A Toolkit for Solving Models with a Lower Bound on Interest Rates of Stochastic Duration

Guide and Examples

Gauti B. Eggertsson, Sergey K. Egiev, Alessandro Lin, Josef Platzer and Luca Riva

A Code Setup

In this Section we explain how to set up codes for the toolkit for the New Keynesian model under OCP as explained in Section 4. The solution algorithm is generated in three functions that have to be run sequentially. The essential inputs of those functions are (1) the matrices that define the model; (2) a parameter structure that is described below; (3) a configuration structure that is described below. There are several optional inputs that we explain at the end of the Section.

A.1 Variables and Matrices

Our toolkit requires the model to be cast in the form specified Equation 5. In addition, a specific ordering of variables and equations is necessary for the code to recognise state variables, shocks and the equation subject to the occasionally binding constraint. In particular, the user should order elements in the vector ξ_t such that the jump variables come first, the nominal rate should be between jump variables and predetermined variables, predetermined variables should follow, and finally shock variables should be at the end of the vector.[45] Furthermore, the order of rows in the matrices A and B must satisfy that the last equation is the one that is slack when in regimes 1 and 2 (e.g. a policy rule for the central bank), and the block preceding it is made by as many identity rows as there are shocks (i.e. equations in the form $\mathbb{E}_t u_{t+1} = u_t$).

As an example, consider Equations (12)-(18) in the absorbing state, where (16) is binding. The system can be written as $A\ \mathbb{E}_t\xi_{t+1} = B\ \xi_t$,[46] where $\xi_t \equiv \begin{bmatrix} \hat{Y}_t & \pi_t & i_t & \phi_{1,t-1} & \phi_{2,t-1} & r^n_{t-1} & u_t \end{bmatrix}'$ and

$$A = \begin{bmatrix} 1 & \sigma & 0 & 0 & 0 & \sigma & 0 \\ 0 & \beta & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & \kappa & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 0 & \sigma & 0 & 0 & 0 & 0 \\ -\kappa & 1 & 0 & 0 & 0 & 0 & 0 \\ \lambda & 0 & 0 & -\frac{1}{\beta} & 0 & 0 & 0 \\ 0 & 1 & 0 & -\frac{1}{\beta}\sigma & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{A.1}$$

A.2 Parameters structure

The parameter structure (we name it *param*) needs to contain the following elements.

1. A scalar element μ, equal to the Markov probability described in the text.

[45] Note that if the system has constants, one can implement them in 2 ways. First, the user could define a predetermined variable – along with its own initial value – that never change value across time. A second way is to increase the shock vector to include a component that does not change value across regimes.

[46] We suggest to use AAA and BBB in place of A and B in order to avoid coding conflicts with the inputs in other scripts.

2. Two vector elements s_h and s_l equal to the values of the two-state Markov process in the absorbing and crisis states. The order needs to be the same as in the vector ξ_t.
3. A scalar element N_S, that is equal to the number of state variables in ξ_t, including the exogenous shocks.

A.3 Configuration structure

The configuration structure (we name it *config*) needs to contain the following elements.

1. A scalar element τ_{max}, equal to the time at which the Markov shock reverts to its absorbing state with probability 1, conditional on being in the crisis state at $\tau_{max} - 1$.
2. A scalar element $max_{length2}$, equal to the maximum length of regime 2. This is a shortcut to save computing time. The toolkit will warn the user if regime 2 requires a higher value for $max_{length2}$.
3. A scalar element *bound*, equal to lower bound in the inequality constraint, e.g. 0 for the zero lower bound.
4. A scalar element *mono*. *mono*=1 means toolkit will run under the assumption that the vector k represents a monotone function.

A.4 The general setup

The user should construct the elements described above and set the code as follows.

```
[D_3,G_3,D_3a]                        = regime3(AAA,BBB,param);
[D_2,G_2]                             = regime2(AAA,BBB,D_3a,param,config);
[D_1,G_1, ResM, max_k,k,T_tilde]      = ...
        regime1(AAA,BBB,D_3a,D_3,D_2,G_3,G_2,param,config);
impulseresponse
```

A.5 The function **regime3.m**

The function `regime3.m` takes as inputs the matrices AAA and BBB, as well as the parameters structure `param`.

```
[D_3,G_3,D_3a] = regime3(AAA,BBB,param);
```

This function provides transition matrices that are then used in the other functions.

