Chapter 6

Extending and Adapting the Model

Abstract This chapter shows different ways of extending or adapting the model built in previous chapters for other contexts. We show how adding and removing modules, how changing the country, how implementing a deterministic approach for some events and how calibrating projection outcomes on other projections.

Keywords Microsimulation · Population projection · Demography · Method · SAS

6.1 A Flexible Model

In previous chapters, we translated a multistate model projecting education in India and its regions into a microsimulation model, then added two new dimensions, labour force participation and sector of activity, for which we showed examples of alternative scenarios.

The framework of the model can be easily adapted for other purposes. Adapting it for another country would only require us to change the population and parameters files accordingly, while changes in the code would be minor, such as changing the name of regions in the migration module, or if the number of categories of education is different, adapting the code to a larger or smaller number of categories accordingly. Depending on our needs, we could also change the modelling of certain events or add events.

In this chapter, we will adapt the model for another projection with different attributes. We will replicate the multistate projection for China from the Wittgenstein Center (Lutz et al. 2018); we will add labour force participation, and we will calibrate the outcomes on the medium variant of the World Population Prospects (United

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-79111-7_6) contains supplementary material, which is available to authorized users.

Nations 2019). Compared to the projection for India we built in Chaps. 3 and 4, this projection model has the following differences:

- It is not multiregional. We need therefore to turn off the domestic migration module and the shift from rural to urban areas;
- The projection is not closed, which means there is international migration;
- The time span is 2015 to 2100.

In this section, we will detail the required changes in the microsimulation model, beginning by naming the scenario. All changes made compare to the code from Chap. 4 are highlighted in yellow.

```
%let scenario_name=Chapter6_China;
```

6.2 Updating Input Files

Before making further changes in the code, input files need to be updated, including the base population and parameters files. The simulation needs to start from a new micro dataset that matches the population of China in 2015 by age, sex and education. We created this new base population with the exact same procedure as in Chap. 2, but using a different input for the aggregated population (taken from Lutz et al. (2018)). The base population, POP_2015.csv is created in the folder Chapter6_China/Population. The code and the input file can be found in the folder Chapter6_China/NewBasePop. Although we only build an example for China in this section, Lutz et al. (2018) provide data with a similar structure for all countries of the world.

Updated parameters are provided in the folder “Chapter6_China/Parameters”. For the demographic and education dimensions, parameters are extracted from the SSP2 scenario of Lutz et al. (2018). Because the projection is not multiregional, there are no files for domestic migration (dom_mig), rural to urban shifts (ur), or for the sector of activity (formal and formal_input), as these dimensions are not included in this adaptation of the model. The structure of other parameter files stays the same and updated parameters can be directly implemented in files. The region variable is also useless, as there is only one region. However, it’s better to keep it in the input file and fill it in with a single value, such as “China”, as illustrated in Fig. 6.1 showing an example for the parameter file fertility.csv. Otherwise, an additional change would be required in the microsimulation code.

For the parameter file for the labour force participation module (lfp_.csv) and for the imputation of labour force participation in the base population (lfp_imput.csv), we used pooled data of waves 2010 to 2017 of the Chinese General Social Survey (CGSS) to re-estimate the logistic regression model predicting the labour force with personal characteristics (see Eq. 4.1 from the labour force participation module presented in Chap. 4). However, the education variable in the CGSS has only 5 categories: no

	A	B	C	D	E	F
1	region	sex	aeggr	edu	year	asfr
2	China	1	15	e1	2015	0.018111
3	China	1	15	e2	2015	0.018111
4	China	1	15	e3	2015	0.016639
5	China	1	15	e4	2015	0.009544
6	China	1	15	e5	2015	9.90E-07
7	China	1	15	e6	2015	9.90E-07
8	China	1	20	e1	2015	0.174476
9	China	1	20	e2	2015	0.174476
10	China	1	20	e3	2015	0.160288
11	China	1	20	e4	2015	0.13555
12	China	1	20	e5	2015	0.076428
13	China	1	20	e6	2015	0.021548
14	China	1	25	e1	2015	0.130338

Fig. 6.1 Screenshot of the parameter file “fertility.csv” (opened with excel)

education, primary completed, lower secondary, upper secondary and postsecondary. Therefore, in the regression model, the category “no education” includes incomplete primary. The parameters are implemented accordingly in the parameter files.

The statistical model also doesn’t include the region, because the projection is not multistate, nor the presence of children at home, because this survey is not suitable for the inclusion of this variable. Therefore, in the parameter files, the variable for the presence of a young kid at home (`young_kid_p`) and its interaction with education (`young_kid_edu_p`) are set to 0 (see Fig. 6.2, which highlights in yellow parameters that are switched to 0). Similarly, the number of categories for the variable “region” is reduced to only one, which is set to “China” and also takes the value 0. Adapting the parameter file in such a way, that is, setting to 0 parameters of variables that are not used in the modelling rather than removing them from the file, allows us to minimize changes in the code of the labour force participation module.

