

# Extended Kalman Filter-Based SOC Estimation with Online Parameter Identification Using Adaptive Forgetting Factor Recursive Least Square Algorithm

## Referensi:

1. "Adaptive Forgetting Factor Recursive Least Square Algorithm for Online Identification of Equivalent Circuit Model Parameters of a Lithium-Ion Battery" oleh Xiangdong Sun et al., 2019 (MDPI Energies)
2. "State of Charge Estimation of Lithium-Ion Battery Based on Improved Forgetting Factor Recursive Least Squares-Extended Kalman Filter Joint Algorithm" oleh Caian Ge et al., 2022 (Elsevier)

Sistem Kendali Prediktif Adaptif

Dr. Ir. Aries Subiantoro., M.S.E.E.

Departemen Teknik Elektro

Fakultas Teknik, Universitas Indonesia



**Davin Nazhif Wilviadli**  
2206055643



**Afif Darda Narendra**  
2206055662



**Haidar Satrio Wibowo**  
2206055712

# Tantangan Estimasi SoC dan Solusi yang Diusulkan

## Situation

Teknologi baterai semakin **banyak digunakan**, terutama pada kendaraan listrik dan sistem energi terbarukan.

Agar aman dan efisien, diperlukan Battery Management System (BMS) yang **akurat**, khususnya dalam hal **estimasi State of Charge**.

Berbagai pendekatan estimasi SoC telah dikembangkan, termasuk metode berbasis **model dan data**.

## Complication

Metode estimasi SoC konvensional seperti kalman filter memiliki **kelemahan** dalam kondisi **dinamis dan perubahan parameter baterai**.

Parameter dari model baterai (seperti R<sub>0</sub>, R<sub>1</sub>, R<sub>2</sub>, C<sub>1</sub>, C<sub>2</sub>) dapat **berubah** terhadap **waktu, suhu, dan siklus penggunaan**.

Algoritma dengan *forgetting factor* tetap (seperti RLS standar) **sulit beradaptasi** dengan perubahan karakteristik baterai dalam jangka panjang.

## Question

Bagaimana **mengembangkan** metode estimasi SoC berbasis model RC orde-2 yang dapat **menyesuaikan diri terhadap perubahan parameter baterai** secara **real-time** dan mempertahankan akurasi tinggi dalam **berbagai kondisi pengoperasian**?

## Answer

Paper ini mengusulkan penggunaan metode **Adaptive Forgetting Factor Recursive Least Squares (AFFRLS)** untuk estimasi parameter model orde-2 baterai.

Dengan memanfaatkan model orde-2 dari sistem RC dan menambahkan mekanisme adaptif pada forgetting factor, metode ini mampu **mengakomodasi perubahan parameter secara akurat**.

Estimasi parameter RC **dikombinasikan** dengan **model Extended Kalman Filter** untuk menghitung nilai SoC secara tidak langsung.

Hasil simulasi menunjukkan bahwa metode AFFRLS **mampu melacak perubahan parameter RC** secara **real-time** dan meningkatkan akurasi estimasi SoC dibandingkan metode RLS konvensional.

# Spesifikasi Baterai dan Data yang Digunakan



## A123 Battery Specification

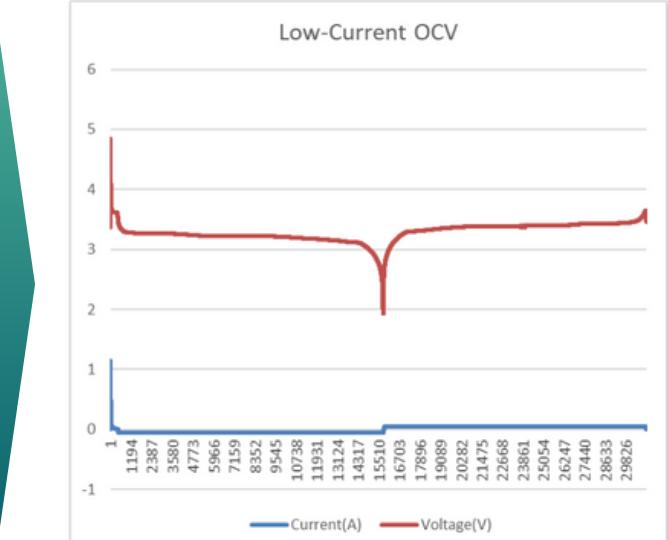
Battery (Parameters)	Specifications (Value)
Capacity Rating	1100 mAh
Cell Chemistry	LiFePO4
Diameter	18 mm
Length	65 mm
Special Notes	Tab length not included in dimensions

Data yang digunakan pada percobaan ini adalah data eksperimen yang dilakukan pada **suhu 20°C**.

## Low-Current OCV Data

Step_Time(s)	Current(A)	Voltage(V)
300.781	0	3.370.858
0,00	1.151.195	3.688.785
5.007.662	78.517	3.601.378
1.001.515	756.315	360.107
1.502.671	730.364	360.107
200.343	706.409	360.107
2.504.185	683.361	3.600.763

⋮

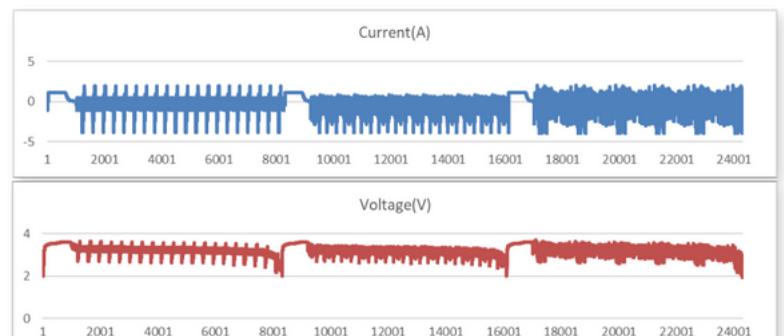


Kami menggunakan data ini untuk membuat persamaan polinomial orde-9 OCV-SOC. Persamaan ini akan **menunjukkan besaran Uocv untuk setiap persen SOC** pada saat *charging* dan *discharging*.

## Dynamic Stress Test Data

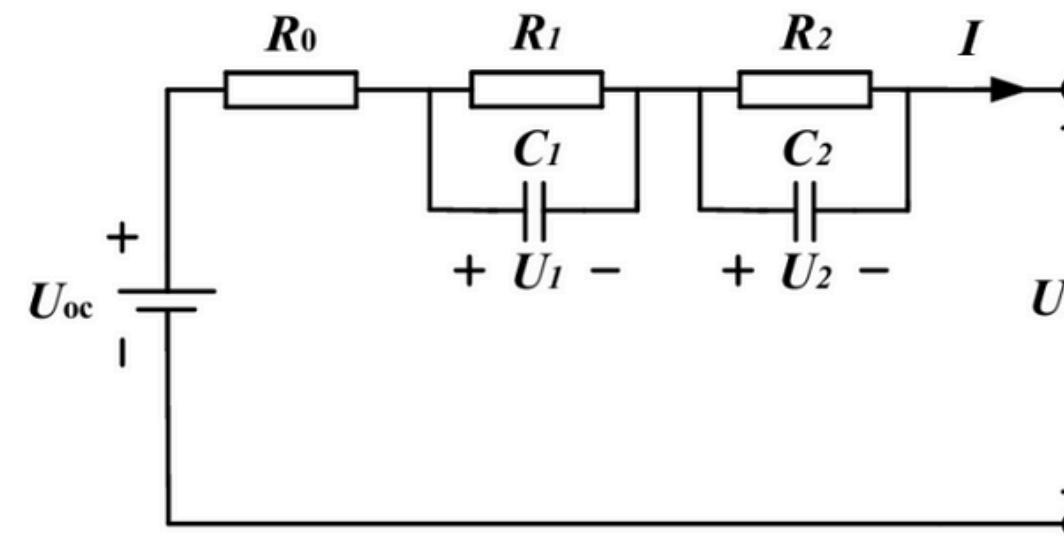
Test_Time(s)	Step_Time(s)	Step_Index	Current(A)	Voltage(V)
3.009.034	3.009.034	1	0	3.011.998
8.014.829	5.005.484	2	-109.937	2.806.096
1.301.605	100.067	2	-109.937	278.763
180.191	1.500.975	2	-109.937	2.770.087
2.302.135	2.001.201	2	-109.937	2.753.159
2.802.568	2.501.634	2	-109.937	2.735.924

⋮



Data ini terbagi menjadi 3 kondisi (DST, US06, dan FUDS). Hanya data DST yang digunakan untuk **optimasi parameter theta** pada AFFRLS.