A.6 The function **regime2.m**

The function `regime2.m` takes as inputs the matrices AAA and BBB, the parameters structure `param`, the configuration structure *config*, as well as one output (D_{3a}) from the function `regime3.m`.

```
[D_2,G_2] = regime2(AAA,BBB,D_3a,param,config);
```

This function provides transition matrices that are then used in the function `regime1.m`.

A.7 The function `regime1.m`

The function `regime1.m` takes as inputs the matrices AAA and BBB, the parameters structure `param`, the configuration structure `config`, as well as the output from the functions `regime3.m` and `regime2.m`.

```
[D_1,G_1, ResM, max_k,k,T_tilde] = ...
        regime1(AAA,BBB,D_3a,D_3,D_2,G_3,G_2,param,config);
```

This function provides a 3 dimensional matrix $ResM$, a scalar max_k, the vector k, the scalar $\tilde{T}$, and the transition matrices in `regime1.m`.

The dimensions of $ResM$ are time, variables, and contingencies. For instance, the element $ResM(5,1,8)$ contains the value of the variable in position 1 in ξ_t, at time 5, for contingency 8.

The vector k is a vector that links contingencies and their respective duration of regime 2. max_k is the maximum value across k, and the scalar $\tilde{T}$ is the period at which regime 1 starts.[47]

Optional inputs There are several optional parameters to `regime1.m`. The user can choose to have any combination of the following:

Option	Input	Description
`verbose`	0 (default) | 1	display real-time diagnostics from the search algorithm
`k_input`	vector	impose arbitrary k
`T_tilde_input`	scalar	input a value for $\tilde{T}$
`R0_search`	0 | 1 (default)	search for $\tilde{T}$

The input of a vector k shall be taken with some caution. Notice that if k has too low entries (e.g. there is too little stimulus), the toolkit will force k to change because regime 3 would feature a violation of the inequality constraint. On the other hand, shall one want to impose a k that is large enough (for instance in the case of a fixed horizon forward guidance experiment), the user should be aware it is necessary to shut down the search for $\tilde{T}$ at the same time to fulfil the purpose. For the third optional input, an issue arises when the user inputs a value for $\tilde{T}$ that is too large. The toolkit will not find a solution because of the algorithm.[48] To avoid this, one should shut down the search for $\tilde{T}$ as well. In this case the toolkit will give a solution that features the value of $\tilde{T}$ chose by the user (otherwise the default value is 1). Note that this solution may violate the inequality constraint in regime 0, exactly because the algorithm for the search of $\tilde{T}$ is shut down. Shutting down the regime 0 algorithm, and setting `T_tilde_input` to τ_{max} corresponds to the case of the ELB never binding.

A.8 The script `impulseresponse.m`

This script generates impulse response functions as weighted averages of the evolution of each variable across all contingencies, weighted by the ex-ante probability of the shock reverting to the absorbing state. The script generates a two-dimensional matrix, where each row is a period and column is a variable. Notice that the matrix `ResM` contains variables in *levels*, so percentage point variations can be obtained by simply multiplying by 100. In addition in some cases, as for example for inflation, the variable is defined on a quarterly basis, so it needs to be multiplied by 4 to yield annualised variations.

[47]To be precise, the scalar $\tilde{T}$ is the period at which the regime that follows regime 0 starts. In most cases this will be regime 1. However, it can be the case that the code switches to regime 2 or 3 immediately after regime 0. This is for example the case if the ELB is never binding.

[48]See Section 3 for more details.

A.9 The function `graphing.m`

In addition, we provide a basic graphics function to produce plots for IRFs as in Section 4. The function has the following required inputs:

- the matrix *IR* containing the impulse response functions, as generated by `impulseresponse.m`;
- a variables structure, that contains the position of each variable;
- a scalar for the horizon of the plots;

This default inputs will produce the average impulse response function for all variables in the model, averaging IRFs across contingencies.

Optional inputs Options for this function allow to plot only a subset of variables and to superimpose impulse response functions for a specific set of contingencies:

Option	Input	Description
`variables`	cell array	plot a subset of variables listed in cell array
`cont_data`	matrix *ResM*	to be used in conjunction with `cont_num`. Provides IRFs for each contingency
`cont_num`	vector	selects contingencies to plot

The line of code below shows an example.

```
graphing(IR,var,30,'variables',{'pi','y','i'},'cont_data',ResM,'cont_num',1:30)
```

This line produces the impulse response function for variables π, y and i, as well as the IRFs specific to contingencies 1 to 30.

B Examples

This section presents a series of examples that are designed to show the features of the toolkit, and to guide the user in the setup required to solve a specific model. The examples are built around the simple two-equation model described by Equations 14-15 and a backward-looking Taylor rule with interest rate smoothing (TTRS).