Other parameters are implemented from regression models. As compared to India, gender gaps are much less prominent in China, as rates of labour force participation for women are only 10 to 15% lower than those of men (see Fig. 6.3). Chinese women with postsecondary education indeed already have rates that are close to those of men. The gender gap exists mainly for individuals with lower education. For both males and females, rates are lower for the populations with no education or primary education for most of adulthood, but become higher past the age of 60, implying that retirement comes later for the minimally educated population.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	sex	edu	young_kic	region	intercept	edu_p	agegr_p	agegr2_p	agegr_edu	agegr2_ec	young_kic	young_kic	region_p
2	1				-0.735								
3	1		0								0		
4	1		1								0		
36	1	e6	1									0	
37	1			China									0
66	0			China									0

Fig. 6.2 Screenshot of the parameter file lfp.csv (opened with excel)

Fig. 6.3 Predicted labour force participation rate by age, sex and education, China. *Source* National Sample Survey on Employment and Unemployment 2017/2018 (India); Chinese General Social Survey 2010–2015 (China)

6.3 Changing the Time Span of the Projection

The time span of this projection is 2015 to 2100 instead of 2010 to 2060. The base population is now `pop_2015.csv` in the population folder. We thus change the code to import accordingly:

```
%import
("C:\Users\Guillaume\Desktop\Microsimulation\scenario_name\Population\
POP_2015.csv", pop.pop_2015);
```

The first thing to do is change the value in the macro loop, calling the simulation accordingly. (As a reminder, the second parameter corresponds to the starting year of the last loop. Thus, 2095 implies that the last loop simulates 2095 to 2100).

```
%loop(2015, 2095);
```

Then, we need to adjust the code in the imputation of the base population section in order to select the proper file, which is `pop_2015` instead of `pop_2010`.

```
/*Imputing the labour force participation and the sector of activity for 2015*/
/*Labour force participation*/
/*Implementing parameters*/

proc sql;
  create table work.lfp_imput as
  select
    p.*,
    t1.intercept, t2.edu_p, t3.agegr_p, t4.agegr2_p, t5.agegr_edu_p,
    t6.agegr2_edu_p, t7.region_p
  from
    pop.pop_2015 p
  (...)
```

Then, in the count of the population, we set the proper year (2015 instead of 2010) in the label for the output files.

```
/*Population by age, sex, region, education, and lfp for 2015*/
proc freq data=work.lfp_imput2 noprint;
table year*agegr*sex*edu*region*formal/list norow nocol nopercent nocum
out=work.output2015(rename=(count=pop) drop=percent);
weight weight;
run;

proc transpose data=work.output2015 out=work.output2015(rename=(_0=inactive _1=Informal
_2=Formal) drop=_name _label_);
var pop;
by year agegr sex edu region;
id formal;
run;

data results.output2015;
set work.output2015;
(...)
run;
```

6.4 Turning Off Modules

There are different ways to turn off a module. If we want to minimize change in the code, we might simply switch values in the parameters files of those modules to 0. Doing so, the event will still be simulated, but since all probabilities are 0, the event will never occur. This is the simpler option, but it has the inconvenience of loading the model with unnecessary calculations, which increases the running time of the simulation. If the sample is large, depending on the power of the computer, this might generate significant unnecessary delay. In the following example, we will show how to turn off the code of the module in order to remove unnecessary calculations from the simulation.

The first thing to do is to remove code lines importing the parameters files of those modules. Because there is no domestic migration, no rural to urban shifts and no sector of activity, the code lines importing parameters (dom_mig.csv, ur.csv, formal.csv and formal_input.csv) should not be read anymore. This can be done either simply by erasing the code, or, as we do in the code below, by embedding it between `/*...*/`. This latter method allows to keep the code without reading it (the color is then changed to green) and can be useful if we want to reactive these codes later.

```
(...)
/*%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\parameters\
dom_mig.csv",param.dom_mig);*/
/*%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\parameters\
ur.csv",param.ur);*/%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\parameters\
lfp.csv",param.lfp); /*%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\parameters\
formal.csv",param.formal);*/
%import ("C
C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\parameters\
lfp_imput.csv",param.lfp_imput);
/*%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\parameters\
formal_imput.csv",param.formal_imput);*/
(...)
```

We also do the same for the code sorting these files.

```
/*proc sort data=param.dom_mig; by region agegr edu sex;run;*/
/*proc sort data=param.ur; by region;run;*/
```

6.4.1 Domestic Migration and Rural to Urban Reclassification

The model for China is multistate, but not multiregional. Therefore, we need to switch off the domestic migration and the rural to urban shifts modules and, consequently, adjust the outputs. Again, we simply embed the code in `/*...*/`.

```
/*Domestic migration module*/
/*proc sort data=work.pop_edu; by region agegr edu sex;run;
data work.pop_dm;
merge work.pop_edu(in=in1) param.dom_mig;
by region agegr edu sex;
if in1;

(...)

drop i a AN_urban--WB_rural a newregion;
run;*/
```

Because this code is not read anymore, the population file “pop_dm” is not created. We therefore need to change the population file read in the next module accordingly, in our case, the fertility module. The fertility module thus needs to start from the last population file created, which is now “pop_edu” rather than “pop_dm”.

```
/*Fertility module*/
/*Adjusting age for exposure*/
data work.pop_birth;
set work.pop_edu;
(...)
```

We then embed similarly the reclassification of rural to urban areas module in order to switch it off.

```

/*Reclassification of rural to urban areas*/
proc sort data=work.pop_birth3; by region; run;
data work.pop_reclass;
merge work.pop_birth3 (in=in1) param.ur;
by region;
if in1;

oldregion2=region;
if death=0 and substr(region,4,5)='rural' then do;
    if rand('uniform')<ur then do;
        region=tranwrd(region,'rural','urban');
        reclass=1; end;
end;

drop ur;
run;*/

```

Again, because the temporary population file “pop_reclass” is not created anymore, we modify the starting population file of the next module (updating the characteristics) with the last one (“pop_birth3”).

```

/*Time module - Updating characteristics*/
data work.pop2;
set work.pop_birth3;
(...)

