Implementasi Struktur *Hammerstein* Untuk Pemodelan *State Of Charge* Baterai

1st Baharuddin Nur Maulana
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
maulanabn@student.telkomuniversity.ac.id

2nd Reza Fauzi Iskandar
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
rezafauzii@telkomuniversity.ac.id

3rd Indra Wahyudin Fathonah
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
indrafathonah@telkomuniversity.ac.id

***Abstrak* — Baterai adalah perangkat yang dapat menyimpan energi listrik untuk kebutuhan setiap saat dan dengan mudah memindahkannya dari satu tempat ke tempat lain. Dengan kemajuan teknologi saat ini, semakin banyak aktivitas yang menggunakan baterai. Namun, seringkali proses pengisian baterai dalam perangkat elektronika melebihi kapasitas sehingga akan mengakibatkan *overcharge* dan penggunaan daya yang terlalu lama akan mengakibatkan baterai cepat habis *(overdischarge)*. Dalam *Battery Management System*, akurasi pengukuran *State of Charge* (SoC) merupakan salah satu aspek terpenting. Estimasi SoC yang akurat tidak hanya dapat melindungi baterai, mencegah pengisian daya yang berlebihan dan memperpanjang masa pakai baterai, tetapi juga memungkinkan aplikasi mengembangkan strategi kontrol rasional untuk menghemat energi. Pada penelitian ini, kami berfokus pada implementasi struktur *hammerstein* untuk pemodelan *state of charge* baterai. Model *hammerstein* digunakan untuk mengatasi kompleksitas nonlinearitas dalam karakteristik SoC baterai sambil mempertahankan sifat linier dari interaksi masukan dan keluaran blok linier. Dari penelitian ini proses estimasi SoC menggunakan model *hammerstein-recursive least square* menunjukkan bahwa estimasi yang dihasilkan memperoleh estimasi yang sangat baik dengan grafik pengukuran dan estimasi yang dihasilkan sangat terlihat berimpit dan perhitungan metrik evaluasi kinerja yang sangat kecil berupa nilai MSE 0,0029, RMSE 0.05473 dan MAPE 0.08114.**

***Kata kunci* — Battery Management System, Hammerstein, State of Charge**

I. PENDAHULUAN

Baterai adalah perangkat yang dapat menyimpan energi listrik untuk kebutuhan setiap saat dan dengan mudah memindahkannya dari satu tempat ke tempat lain [1]. Dengan kemajuan teknologi saat ini, semakin banyak aktivitas yang menggunakan baterai. Misalnya, penggunaan laptop dan handphone oleh masyarakat. Kecenderungan penggunaan perangkat elektronik tanpa henti membuat sebagian pengguna memilih baterai. Namun, seringkali proses pengisian baterai dalam perangkat elektronika melebihi kapasitas sehingga akan mengakibatkan *overcharge* dan penggunaan daya yang terlalu lama akan mengakibatkan baterai cepat habis *(overdischarge)*. Selain itu, sangat penting untuk memantau kinerja baterai karena kebanyakan orang tidak tahu kualitas baterai yang digunakan.

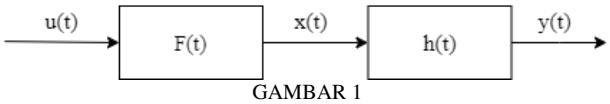
Battery Management System (BMS) merupakan sistem yang berfungsi untuk mengawasi kinerja baterai dan mencegah kegagalan baterai beroperasi diluar speseifikasinya, seperti mengalami *overcharge* atau *overdischarging*. Dalam BMS, akurasi pengukuran *State of Charge* (SoC) merupakan salah satu aspek terpenting. Estimasi SoC yang akurat tidak hanya dapat melindungi baterai, mencegah pengisian daya yang berlebihan dan memperpanjang masa pakai baterai, tetapi juga memungkinkan aplikasi mengembangkan strategi kontrol rasional untuk menghemat energi [2].

