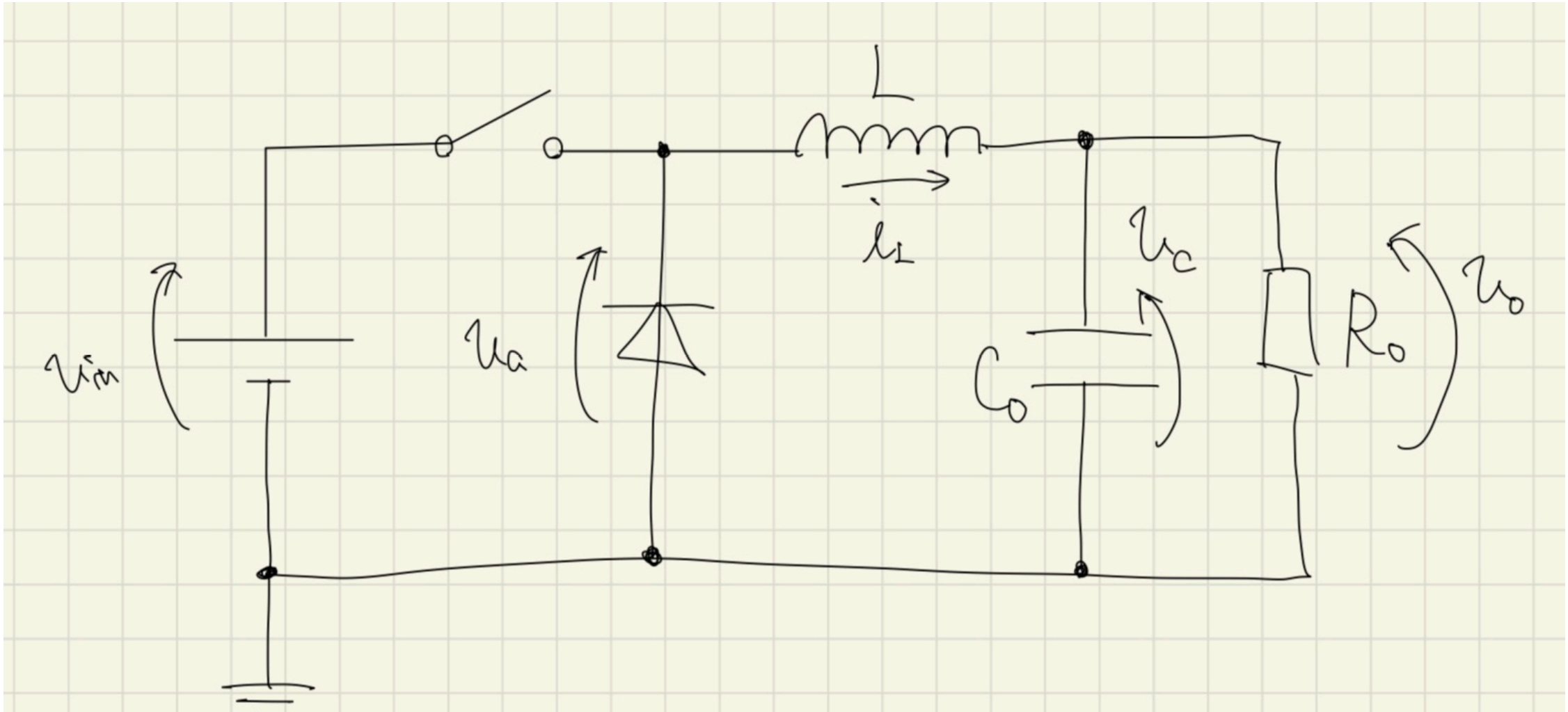
進捗報告

10/27(月)

