Quantile and distribution regression in Stata: algorithms, pointwise and functional inference

Victor Chernozhukov — MIT
Iván Fernández-Val — Boston University
Blaise Melly — Bern University

November 19, 2020
Swiss Stata Conference

Estimation of the conditional distribution

- Given an outcome Y and a vector of regressors X, OLS estimates the conditional expected value assuming linearity:

$$E[Y|X] = X'\beta$$

- We are often interested in the whole distribution and not only in the expected value of Y, e.g. wage inequality, long-term unemployment, low birthweight, financial risk.
- Quantile regression (QR) estimates the conditional quantile function assuming that it is linear in X:

$$Q_Y(\tau|X) = X'\beta(\tau)$$

- Distribution regression (DR) estimates the conditional distribution function assuming that the latent index is linear:

$$F_Y(y|X) = \Lambda(X'\beta(y))$$

- In this presentation I focus on **quantile regression**.

Contributions

- When we started this project, our objective was to implement existing statistical tools for QR and DR:
  - Analytical estimates of the variance-covariance matrix of the coefficients for several regressions allowing for weights, clustering and stratification.
  - Uniform confidence bands and tests of functional null hypotheses based on a wide range of resampling methids.
  - Monotone estimates of the QF and DF.
  - A convenient command to plot the results.
- During the process, we found that it was possible to design faster algorithms:
  - Exact algorithms for the QR process and bootstrapping the QR coefficients.
  - Approximate algorithms for the QR and DR processes.

Related papers

- In "Fast algorithms for the quantile regression process" (forthcoming in Empirical Economics), we describe the new algorithms for QR, prove the validity of the one-step approach, and show with simulated and real data data that our new algorithms provide very large improvements in computation time without significant (if any) cost in the quality of the estimates.
- In "Quantile and distribution regression in Stata: algorithms, pointwise and functional inference", we discuss the implementation in Stata and describe the commands that we have written for QR and DR.

QR estimator

- The QR estimator suggested by Koenker and Bassett (1978) solves the following problem:

$$\widehat{\beta}(\tau) \in \arg\min_{b \in \mathcal{R}^k} \sum_{i=1}^{n} \rho_\tau \left(y_i - x_i' b \right).$$

- The (approximate) derivative of the objective function is

$$\sum_{i=1}^{n} \left(\tau - 1 \left(y_i \leq x_i' \widehat{\beta}(\tau) \right) \right) x_i.$$

  $\Longrightarrow$ only the signs of the residuals matter.
- This is a convex linear program that can be solved relatively efficiently for small data sets with some modifications of the simplex algorithm. Implemented in **qreg**.
- Portnoy and Koenker (1997): interior point and preprocessing algorithms. Implemented in **qrprocess**.

Computation time is still an issue

- These two algorithms decrease significantly the computation time required to estimate one QR.
- Despite these improvements the computation time is still non-negligible for two reasons:
  1. Researchers often estimate many quantile regressions. Some estimation methods or inference procedures require the preliminary estimation of the whole quantile regression process: Koenker and Portnoy (1987), Koenker and Xiao (2003), Machado and Mata (2005), Chernozhukov, Fernandez-Val and Melly (2013).
  2. The bootstrap is often used to perform (functional) inference.

Preprocessing for the QR process

- Suppose that we have already computed $\hat{\beta}(\tau)$ and we want to compute $\hat{\beta}(\tau + \varepsilon)$.
- Only the sign of the residuals matters. Idea: use the residuals at quantile τ to guess the sign of the residuals at quantile $\tau + \varepsilon$.

1. Keep only the $M \equiv m \cdot n^{1/2}$ observations that have the lowest absolute relative residuals at quantile τ. The observations below and above are replaced by two pseudo-observations.
2. Solve the QR problem with these $M + 2$ observations.
3. Check that the sign of the residuals corresponds to the guess made in part (1). If this is not the case, they increase m and go back to step (1).

Preprocessing for the bootstrap

- We have already computed $\hat{\beta}(\tau)$ in the whole sample. We want to compute $\hat{\beta}^*(\tau)$ in the bootstrap sample.
- We use $\hat{\beta}(\tau)$ as a preliminary estimator.
- Everything else is the same.
- Bootstrapping QR is now *faster* than bootstrapping OLS in large enough samples!

