A TOOLKIT FOR EVALUATING THE WELFARE EFFECTS OF FISCAL CONSOLIDATIONS IN LOW-INCOME COUNTRIES USER'S GUIDE

Xin Tang[1]
SPR-AIDU at International Monetary Fund

March 11, 2019

[1]Correspondence: xtang@imf.org

Abstract

This guide introduces a new toolkit that evaluates quantitatively the welfare effects of fiscal reforms involving three broad tax instruments: value added tax (VAT), corporate income tax (CIT), and personal income tax (PIT). The toolkit provides a graphical interface to access a heterogeneous agents incomplete markets model extended to include multiple sectors and regions that captures salient features of low-income economies. It computes both the steady state equilibrium and transitional dynamics between equilibria. The welfare effects are evaluated by consumption equivalence changes, and are further decomposed to an aggregate and a distributional component. All the results can be exported to Microsoft Excel format. This guide shows how to use the toolkit with an example and provides detailed explanations on the technical features.

CONTENTS

1 INTRODUCTION

Low-income countries have low tax revenue to GDP ratio. The average tax to GDP ratio in low-income countries is 15% compared to that of 30% in advanced economies. Meanwhile, these countries are also those in most need of fiscal space for sustainable and inclusive growth. The Fund often recommends membership countries to strengthen revenue mobilization in surveillance work. It is hence important to evaluate the potential welfare effects of different fiscal reform proposals prior to implementation. This note introduces a new toolkit that evaluates quantitatively the welfare effects of fiscal reforms involving three broad tax instruments: value added tax (VAT), corporate income tax (CIT), and personal income tax (PIT).

In particular, the toolkit provides a Graphical User Interface (GUI) for the user to compute the model built in Peralta-Alva et al. (2019) under different parameterizations. The economies in low-income countries have their unique features. In particular, low-income countries usually have a large unproductive agricultural sector, face serious informality issue, lack instruments to insure against idiosyncratic income risks, and observe a sharp rural-urban distinction. As a result, to guide our analysis, we extend the workhorse heterogeneous agents incomplete markets model [Aiyagari (1994)] to cope appropriately with the economic structure of low-income countries.

Specifically, we consider a model with four sectors of different productivity levels—food, manufacturing, services, exporting cash crops—two regions with segmented labor markets—rural and urban—and a unified capital market. Each region is populated by a continuum of households who consume food, manufacturing goods, and services. Each household also faces persistent idiosyncratic productivity shocks that can only be partially insured against using one period risk-free bond. Based on their comparative advantage, households divide their total hours between the formal and informal markets in their dwelling region [Roy (1951)]. The formal and informal labor markets in each region host different sectors. The agricultural sector is hosted exclusively in the rural area, where workers in the formal and informal markets are hired to produce respectively cash crops and food. On the other hand, manufacturing goods and services are provided respectively by urban households who work in the formal and informal markets. A utilitarian government has access to three Ramsey taxes: value added tax (VAT) on food and manufacturing goods consumption, personal income tax (PIT) on formal wage income, and corporate income tax (CIT) on revenues of manufacturing firms and profits from cash crops production.

We walk the user over a quantitative example to show how the users can apply the above model to Fund's operational work with the aid of the toolkit. In the example, we will analyze the welfare effects of a 4 percentage points increase in VAT for a target economy. In particular, we first explain how to orchestrate different components of the toolkit to parameterize the model for a target economy, a process we call *calibration* throughout. Calibrating the model requires only macro-level data such as sectoral consumption/output ratio and regional Gini coefficients that are widely available for many countries. We then show how the calibrated model can be used to conduct policy experiments. We compute both the steady state equilibria with respect to the two fiscal regimes as well as the transition dynamics between the equilibria. We demonstrate how welfare decompositions can be easily implemented by the toolkit, and how to analyze other aspects of the reform by taking advantage of the rich macroeconomic and distributional results computed by the toolkit.

The rest of the note is organized as follows. Section 2 reviews the theoretical framework and introduces to users' not familiar with modern quantitative macroeconomics the concept of calibration. Section 3 provides an overview of the structure of the toolkit and explains how to launch the main interfaces of the toolkit. Section 4 contains detailed instructions on how to calibrate the model, change the policy parameters, compute the steady state as well as transitional dynamics, conduct welfare decompositions, and export results to Microsoft Excel format. Section 5 covers several advanced topics, including exceptional handling, using the toolkit *without* MATLAB, and acceleration by parallel computing. Section 6 concludes. The toolkit is an update from a previous version. Hence for returned users from the previous version, Appendix A lists all the changes in the update, and Appendix B describes the system requirements. In the interests of space, we take the liberty to refrain from describing the algorithms to solve the model and direct the users to our working paper [Peralta-Alva et al. (2019)]. Throughout, the reader is assumed to have knowledge of the general equilibrium version of the standard Cass-Koopmans model equivalent to graduate students who have completed the first year core courses in macroeconomics.

The typographical conventions used in the note are as follows. Variables, MATLAB routines, file names, and buttons in the GUI appear in `Blue Computer Modern Typewriter`. Results printed to the terminal during execution are set in `Black Computer Modern Typewriter` beginning with `>`. The same typeset is also used for commands entered by the users interactively, but begins with `$` (not to be entered).

2 The Model

This section describes the theoretical framework that the toolkit implements. We start by providing a descriptive narration of the structure of the model, and after that we will introduce the formal mathematical presentation. Interested readers are also encouraged to read our companion working paper [Peralta-Alva et al. (2019)] for more details. For the convenience of the readers, we have kept the description of the model mostly the same as in the paper. We then explain from a high level how to map the model to data by introducing the concept of calibration.

2.1 *The Environment*

We extend the standard Aiyagari (1994) heterogenous agents incomplete markets model to capture salient features of low-income economies, namely a large agricultural sector, significant informal economy, and a distinction between urban and rural areas.

Take Ethiopia as an example of a typical developing country. Ethiopia has a purchasing power parity adjusted GDP per capita of about 3% of the U.S level, making it one of the poorest countries in the world. It has a large and unproductive agricultural sector, which employs about 70% of the work force. Agricultural production in Ethiopia is overwhelmingly of a subsistence nature, and a large part of commodity exports are provided by the small agricultural cash crops sector, with coffee as the largest foreign exchange earner, and its flower industry becoming a new source of revenue in recent years. Our model attempts to capture these features in a stylized way.

Our model is a small open economy with two regions—rural and urban—and four sectors—domestic and exporting agriculture (later referred to as food and cash crops), manufacturing, and services.[1] Each region is populated by a continuum of infinitely lived households. The population shares for rural and urban areas are μ^r and μ^u. Each household is endowed with one unit of time, which is divided between the regional formal labor market and informal self-employment. We assume that labor markets in the two regions are segmented, and households cannot migrate.[2]

The agricultural sectors are exclusively hosted in the rural area, with the manufacturing and services sectors housed in the urban area. More specifically, in the rural area, households work informally on their own arable land to grow food. A share μ^f of large farmers hire labor in the formal market to produce both food and cash crops. We use the large farmer to model in a parsimonious way the large-scale agricultural exporting sector and seasonal hiring during the labor intensive stages of agricultural production (e.g., planting and harvesting) which rely on temporarily hired labor.[3] In the urban area, households work informally to provide services and a representative neoclassical firm hires labor in the formal market together with capital to produce manufacturing goods. We assume that urban households face idiosyncratic productivity shocks in the manufacturing sector, while the shocks hit rural households when they work in their own plot.[4] With the combined formal and informal income, households make a consumption-saving decision where they have access to one risk-free asset and divide their total consumption expenditure over food, manufacturing goods, and services optimally. Because different households have different realizations of idiosyncratic shocks, savings and hence total resources available would be different. Heterogeneity in saving thus generates a non-degenerate distribution of households in the model. We maintain the standard assumption in the literature that savings are turned into capital of equivalent value. Further, in our model, cash crops production is modernized in the sense that it also employs capital, meaning that capital is used in both manufacturing goods and cash crops production. Both capital depreciates at rate δ. Large farmers' income sources are revenues from selling domestic and exporting agricultural goods. The income is used to finance consumption and investment in machinery. We let the manufacturing goods be the numeraire, and p^a and p^s be the relative price of food and services. The wage rates are w^m and w^f respectively for urban and rural formal market. The risk-free asset yields a return of r.

We assume that food and services are used exclusively for domestic consumption, and cash crops serve

[1] As will be explained in the calibration section, these sectors should not be interpreted by their literal names. By manufacturing and services, what we really mean is goods and services that produced formally or informally in the urban area. For instance, airline, telecommunication, and modern financial services are mapped to the manufacturing as opposed to services sector in our model. Hence what differentiates them is not their statistical labels, but the way production is organized, here in particular, the production functions.

[2] We argue that this assumption is innocuous in our context, since according to a large literature in labor and macroeconomics, migration in developing countries is usually driven by factors other than taxes, especially for small and moderate changes of tax rates around the status quo level. See Lucas (1997) for a review of the early work, and Lagakos, Mobarak and Waugh (2017) for recent evidence.

[3] For more details on the organization of agricultural production in low-income countries, see the handbook chapter by Eastwood, Lipton and Newell (2010).

[4] The rest of the sectors are assumed to be risk free for simplicity. For the main results to hold, we only need their risks to be quantitatively smaller. Section I.C provides more explanation on the modeling assumption.

only the international market. We assume that in each period, the current account is balanced by the government through importing manufacturing goods. The manufacturing goods are used for consumption and capital. This broadly captures the pattern that developing countries typically export cash crops in exchange for manufactured goods, and fulfill their subsistent needs primarily from domestic sources [Gollin, Parente and Rogerson (2007) and Tombe (2015)].

The government's objective function is utilitarian. It has access to three Ramsey (linear) taxes: valued added tax (VAT, τ^a) on food and manufacturing goods, personal income tax (PIT, τ^w) on households' income from formal markets, and corporate income tax (CIT, τ^r) on manufacturing firms and large farmers. We assume that the government can tax domestic agricultural sector following the evidence presented in Anderson, Rausser and Swinnen (2013) and Adamopoulos and Restuccia (2014). The government spends expenditure G on manufacturing goods, but the expenditure is not directly valued by households. We assume that the government runs balanced budget in every period. This implies that government expenditure varies between equilibria since we study different revenue mobilization scenarios. In the quantitative exercises, we also allow the government to do lump-sum transfers.

Put together, there are five endogenously cleared markets: domestic agricultural and services goods markets, rural and urban labor markets, and the capital market. Demand and supply curves in these markets are characterized by the optimization problems of different economic agents as specified above. Aggregate demand and supply curves are then constructed by integrating these individual policy functions using the stationary distributions over households in the equilibrium. These demand and supply curves can be solved under different prices, and the solution concept of the equilibrium is a vector of prices that clears all the markets.

2.2 *Formal Presentation*

In this section, we lay out the model formally. If you find that you more or less understand how different components of the model interact with each other, this section can be safely skipped.

Preference.—We assume that both types of households and the large farmer share the same preference over sequences of consumption on food, manufacturing goods, and services $\boldsymbol{c}_t = [c_t^a, c_t^m, c_t^s]$:

$$U = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(\boldsymbol{c}_t)\right],$$

where β is the time preference. We assume that the per period utility is log-linear in the three goods:

$$u(\boldsymbol{c}_t) = \log c_t^a + \gamma \log c_t^m + \psi \log c_t^s, \tag{1}$$

where γ and ψ are relative preferences over manufacturing goods and services, and subscript t refers to time periods.[5] With log linear preference, if the household's total consumption expenditure is $\overline{C}_t$, in the optimal

[5]Notice that here we use the conventional homothetic utility function as opposed to the non-homothetic Stone-Geary preference with subsistence requirement $\bar{a}$ that is typically used in the development literature [Restuccia, Yang and Zhu (2008), Lagakos

consumption bundle, the household distributes the budget to c_t^a, c_t^m and c_t^s by shares $1/(1+\gamma+\psi), \gamma/(1+\gamma+\psi)$ and $\psi/(1+\gamma+\psi)$ respectively, which, given taxes and prices, further leads to

$$u(\boldsymbol{c}_t) = u(\overline{C}_t) = \overline{U}_t + (1+\gamma+\psi)\log \overline{C}_t, \tag{2}$$

where

$$\overline{U}_t = \log\left(\frac{1}{1+\gamma+\psi}\cdot\frac{1}{(1+\tau_t^a)p_t^a}\right) + \gamma\log\left(\frac{\gamma}{1+\gamma+\psi}\cdot\frac{1}{1+\tau_t^a}\right) + \psi\log\left(\frac{\psi}{1+\gamma+\psi}\cdot\frac{1}{p_t^s}\right),$$

is a time varying constant. As a result, the consumption-saving decision and optimal consumption bundle decision of the households can be analyzed independently. We use this property in later sections to simplify the notation.

Informal Markets.—By working informally, rural households produce food according to the production function:

$$y_t^a = z^a \varepsilon_t^r (1-h_t^r)^{1-\alpha^a}, \tag{3}$$

where z^a is economy-wide agricultural productivity, $1-h_t^r$ is the share of labor supplied *informally*, and $1-\alpha^a$ is the labor share. We assume decreasing returns to scale here because land transferability in developing countries is usually limited.[6] In the production function (3), we use the idiosyncratic shock ε_t^r to capture the fact that agricultural production is risky. This is especially the case in developing countries which usually lack irrigation system or hydraulic infrastructure to smooth weather shocks, or pest control services to resist outbreak of pest infestation.[7] It also captures unobserved variations in crop yields due to difference in individual farmers' ability or crop choice. Similarly, by working informally, urban households provide services according to the production function:

$$y_t^s = z^s (1-h_t^u)^{1-\alpha^s},$$

where z^s is economy-wide services productivity, $1-h_t^u$ is the share of labor supplied *informally*, and α^s is the labor share.

and Waugh (2013), and Adamopoulos and Restuccia (2014)]. This is because with homothetic preference, we can back out the consumption equivalence of welfare changes from the value functions directly, while with non-homothetic preferences, we have to rely on Monte Carlo simulations. In the literature, the Stone-Geary preference is usually used to generate a large agricultural sector in the spirit of Schultz (1953)'s "food problem," or more recently to increase relative risk aversion of the poor people [Chetty and Szeidl (2007) and Donovan (2018)]. Neither factor is crucial in our setting for several reasons. First, the rural population and hence agricultural employment in the model is exogenously given because we preclude cross-region migration. Also since we do not conduct cross-country comparison in our paper, the size of agricultural sector in our model does not have to show substantial response to changes in aggregate productivity. As a result, our model faces less challenge in generating a large agricultural sector. Second, in the earlier IMF Working Paper version of the paper [Peralta-Alva et al. (2018)], we solve a version of the model with Stone-Geary preference with $\overline{a}$ calibrated to match the food consumption share of people in the bottom quintile of the consumption distribution, but found that both the value and the impact of $\overline{a}$ are small. This alleviates the second concern.

[6]For Ethiopia in particular, see the evidence presented in Chen, Restuccia and Santaeulàlia-Llopis (2017). Wu et al. (2018) provide a discussion of the land and migration policies in China. For land policies in other developing countries in general, please refer to the online appendix of Adamopoulos and Restuccia (2014).

[7]See for example, Duflo and Pande (2007), Allen and Atkin (2017), Burgess et al. (2017), Donovan (2018), and Lagakos, Mobarak and Waugh (2017).

Formal Markets.—We assume that labor productivity in the urban formal market is also subject to idiosyncratic shock ε_t^u. The shock reflects factors such as variations in the labor efficiency caused by matching or fluctuations in employment status. Thus the actual pre-tax labor income per unit labor in the urban area is $w^m\varepsilon_t^u$. Manufacturing firms produce by the production function:

$$y_t^m = z^m(k_t^m)^{\alpha^m}(h_t^m)^{1-\alpha^m},$$

where z^m is manufacturing productivity, k_t^m is the total capital, α^m is the capital share, and h_t^m is the total *effective* labor units hired. Meanwhile, large farmers produce food following the same production function as (3):

$$y_t^{a,f} = z^a(h_t^a)^{1-\alpha^a}.$$

Recall that we assume that the production of cash crops is modernized. This means that it requires modern farm machinery k_t^f:

$$y_t^* = z^*(k_t^f)^{\alpha_1^*}(h_t^*)^{\alpha_2^*},$$

where z^* is exporting sector productivity and α_1^* and α_2^* are factor shares. For the same reason of limited land transferability before, here we assume $\alpha_1^* + \alpha_2^* < 1$. Because agricultural modernization helps greatly to reduce the risk, hence here for simplicity, we assume that that there is no idiosyncratic risk associated with the rural formal market.[8]

Households' Optimization.—To simplify the notation, we let

$$\boldsymbol{C}_t^j = (1+\tau^a)(p^a c_t^{a,j} + c_t^{m,j}) + p^s c_t^{s,j}, \quad j \in \{u, r, f\},$$

denote the total consumption expenditure for urban, rural households, and large farmers.[9] Throughout it is understood that the $\boldsymbol{C}_t^j$s are always spent according to the optimal consumption bundle.

Let $b^{j'}$ represents savings. We assume that the ε^js are Markovian processes. The recursive problem of the urban households is:

$$V^u(b^u, \varepsilon^u) = \max_{\{\boldsymbol{C}^u, b^{u'}, h^u\}} \left\{ u(\boldsymbol{C}^u) + \beta \mathbb{E}[V^u(b^{u'}, \varepsilon^{u'}|\varepsilon^u)] \right\} \tag{4}$$

$$s.t.$$

$$\boldsymbol{C}^u + b^{u'} = (1-\tau^w)\varepsilon^u w^m h^u + p^s z^s (1-h^u)^{1-\alpha^s} + (1+r)b^u.$$

That of the rural households is

$$V^r(b^r, \varepsilon^r) = \max_{\{\boldsymbol{C}^r, b^{r'}, h^r\}} \left\{ u(\boldsymbol{C}^r) + \beta \mathbb{E}[V^r(b^{r'}, \varepsilon^{r'}|\varepsilon^r)] \right\} \tag{5}$$

$$s.t.$$

$$\boldsymbol{C}^r + b^{r'} = (1-\tau^w) w^f h^r + p^a z^a \varepsilon^r (1-h^r)^{1-\alpha^a} + (1+r)b^r.$$

[8] For the reduction of agricultural risks by modernization, see for instance, Duflo and Pande (2007) and Pingali (2007).

[9] Though we assume that large farmers also consume all kinds of goods, because in the quantitative exercise they only consist 3% of the total population, their impacts on consumption are small. It is quantitatively similar to model them as neoclassical firms operating solely in the rural area.

The formulation above implies that more productive households (that is those with higher ε^js) choose to work in the manufacturing sector in the urban area and in their own plots in the rural area. This is consistent with the empirical evidence in La Porta and Shleifer (2014) and Eastwood, Lipton and Newell (2010). In particular, La Porta and Shleifer (2014) show that the formal sector is usually more productive because of human capital privilege, and Eastwood, Lipton and Newell (2010) argue that hired labor is less productive in agricultural sector as a consequence of moral hazard and high costs in monitoring.

The two dynamic programming problems share very similar structure. The solutions to these two problems are policy functions on consumption $c^j(b,\varepsilon)$, saving $b^{j'}(b,\varepsilon)$, and labor supply to formal markets $h^j(b,\varepsilon)$, where $j=u,r$. The joint cumulative distribution functions of households in the rural and urban areas in the steady state are denoted respectively by $\Gamma^r(b^r,\varepsilon^r)$ and $\Gamma^u(b^u,\varepsilon^u)$.

Farmer's and Firm's Optimization.—Specifically, the sequential problem of the large farmer is

$$\max_{\{\boldsymbol{C}_t^f,k_{t+1}^f,h_t^a,h_t^*\}} \sum_{t=0}^{\infty}\beta^t u(\boldsymbol{C}_t^f) \tag{6}$$

$$\begin{aligned} s.t.\quad & \\ & \boldsymbol{C}_t^f + k_{t+1}^f = (1-\tau^r)(\pi_t^f+\pi_t^*) + (1-\delta)k_t^f + \tau^r\delta k_t^f, \\ & \pi_t^f = p^a z^a (h_t^a)^{1-\alpha^a} - w^f h_t^a, \\ & \pi_t^* = z^*(k_t^f)^{\alpha_1^*}(h_t^*)^{\alpha_2^*} - w^f h_t^*, \end{aligned}$$

where CIT is collected over farmer's profits. The manufacturing firm's problem is

$$\max_{\{k_t^m,h_t^m\}} \left\{(1-\tau^r)z^m(k_t^m)^{\alpha^m}(h_t^m)^{1-\alpha^m} - w^m h_t^m - (r+\delta)k_t^m\right\}. \tag{7}$$

Notice that CIT is imposed on firm's revenue since neoclassical firm earns zero profits. The depreciation δ is also paid by the firm.

Government.—To specify the government budget constraint, we introduce several notations to ease the exposition. Define

$$C_t^x = \mu^u \int c_t^{x,u} d\Gamma^u(b_t^u,\varepsilon_t^u) + \mu^r \int c_t^{x,r} d\Gamma^r(b_t^r,\varepsilon_t^r) + \mu^f c_t^{x,f}, \qquad x\in\{a,m,s\},$$

the aggregate consumption of each goods,

$$H_t^u = \int \varepsilon_t^u h_t^u d\Gamma^u(b_t^u,\varepsilon_t^u), \quad H_t^r = \int h_t^r d\Gamma^r(b_t^r,\varepsilon_t^r),$$

as the total efficient units labor supply to the formal markets in urban and rural areas, and

$$y_t^m = z^m(k_t^m)^{\alpha^m}(h_t^m)^{1-\alpha^m},$$

the total revenue of domestic manufacturing firms. Then the government budget constraint is

$$G + \mu^f\tau^r\delta k_t^f = \tau^a(p^a C_t^a + C_t^m) + \mu^f\tau^r(\pi_t^f+\pi_t^*) + \tau^r y_t^m + \tau^w(\mu^u w^m H_t^u + \mu^r w^f H_t^r), \tag{8}$$

where $\mu^f \tau^f \delta k_t^f$ is the tax deduction to agricultural machinery investment.

Stationary Equilibrium.—We define formally the recursive competitive equilibrium in the steady state in this section. The equilibrium along the transition path could be defined similarly. We refer the users to Appendix A.3 of the paper.

Definition 1. (Recursive Competitive Equilibrium) *A recursive competitive equilibrium for the economy consists of equilibrium prices* $\boldsymbol{p} = \{p^a, p^s, w^m, w^f, r\}$*, value functions* $V^j(b^j, \varepsilon^j)$*, consumer decision rules* $\{c^{x,j}(b^j, \varepsilon^j), b^{j'}(b^j, \varepsilon^j), h^j(b^j, \varepsilon^j)\}$*, cumulative distribution functions* $\Gamma^j(b^j, \varepsilon^j)$*, where* $j \in \{u, r\}$ *and* $x \in \{a, m, x\}$*, farmer's decision rules* $\{c^f, k^f, h^r, h^*\}$*, and firm's decisions* $\{k^m, h^m\}$*, for any given policies* $\{\tau^a, \tau^r, \tau^w\}$*, such that*

(i) Given $\boldsymbol{p}$*,* $V^j(b^j, \varepsilon^j)$ *and* $\{c^{x,j}(b^j, \varepsilon^j), b^{j'}(b^j, \varepsilon^j), h^j(b^j, \varepsilon^j)\}$ *solve the households' optimization problems (4) and (5);*

(ii) Given $\boldsymbol{p}$*,* $\{c^f, k^f, h^r, h^*\}$ *solve the large farmer's optimization problem (6);*

(iii) Given $\boldsymbol{p}$*,* $\{k^m, h^m\}$ *solve the firms' optimization problem (7);*

(iv) (Aggregate Consistency) $\Gamma^j(b^j, \varepsilon^j)$*s are stationary distributions corresponding to the joint transition matrices* Π^j *constructed from* $b^{j'}(b^j, \varepsilon^j)$ *and the transition matrices of* ε^j*,* $j = u, r$*:*

$$\begin{aligned}\Pi^j &= \Pr[b_{t+1} = b', \varepsilon_{t+1} = \varepsilon' | b_t = b, \varepsilon_t = \varepsilon] \\ &= \Pr[b = b_{t+1}^{-1}(b', \varepsilon_t = \varepsilon)] \Pr[\varepsilon_{t+1} = \varepsilon' | \varepsilon_t = \varepsilon],\end{aligned}$$

where $b_{t+1}^{-1}(\cdot)$ *is the inverse function of saving* $b'(b, \varepsilon)$*, and we have suppressed the dependence of* j *for simplicity;*

(v) Government budget (8) is balanced;

(vi) Prices $\boldsymbol{p}$ *clear all markets:*

- *Urban Labor Market:*
$$\mu^u \int \varepsilon^u h^u d\Gamma^u(b^u, \varepsilon^u) = h^m.$$
- *Rural Labor Market:*
$$\mu^r \int h^r d\Gamma^r(b^r, \varepsilon^r) = \mu^f (h^a + h^*).$$
- *Capital Market:*
$$\mu^u \int b^{u'} d\Gamma^u(b^u, \varepsilon^u) + \mu^r \int b^{r'} d\Gamma^r(b^r, \varepsilon^r) = k^m.$$
- *Food:*
$$C^a = \mu^r \int z^a \varepsilon^r (1 - h^r)^{1-\alpha^a} d\Gamma^r(b^r, \varepsilon^r) + \mu^f z^a (h^a)^{1-\alpha^a}.$$
- *Services:*
$$C^s = \mu^u \int z^s (1 - h^u)^{1-\alpha^s} d\Gamma^u(b^u, \varepsilon^u).$$

- *Manufacturing Goods:*

$$C^m + \delta(k^m + \mu^f k^f) + G = z^m (k^m)^{\alpha^m} (h^m)^{1-\alpha^m} + \mu^f R^*,$$

where

$$R^* = z^* (k^f)^{\alpha_1^*} (h^*)^{\alpha_2^*},$$

is the revenue from the export sector.

