Jurnal Lontar Physics Today
Vol. 2 No. 3 November 2023, p 127 – 135
e-ISSN 2828-0970
Available Online at http://journal.upgris.ac.id/index.php/UPT
DOI: 10.26877/lpt.v2i3.18128

Pendekatan Neural Network pada Gerak *Unstretch Bungee Jumping* Menggunakan Metode Euler

Tika Wahyuni*, Fardika Armawanto, Diah Ayu Faradita, Nur Khoiri, Affandi Faisal Kurniawan, Joko Saefan
Program Studi Pendidikan Fisika, Universitas PGRI Semarang, Jl. Lontar No. 1 Semarang
*E-mail: tikaw061@gmail.com

Abstrak. Persamaan gerak *unstretch bungee jumping* dapat diselesaikan secara numerik menggunakan metode Euler. Hasil penyelesaian secara numerik menggunakan Euler dapat dikatakan valid apabila metode Odeint mendapatkan hasil yang hampir sama dengan metode Euler. Setelah hasil dari metode numerik tervalidasi, sehingga dapat implementasikan pada metode *neural network*. *Neural network* dapat divariasi menggunakan berbagai epochs untuk memprediksi hasil penyelesaian secara numerik. Dengan epochs 500 didapatkan hasil yang lebih akurat dan mendekati dari hasil odeint.

Kata kunci: Bungee Jumping*, Solusi Numerik, dan* Neural Network

Abstract. The unstretch bungee jumping equation of motion can be solved numerically using the Euler method. The results of numerical solving using Euler can be said to be valid if the Odeint method gets results that are almost the same as the Euler method. After the results of the numerical method are validated, it can be implemented in the neural network method. The neural network can be varied by using various epochs to predict the numerical solution results. With 500 epochs, more accurate and closer results to the odeint outcomes were obtained.

Keywords: Bungee Jumping, numerical solution, Neural Network

1. Pendahuluan

Bungee jumping atau terjun lenting merupakan olahraga dimana seseorang melompat dari ketinggian dengan pergelangan kaki yang diikat dengan tali dan kemudian jatuh secara vertikal ke bawah [1]. *Bungee Jumping* pada bidang fisika merupakan sebuah fenomena perubahan massa seperti gerakan rantai jatuh, perilaku batang atau pegas elastis yang jatuh, dan gerakan *bungee jumping* [2]. Pada banyaknya penelitian, gerak dianggap satu dimensi, tali dimodelkan sebagai tali elastis tak bermassa, pelompat diganti dengan massa titik, efek aerodinamis diabaikan, dan kurva tegangan-regangan tali diasumsikan linier yaitu, hukum Hooke berlaku [3].

Pada *bungee jumping* terdapat fase jatuh bebas dari pelompat saat tali masih kendur, pada fase tersebut terdapat fakta bahwa pelompat mengalami percepatan lebih besar daripada percepatan jatuh bebas akibat gravitasi [4]. Hal tersebut dapat dipahami melalui pendekatan Newton dan hal ini dibuktikan pada hukum kekekalan energi dengan menurunkan persamaan geraknya melalui persamaan Lagrange [5]. Dimana persamaan geraknya dapat dinyatakan dalam persamaan diferensial. Persamaan gerak pada *bungee jumping* menggambarkan gerak benda saat jatuh vertikal dengan memperhitungkan gaya gravitasi yang bekerja pada benda dan rantai pada *bungee jumping*.

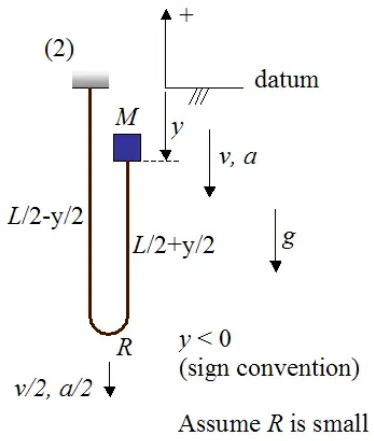
Persamaan gerak pada *bungee jumping* dapat diselesaikan menggunakan analisis numerik seperti metode Euler [6]. Metode Euler merupakan metode numerik yang digunakan untuk menyelesaikan

persamaan diferensial biasa (PDB) [7]. Untuk menghitung kecepatan benda yang jatuh bebas secara vertikal dapat menggunakan metode Euler. Kecepatan benda dapat didekati dengan menggunakan kemiringan garis singgung pada titik waktu tertentu. Sehingga metode Euler ini cocok untuk memperkirakan kecepatan benda saat jatuh bebas pada titik waktu yang berbeda selama benda jatuh.

Setelah dilakukan beberapa percobaan dan penyelesaian secara numerik, persamaan gerak pada *bungee jumping* diprediksi menggunakan *neural network. Neural network* merupakan kecerdasan buatan yang mengajarkan komputer untuk memproses data dengan cara serupa yang dilakukan oleh otak manusia [8]. Pada *neural network* menggunakan sistem jaringan dari unit pemroses kecil yang saling terhubung atau disebut neuron dalam struktur berlapis untuk menyelesaikan masalah-masalah kompleks. Tujuan dari penelitian ini adalah mengembangkan metode baru dalam menganalisis persamaan gerak *bungee jumping* menggunakan *neural network*.