One-step estimator

Idea:

1. Start from one QR computed using a standard algorithm.
2. Obtain sequentially the remaining regression coefficients using first-order Taylor approximations.

- The resulting one-step QR estimator is

$$\hat{\beta}(\tau+\varepsilon) = \hat{\beta}(\tau) - \hat{J}(\tau)^{-1} \frac{1}{n} \sum_{i=1}^{n} \left(\tau + \varepsilon - 1\left(y_i \leq x_i \hat{\beta}(\tau)\right)\right)$$

  where $\hat{J}(\tau)$ is an estimate of $-E\left[f_Y\left(x_i'\beta(\tau)\,|x_i\right) x_i x_i'\right]$.
- This estimator is not numerically identical to the traditional QR estimator but we can show that it is asymptotically equivalent to the QR estimator.

Inference

- The limiting distribution of a finite number of quantile regressions is Gaussian. We provide analytical estimation of their variance allowing for weighting, clustering and stratification. Straightforward to test pointwise hypotheses.
- Simulations show that resampling methods works better. We have implemented the following bootstrap methods: empirical, wild, Bayesian, weighted, subsampling.
- Many policy questions of interest involve functional hypotheses: no effect, constant effect, stochastic dominance. We provide uniform confidence bands and tests.
- Score resampling: instead of recomputing the whole QR process for each bootstrap sample, we can resample the first order asymptotic approximation of the estimate. Much faster.

Simulations: 99 QR, 20 regressors

	Number of observations				
Algorithm	500	1000	5000	10k	50k
Estimation of the whole QR process (99 quantile regressions)					
A. Computing time in seconds					
qreg	5.93	9.01	48.86	119	1,084
qrprocess, m(proqreg)	0.64	0.85	2.93	6.96	57.70
qrprocess, m(profn)	2.02	1.90	3.95	6.55	30.67
qrprocess, m(onestep)	0.40	0.45	1.00	1.77	8.29
B. Performance of the one-step estimator					
proportion not converged	0.12	0.07	0.00	0.00	0.00
relative MSE	2.64	2.15	1.10	1.02	0.99
relative MAE	1.02	1.01	1.01	1.00	0.99

Simulations: pointwise inference

	Number of observations					
Algorithm	500	1000	5000	10k	50k	100k
Bootstrap of a single quantile regression						
Computing time in seconds for $\tau = 0.5$ and $k = 20$						
bsqreg	2.63	3.14	8.19	15.06	99.89	163.3
qrprocess, vce(bootstrap)	0.42	0.63	1.89	3.50	13.55	18.52
qrprocess, vce(multiplier)	0.08	0.10	0.25	0.60	1.36	1.90

