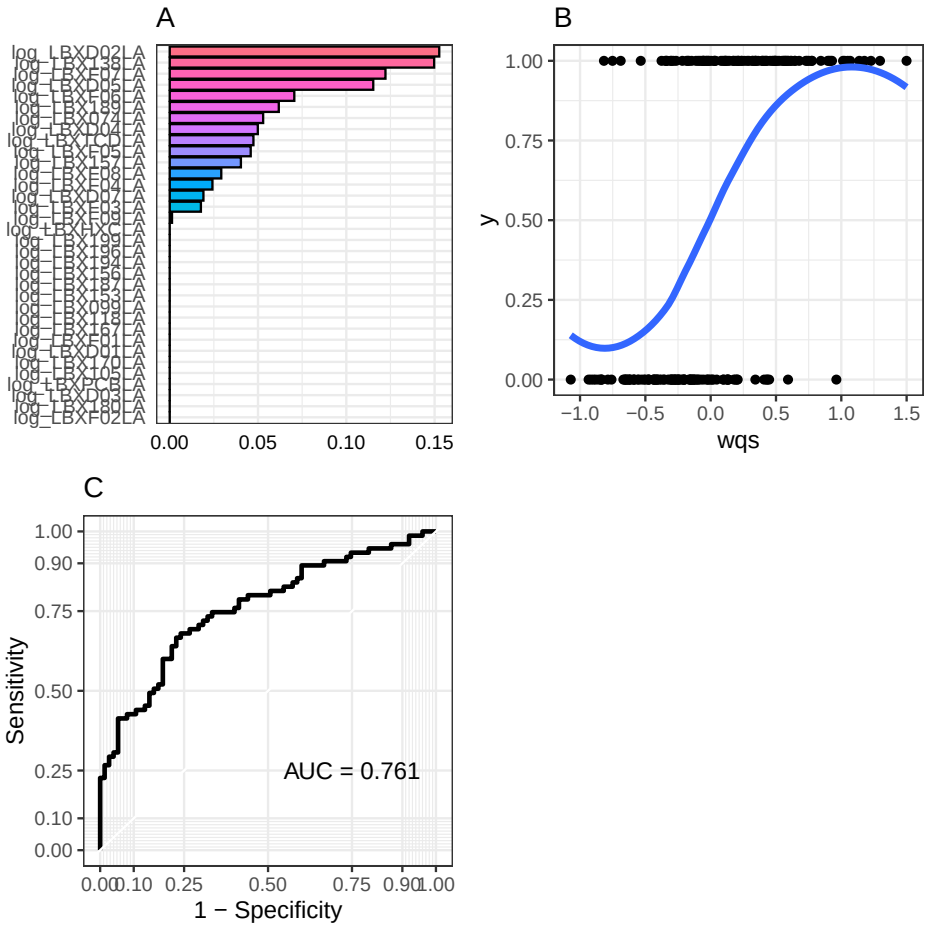

Figure 2: Plots displayed for binary outcomes when `plots = TRUE` and `pred > 0`

From figure 2 we see the per-variable calculated weights, ordered by relative magnitude. Plot B shows a positive relationship between the mixture and the outcome and as we can see from the following code it is statistically significant (p<0.001):

```
R> summary(results2$fit)

Call:
glm(formula = formula, family = family, data = bdtf)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5127  -0.8162   0.1340   0.7440   2.3262

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.1152     0.2540  -0.454    0.650
wqs           3.3557     0.4950   6.779 1.21e-11 ***
sex           0.2871     0.3513   0.817    0.414
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 277.24  on 199  degrees of freedom
Residual deviance: 198.48  on 197  degrees of freedom
AIC: 204.48

Number of Fisher Scoring iterations: 5
```