長崎大学工学部工学科 B4 丸田研究室
35221011 大塚直哉

進捗

Buck コンバータの回路図



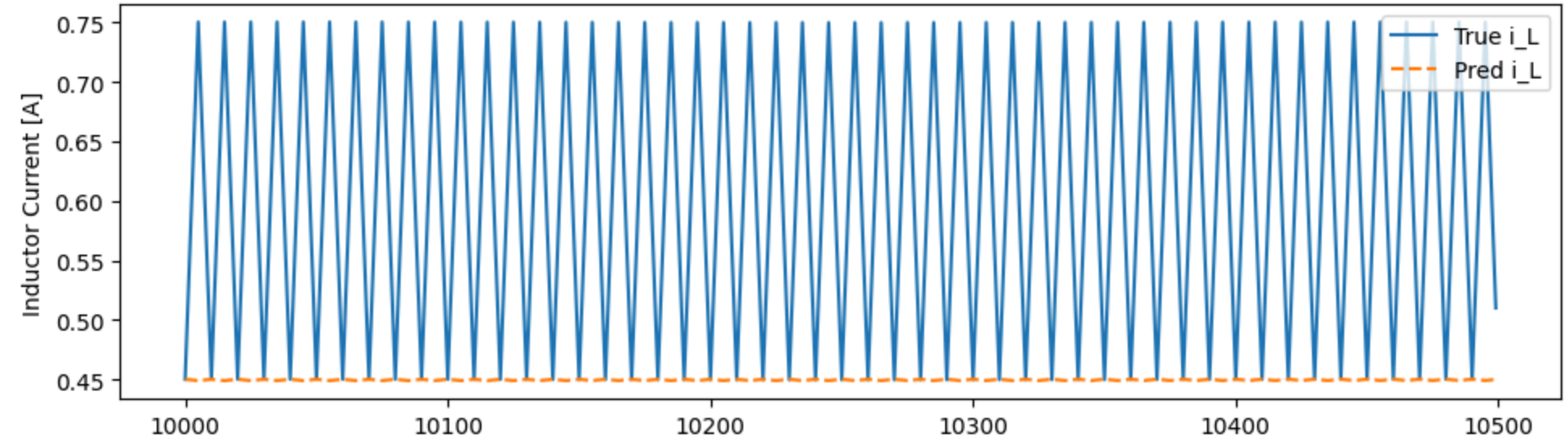
オイラー法による離散化

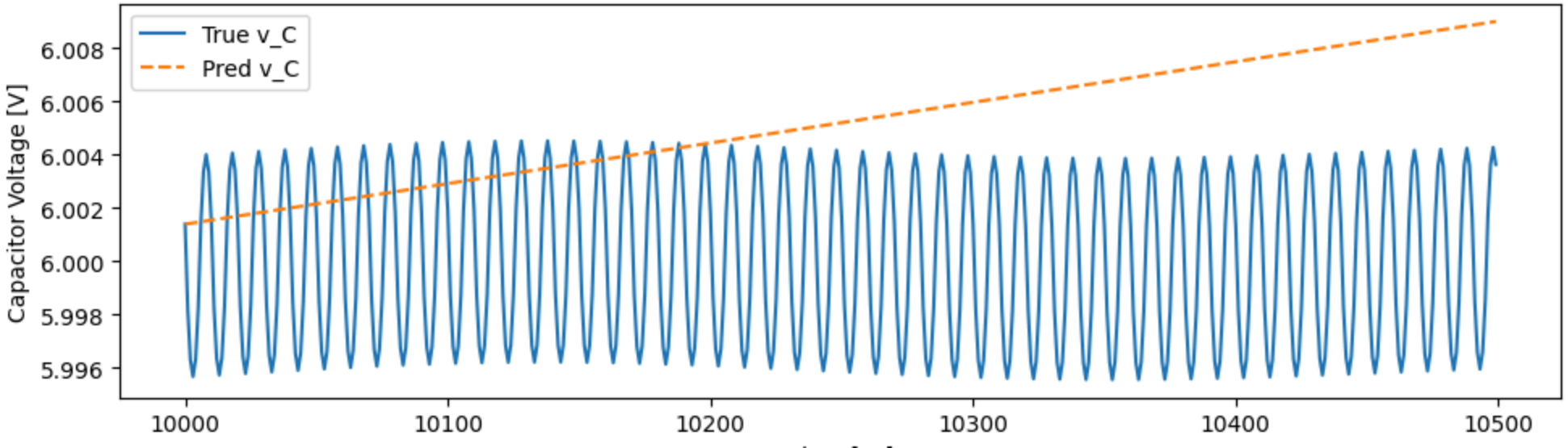
$$i_L(t+\Delta t) = i_L(t) + \frac{\Delta t}{L}(v_a - v_o)$$