```

The fertility module used the variable “region_fert”, which was created in the domestic mobility so that the fertility risk would have the right exposure. Since the module is deactivated, the variable is not created and is not useful anymore. Indeed, the region of the mother cannot change during the period. Therefore, the region in which she gives birth does not need to be tracked anymore. However, replacing it everywhere it was used with the variable “region” would be more complicated than simply creating elsewhere a new variable “region_fert” that is a duplicate of region. We add this line at the beginning of the fertility module, in the section adjusting age for exposure.

```

/*Fertility module*/
/*Adjusting age for exposure and recreating the variable region_fert*/
data work.pop_birth;
set work.pop_edu;
agegr_fert=agegr;
if rand('uniform')<0.5 then agegr_fert=agegr+5;
region_fert=region;
run;

```

Also, in the fertility module, newborns had the possibility of domestic migration, which was assessed by comparing the region of birth and the region of residence at the end of the period. We need to remove this line of code.


```

/*Fertility module*/
(...)

/*Create newborn and set their characteristics*/
output;
if young_kid=1 then do;
    birth=1;
    age_moth=agegr_fert;
    agegr=-5; young_kid=0; death=0; cohort=year; region_birth=region_fert;
    /*dom_mig=0; if region_birth ne region then dom_mig=1;*/
    sex=0; if rand('uniform')<(srb/(srb+1000)) then sex=1;
    eduM=edu_fert;
    edu='e1';
    if 5*asfr>1 then weight=weight*5*asfr;
    output;
end;
run;

```

We then need to withdraw the count of domestic migration and rural reclassification in the section generating the outputs. For this purpose, again we simply embed the corresponding code between `/*...*/`.

```

/*Components of population growth*/
(...)

/*proc freq data=work.pop2 noprint;
table year*agegr*sex*edu*region/list norow nocol nopercnt nocum
out=work.inflow(rename=(count=inflow) drop=percent);
weight weight;
where dom_mig=1 and death=0;
run;

proc freq data=work.pop2 noprint;
table year*agegr*sex*edu*oldregion/list norow nocol nopercnt nocum
out=work.outflow(rename=(count=outflow oldregion=region) drop=percent);
weight weight;
where dom_mig=1 and death=0;
run;

proc freq data=work.pop2 noprint;
table year*agegr*sex*edu*region/list norow nocol nopercnt nocum
out=work.gain_urban(rename=(count=gain_urban) drop=percent);
weight weight;
where reclass=1 and dom_mig ne 1 and death=0;
run;

proc freq data=work.pop2 noprint;
table year*agegr*sex*edu*oldregion2/list norow nocol nopercnt nocum
out=work.loss_rural(rename=(count=loss_rural oldregion2=region) drop=percent);
weight weight;
where reclass=1 and dom_mig ne 1 and death=0;
run;*/
(...)

```

The section cleaning the population file for the next period removes temporary variables created to track demographic events. As there is no more domestic mobility and no rural to urban transitions, variables tracking those events need to be removed from the code.

```

/*Cleaning the population file for next period*/
data pop.pop_&endyr;
set work.pop_lfp2;

if death=1 then delete;
drop birth /*dom_mig reclass oldregion oldregion2*/ region_birth;
run;

```

Finally, in the section merging the population count to the components of growth, we remove datasets for the count of these removed components, inflow, outflow, gain_urban and loss_rural.

```

/*Merging the population count and components of growth*/
(...)
data results.output&endyr;
merge work.outputpop work.birth work.death /*work.inflow work.outflow
work.gain_urban work.loss_rural*/;
by year agegr sex edu region;
(...)

```

6.4.2 Sector of Activity

The model for China we are building also does not include the sector of activity. We thus switch off the module in a similar way. However, this needs to be done twice, because the module includes a subtitle (“Formal–Informal event”) that is also embedded in /*...*/.

```

/*Sector of activity*/
/*Implementing parameters*/
/*proc sql;
create table pop_formall as
select
p.*,
t1.intercept, t2.edu_p, t3.cohort_p, t4.cohort_region_p,
t5.young_kid_p, t6.young_kid_edu_p, t7.region_p

(...)

left join
( select sex, region, region_p
from param.formal
where not missing(region_p)
)t7
on p.sex=t7.sex and p.region=t7.region;
quit;*/

/*Formal - Informal event*/
/*data work.pop_formal2;
set work.pop_formall;

(...)

drop intercept cohort_p edu_p cohort_region_p region_p young_kid_p young_kid_edu_p
exp_form prob_form cohort2;
run;*/

```

Since the population file formal2 is not created anymore, we then change the starting population file in the next step to make it correspond to the last one created, pop_lfp2.

```

/*Cleaning the population file for next period*/
data pop.pop_&endyr;
set work.pop_lfp2;
(...)