All example files set the function `regime1.m` in verbose mode, in order to provide the user with more information on the inner workings of our toolkit.

Example 1 shows how to use the code in its most basic form. We suggest the user to start with that first, as its configuration is often referenced to by subsequent examples.

B.1 Example 1

This example shows how to solve a simple model with state variables.

B.1.1 Model

The equations describing the model are:

$$\mathbb{E}_t x_{t+1} + \sigma \mathbb{E}_t \pi_{t+1} = x_t + \sigma\left(i_t - r_t^n\right) \tag{B.2}$$

$$\beta \mathbb{E}_t \pi_{t+1} = \pi_t - \kappa x_t \tag{B.3}$$

$$0 = -i_t + \max\{0, \phi_i i_{t-1} + (1-\phi_i)(r^* + \phi_\pi \pi_{t-1} + \phi_x x_{t-1})\} \tag{B.4}$$

where r_t^n evolves as a two-state Markov process with constant transition probability μ. Equations B.2-B.4 are stated with all expectational variables on the left-hand side, and current pre-determined variables and constants on the right-hand side, so that they conform to the linear system in Equation 5. Note how B.4 is the equation subject to the bound, so it will be listed as last in the script declaring all the equations (which we conventionally call `equations.m`).

B.1.2 Variables

Our toolkit requires the model to be cast in the form specified by Equation 5. We want to plot, together with the other endogenous variables, impulse responses for the interest rate implied by TTRS in B.4; hence, we define an ancillary jump variable tracking the implied rate: $i_t^{imp} \equiv \phi_i i_{t-1} + (1-\phi_i)(r^* + \phi_\pi \pi_{t-1} + \phi_x x_{t-1})$. As a consequence, the vector of variables ξ_t will be:

$$\xi_t \equiv \left[x_t \; \pi_t \; i_t^{imp} \; i_t \; x_{t-1} \; \pi_{t-1} \; i_{t-1} \; r^* \; r_t^n\right]'$$

The ordering of variables is as important for the correct operation of our toolkit as it was for equations. As noted in A.1, the user should first include jump variables (in this example x_t, π_t and i_t^{imp}), then the variable subject to the occasionally binding constraint (i_t) followed by predetermined variables ($x_{t-1}, \pi_{t-1}, i_{t-1}$), constants ($r^*$) and finally shocks ($r_t^n$). This is achieved by generating a structure called `vars`, which lists as fields the position of each variable in ξ_t (in our codes and examples, we build this structure in the script `variables.m`)

B.1.3 Equations

A very similar structure (generated in our codes in `equations.m`) declares the ordering for the equations. This object does not contain any actual mathematical specification, but rather it simply links labels for each equation (for example `nkpc` or `policy`) to the intended ordering. Once again, the equation subject to the occasionally binding constraint B.4 is ordered last. Here it is called `rule` and listed as number 9. In addition, the identities specifying the shocks (only r_t^n in this example) are to be included second to last. The other seven equations are, in ascending order: equations B.2 and B.3, three equations defining the lagged terms x_{t-1}, π_{t-1} and i_{t-1},[49] one equation defining i_t^{imp} and finally one equation for the constant term r^*. In our implementation of the model, this last term is introduced as a permanent shock, so the equation specifying it is in the form $\mathbb{E}_t r^*_{t+1} = r^*_t$. This will be the first component of what the toolkit treats as Markov disturbances. Section B.1.5 provides further details on how to parametrise it so that it takes a constant value across all regimes.

The coefficients of the linearised system in A and B in 5 are specified in `matrices.m`. Since the matrices are quite sparse, the code initialises them at zero and fills in the non-zero elements in the subsequent lines. Our codes make use of the structures defined in `equations.m` and `variables.m` to locate the position of each variable in every equation. The user is free to specify numerical values for the coefficients directly, or to obtain them from a separate structure dedicated to the model calibration (we take the latter route, see `parameters.m`). The resulting matrices are as follows:

$$A \equiv \begin{bmatrix} 1 & \sigma & 0 & 0 & 0 & 0 & 0 & 0 & \sigma \\ 0 & \beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B \equiv \begin{bmatrix} 1 & 0 & 0 & \sigma & 0 & 0 & 0 & 0 & 0 \\ -\kappa & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & (1-\phi_i)\phi_x & (1-\phi_i)\phi_\pi & \phi_i & 1-\phi_i & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & (1-\phi_i)\phi_x & (1-\phi_i)\phi_\pi & \phi_i & 1-\phi_i & 0 \end{bmatrix} \tag{B.5}$$

Once again, the last row in A and B is the policy rule subject to the occasionally binding constraint. The fourth column contains coefficients for the jump variable subject to the constraint (i_t). The last two columns report coefficients for the shocks: the second to last is a permanent shock used to include a constant term r^*, while the last column is the proper forcing term r_t^n, which follows a two-state Markov process.