$$v_C(t+\Delta t) = v_C(t) + \frac{\Delta t}{C_o}\left(i_L - \frac{v_o}{R_o}\right)$$

```python
class BuckConverterCell(nn.Module):
    def __init__(
        self,
        dt: float = 1e-6,
        L_init: float = 200e-6,
        C_init: float = 100e-6,
        R_init: float = 8.0,
    ) -> None:
        super().__init__()
        self.dt = dt
        self.L = nn.Parameter(torch.tensor(L_init))
        self.C = nn.Parameter(torch.tensor(C_init))
        self.R = nn.Parameter(torch.tensor(R_init))

    def forward(self, h: torch.Tensor, x: torch.Tensor) -> torch.Tensor:
        i_L, v_C = h[:, 0], h[:, 1]
        v_a = x[:, 0]
        di = (self.dt / self.L) * (v_a - v_C)
        dv = (self.dt / self.C) * (i_L - v_C / self.R)
        i_L_next = i_L + di
        v_C_next = v_C + dv
        return torch.stack([i_L_next, v_C_next], dim=1)
```

シミュレーション波形を作成し、学習させる



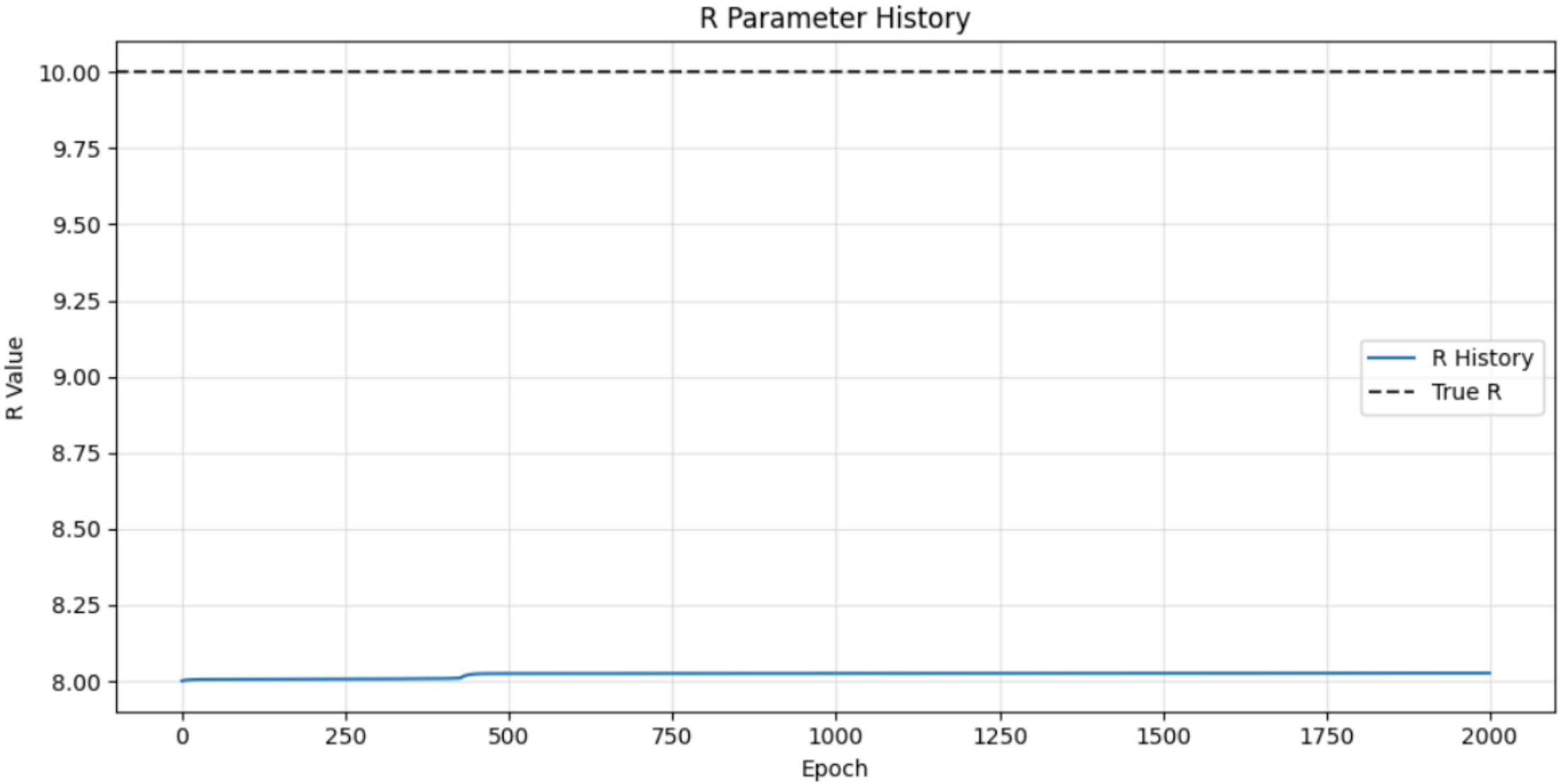


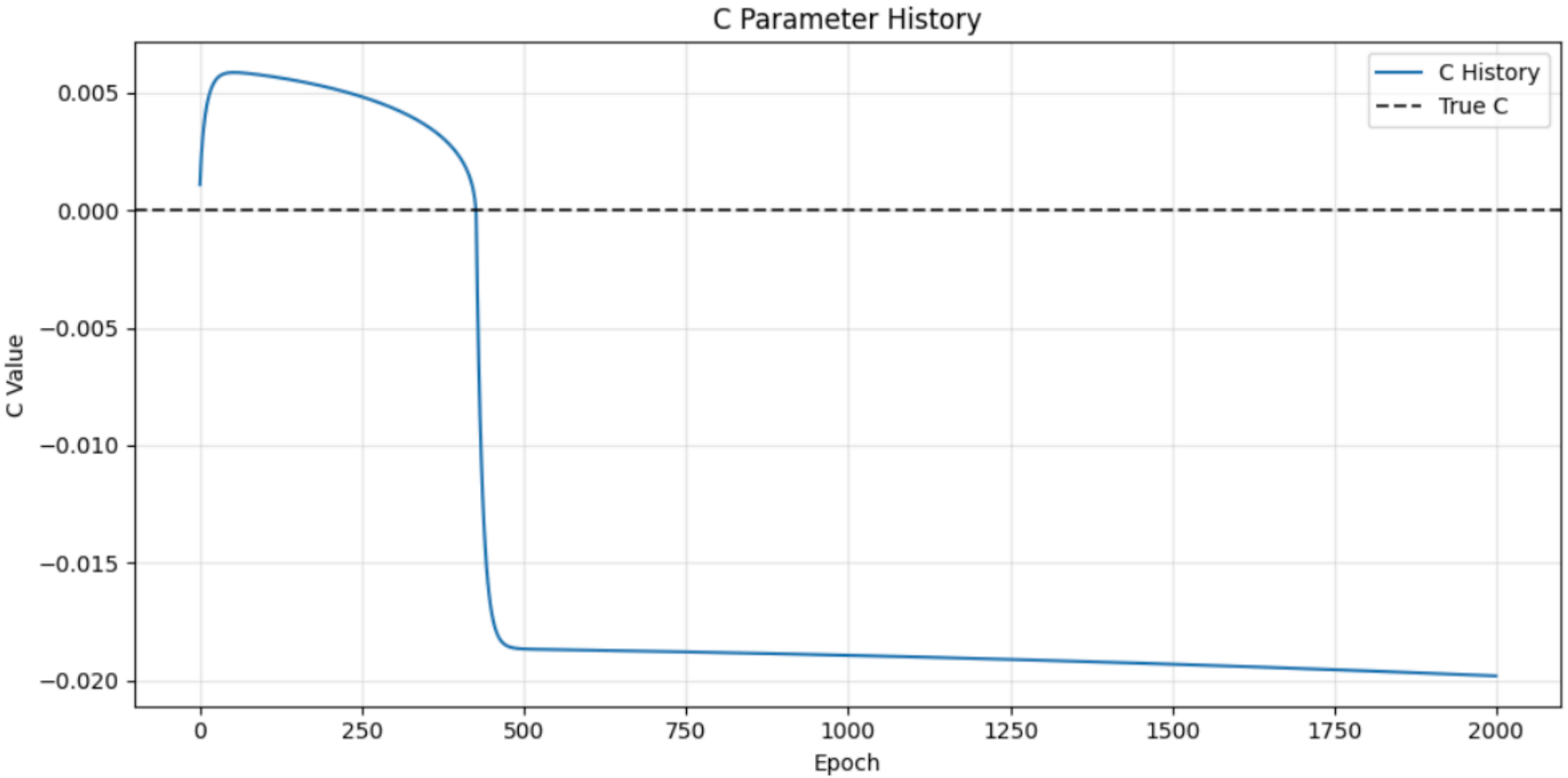
回路パラメータ

```
=== Learned Parameters ===
L = -2.745e-02 H
C = -1.981e-02 F
R = 8.027 Ω
```

```
=== True Parameters ===
L = 100e-6  # インダクタ [H]
C = 47e-6  # コンデンサ [F]
R = 10.0  # 負荷抵抗 [Ω]
```