Menanggapi kebutuhan akan estimasi SoC yang handal, berbagai pendekatan telah diselidiki termasuk metode berbasis model. Dalam pengembangannya, pendekatan model *hammerstein* telah muncul sebagai alternatif metode yang menjanjikan. Model *hammerstein* menggabungkan dua elemen berupa blok nonlinier yang mewakili karakteristik fisik baterai dan blok linier yang menghubungkan keluaran blok nonlinier dengan masukan sistem. Kombinasi ini digunakan untuk mengatasi kompleksitas nonlinearitas dalam karakteristik SoC baterai sambil mempertahankan sifat linier dari interaksi masukan dan keluaran blok linier. Selain itu, metode berbasis model sering digunakan untuk estimasi SoC karena akurasi dan ketahanannya yang baik [3].

Pada penelitian ini, kami berfokus pada implementasi struktur hammerstein untuk pemodelan *state of charge* baterai. Dengan menerapkan model *hammerstein*, diharapkan hasil penelitian ini dapat meningkatkan pemahaman dan akurasi dalam estimasi SoC baterai, yang pada gilirannya akan mendukung pengimplementasian baterai yang lebih andal dan efisien dalam berbagai aplikasi di masa depan.

II. KAJIAN TEORI

A. Model *Hammerstein*



GAMBAR 1
Model *hammerstein*

Pada gambar 1, model *hammerstein* memiliki subsistem statis nonlinier F(t) yang diikuti oleh subsistem dinamis linier h(t) [4]. Jika suatu sistem bersifat linier dan tidak bergantung waktu, maka hubungan masukan-keluaran linier dari sistem tersebut dapat direpresentasikan dengan integral konvolusi [5] [6], seperti yang ditunjukkan pada persamaan (1),

$$y(t) = \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau \quad (1)$$

keterangan :
$y(t)$: hasil integral konvolusi antara $h(\tau)$ dan $x(t-\tau)$
$h(\tau)$: fungsi respons impuls
$x(t-\tau)$: sinyal *input* x pada waktu t dipengaruhi $h(\tau)$
τ : variabel integrasi dalam konvolusi
$d\tau$: integrasi variabel τ dalam rentang $-\infty$ hingga ∞

Kemudian sinyal *input* $x(t)$ dinyatakan dalam bentuk fungsi *step* $u(t)$ yang diolah melalui transformasi nonlinear $F\big(u(t)\big)$, sebagaimana ditunjukkan pada persamaan (2),

$$x(t) = F\left(u(t)\right) \quad (2)$$

dengan, $x(t)$ adalah sinyal *input* pada waktu t dan $F\left(u(t)\right)$ adala fungsi yang menerima *input* $\left(u(t)\right)$

Selanjutnya fungsi nonlinier F(t) diasumsikan sebagai fungsi nonlinier polinomial, yang dapat dimodelkan dengan

$$F\big(u(t)\big) = \sum_{n=1}^{N} a_n[u(t)]^n \quad (3)$$

dengan, $a_n(n = 1, \dots, N)$ adalah koefisien dari fungsi nonlinier polinomial dan N merupakan urutan pemotongan maksimal dari polinomial fungsi nonlinier.

B. *Estimator kernel*

Metode estimasi *kernel* adalah sebuah pendekatan yang digunakan dalam analisis statistik dan pembuatan model untuk mengestimasi fungsi densitas probabilitas dari data. Pendekatan ini memungkinkan untuk mengidentifikasi dan memahami distribusi probabilitas data yang dimiliki tanpa harus membuat asumsi tertentu tentang bentuk distribusinya. Fungsi estimator *kernel* untuk fungsi densitas [7], dapat dijelaskan sebagai berikut:

$$\hat{f}_h = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \quad (4)$$

Keterangan :
$\hat{f}_h$: perkiraan fungsi kernel dari distribusi data
n : jumlah total data
h : *bandwidth* atau lebar pita kernel
K : fungsi kernel
x : variabel *input* data
x_i : input dari sampel data ke-i

$\hat{f}_h(x)$tergantung pada fungsi *kernel* K dan parameter h, seperti yang ditunjukkan dalam persamaan (4). Fungsi *kernel* K menentukan bentuk bobot *kernel*, dan *bandwidth* adalah parameter pemulus h. Berikut jenis fungsi *kernel* yang digunakan,

