

This [readme file](#) describes the replication code made available for

"Provider Incentives and Healthcare Costs: Evidence from Long-Term Care Hospitals"

by Einav, Finkelstein, and Mahoney.

The enclosed archive contains code only. The associated data is confidential data owned by the Centers for Medicare and Medicaid Services (CMS). Researchers wishing to replicate our results will need to obtain their own copy of the data via an independent Data Use Agreement with CMS. CMS has a standard data application process, but interested researchers should feel free to email us for any help or advice about this process, or for precise information for the files that are needed to replicate the results in the paper.

Once the requisite data files are obtained, the code will need to be updated to reflect the format of the researcher's own copy of the data, and to point to the appropriate repository or storage device.

Some of the analysis in the enclosed code also requires two other sources – data files from the American Hospital Association and commercial software offered by the company 3M, called Core Grouping Software ("CGS"). The former was used for results that are primarily in earlier working paper versions of the paper, and the latter is used to compute LTCH payments. These sources are less critical, and researchers can run skip these parts of the code and still replicate most of the published results.

The code is divided into two main parts. The "Data" sub-directory, which includes mostly Stata and SAS code, and replicates the descriptive parts of the paper (primarily in Section 3), and the "Model" subdirectory, which includes mostly Matlab code, and replicates the model-based results of the paper (Sections 4 and 5).

A. "Data" folder (descriptive results reported in Section 3)

A.1 General structure and organization

The code directory is divided into a handful of main directories:

- directories in /raw/ store raw data, with no code or code that performs minimal cleaning
- directories in /derived/ and /derived_local/ take data as inputs (often from /raw/) and produce data as output
- directories in /analysis/ take data as inputs (often from /derived/ and /derived_local/) and produce results as output
- directories in /lib/ store code libraries that are loaded and used by other directories

Here we describe the contents of each code directory, such as those in `/derived/`, `/derived_local/`, and `/analysis/`.

Most directories have a similar structure:

`/code/`

Contains all scripts and code necessary to execute the directory. In most cases there will be a Stata do file or a SAS script with macros that perform the main functionality.

`/code/externals.txt`

Lists the external resources needed to run the directory, calling versions of data or programs saved in one of the Subversion repositories used for the project. Items referenced in this file will be populated in `/externals/` in the directory. This file can be used to locate other portions of code that create datasets used in an analysis.

`/code/links.txt`

References datasets that are too large for version control and so are stored locally on the machine. Items referenced in this file will be populated as symbolic links under `/external_links/` in the directory. This can also be used to located other portions of code that create datasets used in an analysis.

`/code/make.py`

Executes all of the code in the directory. You will not be able to run this code, but you can use it as documentation for the correct order in which the other scripts are executed.

`/externals/`

Contains data not created at runtime that are necessary to execute the directory. The file `/code/externals.txt` lists its contents. These folders are omitted from the replication files.

`/external_links/`

Contains symbolic links to data not created at runtime that are necessary to execute the directory.

The file `/code/links.txt` lists its contents. These folders are also omitted from the replication files.

`/output/` or `/output_local/`

Contains log files, txt or Excel files with tables, pdf files with figures, csv files with data, or dta files with Stata datasets. This is populated once the `/code/` directory is executed. We have left the log files because they do not contain individually identifying information and their descriptions of the dta files generated by the code will likely be helpful in replication.

In this replication archive, the directories `analysis/`, `derived/`, `derived_local/`, `lib/` and `raw/` live under the `/code/` directory. When parsing the list of externals or links under `/code/externals.txt` or `/code/links.txt`, you will see, for example, `/trunk/analysis/`, `/trunk/derived/`, or `/shared_data/derived_local/`. These will be found in the corresponding folder under `/code/` in the replication archive.

A.2 Underlying data files

The project is built from a series of raw Medicare claims files and the Medicare denominator file, which are not made available in this archive. Instead, we provide symbolic links that can be replaced once researchers obtain their own copy of the data files (unless otherwise noted).

