Contents

1 Replicating Oswald (2019) 1

2 Software Requirements 1

3 Installation Instructions 2
- 3.1 Installing `julia` Package `mig` . . . 2
- 3.2 Installing `R` package `migR` . . . 2

4 Replication Instructions 3
- 4.1 Replication Hierarchy . . . 3
- 4.2 Level 3 Replication . . . 3
  - 4.2.1 Section 5.3: Parameter Estimates and Moments . . . 4
  - 4.2.2 Section 6.1: Elasticities with respect to Regional Shocks . . . 5
  - 4.2.3 Section 6.2: Why Do Owners Move Less? . . . 5
  - 4.2.4 Section 6.3: Owner Regret . . . 5
  - 4.2.5 Section 6.4: The value of Migration . . . 5
  - 4.2.6 Section E.2 Comparative Statics of a Regional Price Shock . . . 6
- 4.3 Level 2 Replication . . . 6
  - 4.3.1 Chapter 2 Tables and Figures . . . 6
  - 4.3.2 Section 5.1: Estimation of Exogenous Processes and Moment Generation . . . 7
- 4.4 Level 1 Replication . . . 7
  - 4.4.1 SIPP Download . . . 7
  - 4.4.2 SIPP Extraction . . . 8
  - 4.4.3 FHFA and BEA download . . . 8

1 Replicating Oswald (2019)

Unit Tests Status (click): 

This is the replication kit for *The Effect of Homeownership on the Option Value of Regional Migration*, forthcoming in Quantitative Economics.

2 Software Requirements

1. `julia` version 0.6.x (*not* the latest version of julia) is needed to replicate all results from section 3 onwards in the paper.
2. `R` at least version `3.5` for results in chapter 2 and if you want to rebuild the data inputs to the julia package.
3. `git` (optional, but useful. In general `;-)`)

3 Installation Instructions

3.1 Installing `julia` Package `mig`

All results from the structural model are produced with the package `mig`, which is a sub-folder of this repository. Here is how to install it.

1. In order to use the code, you should clone this repository to your computer by typing in your terminal

   ```
   # I refer to where you want this on your computer as MIGJL
   # `MIGJL` is only an example.
   cd MIGJL
   git clone https://github.com/floswald/migration.git
   ```

   or you manually download the source code as a zip folder from here.

2. Go to where you cloned the package to, e.g. `MIGJL`.

3. We'll call `julia6` the julia v0.6.x executable you installed above, in order to comply with the software requirements. Type this in your terminal:

   ```
   julia6 mig/install.jl
   ```

   to download all dependencies and precompile the package.

4. Given the non-standard structure of this repo, the above script tries to *sym link* directory `MIGJL/mig/` into your standard julia package directory at `~/.julia/v0.6` with unix command `ln -s`. This may not work on a windows machine, but I found this for you to try yourself.

5. Run `julia6 -e 'Pkg.test("mig")';` to run the unit tests on your computer.

6. Observe that the (green!) badge on top of this README indicates that the same tests run on MacOS and Linux on Travis-CI.

3.2 Installing `R` package `migR`

The easiest way to deal with dependencies is to directly install from github with this command:

```
library(devtools)  # install `devtools` if not installed.
install_github("floswald/migR")
```

