RL in SFC models

Vadim Artemov

Definition of terms

- **C** : Consumption goods demand by households
- **G** : Government expenditures
- **Y** : National income
- **WB** : Wage bill
- **T** : Taxes
- **ΔH** : Change in cash money

We have the following transactions matrix describing our system (Source: "Monetary Economics: An Integrated Approach to Credit, Money, Income, Production and Wealth, 2nd ed" by Wynne Godley and Marc Lavoie, 2012):

	1.Households	**2.Production**	**3.Government**	**Σ**
1.Consumption	-Cd	+Cs		0
2.Govt expenditures		+Gs	-Gd	0
3.[Output]		[Y]		
4.Factor income (wages)	+W•Ns	-W•Nd		0
5.Taxes	-Ts		+Td	0
6.Change in the stock of money	-ΔHh		+ΔHs	0
Σ	0	0	0	0

From the table we have the following system of equations, describing our system:

$$\begin{cases} C_s = C_d \\ G_s = G_d \\ T_s = T_d \\ N_s = N_d \\ Y_d = N_s - T_s \\ T_d = 1/2 * N_s \\ C_d = 1/2Y_d + 1/2H_h^{-1} \\ H_s = G_d - T_d + H_s^{-1} \\ Y = C_s + G_s \\ N_d = Y \\ H_n = Y_d - C_d + H_h^{-1} \end{cases}$$

To solve this system, we need some additional assumptions and conditions:

- Government's demand (G_d) is exogenous, and it is our policy
- Total output is a linear combination of this period's and past period's outputs

Then we inroduce our model specific assumptions:

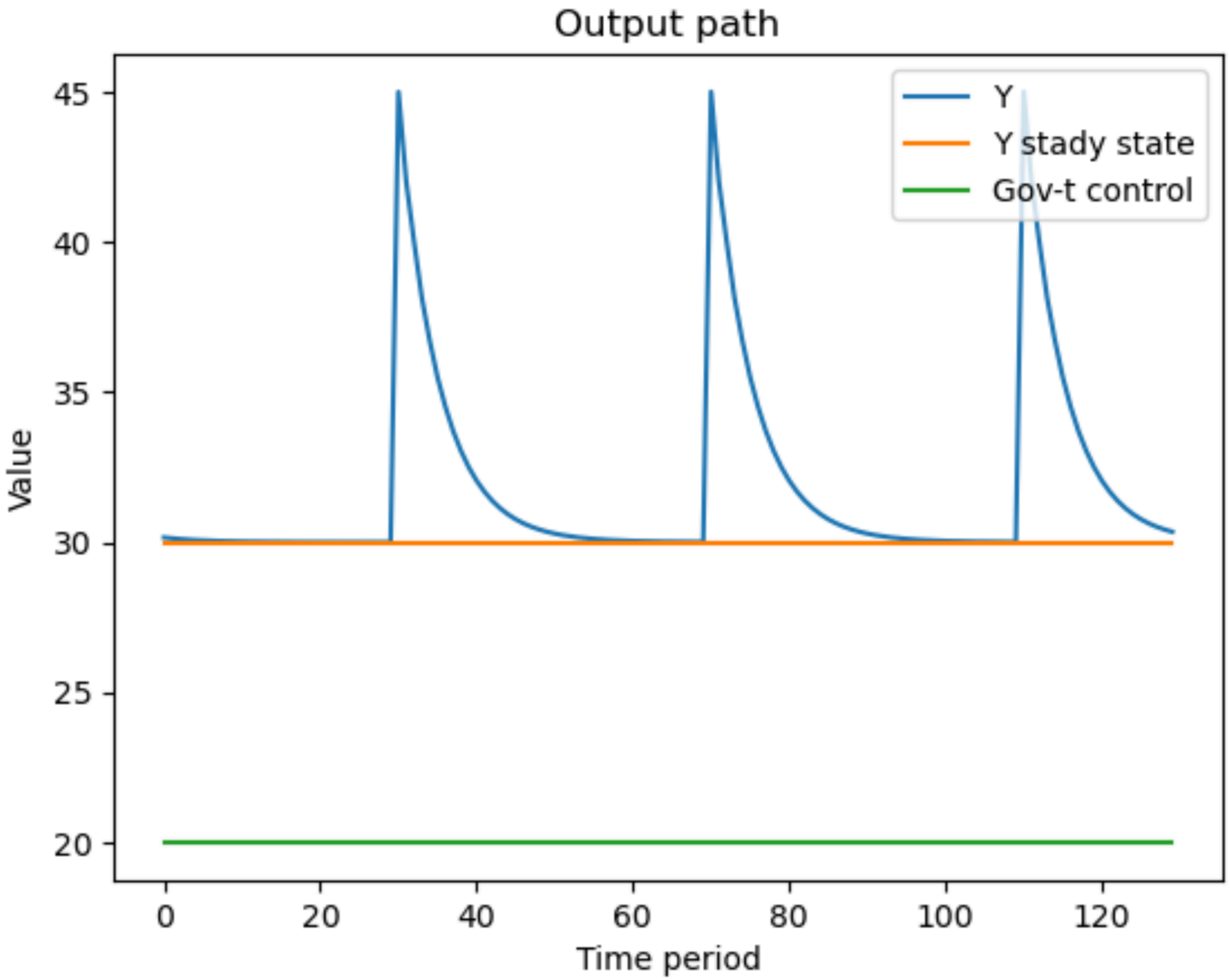
- Initial starting state
- Government minimizes the following objective

$$L = \sum_{i=0}^{N} Y_i - Y_{ss}$$

- Y_i - i-th period output, Y_ss - steady state output (output to which the sustem converges without shocks)
- Every 5th period consumption shock occurs: C_d increases by a factor of 1.5