# 2nd Order Thevenin's Model untuk Baterai Lithium-Ion



Dengan menggunakan **Kirchoff's Voltage Law** dan **Kirchoff's Current Law**

$$U_L = U_{oc}[SOC(t)] - U_1 - U_2 - I(t) \cdot R_0$$

$$C_1 \cdot \frac{dU_1}{dt} = I(t) - \frac{U_1}{R_1}$$

$$C_2 \cdot \frac{dU_2}{dt} = I(t) - \frac{U_2}{R_2}$$



Persamaan 1

$$E = U_L(s) - U_{oc}(s) = -I(s) \left( R_0 + \frac{R_1}{1 + R_1 C_1 s} + \frac{R_2}{1 + R_2 C_2 s} \right)$$

Persamaan 1 dapat ditulis ulang menjadi sebuah fungsi alih

$$G(s) = \frac{E(s)}{I(s)} = -\frac{R_0 s^2 + \frac{R_0 R_1 C_1 + R_0 R_2 C_2 + R_2 R_1 C_1 + R_1 R_2 C_2}{R_1 C_1 R_2 C_2} s + \frac{R_0 + R_1 + R_2}{R_1 C_1 R_2 C_2}}{s^2 + \frac{R_1 C_1 + R_2 C_2}{R_1 C_1 R_2 C_2} s + \frac{1}{R_1 C_1 R_2 C_2}}$$

Diskritisasi fungsi alih menjadi model diskrit dengan **Transformasi Bilinear**

$$s = \frac{2}{T} \cdot \frac{1-z^{-1}}{1+z^{-1}}$$

Persamaan 2

$$G(z^{-1}) = \frac{E(k)}{I(k)} = \frac{\theta_3 + \theta_4 z^{-1} + \theta_5 z^{-2}}{1 - \theta_1 z^{-1} - \theta_2 z^{-2}}$$

Dimana,

$$\theta_1 = \frac{2T^2 - 8R_1 C_1 R_2 C_2}{-T^2 - 2T(R_1 C_1 + R_2 C_2) - 4R_1 C_1 R_2 C_2}$$

$$\theta_2 = \frac{T^2 - 2T(R_1 C_1 + R_2 C_2) + 4R_1 C_1 R_2 C_2}{-T^2 - 2T(R_1 C_1 + R_2 C_2) - 4R_1 C_1 R_2 C_2}$$

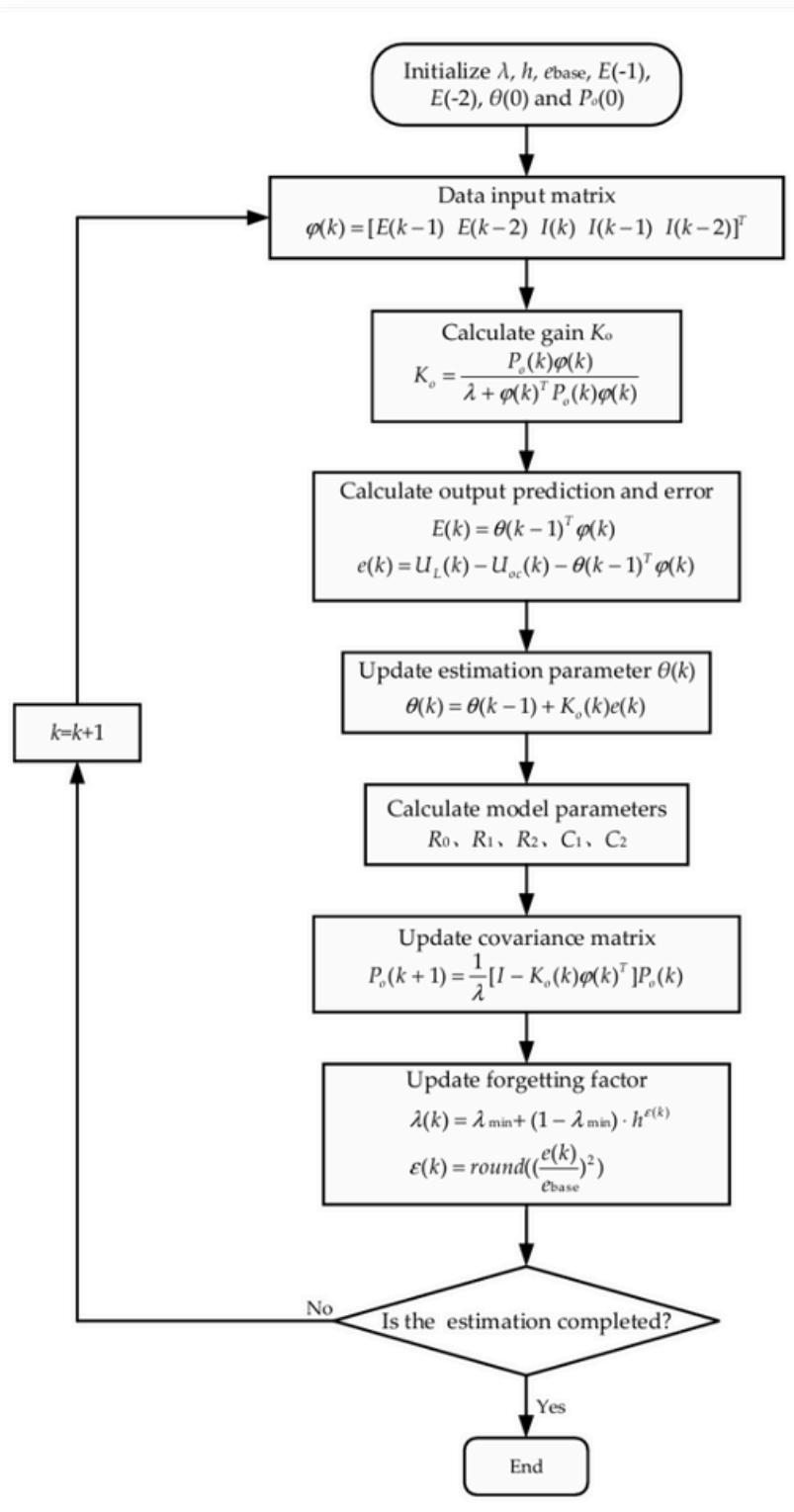
$$\theta_3 = \frac{T^2(R_0 + R_1 + R_2) + 2T(R_0 R_1 C_1 + R_0 R_2 C_2 + R_1 R_2 C_2 + R_2 R_1 C_1) + 4R_0 R_1 C_1 R_2 C_2}{-T^2 - 2T(R_1 C_1 + R_2 C_2) - 4R_1 C_1 R_2 C_2}$$

$$\theta_4 = \frac{2T^2(R_0 + R_1 + R_2) - 8R_0 R_1 C_1 R_2 C_2}{-T^2 - 2T(R_1 C_1 + R_2 C_2) - 4R_1 C_1 R_2 C_2}$$

$$\theta_5 = \frac{T^2(R_0 + R_1 + R_2) - 2T(R_0 R_1 C_1 + R_0 R_2 C_2 + R_1 R_2 C_2 + R_2 R_1 C_1) + 4R_0 R_1 C_1 R_2 C_2}{-T^2 - 2T(R_1 C_1 + R_2 C_2) - 4R_1 C_1 R_2 C_2}$$

# Adaptive Forgetting Factor RLS untuk Identifikasi Theta

## Flowchart Algoritma AFFRLS



Dari persamaan 2 yang berbentuk fungsi alih diskrit, dapat dibentuk persamaan rekursif,

$$G(z^{-1}) = \frac{E(k)}{I(k)} = \frac{\theta_3 + \theta_4z^{-1} + \theta_5z^{-2}}{1 - \theta_1z^{-1} - \theta_2z^{-2}}$$

$$E(k) = \theta_1E(k-1) + \theta_2E(k-2) + \theta_3I(k) + \theta_4I(k-1) + \theta_5I(k-2) + e(k)$$

Persamaan rekursif dapat ditulis dalam persamaan matrix dengan fungsi objektif sebagai berikut,

**Matrix Form**

$$E = \varphi\theta + e$$

**Fungsi Objektif**

$$J = \sum_{t=0}^m [e(k-t)]^2 = e^T e$$

Dimana,

$$\varphi = \begin{bmatrix} E(k-1) & E(k-2) & I(k) & I(k-1) & I(k-2) \\ E(k-2) & E(k-3) & I(k-1) & I(k-2) & I(k-3) \\ & & \vdots & & \\ E(k-m-1) & E(k-m-2) & I(k-m) & I(k-m-1) & I(k-m-2) \end{bmatrix} \quad \theta = [\theta_1 \theta_2 \theta_3 \theta_4 \theta_5]^T$$

$$e = [e(k) e(k-1) \cdots e(k-m)]^T$$

Berdasarkan fungsi objektif di atas, **parameter  $\theta(k)$  akan diperbarui di setiap iterasi dengan persamaan,**

$$\theta(k) = \theta(k-1) + K_o(k)[U_L(k) - U_{oc}(k) - \theta(k-1)^T\varphi(k)]$$

dengan,

$$K_o = \frac{P_o(k-1)\varphi(k)}{\lambda + \varphi(k)^T P_o(k-1)\varphi(k)}$$

$$P_o(k) = \frac{1}{\lambda}[I - K_o(k)\varphi(k)^T]P_o(k-1)$$

# Levenberg-Marquardt untuk Estimasi Parameter RC

Algoritma ini digunakan untuk **menyelesaikan** permasalahan **pencarian akar** dari lima buah **persamaan nonlinier**.