Recall that we have assumed that manufacturing goods are the numeraire in this economy, meaning $p^m = 1$. Therefore out of the six markets clearing conditions, only five of them are independent, since all resource and budget constraints hold with equality automatically leads to manufacturing market clearing (the Walras' Law). In practice, we solve endogenously the prices for the first five markets. In addition, by including the R^* in the manufacturing goods market clearing condition, we have substituted in the balanced current account condition where the government imports manufacturing goods to clear trade surplus.

2.3 *Understanding Calibration*

In this section, we explain the meaning of calibration from a methodological perspective. To do this, we first introduce the solution concept of the model. We then move on to the definition of calibration. If the user is familiar with connecting a general equilibrium model to the data, this section can be safely skipped.

Solution Concept.—The solution of the model is a characterization of the elements specified in Definition 1. That is, the market clearing prices and the allocations and decisions at the prices. Naturally, when the values of the parameters change, the prices and the allocations will change as well. Importantly, we can calculate many statistics from the model that have counterparts in the data, for instance the share of food, manufacturing goods, and services in consumption and production, Gini coefficients, etc. This allows us to use the model as a "measuring device," to measure the effects of policy reforms that can be modeled parsimoniously as changes in model parameters by comparing the allocations of the equilibria before and after the changes. One prominent example is the one that increases VAT by 4 percentage points that we consider in Section 4. Because the only difference between the two specifications is the value of the parameter that you alter, all the changes predicted by the model must be its consequence. This way, we isolate the impact of a policy reform with the aid of the model.

For this purpose, it is important that we work with a solution that replicates salient features of the actual economy. The process of searching for such a parameterization is what we call *calibration*. Often this is done by comparing the model implied values of certain economic moments along several key dimensions with those found in the data. For example, if the user would like to use the model to study how a policy would change the income Gini coefficient, the model should be able to generate a Gini coefficient comparable to that in the data as the starting point. As an example, assume that when $\sigma_u = 0.1$, the urban income Gini implied by the model is 0.25, while if $\sigma_u = 0.2$, it is 0.35. If the actual income Gini in the data is 0.33, we would want to set σ_u to 0.25 as opposed to 0.1. In particular, the parameterization of the model that replicates stylized facts of the economy under the current policy is usually called the *benchmark*. In most cases, it is

desirable that our model matches the data in more than one dimensions. However, in practice, we cannot match the moments perfectly all at once. As a result, usually we choose a parameterization that delivers an overall reasonable fit on all moments. Put differently, calibration is to choose a parameterization of the model, such that the distance between the model implied moments and their data counterparts is minimized. Mathematically, it is a multi-dimensional optimization problem.[10]

Calibration as a Minimization Problem.—It is usually very challenging to find a good calibration, because the minimization problem characterizing the calibration process is highly non-linear. We illustrate that why this is the case in this section. Throughout, we assume that all vectors are column vectors.

Mathematically, a macroeconomic model can be viewed as a *non-linear operator* that maps a group of parameters to values of some moments. Let the spaces of parameters and moments to be $\mathcal{P}$ and $\mathcal{S}$ respectively. The model is thus an operator $\mathcal{F} : \mathcal{P} \to \mathcal{S}$. Under this notation, for a vector $\boldsymbol{p} \in \mathcal{P}$, from the solution of the model, we can calculate the values of a list of moments $\mathcal{F}(\boldsymbol{p}) = \boldsymbol{s}$. Anticipating the discussion in Section 4 and Table 1, in the context of the current paper, $\boldsymbol{p}$ is an ten-dimensional vector $(\psi, \gamma \cdots, z^*)'$, and $\boldsymbol{s}$ is another ten-dimensional vector where s_1 is the model implied service expenditure share in total consumption and s_{10} is the model implied share of exports in total output. With the default parameterization provided by the toolkit, we have $\boldsymbol{p} = (0.4945, 0.8168, \cdots, 0.6845)'$, and $\boldsymbol{s} = (0.2227, 0.3489, \cdots, 0.1678)'$. In addition, for every moment represented by $\boldsymbol{s}$, we can find their data counterparts, which we denote by vector $\boldsymbol{d}$. In the context of the current model, d_1 and d_{10} are the service expenditure share by households and the share of exports in total output found in the data. In our example, $\boldsymbol{d} = (0.2100, 0.3300, \cdots, 0.0830)'$. With the notations defined so far, the mathematical problem of calibration is the following quadratic optimization problem.

Definition 2. (Calibration) *Let $\mathcal{P}$ be the space of all feasible parameter values, and $\mathcal{S}$ be that of all possible values of the moments in interest. A model defines an operator $\mathcal{F} : \mathcal{P} \to \mathcal{S}$. If we let the data counterparts of moments in $\mathcal{S}$ be $\mathbf{d}$, then calibration is a quadratic minimization problem*

$$\boldsymbol{p}^* = \underset{\boldsymbol{p} \in \mathcal{P}}{\operatorname{argmin}} \left\{ [\mathcal{F}(\boldsymbol{p}) - \boldsymbol{d}]' \boldsymbol{W} [\mathcal{F}(\boldsymbol{p}) - \boldsymbol{d}] \right\}, \tag{9}$$

where $\boldsymbol{W}$ is a given weighting matrix.

If we assume $\mathcal{P} = \mathbb{R}^n$, then (9) is simply an unconstrained optimization problem. If we let

$$\mathcal{G}(\boldsymbol{p}) \triangleq [\mathcal{F}(\boldsymbol{p}) - \boldsymbol{d}]' \boldsymbol{W} [\mathcal{F}(\boldsymbol{p}) - \boldsymbol{d}],$$

standard results from mathematical analysis establishes that $\boldsymbol{p}^*$ is a *local* minimizer to (9), if and only if

$$\nabla \mathcal{G}(\boldsymbol{p}^*) = \mathbf{0},$$

and the Hessian Matrix of $\mathcal{G}$ at $\boldsymbol{p}^*$, $D^2\mathcal{G}(p^*)$ is positive definite.

[10] For a formal elaboration of the calibration process, please refer to the handbook chapter by Dawkins, Srinivasan and Whalley (2001).

Minimization problem (9) is mathematically challenging. Besides the complication that first and second order conditions only guarantee local solution, even searching for such a local solution proves to be a daunting task. Specifically, although $\mathcal{G}(\boldsymbol{p})$ is quadratic in $\mathcal{F}(\boldsymbol{p})$, in most cases it is *not* quadratic in $\boldsymbol{p}$. Take the scalar case as an example. When p is a scalar, minimization problem (9) becomes

$$p^* = \underset{p \in \mathcal{P}}{\operatorname{argmin}}\, \mathcal{G}(p),$$

where in practice, the first and second order conditions

$$\frac{d\mathcal{G}(p^*)}{dp} = 0, \quad \frac{d^2\mathcal{G}(p^*)}{dp^2} > 0,$$

usually do *not* hold.

3 Getting Started

In this section, we provide an intuitive introduction on what the users should be expecting to do to use the toolkit in practice. We start with an overview of the general procedure to conduct counterfactual analyses. We then go over the file structure of the toolkit. In the end, we introduce the layout of the interface and explain how each component is mapped to the operations in the first subsection.

3.1 *General Guideline for Quantitative Exercises*

Generally speaking, to use a model quantitatively for policy analyses usually involve the following steps. Throughout, for the ease of exposition, we focus on an increase of VAT of 4 percentage points.

1. Calibrate the model to the economy and compute the benchmark results.
2. Increase the tax rate in the model by 4 percentage points, and solve for the new equilibrium.
3. Analyze the long-run effects of the reform by performing steady state comparison of the results from the two equilibria.
4. Analyze the short-run effects of the reform by computing the full transition path between the two equilibria.

The toolkit provides two main interfaces to aid the user to finish these tasks. The first interface `toolkit_ss` solves the steady state equilibrium, and the second one `toolkit_trans` solves the transition path between two equilibria. Because we calibrate the model by comparing the moments in the steady state to the data, `toolkit_ss` is used both to calibrate the model and to compute the steady state equilibrium. Depending on whether or not the users would like to investigate what happens during the transition, `toolkit_trans` may or may not be needed. But to use `toolkit_trans`, it is necessary to first compute the two steady state equilibria, and hence `toolkit_ss` is always used.

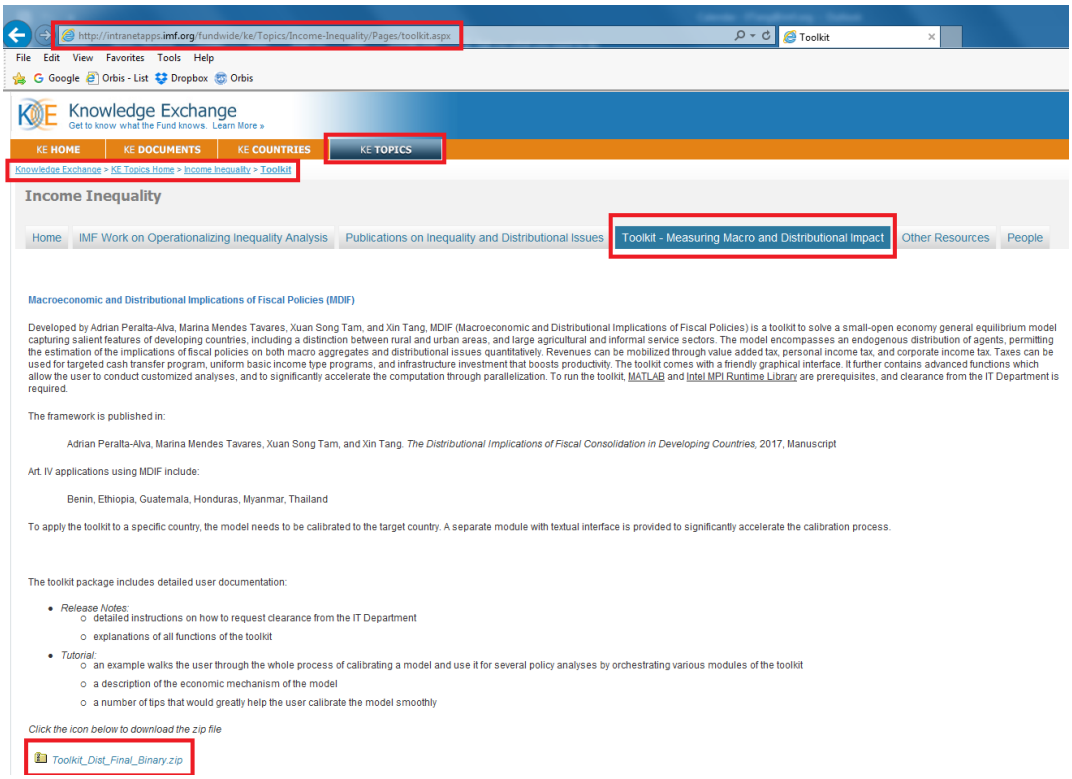

FIGURE 1.—OBTAIN THE TOOLKIT FROM THE FUND'S KE SITE

For the actual analyses, each interface allows the user to export both the micro and macro allocations computed by the model to Microsoft Excel `.xlsx` format. Both interfaces also provide built-in feature to compute the welfare decomposition measured in consumption equivalence.

As explained in the previous section, mathematically, calibration is a complicated minimization problem. Its particular structure invalidates almost all minimization routines and nonlinear solvers that have been developed so far. As a result, calibration of the model has to be done largely by *manually* changing the parameters. Generally speaking, the user usually starts from a known solution and gradually changes the parameter values of the model. For each new parameterization, the user needs to use `toolkit_ss` to solve for the equilibrium, and then to compare the model implied moments with those in the data until a good fit is found. As a result, it is crucial to be able to find a good starting point. For this purpose, we also provide the user a set of MATLAB routines to quickly find such a parameterization.

3.2 *List of Files*

The toolkit can be downloaded as a zip package from the Knowledge Exchange website of the Fund's intranet at (Figure 1)[11]:

[11] We are also preparing to distribute and update the toolkit through GitHub.

`http://intranetapps.imf.org/fundwide/ke/Topics/Income-Inequality/Pages/toolkit.aspx`

After downloading, please unzip the file to a desired location on the hard-drive, for instance `D:/Toolkit/`. Throughout the rest of the guide, we will refer to this folder as the `Parent` folder. We also follow the convention by referring the parent folder later as "`./`". Hence, the absolute path `D:/Toolkit/bin/` will be referred to as `./bin/`. Please do not edit *any* file and their location in the folder unless you are advanced user. Now we go over all the files of the toolkit and explain their functions. We use `Blue Computer Modern Typewriter` letter to indicate how the files are named as a mnemonic. Files marked with an asterisk (`*`) are those that the user can interact with. Please note that the file names do *not* contain the asterisk.

Broadly speaking, the toolkit can be divided into two components based on the functions: an interface that takes instructions from the user, and a set of routines that executes instructions. Each component further contains files that solve the steady state equilibrium, and those that solve the transition path.

- `./`: The parent folder contains files that interact with the user.

  Steady State:

  - `*toolkit_ss.m`: This is the main interface of the toolkit for solving the steady state. This is the core of the toolkit. Executing this file launches the UI shown in the upper panel of Figure 2.
  - `Exp_Excel_SS.m`: This file reads data from one steady state, and exports the data to Microsoft Excel `.xlsx` format.
  - `Welf_SS.m`: This file reads data from two steady states, and computes the welfare decomposition using the first steady state as the benchmark.

  Transitional Dynamics:

  - `*toolkit_trans.m`: This is the main interface of the toolkit for solving the transition path. Executing this file launches the UI shown in the lower panel of Figure 2.
  - `Exp_Excel_Trans.m`: This file reads data from one transition path, and exports the data to Microsoft Excel `.xlsx` format.
  - `Welf_Trans.m`: This file reads data from one transition path, and computes the welfare decomposition with the transitional dynamics considered.

- `./Homogenous/`: This folder contains MATLAB routines assisting the user to find the initial guess for the calibration. The files solve and calibrate the homogenous agents version of the model.

  - `*init.m`: This file initializes value of parameters used in solving the homogenous agents model. Importantly, it also declares a number of variables to be *global* variables, such that they can be shared by all the files below.
  - `*eval_stat.m`: This file solves for the equilibrium prices under current parameterization, and evaluates the data counterparts of the macro statistics implied by the model. This file calls `fh_ss.m`. The equilibrium prices are found using the *Trust Region* algorithm.

  - *cali_z.m: This file calibrates the value of the z's. Specifically, it solves for the value of za, zm and z that minimizes the distance of sectoral share in GDP between the model and the data. The minimizers are solved using *Quasi-Newton* method. This file calls fh_aggr_z.m.
  - *cali_t.m: This file calibrates the value of the tau's. Specifically, it solves for the value of taua, taum and taur that minimizes the distance of total tax revenue and share of different taxes between the model and the data. The minimizers are solved using *Quasi-Newton* method. This file calls fh_aggr_t.m.
  - fh_ss.m: This file is the functional handle used when solving for the equilibrium prices of the steady state. It calculates the allocations for given prices. Files eval_stat.m, fh_aggr_z.m, and fh_aggr_t.m use this file to find the prices that clear all markets.
  - fh_aggr_z.m: This file is the functional handle used when minimizing the distance of productivity related macro aggregates between the model and the data by adjusting the value of z's. This file calls fh_ss.m.
  - fh_aggr_t.m: This file is the functional handle used when minimizing the distance of tax related macro aggregates between the model and the data by adjusting the value of tau's. This file calls fh_ss.m.

- ./bin_ss/: This folder contains the binary executables that do the low-level computation to solve the steady state equilibrium.
  - ss_infra.exe: This file takes the parameter values specified by the GUI and solves for the steady state equilibrium. The parameter values are passed from the GUI as a series of text files. We explain the file structures in more details in Section 5.2. The results are saved as a number of text files in the same folder, and can be exported to Excel format using the GUI.
  - impi.dll: Dynamic-link Library file of Intel MPI Library. Please note that Intel holds the copyright of the file. The file can be freely downloaded from Intel's official website. It is included in the distribution only to simplify the installation procedure on Fund workstation.

- ./bin_trans/: This folder contains the binary executables that do the low-level computation to solve the transition path.
  - trfs_trans.exe: This file takes the parameter values specified by the GUI and solves for the transition path. The parameter values are passed from the GUI as a series of text files. We explain the file structures in more details in Section 5.2. The results are saved as a number of text files in the same folder, and can be exported to Excel format using the GUI.
  - lin_gen.exe: This file generates an initial guess for the path of the prices. Used only for paths with length shorter than 12 periods. We will elaborate in more details in Section 4.
  - impi.dll: Dynamic-link Library file of Intel MPI Library. Please note that Intel holds the copyright of the file. The file can be freely downloaded from Intel's official website. It is included in the distribution only to simplify the installation procedure on Fund workstation.

- `./Documentation/`: This folder contains PDF files for various documents. The PDF documents can be also accessed from within the toolkit through the drop-down `Help` menu. A PDF viewer is required.
  - `Guide.pdf`: This note.
  - `IMF_WP_18_146.pdf`: A copy of the original research article [Peralta-Alva et al. (2019)].
  - `Development_Log.txt`: This file tracks the development log of the toolkit.
- `./Examples/`: This folder contains parameterizations of the examples used in the Guide.
  - `parameters_ss1/2.mat`: Parameters and equilibrium prices of the benchmark and new equilibria. These files can be read using the `Load` function of the toolkit.
  - `VAT_Trans_Xperiods.txt`: Transition paths of `X`-periods. To let the program read the price, please first rename the file to `eprices_out.txt` and copy it to `./bin_trans/`.
- `./Template_SS/`: Text files containing the parameter values read by `infra_ss.exe`. Provided to the user as template for inputting parameter values if MATLAB is not available.
- `./Template_Trans/`: Text files containing the parameter values read by `trfs_trans.exe`. Provided to the user as template for inputting parameter values if MATLAB is not available.

Source code are not distributed for download. Please contact the author for further details.

3.3 *User Interfaces*

In this section, we explain how to use the graphical interface of the toolkit. MATLAB scripts `toolkit_ss.m` and `toolkit_trans.m` are the entry points launch the GUI for steady state and transitional dynamics respectively. Executing the files will launch the GUI as shown in Figure 2. Specifically, `toolkit_ss.m` launches the upper panel, while `toolkit_trans.m` launches the lower one.

There are several ways to execute the files in MATLAB. To invoke the files from the interactive terminal of MATLAB, first set the *working directory* to the parent directory of the toolkit by either executing

```
$ cd d:/toolkit/
```

(assuming that this is the parent folder) in the *command window* panel, or by clicking the circled icon on the top-left of the MATLAB's main UI (upper panel of Figure 3), and select the folder after that. With the parent directory set as the current working directory of MATLAB, you can launch the GUI by executing

```
$ toolkit_ss
```

in the command window. Alternatively, if `.m` files are associated with MATLAB on your operating system, locate the file in Windows Explorer and double click it will launch the MATLAB Editor (lower panel of

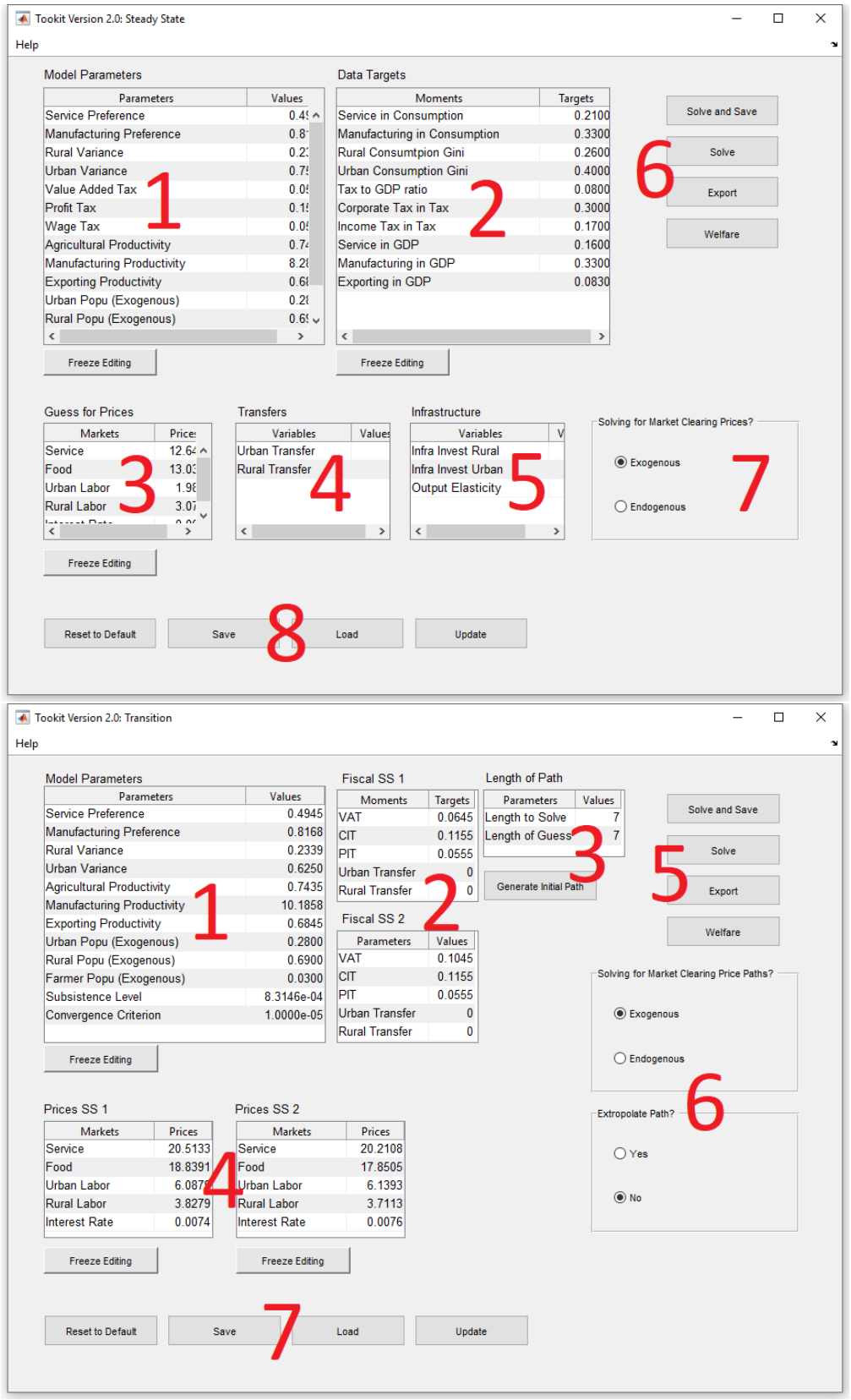

FIGURE 2.—STANDARD USER INTERFACE OF THE TOOLKIT

Upper Panel: Steady State; Lower Panel: Transitional Dynamics.

