Measuring The Natural Rate Using Natural Experiments[*]

Verónica Bäcker-Peral Jonathon Hazell Atif Mian
Princeton LSE Princeton and NBER

August 26, 2023

1 Cleaning

In this section, we describe the code files used to clean the data and produce the main data set. All code for this section can be found in the folder: *natural-rate-replication/code/clean*.

The paper utilizes five data sets, (1) Land Registry Transaction Data, (2) Land Registry Lease Data, (3) Land Registry Closed Lease Files, (4) Rightmove Hedonics Data, and (5) Zoopla Hedonics Data, all of which are described in the main paper. Data sets (3), (4) and (5) are confidential data sets purchased for this project, and therefore have not been included in the replication package. The main code in this section, *master.do*, can not be run without this data. However, the file *public.do* produces a subset of the main cleaned data set using only publicly accessible data. Additionally, a version of the final cleaned data set which does not include confidential variables is also provided so that researchers can replicate the main analysis.

In the following subsections, we describe each step of the cleaning process of the main data sets.

1.1 Cleaning the Land Registry Transaction Data

File name: *PricePaid.do*
Below, we describe the main steps we take to clean the data:

[*] Bäcker-Peral: veronicabp@princeton.edu; Hazell: j.hazell@lse.ac.uk; Mian: atif@princeton.edu.

1. **Drop Missing**: We drop rows with missing addresses (0.2% of data), property tenure (.002% of data), and unknown property type (1.5% of data)

2. **Drop Duplicates**: We keep only one entry for each property/date pair. When there are two entries for the same property on the same, and one is labelled as a "Standard Price Paid" entry and the other is labelled as a "Additional Price Paid" entry, we keep the standard entry and drop the other (0.03% of data). When the same property is sold as a leasehold and freehold at the same date, we drop both entries since we are unable to interpret the meaning of these transactions (0.2% of data). Similarly, if the property is sold as two different types (e.g. flat and detached house) at the same date, we drop both entries (0.05% of data). For the remaining duplicates (0.1% of the data), we take the mean price of the duplicated transaction. We flag these duplicated cases. Our results are robust to excluding them from our main analysis.

1.2 Cleaning the Land Registry Lease Data

File names: *ExtractLeaseTerms.py, Leases.do*
Below, we describe the main steps we take to clean the data:

1. **Extract Numerical Values From Free-Text Lease Terms** (*ExtractLeaseTerms.py*): The Land Registry records lease terms as free-text fields. For instance, a typical lease term might say, "99 years beginning on January 1, 1980." We undertake the following steps to extract the relevant lease details for as many leases as possible:

   (a) **Standardize Text**: We remove typos in the month of the lease using a spell-checker function specialized on month names (for example, Jnuary would be corrected to January). We convert ordinal number to cardinal numbers. We convert important holidays to months (for example, Christmas is converted to December and Lady Day is converted to March).

   (b) **Identifying Start of Lease Date**: We identify the date of the lease by looking for key words (such as "beginning on" or "from") that are followed by a date. We also look for cases where the date of the lease is the same as the lease registration date, for instance if the lease is said to start "on the date as mentioned therein." This is done via the use of Regular Expressions (regex).

   (c) **Identifying Length of Lease**: We identify the length of the lease by searching for a one to four digit number followed by the key word "years."

   (d) **Identifying End of Lease Date**: In some cases, the lease does not provide the length of the lease, but instead provides the end date of the lease (e.g. "A lease from January 1, 1980 to January 1, 2077" which can be inferred to be for 99 years). We identify the end date of the lease when it exists by looking for key words like "expiring on" or "terminating on" followed by a date. This allows us to recover the length of the lease for these entries.

2. **Clean Lease Term Data** (*Leases.do*): Next, we clean the data with extracted lease terms as follows:

   (a) **Drop Missing**: We drop rows which are missing either the initial date of the lease or the lease length (0.9% of the data). We also drop rows which are missing the address (0.4% of the data).

   (b) **Drop Duplicates**: We should only have one lease for each property at a given date. In cases where there are duplicate lease lengths for the same lease, if the duplicate lease lengths are close to each other (i.e. the standard deviation of the lease lengths is less than 10% of the mean lease length) we take the mean across the lease lengths (we do this for 1.6% of the data). If there are very different lease lengths recorded for the same date, we drop the mismatching leases (1.7% of the data).

   (c) **Keep Most Recent Lease**: For our purposes, we are interested in only the active lease for a property. We therefore keep only the most recent lease for each property.

1.3 Cleaning the Hedonics Data

File names: *CleanRightmove.do, CleanZoopla.do*
Below, we describe the main steps we take to clean the data:

1. **Clean Rightmove** (*CleanRightmove.do*):

   (a) **Merge Files**: The Rightmove data is provided in several separate files. Pre-2012 files follow a slightly different convention than post-2012 files, so we reformat them to be consistent.

   (b) **Annualize Prices**: Rental prices are sometimes provided at different frequencies (e.g. monthly rent vs annual rent). All rental prices are annualized. When rental price frequency is not provided, we assume that it is monthly, as recommended by the data provider.