`/raw/Medicare Denominator (100 pct)/`

Holds symbolic links to the denominator files (information about every enrollee)

`/raw/MedPAR (100 pct)/`

Holds symbolic links to the claims files

`/raw/Cost and Use (100 pct)/`

Holds symbolic links to cost and use beneficiary summary files

`/raw/AHA 1998-2014/`

Holds symbolic links to the results of the annual American Hospital Association (AHA) survey (see further documentation at <http://users.nber.org/~jroth/pdf/>)

`/raw/CGS LTCH Parameters/`

Holds payment schedules and other parameters used to compute payments from the CGS software provided by 3M. These are exported from 3M's CGS LTCH price tables.

`/raw/HRR Definitions/`

Holds data necessary to create a ZIP code to HRR crosswalk. Necessary only for the for-profit/nonprofit definition, which was used in earlier versions of the paper. These data is publicly available. We don't include the actual data files, but provide a url from which they can be downloaded

`/raw/Inflation Index/`

Holds the price index which is used to adjust prices for inflation.

A.3 Building from the raw data

The following directories should be run in the order specified in order to create the final datasets for analysis.

`/code/derived_local/MedPAR Append/`

Splits MedPAR files depending on provider type (LTCH, ACH, SNF/IRF) and time period (pre, mid, and post).

`/code/derived_local/Proc Diag Append/`

Keep only 6 procedure codes and 9 diagnosis codes for consistency; splits by facility type.

`/code/derived_local/Denom Append Pre/`

Selects variables of interest from the denominator files for the pre-PPS period.

`/code/derived_local/Denom Append Mid/`

Selects variables of interest from the denominator files for the PPS phase-in period.

(Mostly this is not used; it is in the pipeline due to another project which uses these data.)

/code/derived_local/Denom Append Post/

Selects variables of interest from the denominator files for the PPS period.

/code/derived_local/Death Dates (100 pct)/

Identifies death date for each beneficiary (if it exists).

/code/derived_local/Construct Spells/

Constructs continuous "spells" or "episodes" based on discharge date.

/code/derived_local/Define Origin From Spells/

Updates/corrects the origin field with observed stays in other facilities.

/code/derived_local/Compiled Stays/

Imposes some initial sample restrictions, splits into ACH and LTCH stays
(we are only interested in LTCH stays).

/code/derived/LTCH I Sample/

Imposes additional sample restrictions. ("LTCH I" refers to the internal
name of the project and distinguishes it from the second LTCH project.)

/code/derived_local/CGS Input/

Prepares claims for processing through the CGS software.

/code/derived_local/Run CGS (Calculate)/

Finds the expected payment for every LOS for each LTCH stay, up until the SSO cutoff.

/code/derived/Geography/

Creates a ZIP code to HRR crosswalk. Used only in the for-profit/nonprofit analysis, which is not
included in the current version of this paper.

`/code/derived/AHA Append/`

Identifies facilities as for-profit or nonprofit. (This is mostly not used, but was relevant to some responses to referees.)

`/code/derived_local/Construct Spells/`

Identifies spells (claims whose discharge/admission dates are within 1 day of each other) from claims data. Also creates a subsample of spells following an LTCH stay.

`/code/derived_local/Patient Time in System/`

Determines whether a patient was discharged to home within 90 days of the admission date of a given stay.

`/code/derived_local/Cost and Use Append/`

Computes aggregate utilization data by facility type and year for each patient.

`/code/derived/CPI Annual/`

Cleans the CPI data for each year.

`/code/derived/Post LTCH Discharges/`

Computes the total spending and LOS in the spell following a discharge from an LTCH.

`/code/derived/Pre-PPS LTCH Stays/`

Runs the CGS software to convert pre-PPS DRG codes to PPS codes and compiles all other information about each pre-PPS claim.

`/code/derived/Compiled Stay-Level Data/`

Compiles all information about each PPS claim.

A.4 Creating figures and tables

The following documentation describes our analysis files for generating tables and figures.

`/code/analysis/Facts for LTCH I/`

Generates miscellaneous descriptive facts about the data.

`/code/analysis/Summary Stats PAC/`

Generates summary statistics for the samples.

`/code/analysis/Post LTCH Discharges/`

Summarizes spells following a discharge from an LTCH.

`/code/analysis/Granular Discharge Destinations/`

Tabulates discharge destinations.

`/code/analysis/Mortality Hazard/`

Tests whether mortality (1-day or 30-day) changes sharply at the SSO threshold.