TABEL 1
Jenis fungsi *kernel*

No	Jenis *Kernel*	Persamaan
1	*Kernel Cosinus*	$K(x) = \frac{\pi}{4}\cos\left(\frac{\pi}{4}x\right) I(\lvert x\rvert \le 1)$
2	*Kernel Epanechnikov*	$K(x) = \frac{3}{4}(1 - x^2) I(\lvert x\rvert \le 1)$
3	*Kernel Gaussian*	$K(x) = \frac{1}{\sqrt{2\pi}}\left(\frac{1}{2}(-x^2)\right) -\infty < x < \infty$
4	*Kernel Uniform*	$K(x) = \frac{1}{2} I(\lvert x\rvert \le 1)$

dengan, $K(x)$ adalah fungsi *kernel* K pada titik x dan I adalah fungsi indikator

C. *Recursive Least Square*

Algoritma *Recursive Least Square* digunakan dalam pemrosesan sinyal dan estimasi parameter dalam lingkungan non-stasioner. Algoritma ini digunakan untuk menemukan solusi perkiraan parameter yang optimal dalam model linier dengan mempertimbangkan data yang masuk secara berurutan. *Recursive Least Square* (RLS) standar dilakukan dalam Algoritma 1 [8].

Algorithm 1 : *Recursive Least Square*

Langkah awal dilakukan dengan mengatur vektor berat (w) menjadi nol (tidak ada pengaruh awal),

$$(w)(0) = 0 \quad (5)$$

dilanjutkan dengan menetapkan matriks kovarians awal dengan nilai yang telah ditentukan, untuk menggambarkan seberapa tidak pastinya estimasi awal,

$$\mathbf{P}(0) = \lambda^{-1}\mathbf{I} \quad (6)$$

dimana, λ adalah faktor pelupaan (parameter yang mengontrol sejauh mana estimasi sebelumnya diingat) dan **I** adalah matriks identitas

Kemudian dilakukan pengulangan *(loop)* yang dilakukan oleh algoritma RLS setelah iterasi awal (i=0). Artinya, setelah diinisialisasi pada iterasi awal, algoritma RLS akan terus berjalan untuk setiap iterasi selanjutnya dengan nilai i mulai dari 1 sampai tak terbatas, seperti yang ditunjukkan pada persamaan berikut,

Iterasi untuk $i \ge 1$ (7)

Selanjutnya, dilakukan perhitungan residual *error* untuk menghitung seberapa besar ketidakpastian dalam prediksi dan mempertimbangkan data *input* dan matriks kovarian sebelumnya,

$$r_e(i) = 1 + \boldsymbol{u}(i)^T\mathbf{P}(i-1)\boldsymbol{u}(i) \quad (8)$$

dengan,
$r_e(i)$: faktor skala dalam menghitung *gain vector* (ketidakpastian dalam estimasi)
$\boldsymbol{u}(i)$: vektor *input* pada iterasi ke-i.
$\mathbf{P}(i-1)$: matriks kovarians pada iterasi sebelumnya.
Untuk mendapatkan *gain vektor* $(\boldsymbol{k}_p)$ dilakukan perhitungan sebagai berikut,

$$\boldsymbol{k}_p(i) = \mathbf{P}\,(i-1)\boldsymbol{u}(i)/r_e(i) \quad (9)$$

Kemudian dilakukan perhitungan *error* untuk menghitung seberapa besar *"error"* prediksi pada titik ini,

$$e(i) = d(i) - \boldsymbol{u}(i)^T w(i-1) \quad (10)$$

dimana, $e(i)$: Error residual pada iterasi ke-I dan $d(i)$: Nilai target pada iterasi ke-i.