B.1.4 Initial conditions

The toolkit allows for optional initial conditions only for the pre-determined variables. To simplify the syntax and help the user keep track of what initial values are introduced, these conditions are declared as a column vector of the same length as ξ_t, stored in the field `param.init_cond`. If the field is omitted, the toolkit simply initialises it at zero. In this example, we introduce an initial condition $i_{-1} = \frac{1}{\beta} - 1$.[50] This is implemented in the last line of the script `parameters.m`. As the constant term r^* is introduced in the form of a permanent shock, no further initial conditions are necessary. In the first line of the same

[49] These simple equations are in the form $\mathbb{E}_t y_{t+1} = x_t$, where $y_t = x_{t-1}$. They link the current variable x_t with its lagged value y_t.
[50] Note: this condition applies to the predetermined variable i_{t-1} at $t = 0$, not on the jump variable i_t.

script we also declare the number of state variables, which we pass to the toolkit via the field `param.NS`. In the present example we have three predetermined variables and two shocks, so $N_S = 5$.

B.1.5 Specifying shocks

Finally, we need to parametrise the values for the two-state Markov shock in the high (steady state) and low state (or crisis state), as well as the value for the constant term r^*. The toolkit handles these values as separate vectors for each state. In other words, we need to specify a vector for the values that all shocks in the model take in the low state, and a separate one for the values of each shock in the high state. These vectors are fields in the structure `param` and are called `param.sl` and `param.sh`, respectively. In this example, the first shock is actually a constant term r^*, so it will take the same value $r^* = \frac{1}{\beta} - 1$ in both states of the world. The natural rate r_t^n drops to -0.5% in the low state and reverts back to r^* in steady state. Therefore our two vectors will be as follows:

$$\texttt{param.sl} = \begin{bmatrix} \frac{1}{\beta} - 1 \\ -0.005 \end{bmatrix}, \qquad \texttt{param.sh} = \left(\frac{1}{\beta} - 1\right) \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{B.6}$$

We create these vectors in the script `parameters.m`, in the section named *Shocks*.

This concludes the instructions and customisation specific to this example. For the other parts of `ex_1.m` that are required by the toolkit as standard, we refer the reader to the guide at Section A.

B.2 Example 2

This example is a minor variation of Example 1, and shows how to solve a model under the assumption that the ELB becomes binding at an exogenous period $t = \tilde{T}$. This is implemented by passing two optional parameters to the function `regime1.m`: a value for $\tilde{T}$ itself (option `T_tilde_input`) and a flag to exclude the search for the length of regime 0 in equilibrium (`R0_search`). In this configuration, equilibrium conditions are not satisfied in regime 0. Consequently, the inequality constraint $i_t \geq 0$ is initially violated and the nominal interest rate turns negative in some periods $t < \tilde{T}$ under our calibration.

B.2.1 Model

As in Section B.1.1.

B.2.2 Variables

As in Section B.1.2.

B.2.3 Equations

As in Section B.1.3.

B.2.4 Initial conditions

As in Section B.1.4.

B.2.5 Specifying shocks

As in Section B.1.5

B.2.6 Additional parameters

To force the ELB to be binding from a pre-specified period, we need to include two optional parameters to the inputs of the function `regime1.m`:

1. `RO_search` set to 0, to disable the search for a $\tilde{T}$ that satisfies equilibrium conditions;
2. `T_tilde_input` set to a scalar, in our example $t = 5$, to provide a numerical value for the exogenous start of Regime 1.

For the other parts of `ex_2.m` that are required by the toolkit as standard, we refer the reader to the guide at Section A.

B.3 Example 3

This example shows how to impose an exogenous vector k. As in Example 2, it is implemented as an optional configuration of `regime3.m`, where the user provides a value for k and excludes the search for the equilibrium $\tilde{T}$. For simplicity, in what follows we will work with a constant $k_\tau = 5$ for all τ, so that $k = [5 \ \ldots \ 5]$. This policy prescribes $i_t = 0$ for five additional periods after the natural rate r_t^n has returned to the absorbing state s_h, with no regards to the contingency when the change in regime happens.