```

The section generating outputs also needs to be adjusted. As a reminder, the variable “formal” was used to calculate the active population by adding those working in the formal sector to those working in the informal one. In the code producing the population count output, we replace the variable “formal” with the variable “labour” and label categories accordingly (changes highlighted in yellow).

```

/*Generating outputs*/
/*Population by age sex region education, and lfp*/
proc freq data=pop.pop_&endyr noprint;
table year*agegr*sex*edu*region*labour/list norow nocol nopercent nocum
out=work.outputpop(rename=(count=pop) drop=percent);
weight weight;
run;

proc transpose data=work.outputpop out=work.outputpop(rename=(
0=inactive_1=active) drop=_name_ _label_);
var pop;
by year agegr sex edu region;
id labour;
run;

```

Finally, in the section merging the population count and components of growth, the calculation of the active population from the variables “formal” and “informal” can be removed, as this variable is already included in the dataset.

```

/*Merging the population count and components of growth*/
(...)

data results.output&endyr;
merge work.outputpop work.birth work.death /*work.inflow work.outflow
work.gain_urban work.loss_rural*/;
by year agegr sex edu region;
(...)

/*active=formal+informal;*/
pop=inactive+active;
run;

```

The simulation can then be run. However, we need to adjust in similar way the section imputing the base population. We embed the code for the imputation of the sector of activity in /*...*/, since this variable is not projected.

```

/*Sector of activity*/
/*Implementing parameters*/
/*proc sql;
  create table work.formal_imput as
  select
    p.*,
    t1.intercept, t2.edu_p, t3.cohort_p, t4.cohort_region_p,
    t5.region_p

  (...)
    on p.sex=t5.sex and p.region=t5.region;
quit;
*/
/*Formal - Informal imputation*/
/*data work.formal_imput2;
set work.formal_imput;
(...)

drop intercept cohort_p edu_p cohort_region_p region_p exp_form prob_form cohort2;
run;*/

```

Then, in the population count, we change the variable “formal” to the variable “labour” and set the appropriate labels for categories.

```

/*Population by age, sex, region, education, and lfp for 2015*/
proc freq data=work.lfp_imput2 noprint;
table year*agegr*sex*edu*region*labour /list norow nocol nopercnt nocum
out=work.output2015(rename=(count=pop) drop=percent);
weight weight;
run;

proc transpose data=work.output2015 out=work.output2015(rename=
(0=inactive 1=active) drop=_name_ _label_);
var pop;
by year agegr sex edu region;
id labour;
run;

```

Finally, we remove the code calculating the variable “active” from the variables “formal” and informal”.

```

data results.output2015;
set work.output2015;
(...)
/*active=formal+informal;*/
pop=inactive+active;

run;

```

6.5 Building a Deterministic Module in a Microsimulation Model for International Migration

All events in the modules we have shown so far have been modelled stochastically, with random experiments. For some modules, however, we may want to use a deterministic approach. This could be appropriate if we want to perfectly calibrate an event in a multistate cohort-component projection, or if the sample size for some

subgroups of interest is too small (as there is no Monte Carlo error in the deterministic approach). A microsimulation model can incorporate this kind of modelling. We will show how to model an event deterministically, using international migration as an example.

In the examples we presented in Chaps. 3, 4 and 5, the population was closed: KC et al. (2018) assumed no international migration. However, other scenarios or projections might require the implementation of international migration modules. In this example, international migration is implemented through two new modules, emigration and immigration, which are modelled independently. In the projections of Lutz et al. (2018) that we replicate, international migration is implemented after mortality and education events, but before fertility. International immigrants are therefore not in the population that is at risk of dying or changing their level of education in the period in which they arrive, but they can have children.

Although the deterministic approach has the advantage of having no Monte Carlo error, and though it works quite well for events such as mortality and international migration, it can be very complex for other events, such as fertility. It requires the duplication of all women of reproductive age and the consequent adjustment of the weight of the new births, based on the fertility rate. In doing so, the number of rows in the dataset quickly becomes very large (all women aged 15–49 are duplicated in each step). The same issue occurs for interregional migration.

6.5.1 *Emigration*

If implemented stochastically, the emigration event can be modelled in the same way as mortality, by simply comparing the emigration rate to a random number, flagging emigrants, and deleting them from the dataset at the beginning of the next period. When implemented deterministically, no observations are deleted: only sample weights change.

The parameters file for the emigration includes emigration rates (emx) by age, sex and education, as shown in Fig. 6.4. For this scenario, emigration rates are thus constant throughout the projection, but this can be changed by simply adding a column “year” to the parameters file.

The international migration event occurs after the education event and is therefore implemented right after the education module. It starts from the temporary population file `pop_edu`. We merge the parameter file `param.emig` to the population file `pop_edu` in the same way we did for events modelled with a stochastic approach.

```
/*Emigration module - Deterministic*/
proc sort data=work.pop_edu; by agegr edu sex;run;
data work.pop_emig;
merge work.pop_edu(in=in1) param.emig;
by agegr edu sex;
if in1;
(...)
```

	A	B	C	D
1	sex	agegr	edu	emx
2	0	0	e1	0.000238
3	1	0	e1	0.000238
4	0	5	e1	0.000349
5	1	5	e1	0.000344
6	0	10	e1	0.000487
7	1	10	e1	0.000461
8	0	15	e1	0.000666
9	1	15	e1	0.000638
10	0	20	e1	0.000865
11	1	20	e1	0.000879