Setelah melalui proses perhitungan *error*, dilanjutkan dengan memperbarui vektor berat (w), untuk menyesuaikan parameter model agar lebih cocok dengan data terbaru,

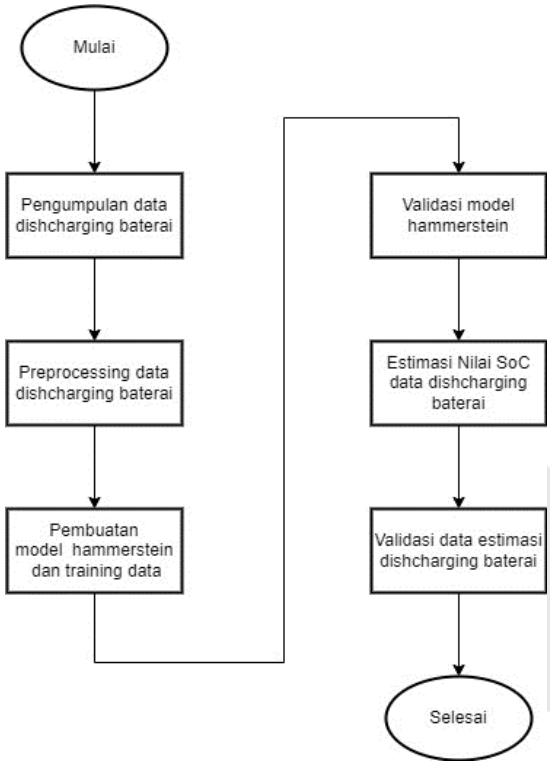
$$\boldsymbol{w}(i) = \boldsymbol{w}(i-1) + \boldsymbol{k}_p(i)e(i) \quad (11)$$

Selanjutnya untuk memperhitungkan perubahan yang terjadi dan memperbarui sejauh mana estimast parameter ini dilakukan secara akurat, maka dilakukan perhitungan pembaharuan nilai matriks kovarian, seperti berikut

$$\mathbf{P}(i) = [\mathbf{P}(i-1) - \mathbf{P}(i-1)\boldsymbol{u}(i)\boldsymbol{u}(i)^T\mathbf{P}(i-1)/r_e(i)] \quad (12)$$

Kemudian tahapan-tahapan perhitungan tersebut dilakukan terus menerus pada setiap data yang masuk.

III. METODE



GAMBAR 3
Rancangan program

Adapun proses pengolahan data yang dilakukan :

Pada proses pengolahan data ini menggunakan *software Google Colaboratory* dan bahasa pemrograman *python*. Tahap awal dimulai dengan melakukan pengumpulan data *discharging state of charge* yang didapatkan dari proses pengukuran set up lithium NMC dengan kapasitas 3000 mAh dan tegangan operasi 3,7 VDC yang dillakukan oleh peneliti terdahulu. Dimana pengukuran tersebut menghasilkan data berupa siklus, tegangan, arus, kapasitas dan perhitungan nilai SoC. Selanjutnya data yang terkumpul kemudian dilakukan *preprocessing* dan visualisasi data untuk menghapus data yang tidak valid, melengkapi data yang hilang dan memfilter data yang tidak perlu serta digunakan untuk memahami struktur data sebelum dilakukan pemrosesan dan analisis lebih lanjut.

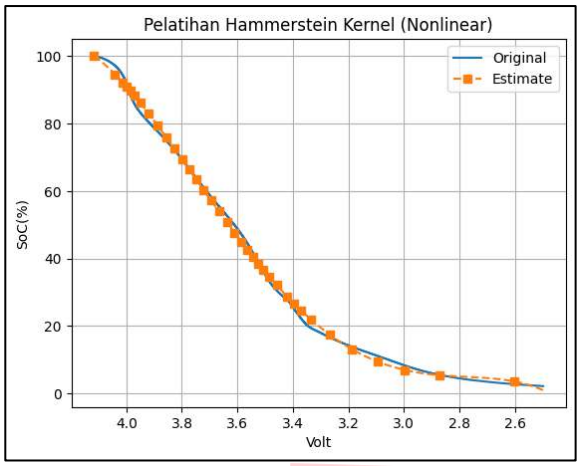
Data berupa nilai tegangan dan hasil perhitungan SoC kemudian diolah dengan menggunakan pemodelan *hammerstein* yang menggabungkan model blok non-linear dan blok linear. Selanjutnya, dilakukan proses identifikasi parameter dari model *hammerstein*. Pada proses ini, teknik identifikasi parameter yang digunakan adalah teknik algoritma *kernel* sebagai blok nonlinier dan *recursive least squares* sebagai blok linier. Setelah proses pembuatan model, dilakukan proses *training* data untuk melatih atau mengajarkan model *hammerstein-kernel* agar dapat mempelajari pola dan karakteristik data yang ada dan menghasilkan prediksi atau *output* yang akurat ketika diuji dengan data yang belum pernah dilihat sebelumnya. Selanjutnya, dilakukan validasi model untuk mengukur kemampuan model dan membuat prediksi akurat yang sebelumnya tidak diketahui.Validasi model ini bertujuan untuk memastikan bahwa model dapat memberikan estimasi SoC yang akurat dan andal saat diterapkan pada serangkain siklus yang terdapat dalam dataset (dalam hal ini proses *training* dan validasi data dilakukan pada blok nonlinier terlebih dahulu).