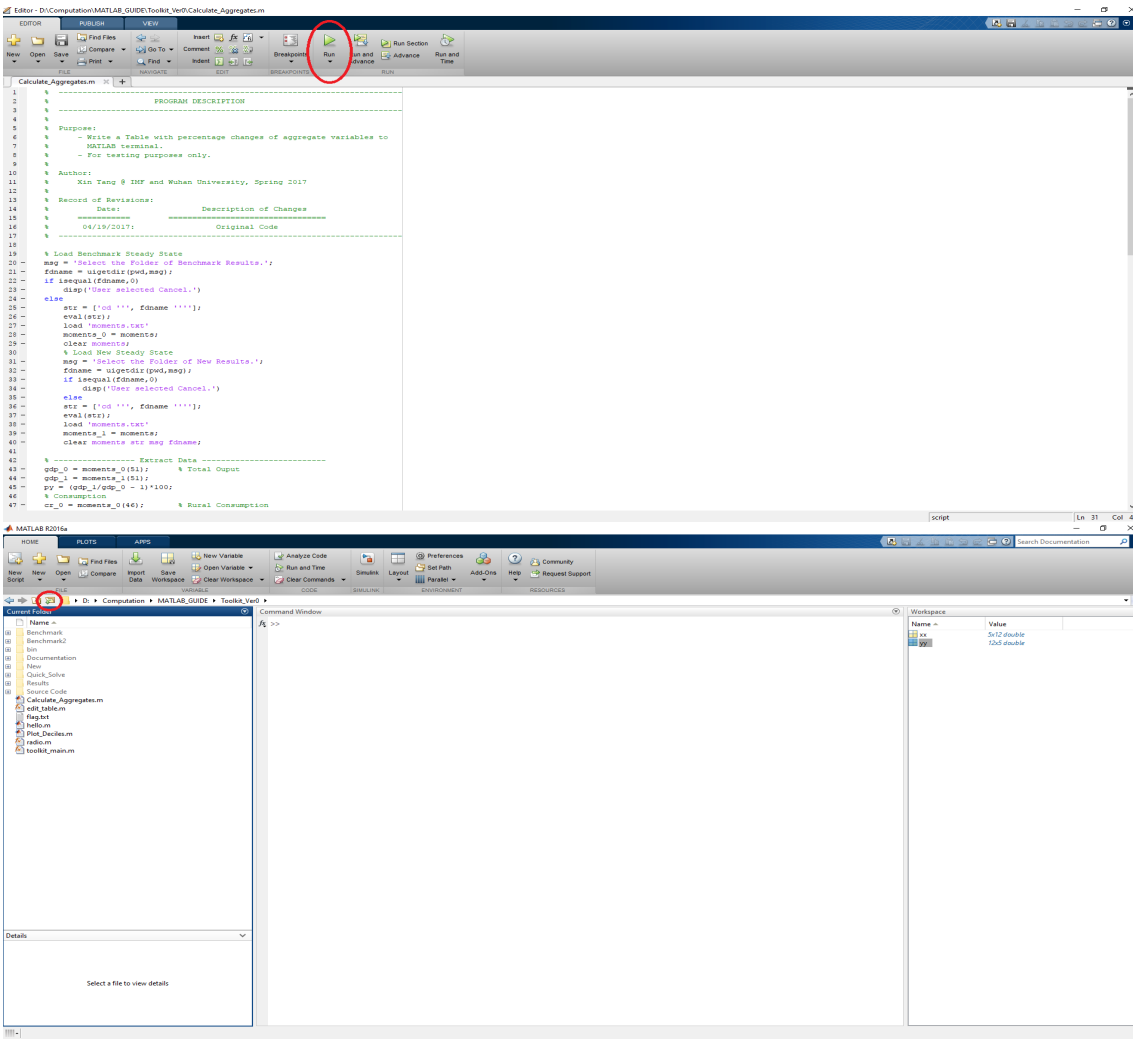

FIGURE 3.—MAIN UI OF MATLAB

Figure 3). Click the circled button `Run` will also launch the GUI of the toolkit. The same operation works for `toolkit_trans.m` as well.

The two interfaces implement similar functions, with `toolkit_ss` for steady state, and `toolkit_trans` for transition. In particular, the user can

1. pass parameter values to the model to solve for the steady state equilibrium or transition path;
2. compare model implied moments in the steady state with those in the data;
3. compute the welfare decompositions both between steady states and with the full transitional dynamics considered;
4. export the simulated data to Microsoft Excel format.

A standard exercise using the toolkit usually takes the following steps. The user should always start with solving the steady states equilibria. The number of the panels refers to `toolkit_ss` (upper panel of Figure 2).

1. Calculate the data moments of the targeting economy. These estimated moments are then used to update the values in Panel 2.

2. Calibrate the model by changing the value of the parameters in Panel 1, such that the model implied moments are considered close enough to those estimated from the data. For each new parameterization, an initial guess of equilibrium price needs to be supplied by the user through Panel 3. At this stage, the files solving the homogenous agents version of the model could be used as a supporting tool to find a initial set of parameter values. Panel 6 issues the command to solve the model.

3. Once the user view the model as suitably calibrated, solve the model under the benchmark calibration and save the results.

4. Use the calibrated model to conduct various counterfactual experiments of different fiscal policies. The user can choose the type of policies that she would like to conduct by changing the values in Panels 4 and 5. Solve the model and save the results.

5. Conduct steady state welfare comparison and export other results for the equilibria in Excel format using Panel 6.

If at this stage, the application requires the user to analyze the transition dynamics, `toolkit_trans` (lower panel of Figure 2) should be used. The number of the panels now refers to `toolkit_trans` (lower panel of Figure 2) as well.

6. Enter the parameter values to Panel 1, fiscal policies and equilibrium prices in the two steady states to Panels 2 and 3.

7. Set a short transition period in Panel 3 and solve for a sample path using Panel 5.

8. Set a longer transition period in Panel 3 and solve for the transition path using Panel 5. In this step, the transition path in Step 7 will automatically be used as the new initial guess.

9. Keep increasing the length of the transition in Panel 3 until the distributions of households converge (more on this in Section 4.3) and save the results.

10. Compute the welfare decompositions and export other results over the transition in Excel format using Panel 5.

TABLE 1
PARAMETERS TO BE CALIBRATED ENDOGENOUSLY

Parameter	Notation	Targets	Data
Service Good Preference	ψ	Service Share in Consumption	0.33
Manufacturing Good Preference	γ	Manufacturing Share in Consumption	0.21
Rural Shocks Variance	σ_r^2	Rural Consumption Gini	0.26
Urban Shocks Variance	σ_u^2	Urban Consumption Gini	0.40
Value Added Tax	τ^a	Tax Revenue as a fraction of GDP	0.08
Profit Tax	τ^r	Corporate Tax Share in Tax Revenue	0.30
Income Tax	τ^w	Income Tax Share in Tax Revenue	0.19
Agricultural Productivity	z^a	Domestic Agricultural Share in GDP	0.34
Manufacturing Productivity	z^m	Domestic Manufacturing Share in GDP	0.38
Exporting Productivity	z^*	Exporting Agricultural Share in GDP	0.10

4 AN EXAMPLE

In this section, we demonstrate how to use the toolkit with a concrete example. The policy scenario we study is a low-income country that considers to strengthen revenue mobilization. The policy maker proposes to collect additional tax revenue by increasing the VAT rate by 4 percentage points. The policy maker would like to simulate the potential welfare costs of the tax increase prior to implementing the reform. Broadly speaking, the welfare costs of the tax reform on the economy as a whole depend on who bears the tax burden and by how much. For this purpose, it is important that our model matches well 1) the relative share of each goods in consumption and production; 2) the distribution of resource across households; and 3) the effective rate of each tax. Loosely speaking, the first two sets of targets pin down who gets taxed, and the last set determines by how much are these people taxed. As a result, we would like to calibrate our economy to the 10 targets of the economy specified in the last two columns of Table 1. We further assume that $\mu^u = 0.28$ and $\mu^r = 0.69$.

Specifically, the 10 targets include the relative consumption and production share of different goods, the tax structure, and regional consumption Gini coefficients. It is important to emphasize that the user can alternatively choose to match the regional income Gini coefficients if consumption Gini coefficients are not available. By default, the toolkit prints the information of the consumption Gini coefficients to the terminal, but income Gini coefficients can be exported using the `Export` feature of the toolkit.[12] There are 10 corresponding parameters in the model that help us to achieve this (the first two columns of Table 1). Though Table 1 seems to suggest that there is a one-to-one relationship between a parameter and a data moment, all the 10 parameters in practice need to be calibrated *jointly*, since changes in one parameter will result in changes of all model implied moments due to general equilibrium effects. However, the parameters matched with the targets are indeed the most responsive ones.

[12]Though the toolkit also exports wealth Gini coefficients, we do not recommend that the user target the model to them. The reasons are first that wealth data are notoriously noisy, even in advanced economy like the United States; and second that the Aiyagari (1994) type model has trouble in matching the wealth distribution in the United States. See Castañeda, Díaz-Giménez and Ríos-Rull (2003) and more recently Hubmer, Krusell and Smith (2018).

In practice, to calibrate 10 parameters together is a daunting task. In the toolkit, we offer a supporting module to keep the calibration process more tractable. The idea of simplification is as follows. First notice that moments in Table 1 can be categorized into two groups, the two distributional moments (rural and urban consumption Gini coefficients), and the remaining macro aggregates. By definition, distributional implications can only be analyzed with the heterogenous agents model. However, a homogenous agents counterpart of the current model also yields implications on the 8 macro aggregates. The corresponding macro aggregates are usually similar in the homogenous and heterogenous agents model.[13] The advantage of the homogenous agents model is that many of the prices and allocations in the homogenous agents version of the model can be characterized *analytically*, which considerably improves the speed and robustness of the nonlinear solvers adopted.[14] As a result, in the supporting module, we provide several MATLAB routines that can be used to calibrate a homogenous version of the model to the macro aggregates fairly quickly. This provides the user a reasonable starting point for the more complicated heterogenous agents model. Starting from the calibrated parameters in the homogenous agents model, the user could first gradually adjust the values of σ_r^2 and σ_u^2 to reach reasonable fit of the model on distributional moments, and then manually do minor adjustments on the parameters to achieve an overall acceptable match.

With the model suitably calibrated, we first change the VAT rate and solve for the new equilibrium. We then compute the welfare costs of the reform by comparing results from the new and benchmark equilibria. After that, we use `toolkit_trans` to solve the transition path between the two steady states, and analyze the short-run dynamics. In the rest of the section, we show step-by-step how to calibrate the model in Section 4.1. Steady state and transitional dynamics analyses follow in Sections 4.2 and 4.3 respectively.

4.1 *Calibrating the Model*

In sum, calibrating the model involves three steps:

1. Calibrate the homogenous agents version of the model to match the macro aggregates.

2. Use the calibrated parameters in step 1 as the starting point to calibrate the heterogenous agents version of the model to match the urban and rural Gini coefficients.

3. Make minor adjustments to the model parameters to further improve the fit of the model.

We now explain how each of the three steps can be implemented with the toolkit.

1. *Homogenous Agents Version Model Calibration.*

   (a) *Introduction of Components of the Toolkit Module.*

[13] Of course, this is not a theory. But we have verified in many exercises that this is the case for our model. See also Aiyagari (1994) and Krusell and Smith (1998).

[14] See Appendix C for details on the solution of the homogenous agents version of the model.

TABLE 2
DATA DICTIONARY OF VARIABLES WHOSE VALUES NEED MANUAL ADJUSTMENTS

Variables	Names	Variables	Names
Food Price	pa	Urban Population	mu_u
Service Price	ps	Rural Population	mu_r
Rural Wages	wf	Large Farmers	mu_f
Manu. Preference	gamma	Agri. Share in Output	yshare_a_data
Serv. Preference	psi	Manu. Share in Output	yshare_m_data
Value Added Tax	taua	Exp. Share in Output	yshare_x_data
Business Tax	taur	Tax/GDP Ratio	tyratio_data
Income Tax	tauw	VAT in Tax	tshare_v_data
Agri. Productivity	za	Income Tax in Tax	tshare_p_data
Manu. Productivity	zm	Business Tax in Tax	tshare_b_data
Exp. Productivity	z		

As is introduced in Section 3.2, there are four interface components for calibrating the homogenous agents version model. `init.m` initializes the preprogrammed parameters and global variables. `eval_stat.m` solves for the equilibrium prices under current parameterization and calculates macro aggregates in the equilibrium. The remaining two *optional* routines, `cali_z.m` and `cali_t.m`, *locally* calibrate the productivity parameters and the tax parameters.

(b) *Initialize all the variables by executing*

```
$ init
```

This routine initializes all the global variables that are used in later computation. The variables include pre-programmed parameters that the users are not supposed to change, initial guesses for market equilibrium prices, parameters that are to be calibrated endogenously, and the value of the targets in the data. A dictionary of the name of the variables and their corresponding is provided in Table 2. After executing `init.m`, you can check the value of any variables in Table 2 by typing the variable names in the command window of MATLAB. For instance,

```
$ yshare_a_data
>
yshare_a_data =

    0.4270
```

shows the data target of agricultural in production. You can change the value of the variable using standard MATLAB assignment statement:

```
$ yshare_a_data = 0.4300
>
```

```
yshare_a_data =

    0.4300
```

At this stage, please change all data targets to those in the data of your particular application. That is, please change the values of the variables with suffix `_data` in the bottom half of the right panel of Table 2 to those found in the data. ONE EXTREMELY IMPORTANT THING IS THAT ANYTIME YOU EXECUTE `init.m` AGAIN, ALL VARIABLES WILL BE SET TO THE VALUES SPECIFIED IN THE FILE!

(c) *Solve for the equilibrium prices and calculate model implied aggregate moments by executing*

```
$ eval_stat
```

The output should look like:

```
> Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

=====================================================================
                  Match of Moments
---------------- Sectoral GDP Share --------------------------------
Moments          Para      Value     Model     Data      Difference
---------------------------------------------------------------------
Agri in Y         za       0.6489    0.3484    0.4270     -0.0786
Manu in Y         zm       8.2390    0.4219    0.3300     0.0919
Exp in Y          z        0.6922    0.0442    0.0830     -0.0388
------------------ Tax Structure ----------------------------------
Value in T       taua      0.0803    0.5522    0.4500     0.1022
Income in T      tauw      0.0589    0.1770    0.1700     0.0070
Corp in T        taur      0.1305    0.2708    0.3800     -0.1092
Tax/GDP          taua      0.0803    0.0920    0.0800     0.0120
=====================================================================
Equilibrium Prices: ps, pa, w, wf, r
   33.5311   37.3463   33.9460    7.3366    0.0417
```

The top panel shows that the `fsolve` routine invoked to solve the equilibrium price has converged. You may encounter different messages in practice that indicate failure of the routine. When this happens, consider use a different initial guess for prices. Among the five prices, by the property of the model, interest rate `r` always equals to $1/\beta - 1 = 0.0417$. Urban wage `w` can also be pinned down given `zm` and `r`. Therefore only `ps, pa` and `wf` need to be solved numerically, and hence require initial guesses. The middle panel shows information about the moments, parameters that pertain most closely to the moments, the current value of the parameters, the value of the moments implied by the model, the data targets, and finally the difference between the model and data. The bottom panel lists the equilibrium prices of the five markets. Notice that because all prices `ps, pa, w, wf, r` are set as global variables, their values will be automatically updated. These values also persist across executions. Hence, if the user does not change their values, these equilibrium prices will serve as the initial guesses of the next time the equilibrium is solved. Due to this automatic updating, please be extra cautious about the states of your initial guesses whenever you are solving for the equilibrium prices. In practice, because there will be cases where the algorithm does not converge but the prices are updated to weird values, it is a good habit to use a text file to keep track of the above table explicitly. In general, when working with the command window, you will find it extremely helpful to keep a log of what are the commands you have issued, and what are the output from the terminal in a separate text file. We recommend that the user do this throughout their whole exercise.

(d) *Gradually change the population share* `mu_u` and `mu_r` *to those of the economy you are studying.* For instance, the initialization provided in `init.m` is `mu_u = 0.28` and `mu_r = 0.69`. If your economy has a `mu_u = 0.40` and `mu_r = 0.57` (assuming a 3% share of large farmers), you may want to start from the equilibrium in Step (b), and solve the equilibria for the cases of `mu_u = 0.32, mu_r = 0.65`, `mu_u = 0.36, mu_r = 0.61`, etc., until you have found the equilibrium with `mu_u = 0.40` and `mu_r = 0.57`. The reason to do this with small step size is that it is very fast to find an equilibrium with a reasonable initial guess, but the model could be unstable if your initial guess is far off the true solution. If you change the population share by too much, it is possible that because the true equilibrium prices have changed by so much, your initial guess fails to converge numerically. In practice, we find that decompose the total change into 3 to 4 steps would work reasonably robust.

We recommend the users to use the Command Window of MATLAB to update the parameters. That is starting from the initial value, execute the pair of statements

```
$ mu_u = mu_u + 0.04;
mu_r = mu_r - 0.04;
eval_stat;
```

repeatedly until `mu_u = 0.40` and `mu_r = 0.57`. In the above example, we have assumed that the share of large farmers are fixed exogenously at `mu_f = 0.03`. Since it is not quantitatively very important how we should map the large farmers to the data as long as the share is small, it is recommended that the user keep its share at 3%. However, if you do wish to change `mu_f`, make

TABLE 3
DATA DICTIONARY OF ALLOCATIONS

Allocations	Variables	Allocations	Variables
Food Demand	`pa*ca`	Food Supply	`ya`
Manu Consumption	`cm`	Manu Supply	`ym`
Serv Demand	`ps*cs`	Serv Supply	`ys`
U.Labor Demand	`hm`	U.Labor Supply	`mu_u*hu`
R.Labor Demand	`mu_f*(ha+hs)`	R.Labor Supply	`mu_r*hr`
Capital Demand	`km`	Saving Supply	`bur`
R.Hired in Food	`ha`	R.Hired in Exp	`hs`
Export	`yx`	Value.Tax	`vtax`
Inc.Tax	`ptax`	Corp.Tax	`btax`

sure that `mu_u + mu_r + mu_f = 1.00`.

In addition, you may encounter the case that if you keep the other parameters unchanged, even for a tiny variation in the population share, you cannot find the market clearing prices. If this is the case, consider changing other parameters to shoot the economy to a different solution region, and come back to the population share after a new equilibrium can again be established. Technically, you can change any of the 8 parameters that need to be calibrated. But our experience is that preference `gamma` and `psi` would most likely set things back to order. It this does not work, consider changing the productivities `za, zm` and `z`. You can also check the demand and supply in each market to investigate the reasons that cause the equilibrium break down. A list of the variables characterizing the equilibrium is summarized in Table 3.

(e) *Change the preferences* `gamma` *and* `psi` *to those characterizing the economy you are interested in.*

If the consumption share of manufacturing and services goods in consumer expenditure are π_m and π_s, then we can show theoretically that[15]

$$\gamma = \frac{\pi_m}{1-\pi_m-\pi_s}, \quad \text{and} \quad \psi = \frac{\pi_s}{1-\pi_m-\pi_s}. \tag{10}$$

As in Step (c), please adjust the values of `gamma` and `psi` gradually toward the values specified in (10). We recommend a step size of 25% the gap of one parameter each time. If this does not work, consider a smaller step size. Our experience up to now is that the model should behave relatively stable along `gamma` and `psi`.

(f) *Calibrate* `za, zm, z` *and calibrate* `taua, tauw, taur`.

We recommend the user adjust the values of these variables *manually* in the same way as the previous parameters. One crucial difference here is that the target value of the parameter is not known a priori. We know we would like a parameter that makes the model behave in a certain way, but we do not know what should be the value itself. For example, consider the results in Step (c), which we attached here again for the convenience of the reader:

[15] Please see the paper for details.

```
>
==========================================================================
                    Match of Moments
---------------- Sectoral GDP Share -------------------------------
Moments          Para      Value     Model     Data      Difference
--------------------------------------------------------------------------
Agri in Y         za       0.6489    0.3484    0.4270     -0.0786
Manu in Y         zm       8.2390    0.4219    0.3300     0.0919
Exp in Y          z        0.6922    0.0442    0.0830     -0.0388
------------------ Tax Structure ----------------------------------
Value in T       taua      0.0803    0.5522    0.4500     0.1022
Income in T      tauw      0.0589    0.1770    0.1700     0.0070
Corp in T        taur      0.1305    0.2708    0.3800     -0.1092
Tax/GDP          taua      0.0803    0.0920    0.0800     0.0120
==========================================================================
```

Consider the parameter `zm` here. The goal is to change `zm` such that the share of manufacturing goods in GDP generated by the model (now 0.4219) is close to the level in the data (now 0.3300). As of now, the model delivers a value that is too large, so we want to *reduce* `zm`, such that the model implies a manufacturing share close to 0.33. The table is presented such that whenever the model delivers a value small than observed in the data, the corresponding parameter should be *increased*, and vice versa. Hence, for the current example, the user should increase `za, z, taur` and decrease `zm, taua` and `tauw`.

Two points warrant some explanation. First, because there is general equilibrium feedback and because that these targets are relative shares (so they always add up to one by construction), changes in one parameter would simultaneously affect the model's fit on other moments. After changing one parameter, the direction for further adjustment will also change. Second, though there are four entries in the "Tax Structure" panel, there are only three tax rates to adjust because by construction, the three tax shares always add up to one, hence with the additional tax to GDP ratio as a target, there are only three independent equations.

In the previous version of the toolkit, we provided with the users two scripts `cali_z.m` and `cali_t.m` that are used to locally calibrate `za, zm, z` and `taua, tauw, taur` automatically respectively. For backward compatibility purpose, we still include them in the current distribution. But their use is deprecated for two reasons. First and foremost is that their robustness is poor and frequently causes numerical failure. In many applications, they actually work worse than the just doing the job manually. Though in general it is very fast to calibrate the homogenous agents version of the model, numerical issues can sometimes be annoying. Second is that the whole purpose of calibrating the homogenous agents version of the model is to provide a initial starting point for calibrating the full model later. The initial guess does not have to be super

accurate. Trying to perfect the calibration for the homogeneous agents version of the model is just wasting time.

But in case the user would still like to try the two functions out, we nonetheless include a short explanation on how they work. Both functions take the current value of the variables as initial guesses, and use Quasi-Newton method to look for a minimizer. Before you call these two scripts, it is HIGHLY RECOMMENDED that you first adjust the parameter values manually such that the model implied moments are reasonably close to those in the data (usually with less than 20% deviation). Take the productivity parameters for example. Before you execute `cali_z.m` in MATLAB, you would want to repeatedly adjust the values of `za, zm` and `z`, and use `eval_stat.m` to solve for the corresponding equilibrium prices and allocations. When you think that the distance between the model and data for `Agri in Y, Manu in Y` and `Exp in Y` in the middle panel reported is relatively small, execute

```
$ cali_z
```

and you should get the results that look like the following:

```
> Calibrated Values: za, zm, z
    0.7545    5.1929    0.6762
Sum of Square of Difference: 0.0070,
```

where the top row shows the values of the variables that minimize the distance, and the bottom row shows the sum of squares of the deviations. Executing `eval_stat.m` again, you would get the following output:

```
> Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

=====================================================================
                  Match of Moments
---------------- Sectoral GDP Share --------------------------------
Moments          Para      Value     Model     Data      Difference
---------------------------------------------------------------------
Agri in Y         za       0.7545    0.3511    0.4270     -0.0759
Manu in Y         zm       5.1929    0.3532    0.3300     0.0232
Exp in Y          z        0.6762    0.1088    0.0830     0.0258
------------------ Tax Structure ----------------------------------
```

```
Value in T      taua      0.0803    0.5468    0.4500     0.0968
Income in T     tauw      0.0589    0.1608    0.1700     -0.0092
Corp in T       taur      0.1305    0.2924    0.3800     -0.0876
Tax/GDP         taua      0.0803    0.0936    0.0800     0.0136
=====================================================================
Equilibrium Prices: ps, pa, w, wf, r
   16.9404   18.3918   16.4097    4.4033    0.0417
```

Compare the above results with the one in step (c), we see that both the values for `za, zm, z` and the equilibrium prices changed significantly (but notice that taxes do not change). The fit of the model on production share is dramatically improved, showing the effect of the minimization routine.

Repeat the process for taxes `taua, tauw` and `taur`, and execute `cali_t.m` and `eval_stat.m`, we get the following results.

```
> Calibrated Values: taua, tauw, taur
    0.0552    0.0545    0.1477
Sum of Square of Difference: 0.0000
```

and

```
> Equation solved at initial point.

fsolve completed because the vector of function values at
the initial point is near zero as measured by the
default value of the function tolerance, and the problem
appears regular as measured by the gradient.

<stopping criteria details>

=====================================================================
                  Match of Moments
---------------- Sectoral GDP Share -------------------------------
Moments         Para      Value     Model     Data      Difference
---------------------------------------------------------------------
Agri in Y        za       0.7545    0.3598    0.4270     -0.0672
Manu in Y        zm       5.1929    0.3469    0.3300     0.0169
Exp in Y         z        0.6762    0.1063    0.0830     0.0233
------------------ Tax Structure ----------------------------------
Value in T      taua      0.0552    0.4500    0.4500     0.0000
```

```
Income in T      tauw      0.0545    0.1700    0.1700     -0.0000
Corp in T        taur      0.1477    0.3800    0.3800      0.0000
Tax/GDP          taua      0.0552    0.0800    0.0800     -0.0000
================================================================
Equilibrium Prices: ps, pa, w, wf, r
   16.9235   18.9061   16.2401    4.4916    0.0417
```

Notice that the model matches the data on taxes extremely well. However, such inch-perfect match rarely happens in practice, and when it does, it usually is more of a sign of problems than achievements. The users should not expect they get a fit of such accuracy in general. We say that such overly good fit is sometimes problematic because it usually indicates that the minimizer is a saddle point, where a slight perturbation on either the calibration targets or the initial guess would lead to non-convergence.