L Parameter History
L History
True L
0.000
−0.005
−0.010
−0.015
−0.020
−0.025
L Value
0
250
500
750
1000
1250
1500
1750
2000
Epoch

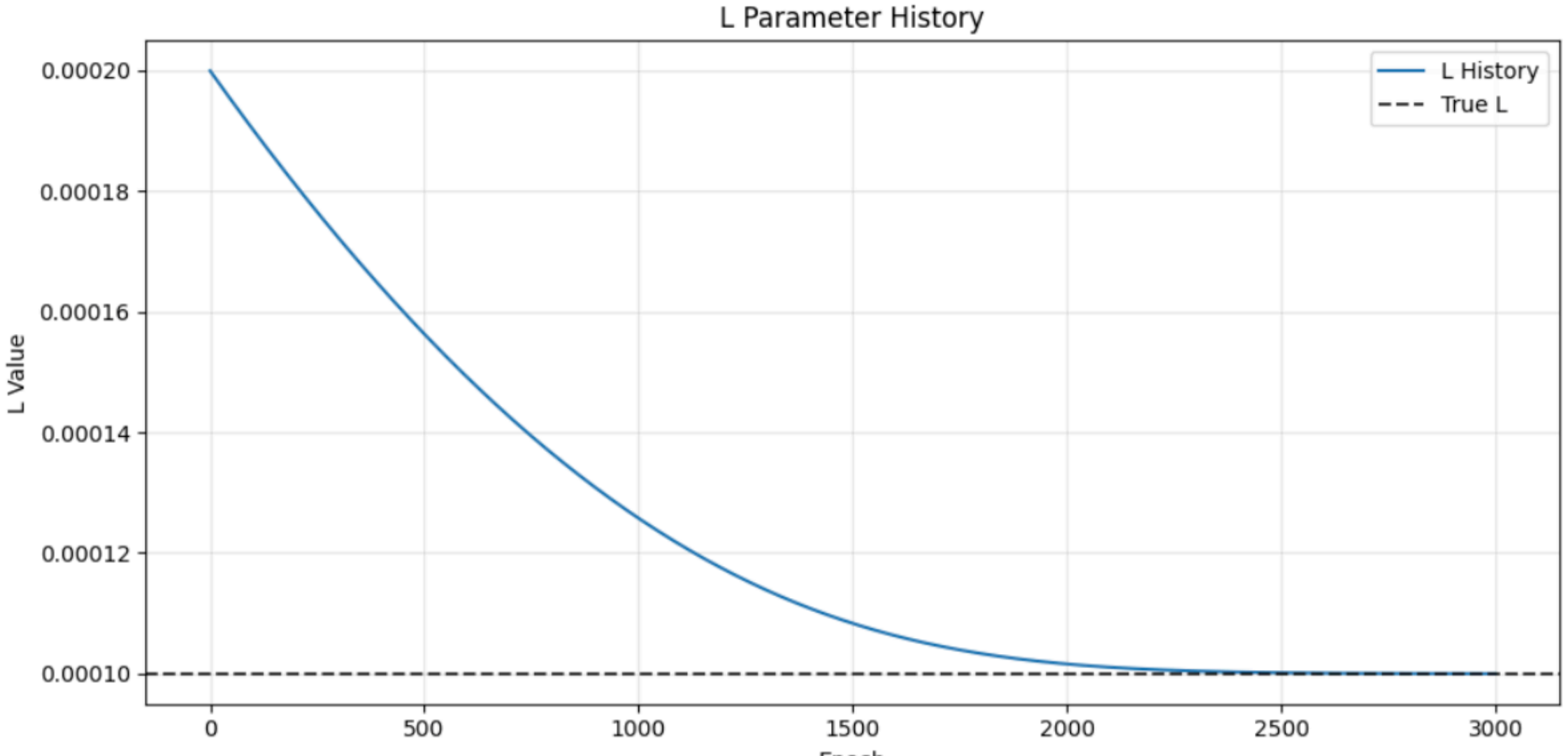

R Parameter History
10.00
9.75
9.50
9.25
9.00
8.75
8.50
8.25
8.00
R Value
0
250
500
750
1000
1250
1500
1750
2000
Epoch
R History
True R