## 1 Hitung error

Tujuannya adalah meminimasi fungsi

$$\mathbf{e} = \boldsymbol{\theta}_{\text{model}} - \boldsymbol{\theta}_{\text{exp}}$$

$$\min_{\mathbf{p}} \|\boldsymbol{\theta}_{\text{model}}(\mathbf{p}) - \boldsymbol{\theta}_{\text{exp}}\|^2$$

## 2 Hitung Jacobian

$$J_{ij} = \frac{f_i(p_j + \delta) - f_i(p_j)}{\delta}$$

## 3 Hitung Hessian dan Vektor Gradien

$$\mathbf{H} = \mathbf{J}^\top \mathbf{J} + \mu \mathbf{I}$$

$$\mathbf{g} = \mathbf{J}^\top \mathbf{e}$$

4

## Update Parameter

$$\Delta \mathbf{p} = \mathbf{H}^{-1} \mathbf{g} \quad \boldsymbol{\theta}_{\text{new}} = \boldsymbol{\theta}_{\text{old}} - \Delta \boldsymbol{\theta}^\top$$

$$\mathbf{p}_{\text{new}} = \mathbf{p} - \Delta \mathbf{p} \quad \mathbf{e}_{\text{new}} = \boldsymbol{\theta}_{\text{model}}^{\text{new}} - \boldsymbol{\theta}_{\text{exp}}$$

5

## Uji Konvergensi

$$\text{Jika } \|\mathbf{e}_{\text{new}}\| < \|\mathbf{e}\| : \begin{cases} \mathbf{p} \leftarrow \mathbf{p}_{\text{new}} \\ \mu \leftarrow \frac{\mu}{\beta}, \quad \beta > 1 \end{cases}$$

$$\text{Jika } \|\mathbf{e}_{\text{new}}\| \geq \|\mathbf{e}\| : \begin{cases} \mathbf{p} \leftarrow \mathbf{p} \quad (\text{tetap}) \\ \mu \leftarrow \mu \cdot \beta \end{cases}$$

6

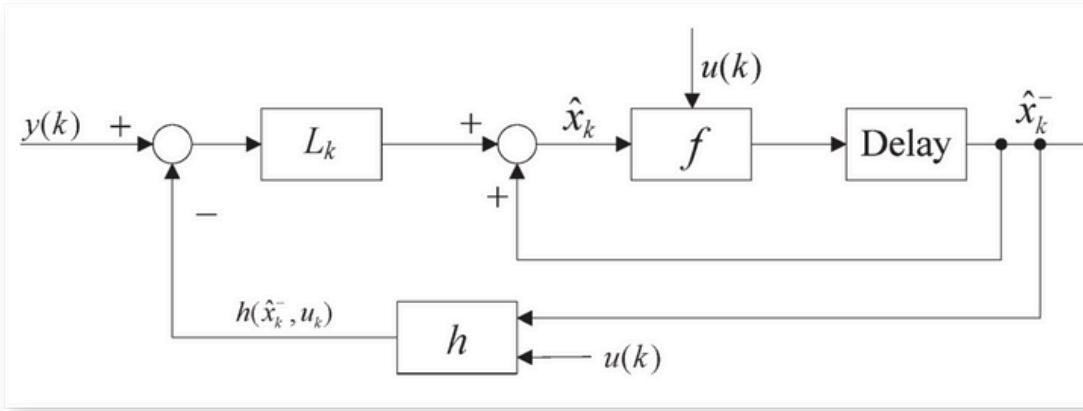
## Hentikan Iterasi Saat

$$\|\mathbf{e}_{\text{new}}\| < \|\mathbf{e}\| \quad \text{dan} \quad \|\mathbf{e}_{\text{new}} - \mathbf{e}\| < \text{tol}$$



# Extended Kalman Filter untuk Estimasi SoC

## Block Diagram Extended Kalman Filter



EKF merupakan versi dari Kalman Filter untuk menyelesaikan permasalahan **sistem yang nonlinier**.

## Nonlinear State Space Model

$$\begin{cases} x(t+1) = f(x(t), u(t)) + w(t) \\ y(t) = h(x(t)) + v(t) \end{cases}$$

## Jacobian Matrix

$$A_{k-1} = \frac{\partial f(x_{k-1}, u_{k-1})}{\partial x_{k-1}} \Big|_{x_{k-1}=\hat{x}_{k-1}}$$

$$C_k = \frac{\partial h(x_k, u_k)}{\partial x_k} \Big|_{x_k=\hat{x}_k^-}$$

Parameter-parameter yang digunakan pada perhitungan EKF

$$\begin{aligned} \hat{x}_k^- &= f(\hat{x}_{k-1}, u_{k-1}), \\ P_k^- &= A_{k-1}P_{k-1}A_{k-1}^T + Q(t), \\ L_k &= P_k^- C_k^T [C_k P_k^- C_k^T + R(t)]^{-1}, \\ \hat{x}_k &= \hat{x}_k^- + L_k [y_k - h(\hat{x}_k^-, u_k)], \\ P_k &= (I - L_k C_k) P_k^-, \end{aligned}$$

State estimate time update  
Error covariance time update  
Kalman gain matrix  
State estimate measurement update  
Error covariance measurement update

## Langkah-Langkah Estimasi Parameter Menggunakan EKF

- Prediksi keadaan sistem ke waktu berikutnya
- Perbarui ketidakpastian estimasi
- Melakukan kalkulasi untuk mendapatkan *Kalman Gain*
- Estimasi yang sudah dibandingkan dengan error saat ini
- Memperbarui error covariance untuk saat ini

Pada setiap iterasinya, parameter SoC diperbaharui menggunakan persamaan berikut,

$$\hat{x}_k = \hat{x}_k^- + L_k \cdot \left( y_k - \hat{v}_{1,k}^- - \hat{v}_{2,k}^- - iR_0 - \frac{\partial f(\hat{s}_k^-)}{\partial \hat{s}_k^-} \right)$$

dimana,

$$\frac{\partial f(s)}{\partial s} = \text{Koreksi Model Nonlinear OCV untuk SoC}$$

$y_k$  = Tegangan Aktual Baterai

$i \cdot R_0$  = Tegangan Drop Resistansi Internal Baterai

$\hat{v}_{1,k}^-$ ,  $\hat{v}_{2,k}^-$  = Tegangan Prediksi pada C1 dan C2

Turunan persaman didapat dengan menurunkan persamaan polinom Uocv(SoC) orde-9

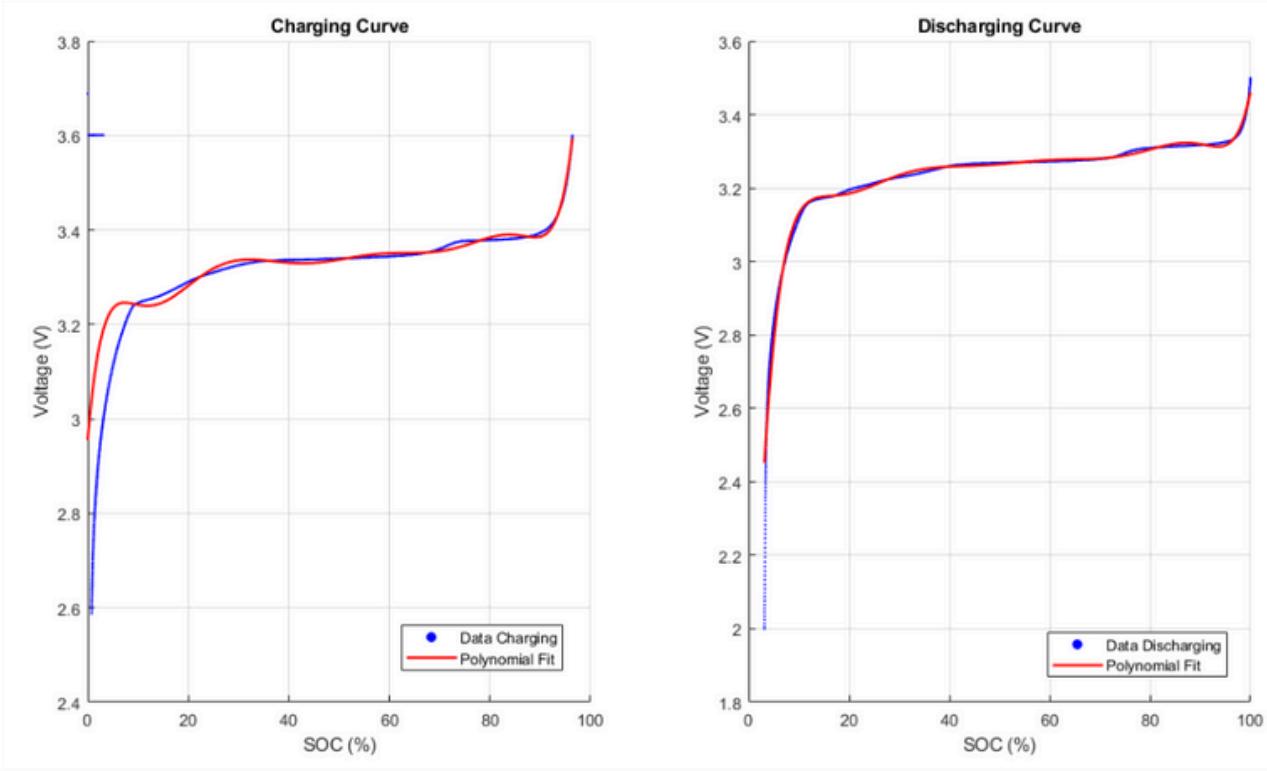
$$\frac{\partial f(s)}{\partial s} = 9a_1s^8 + 8a_2s^7 + 7a_3s^6 + 6a_4s^5 + 5a_5s^4 + 4a_6s^3 + 3a_7s^2 + 2a_8s + a_9$$

# Kurva OCV-SoC serta Mapping Uocv Data DST

**Coulomb-Counting** dilakukan pada data arus OCV, hasilnya adalah data SoC. **Polynom fitting orde-9** dilakukan pada data ini untuk mendapatkan persamaan tegangan *open-circuit* sebagai fungsi SoC, **Uocv(SoC)**.