`/code/analysis/Mortality Hazard Graph/`

Plots mortality hazard by relative LOS, with and without CIs.

`/code/analysis/SSO Threshold Variation/`

Uses variation in SSO threshold to test whether an increased SSO threshold affects mortality.

`/code/analysis/Payment Schedule/`

Computes payment schedule by relative LOS for pre-PPS and PPS, and calculates summary statistics for the simplified schedule used in the model.

`/code/analysis/Discharge Rates by Rel LOS/`

Plots the share of patients discharged to various locations by relative LOS.

/code/analysis/Stay-Level Procedures/

Summarizes the most common procedures in pre-PPS and PPS periods.

A.5 Code libraries

We also provide some of the Stata ado files and other code modules necessary to replicate our results. These scripts can be found under */code/lib/*. Note that some software (namely, CGS) is excluded from the delivered library, even though it is referenced at some point in our analysis.

B. “Model” folder (model-based results described in Sections 4 and 5)

Our model directory is structured as follows:

/Matlab/ contains matlab functions for the dynamic discrete choice model. */Matlab/main()* is the driver script to run the model

/Output/Figure is the directory for Matlab figure outputs

/Output/Estimates_Log.xlsx keeps track of our parameter estimates. We also export model results to this excel file, which we use to create tables and figures

/Workspace/ stores results from model results

B.1 Model version naming convention

At the beginning of */Matlab/main()*, we specify the model versions.

A model estimation’s version name takes the form of “vX_Y(_Z)”, where X denotes the grand version,

Y denotes the estimation type, and Z is a tag for the model.

Grand version:

We currently work with v17 and v18. In v17, we estimate two sets of parameters (LTCH and post-LTCH) separately. In v18, we take the estimated parameters from v17 as the starting values and estimate them jointly. We only generate and store model as well as counterfactual outcomes from model v18 for our baseline results. In our two alternative static models (estimation types a8 and a9), we only run and store results from model v17.

Estimation types:

1. a3: Main specification, pooling moments from both pre- and post-PPS moments.
2. a5: Alternative specification. We assume the health autocorrelation is time varying
3. a3_d: Main specification, with post-PPS period only.
4. a8: Myopic model. we assume LTCHs do not consider option values.
5. a9: Static model. We assume LTCHs make discharge decisions based on patients' initial health status.

When storing results, we add "Z" in front of estimate types to indicate results of v18. Without "Z", we store results from v17. The main estimation which produces parameter estimates as well as counterfactual results is Za3_pool.

B.2 Matlab process overview

Model driver:

/Matlab/main() includes the following

1. set up directory paths
2. set up model version as discussed above
3. set up run number of each model version. The combination of "version" and "run" uniquely identifies each iteration of estimation. We need this combination to reference store results in /Workspace/
4. set up initial values for parameters in function setup_params()

5. configure and set up model in `set_model_params_and_globals()`. This function specifies which sets of target moments to import, specifies the payment schedules based on our indication of the “version” name, determines weighting matrix, etc.
6. estimate the model in `estimate_model()`. Results are saved in `/Workspace/` at the end of estimation.
7. plot model fit and optimal policy function with `plot_model_results()`
8. simulate results under various counterfactual payment schemes with `counterfactuals()`
9. conduct robustness exercises in `robustness()`

B.3 Estimation

As discussed above, we proceed with two steps in our estimation. We first estimate LTCH and post-LTCH parameters separately (v17). Then, we take these estimates as initial values as estimate these two sets of parameters jointly, when we also calculate standard error (v18). We need to set global variable `grandver` to v17 or v18 in main to indicate which step we are running. We keep track of estimated parameters as well as their initial values in each model version and run in `/Output/Estiamtes_Log.xlsx`

The main script for estimation is `estimate_model()`. It calls `objective_function()`, which includes fitting pre-PPS moments, post-PPS moments, and post-LTCH payment moments if we are running v18. `estimate_model()` also calculates standard error if running v18.

`objective_function()` takes in model configuration set up, data moments, weighting matrix, and returns the weighted error between predicted and target moments

`collect_obj_results()` returns the predicted moments, as well as a full set of model outcomes, including health distribution at discharge, continuation values, and optimal discharge policy

`evaluate_objective()` runs pre-/post-PPS value function iterations in our dynamic models, or calls `static_simulation_forward()` to run value function iterations of the alternative static model. If we are running v18, `estimate_post_ltch_payments()` fits post-LTCH moments.