Langkah selanjutnya kemudian dilakukan proses *training* dan validasi data untuk melatih atau mengajarkan model *hammerstein-recursive least square* agar dapat mempelajari pola dan karakteristik data yang ada dan menghasilkan prediksi atau *output* yang akurat(dalam hal ini proses *training* dan validasi data dilakukan pada blok linier). Kemudian untuk memastikan hasil estimasi SoC akurat dilakukan juga perhitungan metrik evaluasi kinerja, seperti nilai MSE *(Mean Squared Error)*, RMSE *(Root Mean Square Error)* dan MAPE *(Mean Absolute Percentage Error)* untuk mengukur kualitas dan kinerja dari model estimasi yang telah dibuat.

IV. HASIL DAN PEMBAHASAN

A. Pembahasan hasil model *hammerstein* blok non-linier

Pada pengujian model *hammerstein* blok non-linear ini, data yang digunakan merupakan data *discharging* baterai. Data *discharging* ini menghasilkan data berupa jumlah siklus, nilai arus, tegangan, kapasitas dan nilai hasil perhitungan SoC. Setelah itu, data berupa tegangan dan nilai SoC tersebut dilakukan proses identifikasi menggunakan model *hammerstein* dan *kernel* agar diperoleh modelnya. Selanjutnya, model yang dihasilkan oleh proses identifikasi tersebut dilakukan pengujian untuk melihat performanya. Berdasarkan Gambar 4 proses *training hammerstein-kernel* menunjukkan grafik pengukuran dan estimasi yang dihasilkan terlihat berimpit, yang menunjukkan bahwa plot model mampu mempelajari pola dan karakteristik data yang ada serta menghasilkan prediksi atau output yang akurat ketika diuji dengan data yang belum pernah dilihat sebelumnya.

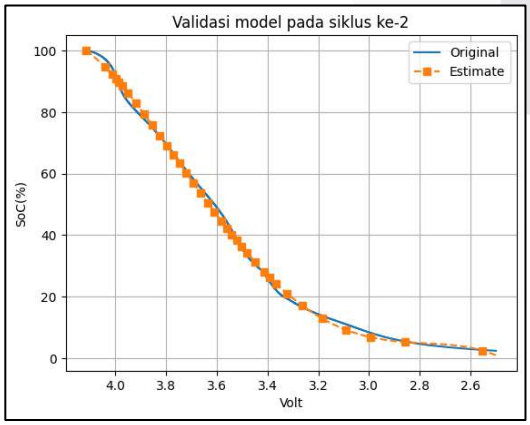


GAMBAR 4
Proses *training hammerstein-kernel*

Selanjutnya dari proses *training* pada pelatihan *hammerstein kernel* (non-linear) diperoleh model estimasi *system* dalam bentuk polinomial (13)

$$P_{discharging}(u) = -59.7092u^4 + 779.9234u^3 - 3733.8982u^2 + 7813.2983u - 6049.3245 \quad (13)$$

