Preferences and Performance in Simultaneous First-Price Auctions: A Structural Analysis

By M. Gentry, T. Komarova, and P. Schiraldi

Data Availability Statement

All raw input data is in the folder *data*. These data are collected from the following sources.

1. Data on bids, project lettings, and contract characteristics were collected from letting reports issued by the Michican Department of Transportation (MDOT). Letting reports are publicly available via MDOT's contracting website: www.bidx.gov/mi/main. We accessed these reports in February 2015 and compiled them into the datasets *bids.xls*, *contracts.xls*, and *subcontracts.xls* in the folder *data*, which provide bid-level, contract-level and subcontract-level data respectively. We also provide a crosswalk linking bidder names to unique bidding identifiers in the dataset *key_biddername_id.dta*.

2. Data on firm and locations were collected from several sources in early 2015. Our primary source was company websites. Some companies did not have a website, and even those which did have websites did not necessarily have information about all their plant locations. In these situations, we used secondary sources such as http://start.cortera.com/company, http://www.yellowpages.com/mi/ and http://www.whitepages.com to collect the necessary data. Our final company location data are recorded in the datasets *company_location.dta* and *company_location2.dta*. We also provide a crosswalk linking firm and plant locations to firm identifiers in the dataset *bidders_ids_locs.dta*.

3. For a small number of bidder-project pairs, we were unable to compute distance directly due to being unable to identify either bidder or project locations. For these pairs, we filled in distances using an alternative distance measure graciously supplied to us by Professor Paulo Somaini. We have included this supplemental distance data in the replication package as the Stata file *bidder_dist_extra.dta*.

Both the MDOT letting data and our firm location data were collected directly from publicly available sources. We have also made the supplemental distance data (3) public in the replication package as described above.

System Requirements

Our code is written in a combination of Stata, Matlab, Python, and Julia. To run the code from start to finish, including counterfactual bootstrap replications, we estimate that two CPU weeks would be required on a modern desktop. We provide version information and package requirements for each program below.

- *Stata:* We used Stata to generate summary tables and figures, to run regressions, and to format data for use in our structural estimation routines. All code was run in Stata 15. Our code uses the external ado file outreg2, which can be installed by typing "ssc install outreg2" at the Stata command line.

- *Matlab:* We use Matlab to run our main structural estimation algorithm and to run some parts of our Monte Carlo simulations. Our code was last run in Matlab 2021a. Our code depends on the qsimvnv module, written and copyrighted by Alan Gentz, for simulating high-dimensional normal probabilities. This module was distributed under an open-access license which allows redistribution so long as original copyright information is retained within the code. We have therefore included the qsimvnv module with our replication code. We also use functions in Matlab's Optimization Toolbox and Statistics Toolbox.

- *Python:* We use Python to create backlog variables, solve for counterfactual VCG allocations, and compute counterfactual revenue and costs. Our code was last run in Python 3.7.3. Our code uses Gurobi 9.1, which can be installed under an academic license from www.gurobi.com, to solve large-scale integer programming problems. Our code also depends on several user-installed Python packages, all of which may be installed using the Python package manager. We list each package with the corresponding version number under which the code was last run:

    - Pandas 0.24.2
    - Numpy 1.16.2
    - Scipy 1.2.1
    - Networkx 2.2
    - Gurobipy (bundled with Gurobi 9.1)

- *Julia:* We use Julia for some parts of our Monte Carlo simulations reported in Appendix G. Our code was last run using Julia 1.5.3. In addition to built-in Julia modules, our code depends on the following user-installed Julia packages, all of which may be installed with the Julia package manager. We list each package with the corresponding version number under which the code was last run:

    - DataFrames 0.22.5
    - DataFramesMeta 0.6.0
    - Distributions 0.24.6
    - NumericalIntegration 0.2.0
    - StatsBase 0.33.2
    - JLD 0.11.0
    - ForwardDiff 0.10.14
    - Interpolations 0.13.1
    - Optim 1.2.0
    - DiffResults 1.0.3
    - Formatting 0.4.2
    - LaTeXStrings 1.2.1
    - LaTeXTabulars 0.1.1

Replication Instructions

Our input data are in the folder *data*. The Excel files "bids.xls," "contracts.xls," and "subcontracts.xls" contain bid-level, contract-level, and subcontract-level data collected from MDOT letting reports (as described in our data accessibility statements above).