At this stage, we assume that the user is satisfied with the match of the homogenous agents version model under the current parameterization. In what follows, we will use this parameterization as the starting point, and use the GUI of the toolkit to calibrate the heterogenous agents version model. The parameter values we bring to the GUI is as specified in the last table printed to the terminal, with $\gamma = 0.8168$ and $\psi = 0.4945$.

2. *Heterogenous Agents Version Model Calibration.*

   (a) *An Overview of the GUI's Elements.*

   Here we use the GUI of `toolkit_ss` (upper panel of Figure 2). Panels 1 and 2 list the 10 parameters and the corresponding moments in the same order as in Table 1. Notice that we also include the population shares and subsistence level in Panel 1 to allow the users to change their values. The *Subsistence Level* is included for backward compatibility, and is set to a very small number. In the current version of the model, the preference is set to be homothetic. See Section 2.2 and in particular footnote 5 for details. Panel 3 is used to enter the initial guess of the market clearing prices. If the equilibrium price vector has to be solved endogenously, these prices are used as the initial guess provided to the nonlinear solver. The radio button in Panel 7 needs to be set to `Endogenous`. If the prices entered are themselves the equilibrium prices already and the model only needs to be *evaluated* at that price, the radio button in Panel 7 should be set to `Exogenous`. When launched, the GUI is initialized with a set of parameter values, data targets, and equilibrium prices that represent a low-income economy. If we click `Solve` in Panel 6 with all other parts of the toolkit untouched, the toolkit *evaluates* the economy with parameter values in Panel 1 at the *equilibrium price* in Panel 3, and compares the model implied moments with the data in Panel 2.

   (b) *Verify that when the parameter values in step 1.(f) is supplied to the heterogenous agents version model with very small variance in income shocks, the moments calculated from the homogenous and heterogenous agents versions of models are similar.*

To do this, update all parameter values (except the subsistence level) in Panels 1 to 3 of the GUI to the levels of the last calibration in the homogenous agents version model. Notice that in the homogenous version of the model, the variance of the income shocks is zero by definition. In this case, we cannot separately pin down urban and rural consumption C^u and C^r, and urban and rural savings b^u and b^r. Only the aggregate consumption $\mu^u C^u + \mu^r C^r$, and aggregate saving $\mu^u b^u + \mu^r b^r$ can be uniquely determined.[16] For this reason, in the heterogenous agents version model, we cannot set both variances to zero which will crush the algorithm. In practice, we recommend the users to set both variances to 0.10. Larger variance will deviate the equilibrium too much from that of the homogenous version model, thereby causing potential non-convergence issues. On the contrary, too small variance would lead to numerical indeterminacy. If the algorithm fails to converge with variances set to 0.10, try 0.075 or 0.05 instead.

In addition, properties of the standard incomplete markets model [Huggett (1993), Aiyagari (1994)] dictate that the equilibrium interest rate would be lower in incomplete markets, and the elasticity of saving supply to interest rate approaches *infinity* when $r \to 1/\beta - 1$. Therefore, we recommend the user set the initial guess of $r = 0.040$. Consider a smaller value (for instance 0.038) if 0.040 does not work. With all the parameters specified as above, `evaluate` the heterogenous agents version model at the initial guess by keeping the radio button in Panel 7 at `Exogenous` gives us the following results (the numbers may vary slightly).

```
> Market Clearing Conditions:
             Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =     0.0473    0.0294    6.8121   -0.0334   -0.0492
    Price =    16.9235   18.9062   16.2401    4.4917    0.0400

 Model Fit:
Parameter       Moments     Para.Values    Errors    Model     Data
Serv.Pref       Serv.C           0.4945    0.0129   0.2229   0.2100
Manu.Pref       Manu.C           0.8168    0.0189   0.3489   0.3300
Var.Rural       R.Gini.C         0.1001   -0.1287   0.1313   0.2600
Var.Urban       U.Gini.C         0.1000   -0.2895   0.1105   0.4000
Tax.A           TAX/GDP          0.0552   -0.0047   0.0753   0.0800
Tax.Corp        CorpT.T          0.1477    0.1138   0.4138   0.3000
Tax.Inc         IncT.T           0.0545   -0.0053   0.1647   0.1700
Agri.Z          Serv.Y           0.7546   -0.0304   0.1296   0.1600
Manu.Z          Manu.Y           5.1929    0.1409   0.4709   0.3300
Export.Z        Export.Y         0.6762   -0.0006   0.0824   0.0830
Agri.Z          Agri.Y           0.7546   -0.1099   0.3171   0.4270

 Equilibrium Prices:
```

[16] See Appendix C for details.

```
          ps   16.9235000000
          pa   18.9062000000
          w    16.2401000000
          wf    4.4917000000
          r     0.0400000000

 Total Tax Revenue:    1.32163779530559
 Net Tax:              1.32163779530559

Time =   2 mins  48 secs Elapsed
```

We can see from the above table that 1) except for the urban labor market, all other markets are close to clearing at the initial guess found with the homogenous agents version of the model; and 2) all the aggregate moments are numerically similar to those found in the homogenous agents version of the model. We then instruct the toolkit to `solve` for the equilibrium with heterogenous agents by setting the radio button in Panel 7 to `Endogenous`, which yields the following results.

```
> Market Clearing Conditions:
             Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =     0.0000   -0.0000    0.0000    0.0000    0.0000
    Price =    17.3490   17.8509   15.2708    4.1188    0.0375

 Model Fit:
Parameter       Moments     Para.Values    Errors    Model     Data
Serv.Pref       Serv.C           0.4945    0.0128    0.2228    0.2100
Manu.Pref       Manu.C           0.8168    0.0188    0.3488    0.3300
Var.Rural       R.Gini.C         0.1001   -0.1468    0.1132    0.2600
Var.Urban       U.Gini.C         0.1000   -0.3086    0.0914    0.4000
Tax.A           TAX/GDP          0.0552   -0.0015    0.0785    0.0800
Tax.Corp        CorpT.T          0.1477    0.0705    0.3705    0.3000
Tax.Inc         IncT.T           0.0545    0.0023    0.1723    0.1700
Agri.Z          Serv.Y           0.7546    0.0265    0.1865    0.1600
Manu.Z          Manu.Y           5.1929    0.0104    0.3404    0.3300
Export.Z        Export.Y         0.6762    0.0315    0.1145    0.0830
Agri.Z          Agri.Y           0.7546   -0.0684    0.3586    0.4270

 Equilibrium Prices:
          ps   17.3489927726
          pa   17.8508506468
          w    15.2708111665
```

```
          wf     4.1187845929
          r      0.0374741366

 Total Tax Revenue:     1.05021576846292
 Net Tax:               1.05021576846292

Time =  35 mins  14 secs Elapsed
```

It is reassuring to find that the macro aggregates in equilibrium are very close to those in the homogenous agents version model. Also notice that the implied Gini coefficients are small as expected. We would then use this equilibrium as the starting point of the next stage to calibrate the heterogenous agents model.

Two more comments here. First notice that at the market clearing prices of the homogenous agents version model, manufacturing market exhibits huge excess demand, in the sense that firms demand more labor than workers supply. If the excess demand is too large, sometimes the algorithm fails to converge. If this happens, increase the wage `w` slightly or decrease the interest rate `r` slightly to move the economy closer to the true equilibrium. Second, the urban labor market does not clear exactly when the nonlinear solve converges. This is due to the indeterminacy of the model with small heterogeneity. Sometimes using this price as the initial guess and solve for the price again would yield a more accurate equilibrium, since you start from a point that is closer to the true solution than before. But it is not necessary to do so unless the excess demand or supply is very large, since the whole purpose here is to gradually move the economy to one that provides a good approximation to the real world. Starting from the solution of the homogenous agents version model, so long as we are moving alone the right direction, it is not necessary that in all intermediate equilibria, all markets are cleared at very high accuracy. What matters is the accuracy of the final calibration.

(c) *Increase the urban and rural income variances, such that the model implies Gini coefficients similar to those in the data.*

At this stage, you would want to adjust the model fit on consumption inequality.[17] This is done by increasing the variance of the shocks. When adjusting the variance, we still STRONGLY RECOMMEND the users to increase the variance gradually. When testing the toolkit, we use step size as small as 0.025 to let the initial guess to stay around the local converging region of the true equilibrium. For the example used in this notes, we start by increasing both the rural and urban

[17] If the user needs to match the income Gini coefficients, in each execution, the user should first export the results to Excel format. The income Gini coefficients can be found in the file `Macro_Aggregates.xlsx`. We elaborate this more in Section 4.2. However, because in this family of models, consumption Gini coefficients are tightly connected to income Gini coefficients due to consumption smoothing [Kimball (1990) and Carroll (1997)], it is not necessary to export the income Gini coefficients every time. The user could wait until the consumption Gini coefficients more or less reach the level of the income Gini in the data, and only start comparing the income Gini coefficients from the model to the data from then on. Notice for the same reason, it is usually very difficult to match both the consumption and income Gini coefficients at the same time [Domeij and Heathcote (2004)].

variance by 0.025 until the rural variance is 0.2250. This delivers a reasonable level of the rural consumption Gini. We then start increasing *only* the urban variance gradually by a step size of 0.05 until 0.6000. The results are as follows.

```
> Market Clearing Conditions:
              Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =     0.0000   -0.0000    0.0000   -0.0000   -0.0000
    Price =     8.4880    9.7480    1.7707    2.6489    0.0071

Parameter       Moments    Para.Values    Errors    Model     Data
Serv.Pref       Serv.C          0.4945    0.0128    0.2228    0.2100
Manu.Pref       Manu.C          0.8168    0.0188    0.3488    0.3300
Var.Rural       R.Gini          0.2251   -0.0352    0.2248    0.2600
Var.Urban       U.Gini          0.6000   -0.0623    0.3377    0.4000
Tax.A           TAX/GDP         0.0552    0.0007    0.0807    0.0800
Tax.Corp        CorpT.T         0.1477    0.0374    0.4174    0.3800
Tax.Inc         IncT.T          0.0545   -0.0295    0.1405    0.1700
Agri.Z          Serv.Y          0.7546    0.0252    0.1852    0.1600
Manu.Z          Manu.Y          5.1929   -0.1327    0.1973    0.3300
Export.Z        Export.Y        0.6762    0.1786    0.2616    0.0830
Agri.Z          Agri.Y          0.7546   -0.0710    0.3560    0.4270
          ps    8.4880265846
          pa    9.7479847148
          w     1.7707347404
          wf    2.6489487237
          r     0.0071080062
```

We can see from the table above that compared to our initial starting point with very small variance, the fit of the model on consumption inequality improves substantially. The consumption and tax shares show only minor changes. The production shares, however, change significantly. In particular, as we increase the variances of income, the manufacturing share decreases while the export share increases. Therefore, in the last stage of the calibration, we will increase the productivity of the manufacturing sector.

3. *Minor adjustments on the model parameters.*

   (a) *Some Final Adjustments.*

   As shown above, our model is already at an equilibrium that is close to one that we can use for policy analysis. Some minor adjustments suffice to complete the calibration. In this example, we increase the productivity of the manufacturing sector from 5.29 to 10.19, and rural and urban income variances to 0.23 and 0.63 respectively. We also did some minor adjustments on

agricultural and export productivity, as well as tax rates. The direction to change the parameters follows the same "smaller than data then increase" rule as in step 1.(f). The final equilibrium is as follows.

#Benchmark:

```
> Market Clearing Conditions:
             Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =     0.0000    0.0000    0.0000    0.0000   -0.0000
    Price =    20.5133   18.8391    6.0879    3.8279    0.0074

Parameter       Moments     Para.Values    Errors    Model     Data
Serv.Pref       Serv.C           0.4945    0.0144    0.2244    0.2100
Manu.Pref       Manu.C           0.8168    0.0182    0.3482    0.3300
Var.Rural       R.Gini           0.2339   -0.0012    0.2588    0.2600
Var.Urban       U.Gini           0.6250    0.0030    0.4030    0.4000
Tax.A           TAX/GDP          0.0645   -0.0018    0.0782    0.0800
Tax.Corp        CorpT.T          0.1155    0.0000    0.3000    0.3000
Tax.Inc         IncT.T           0.0555    0.0242    0.1942    0.1700
Agri.Z          Serv.Y           0.7435    0.0174    0.1774    0.1600
Manu.Z          Manu.Y          10.1858    0.0587    0.3887    0.3300
Export.Z        Export.Y         0.6845    0.0130    0.0960    0.0830
Agri.Z          Agri.Y           0.7435   -0.0891    0.3379    0.4270
          ps   20.5133438772
          pa   18.8391036004
          w     6.0878980669
          wf    3.8278637784
          r     0.0074103014

 Total Tax Revenue:    1.35075904469646
 Net Tax:              1.35075904469646
 GDP:                  17.2649822726739

Time =   1 mins   1 secs Elapsed
```

(b) *Some Final Remarks.*

First that notice we can certainly further improve the fit of the model along some dimensions. For instance, revising rural income shocks and decreasing urban income shocks could potential match the inequality data inch-perfect. However, the quantitative results is very unlikely to change by much. Therefore, the benefit of looking for a very accurate fit is much less than the cost. As people say, it is important to know "when to stop."

Second, notice that in the production shares, our model over-predicts the export share and under-predicts the domestic agricultural share. Intuitively, it would seem that we should reduce the productivity of the export sector to get a better fit. However, if we do this, we would find that in our model, export share mainly substitutes with that of the manufacturing sector. The reason is that we assume that our model is a small open economy, where trade is balanced by importing manufacturing goods. Hence a decrease in the export share means that the imported manufacturing good also decreases. As a result, the domestic economy has to produce more manufacturing good to meet the demand. On the other hand, the size of the agricultural sector is mainly affected by the preference, and not by domestic agricultural productivity. Changes in z^a mainly changes the price of the domestic food, leaving the production share virtually unchanged. Theoretically, it is because in the model we assume that people do not migrate between rural and urban areas, meaning that the rural population is exogenously assumed to work in the agricultural sector regardless of working on their own plots or for the large farmers (except a small share on cash crops though). As a result, labor supply decisions are inelastic to agricultural productivity level, leaving the production of domestic food in lack of a first order response. But changes in the preference parameters would deteriorate the fit of the model over consumption expenditure shares. The overall lesson that we learn from this is that when working with structural models, it is sometimes very difficult to achieve good match along all dimensions of the model due to some inevitable modeling assumptions. Though undesirable, some trade-off always has to be made. Third, in the bottom panel of the above table, we report `Total Tax Revenues` and the level of non-productive government expenditure (`Net Tax`). The two numbers are the same here because during calibration we have assumed that the government only uses taxes in non-productive way. If for instance that the government uses some of the tax revenues for a cash transfer (as in the example below), `Net Tax` will reflect non-productive use of tax revenues alone and will be strictly less than total tax revenues.

Fourth, the user may notice that due to general equilibrium feedback, values on all moments will change even if you have only changed the value of one parameter. Therefore it is very helpful to compare results from two different equilibria to keep track of the changes caused by varying parameter values. I personally find it helpful to have a text document that saves the results shown in MATLAB command window as a log file.

Now let us assume that you are happy with the fit of the model with the last parameterizations, and would like to save the results as a benchmark case for future references. To do this, first update the equilibrium prices to Panel 3, set radio button in Panel 7 to `Exogenous`, and this time let us click `Solve and Save` in Panel 6. Please be aware that the equilibrium prices will NOT be updated automatically to Panel 3.[18] All else would be the same as when you click `Solve`. But when the computation completes, you will be prompt to choose a folder where you would like to save the results of the model (Figure 4). Due to the way MATLAB interacts with the operating system, please avoid using folder path that contains `space`. Therefore, the only

[18] We choose not to let the toolkit update the equilibrium prices in the GUI automatically because there will be cases that the algorithm fails to converge and the resulting prices be very weird.

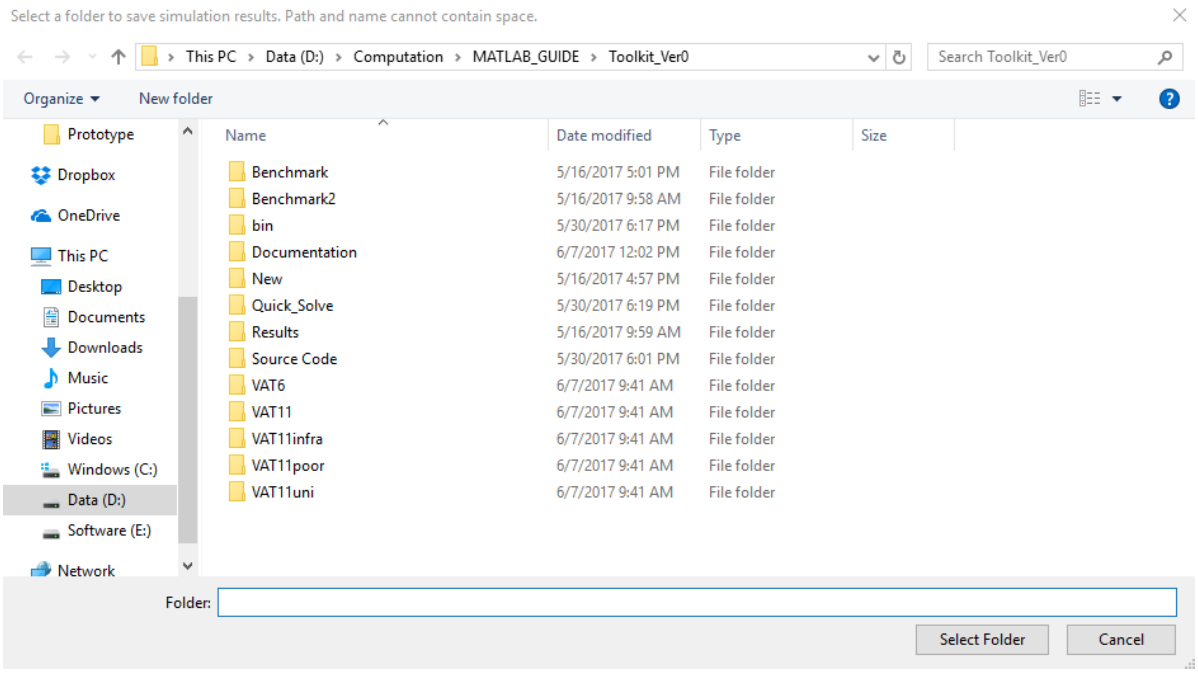

FIGURE 4.—SAVE THE COMPUTATION RESULTS

difference between `Solve and Save` and `Solve` is that for `Solve`, the computation results are automatically saved to the folder `Quick_Solve`, while you are asked to provide a path with `Solve and Save`. In many situations during the calibration, you would simply want to see how the model fits the data, and do not really care too much about the intermediate results. In this case, `Solve` is pretty handy as having to select a folder each time may eventually become annoying if this has to be done several dozens of times. Please be aware that the next time `Solve` is invoked, previous results will be overwritten automatically. Besides the results, you may want to save the current parameterization as well. This can be done by clicking the `Save` in Panel 8. Once clicked, a similar window like Figure 4 will prompt you to enter a folder path *and* a file name. The `.mat` file will save all the data shown on the GUI. Next time you would like to recover some previous parameterizations, simply click `Load`, and a window will prompt you to choose the file you want to load. Please make good use of the `Save` function whenever you feel that you have made some progress in the calibration process. These intermediate states could serve as the starting point if you feel lost, but do not want to go all the way back to the default calibration provided by the toolkit. Button `Reset to Default` resets all parameters in the GUI to the initial value provided by the toolkit. Button `Update` is only used when you want to exploit the parallel design of the code (explained in Section 5.3). This completes our calibration process.

4.2 *Steady State Comparisons*

In this section, we use the calibrated model to study the welfare effects of reform that increases the VAT rate from 6.45% to 10.45% by 4 percentage points. We first explain how the welfare costs are measured and how they are decomposed to an aggregate and a distributional component. We then show how to use the GUI to compute the new equilibrium, do the welfare decomposition, and how the exported results are organized. To ease exposition, we label the benchmark results as `#Benchmark`.

Welfare Decomposition.—We use the consumption equivalence change to measure the welfare costs.[19] We will only go over the intuition in this section, and we encourage the user to check Appendix B of our paper for details in math. In essence, the consumption equivalence change measures how much consumption an average consumer would like to give up permanently in exchange that the policy reform does not happen. If we let the utility of an average consumer in the benchmark and in the new equilibrium to be $V(c_0)$ and $V(c_1)$, where c_0 and c_1 are the consumption level associated to the utility. Then the consumption equivalence λ is such that

$$V((1+\lambda)c_0) = V(c_1),$$

holds. Hence, if $\lambda = -2\%$, then it means that an average consumer would like to permanently give up 2% of the current consumption to prevent the reform from being implemented. We further decompose the welfare costs into an aggregate component and a distributional one following Domeij and Heathcote (2004). This allows us to gauge whether the costs of a fiscal reform are caused by the shrinkage of the size of the economy, so that everyone receives a smaller pie, or they are because from a utilitarian point of view, the resources are distributed unevenly. As an example, if a tax reform reduces output and consumption Gini coefficients at the same time, the decomposition provides a way to measure whether *from the view of a utilitarian government*, the reform overall worths pursing. If we denote the aggregate consumption in the two equilibria to be C_0 and C_1, then the aggregate component λ_a is such that

$$V((1+\lambda_a)c_0) = V\left(\frac{C_1}{C_0}c_0\right),$$

holds. The distributional component λ_d is calculated as the residual by

$$(1+\lambda_d)(1+\lambda_a) = 1+\lambda.$$

VAT Rate Increase.—To do the analysis using the toolkit, first change the value of VAT to 10.45% in Panel 1, set the radio button in Panel 7 to `Endogenous`, and then click `Solve and Save`. The command window of MATLAB will print the following information.

```
> Market Clearing Conditions:
             Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =     0.0000   -0.0000    0.0000    0.0000    0.0000
```

[19] This way of measuring welfare costs can be traced back to exercises to measuring the welfare costs of business cycle by Lucas (1987) and Storesletten, Telmer and Yaron (2001).

```
    Price =    20.2108   17.8505    6.1393    3.7113    0.0076

Parameter       Moments     Para.Values    Errors    Model     Data
Serv.Pref       Serv.C          0.4945    0.0209    0.2309    0.2100
Manu.Pref       Manu.C          0.8168    0.0153    0.3453    0.3300
Var.Rural       R.Gini          0.2339   -0.0031    0.2569    0.2600
Var.Urban       U.Gini          0.6250    0.0059    0.4059    0.4000
Tax.A           VAT.T           0.1045    0.0212    0.1012    0.0800
Tax.Corp        CorpT.T         0.1155   -0.0613    0.2387    0.3000
Tax.Inc         IncT.T          0.0555   -0.0156    0.1544    0.1700
Agri.Z          Serv.Y          0.7435    0.0164    0.1764    0.1600
Manu.Z          Manu.Y         10.1858    0.0703    0.4003    0.3300
Export.Z        Export.Y        0.6845    0.0165    0.0995    0.0830
Agri.Z          Agri.Y          0.7435   -0.1032    0.3238    0.4270
          ps   20.2107926451
          pa   17.8504562279
          w     6.1392647747
          wf    3.7113439115
          r     0.0075659625

 Total Tax Revenue:    1.71979919921361
 Net Tax:              1.71979919921361
 GDP:                  16.9941482435664
```

We find immediately that the total tax revenue increase from 1.35 to 1.72 as a result of the increase in tax rate. This result indicates that the economy is on the left half of the Laffer curve.

To do the welfare decomposition, click `Welfare` in Panel 6, and a window like Figure 4 will pop up to prompt you select the folders where you have saved the results to compare. You will be asked twice to select folders, the first time you choose the one containing the benchmark results, and the second time you select the one hosting the new equilibrium. After this, the following table will be printed to MATLAB terminal:

```
>
============== % Change of Aggregate Variables ================
                Rural             Urban              Whole
---------------------------------------------------------------
Total          -5.0806           -0.6704            -3.7154
Aggregate      -5.2584           -0.2855            -2.6081
Distributional  0.1877           -0.3860            -1.1369
===============================================================
Welfare Decomposition exported to the New Steady State folder.
```

```
File Name: welf_decomp.xlsx
```

An Excel spreadsheet welf_decomp.xlsx containing the able table will also be created under *the folder of the new steady state*.

The welfare decomposition shows that for the economy as a whole, a fiscal reform of increasing VAT rate by 4 percentage points cause welfare costs equivalent to a 4.6% permanent decrease in consumption. From a utilitarian point of view, 3.8% is due to the reduction in the size of the economy, while 0.8% is due to a less egalitarian distribution of resource. In addition, we find that the costs are much larger (3.5 times) to the rural population, suggesting a very unevenly distribution of tax incidence across regions for VAT.