Fig. 6.4 Screenshot of the parameters file emig.csv (opened with excel)

In the deterministic approach, the number of people who don’t emigrate is determined by adjusting weights according to emigration rates. We store the weight at time t in a temporary variable “old_weight”. For those who survived the mortality module ($\text{death} = 0$), the new weight at time $t + 5$ (we attribute a new value to variable “weight”) is then calculated by multiplying the weight at time t (old_weight) by $1 - \text{emigration rate (emx)}$. In a temporary variable “nb_emig”, we calculate the difference between weights at time t and $t + 5$ (old_weight-weight). When we generate outputs, the sum of this variable will give the number of emigrants for the period.

```
(...)  
old_weight=weight;  
if death=0 then weight=old_weight*(1-emx);  
nb_emig=old_weight-weight;  
  
drop emx;  
run;
```

6.5.2 Immigration

The international immigration module doesn’t require any statistical calculation. Immigrants are simply implemented by merging an immigration microdata file to the population file of the period. This immigration file should have exactly the same structure as the population file. However, we need to set assumptions for the number

and composition of the immigrants. For this, we use the aggregated number of immigrants in China by age, sex, education and year, taken from Lutz et al. (2018), which are generated from international migration flows, taking into account changes in the composition of the world population. From this file, we create the immigration microdata file using the same code as the one used to create the base population (see Chap. 2), but we add a variable “immig = 1” to keep track of immigrants. The output CSV file is labelled “immig.csv” and is stored in the subfolder “parameter”. The complete code and the aggregated CSV file can be found in the subfolder “ImmigrationFile” of this chapter.

The immigration file, from which we select only immigrants of the period (year = &styr), can then be concatenated to the population file of the projection, right after the emigration module in our example.

```
/*Immigration module*/
data work.pop_immig;
set work.pop_emig param.immig (where=(year=&styr));
run;
```

Since we implanted new modules in the middle of the simulation, we need to change the population file on which the next module (fertility) is built, which is now pop_immig instead of pop_edu.

```
/*Fertility module*/
/*Adjusting age for exposure*/
data work.pop_birth;
set work.pop_immig;
(...)
```

Finally, in the assumptions taken from Lutz et al. (2018), the age of immigrants is their age at the end of the period. We therefore need to exclude immigrants of the period (if immig ne 1) when we increment age in the time module.

```
/*Time module - Updating characteristics*/
data work.pop2;
set work.pop_birth3;
if immig ne 1 and (death=0 or agegr=-5) then agegr=agegr+5;
if death=1 and agegr ne -5 and rand('uniform')<0.5 then agegr=agegr+5;
year=year+5;
run;
```

6.5.3 Adjusting the Exposure in the Fertility Module

As mentioned previously, the age at immigration is the one at the end of the period. Since migrants are submitted to the fertility event, some of them could give birth before aging. For the rest of the population, “age” corresponds to age of the beginning of the period. We therefore exclude immigrants (if immig ne 1) when determining their age at birth.

```

/*Fertility module*/
/*Adjusting age for exposure*/
data work.pop_birth;
set work.pop_immig;
agegr_fert=agegr;
if immig ne 1 and rand('uniform')<0.5 then agegr_fert=agegr+5;
(...)

```

For immigrants, age at birth could be their age at the beginning of the period. Accordingly, we adjust it randomly for half of them.

```

(...)
if immig = 1 and rand('uniform')<0.5 then agegr_fert=agegr-5;
region_fert=region;
run;

```

Immigrants and emigrants are both exposed to the fertility event for part of the year. Therefore, some other minor changes are required in the code of the fertility module. The variable “immig” included in the immigration file allows us to track immigrants of the period. Assuming they arrive in the middle of the period, we add a condition similar to the one used for deaths that allows only half of immigrants to be exposed to the fertility event.

```

(...)
/*Fertility event*/
young_kid=0;
if death=0 or (death=1 and rand('uniform')<0.5)
or (immig=1 and rand('uniform')<0.5) then do;
    if rand('uniform')<5*asfr then young_kid=1;*Has a kid age 0-4;
end;
(...)

```

For emigrants, however, we don't have a variable tracking them, since they are calculated by weight adjustments. We therefore need to add one line in the section for creating a newborn in order to transfer randomly either the old weight (before emigration) or the new weight (after emigration) of the mother to the baby.

```

(...)
/*Create newborn and set their characteristics*/
output;
if young_kid=1 then do;
    birth=1;
    age_moth=agegr_fert;
    agegr=-5; young_kid=0; death=0; cohort=year; region_birth=region_fert;
    /*dom_mig=0; if region_birth ne region then dom_mig=1;*/
    sex=0; if rand('uniform')<(srb/(srb+1000)) then sex=1;
    eduM=edu_fert;
    edu='e1';
    if rand('uniform')<0.5 then weight=old_weight;
    if 5*asfr>1 then weight=weight*5*asfr;
    output;
end;
run;

```


6.6 Adjusting Outputs and the Population File for the Next Period

Three temporary variables were used in the new emigration and immigration modules, “old_weight” (which was the weight of individuals before emigration), “nb_emig” (used for compiling the number of emigrants) and “immig” (which tracks immigrants). These variables need to be removed, since for the next period, immigrants should not be counted as new immigrants and should be exposed to the complete risk of fertility. We simply add these three variables to the section for cleaning the population for the next period, next to the other temporary variables to drop.

```
/*Cleaning the population file for next period*/
data pop.pop_&endyr;
set work.pop_lfp2;

if death=1 then delete;
drop birth immig old_weight nb_emig /*dom_mig reclass oldregion oldregion2*/
region_birth;
run;
```