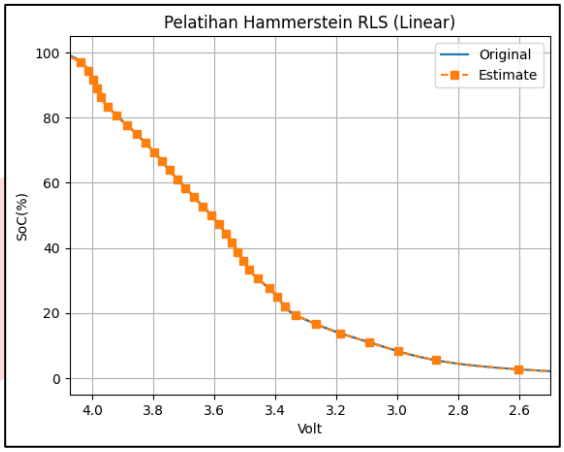
Nilai polinomial yang didapatkan dari persamaan (13) tersebut merupakan nilai yang digunakan untuk menggambarkan hubungan non-linier antara masukan dan keluaran sistem. Selain itu, dengan menggunakan nilai polinomal yang diperoleh, kita dapat memperkirakan keluaran *(output)* yang dihasilkan berdasarkan nilai inputannya. Dimana grafik yang dihasilkan akan menunjukkan bagaimana nilai polinomial ini menggambarkan hubungan non-linier. Hal tersebut, memungkinkan kita dapat melakukan proses estimasi terhadap respons sistem yang dihasilkan. Selanjutnya, dilakukan proses validasi dan perhitungan metrik evaluasi kinerja, berupa hasil perhitungan MSE, RMSE, dan MAPE pada model *hammerstein-kernel* yang diujikan. Dalam hal ini akan diujikan pada siklus ke-2. Berdasarkan Gambar 5 proses validasi *hammerstein-kernel* menunjukkan grafik pengukuran dan estimasi yang dihasilkan terlihat berimpit dan perhitungan metrik evaluasi kinerja yang relatif kecil berupa nilai MSE 2.3816, RMSE 1.5432 dan MAPE 4.1299.



GAMBAR 5
Proses validasi model *hammerstein-kernel*

B. Pembahasan hasil model *hammerstein* blok linier

Setelah dilakukan proses penerapan model *kernel* pada blok non linier, kemudian dilakukan penerapan model *Recursive Least Square* (RLS) pada blok linier. Berdasarkan Gambar 6 proses *training hammerstein-recursive least square* menunjukkan grafik pengukuran dan estimasi yang dihasilkan terlihat sangat berimpit, yang menunjukkan bahwa plot model *hammerstein* mampu memprediksi data *state of charge* (SoC) dengan baik selama proses *training*.

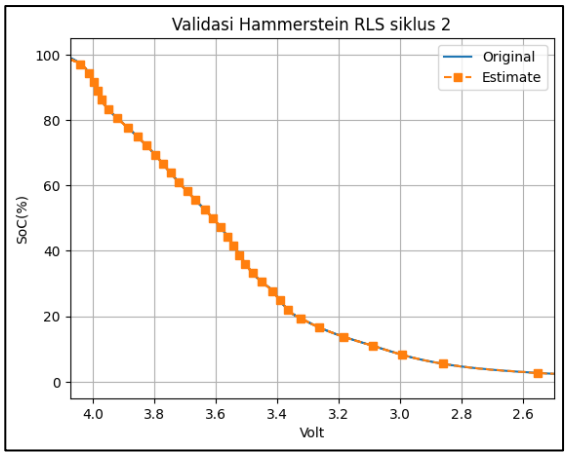


GAMBAR 6
Proses *training hammerstein-recursive least square*

Pada penerapan model *Recursive Least Square* (RLS) sebagai blok linier memberikan hasil identifikasi model linier yang ditunjukkan pada persamaan fungsi transfer (14).

$$G(z) = \frac{-0{,}3834z^{-1} + 0{,}4037z^{-2}}{1 - -0{,}4403z^{-1} - 0{,}5284z^{-2}} \quad (14)$$

