NOTES ON OBLIVIOUS EQUILIBRIUM PROGRAMS

NOVEMBER 2006

1 General Comments

1. These are the same programs as the ones posted in May 2006, but they include new versions of the error bounds.
2. The programs are for Matlab.
3. The programs assume the model primitives described in Weintraub, Benkard and Van Roy (2005). Later, we explain how to customize the programs to different model primitives.
4. *main.m* is the main program that calls the functions to compute oblivious equilibrium, the error bounds, and some industry statistics. This is the program you need to run.
5. *setup.m* is the program where all the parameters are set. If you want to use the same functional forms as in Weintraub, Benkard and Van Roy (2005) you only need to modify the parameters here. *main.m* calls *setup.m*.
6. *main.out* is the output file generated with all the results.
7. Please write to Gabriel Weintraub (gweintra@stanford.edu) with any comments, questions, or if you find an error. Also, if you end up using the algorithms for an application please let Gabriel know. Thank you very much.

2 Customization

We explain what programs must be modified to accommodate different model primitives.

2.1 Profit Function

1. Modify *profit.m* to introduce a new profit function.
2. If the new function needs to use different first order conditions to solve the pricing game, also modify *focprice.m*.

3. Introduce extra parameters required for the new profit function in *setup.m*. Additionally, update structure *profpar* in *setup.m* to pass parameters to functions.

4. Modify *main.out* to incorporate appropriate output printing parameters.

2.2 Entry Process

1. If variable fix_n $= N > 0$ in *setup.m*, the industry has a fixed number of firms, N, and there is no entry and no exit.

2. If variable fix_n $= 0$ and variable $entryproc = 1$ in *setup.m*, then the entry process is Poisson.

3. If variable fix_n $= 0$ and variable $entryproc = 2$ in *setup.m*, then the entry process is deterministic, but still satisfies a zero profit condition.[1]

2.3 Sell-off Value Distribution

1. Modify *contprob.m* to introduce a different sell-off value distribution.

2. Introduce extra parameters required for the new distribution in *setup.m*. Additionally, update structure *transition* in *setup.m* to pass parameters to functions.

3. Modify *main.out* to incorporate appropriate output printing parameters.

2.4 Transition Dynamics

1. Modify *tranprob.m* to introduce different transition dynamics.

2. Modify *optinv.m*, that computes optimal investment, accordingly. Some transition dynamics lead to a closed form solution for optimal investment given a value function. This is our case. If this is not the case, a numerical optimization procedure would be needed.

3. Introduce extra parameters required for the new transition dynamics in *setup.m*. Additionally, update structure *transition* in *setup.m* to pass parameters to functions.

4. Modify *main.out* to incorporate appropriate output printing parameters.

[1] Note that the zero profits condition typically requires a fractional number of entrants to be satisfied exactly, so to accommodate this we instead randomized the number of entrants between the two neighboring integers. For example, if the equilibrium entry rate is 2.5, then the number of entrants is 2 or 3 with probability 0.5. Allowing for fractional numbers ensures existence of equilibrium.