B.3 Plot model results

`plot_model_results()` runs the following with estimation results:

`plot_model_fits()` plots predicted vs targeted moments

`export_moment_fit_xls()` exports predicted and target moments to `/Output/Estimates_Log.xlsx`

`plot_policy_function()` generates the implied policy functions

B.4. Counterfactuals

`counterfactuals()` conducts simulate counterfactual outcomes under a set of Medicare payment schemes. We only simulate counterfactuals under our baseline model `v18_a3_pool`

`initialize_cfStruct()` sets up counterfactual schemes

`create_counterfactual_payment_schedules()` generates counterfactual Medicare payment schedules

`run_counterfactual()` evaluates `objective_function()` under each counterfactual payment scheme

`calculate_counterfactual_payments_allCF()` aggregates counterfactual results. Results with tag 'Cfd' allows LTCHs to adjust discharge policy under counterfactual schedules. Results with tag 'Bld' holds discharge policy fixed and only considers the mechanical change in outcomes under different payment schedules

`calculate_post_mort()` calculates post discharge mortality rate under the baseline as well as each counterfactual scheme

`plot_counterfactuals_continuum()` and `plot_counterfactuals_continuum_frontier()` consider a full spectrum of SSO timing, to find the counterfactual that minimize Medicare spending while keeping LTCH weakly better

`plot_policy_function_overlaid()` calculates discharge policy thresholds under each counterfactual

`plot_counterfactuals_by_destination()` calculates discharge shares under each counterfactuals

B.5 Robustness

`robustness()` performs a series of exercise to understand intuition behind our model. We only perform these exercise for our baseline model `v18_a3_pool`

`plot_V()` performs a set of exercises to understand the dynamic nature of the model

- a. plot continuation values (Vs) at different time and health status to understand of the dynamic nature of the model.
- b. plot probability of staying at LTCH until SSO across health status
- c. sensitivity of continuation values (Vs) to perturbation of selected parameters: gamma, scale, and preference

`params_perturb()` examines the sensitivity of predicted moments to parameter perturbation

- a. average sensitivity of discharge moments to parameters. For each discharge destination, we group discharge moments to 4 groups: early, pre-SSO threshold, near-SSO threshold, and post-threshold. We group LTCH parameters to 3 groups: initial health distribution, health transition AR(1) process, LTCH preference
- b. detailed sensitivity of all discharge moments to initial health mean and scale for logit error

C. Tables and figures

We show below how to replicate tables and figures in the paper and appendix

Table 1: `/Data/code/analysis/Summary Stats PAC (SummaryStats.xlsx)`

Table 2: `/Data/code/analysis/Post LTCH Discharges (PostDischargePanel.xlsx)`

Table 3: /Model/Matlab/calculate_counterfactual_payments_allCF/Model/Output/Estimates_Log.xlsx, tab Table 4 lookups, from results in tab /Model/Matlab/plot_counterfactuals_continuumsysPayZa3_pool_nlgmmCfd, LGRIZa3_pool_nlgmmCfd

App Table A1: /Data/code/analysis/Granular Discharge Destinations/ (granular_dests.xlsx) (plus some minimal Excel work)

App Table A2: /Data/code/analysis/Post LTCH Discharges (PostDischargePanel.xlsx)

App Table A3: /Data/code/analysis/Mortality Hazard/ (rd_post_death_coeffs.xlsx; slp_post_death_coeffs.xlsx)

App Table A4: /Data/code/analysis/SSO Threshold Variation (sso_iv_coeffmat.xls; sso_var_coeffs.txt)

App Table A5: /Model/Matlab/estimate_model (/Model/Output/Estimates_Log.xlsx, tab Table 3 lookups, "Myopic Model", model version v18_a8)

App Table A6: /Model/Matlab/estimate_model (/Model/Output/Estimates_Log.xlsx, tab Table 3 lookups, "Static Model", model version v18_a9)

App Table A7: /Model/Matlab/params_perturb (/Model/Output/Estimates_Log.xlsx, tab perturb4x3Za3_pool) /Model/Matlab/calculate_counterfactual_payments_allCF /Model/Output/Estimates_Log.xlsx, tab Table A8 lookups, from results in tab)