In plot C we show the Receiver Operating Characteristic (ROC) curve related to the predictive model: we can see that the cutoff that is closer to the left-hand border and the top border has around 70% sensitivity (true positive) and 30% specificity (false positive).
In this case two more parameters are returned by the `gwqs()` function: `df_pred`, which is a `data.frame` including a first column the actual value of the dependent variable and as a second column the predicted values; and `pindex`, the dataset rows identifying the observations used for prediction.
The `gwqs` function implements the `predict` function to run the predictive model. The following code shows how to reproduce the prediction:

```
R> # create a dataset exluding the data where we want to apply the prediction
R> # and define the group variable to identify the test and validation dataset
R> wqs_data$group <- 0
R> wqs_data$group[results2$vindex] <- 1
R> wqs_data_train <- wqs_data[-results2$pindex,]
R> # fit the model on the training dataset
R> results2_pred <- gwqs(y_bin ~ wqs + sex, mix_name = toxic_chems,
+                   data = wqs_data_train, q = NULL, validation = NULL,
+                   b = 2, valid_var = "group", b1_pos = TRUE,
+                   b1_constr = FALSE, family = binomial, seed = 2018)
R> # creat the dataset on which we apply the prediction
R> wqs_data_pred <- wqs_data[results2$pindex,]
R> # create wqs variable for the prediction dataset
R> mix_matrx <- as.matrix(wqs_data_pred[, rownames(results2$final_weights)])
R> wqs_data_pred$wqs <- as.numeric(mix_matrx%*%results2$final_weights$mean_weight)
R> # apply the predict() function
R> pred <- predict(results2$fit, newdata = wqs_data_pred, type = "response")
R> df_pred <- data.frame(y = wqs_data_pred$y_bin, p_y = pred)
R> # plot the roc curve
R> gg_roc <- ggplot(df_pred, aes(d=y, m=p_y)) + geom_roc(n.cuts = 0) +
+    style_roc(xlab = "1 - Specificity", ylab = "Sensitivity")
R> auc_est <- plotROC::calc_auc(gg_roc)
R> gg_roc + annotate("text", x=0.75, y=0.25,
+                     label=paste0("AUC = ", round(auc_est[, "AUC"], 3)))
```

2. Discussion

WQS regression is a new method that allows the investigation of the associations between mixtures of predictors and continuous, count, or categorical data. This approach is particularly robust against outliers and extreme values because of the ranking procedure used, and is additionally robust against collinearity through the constraints imposed during weight estimation and application of an ensemble estimation procedure. As well, the capacity for covariate adjustment and the simplicity of model interpretation are among the greatest strengths of this approach, and underlie its applicability to health-related research. Through the weighted index we are able to identify the combined impact of multiple predictors on a given outcome, while in the estimation of the weights we may simultaneously assess the discrete effects of contributing variables, with coadjustment for the overall mixture and relevant covariates.
The package **gWQS** provides a robust, generalizable implementation of this methodology in R extending the application of the model to continuous, binary, multinomial and count data applying the corresponding log-likelihood for each type of regression (zero inflated likelihoods are also available). The new version of the package allows also to run a stratified analysis for a categorial variable.
Future versions of the package will provide the ability to fit additional generalised linear models.

Acknowledgments

This package was developed at the CHEAR Data Center (Dept. of Environmental Medicine and Public Health, Icahn School of Medicine at Mount Sinai) with funding and support from NIEHS (U2C ES026555-01) with additional support from the Empire State Development Corporation.

References

Ghalanos A, Theussl S (2015). ***Rsolnp**: General Non-linear Optimization Using Augmented Lagrange Multiplier Method*. R package version 1.16, URL https://cran.r-project.org/package=Rsolnp.

Ye Y (1987). "Interior Algorithms for Linear, Quadratic, and Linearly Constrained Non-Linear Programming."

Affiliation:

Stefano Renzetti
Department of Occupational Health
University of Brescia
Piazzale Spedali Civili, 1, 25123 Brescia BS Italy
E-mail: stefano.renzetti@unibs.it

Chris Gennings
Department of Environmental Medicine and Public Health
Faculty in Biostatistics
Icahn School of Medicine at Mount Sinai
1 Gustave L. Levy Place New York, NY 10029
E-mail: chris.gennings@mssm.edu

Paul C. Curtin
Department of Environmental Medicine and Public Health
Faculty in Biostatistics
Icahn School of Medicine at Mount Sinai
1 Gustave L. Levy Place New York, NY 10029
E-mail: paul.curtin@mssm.edu