Finally, we want to include immigrants and emigrants when generating outputs of the components of growth. Since immigrants are tracked with the variable “immig”, we can use it with a proc freq to sum up the number of immigrants by age, sex and education, as we do for other components such as deaths and births.

```
/*Generating outputs*/
(...)
/*Components of population growth*/
proc freq data=work.pop2 noprint;
table year*age_moth*eduM*region_birth/list norow nocol nopercnt nocum
out=work.birth(rename=(count=births age_moth=agegr eduM=edu region_birth=region)
drop=percent);
weight weight;
where birth=1;
run;

proc freq data=work.pop2 noprint;
table year*agegr*sex*edu*region/list norow nocol nopercnt nocum
out=work.death(rename=(count=deaths) drop=percent);
weight weight;
where death=1;
run;

proc freq data=work.pop2 noprint;
table year*agegr*sex*edu*region/list norow nocol nopercnt nocum
out=work.immigrants(rename=(count=immigrants) drop=percent);
weight weight;
where immig=1;
run;
```

Emigrants don’t have a variable tracking them, since they are computed by weight adjustment. To calculate their total number, we need to sum up the variable nb_emig that was calculated in the emigration module by subtracting the weight at the end of the period from the weight at the beginning of the period. This can be done with the tabulate procedure. The outputs table is stored in a dataset “emigrants”. In this

dataset, we rename the column counting the sum of emigrants (nb_emig_sum) for “emigrants” and drop the unused variables “_table_”, “_page_” and “_type_”. The statement “var” is used to identify continuous variables, such as nb_emig, while the statement class identifies the categorical variables (year, agegr, sex, edu and region). The table is then built to have the same structure as other datasets for components: we want the sum of emigrants by year, age, sex, education and region.

```
proc tabulate data=work.pop2 out=work.emigrants(rename=(nb_emig_sum=emigrants)
drop=_table_ _page_ _type_);
var nb_emig;
class year agegr sex edu region;
table year*agegr*sex*edu*region, (sum)*nb_emig;
run;
```

Finally, in the section merging population count and components of growth, we add a dataset for the number of immigrants and emigrants.

```
/*Merging the population count and components of growth*/
(...)

data results.output&endyr;
merge work.outputpop work.birth work.death work.immigrants work.emigrants
/*work.inflow work.outflow work.gain_urban work.loss_rural*/;
by year agegr sex edu region;
(...)
```

6.7 Calibrating Simulation Outcomes

For different reasons, we might want to calibrate the projection outcomes on other projections or estimates, either for a specific year or for the entire time span. For instance, the base population of the model we built for India is for 2010, but since then, more recent population estimates by age and sex have been released. We therefore might want to calibrate our outcomes on the more recent estimates, at some point of the projection, such as 2015 or 2020.

A calibration at the end of each period on the outcome of a simpler cohort-component model can also be performed to make two models comparable, or to facilitate the implementation of broader general assumptions. For instance, since our assumptions in the fertility module are age-, region- (for India), and education-specific fertility rates, the total fertility rate (TFR) of India or China will depend on the composition of population, which itself depends on other demographic components of the projection. In other words, we do not know a priori exactly what the forecasted TFR will be, as we do in a cohort-component model. When calibrating on such a model, the underlying assumptions should not be interpreted in terms of their absolute values, but rather in terms of the differences among subgroups.

As example of this calibration, we will calibrate in this section our microsimulation projections of India and China on the projection outcomes of the medium variant scenario of the World Population Prospects (United Nations, 2019) by age, sex and

year. Doing this will increase the consistency between the demographic assumptions, as they will come from the same source, and this will therefore make their outcomes more comparable. Table 6.1 summarizes the broad assumptions.

For India, demographic assumptions for the whole country are about the same in both projections. The main difference between the two projections is the inclusion of the education and sub-regions dimensions in KC et al. (2018). The difference in the net migration assumptions has a negligible impact on the outcomes, as the numbers (about -500 k per year) are marginal compared to the total population size of the country. Consequently, both projections yield very similar results in terms of age and sex size and composition.

As for China, the difference between the SSP2 scenario of the projection of Lutz et al. (2018) that we used to build the microsimulation model and the medium variant of the World Population Prospects is much more appreciable: the present and future total fertility rates are higher by about 0.3 children per woman in the latter. Detailed explanations concerning the different assumptions for fertility can be found in Basten et al. (2014). Calibrating our microsimulation on those of the World Population Prospects will thus indirectly increase equally the age- and education-specific fertility rates that we took from Lutz et al. (2018) in order to match the number of 0–4 year olds at the end of each period that would be given by a TFR of around 1.7–1.8, as assumed by the World Population Prospects. As the modelling of education is kept as well as differentials in demographic behaviours by educational attainment, the calibration will allow the model to merge the education component of Lutz et al. (2018) and the labour force dimension we added to the official projection of the United Nations.