Besides welfare costs, the model generates a wealth of macroeconomic and distributional predictions. The toolkit allows the user to export all these results to Microsoft Excel format that can be readily analyzed. To do this, the user should use the Export button in Panel 6. Upon clicking, you will be prompt to choose the folder containing the results that you would like to export. The exporting process will take several seconds, and meanwhile, the Command Window of MATLAB will display the following messages:

```
> Macro Aggregates Exported!
Urban Distribution on Saving Exported!
Rural Distribution on Saving Exported!
Results Export Complete!
```

The same folder will be the destination folder of the exported files. Five spreadsheet files will be created in the folder:

- Macro_Aggregates.xlsx
- Rural_Distribution_Consumption.xlsx
- Rural_Distribution_Saving.xlsx
- Urban_Distribution_Consumption.xlsx
- Urban_Distribution_Saving.xlsx

Spreadsheet Macro_Aggregates.xlsx contains macro aggregates of the equilibrium. The content inside the file should be straight forward to read, and in the interest of space, we omit the discussion here. The other four files contain the actual simulated cross-section of households from which distributional features of the model are calculated. The rural and urban files share the same structure, and hence we explain the urban files as an example.

Recall that in Section 2.2, we show that in the steady state equilibrium, there is a stationary distribution of households characterized by the joint distribution $\Gamma(b^u, \varepsilon^u)$ of savings b^u and idiosyncratic shocks ε^u. In our numerical solution, we use 10,001 nodes to discretize the saving's space and 15 states to approximate the idiosyncratic shocks. Hence, if we denote the savings node to be $[b_1, \cdots, b_{10001}] \triangleq \mathcal{B}$ and shocks node $[\varepsilon_1, \cdots, \varepsilon_{15} \triangleq \mathcal{E}]$, simulated households in the model are uniquely identified by the pair (b_i, ε_j). In Urban_Distribution_Saving.xlsx, we report the probability density function (PDF) of $\Gamma(b^u, \varepsilon^u)$.

The PDF will contain $10,001 \times 15 = 150,015$ nodes. Since households are identified up to (b_i, ε_j), all households with the same (b_i, ε_j) behave in exactly the same way, and hence are considered identical in the model. The density of (b_i, ε_j) can then be interpreted as number of people with state (b_i, ε_j). Therefore, Urban_Distribution_Saving.xlsx could simply be thought of as a cross-section data with sampling weights. A snippet of the data is as follows.

ID	Saving	Income Shock	Measure	...	Manu. Cons
31	5.7034	1	0.000047	...	0.5261
32	5.9000	1	0.000046	...	0.5367

The data indicate that household of ID 31 has saving 5.7034 and the lowest income shock. There is measure 0.000047 of such households, and each of them spends 0.5261 on manufacturing goods. Due to the space limitation, we omit several variables in the above code snippet. The actual data contain in addition information on food and service consumption expenditures, as well as fraction of hours supplied to the formal labor market. The households are ordered first by savings and then by shocks. This means the 150,015 households are ranked as first 10,001 households with ε_1, with savings increase from b_1 to b_{10001}, and then another 10,001 households with ε_2, until the last 10,001 households with ε_{15}. The measure of all households in each spreadsheet sums up to one.

Similarly, a data snippet of Urban_Distribution_Consumption is as follows.

ID	Consumption	Measure	Total Income	...	Interest Income
1299	25.00	0.000169	25.87476	...	0.328283
1300	25.02	0.000168	25.53988	...	0.330374

The results are interpreted in a similar fashion as before. For instance, sample ID 1299 has spends 25.00 in consumption. There is a measure of 0.000169 of such households, each of them receives 25.87 as total income, of which 0.328 is from interest. The actual data contain additional information on total saving, non-wage income and wage income for each household.

The user can thus use the above data to do a fruitful of analyses. For instance, the user could calculate the 90-10 ratio of the consumption distribution instead of just the Gini coefficient. The user could study the change in the whole distribution of consumption, savings, and wage income for people with different shocks. The user could also calculate the consumption level at the 10th-percentile of a benchmark steady state, and calculate how many households in a new steady state have consumption level higher than that. Here we refrain from attempting to enumerate all possible analyses and leave the user to find their own creative ways to use the simulated data. Just be aware that if you want to construct a distribution of the *total* population, remember to always multiply the corresponding urban/rural measure by their relative size in population.

Modeling Public Expenditure.—Besides tax reforms, the toolkit also allows the user to study the impacts of regional cash transfer program and regional infrastructure investment. The policy parameters are entered

respectively in Panels 4 and 5. In particular, in Panel 4, `Urban Transfer` is a uniform cash transfer to every urban household, and `Rural Transfer` is a similar transfer to all rural households. Setting the two values equal lead to a policy of *universal basic income*. Infrastructure investment in Panel 5 warrants some explanation. We assume that public investment in the rural area increases the productivity for domestic food and cash crops production, while that in the urban area elevates the productivity of the manufacturing sector. Specifically, suppose that in Panel 5, we set `Infra Invest Rural` to k_g^r, and `Output Elasticity` to α^g, then this means that

$$z^x = z^x[\exp(k_g^r)]^{\alpha^g}, \quad x = \{a, *\}. \tag{11}$$

Therefore, if $k_g^r = 0.18$ and $\alpha^g = 0.1$, agricultural and cash crops productivity are both boosted by $[\exp(0.18)]^{0.1} = 1.82\%$. The value of `Output Elasticity` controls how effective is public investment in a country, and is to some extent calls for discretionary decision of the user. An approach I usually take, is to first calculate k_g^r as a percentage of GDP, say 2%. Then I consult with the desk economists or the authority to have an idea of how much of an increase in productivity do they find reasonable with such an investment. I then back out α_g from Equation (11). These policies can be used both separately and in tandem with each other.

4.3 *Transitional Dynamics*

In this section, we show how to compute the transition path between the benchmark and new equilibria. In this section, we will use the GUI `toolkit_trans`, and the number of the panel now refers to this GUI as well (lower panel of Figure 2). Before working with the GUI, we first introduce the solution concept of solving for a transition path. Again, we will not go over the math in great details, and we take the liberty to direct the user to Appendices A.3 and A.4 of the paper for a formal treatment.

Solution Concept.—We assume that the model is perfect foresight along the transition path [Ríos-Rull (1999) and Domeij and Heathcote (2004)], meaning that all economic agents predict correctly the evolution of all prices and policy variables along the transition path. Assuming that the transition takes T periods, we will explain how T is determined shortly after. In period 1, the economy is at the old steady state and starting from period T the economy is at the new steady state. We assume that all agents hold the anticipation that all policies stay at the benchmark level until $t = 1$ (included). At $t = 2$, the policies change to the new equilibrium level as a surprise and remain at the new level forever. Starting at $t = 2$, all agents in the economy correctly anticipates that they are in the new policy regime from then on. More importantly, they correctly predict the pathes of the actual market clearing prices.

Theoretically, the transition from the old steady state to the new one is an *asymptotical* process, meaning that in theory, it takes an infinite period of time reach the new steady state. As a result, computationally, we are always solving for a finite period approximation of the true path. What we want in essence is that the distributions of households $\Gamma^u(b^u, \varepsilon^u)$ and $\Gamma^r(b^r, \varepsilon^r)$ gradually change from those in the first steady state to those in the second. In practice, during the transition, the price will first stabilize at the new equilibrium level, and the distributions take some additional time to converge. Put differently, the length of the transition

is determined in a similar spirit as how a boundary problem of differential equations is solved, with the distributions of households at the new steady state as the boundary condition. Mathematically, we say that we have found an approximation to the true transition path, when the distributions of households at $T-1$ (recall that we have assumed that in period T the economy is at the new steady state) are close to those in the new steady state $\Gamma^u(b^u,\varepsilon^u)$ and $\Gamma^r(b^r,\varepsilon^r)$ by some tolerance level ε_d under certain norm:

$$\max\{\|\Gamma^u_{T-1}(b^u,\varepsilon^u)-\Gamma^u_n(b^u,\varepsilon^u)\|,\|\Gamma^r_{T-1}(b^r,\varepsilon^r)-\Gamma^r_n(b^r,\varepsilon^r)\|\}<\varepsilon_d. \tag{12}$$

The transition length under accuracy ε_d is the smallest T such that (12) holds. Denote this threshold by $T^(\varepsilon_d)$. By the definition of steady state, for an ε_d small enough, any path with $T>T^*(\varepsilon_d)$ will be very similar to a length-$T^*(\varepsilon_d)$ path quantitatively. Hence in practice, we usually solve for a relatively long path. In general, longer path takes more time to solve. However, since most impacts from a policy reform usually mature at the beginning, we can accelerate the solution process by extrapolation. We will explain shortly after how this is done.

Using the Toolkit.—The toolkit at this moment only supports transition path for tax and transfers reform, but not infrastructure investment. There are two reasons for this decision. First is that as we will show below, for low-income countries, the transition is usually fast and hence steady state comparisons provide approximations to the overall welfare costs of a reform. Second is that the numerical algorithm is much less robust with productivity changes. As a result, at least for the purpose of evaluating the welfare costs, the user can use steady state comparison as backup.

Solving the transition path is no easy task, especially the structure of the model dictates that we cannot use the tatonnment algorithm [Auerbach and Kotlikoff (1987)], but have to rely on direct nonlinear solver instead. Consequently, we are in even more dire need of a good initial guess—now five paths of prices—for the solver to work. For this purpose, we highly recommend that the user begin with solving for a short transition path, for instance 5 periods, and later use shorter path as initial guess for longer path. We then gradually extend the length of the transition, until (12) is satisfied. Specifically, the exercise is implemented as follows.

1. *Enter the parameter values for the two steady states.*

   Parameters that are unchanged between the steady states are listed in Panel 1. Except for a new `Convergence Criterion` we will touch upon later, all parameters are the same as before. Policies of the first and second steady states are entered respectively in the upper and lower panels of Panel 2. The variables include the three tax rates and two transfers. The equilibrium prices for the two steady states are entered in the left and right panel of Panel 3. After all these, use the `Save` button in Panel 7 as before to save all the data on the GUI to a `.mat` file. This could save the user a lot of efforts in the future.

   PLEASE BE EXTRA CAUTIOUS TO MAKE SURE THAT THE VALUES ARE ENTERED CORRECTLY, OTHERWISE THE ALGORITHM FAILS ALMOST SURELY.

2. *Solve for a 5 periods transition path.*

(a) Enter the length of transition in Panel 3. There are two entries in Panel 3. `Length to Solve` refers to the length of transition that the user would like to solve for. On the other hand, `Length of Guess` corresponds to the length of the initial guess that the user supplies. Recall that we just mentioned above that we will later use the solution from a shorter transition path to generate an initial guess for a longer path. The distinction of the two entries serve specifically this purpose.

For now, since this is the first time we are solving for any path, so there is not initial guess to work with. We set `Length to Solve` to 7. Because we count the original and new steady states as the first and last periods, so for a length-T path, we always enter $T + 2$.

Because this is the first time we execute the program, there is no previous solution to use as initial guess. We click the `Generate Initial Path` button in Panel 3 to generate a crude guess. Based on our experience, a crude linear guess usually works well for a short transition, but not for longer transition (more than 10 periods). In addition, at one time, there is only one initial path in the solution folder, hence clicking this button will automatically overwrite the current path. Because the transition path is very time-consuming to solve, PLEASE ONLY USE THIS BUTTON THE FIRST TIME YOU SOLVE FOR A PATH and PLEASE THINK TWICE ANYTIME YOU FEEL TEMPTED TO CLICK THAT BUTTON.

(b) Because we are solving for the path endogenously and we are not extrapolating any existing path, for the two radio buttons in Panel 6, we set the one in the upper panel to `Endogenous` and the one in the lower panel to `No`. Click `Solve` and the following message (with some omission) will start being printed to the terminal.

```
>  Transition Period =   7
 Solve for equilibrium price.
  Convergence should start from iteration   27
 Load existing path.
 Tolerance Level =  1.0E-05
 urban dist error nt-1 =   3.658259194448156E-006
 urban dist error nt   =   9.441884687294566E-010
 rural dist error nt-1 =   5.483522699666760E-005
 rural dist error nt   =   9.969929049002957E-010
           1 max. error on fvec   0.251095328158094
    ...
          26 max. error on fvec   0.251095313560150
 urban dist error nt-1 =   2.596068121274745E-005
 urban dist error nt   =   9.441884687294566E-010
 rural dist error nt-1 =   1.748188405784126E-004
 rural dist error nt   =   9.969928771447201E-010
          27 max. error on fvec   8.538100593743536E-002
 urban dist error nt-1 =   3.866142587648572E-005
 urban dist error nt   =   9.441884687294566E-010
```

```
rural dist error nt-1 =   3.700178520354910E-004
rural dist error nt   =   9.969928771447201E-010
   ...
urban dist error nt-1 =   4.687563334501710E-005
urban dist error nt   =   9.441884687294566E-010
rural dist error nt-1 =   4.610464411978973E-004
rural dist error nt   =   9.969928771447201E-010
         31 max. error on fvec   5.641653281429626E-006
Time    31.1979166666667
finished
```

The intermediate results contain vital information to track the status of the program. `Transition Period = 7` indicates that the program is solving for a path with $7 - 2 = 5$ periods of transition. `Solve for equilibrium price` shows that in the current execution, we are solving endogenously for an equilibrium path, as opposed to evaluate the allocations along the transition *at an equilibrium path*. In what follows, the nonlinear solver will keep iterate and update the prices until one where all markets along the transition clear. Recall that in each period, there are 5 prices to solve. With 5 periods between the steady states, we need to solve for $5 \times 5 = 25$ prices. Numerically, this is a nonlinear system of equations consisting 25 unknowns. The MINPACK solver `hybrd` we call will evaluate the gradient along each direction at the initial guess. This takes 1 (Initial Evaluation) + 25 (Compute the Gradient) = 26 iterations. Hence the solver suggests `Convergence should start from iteration 27`. The message `Load existing path` means that the equilibrium path has the same length as the initial guess. `Tolerance Level = 1.0E-05` refers to the criterion of convergence followed by `hybrd`. These information together provide an overview of what the program currently is solving for.

In each iteration, a block of information is sent to the terminal. `urban dist error nt-1` reports $|\Gamma^u_{T-1}(b^u, \varepsilon^u) - \Gamma^u_n(b^u, \varepsilon^u)|$ and `urban dist error nt` evaluates the convergence of $\Gamma^u_n(b^u, \varepsilon^u)$ at the new steady state. The two `rural` entries report the same statistics but for the rural area. `1 max. error on fvec` suggests that for iteration 1, the maximum excess demand in all markets is 0.2511. The information is crucial in diagnosing the status of the solver. `dist error nt` is used to check whether the parameters, policies, and prices passed to the model provide indeed two steady states as end points. If this is the case, the errors should be numbers smaller than the order of 10^{-9}. Here to keep the information succinct, only the new steady state is reported though. Hence the user should be careful to make sure the first steady state is entered correctly. The information provided by `dist error nt-1` is only meaningful when an equilibrium path has been found. At that point, they should be less than the ε_d in Equation (12) desired by the user. In the current example, it is 4.6×10^{-4}, which is much larger than conventional standard.

We also see from the messages that `max. error on fvec` indeed starts decreasing dramatically from the 27th iteration. In just a couple of iterations, it reduces to a level of 5.6×10^{-6} at the

31st iteration, where the solver determines that a solution has been found. Broadly speaking, the solver usually terminates when `error on fvec` (here 5.6×10^{-6}) is less than the `Tolerance` (here 1.0×10^{-5}). However, because `hybrd` uses relative change of `fvec` in two consecutive iterations instead of the absolute value of the last iteration to decide whether the execution should be terminated, the correspondence is not exact.[20] This means sometimes you will see that the iteration continues even when `fvec` is clearly smaller than the `Tolerance`. If this happens, set the `Tolerance Level` in Panel 1 to a larger number and restart the program.

(c) The equilibrium prices will be saved to the text file `eprices_out.txt` under `./bin_trans/`. Recall that we said that the solver always need an initial guess, this very file is the one the program will always look for. When there is no such file under the folder, the button `Generate Initial Path` in Panel 3 creates one. Notice that because the same file serves both as the input and the output of the program, and at one time there is only one copy on the hard-drive, please be extra cautious about what is the content in the file. For this reason, we strongly recommend that if the user would like to try something risky, always create a copy of `eprices_out.txt` in case the solver overwrites a useful version. This is especially important since the transition path takes very long time to solve. As a benchmark, on a desktop with Intel i7-6700 CPU running at 3.4GHz, a path of 5 periods takes 30 minutes to solve when paralleled with 3 cores. On a typical Fund laptop running in serial, it will probably take around 2 hours.

3. *Solve for longer transition path using existing solution as initial guess.*

(a) At this moment, the file `eprices_out.txt` contains an equilibrium path of 5 periods. We would like to extend it to solve for a longer path. We recommend that the user gradually increase the length of path for numerical concerns. For instance, in this example, eventually we want to solve for a 40-periods length transition. We do this by first extend the 5-periods path to 20 periods, and then the 20-periods one to 40 periods.

To do this, we set in Panel 3 `Length of Guess` to `7` and `Length to Solve` to `22`. In Panel 6, keep the upper radio button at `Endogenous` and change the lower radio button to `Yes`. Click `Solve` in Panel 5 again, and the terminal prints the following information.

```
>  Transition Period =  22
 Solve for equilibrium price.
  Convergence should start from iteration  102
 Extend Path Length from   7
 Tolerance Level =  1.0E-05
 urban dist error nt-1 =   3.388630255988568E-006
 urban dist error nt   =   9.441884800051592E-010
 rural dist error nt-1 =   3.008046945415788E-005
 rural dist error nt   =   9.969928771447201E-010
           1 max. error on fvec   7.226445022769212E-002
```

[20] See the documentation of MINPACK for further details.

```
   ...
        101 max. error on fvec   7.226425428683569E-002
 urban dist error nt-1 =   2.017321624202609E-005
 urban dist error nt   =   9.441884800051592E-010
 rural dist error nt-1 =   1.896349924254692E-004
 rural dist error nt   =   9.969929187780835E-010
        102 max. error on fvec   6.792196496895997E-003
   ...
 urban dist error nt-1 =   1.997738972030032E-005
 urban dist error nt   =   9.441884800051592E-010
 rural dist error nt-1 =   1.872940885221733E-004
 rural dist error nt   =   9.969929187780835E-010
        104 max. error on fvec   2.572605907857906E-006
```

The structure of the information is pretty much the same as before. We see that now the convergence starts from iteration 102 $(= 20 \times 5 + 2)$, and the program solves for the equilibrium path by `Extend(ing) Path Length from 7`. We also find that at a longer equilibrium path, Equation (12) holds with a smaller ε_d. With 5 periods, it is $4.6 \times 10^{-}4$, while it is 1.9×10^{-4} now.

(b) Now the file `eprices_out.txt` contains the equilibrium path with 20 periods. We repeat the same process again to extend the path to 40 periods. In particular, we set in Panel 3 `Length of Guess` to 7 and `Length to Solve` to 22. In Panel 6, keep the upper radio button at `Endogenous` and change the lower radio button to `Yes`. Click `Solve` in Panel 5 again, and the terminal prints the following information.

```
> Transition Period =  42
 Solve for equilibrium price.
  Convergence should start from iteration  202
 Extend Path Length from  22
 Tolerance Level =  1.0E-05
 urban dist error nt-1 =   1.300780930791473E-006
 urban dist error nt   =   9.441884669947331E-010
 rural dist error nt-1 =   7.087141197209545E-006
 rural dist error nt   =   9.969927938779932E-010
   ...
 urban dist error nt-1 =   3.416289015533737E-006
 urban dist error nt   =   9.441884669947331E-010
 rural dist error nt-1 =   3.670024996910159E-005
 rural dist error nt   =   9.969929049002957E-010
        204 max. error on fvec   8.444254233097581E-005
 Time    259.572135416667
```

4. *Evaluate the model at equilibrium path to compute allocations.*

   At this moment, the file `eprices_out.txt` contains the equilibrium path with 40 periods. Let us assume that we are happy with it as an approximation to the true path with $\varepsilon_d = 3.7 \times 10^{-5}$. Now we would like to run the program for one last time *at the equilibrium* to save the allocations during the transition to the hard-drive. This last execution is needed because the program only writes results to the hard-drive when the upper radio button of Panel 6 is set to `Exogenous`. This choice is made to accelerate the program since I/O operations with the hard-drive is very time-consuming, and all the results we need are the equilibrium prices for the intermediate steps. Also since we would like to save the results at this execution, instead of click `Solve`, we will use `Solve and Save` in Panel 5 to launch the program.

```
> Transition Period =  42
 Evaluate at Equilibrium Path.
 Load existing path.
 urban dist error nt-1 =   3.390641795792543E-006
 urban dist error nt   =   9.441884687294566E-010
 rural dist error nt-1 =   3.665754255765402E-005
 rural dist error nt   =   9.969929049002957E-010
           1 max. error on fvec   8.448600252108918E-005
 Time    3.06145833333333
 finished
```

   This last execution should be pretty fast though. When the program completes, a window will pop-up asking you to specify the folder to save all the results.

Analyzing the Results.—The same options are available for analyzing the transition path, namely welfare decomposition and exporting other results to Microsoft Excel format. Like before, click the button `Welfare` on Panel 5 computes the welfare decomposition. The following results are printed to the terminal.

```
>
============ % Change of Aggregate Variables ==================
                Rural             Urban             Whole
------------------------------------------------------------
Transition
Total          -5.1944           -0.9471           -3.9874
Aggregate      -4.9630           -0.4960           -2.5813
Distributional -0.2435           -0.4534           -1.4434
------------------------------------------------------------
```

```
Steady State
Total          -5.1670          -0.6794          -3.8929
Aggregate      -5.2583          -0.2856          -2.6082
Distributional  0.0963          -0.3949          -1.3190
============================================================
```

Here both the welfare decompositions from steady state comparisons (lower panel) and those with the full transitional dynamics (upper panel) considered are reported. Notice that they are quantitatively similar, in contrary to usually found in the literature studying tax reforms in the United States [Domeij and Heathcote (2004) and Bakış, Kaymak and Poschke (2015)]. The reason is that low-income countries are typically low in capital stock, the adjustment of this state variable and hence the convergence to the new steady state is fast, leading to similar welfare decompositions. See the paper for more details. Of course, there is no a priori reason that they will always be similar. They authors need to exert some discretion when using the toolkit in real applications. Further, it is reassuring that the steady state welfare decompositions here shows the same results as before.