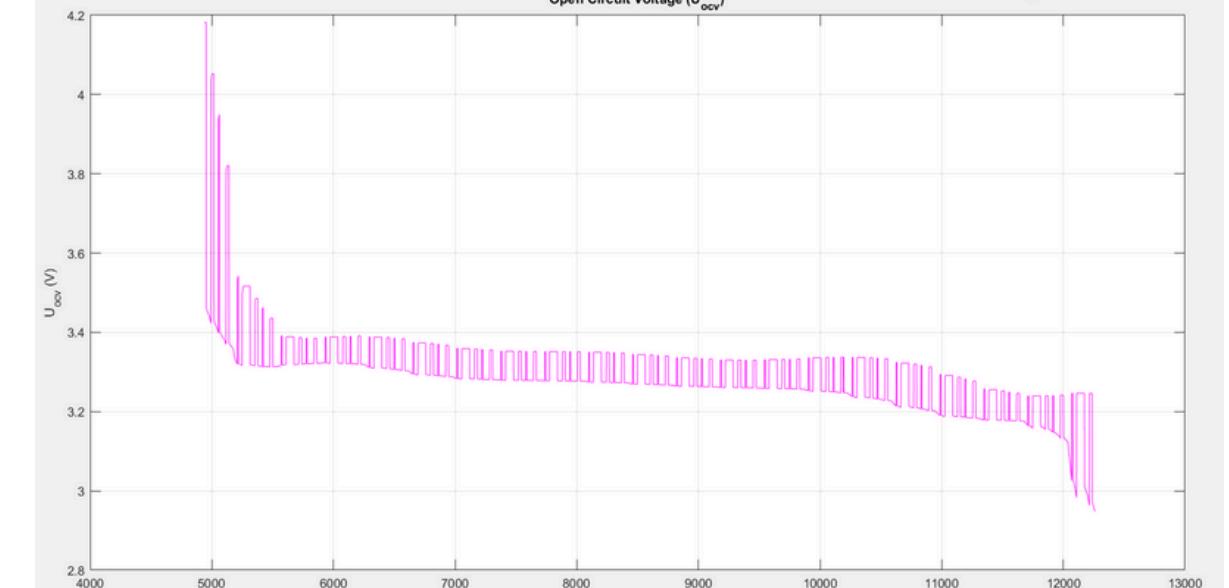
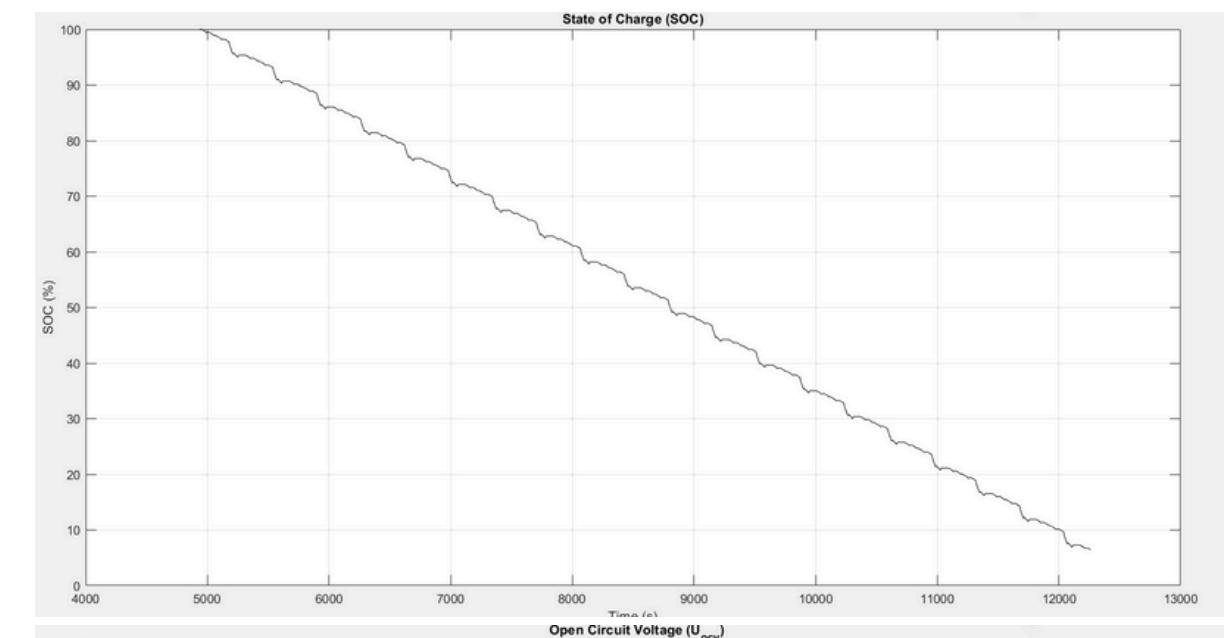
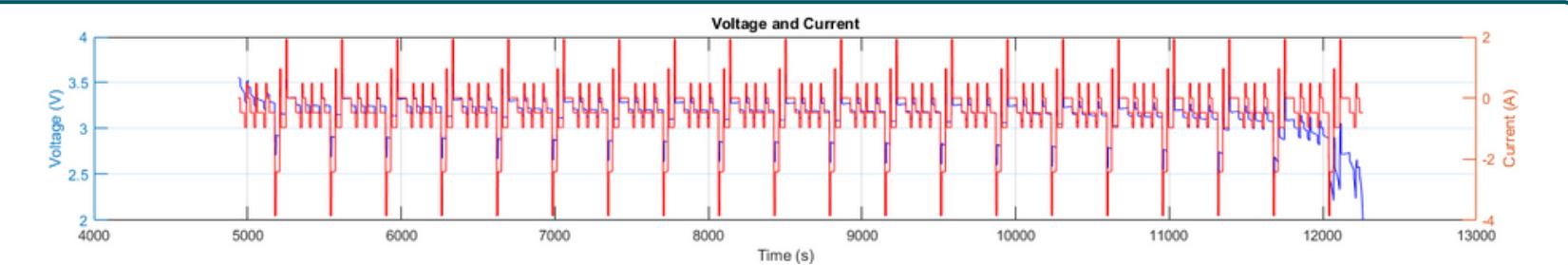
$$U_{\text{ocvCha}} = 3813 \cdot \text{SOC}^9 - 16906 \cdot \text{SOC}^8 + 31678 \cdot \text{SOC}^7 - 32618 \cdot \text{SOC}^6 + 20076 \cdot \text{SOC}^5 \\ - 7522 \cdot \text{SOC}^4 + 1671 \cdot \text{SOC}^3 - 205 \cdot \text{SOC}^2 + 12 \cdot \text{SOC} + 3$$

$$U_{\text{ocvDis}} = 3279 \cdot \text{SOC}^9 - 15807 \cdot \text{SOC}^8 + 32591 \cdot \text{SOC}^7 - 37545 \cdot \text{SOC}^6 + 26475 \cdot \text{SOC}^5 \\ - 11764 \cdot \text{SOC}^4 + 3265 \cdot \text{SOC}^3 - 541 \cdot \text{SOC}^2 + 49 \cdot \text{SOC} + 1$$