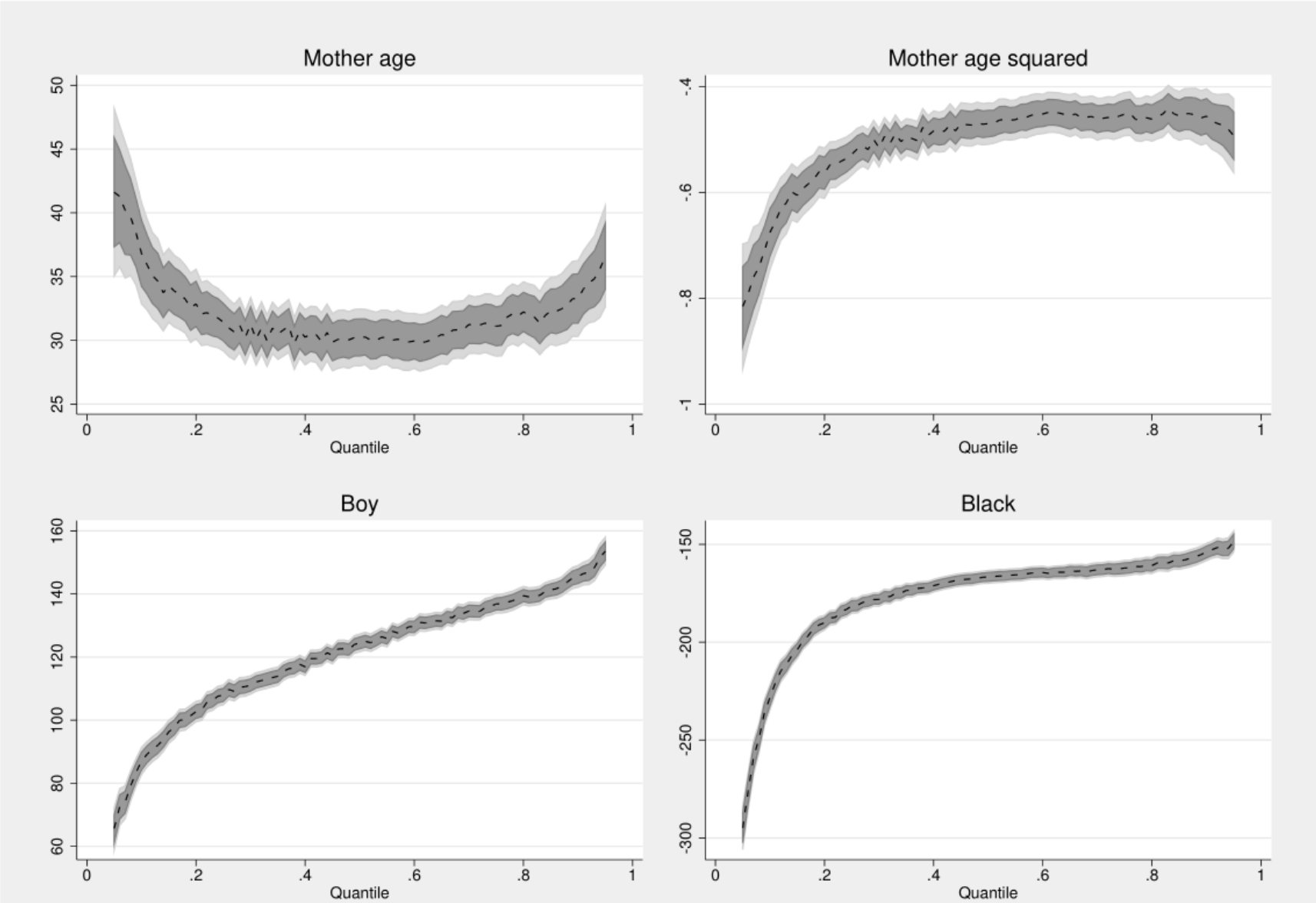
Simulations: uniform inference

	# of observations			
Bootstrap method	100	500	1000	5000
Panel 1: Quantile regression, empirical size				
empirical	0.02	0.03	0.04	0.05
empirical, 1-step	0.00	0.01	0.03	0.05
multiplier	0.03	0.05	0.06	0.07
multiplier, 1-step	0.07	0.06	0.07	0.07
Panel 2: Quantile regression, empirical power				
empirical	0.00	0.07	0.22	1.00
empirical, 1-step	0.00	0.02	0.14	0.99
multiplier	0.01	0.11	0.27	1.00
multiplier, 1-step	0.04	0.14	0.28	1.00