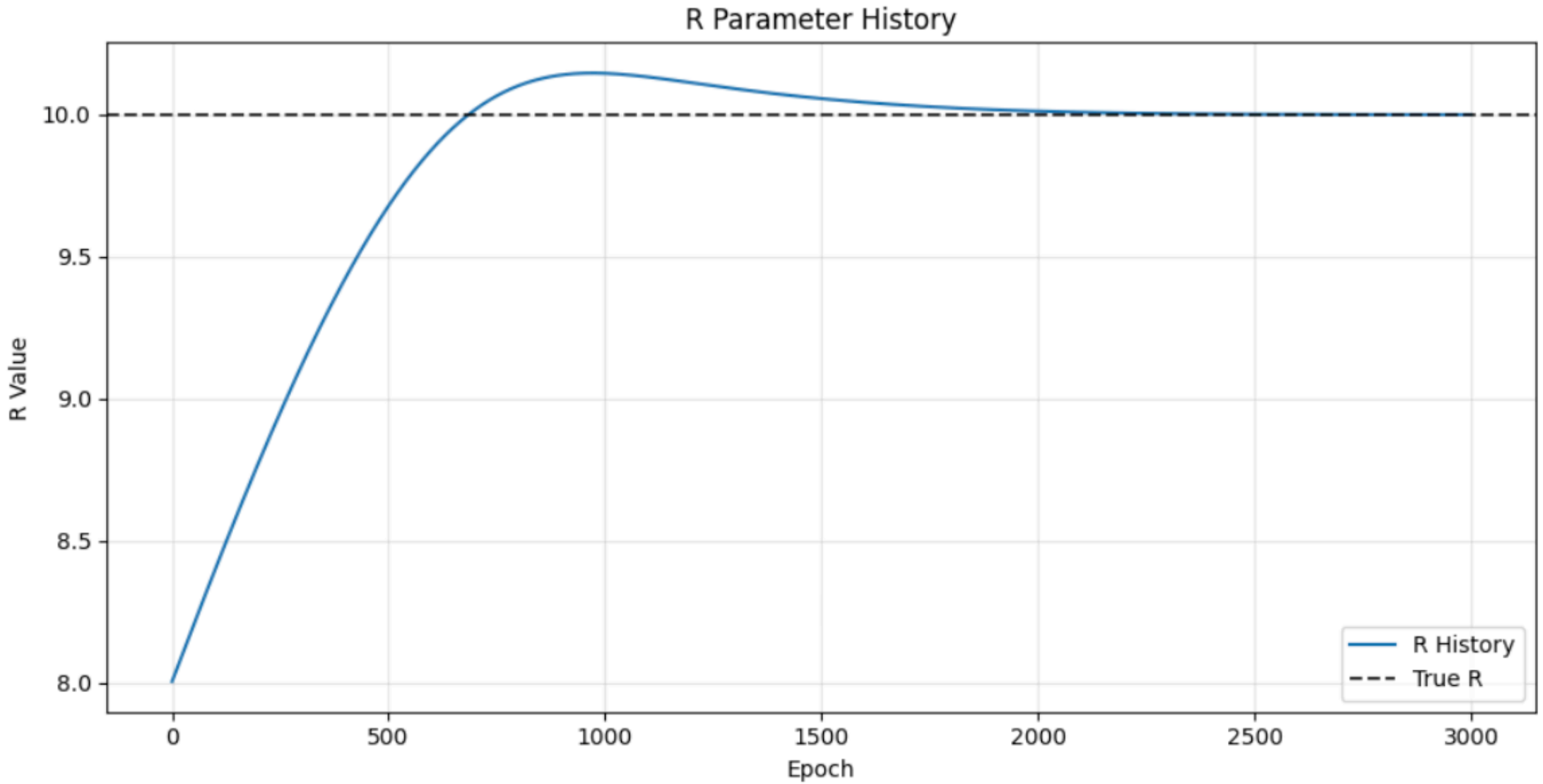

C Parameter History
C History
True C
0.005
0.000
−0.005
−0.010
−0.015
−0.020
C Value
0
250
500
750
1000
1250
1500
1750
2000
Epoch

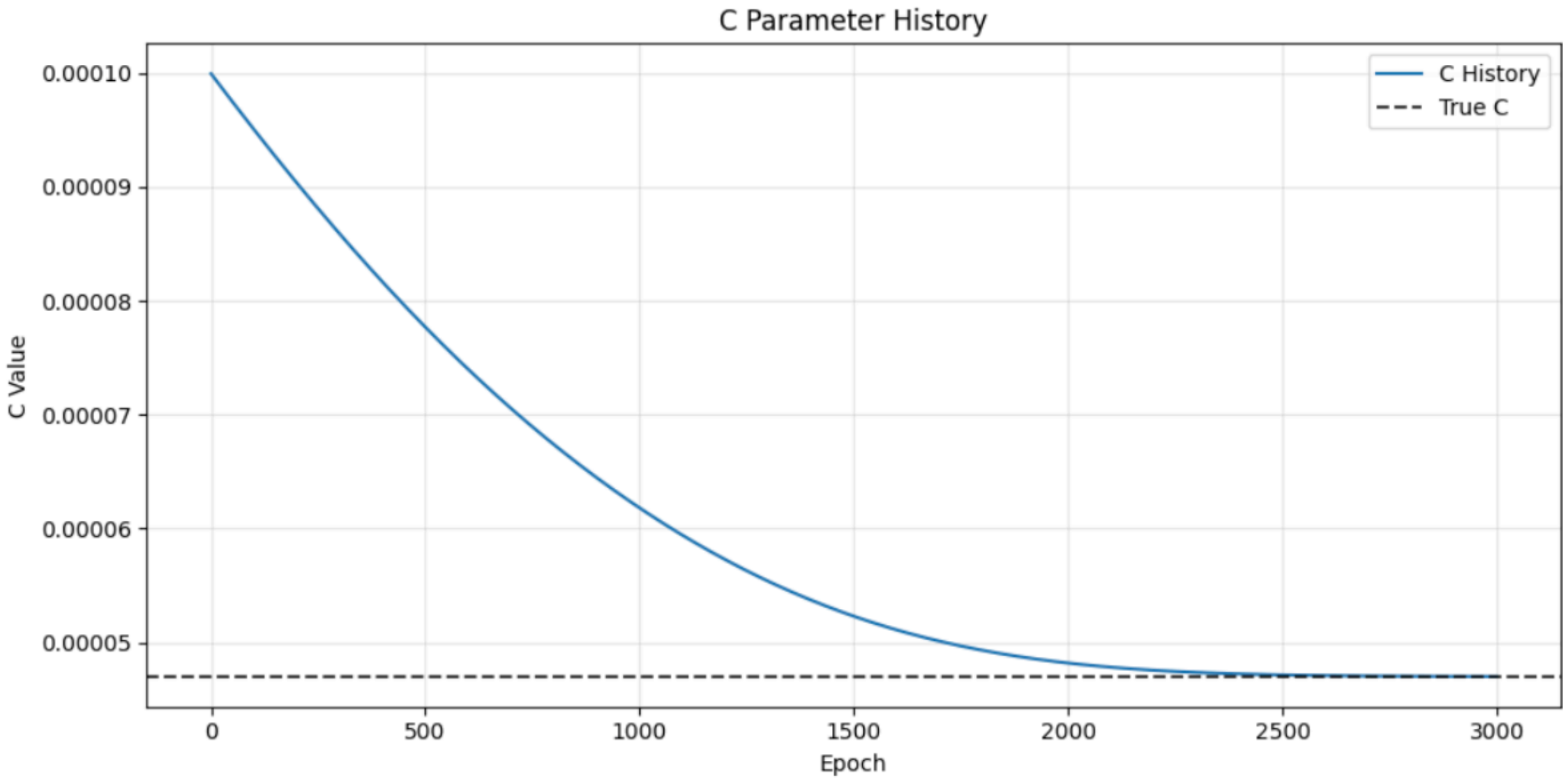
回路パラメータが負の値になってしまうことを防ぐ

```
class BuckConverterCell(nn.Module):
    def __init__(
        self,
        # 略
    ) -> None:
        super().__init__()
        self.dt = dt
        # パラメータを対数空間で学習（正の値を保証）
        self.log_L = nn.Parameter(torch.log(torch.tensor(L_init)))
        self.log_C = nn.Parameter(torch.log(torch.tensor(C_init)))
        self.log_R = nn.Parameter(torch.log(torch.tensor(R_init)))

    def forward(self, h: torch.Tensor, x: torch.Tensor) -> torch.Tensor:
        i_L, v_C = h[:, 0], h[:, 1]
        v_a = x[:, 0]
        L = torch.exp(self.log_L)
        C = torch.exp(self.log_C)
        R = torch.exp(self.log_R)

        # 略

        return torch.stack([i_L_next, v_C_next], dim=1)
```