Button Export has similar function as in the steady state case. When executed, the terminal prints the following information

```
> Please wait until the terminal reports all results exported.
Model Parameters Exported!
Macro Aggregates Exported!
Urban Distribution Exported!
Rural Distribution Exported!
All Results Exported!
```

A number of Excel files are then created in the same folder:

- model_parameters.xlsx
- Macro_Aggregates.xlsx
- Rural_Distribution_NT1.xlsx
- Rural_Distribution_SS1.xlsx
- Rural_Distribution_SS2.xlsx
- Urban_Distribution_NT1.xlsx
- Urban_Distribution_SS1.xlsx
- Urban_Distribution_SS2.xlsx
- Welfare_Distribution.xlsx
- welf_decomposition.xlsx

Most files maintain the same structure as in the steady state case. File model_parameters summarizes the parameter values corresponding to the solution. Macro_Aggregates contains the same information as before, with a time dimension added. The Rural_Distribution_XXs share the same structure as

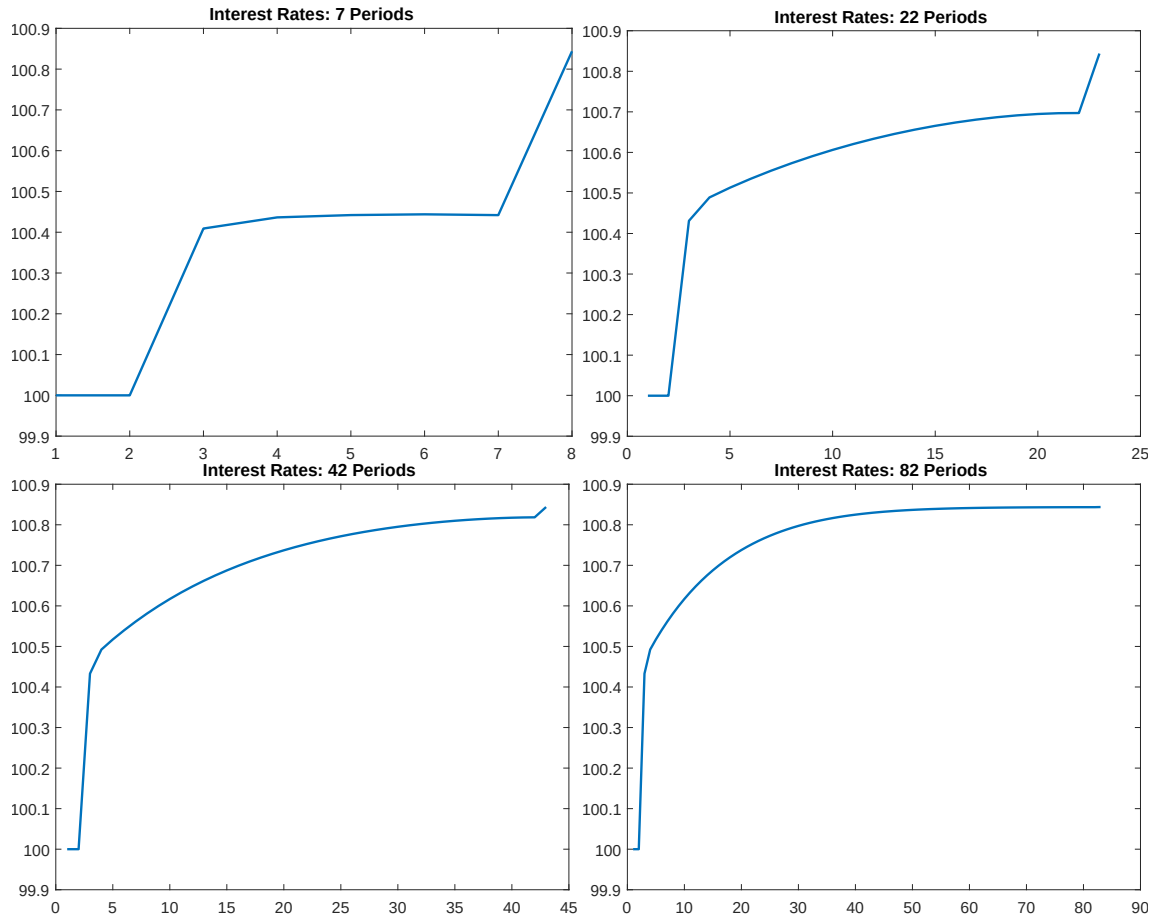

FIGURE 5.—EVOLUTION OF THE PATH FOR INTEREST RATES

`Rural_Distribution_Saving`, but separately for the old (`SS1`) and new steady states (`SS2`) and the first period the reform enacts (`NT1`). The indirect utility at the first period of the reform summarizes the overall impact of the reform to each household with the full transition included. `Welfare_Distribution` summarizes the distributions of indirect utility for all households in the three periods. Importantly, by the definition of steady state, the measure of households in the first steady state and the first period the reform falls on the economy should be the same.[21] As a result, we do not list the probability measure of `NT1` in the file. The user can use the information to compute from a political economy perspective what is the probability that a fiscal reform gets approved and perhaps more important what are the characteristics of the households likely to support the reform by identifying whose indirect utility increases and whose decreases. In the current application, the welfare of all households decrease because taxation always distorts the economy in a standard neoclassical setting. However, the question would be much more interesting for the case of revenue neutral tax reform which reshuffles tax incidence.

Approximating a Very Long Transition Path.—Solving for a long transition path can be extremely time

[21] In practice, there will be small numerical errors though, since the convergence is asymptotical.

consuming even running on cluster. For instance, a 40-periods transition takes about 4.5 hours to solve in parallel on the author's desktop, and an 80-periods one would take as long as 12 hours. On a Fund laptop, in the worst scenario, it could take as much as two days to solve an 80-periods path. This is clearly undesirable. In this paragraph, we provide the user a way to approximate a long path by linear extrapolating an existing solution.

In Figure 5, we show the equilibrium path for interest rates (`r`) for length 7, 22, 42, and 82 (with the two steady states included). While the path for 7 periods differ quite substantially, the rest three paths are quantitatively similar for the bulk of the transition except for the jump in the last period. In fact, the shape of the path is stabilized for the 20-periods solution. Moreover, though the path itself is concave, its tail is more or less linear. This observation suggests that we can construct a long transition path simply by connecting the end of a short path and the new steady state with a long straight line. Notice that this is essentially the extrapolated guess we use the toolkit to construct in the previous step. But instead of using it as an initial guess, now we are treating it as the true equilibrium path. Hence, the only difference here is that we now execute the program with the upper radio button in Panel 6 set to `Exogenous`. Upon execution, the terminal prints the following information:

```
> Transition Period =  82
 Evaluate at Equilibrium Path.
 Extend Path Length from  42
 urban dist error nt-1 =   1.330380007148052E-007
 urban dist error nt   =   9.441884643926479E-010
 rural dist error nt-1 =   2.740405528315129E-006
 rural dist error nt   =   9.969928771447201E-010
          1 max. error on fvec   1.262519793314709E-002
 Time    3.61119791666667
 finished
```

We find that though `fvec` gets larger, the ε_d gets significantly smaller as expected. The welfare decompositions are virtually the same as before again:

```
>
============ % Change of Aggregate Variables ==================
                Rural             Urban              Whole
-------------------------------------------------------------
Transition
Total          -5.2027           -0.9524            -3.9949
Aggregate      -4.9633           -0.4987            -2.5829
Distributional -0.2518           -0.4559            -1.4494
-------------------------------------------------------------
Steady State
```

```
Total           -5.0025          -0.6673          -3.7709
Aggregate       -5.2583          -0.2856          -2.6082
Distributional   0.2700          -0.3828          -1.1938
============================================================
```

As a result, in practice, if the user's application is not time-sensitive, we recommend that the user solve for the path endogenously until around 40-periods, and extrapolate the solution from then on if a longer path is needed. If the exercise is time-sensitive, a 20-periods solution works also reasonably well as a base for extrapolation. However, we do not recommend the user work with a path shorter than 20-periods as it is likely that the shape of the path has yet stabilized. On the other hand, because of the scale of the problem, extrapolate the path by too long could eat up too many memory space and subsequently crush the program on computers with limited memory.

5 Advanced Topics

Up to now, we have covered most topics relevant to putting the toolkit into practice. With the knowledge so far, if everything goes smoothly, the user should be able to craft their own applications using the toolkit. In this section, we provide some additional knowledge. We first introduce several most common causes that the program does not work numerically, and more importantly how to deal with them. We then show how to use the toolkit without the GUI. This would allow the user to continue using the toolkit without MATLAB. Lastly, we provide detailed instructions on how to accelerate the toolkit by paralleling.

5.1 *Exceptional Handling*

Obviously, it would be nearly impossible to enumerate all possible reasons that the program could fail. However, based on our experience, the cases surveyed here are the most likely ones. Despite its complication and a very long execution time, transition paths are pretty robust to solve. It almost always works for transition paths shorter than 20 periods. In cases of numerical failure with longer path, the user can almost always solve the problem by relaxing the `Tolerance Level` or approximate a long path by extending the shorter one. Therefore, all the issues surveyed here relates to solving the steady state. Notice that because solving the model is a challenging numerical exercise, so none of the solutions below is guaranteed to work. The user may inevitably need to resort to their own economic intuitions to sort out the true culprit of a failure.

1. *Unrealistic subsistence level $\overline{a}$.*

   Theoretically, we have assumed $\overline{a} = 0$. For backward compatibility, we still $\overline{a}$ in the program, but set it to a numerically very small number. For this reason, the issue rarely raises. Despite this, in rare occasions, the user may still encounter the error message:

   ```
   > Lower bounds smaller than upper bounds.
   ```

This message originates from the FFSQP routine used in solving the dynamic programming problem. In most cases, this error is caused by a level of the subsistence requirement $\overline{a}$ that is too *high*. Recall that with the Stone-Geary preference, households must first allocate $p^a\overline{a}$ out of their total resource I. Therefore, if $p^a\overline{a} \geq I$, the routine would voice the above complain. In economics, this means that the $\overline{a}$ we choose is too high, such that in the model for some agents, even if they spend all their income, they cannot afford basic food expenditure. As we do not starve people in the model, this case should be ruled out. This problem can be easily solved by setting $\overline{a}$ to a even smaller number.

2. *Undefined excess demand due to negative prices.*

   Sometimes the intermediate results would show something like

```
>
         Service.Y    Agri.Y    Manu.W    Farm.W  Interest
Error =  ************************************************
```

   instead of excess demand on each markets in real number. Any market that has a negative price would trigger this error message. This is usually the sign that you cannot adjust the parameters along certain dimensions any further without changing other parameters. A general principle to deal with this issue is to first find which market(s) cause(s) the problem, and then try change other parameters such that the equilibrium price(s) that(those) market(s) are numerically positive. At the operational level, there are several ways to do it based on our experience.

   (a) *Adjust the equilibrium interest rate by changing the manufacturing productivity.*—Notice that in this model, the equilibrium interest rate is very low (0.74% in our final calibration). As the user further increases the income variance to target a higher Gini coefficient, the precautionary motive would continue to drive the equilibrium interest down, eventually to a level that numerically indistinguishable from zero. In this case, `hybrd` is very likely to shoot the guess for equilibrium interest rate to a negative number. The usual way to deal with this is to increase the manufacturing sector's productivity. By the first order condition of the manufacturing firms, this would increase the demand for capital, which pulls up the equilibrium interest rate. Negative prices in other markets could be treated similarly, we leave to the users to find out what are the parameters that numerically affect the equilibrium prices as they do not happen so often.

   (b) *Ignore the error message temporarily.*—Sometimes this error happens because the objective function of the minimization problem is *flat* along some dimensions, therefore `hybrd` updates the guess for prices overly aggressively. For instance, suppose that the user supplies an initial guess for urban wage of 3.00. It could be that the true equilibrium price is 2.00, but because the gradient is very small around 3.00, the routine aggressively updates the price guess to -0.50, therefore triggering the error. If this is the case, the user could temporarily ignore the error message for several iterations. `hybrd` will ignore the current guess when calculating the new composite gradient. Sometimes the new gradient moves the model to a different region where this issue does

not exist. However, if the error happens repeatedly, consider restart the program from a different initial guess.

3. *Infinite cycling.*

   This is a problem that usually happens when you start from a bad initial guess. It is technically not an "error," since the program will just run on and on indefinitely, and no error message would be triggered. As a result, the user needs to read carefully the intermediate results printed on the terminal to detect an infinite cycle. An example is given below, where for the ease of reference, we have labeled each round of iteration with #.

```
> Start Evaluating the Model.
 Start solving general equilibrium with tol  0.0001
 Start Urban Problem
 Start Rural Problem

#1           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0142    0.0067    0.0000   -0.0000    0.0000
    Price =    11.8696   13.1809    2.0187    3.0894    0.0025
 Start Urban Problem
 Start Rural Problem

#2           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0143    0.0067   -0.0022   -0.0000   -0.0000
    Price =    11.8733   13.1809    2.0187    3.0894    0.0025
 Start Urban Problem
 Start Rural Problem

#3           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0142    0.0066    0.0011   -0.0001   -0.0000
    Price =    11.8696   13.1850    2.0187    3.0894    0.0025
 Start Urban Problem
 Start Rural Problem

#4           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0142    0.0067   -0.0074   -0.0000   -0.0000
    Price =    11.8696   13.1809    2.0193    3.0894    0.0025
 Start Urban Problem
 Start Rural Problem

#5           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
```

```
    Error =    -0.0142    0.0067   -0.0005    0.0002    0.0000
    Price =    11.8696   13.1809    2.0187    3.0903    0.0025
 Start Urban Problem
 Start Rural Problem

#6           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0142    0.0067    0.0004   -0.0000   -0.0000
    Price =    11.8696   13.1809    2.0187    3.0894    0.0025
 Start Urban Problem
 Start Rural Problem

#7           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =     0.0050    0.0002    2.0667   -0.0000    0.0042
    Price =    10.7367   13.3163    2.0639    3.1069    0.0021
 Start Urban Problem
 Start Rural Problem

#8           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0143    0.0062   -0.4324   -0.0000   -0.0009
    Price =    12.0175   13.3348    2.0549    3.1092    0.0025
 Start Urban Problem
 Start Rural Problem

#9           Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0143    0.0067   -0.0022   -0.0000   -0.0000
    Price =    11.8733   13.1809    2.0187    3.0894    0.0025
 Start Urban Problem
 Start Rural Problem

#10          Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Error =    -0.0142    0.0066    0.0011   -0.0001   -0.0000
    Price =    11.8696   13.1850    2.0187    3.0894    0.0025
```

Notice that iteration #10 is exactly the same as #3. If left unattended, the code will just cycle indefinitely. The reason this happens is that under the current parameterization, urban labor market is more responsive to services price than the services market itself. When the algorithm attempts to reach an equilibrium on the service market, the side effect on the urban labor market is too strong, which then triggers the exceptional handling feature of `hybrd` to treat this update as a bad guess and start all over again.[22] At this stage, we feel that it is useful to explain the intermediate results in more details.

[22] Theoretically, this is because the Jacobian matrix of the problem is not diagonal-dominated.

(a) *The General Principle.*—Recall that in Section 4.3, we introduced that the program updates the guess for the solution by computing the gradient. Here it works in exactly the same way. The program starts by evaluating the excess demand at the initial guess (#1). It then vary the price of the five markets one by one to calculate the gradient along each dimension (#2 to #6). The algorithm then calculates the direction of the next guess of the price vector that moves all market excess demands toward zero. Future guesses are updated dynamically based on how the current guess moves the equilibrium.

(b) *Sign of Convergence.*—An ideal sign of convergence is that starting from #7 (recall the $2T + 2$ in the case of transition), the sign of the excess demand alternates between positive and negative, with the *absolute value* decreasing in the process. The first six iterations will usually show only minor changes in the excess demand, since the purpose of these steps is to evaluate the gradient, not to update the actual prices. Notice that as explained previously, if there are negative prices or other signs of "bad guess," the algorithm will drop the guess, return to the latest saved price level, and restart #1 to #6. Occasionally, the intermediate results would seem to suggest that the guess is "drifting away" from the equilibrium. If this happens, the user may want to wait for a couple of additional iterations. If the algorithm is still updating the price guess in a numerically significant way (the significant digit should be at least 10^{-3}), then it is possible that after a couple of iterations, the model will move to a converging region. But if the user captures sign of cycling, terminate the code (by pressing `Ctrl + C` in the Command Window of MATLAB) and start with a different initial guess.

(c) *Direction of Manual Adjustments.*—Due to the local convergence property of the model, in many situations the user needs to supply with the model a reasonable local guess of equilibrium prices. Unfortunately, this cannot be automated and requires discretionary judgement of the user. However, there are some general guidelines that the user could follow in search for a good initial guess. The numbers reported on row `Error` is the excess demand on each market. For instance, in #1, the `-0.0142` of row `Error` under column `Service.Y` means that when the price of service good equals `11.8696`, the supply of service good exceeds the demand by `0.0142`. Hence the current price is higher than the equilibrium price, and a lower guess should be provided. As a short cut, the direction of the adjustment should be the same as the sign of the excess demand, meaning that a negative excess demand indicates the initial guess should be lower, and vice versa. In calculating the actual number of the initial guess, we recommend the user start with a 10% change of the current guess. If this changes the excess demand by too little, consider gradually increase the step size. Notice that you may have to change the guess for multiple markets. For instance, in the above example, initial guess

```
              Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Price =     11.8696   13.1809    2.0187    3.0894    0.0025
```

leads to cycling behavior, while a different guess with

```
              Service.Y    Agri.Y    Manu.W    Farm.W  Interest
    Price =     10.5188   13.6516    2.7099    3.0508    0.0025
```

yields convergence. Most important changes of the new initial guess are service good price and urban wage, which is the particular reason that the algorithm fails.

(d) *Transitional Dynamics.*—If the program fails to converge for the transition path, mostly likely it is because of the same reason. The problem here is that it is much harder to come up with a different initial guess, since here the guess contains five nonlinear curves. Luckily though, the program rarely runs into any issue with path shorter than 20 periods, and in most cases relaxing the `Tolerance Level` solves the problem with very little compromise in computational accuracy. The user could also sidestep the issue by extending an existing short path linearly, if the problem only happens with long path.

5.2 *Using the Toolkit without MATLAB*

As is introduced in Section 3.2, the GUI of the toolkit merely provides an interface to let the user issue instructions easily. The GUI is also the only component of the toolkit that requires MATLAB. Therefore, if the user understands how the GUI interacts with the files in the hard-drive, the toolkit can be well be used without MATLAB. To convert the results to Excel and to compute welfare decomposition, GNU Octave can be used instead.

GNU Octave is software featuring a high-level programming language—primarily intended for numerical computations—that uses syntax that is mostly compatible with MATLAB. Since it is part of the GNU Project, it is free software under the terms of the GNU General Public License.[23] Octave can be downloaded from `https://www.gnu.org/software/octave/`. To export results to Excel, the user needs to first install the `Package IO`. To do this, execute the following command in the terminal of Octave:

```
$ pkg install -forge io
```

Everytime Octave launches, the user needs also to load the package by executing:

```
$ pkg load io
```

Now we will go over the functions of the GUI panels one by one. We start with `toolkit_ss`. For the steady state GUI, all text files are hosted under the folder `./bin_ss/`

- Panel 1: The parameter values of the model are set through Panel 1. To do this, single-click any entry of column `Value`, the background of the cell will turn to blue, and a cursor will prompt you to input the value. If the value entered is *different* from that in the table, the toolkit will update the values specified in `param_matlab.txt`, which later on is read by `ss_infra.exe`.
- Panel 2: This panel summarizes the calibration targets. The targets are changed by inputting values to column `Targets` in the same way as before. To avoid overwrite the values by accident, you can

[23] GNU Octave is one of the major free alternatives to MATLAB. Other alternatives include Scilab and FreeMat. Scilab, however, puts less emphasis on (bidirectional) syntactic compatibility with MATLAB than Octave does.

disable editing by setting the button Freeze Editing to be depressed. This panel updates the values in targets_matlab.txt.

- Panel 3: This panel is used to set values of market prices. The table can be edited as usual, and the values will be passed to ep_matlab.txt.
- Panels 4 and 5: This panel is used to set values of fiscal policies. Files transfer_matlab.txt and infra_matlab.txt are updated respectively by Panels 4 and 5.
- Panel 6: This panel allows the users to issue different instructions to the model. We will go over the functions from top to bottom.
  - Solve and Save: This button executes the program infra_ss.exe and upon completion, copies every text files (.txt) to a folder. Button Solve does the same. The difference is that with Solve and Save, a system window will pop up to prompt you to select a folder to save the results, while with Solve a folder Quick_Solve is automatically created to save all results. Due to the way MATLAB interacts with the operating system, please avoid using space in folder name, when necessary use the underscore _ instead.
  - Export: This button calls the MATLAB routine Exp_Excel_SS.m to transform the simulated data of the model from text format to Microsoft Excel format. When clicked, a system window will pop up to prompt the user to select a folder that contains the results that the user would like to convert.
  - Welfare: This button calls the MATLAB routine Welf_SS.m to compute the welfare decomposition. When clicked, a system window will pop up to prompt the user to select the folders containing results of the benchmark and new steady states accordingly.
- Panel 7: As explained before, you can specify whether you would like to simply evaluate the model again, or you wish the market clearing prices be solved endogenously. This is done by switch the radio button between Exogenous and Endogenous. Parameters in flag.txt will be updated to 1, if the button is switched to Endogenous and 0 otherwise.
- Panel 8: This group of buttons allows you to modify all parameters on the toolkit altogether.
  - Reset to Default: This button resets values of all parameters to the default value of an example economy. This could be helpful if you have made too many changes and lost track of the model, and would like to start all over from the very beginning.
  - Save: This button allows you to save the parameterization of the model as is shown on the GUI of the toolkit to a MATLAB .mat file, which can be loaded later on. When clicked, a system window will pop up and the user can specify both the location and name of the file. For the same reason as specified before, please avoid using space in folder name. In the .mat file, the data are saved as MATLAB *cell array* for saving *derived type data*, a counterpart of *structure* in C/C++.
  - Load: This button loads previously saved parameterizations. When clicked, a system window will pop up and the user can specify both the location and name of the file.

  - `Update`: This button updates all value of parameters shown in the GUI of the toolkit to the respective `.txt` files as specified above. This button is used to pass the parameters to the text files. If you are invoking the computational routine from within the toolkit, the toolkit contains safeguard features that guarantees the parameter values will be passed to these files before hand. However, parameter values have to be passed explicitly to those text files by this button (or update manually if the GUI is not available) if the user wants to execute the program in parallel.

We now turn to `toolkit_trans`. The two GUIs are structured following the same philosophy and hence can be understood more or less in the same way. We thus summarizes the information much succinctly. For the transition GUI, all text files are hosted under the folder `./bin_trans/`

- Panel 1: The panel updates `param_matlab.txt`, which later on is read by `trfs_trans.exe`.
- Panel 2: This panel updates `fiscal1_matlab.txt` and `fiscal2_matlab.txt`.
- Panel 3: The upper panel updates `length_matlab.txt`. The button executes `lin_gen.exe`, which reads `price1_matlab.txt`, `price2_matlab.txt`, and `length_matlab.txt`, and generates the text file `eprices_out.txt` as initial guess for the path.
- Panel 4: This panel updates `price1_matlab.txt` and `price2_matlab.txt`.
- Panel 5: This panel works in the same way as Panel 6 of `toolkit_ss`.
  - `Solve and Save`: This button executes the program `trfs_trans.exe` and upon completion, copies every text files (`.txt`) to a folder. Button `Solve` also does the same here. Again, the difference is that with `Solve and Save`, a system window will pop up to prompt you to select a folder to save the results, while with `Solve` a folder `Quick_Solve` is automatically created to save all results. Due to the way MATLAB interacts with the operating system, please avoid using `space` in folder name, when necessary use the underscore `_` instead.
  - `Export`: This button calls the MATLAB routine `Exp_Excel_Trans.m` to transform the simulated data of the model from text format to Microsoft Excel format. When clicked, a system window will pop up to prompt the user to select a folder that contains the results that the user would like to convert.
  - `Welfare`: This button calls the MATLAB routine `Welf_Trans.m` to compute the welfare decomposition. When clicked, a system window will pop up to prompt the user to select the folders containing results of the benchmark and new steady states accordingly.
- Panel 6: The upper panel updates `endo_flag.txt`, with `Exogenous` set the value in the file to `0` and `Endogenous` to `1`. Similarly, the lower panel updates `extend_flag.txt`, with `Yes` to `1` and `No` to `0`.
- Panel 7: This panel works virtually in the same way as before. One crucial point here is that `Reset to Default` and `Update` will execute `lin_gen.exe` to generate a new `eprices_out.txt`, which could accidentally overwrite existing solutions if used carelessly. The user should be extra cautious here.

To use the toolkit without MATLAB, what the user needs to do is the following.

1. Update the model parameters by changing the values in the text files under the folders `./bin_ss/` and `./bin_trans/`. Templates are provided in `./Template_SS/` and `./Template_Trans/` respectively.
2. Execute `./bin_ss/ss_infra.exe` or `./bin_trans/trfs_trans.exe` from the operating system.
3. Manually copy all `*.txt` files under `./bin_ss/ss_infra.exe` or `./bin_trans/trfs_trans.exe` to desired locations.
4. Use GNU Octave to run `Exp_Excel_SS` to export all the results to Excel format, or `Welf_SS` to compute welfare decompositions. To do this, simply set the working directory of Octave to the `Parent` folder and execute (case sensitive)

   ```
   $ Exp_Excel_SS
   ```

   as in Section 3.3. The corresponding files for transition are `Exp_Excel_Trans` and `Welf_Trans`.

One caveat though is that Octave can be substantially slower compared to MATLAB. I mean, much much slower. But it will work eventually, so please be patient.

5.3 *Execute the Toolkit in Parallel*

This section explains step by step how to exploit the multi-core architecture of modern CPUs to speed-up the execution of the code. This could potentially allow the user to gain performance improvement of more than 50%. The performance of scientific computing relies heavily on CPU clock rate. Impact from other factors like instruction sets or CPU cache size are much less important. Therefore, if performance is what you care a lot about, you may want to set the power plan of your Operating System to *high performance* instead of *balance* (or worse *power saving*) to allow the CPU running on its full clock rate. It may sound like a suggestion from "Captain Obvious," yet surprisingly, it is frequently overlook by many people.

CPUs made of silicon generate too much heat that cannot be cooled in an economic way under current technology when the clock rate approaches 4GHz. Therefore modern CPU manufacturers stop speeding up the processors by continuing increasing the clock rate. Instead, they choose to integrate multiple cores on one processor. As a result, if you are running this toolkit in serial mode, it is unlikely that you get a noticeable performance improvement by simply running the toolkit on a computer equipped with a CPU with more cores. Put it straight, if you want the toolkit to run faster on a more expensive computer, you need to run the code in parallel mode.

Because running an executable file in parallel from within another process (MATLAB) is prohibited by the operating system for thread safety concerns, we are not able to integrate the option of parallel computing into the GUI. Therefore, to use the toolkit in parallel, some command-line operations are necessary. However,

the user does not have to memorize a complicated set of commands. Therefore, even if the user is not comfortable of interacting with the computer through a terminal, the operations would turn out to be quite easy. Essentially what the user needs to do is very similar to using the toolkit without MATLAB. The only difference here is that in Step 2 of the above workflow, instead of executing for instance `./bin_ss/ss_infra.exe` directly, the user should call it by executing `mpiexec`. We explain how to do this now.