B.3.1 Model

As in Section B.1.1.

B.3.2 Variables

As in Section B.1.2.

B.3.3 Equations

As in Section B.1.3.

B.3.4 Initial conditions

As in Section B.1.4.

B.3.5 Specifying shocks

As in Section B.1.5

B.3.6 Additional parameters

To force regime 2 to last for five periods, the user has to include two optional parameters to the inputs of the function `regime1.m`:

1. `R0_search` set to 0, to disable the search for a $\tilde{T}$ that satisfies equilibrium conditions;
2. `k_input` set to a row vector of length τ_{max}, in our example $k = [5 \ \ldots \ 5]$, specifying the duration of regime 2 in every contingency τ.

For the other parts of `ex_3.m` that are required by the toolkit as standard, we refer the reader to the guide at Section A.

B.4 Example 4

This example shows how to solve a model under a deterministic AR(p) shock. For ease of exposition, what follows describes the simplest possible case, where $p = 1$. The implementation is based on setting the parameter $\mu = 1$, so that the two-state Markov disturbance reverts to steady state at $t = 2$ with probability one. Hence, the shock acts as a one-off unanticipated innovation (ε_t) for the deterministic autoregressive shock of the model (r_t^n). The shock itself and its law of motion are then specified as a pre-determined variable. The resulting impulse response is then identical to the second contingency $\tau = 2$, since all others have zero probability.

B.4.1 Model

The equations describing the model are the same as in Example 1 (B.2-B.4), plus the *AR(1)* process for the natural real interest rate:

$$r_{t+1}^n = (1-\rho)r^* + \rho r_t^n + \varepsilon_t \tag{B.7}$$

where ε_t evolves as a two-state Markov process with constant transition probability $\mu = 1$. Note that equation B.7 will be listed fourth to last in the script declaring all the equations (`equations.m`), as r_t^n is now treated as a pre-determined variable.

B.4.2 Variables

Since we now have an AR(1) process for r_t^n, the vector of variables ξ_t will be:

$$\xi_t \equiv \left[x_t \; \pi_t \; i_t^{imp} \; i_t \; x_{t-1} \; \pi_{t-1} \; i_{t-1} \; r_t^n \; r^* \; \varepsilon_t\right]'$$

where ε_t evolves as a two-state Markov process with degenerate transition probability $\mu = 1$. The variable for the natural rate r_t^n is now ordered third to last, among the other pre-determined states x_{t-1}, π_{t-1} and i_{t-1}. We refer the reader to B.1.2 for further details on the implementation.

B.4.3 Equations

The matrices declared in the script `matrices.m` are as follows:

$$A \equiv \begin{bmatrix} 1 & \sigma & 0 & 0 & 0 & 0 & 0 & \sigma & 0 & 0 \\ 0 & \beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B \equiv \begin{bmatrix} 1 & 0 & 0 & \sigma & 0 & 0 & 0 & 0 & 0 & 0 \\ -\kappa & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & (1-\phi_i)\phi_x & (1-\phi_i)\phi_\pi & \phi_i & 0 & 1-\phi_i & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho & r^* & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & (1-\phi_i)\phi_x & (1-\phi_i)\phi_\pi & \phi_i & 0 & 1-\phi_i & 0 \end{bmatrix} \tag{B.8}$$

The only substantive difference with the matrices in B.1.3 is the presence of the equation for the AR(1) process for r_t^n in the fourth-to-last line of A and B.

B.4.4 Initial conditions

Since r_t^n is now a pre-determined variable, it requires an initial condition for the value it assumes in regime 0. For simplicity, we choose the steady-state value $r_{-1}^n = \frac{1}{\beta} - 1$. As in B.1.4, this will be declared at the end of the script `parameters.m`.

B.4.5 Specifying shocks

We modify the vectors specified in B.1.5 so that the two-state Markov disturbance ε_t has no effect on the model after the first period. Trivially, this is done by setting its steady-state value to zero. Our vectors of shocks will be therefore:

$$\texttt{param.sl} = \begin{bmatrix} \frac{1}{\beta} - 1 \\ -0.005 \end{bmatrix}, \qquad \texttt{param.sl} = \begin{bmatrix} \frac{1}{\beta} - 1 \\ 0 \end{bmatrix} \tag{B.9}$$

B.4.6 Additional parameters

For the other parts of `ex_4.m` that are required by the toolkit as standard, we refer the reader to the guide at Section A. Since the model is deterministic, it should be noted that it can be solved by setting $\tau_{max} = 2$, as the only relevant contingency is $\tau = 2$.