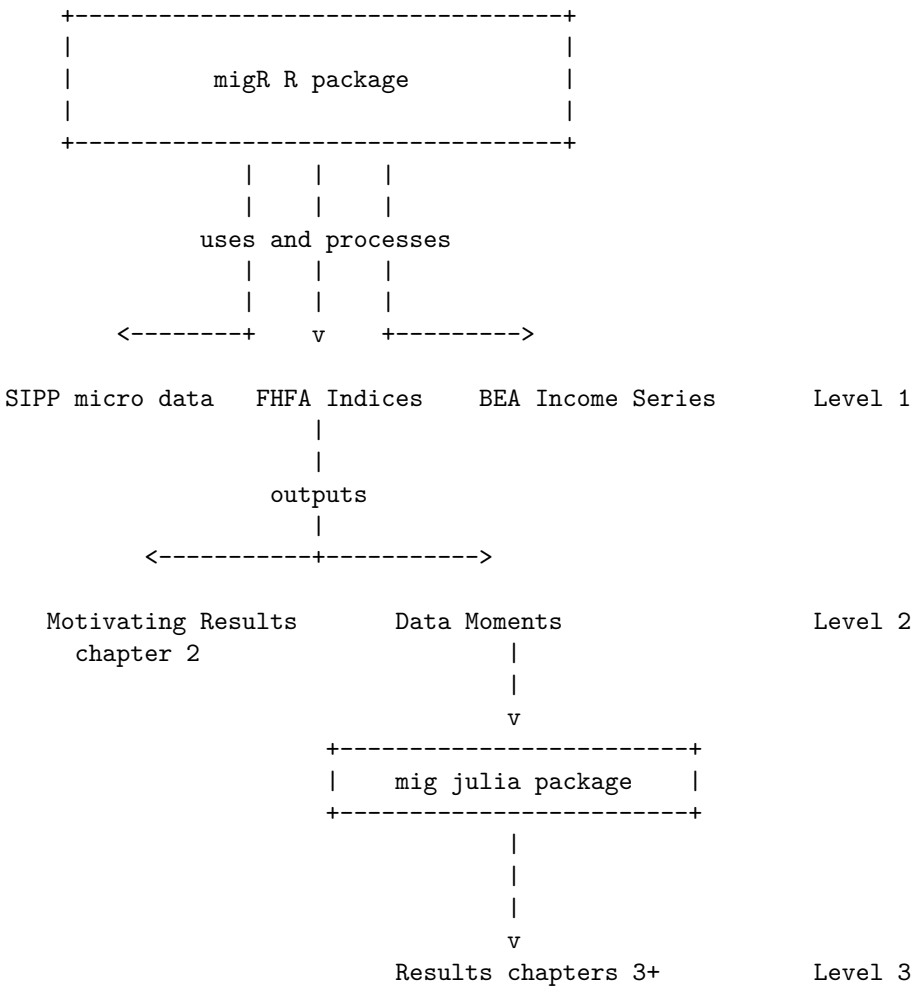
Alternatively, you can install the package using `RStudio`, but you need to make sure yourself that all dependencies are installed. Download the migR_1.0.tar.gz package and install as described.

4 Replication Instructions

This section describes how to replicate all results in the paper.

4.1 Replication Hierarchy

Results in the main text are numbered in standard latin (1,2,...) whereas results in the online appendix are prefixed by a section letter (A.1, A.2,...). There are several levels at which a replication exericse could start, related to how much of the input data one wants to take as given. The structure of this could be illustrated in the following way:



As it is probably most relevant for many users, I will start at the highest `Level 3`, taking all input data as given, then move down the chain.

4.2 Level 3 Replication

This level uses the pre-built input datasets in `MIGJL/mig/in` to run the structural model. The simplest interface is to use the run file which you can launch on your command line in directory `MIGJL/mig` as follows:

```
> cd MIGJL/mig
> julia6 run.jl --help
```

```
The Effect of Homeownership on the Option
Value of Regional Migration (Oswald, 2019)

    | Welcome to the run file of my paper. Please
    | see below how to run the code.
    | Thanks,
    | Florian

Usage:
    run.jl -h | --help
    run.jl --version
    run.jl estim (bgp|grad|slices|stderrors) [--nworkers=<nw>] [--maxiter=<maxit>][--npoints=<npts>]
    run.jl test
    run.jl experiment (elasticity|ownersWTP|ownersWTP2|moneyMC|decomp) [--nworkers=<nw>] [--shock=<sh>]
    run.jl experiment moversWTP [--nworkers=<nw>] [--nosave] [--region=<reg>]
    run.jl experiment noMove [--yshock=<ys>] [--pshock=<ps>] [--nosave] [--nworkers=<nw>]

Options:
    -h --help           Show this screen.
    --nworkers=<nw>     use <nw> of workers for task. [default: 1]
    --npoints=<npts>    number of points to use [default: 10].
    --maxiter=<maxit>   max number of iterations in estimation [default: 500].
    --region=<reg>      in which region to run experiment [default: 1].
    --nosave            don't save experiment output. If you set it, it doesn't save.
    --neg               perform negative elasticity shock. If you set it, it does negative.
    --shock=<sh>        type of shock to apply [default: q]
    --yshock=<ys>       shock applied to regional income [default: 1.0]
    --pshock=<ps>       shock applied to regional price [default: 1.0]
    --version           show version
```

4.2.1 Section 5.3: Parameter Estimates and Moments

4.2.1.1 Table 8: Parameter Estimates

You perform estimation of the model by running

```
julia6 run.jl estim grad
```

optionally specifying the number of workers desired. Standard errors are subsequently obtained by running

```
julia6 run.jl estim stderrors
```

4.2.1.2 Figures 4 and D.1 and tables D.1 and D.2

The fit of the model is illustrated by comparing model and data moments. You can write a table to this extent to disk by running in a `julia6` session

```
> julia6
               _
   _       _ _(_)_     |  A fresh approach to technical computing
  (_)     | (_) (_)    |  Documentation: https://docs.julialang.org
   _ _   _| |_  __ _   |  Type "?help" for help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 0.6.2 (2017-12-13 18:08 UTC)
```

```
 _/ |\__'_|_|_|\__'_|  |  Official http://julialang.org/ release
|__/                   |  x86_64-apple-darwin14.5.0

julia> using mig

julia> runObj(printm = true)
```

The figures are then created with function `plot_moment_fit` in `R` package `migR`.