Throughout, we use the steady state case as example. You will need to type the commands in blue in the *Command Prompt* of Windows. To run the toolkit in parallel, the user also has to install Intel MPI Runtime Library. The library can be freely downloaded from Intel's official website. If you want to install the library on Fund's workstation, please contact ITD. Also since Intel frequently changes the layout of its website, please make use of search engine for addresses. Once the library is successfully installed, please follow the step below.

1. Press `Windows + R` to bring the *Run* window, type `cmd` in the *Open* text box and press `enter`, or click the press button `OK`. This will launch the Command Prompt of Windows.

2. Verify that Intel MPI Runtime Library has been properly installed.

3. If you are running the Intel MPI on the current Windows account for the first time, you need first to register your current Windows account to the MPI environment. To do this, type and execute

   ```
   $ mpiexec -register
   ```

   You will be prompt to set a user's name and credential. Please enter the *account name* of your *current Windows account* following the on-screen message

   ```
   > account (domain\user)
   ```

   and press enter. You then will be prompt to set the password and confirm your password. Enter your password following

   ```
   > password
   ```

   Please use the same user's name and credentials as your current Windows account. In particular, since the IMF IT policy requires that the password be reset every 3 months, the user needs to update the credentials each time the system's password is reset.[24]

4. Switch to the `./bin_ss/` directory of the toolkit where the binary files are located. By default, `cmd` leaves you at

[24] Essentially, `mpiexec` uses the account name and the password to gain access to the operating system.

```
> C:\Users\Account Name
```

Recall that we have set the `Parent` folder to `D:/Toolkit/`. To switch to that folder, first type and execute `$ d:` to go to the `d` drive. Then type and execute the following command.

```
$ cd toolkit/bin_ss
```

This should bring you to `d:\Toolkit\bin_ss`, where `ss_infra.exe` is located. Notice that Windows Command Prompt commands are *case-insensitive*.

Two other commands that might come into handy at some point. `cd..` takes you back to the previous level, and `cd\` goes all the way back to the root folder of the current partition. Hence if you are at `D:\Toolkit\bin`, `cd ..` takes you to `D:\Toolkit`, and `cd\` changes the directory to the root folder `d:`.

5. Now type and execute

```
$ mpiexec -np 3 ss_infra.exe
```

to execute the binary file to solve the model in parallel. You can change the number of cores to use by changing the `3` following the `-np` (which is short for `n`umber of `p`rocessors). Notice that the computation is parallelized with respect to the `15` idiosyncratic states, please make sure that the number of cores you are using can be divided by 15. Put it straight, only `3`, `5`, and `15` cores would work. Otherwise the process will run into deadlock. Also how many cores you can distribute your work to accelerate the execution depends only on the number of *physical cores* of your CPU, not the number of *threads* that could be implemented. For instance, Intel Hyper-Threading Technology uses processor resources more efficiently to enable multiple threads to run on each core. However, because the code has not been programmed to take advantage of this feature, if you are running a version of the model with 15 states approximating the idiosyncratic shocks, on a CPU with 4 cores and 8 threads, you would not get performance improvement by changing the number of cores from `3` to `5`.

Finally, it is not necessary to type the command every time you run the code as long as you have *not* closed the current Command Prompt window. You can repeat the previous commands by pressing the up-arrow ↑ (which rewinds past commands as in MATLAB) or `F3`.[25]

6. At this stage, the program will compute the model and save the results to the current folder. The user can then work with the results in the same way as before.

[25] Incidentally, Windows Command Prompt uses the same command list as *Microsoft DOS (Disk Operating System)*. If you would like to know more about how to interact with the operating system through command-line environment, please refer to reference on MS-DOS.

6 CONCLUSION

In this note, we introduced a new quantitative framework suitable for the analysis of macroeconomic, distributional, and welfare impacts of various fiscal policies in LICs. A toolkit with user friendly interface is provided to easily take the framework into practice. In this note, we first introduce the model and the structure of the toolkit. We then illustrate how to work with the toolkit with an example of increasing VAT rate by 4%. We later cover advanced topics on exceptional handling, how to use the toolkit when MATLAB is not available, and how to accelerate the execution by parallel with very little efforts. We hope the toolkit could prove useful for a wide audience including economists from the Fund and the Bank, country authorities, researchers, and students. We also expect that more quantitative tools can be brought to the policy world in a similar manner.

APPENDIX

A UPDATES TO VERSION 2.0

Major Updates:

1. Two interfaces, `toolkit_ss` and `toolkit_trans`, respectively for analyzing steady states and transitional dynamics are provided.
2. `toolkit_ss`:
   (a) welfare decomposition now supported
   (b) infrastructure investment can be made together with uniform basic income
   (c) infrastructure investment can be made separately for urban and rural
3. `toolkit_trans`:
   (a) new interface to solve transitional dynamics
   (b) supports only cash transfer policy
   (c) exports data to Excel format
   (d) supports welfare decomposition
4. Full scale update in documentation.
5. Simplification in installation procedure.

Minor Updates:

1. Deletion of unused features: government balance sheet and consumption changes by deciles.
2. Deletion of computational accuracy control.
3. Simplification of UI.
4. Improvement on messages sent to the terminal.

B SYSTEM REQUIREMENTS

The Graphical User Interface (GUI) of the toolkit is designed using MATLAB 2016b. A MATLAB copy of version higher than 2014b is required to run the GUI due to several changes in *class*-related syntax. An Octave version of higher than 3.8.0 is required to analyze the results if MATLAB is not available. Currently, GNU Octave cannot be used to launch the GUIs. For that, MATLAB is required. The binary files are compiled from Fortran code using Intel Visual Fortran (IVF) Compiler XE 14.0.0.103 [IA-64] on Microsoft

Windows Operating System with CPU of Intel x86-64 architecture. Runtime library of IVF is integrated through static linking. Runtime library of Intel Message Passing Interface (MPI) library is invoked through dynamic linking, and therefore is a prerequisite if the users execute the program in parallel mode. If the user uses the toolkit directly, only Windows operating systems and CPU compatible with Intel x86-64 architecture are supported. If the user would like to use the toolkit under a different environment, for instance on operating systems like Linux, Unix, or macOS, or with Intel IA-32 (or non-Intel) architecture CPUs, please recompile the source code. In this case, the Software Development Kit (SDK) of Intel MPI is required.

- Operating Systems: Microsoft Windows 7 Service Pack 1, Windows 8, Windows 8.1, Windows 10. Windows Vista and XP are *not* supported.
- Processors: Intel x86-64 processors supporting SSE2 (Streaming Single Instruction Multiple Data Extension 2) instruction set.
- RAM: 2GB is required.
- Libraries: Intel MPI Runtime Library (only when executed in parallel).

C SOLVING THE HOMOGENOUS AGENTS VERSION OF THE MODEL

In this appendix, we provide details on how the homogenous agents version of the model is solved.

C.1 *The Model*

The model contains three types of agents, rural household, urban household, and large farmers, with measure μ^r, μ^u, and μ^f respectively. All agents have the same log-linear preference over food c^a, manufacturing goods c^m, and service c^s:

$$u(c^a, c^m, c^s) = \log(c^a - \overline{a}) + \gamma \log c^m + \psi \log c^s, \tag{C.1}$$

where $\overline{a}$ is the subsistence level. Let the price of the goods be respectively p^a, p^m, and p^s. The preference implies that given total consumption expenditure C, household first spends $p^a\overline{a}$ to ensure subsistence, and distributes the rest money by fraction $1/(1+\gamma+\psi), \gamma/(1+\gamma+\psi)$, and $\psi/(1+\gamma+\psi)$ for the three goods respectively. This is formally summarized in the following Lemma.

Lemma 1. (Optimal Consumption Bundle) *For the Stone-Geary preference (C.1), given total consumption expenditure C, the optimal consumption bundle is*

$$\begin{aligned} c^a &= \left(\frac{1}{1+\gamma+\psi}\right)\frac{1}{p^a}\overline{C} + \overline{a}, \\ c^m &= \left(\frac{\gamma}{1+\gamma+\psi}\right)\frac{1}{p^m}\overline{C}, \\ c^s &= \left(\frac{\psi}{1+\gamma+\psi}\right)\frac{1}{p^s}\overline{C}, \end{aligned}$$

where $\overline{C} = C - p^a\overline{a}$ is the consumption expenditure net of subsistence requirement.

Proof. The proof is straightforward. The optimization problem is

$$\max_{c^a,c^m,c^s} \log(c^a - \overline{a}) + \gamma \log c^m + \psi \log c^s$$
$$s.t. \qquad p^a c^a + p^m c^m + p^s c^s = C.$$

The Lagrangian of the optimization problem is

$$\mathcal{L} = \log(c^a - \overline{a}) + \gamma \log c^m + \psi \log c^s + \lambda(C - p^a c^a + p^m c^m + p^s c^s),$$

where λ is the multiplier. The first order conditions are

$$\frac{1}{c^a - \overline{a}} = \lambda p^a, \qquad \frac{1}{c^m} = \lambda p^m, \qquad \frac{1}{c^s} = \lambda p^s.$$

The results follow immediately by substituting the first order conditions into the budget constraint. Notice that for the optimization problem to be well-defined, $\overline{C}$ has to be positive, meaning that the consumer has to at least be able to pay for the subsistence consumption. As is stated in Section 3.3 of the *Tutorial*, a lot of situations where the algorithm fails is caused by this issue. □

Lemma 1 implies that the consumption-saving decision and the optimal consumption bundle decision can be analyzed separately. Further, notice that *given* prices p^a, p^m, and p^s, consumption c^a, c^m, and c^s are *affine* in C. We will utilize this property later when we solve for the steady state in Section 5.2.2.

Each member of urban and rural households makes decisions on how much to consume and how much to save. Their savings are transformed to capital that are rent by the manufacturing firms. The urban and rural formal labor markets where household members supply labor in exchange for wages are separated. Manufacturing firms hire urban labor for wage w^m, and large farmers hire rural labor for wage w^f. Hence each household member distributes her unit labor endowment between the formal and informal purposes. Manufacturing firms produce manufacturing goods by combining capital and urban labor, while large farmers organize rural labor to produce domestic and exported agricultural goods. All service goods are produced entirely by informal labor of urban households. Rural household members use their informal labor to produce domestic agricultural food.

Domestic agricultural goods and service goods are only used for consumption. Exported agricultural goods only serve the international market. Manufacturing goods in this economy are used for several purposes which we will describe shortly.

Government collects taxes and does lump-sum transfers. The taxes include consumption taxes on domestic agricultural goods and manufacturing goods τ^a, labor income tax τ^w, corporate profit taxes τ^r. Government runs a balanced budget every period, therefore tax revenues in excess of transfers are used to purchase manufacturing goods that are later used in non-productive way. In addition, in this model some agricultural goods are exported. We assume that there is no capital account and official foreign reserves, hence the current account is balanced every period as well. The current account is balanced by importing manufacturing good from abroad. Furthermore, since agricultural goods and service goods are used solely for consumption,

investment of manufacturing firms and large farmers is also made with manufacturing goods. Notice that in steady state, investment equals depreciation.

Therefore, the total amount of manufacturing goods that are available in the economy is those produced by the manufacturing firms and those imported. These goods are used for domestic consumption, investment, and government purchase.

With the general structure described as above, now we introduce the model formally.

Optimization Problems.—The sequential problem of the urban households is

$$\max_{\{C_t^u, h_t^u, b_{t+1}^u\}} \sum_{t=0}^{\infty} \beta^t u(C_t^u)$$

$$s.t.$$

$$C_t^u + b_{t+1}^u = (1-\tau^w) w^m h_t^u + p^s z^s (1-h_t^u)^{1-\alpha^s} + (1+r) b_t^u.$$

The sequential problem of the rural household is

$$\max_{\{C_t^r, h_t^r, b_{t+1}^r\}} \sum_{t=0}^{\infty} \beta^t u(C_t^r)$$

$$s.t.$$

$$C_t^r + b_{t+1}^r = (1-\tau^w) w^f h_t^r + p^a z^a (d^r)^{\alpha^a} (1-h_t^r)^{1-\alpha^a} + (1+r) b_t^r.$$

Similarly, the sequential problem of large farmers is

$$\max_{\{C_t^f, h_t^a, h_t^*, k_{t+1}^f\}} \sum_{t=0}^{\infty} \beta^t u(C_t^f)$$

$$s.t.$$

$$\begin{aligned} C_t^f + k_{t+1}^f &= (1-\tau^r)(\pi_t^f + \pi_t^*) + (1-\delta) k_t^f + \tau^r \delta k_t^f, \\ \pi_t^f &= p^a z^a (d^a)^{\alpha_1^a} (h^a)^{1-\alpha_1^a} - w^f h^a, \\ \pi_t^* &= p^* z^* (d^*)^{\alpha_1^*} (h^*)^{\alpha_2^*} (k_t^f)^{1-\alpha_1^*-\alpha_2^*} - w^f h^*. \end{aligned}$$

Finally, the static manufacturing firms problem is

$$\max_{\{k_t^m, h_t^m\}} \left\{ (1-\tau^i) z^m (k^m)^{\alpha^m} (h^m)^{1-\alpha^m} - w^m h^m - (r+\delta) k^m \right\}.$$

To specify the government budget constraint, we introduce several notations to ease exposition. Define

$$C_t^x = \mu^u c_t^{x,u} + \mu^r c_t^{x,r} + \mu^f c_t^{x,f}, \qquad x = \{a, m, s\},$$

as the aggregate consumption of each goods. If we let G be government expenditure, then the government budget constraint is

$$G + \mu^f \tau^f \delta k_t^f = \tau^a (p^a C_t^a + C_t^m) + \mu^f \tau^r (\pi_t^f + \pi_t^*) + \tau^w (\mu^u w^m h^u + \mu^r w^f h^r).$$

Market Clearing Conditions.—There are altogether six markets. Since we will be focus on the steady state equilibrium, we suppress the subscript t in the market clearing conditions. These conditions are listed as follows.

(i) Urban Labor Market:

(C.2) $$\mu^u h^u = h^m.$$

(ii) Rural Labor Market:

$$\mu^r h^r = \mu^f (h^a + h^*).$$

(iii) Capital Market:

$$\mu^u b^u + \mu^r b^r = k^m.$$

(iv) Domestic Agricultural Goods Market:

$$C^a = \mu^r z^a (d^r)^{\alpha^a} (1 - h^r)^{1-\alpha^a} + \mu^f z^a (d^a)^{\alpha_1^a} (h^a)^{1-\alpha_1^a}.$$

(v) Service Market:

$$C^s = \mu^u z^s (1 - h^u)^{\alpha^s}.$$

(vi) Manufacturing Markets:

$$C^m + \delta(k^m + \mu^f k^f) + G = z^m (k^m)^{\alpha^m} (h^m)^{1-\alpha^m} + \mu^f R^*,$$

where $R^* = p^* z^* (d^*)^{\alpha_1^*} (h^*)^{\alpha_2^*} (k_t^f)^{1-\alpha_1^*-\alpha_2^*}$ is the revenue from exporting.

C.2 *The Steady State Equilibrium*

Equilibrium Conditions.—Notice that compared to the model in the previous section, once we shut down the idiosyncratic labor shocks, the model is deterministic. The deterministic steady state is thus a vector of prices and allocations, which greatly simplifies the computation. Since the manufacturing good price is normalized to one, and Lemma 1 allows us to separate consumption saving decision and optimal consumption bundle decision, the steady state of the economy is characterized by the following prices and allocations:

- 5 Prices: $\{p^a, p^s, w^m, w^f, r\}$;
- 10 Allocations of Consumers: $\{C^u, C^r, C^f, b^u, b^r, k^f, h^u, h^r, h^a, h^*\}$;
- 3 Allocations of Firms and Government: $\{k^m, h^m, G\}$.

In total, there are $5 + 10 + 3 = 18$ variables. Therefore, we need respectively 18 equations to pin down their values. Using the fact that in steady state, for all variables $x_{t+1} = x_t$, we can write the 18 equations as follows.

- Consumers' Euler Equation:

$$\beta(1+r)=1.$$

- Urban Consumers:

$$\begin{aligned}&\text{F.O.C of } h^u: && (1-\tau^w)w^m=(1-\alpha^s)p^s z^s(1-h^u)^{-\alpha^s},\\ &\text{Budget}: && C^u=(1-\tau^w)w^m h^u+p^s z^s(1-h^u)^{1-\alpha^s}+rb^u.\end{aligned}$$

- Rural Consumers:

$$\begin{aligned}&\text{F.O.C of } h^r: && (1-\tau^w)w^f=(1-\alpha^a)p^a z^a(d^r)^{\alpha^a}(1-h^r)^{-\alpha^a},\\ &\text{Budget}: && C^r=(1-\tau^w)w^f h^r+p^a z^a(d^r)^{\alpha^a}(1-h^r)^{1-\alpha^a}+rb^r.\end{aligned}$$

- Large Farmers:

$$\begin{aligned}&\text{F.O.C of } k^f: && \frac{1}{\beta}=(1-\tau^r)(1-\alpha_1^*-\alpha_2^*)p^* z^*(d^*)^{\alpha_1^*}(h^*)^{\alpha_2^*}(k^f)^{-\alpha_1^*-\alpha_2^*}+1-\delta+\tau^r\delta,\\ &\text{F.O.C of } h^a: && (1-\alpha_1^a)p^a z^a(d^a)^{\alpha_1^a}(h^a)^{-\alpha_1^a}=w^f,\\ &\text{F.O.C of } h^*: && \alpha_2^* p^* z^*(d^*)^{\alpha_1^*}(h^*)^{\alpha_2^*-1}(k^f)^{1-\alpha_1^*-\alpha_2^*}=w^f,\\ &\text{Budget}: && C^f+\delta(1-\tau^r)k^f=(1-\tau^r)(\pi^f+\pi^*),\end{aligned}$$

where

$$\begin{aligned}\pi^f&=p^a z^a(d^a)^{\alpha_1^a}(h^a)^{1-\alpha_1^a}-w^f h^a,\\ \pi^*&=p^* z^*(d^*)^{\alpha_1^*}(h^*)^{\alpha_2^*}(k^f)^{1-\alpha_1^*-\alpha_2^*}-w^f h^*.\end{aligned}$$

- Government:

$$G+\mu^f\tau^r\delta k^f=\tau^a(p^a C^a+C^m)+\mu^f\tau^r(\pi^f+\pi^*)+\tau^w(\mu^u w^m h^u+\mu^r w^f h^r).$$

- Manufacturing Firms:

$$\begin{aligned}&\text{F.O.C of } k^m: && \alpha^m z^m(k^m)^{\alpha^m-1}(h^m)^{1-\alpha^m}=r+\delta,\\ &\text{F.O.C of } h^m: && (1-\alpha^m)z^m(k^m)^{\alpha^m}(h^m)^{-\alpha^m}=w^m.\end{aligned}$$

- Market Clearing Conditions:

$$\begin{aligned}&\text{Urban Labor}: && \mu^u h^u=h^m,\\ &\text{Rural Labor}: && \mu^r h^r=\mu^f(h^a+h^*),\\ &\text{Capital}: && \mu^u b^u+\mu^r b^r=k^m,\\ &\text{Food}: && C^a=\mu^r z^a(d^r)^{\alpha^a}(1-h^r)^{1-\alpha^a}+\mu^f z^a(d^a)^{\alpha_1^a}(h^a)^{1-\alpha_1^a},\\ &\text{Service}: && C^s=\mu^s z^s(1-h^u)^{\alpha^s},\\ &\text{Manufacturing}: && C^m+\delta(k^m+\mu^f k^f)+G=z^m(k^m)^{\alpha^m}(h^m)^{1-\alpha^m}+\mu^f R^*,\end{aligned}$$

where $\forall x\in\{a,m,s\}$,

$$C^x=\mu^u c^{x,u}+\mu^r c^{x,r}+\mu^f c^{x,f}.$$

The 18 equations are highly nonlinear, hence it would be reckless to try solving them altogether as one giant system. Closer inspection indicates that the unknowns can be solved "block-wisely," meaning that we can break the whole system down to several sub-systems with less unknowns. Specifically, as we will demonstrate in details below, we will first guess a pair of $\{p^a, p^s\}$, and characterize all other prices and allocations using 16 of the 18 equations except the market clearing conditions for food and service markets. We then use the two market clearing conditions to update the guesses and iterate until the markets are cleared.

Solving the Steady State Equilibrium.—First notice that household's intertemporal Euler equations imply that interest rate is always

$$r = \frac{1}{\beta} - 1. \tag{C.3}$$

Combining with the F.O.Cs of manufacturing firms:

$$\alpha^m z^m (k^m)^{\alpha^m - 1} (h^m)^{1-\alpha^m} = r + \delta, \tag{C.4}$$

$$(1-\alpha^m) z^m (k^m)^{\alpha^m} (h^m)^{-\alpha^m} = w^m, \tag{C.5}$$

we can solve for urban wage w^m and manufacturing capital-labor ratio $\kappa^m = k^m/h^m$.

Next, for any guess of p^a, we can use F.O.Cs of large farmers and rural household, and rural labor market clearing condition to solve for $\{h^a, h^*, k^f, w^f\}$. In particular, the subsystem is given by

$$\frac{1}{\beta} = (1-\tau^r)(1-\alpha_1^* - \alpha_2^*) p^* z^* (d^*)^{\alpha_1^*} (h^*)^{\alpha_2^*} (k^f)^{-\alpha_1^* - \alpha_2^*} + 1 - \delta + \tau^r \delta, \tag{C.6}$$

$$(1-\alpha_1^a) p^a z^a (d^a)^{\alpha_1^a} (h^a)^{-\alpha_1^a} = w^f, \tag{C.7}$$

$$\alpha_2^* p^* z^* (d^*)^{\alpha_1^*} (h^*)^{\alpha_2^* - 1} (k^f)^{1-\alpha_1^* - \alpha_2^*} = w^f, \tag{C.8}$$

$$(1-\tau^w) w^f = (1-\alpha^a) p^a z^a (d^r)^{\alpha^a} \left[1 - \frac{\mu^f}{\mu^r}(h^a + h^*)\right]^{-\alpha^a}, \tag{C.9}$$

where in (C.9) we have substituted in the rural labor market condition

$$\mu^r h^r = \mu^f (h^a + h^*). \tag{C.10}$$

Notice that Equations (C.6) to (C.8) are log-linear in the unknowns. Though unfortunately because of Equation (C.9), the whole system is not log-linear, we still can exploit the log-linearity of (C.6) to (C.8) to enhance the robustness of the algorithm. We leave the algebraic details to the appendix. With $\{h^a, h^*\}$, Equation (C.10) can be used to calculate h^r.

Third, for any guess of p^s, F.O.C of urban household

$$(1-\tau^w) w^m = (1-\alpha^s) p^s z^s (1-h^u)^{-\alpha^s}, \tag{C.11}$$

yields the value of h^u. Urban labor market clearing condition can then be invoked to calculate h^m

$$h^m = \mu^u h^u, \tag{C.12}$$

which further gives us the value of k^m from κ^m. Notice that here we use the fact that we have already solved w^m and κ^m before.

Now we are left with the consumption saving decisions of households to solve. More specifically, the remaining unknowns are $\{C^u, C^r, C^f, b^u, b^r, G\}$. From large farmer's budget constraint

$$C^f + \delta(1-\tau^r)k^f = (1-\tau^r)(\pi^f + \pi^*), \tag{C.13}$$

we can solve for C^f. This leaves us with 5 unknowns. Further notice that Lemma 1 shows that households' consumption on each good is affine in total consumption expenditure C^u, C^r, and C^f. Thus the manufacturing good market clearing condition

$$C^m + \delta(k^m + \mu^f k^f) + G = z^m (k^m)^{\alpha^m} (h^m)^{1-\alpha^m} + \mu^f R^*, \tag{C.14}$$

and the government balance budget condition

$$G + \mu^f \tau^r \delta k^f = \tau^a (p^a C^a + C^m) + \mu^f \tau^r (\pi^f + \pi^*) + \tau^w (\mu^u w^m h^u + \mu^r w^f h^r), \tag{C.15}$$

are both linear in $\{C^u, C^r, G\}$. Since urban and rural household budget constraints

$$C^u = (1-\tau^w) w^m h^u + p^s z^s (1-h^u)^{1-\alpha^s} + r b^u, \tag{C.16}$$

$$C^r = (1-\tau^w) w^f h^r + p^a z^a (d^r)^{\alpha^a} (h^a)^{1-\alpha^a} + r b^r, \tag{C.17}$$

and capital market clearing condition

$$\mu^u b^u + \mu^r b^r = k^m, \tag{C.18}$$

are all linear in $\{C^u, C^r, C^f, b^u, b^r, G\}$, Equations (C.14) to (C.18) form a linear system.