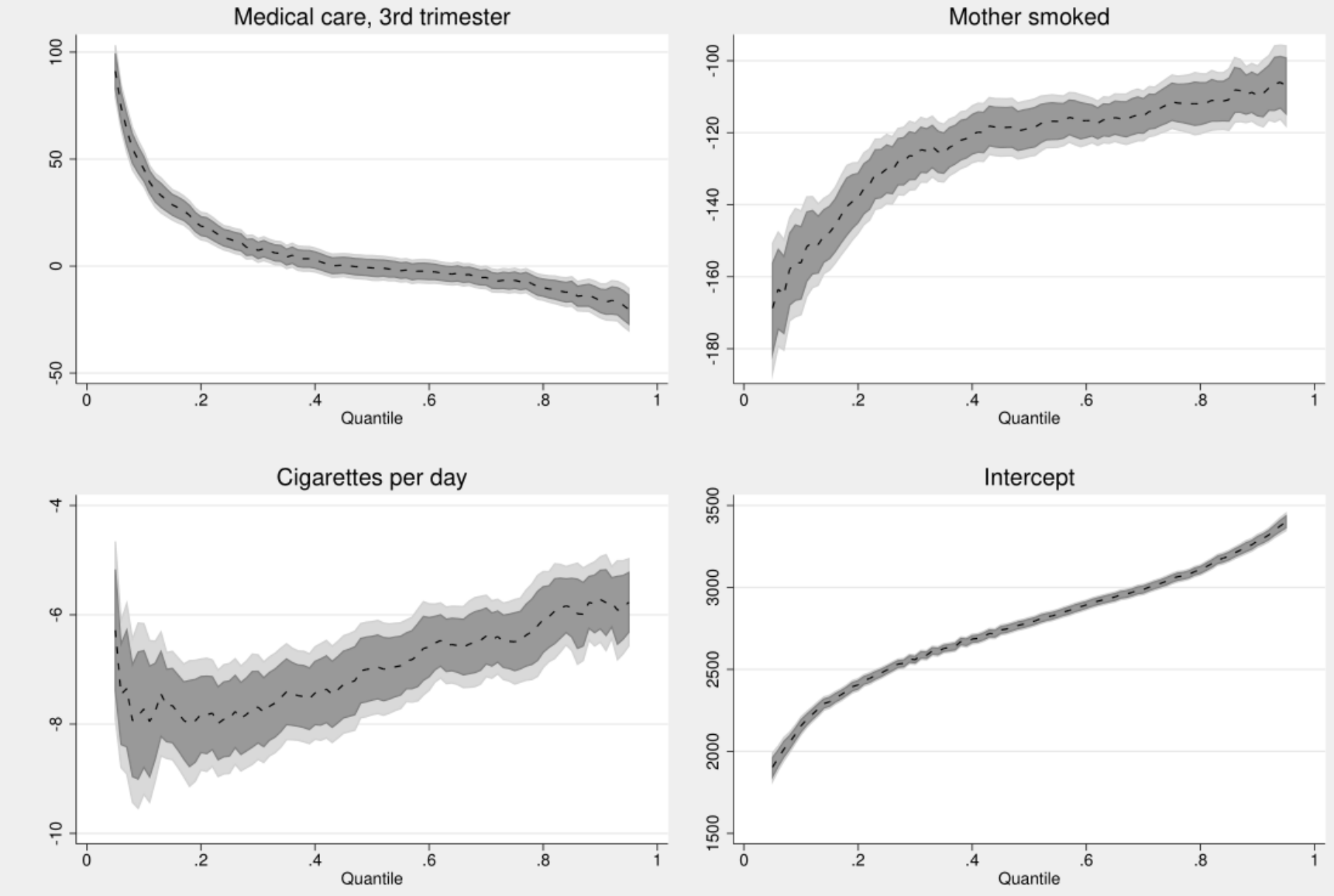
Determinants of infant birth weight

- Update of the results obtained by Koenker and Hallock (2001).
- 2017 Natality Data provided by the National Center for Health Statistics.
- 2,194,021 observations, 14 regressors, 91 quantile regressions, 100 bootstrap replications.
- Estimation on my laptop.
- With built-in Stata commands it should take about 75 days.
- One-step estimator with multiplier bootstrap takes less than 30 minutes.

Some coefficients



Some other coefficients



Stata commands

- `qrprocess` *depvar indepvars* [*if*] [*in*] [*weight*], [*options*]
  - `quantiles`(*numlist*): quantile(s) τ.
  - `method`(*string*): algorithm.
  - `vce`(*string*): method to compute the VCE.
  - `functional`: activates the tools for functional inference.
- `drprocess` *depvar indepvars* [*if*] [*in*] [*weight*], [*options*]
  - `thresholds`(*numlist*): threshold(s) y.
  - `method`(*string*): algorithm.
  - `vce`(*string*): method to compute the VCE.
  - `functional`: activates the tools for functional inference.
- `plotprocess` [*namelist*], [*intervaltype*]
  - *namelist*: parameters are to be plotted.