4.2.2 Section 6.1: Elasticities with respect to Regional Shocks

4.2.2.1 Tables 9 and E.1

```
julia6 --color=yes run.jl experiment elasticity --nworkers=9
julia6 --color=yes run.jl experiment elasticity --nworkers=9 --shock=p
```

Users may find it helpful to refer to script `MIGJL/replicate-AWS.sh` to see how I ran this on an AWS cluster built with cfncluster.

4.2.3 Section 6.2: Why Do Owners Move Less?

Table 10 is created by running

```
julia6 --color=yes run.jl experiment decomp
```

4.2.4 Section 6.3: Owner Regret

Table 12 is created by running:

```
julia6 --color=yes run.jl experiment ownersWTP2 --nworkers=9
```

4.2.5 Section 6.4: The value of Migration

Table 11 is created by running:

```
julia6 --color=yes run.jl experiment noMove --nworkers=9
```

Version 2 and 3 of that experiment, i.e. tables E.11 and E.12 in the online appendix are similarly created with

```
julia6 --color=yes run.jl experiment noMove --nworkers=9 --yshock=0.99 --pshock=0.99
julia6 --color=yes run.jl experiment noMove --nworkers=9 --yshock=0.95 --pshock=0.9
```

Figures 5 and 6, along with several other figures, can be created, after running the above experiments, by doing in a julia6 session

```
julia> using mig

julia> n = mig.read_noMove()    # reads baseline version (table 11)

julia> mig.plot_noMove(n)
```

4.2.6 Section E.2 Comparative Statics of a Regional Price Shock

The tables in that section can be reproduced by running

```
julia6 run.jl experiment scenarios --nworkers=9
```

4.3 Level 2 Replication

This section is concerned with code in a different repository at https://github.com/floswald/migR.

This replication level is taking as given the input data in folder `data/` of the `migR` package. The package help manual with function reference is available as a pdf at the root of the package or online as a searchable website.

You can obtain all code and prebuilt data on the commandline with

```
git clone https://github.com/floswald/migR.git
```

or by downloading the precompiled package as described above

4.3.1 Chapter 2 Tables and Figures

To replicate, you would invoke `R` and load the package with

```
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> library(migR)

>
```

4.3.1.1 Table 2

With the package loaded as above, you get table 2 with function `CPS.distance`

4.3.1.2 Table 3

Function `Sipp.SumStats`.

4.3.1.3 Table 4

Function `SippProbitMove`.

4.3.1.4 Figure 1

Function `PlotSippMigrationRates`.

4.3.1.5 Figure 2

Function `correlograms`.

4.3.2 Section 5.1: Estimation of Exogenous Processes and Moment Generation

To parametrize the structural model, several processes need to be estimated first. This section describes the functions in charge of this task.

4.3.2.1 Tables 5, B.4 and Figures 3, B.3, B.1 and B.2

This is produced by function `Export.VAR`

4.3.2.2 Table 6

This is produced by function `both_prices_output`

4.3.2.3 Income Process

Table and figure C.1 are produced by function `Export.IncomeProcess`

4.3.2.4 Estimation of Movers' Copula

Figures C.2 and C.3 as well as Table C.2 is produced by `Sipp.wage_residual_copulas`

4.3.2.5 Creation of Moments

Function `Sipp.moments` returns a data.table with data moments.

4.3.2.6 Exporting All Inputs for `julia`

Function `Export.Julia` writes all relevant data to disk in `.rda` format.

4.4 Level 1 Replication

4.4.1 SIPP Download

This is the lowest level i.e. it starts with SIPP data acquisition. I used code previously published at ajdamico/asdfree, but it has since evolved. I have a fork the previous version, however, and so the process starts with this:

1. `git clone https://github.com/floswald/asdfree`
2. `cd asdfree/SIPP`
3. uncommenting as warranted, execute scripts `downxxxx.R`.

4.4.2 SIPP Extraction

Once the SIPP database is created locally, we can extract data from it. The function is `Extract.wrap`.

4.4.3 FHFA and BEA download

Functions `get_BEA_persincome` and `download.FHFA` get the macro data series. Notice that `get_BEA_persincome` depends on external package EconData