When we solve the system with the abovementioned assumptions we get:

$$\begin{cases} G_s = G_d = set\ by\ Actor \\ C_s = C_d = 20/3 + 4/6H_h^{-1} \\ Y = (20/3 + 4/6H_h^{-1} + 10)/5 + 4 * Y^{-1}/5 \\ Y_d = 1/2Y \\ T_s = T_d = 1/2Y \\ N_s = N_d = Y \\ H_s = 20 - 1/2Y + H_s^{-1} \\ H_h = 1/2Y - C_d \end{cases}$$



Output path
Y
Y stady state
Gov-t control
45
40
35
30
25
20
Value
0
20
40
60
80
100
120
Time period

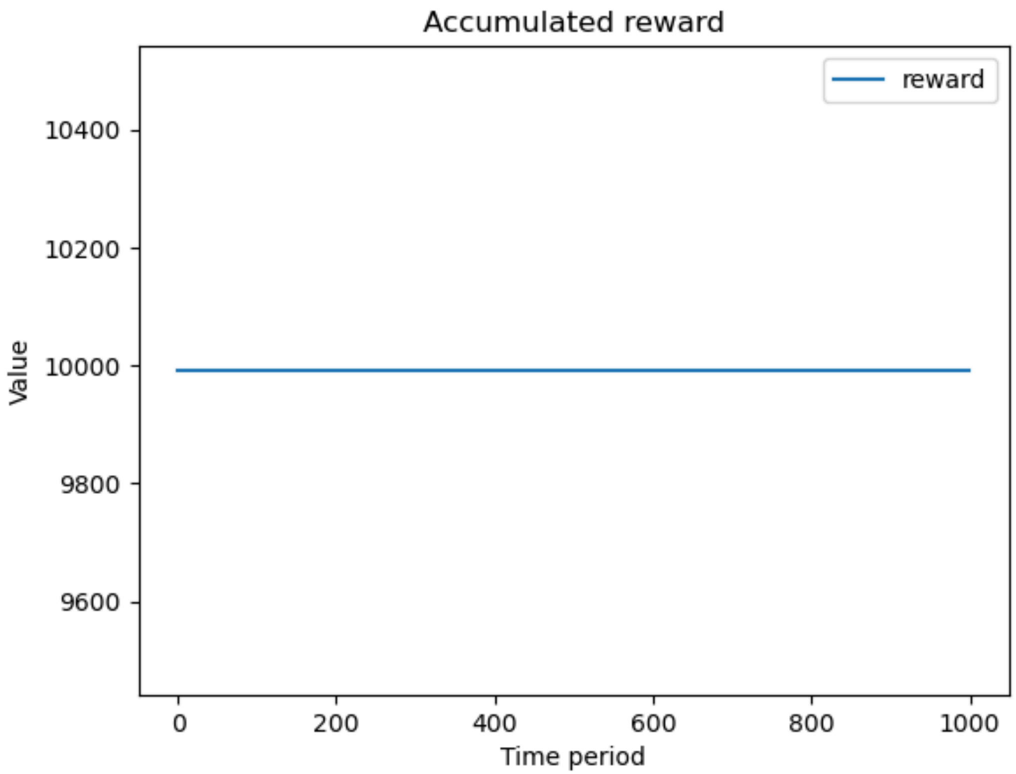
One-step Actor–Critic (episodic)

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value parameterization $\hat{v}(s,\mathbf{w})$
Parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$

Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^{d}$
Repeat forever:
  Initialize S (first state of episode)
  $I \leftarrow 1$
  While S is not terminal:
    $A \sim \pi(\cdot|S, \boldsymbol{\theta})$
    Take action A, observe S', R
    $\delta \leftarrow R + \gamma \hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} I \delta \nabla_{\mathbf{w}} \hat{v}(S,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} I \delta \nabla_{\boldsymbol{\theta}} \ln \pi(A|S, \boldsymbol{\theta})$
    $I \leftarrow \gamma I$
    $S \leftarrow S'$

```python
# Observation -----> Action
class Actor(nn.Module):

    def __init__(self, observation_space, action_space):
        super(Actor, self).__init__()
        self.input_layer = nn.Linear(observation_space, 128)
        self.output_layer = nn.Linear(128, action_space)

    def forward(self, x):
        x = self.input_layer(x)
        x = F.relu(x)
        actions = self.output_layer(x)
        action_probs = F.softmax(actions, dim=1)

        return action_probs

# State -----> Reward
class Critic(nn.Module):

    def __init__(self, observation_space):
        super(Critic, self).__init__()
        self.input_layer = nn.Linear(observation_space, 128)
        self.output_layer = nn.Linear(128, 1)

    def forward(self, x):
        x = self.input_layer(x)
        x = F.relu(x)
        state_value = self.output_layer(x)

        return state_value
```

```
DISCOUNT_FACTOR = 0.999
NUM_EPISODES = 1000
MAX_STEPS = 10000
```



Possible reasons for failure:

- Hands from the wrong place - some errors in code logic implementation
- Explosion of the model (because it is linear)

Possible solutions:

- Debug
- Retruing with different parameters/starting point