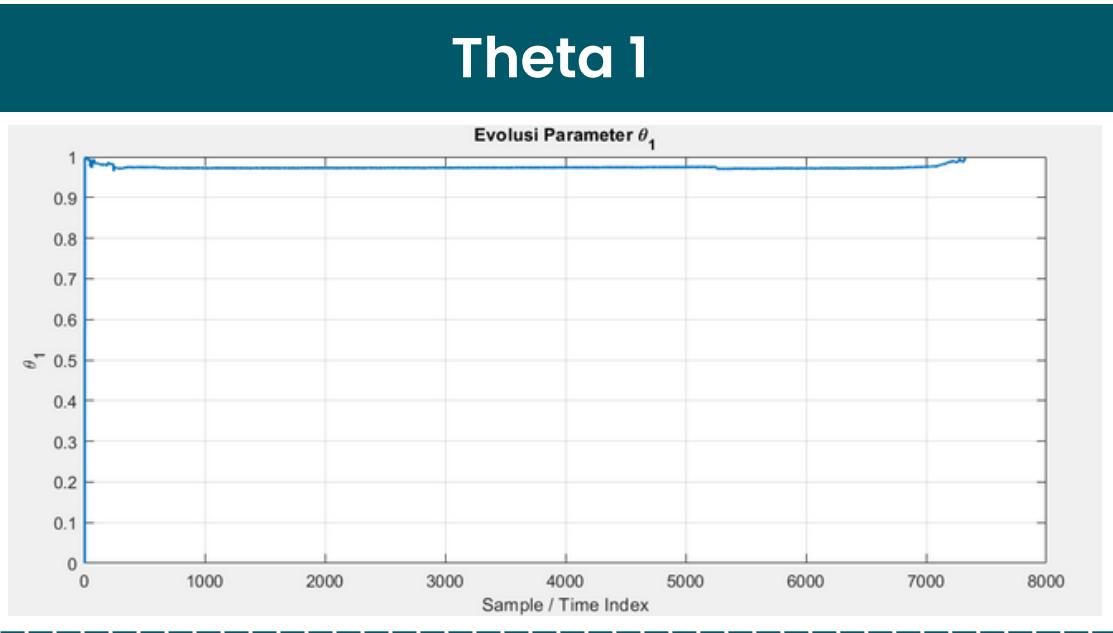
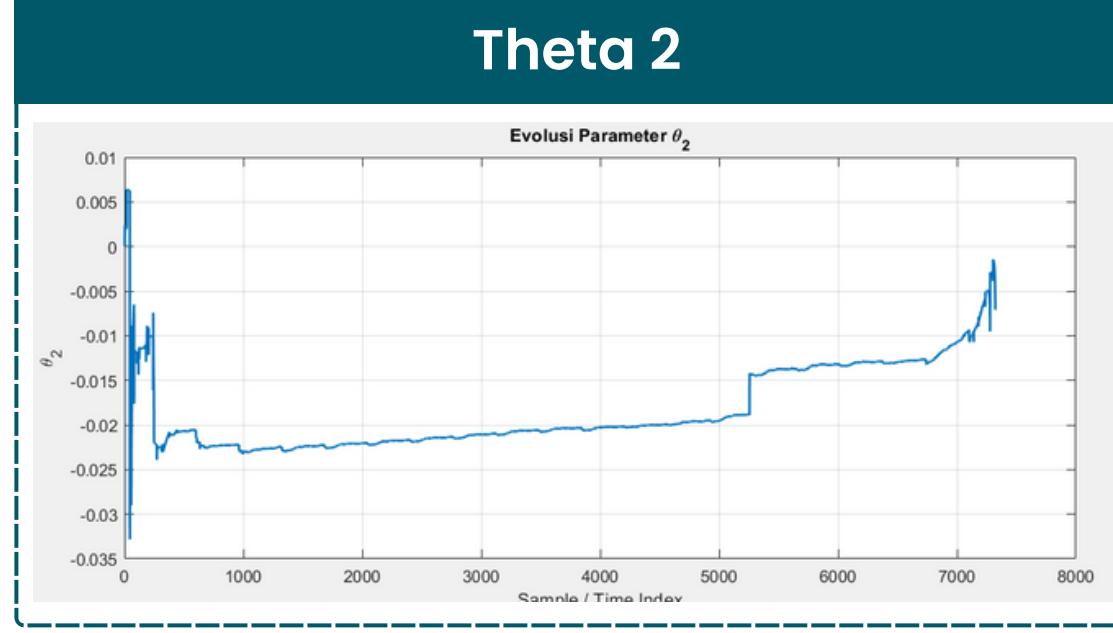
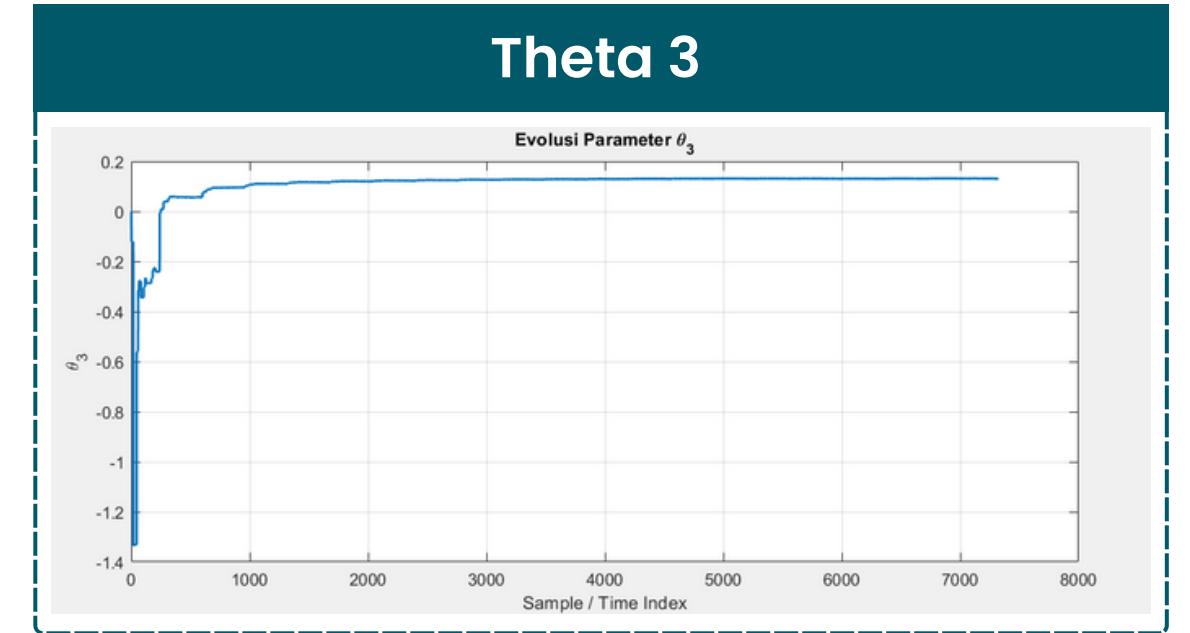
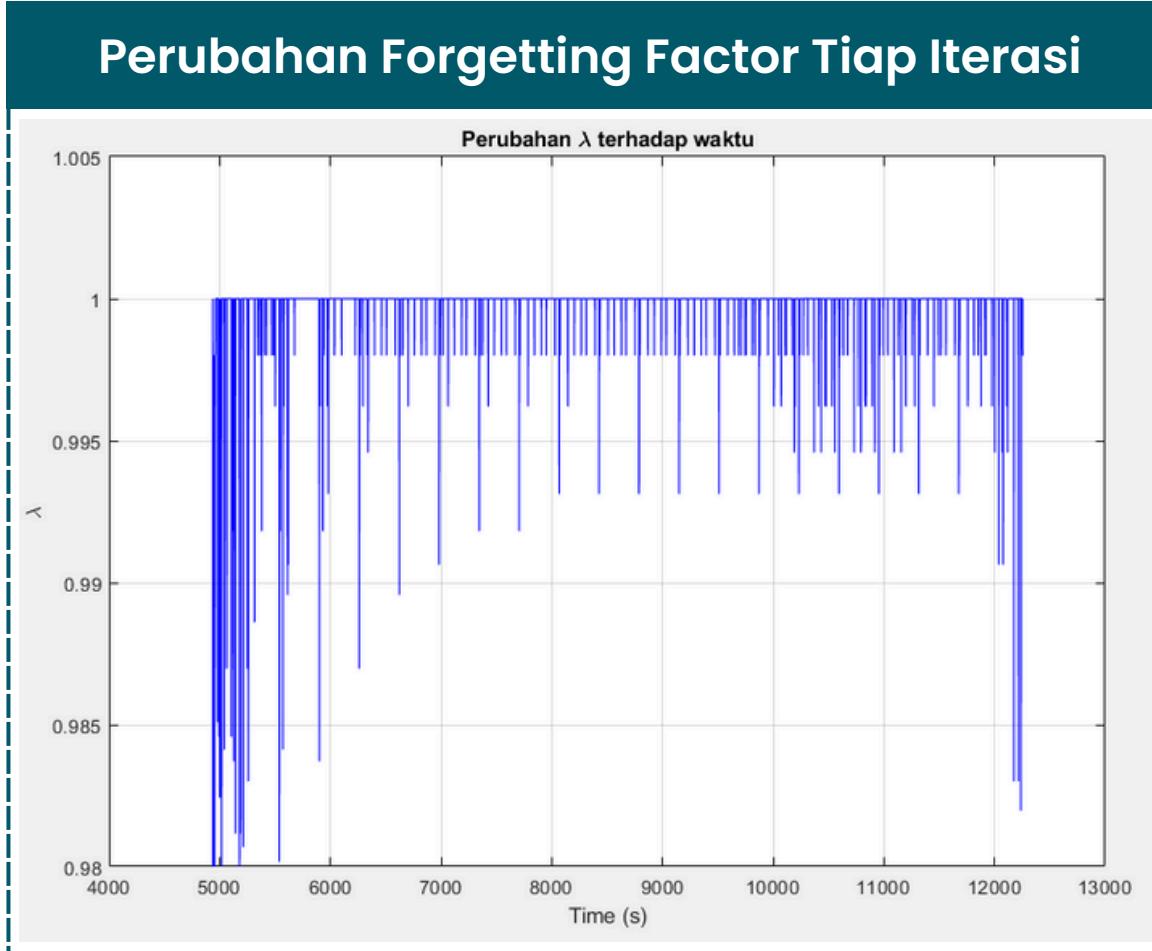
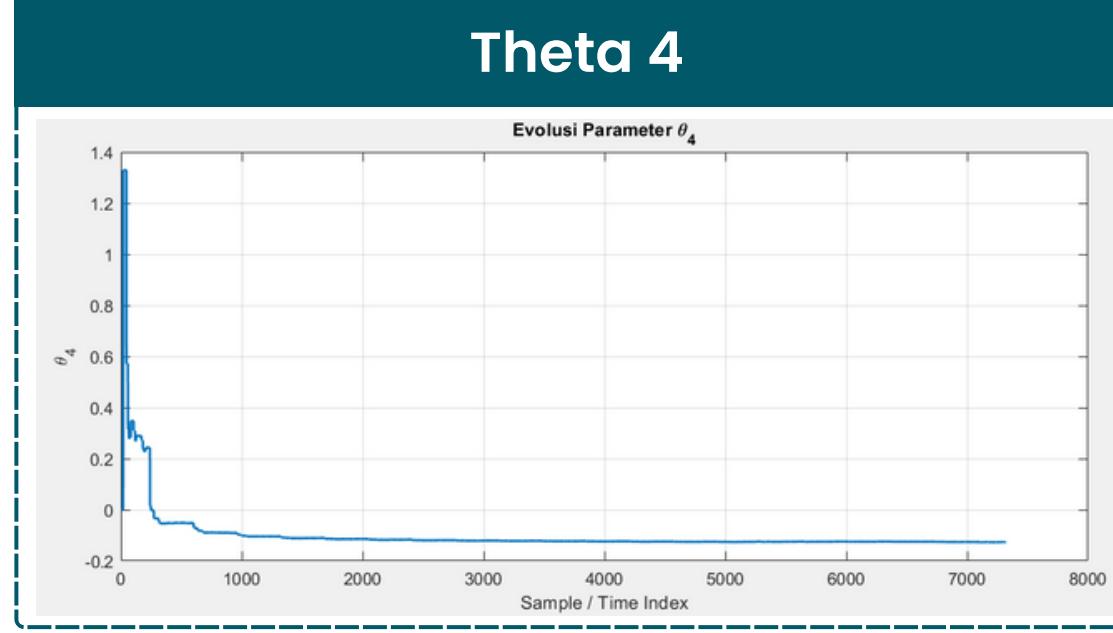
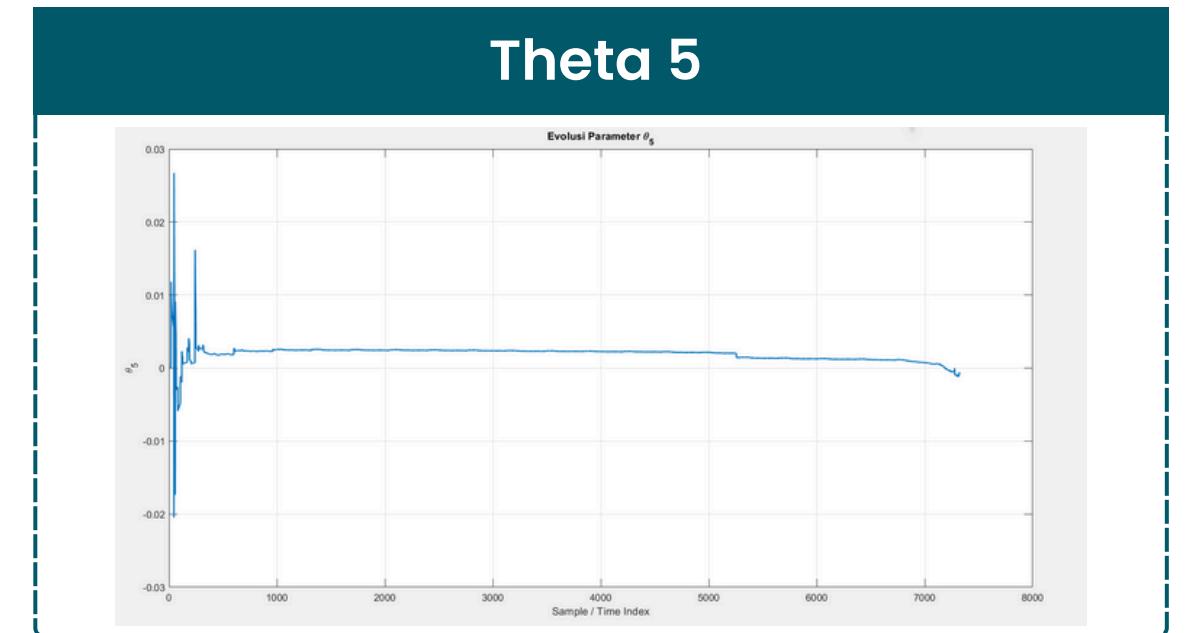
Polinom OCV-SoC digunakan untuk memetakan besaran Uocv dari data DST yang akan digunakan untuk AFFRLS.