結果(1つのシミュレーション波形、エポック3000)




R Parameter History
10.0
9.5
9.0
8.5
8.0
R Value
0
500
1000
1500
2000
2500
3000
Epoch
R History
True R


C Parameter History
C History
True C
C Value
0.00010
0.00009
0.00008
0.00007
0.00006
0.00005
0
500
1000
1500
2000
2500
3000
Epoch

課題

1. 先行研究のソースコード(DAB コンバータ)では、log 変換から exp 変換を行っていなかったのに、回路パラメータが負になっていなかった。おそらく、Clamper を使用しているから

1. 元の Clamper の範囲: `[(10e-6, 200e-6), (1e-3, 5e0), (0.8, 1.2)]`
2. 広くした Clamper の範囲: `[(-100e-6, 200e-6), (-5, 5), (-1.2, 1.2)]`

Clamper の範囲を広くするだけなら、回路パラメータは負の値にならず、おおよそ正しいパラメータになった

`[6.29e-05, 1.2857785, 0.9992431]`

Clamper の範囲を広くするかつ、回路パラメータの初期値を変更したら、回路パラメータが負になってしまうことがあった

回路パラメータの初期値: `RL, Lr, n = 8, 200e-6, 2`
回路パラメータ: `[0.0001165, -1.7660956, 1.2]`

何らかの制約によって、回路パラメータが負になってしまうのを防ぐ必要がある

2. 先行研究のソースコード(DAB コンバータ)では、Epoch 数が 75 でも損失が 0.012 になっていた

1. 仮説: 学習データが自分のコードだと 1 つだけだが、先行研究のコードでは 12 個あるから
2. 仮説: 学習率が全ての回路パラメータで同じ値にしていたが、先行研究のコードでは、回路パラメータごとに学習率を設定していたから
    i. 各パラメータのオーダーが異なるので、学習率を変える必要がある
3. 仮説: DAB の方が Buck よりも学習が簡単であるから
4. ~~仮説: 初期値が自分の設定したものより正解値に近かったから~~

NA

1. 先行研究のコードで、Clamper の範囲を広く(負の範囲も含めて)設定して学習させてみる。
    i. どの初期値、どの Clamper の範囲を使ったら、回路パラメータが負にならないかを調べる
2. 先行研究のコードで、回路パラメータの初期値を正解値から遠ざけて学習させてみる。
3. 学習データを増やす
    i. 入力電圧やスイッチング周波数を変えて学習データを増やす
4. 学習率を回路パラメータごとに変えて学習させてみる。