In this section, we will explain the code for calibration implemented in the model of China. The code calibrating the model of India is however the same and can be found in the file “Chapter 6—India calibrated.sas”, located in the folder “Chapter6—India calibrated” in which other files necessary for the projection are also located.

The file calibration.csv, located in the subfolder “param”, contains the population size by age, sex and year from the World Population Prospects (United Nations, 2019). Figure 6.5 shows an excerpt of the file. The column “pop_wanted” represents the population that will be used for the calibration. At the end of each period, we want our outcomes to match with these. We will proceed by adjusting the individual weights accordingly. Note that the older age group in this file is 100 (for 100+), while in our projection, it goes to 125. We will need to take this into consideration when we will calculate the adjustment factors.

The first thing to do is to import this file and sort it. This is done the same way as we did for all other parameter files, with the macro import and the sort procedure in the section for importing files.

Table 6.1 Summary of demographic assumptions for different projections

	India—World Population Prospects 2019 (medium variant)	India—KC et al. (2018)/uncalibrated microsimulation	China—World Population Prospects 2019 (medium variant)	China—Lutz et al. (2018) (SSP2)/uncalibrated microsimulation
Total fertility rate	2015–2020: 2.24	2015–2020: 2.20	2015–2020: 1.69	2015–2020: 1.44
	2055–2060: 1.76	2055–2060: 1.80	2055–2060: 1.76	2055–2060: 1.41
Life expectancy	2015–2020: 69.3	2015–2020: 70.8	2095–2100: 1.77	2095–2100: 1.49
	2055–2060: 76.1	2055–2060: 81.9	2015–2020: 76.6	2015–2020: 76.6
Net migration (annual)			2055–2060: 83.0	2055–2060: 83.9
	2015–2020: –533 K	2015–2020: 0	2095–2100: 87.6	2095–2100: 93.3
	2055–2060: –484 K	2055–2060: 0	2015–2020: –348 K	2015–2020: –184 K
			2055–2060: –310 K	2055–2060: –148 K
			2095–2100: –310 K	2095–2100: –89 K

Fig. 6.5 Screenshot of the parameter file calibration.csv (opened with excel)

	A	B	C	D
1	year	sex	agegr	pop_wanted
2	2015	0	0	46437638
3	2015	0	5	45436987
4	2015	0	10	44264668
5	2015	0	15	46554380
6	2015	0	20	51864604
7	2015	0	25	66864149
8	2015	0	30	51711405
9	2015	0	35	49705968
10	2015	0	40	61857540
11	2015	0	45	63435589
12	2015	0	50	51414420
13	2015	0	55	40972684
14	2015	0	60	40173473
15	2015	0	65	25506514
16	2015	0	70	16608422
17	2015	0	75	11127053
18	2015	0	80	5816694
19	2015	0	85	2143736
20	2015	0	90	518059
21	2015	0	95	86423
22	2015	0	100	7600
23	2020	0	0	44456332
24	2020	0	5	46320144
25	2020	0	10	45349923
26	2020	0	15	44103122
27	2020	0	20	46273865

```

%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\
parameters\immig.csv",param.immig);

%import
("C:\Users\Guillaume\Desktop\Microsimulation\&scenario_name\
parameters\calibration.csv",param.calibration);

(...)

proc sort data=param.immig; by agegr edu sex year;run;
proc sort data=param.calibration; by year sex agegr;run;

```

The calibration module is implemented directly in the simulation loop, once all demographic events are completed, i.e. between the time module and the labour force participation module. It thus starts from the temporary population file `pop2`. First, we make the categories of the age group variable match those of the calibration file. All age groups above 100 are thus reassigned to 100, which now includes all of the population aged 100 and above. Before doing this, we store the old age group variable into a temporary variable `agegr2` that will be used later.

```
/*Calibration*/
data work.pop2;
set work.pop2;
agegr2=agegr;
if agegr>=100 then agegr=100;
run;
```

Using this redefined age group variable, we then produce an output of the population count by age and sex with the FREQ procedure. We also include the variable “year”, since we want the year to appear in the output in order to allow us to match it with the appropriate year of the calibration file. We use the `list` option to have all counts in a single column, and the options `nocol`, `nopercent` and `norow` to remove all percentages from the output. With the `out` option, we store the resulting output table in the work library under the name “simul”, from which we drop the column “percent”.

```
proc freq data=work.pop2;
table sex*agegr*year/list nocol nopercent norow out=work.simul(drop=percent);
weight weight;
run;
```

The column “count” in the table “work.simul” thus represents the population by age and sex at the end of the period, as simulated without calibration. We now need to calculate the adjustment factor for individual weights. We create the temporary file “factor” by merging `work.simul` with the calibration file (`param.calibration`) by year, sex and `agegr`. As a reminder, in the file `param.calibration`, the column “pop_wanted” represents the population we want. The adjustment factor is then calculated by dividing `pop_wanted` by `count`. So if we simulate 1000 individuals for a specific group, while we want 1100, the adjustment factor is 1.1, meaning that the weights of all individuals in this specific group need to be increased by 10% in order to get the target population. As “pop_wanted” and “count” will not be used anymore, we can drop them with the `drop` statement.

```
data work.factor;
merge work.simul(in=in1) param.calibration;
if in1;
by year sex agegr;
factor=pop_wanted/count;
drop pop_wanted count;
run;
```

We can now adjust the weights in the population file “work.pop2”. After sorting it correctly, we merge it by sex and age group with the file “factor” we just created. The variable “weight” can then be adjusted by multiplying it by the variable “factor”.