Data cleaning, merging, and reduced form analysis are in the folder *reducedform*. To reproduce our analysis from the raw files in *data*, the following scripts should be run in sequence. All scripts should be run from the folder *reducedform*.

- **0_create_backlog.py**
  Create backlog from raw data and save merged bids_with_backlogs.xls dataset.

- **1_clean_merge_data.do**
  Generate relevant variables and clean original data. This script also reproduces the results in Tables 1-3 and Figure 1.

- **2_export_structural_data.do**
  Export data used for the structural estimation.

All files for the main structural analysis are in the folder *structural*. To reproduce the structural component of the analysis from the cleaned data produced by the scripts in *stata*, one should run the following scripts in sequence from the folder *structural*:

- **a_bid_distribution.m**
  Estimate the bid distribution. This reproduces the results in Table 4 and Figure 2.

- **b_prep_data_run.m**
  Reshape the data, compute the variables in K, the instruments, compute the winning probability, the Υ and Ψ in equations 4 and 5, match the data and compute the kernel weights.

- **c_parameters.do**
  Estimate the structural parameters in Table 5.

- **d_compute_gg0.m**
  Compute the derivative of moment condition WRT to bid distribution parameters to use in e_estimate.

- **e_estimate.m**
  Correct the standard error in Table 5 to account for the first stage estimates. This reproduces the results in Table 5 and Table 6.

- **f_counterf_sample.m**
  Compute the counterfactural samples.

- **g_format_vcg_inputs.py**
  Translate Matlab inputs from f_counterf_sample.m into format expected by Python VCG simulation solver.

- **h_run_vcg_sims.py**
  Run VCG simulations in Python, saving results of each bootstrap replication as CSV files.

- **I_tabulate_vcg_results.py**
  Compile output from bootstrapped CSV replications into table form. This reproduces the results reported in Table 7.

Finally, files for the Monte Carlo and numerical simulations in Appendix G are in the folder *appendixg*. The executable scripts are labelled "main_x" and should be run from the folder *appendixg* in the following sequence:

- **Main_appendixG1_0_solve_eqm.m**
  Solve for an approximate equilibrium for a single DGP on a discrete bidding grid. This code must be run once for each of the 12 data generating process configurations used in the Monte Carlo exercise. The relevant configurations are documented in the script.

- **Main_appendixG1_1_run_mc_sims.m**
  Run 1000 Monte Carlo simulations underlying Table 8 in Appendix G.1.

- **Main_appendixG1_2_tabulate_results.m**
  Consolidate saved Monte Carlo results into Latex output for Table 8 in Appendix G.1.

- **Main_appendixG2_0_run_mc_sims.jl**
  Julia script to run 1000 Monte Carlo simulations of one-step and two-step GMM estimators in the data generating process for Appendix G.2. This code must be run once for each of the following values of the variable nobs on line 16: nobs=250, 500, 1000.

- **Main_appendixG2_1_tabulate_results.jl**
  Consolidate saved Monte Carlo results into Latex output for Table 9 in Appendix G.2. This code must be run once for each of the following values of the variable *nobs* defined on line 9: *nobs*=250, 500, 1000.

- **Main_appendixG3_0_solve_eqm.m**
  Solve for an approximation equilibrium on a discrete bid grid for each example data generating process considered in Table 10 of Appendix G.3. This code must be run once for each of the 12 example specification scripts (e.g. "ex1a_asy_noK.m" the 11 subsequent scripts beginning "ex") prior to running the next main script.

- **Main_appendixG3_1_tabulate_results.m**
  Create Latex output for Table 10 in Appendix G.3 based on saved results from the solved equilibria for each of the 12 DGPs solved in the preceding step.

Data Citations

Bid Express. *Michigan Department of Transportation.* https://www.bidx.com/mi/main. Accessed February 2015.

Cortera. *Company Search.* https://start.cortera.com/company/ Accessed February 2015.

Somaini, Paulo (2020): "Identification in Auction Models with Interdependent Costs." *Journal of Political Economy* 128 (10): 3820—3871.

YellowPages.com. *Online Yellow Pages Directory.* https://www.yellowpages.com/ Accessed February 2015.

WhitePages.com. *Online White Pages Directory.* http://www.whitepages.com Accessed February 2015.