## Arus dan Tegangan Data DST



SoC dan Besaran Uoc pada Waktu Tersebut

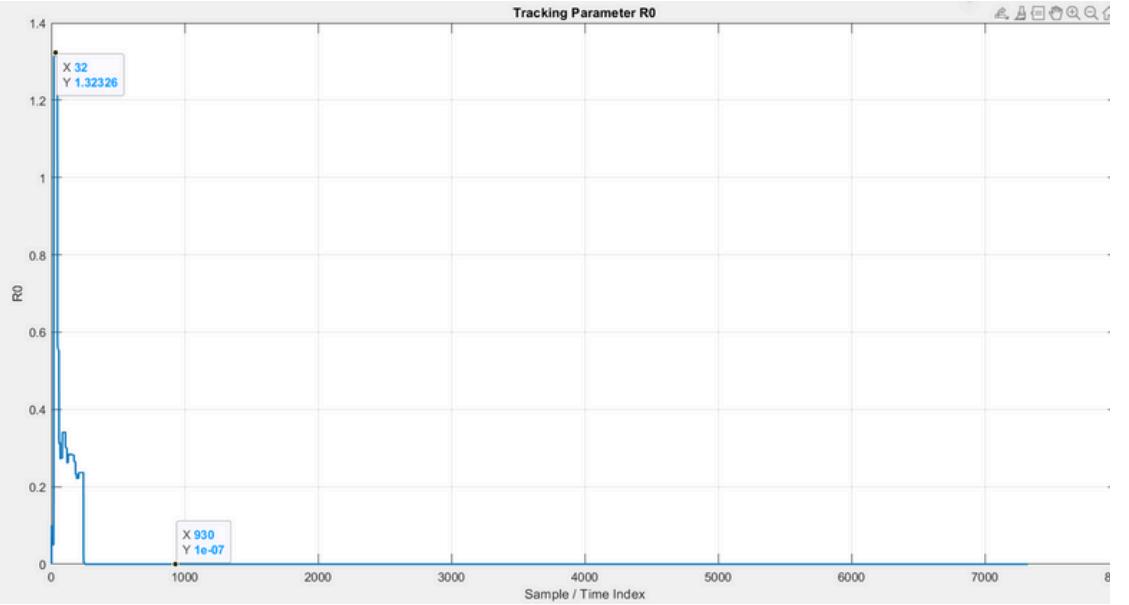
# Hasil Estimasi AFFRLS untuk Parameter Theta

**Theta 1****Theta 2****Theta 3****Perubahan Forgetting Factor Tiap Iterasi****Theta 4****Theta 5**

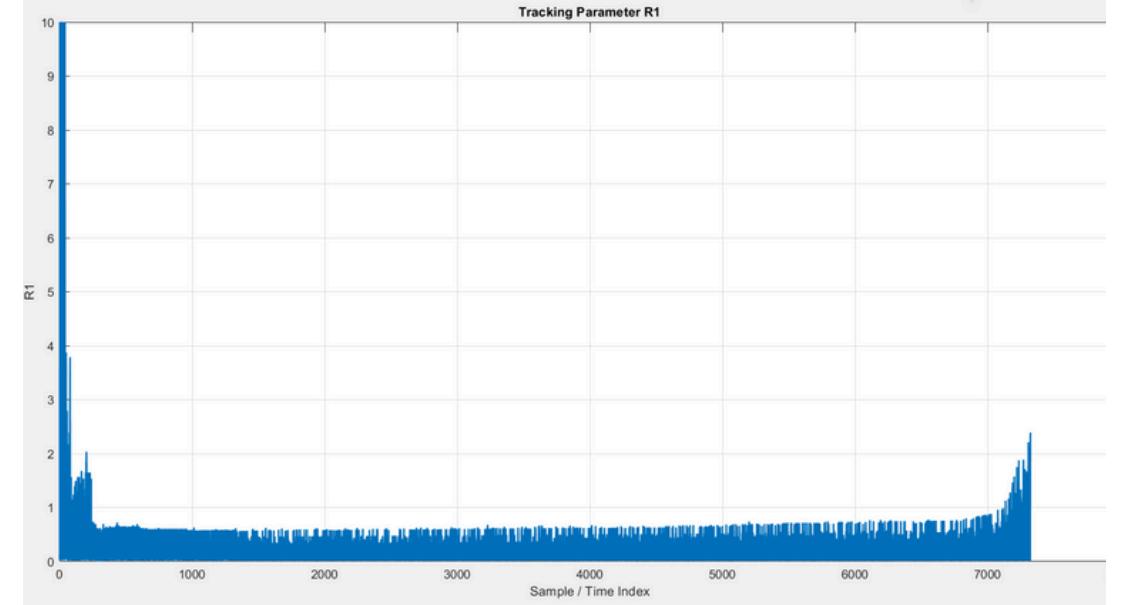
Parameter *forgetting factor*  $\lambda_{min} = 0.98$ , sensitivitas  $h = 0.9$ , dan batas error  $e_{base} = 0.05$  dipilih untuk **menyeimbangkan kecepatan dan akurasi identifikasi parameter** dalam AFFRLS. Nilai  $h$  yang mendekati 1 membuat perubahan  $\lambda$  lambat (stabil), sedangkan  $e_{base}$  menentukan seberapa responsif  $\lambda$  terhadap kesalahan identifikasi.