App Table A8: /Model/Matlab/plot_counterfactuals_continuum (sysPayZa3_pool_nlgmmCfd, sysPayZa5_pool_nlgmmCfd, sysPayZa3_d_pool_nlgmmCfd, LGRIZa3_pool_nlgmmCfd, LGRIZa5_pool_nlgmmCfd, LGRIZa3_d_pool_nlgmmCfd)

Figure 1: /Data/code/analysis/Payment Schedule (Moments generated from avg per-diem and jump in moments_adj.txt)

Figure 2: /Data/code/analysis/Summary Stats PAC (SumStats_all_Flows.txt)

Figure 3: /Data/code/analysis/Discharge Rates by Rel LOS (hist_los_pre_post.pdf, hist_los_pre_post_lo.pdf, hist_los_pre_post_hi.pdf, hist_los_pre_post_death.pdf)

Figure 4: /Data/code/analysis/Post LTCH Discharges (post_ltch_drgprice_post_pps.pdf)

Figure 5: /Data/code/analysis/Mortality Hazard Graph (mort_haz_1day.pdf, mort_haz_30day.pdf)

Figure 6: /Data/code/analysis/SSO Threshold Variation (sso_thres_util_day.pdf, sso_thres_mort30day.pdf, sso_thres_mort60day.pdf, sso_thres_mort90day.pdf)

Figure 7: /Model/Matlab/plot_policy_function (/Model/Output/Estimates_Log.xlsx, tab policy_functions_post_Za3_pool)

Figure 8: /Model/Matlab/plot_V (/Model/Output/Estimates_Log.xlsx, tab plot_VZa3_pool, PrSSOZa3_pool)

Figure 9: /Model/Matlab/exportxls_payment_schedules (/Model/Output/Estimates_Log.xlsx, tab payment_schedules_postZa3_pool)

Figure 10(a): /Model/Matlab/plot_policy_function_overlaid (/Model/Output/Estimates_Log.xlsx, tab cf_polF_postZa3_pool)

Figure 10(b-d): /Model/Matlab/plot_counterfactuals_by_destination (/Model/Output/Estimates_Log.xlsx, tab cf_share_a_Za3_pool cf_share_s_Za3_pool, cf_share_m_Za3_pool)

Figure 11(a): /Model/Matlab/exportxls_payment_schedules (/Model/Output/Estimates_Log.xlsx, tab payment_schedules_postZa3_pool)

Figure 11(b-d): /Model/Matlab/plot_counterfactuals_continuum_frontier (/Model/Output/Estimates_Log.xlsx, tab LGRI_frontier_postZa3_pool)

App Figure A1: /Data/code/analysis/Payment Schedule (payment_schedule_pre_post.pdf)

App Figure A2: /Data/code/analysis/Mortality Rates by Rel LOS (mort30_dschrge_pre_post.pdf)

App Figure A3: /Data/code/analysis/Mortality Hazard Graph (mort_haz_rd_1day_cis.pdf, mort_haz_rd_30day_cis.pdf)

App Figure A4: /Data/code/analysis/Mortality Hazard (perturb_coeffs_pre_post_1day.pdf, perturb_coeffs_pre_post_30day.pdf)

App Figure A5: /Model/Matlab/plot_model_fit (/Model/Output/Estimates_Log.xlsx, tab moments_pre_Za3_pool, moments_post_Za3_pool)

App Figure A6: /Model/Matlab/plot_model_fit (/Model/Output/Estimates_Log.xlsx, tab moments_pre_a8_pool, moments_post_a8_pool)

App Figure A7 /Model/Matlab/plot_model_fit (/Model/Output/Estimates_Log.xlsx, tab moments_pre_a9_pool, moments_post_a9_pool)

App Figure A8: /Model/Matlab/calculate_post_mort (/Model/Output/Estimates_Log.xlsx, tab post_mortZa3_pool, post_morta9_pool)

App Figure A9: /Model/Matlab/params_perturb (/Model/Output/Estimates_Log.xlsx, tab perturbParZa3_pool)

App Figure A10: /Model/Matlab/plot_V (/Model/Output/Estimates_Log.xlsx, tab perturbVd_Za3_pool)