2. **Clean Zoopla** (*CleanZoopla.do*): The Zoopla data set was already in a usable format as provided to us, so we perform very minor cleaning edits.

1.4 Merging and Finalizing All Data

File names: *MergeHMLR.py, MergeHMLRHedonics.py, MergeHMLR.do, GetControls.py, GetControlProperties.py, FinalizeData.do, FinalizeExtensions.do, HazardRate.do, EnglishHousingSurvey.do, AdditionalDatasets.do*
Below, we describe the main steps we take to merge and finalize the data:

1. **Matching With a Fuzzy Merge** (*MergeHMLR.py, MergeHMLRHedonics.py*): We develop a fuzzy merge algorithm to match properties according to their address. This algorithm is described in more detail in the main text. We use the algorithm to match transactions from data set (1) to leases from data sets (2) and (3), and we also use it to match transactions from (1) to hedonic entries from data sets (4) and (5).

2. **Merging + Finalizing Data** (*MergeHMLR.do, FinalizeData.do*):

   (a) **Merge HMLR**: First, we merge leases from data set (3) to transactions from data set (1) using a unique match key provided to us by the Land Registry. We then merge leases from data set (2) to transactions based on our match keys from the fuzzy merge.

   (b) **Drop Incongruities**: We drop transactions in which the only lease which matches to a transaction begins more than a year after the transaction is recorded, because it was likely originated after the transaction took place (3.5% of data). We also drop cases where the implied lease duration is negative (.02% of data).

   (c) **Merge Hedonics**: We merge in hedonic data from Rightmove and Zoopla sales and rental listings using our fuzzy merge keys. When there are multiple listings for a property, we keep the listing that is closest in date to each transaction. In cases where there are both Rightmove and Zoopla listings for a property, we keep the Rightmove data because it is more comprehensive. However, there is a very high correlation (about 90%) between Rightmove and Zoopla characteristics. We remove incorrect or abnormal hedonic entries (e.g. bedroom/bathroom count is zero or greater than 10, floor area is greater than 2,000 sq. meters, or rental price is greater than 70% of the sale price).

   (d) **Create Residualized Price Measures**: At this point, we residualize log price on several different variations of our hedonic characteristics. These residualized price variables will be used in our analysis.

   (e) **Merging Other Useful Data**: We also merge in geographical data on the region for each property (e.g. Greater London, West Midlands, Wales, etc.), the Local

Authority to which each property belongs to, and the latitude and longitude coordinate of each property.

(f) **Identifying Extensions**: We identify extensions as properties for which we observed a closed lease and a subsequent lease for at least 30 years more than the original lease at extension time. We drop a small number of properties for which there is an implied negative extension amount (0.9% of extensions). We exclude a small number of properties for which the implied duration at extension is more than 300, since these are likely mistakes (0.2% of extensions). We also exclude a small number of leases for which the closed lease is recorded to expire more than a year after the post-extension transaction takes place, since it is unclear whether the extension took place before or after this transaction (0.8% of extensions).

3. **Getting Controls** (*GetControls.py, GetControlProperties.py*): For each of our extensions, we identify appropriate purchase and sale controls by the procedure described in the main text. The first program returns an index for the purchase and sale price of the control, which is the mean price across all purchase and sale controls, respectively. The second, returns identifiers for each control property included in the purchase and sale controls.

4. **Finalizing Extension Sample** (*FinalizeExtensions.do*):

   - **Dropping Extreme Values**: In our main sample, we drop the top and bottom 0.5% of the data that are outliers based on the difference-in-difference estimator. We also drop the 1% of extensions which have holding periods of less than one year and therefore are likely "flippers," as described in the main text. Finally, we remove the approximately 1% of extensions with very short durations ($T < 30$) since these may incorporate more of the short end of the yield curve and therefore do not approximate y^* as closely as the experiments taking place at higher durations. All of our main results are, however, robust to including all of the dropped experiments.

5. **Create Hazard Rate** (*HazardRate.do*) For each duration, we calculate the conditional probability of extension utilizing the methodology described in the main text.

6. **Clean English Housing Survey Data** (*EnglishHousingSurvey.do*) We compile confidential data from the Special License version of the English Housing Survey. When there are differences in the formatting of responses over time, we standardize it.

7. **Create Additional Data Sets** (*AdditionalDatasets.do*):

- We reshape the data from the experiments so that it can be directly used to produce the event study figure in the main text.
- We merge the control properties identified by *GetControlProperties.py* with the main data so that we can analyze the characteristics of control properties.
- We estimate r_K^* using our experimental methodology as well as a cross-sectional approach for all possible combinations of hedonic controls, to be used in our estimate stability figure in the main text.
- We calculate the housing risk premium and predicted growth rate of rents using a VAR.
- We compute an annual, quarterly and monthly time-series of r_K^*, to be used in the analysis.

All Python code is parallelized and uses the maximum amount of cores by default. To run the code in serial, you must pass the keyword "–parallelize False" when running the code.

2 Analysis

The code to produce all the figures and tables in the paper can be found in the folder: *natural-rate-replication/code/analysis*. The code is divided into programs that correspond to each section in the text. For section where the some of the figures were produced in Python as well as in Stata, two programs are provided. Python analysis code is written in Jupyter notebooks.