The problem here is that the equations in the system (C.14) to (C.18) are *not* independent with each other. As a result, we *cannot* separately pin down C^u, C^r, b^u and b^r. More specifically, we show in Appendix B that we can only determine the aggregate consumption and saving for urban and rural households

$$C^{ur} = \mu^u C^u + \mu^r C^r,$$
$$b^{ur} = \mu^u b^u + \mu^r b^r.$$

This is because in the steady state of a representative economy, the Euler equations of urban and rural households coincide with each other [Equation (C.3)]. Hence from an aggregate point of view, $\mu^u C^u, \mu^r C^r$ and $\mu^u b^u, \mu^r b^r$ are indistinguishable.

Finally, with all the variables solved, we can calculate demand and supply of food and service markets

$$C^a = \mu^r z^a (d^r)^{\alpha^a} (1-h^r)^{1-\alpha^a} + \mu^f z^a (d^a)^{\alpha_1^a} (h^a)^{1-\alpha_1^a}, \tag{C.19}$$

$$C^s = \mu^u z^s (1-h^u)^{1-\alpha^s}. \tag{C.20}$$

Fortunately, domestic food and service market clearing conditions only depend on aggregate consumption as well, meaning that the collinearity does not affect our solution. Hence we can still use these two conditions to update guesses of p^a and p^s until the two markets are cleared.

TABLE C1
PARAMETERS TO BE CALIBRATED ENDOGENOUSLY

Parameter	Notation	Targets
Service Good Preference	ψ	Service Share in Consumption
Manufacturing Good Preference	γ	Manufacturing Share in Consumption
Value Added Tax	τ^a	Tax Revenue as a fraction of GDP
Profit Tax	τ^r	Corporate Tax Share in Tax Revenue
Income Tax	τ^w	Income Tax Share in Tax Revenue
Agricultural Productivity	z^a	Domestic Agricultural Share in GDP
Manufacturing Productivity	z^m	Domestic Manufacturing Share in GDP
Exporting Productivity	z^*	Exporting Agricultural Share in GDP

C.3 *Calibrating the Model*

In the homogenous version of the model, there are in total 8 parameters to be calibrated: $\{\gamma, \psi, z^a, z^m, z^*, \tau^a, \tau^r, \tau^w\}$. Broadly speaking, to calibrate the model is to adjust the values of the 8 parameters, such that some model implied moments are close to those found in the data. The correspondence between the model parameters and economic moments are summarized in Table C1. Mathematically, the whole general equilibrium model could be viewed as a mapping between the parameter values and the model implied moments. The mapping in most cases will be extremely nonlinear and dis-continuous, so direct application of standard minimization routines would fail almost surely.[26] As a result, it is crucial to reduce the dimension of the problem as much as possible.

Similarly to the solution strategy we employ when solving the model, the calibration of the model can be done in a "block-wise" way as well. More specifically, the parameters are divided into three groups: preference $\{\psi, \gamma\}$, productivities $\{z^a, z^m, z^*\}$, and taxes $\{\tau^a, \tau^r, \tau^w\}$. Each group of parameters have very limited effects on the performance of the model over other dimensions, and hence could be handled separately.

Preference.—By Lemma 1, we can show that *if* $\overline{a} = 0$, then the consumption shares of different goods in total expenditure are

$$\frac{(1+\tau^a)p^a c^a}{C} = \frac{1}{1+\gamma+\psi}, \quad \frac{(1+\tau^a)c^m}{C} = \frac{\gamma}{1+\gamma+\psi}, \quad \text{and} \quad \frac{p^s c^s}{C} = \frac{\psi}{1+\gamma+\psi}.$$

Since the calibration of the homogenous agents version of the model is supposed to provide an initial parameterization of the full model, a reasonable approximation would suffice. In addition, because we set $\overline{a}$ to be very small, to reduce the complexity of the problem to enhance robustness, we calculate γ and ψ directly from the data as an approximation. In particular, if we assume the manufacturing and service share in consumption in the data are respectively π_m and π_s, we solve γ and ψ from

$$\frac{\gamma}{1+\gamma+\psi} = \pi_m, \quad \text{and} \quad \frac{\psi}{1+\gamma+\psi} = \pi_s,$$

[26] See Section 3.1 of the *Tutorial* for details.

which yields

$$\gamma = \frac{\pi_m}{1-\pi_m-\pi_s}, \quad \text{and} \quad \psi = \frac{\pi_s}{1-\pi_m-\pi_s}.$$

Minor adjustments could then be applied on basis of the above theoretical values.

Productivity.—The minimization problem of the productivities is standard, except that we use logarithmic transformation to add curvature to the minimization problem. More specifically, denote the production share in the data as $\overline{\kappa}_a, \overline{\kappa}_m$, and $\overline{\kappa}_x$, instead of solving

$$\max_{z^a, z^m, z^*} [\kappa_a(\boldsymbol{Z}) - \overline{\kappa}_a]^2 + [\kappa_m(\boldsymbol{Z}) - \overline{\kappa}_m]^2 + [\kappa_x(\boldsymbol{Z}) - \overline{\kappa}_x]^2,$$

numerically, we solve

$$\max_{\log(z^a), \log(z^m), \log(z^*)} [\kappa_a(\boldsymbol{Z}) - \overline{\kappa}_a]^2 + [\kappa_m(\boldsymbol{Z}) - \overline{\kappa}_m]^2 + [\kappa_x(\boldsymbol{Z}) - \overline{\kappa}_x]^2.$$

Theoretically, both problems should yield the same solution. However, in practice, because the gradient of the problem with respect to z is small when the the objective function is flat, the logarithmic version increases the variation of $\boldsymbol{Z}$ to facilitate the application of gradient based optimization routines like the Quasi-Newton Method.

Taxes.—The minimization problem of the taxes is also standard. One caveat is that we *cannot* use τ^a, τ^r and τ^w to target share of value added taxes, corporate taxes, and income taxes in total tax revenue independently. This is because in the model there are only three taxes, so by definition their shares add up to one. This means that if we are targeting the tax shares only, then we are targeting two moments with three parameters, which results in multiple solutions. Therefore, one of the taxes needs to be adjusted to target a different moment, which we choose the total tax revenue as a fraction of GDP. Notice that to match the model to the data, the data have to be adjusted such that all taxes be mapped to one of the three model categories, and the empirical counterpart of the three taxes add up to one.

In addition, in both minimization problems, we choose to minimize the raw sum of squares of model-data difference as opposed to percentage deviations, because numerically the latter is very unstable, and often makes the algorithm collapse. But again, in theory, they should give you the same solution.

C.4 *Math Appendix*

Rural Labor Market.—In the section, we show the detailed derivation of using the log-linearity of Equations (C.6) to (C.9) to simplify the nonlinear system to a one-dimensional nonlinear equation. To ease exposition, we list the four equations below again:

$$\frac{1}{\beta} = (1-\tau^r)(1-\alpha_1^*-\alpha_2^*)p^* z^* (d^*)^{\alpha_1^*}(h^*)^{\alpha_2^*}(k^f)^{-\alpha_1^*-\alpha_2^*} + 1 - \delta + \tau^r \delta, \tag{C.21}$$

$$(1-\alpha_1^a) p^a z^a (d^a)^{\alpha_1^a}(h^a)^{-\alpha_1^a} = w^f, \tag{C.22}$$

$$\alpha_2^* p^* z^* (d^*)^{\alpha_1^*}(h^*)^{\alpha_2^*-1}(k^f)^{1-\alpha_1^*-\alpha_2^*} = w^f, \tag{C.23}$$

$$(1-\tau^w) w^f = (1-\alpha^a) p^a z^a (d^r)^{\alpha^a} \left[1 - \frac{\mu^f}{\mu^r}(h^a + h^*)\right]^{-\alpha^a}, \tag{C.24}$$

Let $x = \log h^a, y = \log h^*, z = \log k^f$, and $u = \log w^f$, our objective is to use Equations (C.21) to (C.23) to write Equation (C.24) as an equation of u. Taking logarithmic on both sides of (C.21), we have

$$\alpha_2^* y - (\alpha_1^* + \alpha_2^*) z = A, \tag{C.25}$$

where

$$A = \log \left[\frac{1/\beta - 1 + \delta - \tau^r \delta}{(1-\tau^r)(1-\alpha_1^* - \alpha_2^*) p^* z^* (d^*)^{\alpha_1^*}} \right].$$

Similarly, for (C.22), we have

$$\alpha_1^a x + u = B, \tag{C.26}$$

where

$$B = \log \left[(1-\alpha_1^a) p^a z^a (d^a)^{\alpha_1^a} \right].$$

For (C.23), we have

$$(1-\alpha_2^*) y + (\alpha_1^* + \alpha_2^* - 1) z + u = C, \tag{C.27}$$

where

$$C = \log \left[\alpha_2^* p^* z^* (d^*)^{\alpha_1^a} \right].$$

And for (C.24), we have

$$u + \alpha^a \log \left[1 - \frac{\mu^f}{\mu^r} (\mathrm{e}^x + \mathrm{e}^y) \right] = D, \tag{C.28}$$

where

$$D = \log \left[\frac{(1-\alpha^a) p^a z^a (d^r)^{\alpha^a}}{1-\tau^w} \right].$$

Notice that (C.25) defines a linear relation between y and z, which combines with the linear relation in (C.26) yields a linear relation of y and u. With the linear relation of x and u defined in (C.26), we can simplify (C.28) to a uni-dimensional root finding problem. Specifically, (C.25) gives

$$z = \frac{\alpha_2^* y - A}{\alpha_1^* + \alpha_2^*}.$$

Substitute the above equation to (C.27), we get

$$(1-\alpha_2^*) y + \left(\frac{\alpha_1^* + \alpha_2^* - 1}{\alpha_1^* + \alpha_2^*} \right) (\alpha_2^* y - A) + u = C.$$

Rearranging terms, we have

$$y = \frac{1}{E} \left[C + \frac{\alpha_1^* + \alpha_2^* - 1}{\alpha_1^* + \alpha_2^*} A - u \right], \tag{C.29}$$

where

$$E = 1 - \alpha_2^* + \frac{\alpha_2^* (\alpha_1^* + \alpha_2^* - 1)}{\alpha_1^* + \alpha_2^*}.$$

Similarly, (C.26) can be rearranged to

$$x = \frac{1}{\alpha_1^a}(B - u). \tag{C.30}$$

Substitute (C.29) and (C.30) to (C.28), we get the uni-dimensional nonlinear equation, for which numerical methods are much more robust than multi-dimensional cases.

Consumption-Saving Decision.—In this section, we document the details pertaining to the algebraic derivation of the linear system used to solve the consumption saving decisions $\{C^u, C^r, b^u, b^r, G\}$ of households. We list the five equations for ease of exposition:

$$C^m + G = z^m (k^m)^{\alpha^m} (h^m)^{1-\alpha^m} + \mu^f R^* - \delta(k^m + \mu^f k^f) \tag{C.31}$$

$$-\tau^a(p^a C^a + C^m) + G = -\mu^f \tau^r \delta k^f + \mu^f \tau^r (\pi^f + \pi^*) + \tau^w(\mu^u w^m h^u + \mu^r w^f h^r) \tag{C.32}$$

$$C^u - rb^u = (1 - \tau^w) w^m h^u + p^s z^s (1 - h^u)^{1-\alpha^s} \tag{C.33}$$

$$C^r - rb^r = (1 - \tau^w) w^f h^r + p^a z^a (d^r)^{\alpha^a} (1 - h^r)^{1-\alpha^a} \tag{C.34}$$

$$\mu^u b^u + \mu^r b^r = k^m. \tag{C.35}$$

First notice that, Equations (C.31) and (C.32) are written in C^a and C^m, not in C^u and C^r. However, Lemma 1 can be invoked to show that C^m and C^a are linear in C^u and C^r. We begin by doing this transformation.

Let the optimal consumption bundle share be given by

$$\begin{aligned}
\pi_a &= \left(\frac{1}{1+\gamma+\psi}\right) \frac{1}{p^a(1+\tau^a)}, \\
\pi_m &= \left(\frac{\gamma}{1+\gamma+\psi}\right) \frac{1}{1+\tau^a}, \\
\pi_s &= \left(\frac{\psi}{1+\gamma+\psi}\right) \frac{1}{p^s},
\end{aligned}$$

then for households of type $x \in \{u, r, f\}$, we have

$$\begin{aligned}
c^{x,a} &= \pi_a[C^u - (1+\tau^a)p^a\overline{a}] + \overline{a}, \\
c^{x,m} &= \pi_m[C^u - (1+\tau^a)p^a\overline{a}], \\
c^{x,s} &= \pi_s[C^u - (1+\tau^a)p^a\overline{a}].
\end{aligned}$$

Using the definition that $C^m = \mu^u c^{u,m} + \mu^r c^{r,m} + \mu^f c^{f,m}$, we can show that

$$C^m = \pi_m(\mu^u C^u + \mu^r C^r + \mu^f C^f) - \pi_m(1+\tau^a)p^a\overline{a}, \tag{C.36}$$

which is linear in C^u and C^r. By similar reasoning, we can show that

$$C^a = \pi_a(\mu^u C^u + \mu^r C^r + \mu^f C^f) + \overline{a}[1 - \pi_a(1+\tau^a)p^a], \tag{C.37}$$

$$C^s = \pi_s(\mu^u C^u + \mu^r C^r + \mu^f C^f) - \pi_s(1+\tau^a)p^a\overline{a}. \tag{C.38}$$

If we let the system (C.31) to (C.35) be compactly represented as $\boldsymbol{AX} = \boldsymbol{B}$:

$$\begin{bmatrix} A_{11} & A_{12} & \cdots & A_{15} \\ A_{21} & A_{22} & \cdots & A_{25} \\ \vdots & \vdots & \ddots & \vdots \\ A_{51} & A_{52} & \cdots & A_{55} \end{bmatrix} \begin{bmatrix} C^u \\ C^r \\ b^u \\ b^r \\ G \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_5 \end{bmatrix}, \tag{C.39}$$

the system is characterized by $\boldsymbol{A}$ and $\boldsymbol{B}$.

Substitute Equation (C.36) into (C.31), we show that

$$A_{11} = \pi_m \mu^u, \quad A_{12} = \pi_m \mu^r, \quad A_{13} = A_{14} = 0, \quad \text{and} \quad A_{15} = 0,$$

and

$$B_1 = z^m (k^m)^{\alpha^m} (h^m)^{1-\alpha^m} + \mu^f R^* - \delta(k^m + \mu^f k^f) - \pi_m \mu^f C^f + \pi_a (1+\tau^a) p^a \overline{a}.$$

Likewise, substitute Equations (C.36) and (C.37) into (C.32), we find that

$$\begin{aligned} A_{21} &= -\tau^a (p^a \pi_a \mu^u + \pi_m \mu^u), \\ A_{22} &= -\tau^a (p^a \pi_a \mu^r + \pi_m \mu^r), \\ A_{23} &= A_{24} = 0, \quad \text{and} \quad A_{25} = 1, \end{aligned}$$

and

$$\begin{aligned} B_2 = {} & \mu^f \tau^r (\pi^f + \pi^*) + \tau^w (\mu^u w^m h^u + \mu^r w^f h^r) - \mu^f \tau^r \delta k^f + \tau^a p^a \pi_a \mu^f C^f \\ & + \tau^a p^a \overline{a} [1 - \pi_a (1+\tau^a) p^a] - \tau^a \pi_m p^a (1+\tau^a) \overline{a} + \tau^a \pi_m \mu^f C^f. \end{aligned}$$

Direct mapping of Equations (C.33) to (C.35) to matrix notations yields

$$\begin{aligned} & A_{31} = 1, \quad A_{32} = 0, \quad A_{33} = -r, \quad \text{and} \quad A_{34} = A_{35} = 0, \\ & B_3 = (1-\tau^w) w^m h^u + p^s z^s (1-h^u)^{1-\alpha^s}, \end{aligned}$$

and

$$\begin{aligned} & A_{41} = 0, \quad A_{42} = 1, \quad A_{43} = 0, \quad A_{44} = -r, \quad \text{and} \quad A_{45} = 0, \\ & B_4 = (1-\tau^w) w^f h^r + p^a z^a (d^r)^{\alpha^a} (1-h^r)^{1-\alpha^a}, \end{aligned}$$

and finally

$$A_{51} = A_{52} = 0, \quad A_{53} = \mu^u, \quad A_{54} = \mu^r, \quad A_{55} = 0, \quad \text{and} \quad B_5 = k^m.$$

To see that Equations (C.31) to (C.35) are not independent with each other, multiply (C.31) by $1/\pi_m$ and (C.32) by $-1/\tau^a(p^a\pi_a + \pi_m)$, we find that in both equations, C^u and C^r enter jointly as $\mu^u C^u + \mu^r C^r$. Moreover, multiply (C.33) by μ^u and (C.34) by μ^r, and add the two equations together, we find that C^u, C^r and b^u, b^r enter the equation jointly as $\mu^u C^u + \mu^r C^r$ and $\mu^u b^u + \mu^r b^r$. Since Equation (C.35) is also in

$\mu^u b^u + \mu^r b^r$, only three of the five equations are linear independent with each other. As a result, only $\mu^u C^u + \mu^r C^r$, $\mu^u b^u + \mu^r b^r$, and G can be independently solved.

The demand of domestic food and service goods are therefore respectively

$$\begin{aligned}
&\text{(Food)}: \quad \pi_a(\mu^u C^u + \mu^r C^r + \mu^f C^f) + \overline{a}[1 - \pi_a p^a(1+\tau^a)], \\
&\text{(Service)}: \quad \pi_s(\mu^u C^u + \mu^r C^r + \mu^f C^f) - \pi_s p^a(1+\tau^a)\overline{a}.
\end{aligned}$$

REFERENCES

Adamopoulos, Tasso, and Diego Restuccia. 2014. "The Size Distribution of Farms and International Productivity Difference." *American Economic Review*, 104(6): 1667–1697. 8, 9

Aiyagari, S. Rao. 1994. "Uninsured Idiosyncratic Risk and Aggregate Saving." *The Quarterly Journal of Economics*, 109(3): 659–684. 5, 6, 23, 24, 33

Allen, Treb, and David Atkin. 2017. "Volatility and the Gains from Trade." *Manuscript*. 9

Anderson, Kym, Gordon Rausser, and Johan Swinnen. 2013. "Political Economy of Public Policies: Insights from Distortions to Agricultural and Food Markets." *Journal of Economic Literature*, 51(2): 423–477. 8

Auerbach, Alan J., and Laurence J. Kotlikoff. 1987. *Dynamic Fiscal Policy.* Cambridge:Cambridge University Press. 45

Bakış, Ozan, Barış Kaymak, and Markus Poschke. 2015. "Transitional Dynamics and the Optimal Progressivity of Income Redistribution." *Review of Economic Dynamics*, 18(3): 679–693. 51

Burgess, Robin, Olivier Deschenes, Dave Donaldson, and Michael Greenstone. 2017. "Weather, Climate Change, and Death in India." *Manuscript*. 9

Carroll, Christopher D. 1997. "Buffer-Stock Saving and the Life Cycle/Permanent Income Hypothesis." *The Quarterly Journal of Economics*, 112(1): 1–55. 35

Castañeda, Ana, Javier Díaz-Giménez, and José-Víctor Ríos-Rull. 2003. "Accounting for Earnings and Wealth Inequality." *Journal of Political Economy*, 111(4): 818–857. 23

Chen, Chaoran, Diego Restuccia, and Raül Santaeulàlia-Llopis. 2017. "The Effects of Land Markets on Resource Allocation and Agricultural Productivity." *NBER Working Paper No. 24034*. 9

Chetty, Raj, and Adam Szeidl. 2007. "Consumption Commitments and Risk Preferences." *The Quarterly Journal of Economics*, 122(2): 831–877. 9

Dawkins, Christina, T.N. Srinivasan, and John Whalley. 2001. "Calibration." In *Handbook of Econometrics*. Vol. 5, , ed. James J. Heckman and Edward Leamer, Chapter 58, 3653–3703. Amsterdam:North-Holland. 14

Domeij, David, and Jonathan Heathcote. 2004. "On the Distributional Effects of Reducing Capital Taxes." *International Economic Review*, 45(2): 523–554. 35, 40, 44, 51

Donovan, Kevin. 2018. "Agricultural Risk, Intermediate Inputs, and Cross-Country Productivity Differences." *Manuscript*. 9

Duflo, Esther, and Rohini Pande. 2007. "Dams." *The Quarterly Journal of Economics*, 122(2): 601–646. 9, 10

Eastwood, Robert, Michael Lipton, and Andrew Newell. 2010. "Farm Size." In *Handbook of Agricultural Economics*. Vol. 4, , ed. Prabhu L. Pingali and Robert E. Evenson, Chapter 65, 3323–3397. Burlington:Elsevier BV. 7, 11

Gollin, Douglas, Stephen L. Parente, and Richard Rogerson. 2007. "The Food Problem and the Evolution of International Income Levels." *Journal of Monetary Economics*, 54(4): 1230–1255. 8

Hubmer, Joachim, Per Krusell, and Anthony A. Smith. 2018. "A Comprehensive Quantitative Theory of the U.S. Wealth Distribution." *NBER Working Paper No. 23011*. 23

Huggett, Mark. 1993. "The Risk-Free Rate in Heterogeneous-Agent Incomplete-Insurance Economies." *Journal of Economic Dynamics and Control*, 17: 953–969. 33

Kimball, Miles S. 1990. "Precautionary Saving in the Small and in the Large." *Econometrica*, 58(1): 53–73. 35

Krusell, Per, and Anthony A. Smith. 1998. "Income and Wealth Heterogeneity in the Macroeconomy." *The Journal of Political Economy*, 106(5): 867–896. 24

Lagakos, David, and Michael E. Waugh. 2013. "Selection, Agriculture, and Cross-Country Productivity Differences." *American Economic Review*, 103(2): 948–980. 8

Lagakos, David, Mushfiq Mobarak, and Michael E. Waugh. 2017. "The Welfare Effects of Encouraging Rural-Urban Migration." *Manuscript*. 7, 9

La Porta, Rafael, and Andrei Shleifer. 2014. "Informality and Development." *Journal of Economic Perspectives*, 28(3): 109–126. 11

Lucas, Robert E. 1987. *Models of Business Cycles.* Cambridge, Massachussets:Basil Blackwell. 40

Lucas, Robert E.B. 1997. "Internal Migration in Developing Countries." In *Handbook of Population and Family Economics*. Vol. 1B, , ed. Mark R. Rosenzweig and Oded Stark, Chapter 13, 721–798. Elsevier Science B.V. 7

Peralta-Alva, Adrian, Xuan Song Tam, Xin Tang, and Marina Mendes Tavares. 2018. "The Macroeconomic and Distributional Implications of Fiscal Consolidations in Low-income Countries." *IMF Working Paper (18/146)*. 9

Peralta-Alva, Adrian, Xuan Song Tam, Xin Tang, and Marina Mendes Tavares. 2019. "The Welfare Implications of Fiscal Consolidations in Low-income Countries." *IMF Working Paper (18/146)*. 5, 6, 19

Pingali, Prabhu L. 2007. "Agricultural Mechanization: Adoption Patterns and Economic Impact." In *Handbook in Agricultural Economics*. Vol. 3, , ed. Robert E. Evenson and Prabhu L. Pingali, Chapter 54, 2779–2805. North Holland:Elsevier B.V. 10

Restuccia, Diego, Dennis Tao Yang, and Xiaodong Zhu. 2008. "Agricultural and Aggregate Productivity: A Quantitative Cross-Country Analysis." *Journal of Monetary Economics*, 55(2): 234–250. 8

Ríos-Rull, José-Víctor. 1999. "Computation of Equilibria in Heterogeneous-agent Models." In *Computational Methods for the Study of Dynamic Economies*. , ed. Ramon Marimon and Andrew Scott, 238–273. New York:Oxford University Press. 44

Roy, Andrew Donald. 1951. "Some Thoughts on the Distribution of Earnings." *Oxford Economic Papers*, 3(2): 135–146. 5

Schultz, Theodore William. 1953. *The Economic Organization of Agriculture.* New York:McGraw-Hill. 9

Storesletten, Kjetil, Christopher I. Telmer, and Amir Yaron. 2001. "The Welfare Cost of Business Cycles Revisited: Finite Lives and Cyclical Variation in Idiosyncratic Risk." *European Economic Review*, 45(7): 1311–1339. 40

Tombe, Trevor. 2015. "The Missing Food Problem: Trade, Agriculture, and International Productivity Differences." *American Economic Journal: Macroeconomics*, 7(3): 226–258. 8

Wu, Yiyun, Xican Xi, Xin Tang, Deming Luo, Baojing Gu, Shu Kee Lam, Peter Vitousek, and Deli Chen. 2018. "Policy Distortions, Farm Size, and the Overuse of Agricultural Chemicals in China." *Proceedings of the National Academy of Sciences of the United States of America*, 115(27): 7010–7015. 9