Code documentation:

This file describes the code used in Hortacsu, Luco, Puller, and Zhu (2019). There are two main directories with the code used in the paper. The directories differ on whether the code is written in Matlab or Stata. Importantly, we obtained our data through a one-time arrangement with the Public Utility Commission of Texas. This arrangement implies that the data on bids and marginal costs cannot be made public (see Hortacsu and Puller 2008). We obtained the data identifying background information of the different firms from public sources described in the main text. For this reason, these data are identified in the estimation code (these data are hard coded).

1) Matlab: Matlab files are used for two purposes, estimation and computation of counterfactuals. We describe the associated files below.

1.1) Estimation:

1.1.1) The file main.m is used to estimate all specifications reported in the paper. This file is called through a job scheduler and uses a single vector as input (labeled rr, more on this below). This means that the file can be called from the Matlab prompt using main(rr) where rr is a vector.

The first lines of the code load the data (proprietary, see above). The data consists on bids (called full_pqs.csv), marginal costs (mcosts.csv), total demand (totdemand_mcprice.csv) and a measure of firm size (firm_size_new.csv).

After loading the data, the file performs some ‘house-cleaning’ identifying the auctions that are studied. Once these auctions are identified, the code calls the function see_contract.m that computes the contract positions of the firms following the method described in the paper.

The code then identifies the bids of the firms that are included in the fringe (not modeled as strategic players), and defines the specification to be estimated and the initial values. The initial values correspond to those of the vector rr identified above.

The paper presents seven specifications. These can be estimated by commenting out the corresponding piece of code as identified within the code. Note that the covariates associated with specifications 2-6 are hard coded within the code.

Finally, the optimization routine is called. The function that computes the objective function for the first six specifications is called dif_mcp_sT.m. The objective function for the last specification is dif_mcp_sTweek.m. The last lines of the file main.m save the output of estimation that corresponds to the value of the objective function, the specification being estimated, and the estimated parameters.

The following files are used by main.m

see_contract.m: this file computes the contract positions of the bidders.

find_contract.m: called by see_contract. Intersects mc and bid curve.

intersect_stepfn_grid.m: finds the intersection points of the two step functions.

adapt_to_grid2.m: intersect step functions.

intersect_grid.m: used by intersect_stepfn_grid.m.

find_firm_size_CH.m: identifies the capacity of each firm.

fringe_dat.m: this file computes the aggregate supply of the fringe.

fringe.m: this file is called by fringe_dat.m to compute the aggregate supply of the fringe.

1.1.2) The file dif_mcp_sT.m computes the objective function that is minimized. The file is called from main.m as part of the estimation routine. The file loops over auctions and for each auction and strategic firm computes the predicted bids for every possible type that a firm can be. For types 0 and 1 this is done separately than for all other types as type 0 players are non-strategic and type 1 players assume everybody is type 0. For types 2 to K, the code first identifies the data for a specific player in a specific auction and computes the predicted bid for all players. This uses as inputs the predicted bids for types 0 and 1. This is done using the file find_expost_optimum.m. Once the bids for all bidders when they are type 2 are computed, the code computes the bids for when the bidders are type 3. This continues, sequentially, until reaching type K. Finally, the code computes the objective function by measuring the distance between the predicted and realized bids, weighting these differences by a triangular distribution, adding across all price points, types, firms, and auctions. The following files are used to compute the predicted bids:

find_expost_optimum.m: this file computes a bidders ex-post optimal bid.

agg_supply_smooth.m: this file smooths the aggregate supply function.

expostbid.m: finds the pairs (p,q) that satisfy the first-order conditions.

find_rd.m: finds the residual demand faced by a bidder.

agg_other_bid_truc.m: aggregates rivals’ bids.

grid_new.m: recomputes the evaluation grid.

sel_type_pr_smooth.m: computes the type distribution for a bidder.

incremental.m: computes incremental quantities in a bidder’s bid.

1.1.3) The file main_19_20.m is used to examine the robustness of the estimates to a different time period. This file uses the same functions as main.m identified above.

1.1.4) The file main_sample_sel.m is used to examine the robustness of the results to different samples of firms included in the CH. This file uses the same functions as the file main.m identified above.