# Hasil Estimasi LM untuk Parameter RC Model Elektrikal

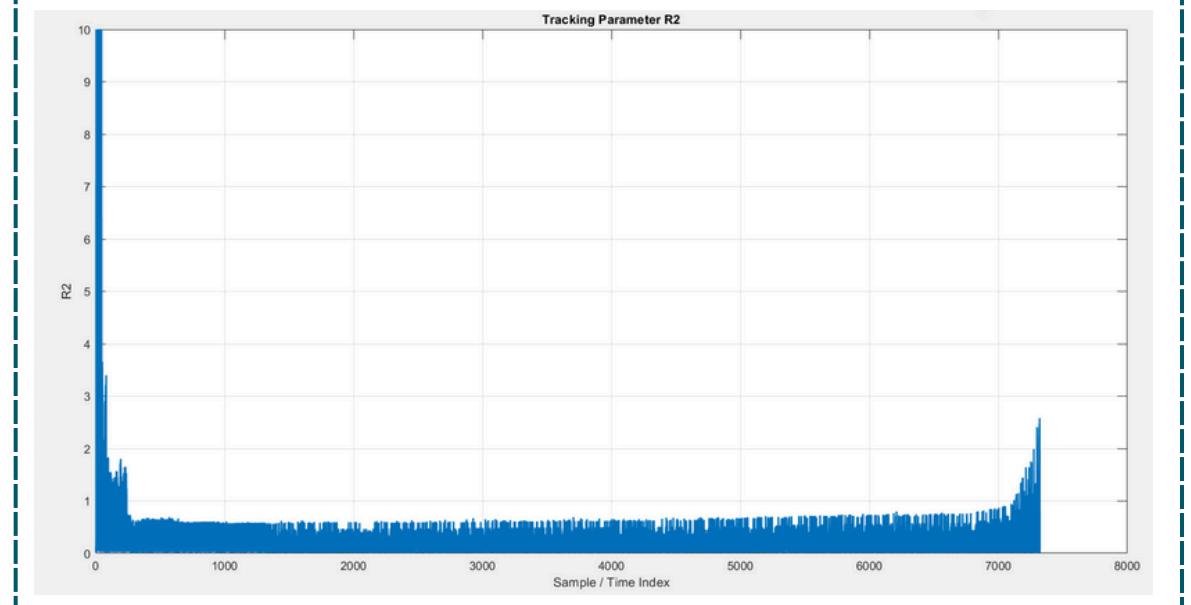
R0



R1

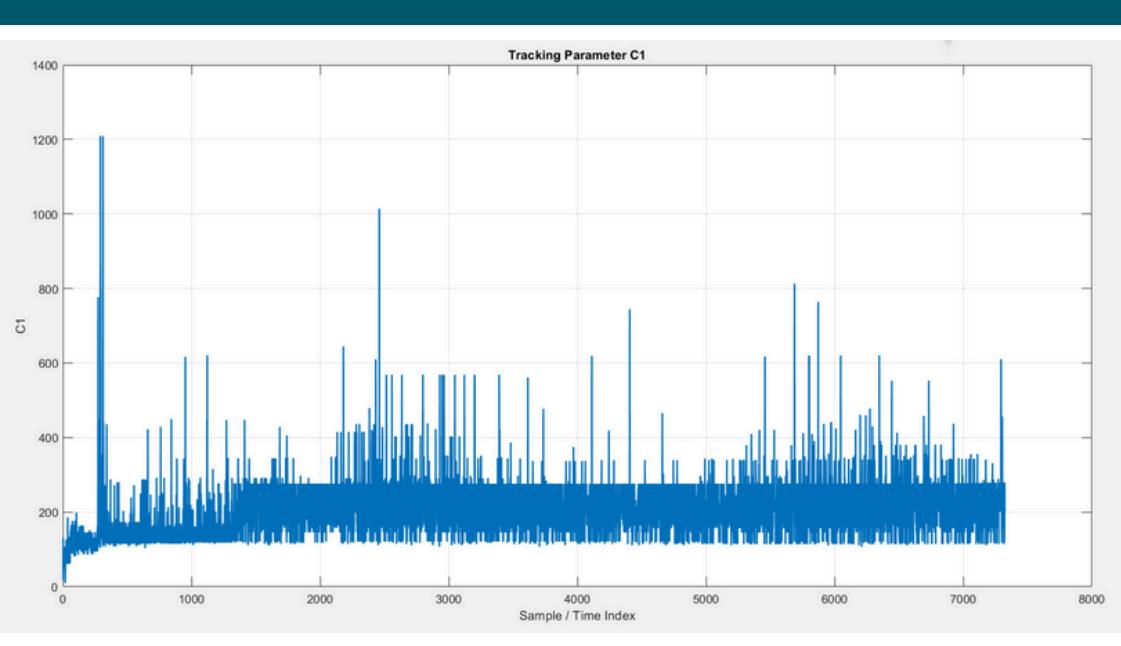


R2

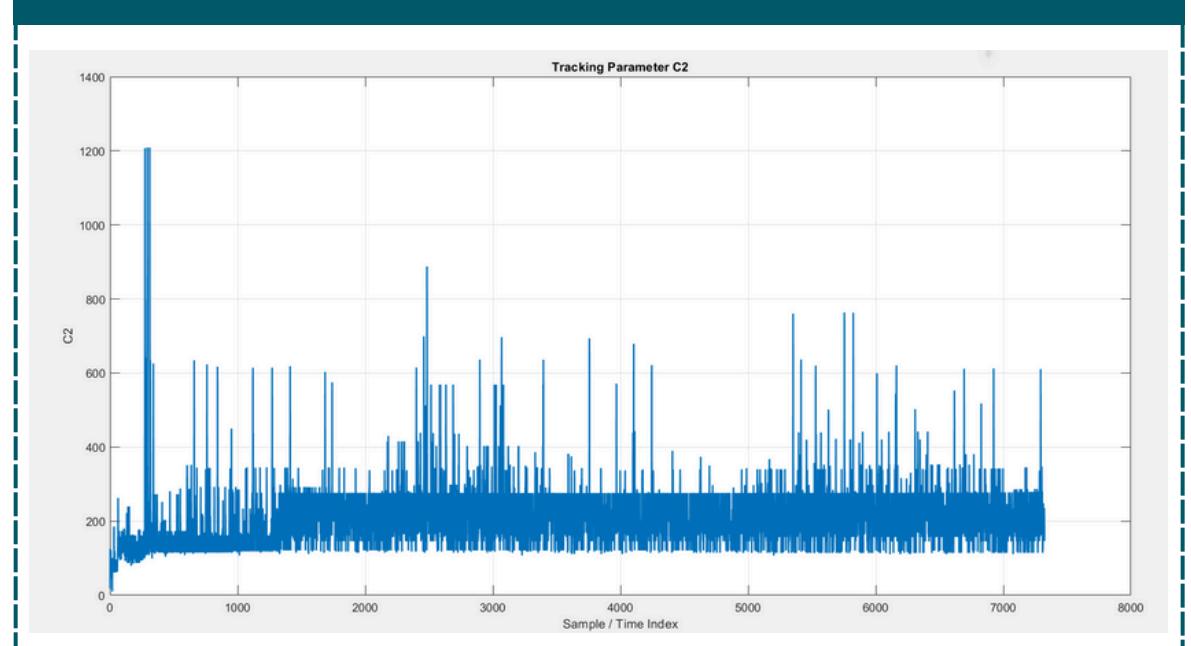


Setelah, *theta* (AFFRLS) didapat, algoritma LM digunakan untuk *mapping* antara RC terhadap *theta* (AFFRLS). Algoritma LM terlihat **dapat memprediksi nilai parameter RC** untuk tiap waktu **kecuali di parameter R0**. nilai R0 **menyentuh lower bound** pada perhitungan LM. Setelah proses ini, semua nilai parameter RC digunakan untuk memprediksi SoC dengan EKF di step selanjutnya.