With the variable `agegr2` we created before, we also reassign the detailed age group categories for the population aged 100. Variables “factor” and “agegr2” will not be used after this point and are therefore dropped from the file.

```
proc sort data=work.pop2; by sex agegr;run;
data work.pop2;
merge work.pop2 work.factor;
by sex agegr;
weight=weight*factor;
agegr=agegr2;
drop factor agegr2;
run;
```

The microsimulation now calibrates the population by age and sex on the projections of the World Population Prospects at the end of each period. To get consistent trends in the final outputs, we also need to calibrate the base population. For this purpose, we use the exact same code as described above, but replace the population file “work.pop2” with “pop.pop_2015” (highlighted in yellow). This code is implemented right before the macro simul, after importing and sorting files.

```
/*Calibration of the base population*/
data pop.pop_2015;
set pop.pop_2015;
agegr2=agegr;
if agegr>=100 then agegr=100;
run;

proc freq data=pop.pop_2015;
table sex*agegr*year/list nocol nopercnt norow out=work.simul(drop=percent);
weight weight;
run;

data work.factor;
merge work.simul(in=in1) param.calibration;
if in1;
by year sex agegr;
factor=pop_wanted/count;
drop pop_wanted count;
run;

proc sort data=pop.pop_2015; by sex agegr;run;
data pop.pop_2015;
merge pop.pop_2015 work.factor;
by sex agegr;
weight=weight*factor;
agegr=agegr2;
drop factor agegr2;
run;
```

Since the calibration module readjusts the population size by age and sex at every step, the components of demographic growth (births, deaths, migrants, etc.) produced in the outputs are not accurate anymore. Interpretations of these components should thus be made with caution.

6.8 Overview of Results

In Fig. 6.6, we compare the final age pyramid by education from the microsimulation before and after calibration with the one from the medium variant from the World Population Prospects (United Nations, 2019) and from the scenario SSP2 from the multistate model of Lutz et al. (2018). As expected, when not calibrated, demographic components of the microsimulation replicate those of the Lutz et al. (2018) approximately. The total population starts slowly declining in 2025 from 1.4 to 0.8G in 2100. The population is then very old, with modal age groups between 75 and 89, but highly educated, with almost all people of working age having at least an upper secondary education and more than half having postsecondary education.

The calibrated microsimulation yields the same age-sex structure as the World Population Prospects. Both yield a much younger age structure in 2100 and a smaller population, as compared to the non-calibrated scenario (which replicates the demographic growth of Lutz et al. (2018)). These differences are due to the underlying assumptions and not to the modelling. Indeed, the difference is mainly explained by different fertility assumptions. In Lutz et al. (2018), the yearly TFRs range from 1.35 to 1.5 between 2020 and 2100, while they range between 1.7 and 1.8 in the medium variant of the World Population Prospects.

We did a similar calibration for the model of India (see folder Chapter6_India calibrated) in order to make the microsimulation model of India comparable to that of China. In Fig. 6.7, we compare results for China and India. By 2025–2030, India will become the world’s largest country in terms of both population size and working-age population (15–64), exceeding China. Both the total and working-age populations will be growing continuously in India, while they will start declining in China. This outcome is well known from other multistate or other cohort-component projections (Lutz et al. 2018; UN 2017). The new outcome of our microsimulation projection model is the labour force participation dimension. We saw in Chap. 4 that India is deprived of many potential workers in its working-age population due to the very low labour force participation of women, while gender gaps are much smaller in China. Consequently, as shown in Fig. 6.7, China will remain the world leader in terms of actual workers for several more decades, until 2060, despite having a lower working-age population from 2030. Indeed, the number of workers in China will also be declining, from 840 M in 2015 to 629 M in 2060, but the gap in labour force size in 2015 (840 M vs 458 M) between both countries is much higher than the gap for the working-age population (1022 M vs 848 M). Without a change in labour force participation rates, much more time is thus required for India to reach the level of China in terms of number of workers. This shows again the high stakes of including labour force participation as source of heterogeneity in population projections.

Fig. 6.6 Comparison of the projected age pyramid in 2100 by education in China from multistate models and from microsimulation

Fig. 6.7 Projected total population size, working-age population (15–64) size, and labour force size, India and China, 2015–2060

References

- Basten, S., Sobotka, T., & Zeman, K. (2014). Future fertility in low fertility countries. In: W. Lutz, W. P. Butz, S. KC (Eds.), *World population and human capital in the 21st century* (pp. 39–146). Oxford: Oxford University Press.
- KC, S., Wurzer, M., Springer, M., & Lutz, W. (2018). Future population and human capital in heterogeneous India. *Proceedings of the National Academy of Sciences of the United States of America*, 115, 8328. <https://doi.org/10.1073/pnas.1722359115>
- Lutz, W., Goujon, A., KC S., et al. (Eds.). (2018). *Demographic and human capital scenarios for the 21st century*. Luxembourg: Publications Office of the European Union.
- UN. (2017). *World population prospects: The 2017 revision, key findings and advance tables*. New York: United Nations Population Division, Department of Economic and Social Affairs.
- United Nations. (2019). *World population prospects: The 2019 revision*. New York, NY: Department of Economic and Social Affairs, Population Division.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.