1.1.5) bootstrap_baseline.m is used to estimate standard errors using 250 bootstrap samples. It takes an integer as input and it is executed in the same way as main.m. The file uses the integer provided as input as a seed to randomly choose the sample that is used to compute standard errors. An example submission file is provided as submit_bootstrap.sh, that uses the file BIV_2018.txt as input and executes independent bootstrap estimations using a job scheduler with a matlabsubmit command associated to it. The .sh file can be modified to execute the estimation routine.

1.2) Counterfactuals: inq_iq_counterfactuals.m is used to process all counterfactuals in which strategic sophistication is increased without changing concentration. merger_counterfactuals.m is used to process all merger counterfactuals for the baseline specification.

1.2.1) inq_iq_counterfactuals.m: This file can be run directly on the Matlab prompt. The file first does some ‘house-cleaning’ identifying relevant data and setting the number of draws that are used in simulation. Then, the file uses the following functions to compute the counterfactuals:

agg_supply.m: generates aggregate supply.

find_auc_CH.m: finds the auctions that are studied based on having both bid and mc cost data for strategic players.

find_bids.m: identifies bids from strategic players.

find_bids_pre_draw.m: computes bid for all auctions, bidders, and types.

fringe_dat.m: this file computes the aggregate supply of the fringe.

find_DEC_INC_ALL.m: find auctions on DEC/INC side or choose all the auctions

find_individualcost.m: computes costs by "integrating" under MC function

find_mktclr_new.m: find mkt clearing price.

find_profit1.m: computes profits.

find_qcall_new.m: calculates the market clearing quantity for each bidder.

find_X_Xnew_s.m: find covariate matrix before and after change in sophistication.

inc_IQ_s_draw.m: computes all counterfactuals when sophistication increases exogenously.

replace_bids.m: puts (p,q) pairs in bid format.

see_contract.m: computes the contract positions of the bidders.

type_draw_IncIQ_replace.m: computes type draws given estimated parameters.

1.2.2) merger_counterfactuals.m: This file is used to compute the merger counterfactuals. This file can be executed on the Matlab prompt. The file first loads data and then computes each of the counterfactuals. Files used in this routine are:

agg_supply.m: generates aggregate supply.

find_ahead_sch.m: computes day ahead schedule.

find_ahead_sch_merger.m: computes the aggregated day-ahead schedule of the merging firms.

find_auc_CH.m: identifies the auctions being studied.

find_auc_QC.m: identifies auctions in which contract positions can be computed.

find_bids_pre_draw.m: computes bids for all auctions, firms, and types.

find_DEC_INC_ALL.m: find auctions on DEC/INC side or choose all the auctions

find_firms_merger.m: identifies firms that merge.

find_individualcost.m: computes the individual generating cost.

find_mc_merger_ahead.m: computes marginal cost.

find_org_mc.m: puts cost data in correct format.

find_qcon_merger.m: finds contract positions after merger.

find_X_Xnew_merger_s.m: computes size of the merged firm.

fringe_dat.m: see above.

incremental.m: see above.

mc_merger_ahead.m: computes marginal cost post merger.

merge_firms.m: identifies merging firms.

merger_s_draw_allbidders.m: computes all counterfactual output for the mergers.

replace_bids_merger.m: puts bids in right format.

see_contract.m: see above.

type_draw_merge_replace.m: generates draws for counterfactuals.

1.3) all_plots.m creates the plots reported in the paper. This file can be executed in the Matlab prompt.

2) Stata: These files can be executed as regular Stata files. The files were created using Stata 13.

1.1) Reduced-form evidence: table3_reduced_form_evidence_outage.do This code is used to estimate the regressions in Section VI (Table 3).

1.2) Model fit: profits_fit_outage_and_full.do This code is used to estimate the regressions on model fit. The code performs estimation to measure the fit of the model under the baseline specification and for the outcome of the out-of-sample prediction exercise.

1.3) Process data from counterfactual simulations: process_counterfactuals.do This code reads, one at a time, the csv files generated during the counterfactual simulations, and it is used to compute changes in costs relative to the baseline.

1.4) Appendix_table_B1.do: This code is used to estimate the regressions on learning that are reported in Online Appendix B.