Fungsi transfer yang didapatkan dari persamaan (14) tersebut, menggambarkan respons sistem blok linier terhadap inputan blok non-linier yang telah diproses sebelumnya oleh blok non-linier (dalam hal ini, nilai polinomial yang diperoleh). Fungsi transfer yang diperoleh merupakan nilai perbandingan antara transformasi z dari nilai keluaran dan masukan sistem. Selanjutnya, dilakukan proses validasi dan perhitungan metrik evaluasi kinerja, berupa hasil perhitungan MSE, RMSE, dan MAPE pada model *hammerstein-recursive least square* yang diujikan. Dalam hal ini akan diujikan pada siklus ke-2. Berdasarkan Gambar 7 proses validasi hammerstein-recursive least square menunjukkan grafik pengukuran dan estimasi yang dihasilkan terlihat sangat berimpit dan perhitungan metrik evaluasi kinerja yang sangat kecil berupa nilai MSE 0,0029, RMSE 0.05473 dan MAPE 0.08114.



GAMBAR 7
Proses validasi *hammerstein-recursive least square*

V. KESIMPULAN

Dengan menggabungkan nilai polinomial dan fungsi transfer dalam model *hammerstein*, kita dapat memodelkan dan memahami sepenuhnya respons sistem baterai selama proses pengosongan. Polinomial digunakan untuk menggambarkan hubungan nonlinier antara *input* dan *output*, sedangkan fungsi transfer digunakan untuk menganalisis dan memprediksi respons sistem linier terhadap *input* nonlinier yang diproses oleh blok nonlinier. Dengan demikian, keduanya bekerja sama untuk memberikan gambaran sistem baterai yang lebih lengkap dalam konteks pemodelan *hammerstein*. Hasil yang diperoleh dari proses model *hammerstein-kernel* menunjukkan bahwa estimasi yang dihasilkan memperoleh estimasi yang baik. Hal ini dibuktikan dengan grafik pengukuran dan estimasi yang dihasilkan terlihat berimpit dan perhitungan metrik evaluasi kinerja pada siklus ke-2 yang relatif kecil berupa nilai MSE 2.3816, RMSE 1.5432 dan MAPE 4.1299. Hasil yang diperoleh dari proses estimasi SoC *hammerstein-recursive least square* menunjukkan bahwa estimasi yang dihasilkan memperoleh estimasi yang sangat baik dengan akurasi yang tinggi. Hal ini dibuktikan dengan grafik pengukuran dan estimasi yang dihasilkan sangat terlihat berimpit dan perhitungan metrik evaluasi kinerja yang sangat kecil berupa nilai MSE 0,0029, RMSE 0.05473 dan MAPE 0.08114 pada data siklus ke-2.

REFERENSI

[1] A. D. Isnaini, Suwandi and R. F. Iskandar, "Estimation State Of Charge Of Lithium Ion Battery Using Coulomb Counting Method," *Estimation State Of Charge Of Lithium Ion Battery Using Coulomb Counting Method,* p. 650, 2017.

[2] W. Y. Chang, "The State of Charge Estimating Methods for Battery: A Review," *Hindawi Publishing Corporation,* pp. 1-7, 2013.

[3] J. Meng, G. Luo, M. Ricco, M. Swierczynski, D.-I. Stroe and R. Teodorescu, "Overview of Lithium-Ion Battery Modeling Methods for State-of-Charge Estimation in Electrical Vehicles," *Applied Sciences,* p. 659, 2018.

[4] C. M. Cheng, Z. K. Peng, W. M. Zhang and G. Meng, "Volterra Series Based Nonlinear System Modeling and its Engineering Applications: A State of The Art Review," *Mechanical Systems and Signal Processing,* pp. 340-364, 2017.

[5] E. J. Dempsey and D. T. Westwick, "Identification of Hammerstein Models With Cubic Spline Nonlinearities," *IEEE Trans. Biomed. Eng,* pp. 237-245, 2004.

[6] E. W. Bai and M. Fu, "A Blind Approach to Hammerstein Model Identification," *IEEE Trans. Signal Process,* p. 1610–1619, 2002.

[7] M. P. Wand and M. C. Jones, "Kernel Smoothing," *Chapman and Hall,* 1995.

[8] S. Haykin, Adaptive Filter Theory, Prentice-Hall, 2002.

.