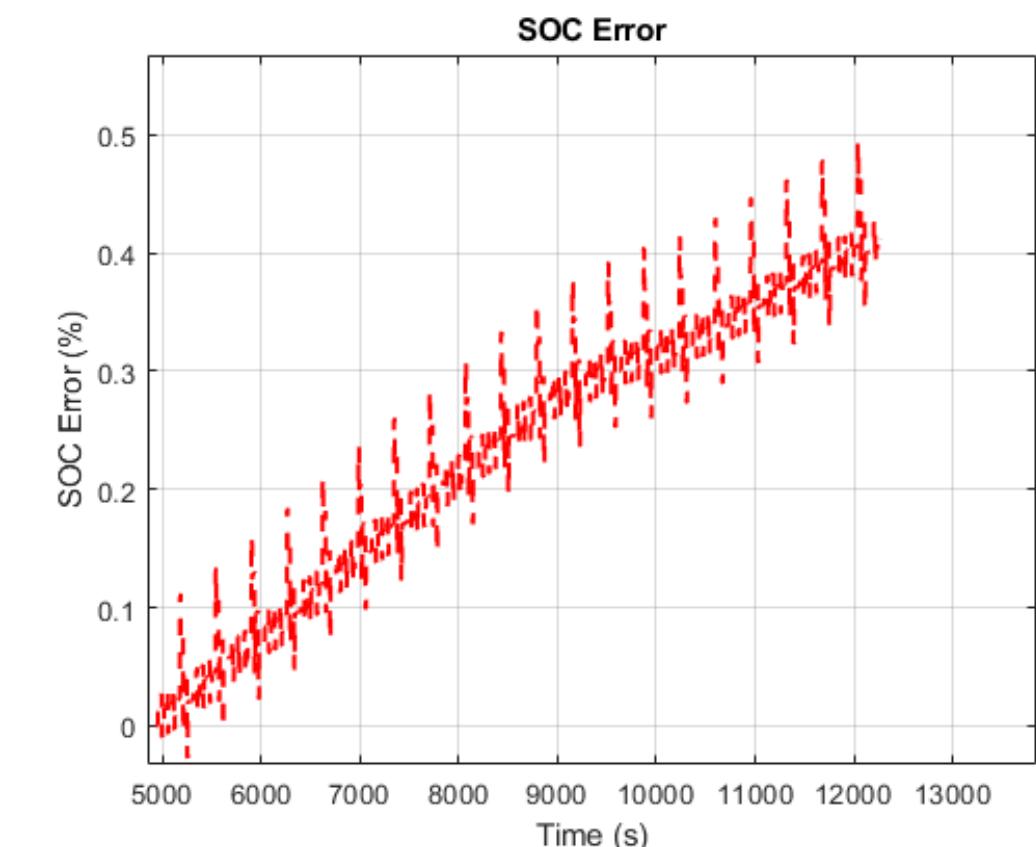
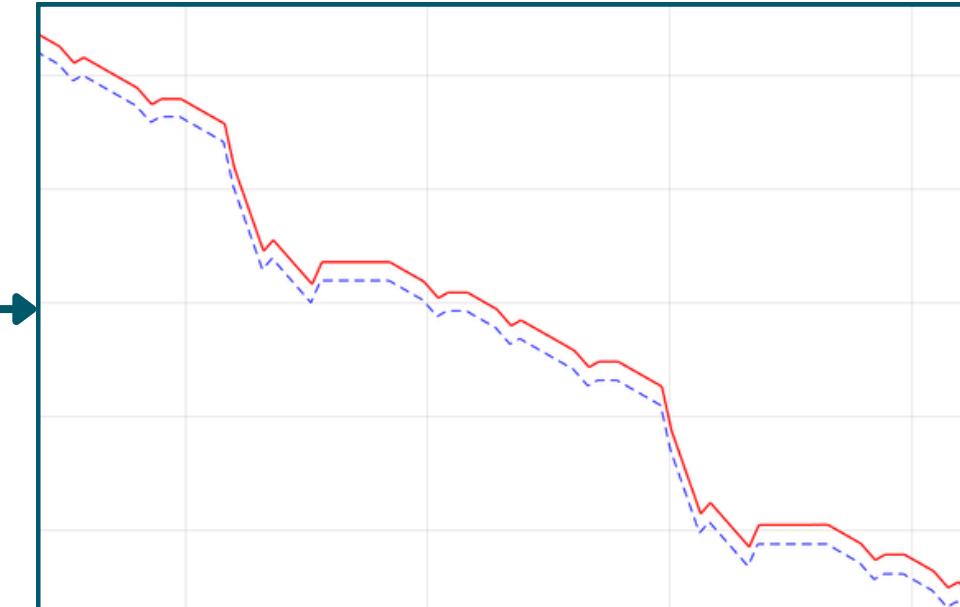
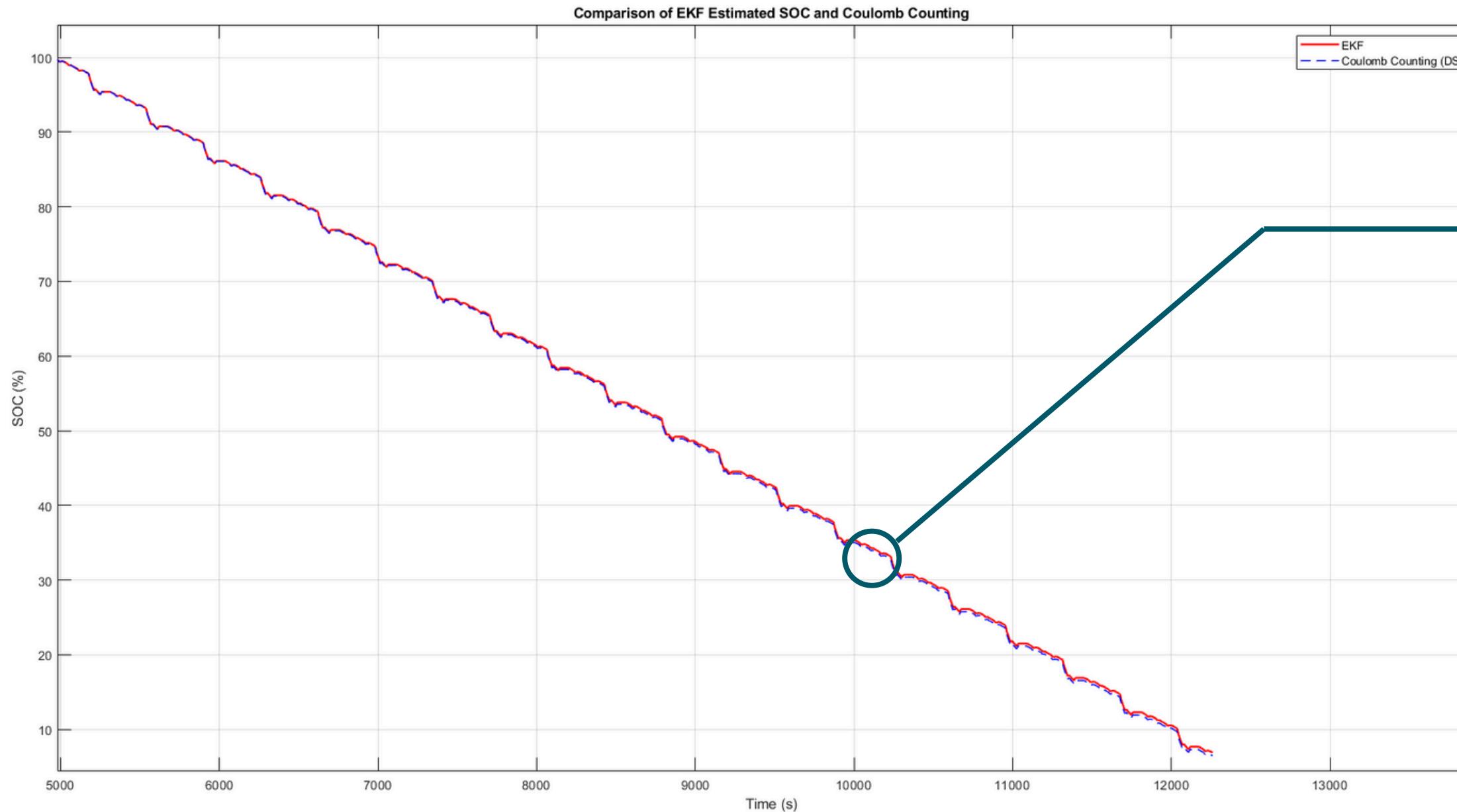
C1



C2



# Hasil Estimasi SoC Menggunakan EKF



Metode Extended Kalman Filter **berhasil** digunakan untuk mengestimasi SoC baterai berdasarkan nilai parameter RC yang dinamis. Hasil simulasi menunjukkan bahwa **nilai MAE** dari estimasi SoC menggunakan Extended Kalman Filter hanya sebesar **0.2384%**

# Terima Kasih

Lampiran program dan *resource* proyek:

<https://github.com/PinZapPin/BMS>