1.1 *Bungee Jumping*

Skema analisis sederhana dari representasi *bungee jumping* menunjukkan bahwa percepatan a lebih besar daripada percepatan gravitasi g saat benda jatuh yang terikat pada rantai tanpa gesekan pada posisi awal. Benda direpresentasikan sebagai pelompat dengan massa M yang diikatkan pada rantai dengan massa m dan panjang L. Terdapat sebuah rantai yang berbentuk U, dimana masing-masing sisi rantai memiliki panjang $L/2$, dan terdapat kelengkungan di bagian bawah dengan jari-jari R. Dimana pada satu sisi rantai tergantung pada penyangga dan sisi lain jatuh bebas ketika benda jatuh secara vertikal ke bawah, bagian rantai yang jatuh tetap berada di bawahnya selama bagian awal jatuhnya ketika rantai kendur. Percepatan gravitasi adalah g yang dapat diilustrasikan pada Gambar 1.



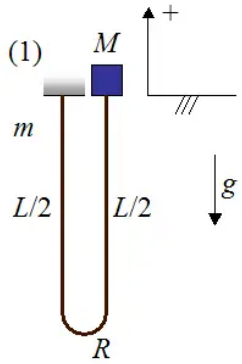
Gambar 1. Posisi awal benda jatuh

Skema analisis sederhana dari representasi *bungee jumping* bahwa $a > g$ pada jatuhnya benda yang diikatkan pada sebuah rantai tanpa gesekan setelah benda jatuh. Posisi *jumper* dan tali diatur sebagai fungsi dari y. Panjang rantai yang lurus dinyatakan sebagai fungsi dari y, kecepatan vertikal benda yang jatuh dinyatakan sebagai v, dan percepatan vertikal dinyatakan sebagai a. Secara geometri kecepatan vertikal ujung lengkungan adalah $v/2$, dan percepatan vertikal ujung bawah lengkungan adalah $a/2$ yang diilustrasikan pada Gambar 2.

Persamaan gerak pada *bungee jumping* ini berupa persamaan diferensial orde dua yang dirumuskan menurut [5]

$$\frac{d^2y}{dt^2} = g + \frac{\frac{1}{2}\mu\dot{y}^2}{\mu(L-y)+2L} \tag{1}$$

dengan μ merupakan m/M.



Gambar 2. Posisi benda setelah jatuh

1.2 *Metode Numerik*

Metode numerik merupakan metode yang digunakan untuk memecahkan suatu permasalahan matematis dengan persamaan diferensial yang tidak dapat diselesaikan dengan metode analitik. Metode numerik sering digunakan untuk menyelesaikan persamaan diferensial kontinyu menjadi diferensial diskret.

Metode Euler merupakan metode numerik yang paling sederhana dan umum digunakan untuk menyelesaikan persamaan diferensial biasa (PDB) orde satu dengan nilai awal yang diberikan. Akan tetapi metode Euler ini mempunyai ketelitian yang kurang akurat. Peneliti di bidang matematika terapan menemukan bahwa metode Euler tidak efisien dan rentan akan kesalahan [9]. Metode Euler lebih akurat untuk pendekatan solusi PDB ketika langkah waktu yang digunakan relative kecil [10]. Rumusan umum untuk metode Euler sebagai berikut:

$$y_{x+h} = y_x + hy'_x \tag{2}$$

1.3 *Neural Network*

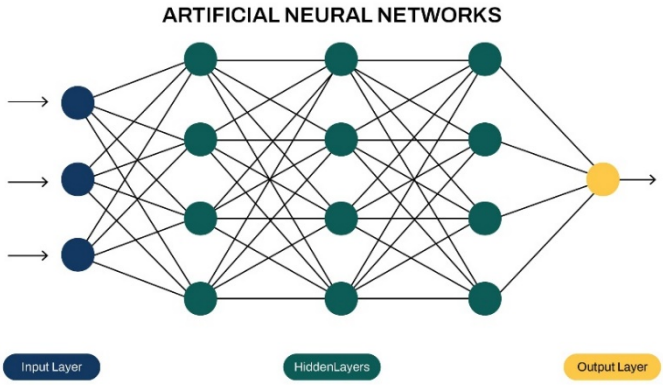
Neural Network merupakan sebuah algoritma komputasi yang dimodelkan seperti struktur dan fungsi otak manusia yang dirancang untuk mengenali pola dan mengkalsifikasikan data berdasarkan data yang masuk. *Neural network* digunakan untuk mendapatkan solusi numerik dari permasalahan persamaan diferensial dan masalah nilai awal di berbagai bidang matematika, fisika, dan ilmu keknik.

Neural Network terdiri dari unit pemrosesan yang saling behubungan yang diatur dalam lapisan-lapisan yang disebut *node* atau neuron. Setiap *node* atau neuron mempresentasikan fungsi *output* tertentu yang disebut fungsi aktivasi [11]. *Node* (neuron) berfungsi untuk memecahkan masalah dari sebuah data. Node mewakili bobot untuk sinyal yang melewati koneksi atau disebut dengan bobot, yang setara dengan memori pada *neural network* [12]. Nilai bobot pada *node* atau neuron akan menentukan suatu *output* jaringan itu sendiri hingga neuron berikutnya. *Output* jaringan akan bervariasi tergantung pada bagaimana jaringan terhubung, nilai bobot, dan fungsi insentif [11].

Pada *neural network*, bobot awal akan bersifat acak selama pelatihan model, tetapi bobot ini diperbarui secara berulang untuk belajar memprediksi output yang benar. Dengan menguraikan jaringan, kita dapat mendefinisikan beberapa blok bangunan logis seperti neuron, lapisan, bobot, *input*, *output*, fungsi aktivasi di dalam neuron untuk menghitung proses pembelajaran, dan sebagainya.

Alam *neural network* unit pemrosesan neuron merepresentasikan objek yang berbeda-beda. Jenis unit pemrosesan dalam *neural network* dibagi menjadi tiga kategori. Lapisan unit pemrosesan pada *neural network* dapat diilustrasikan pada Gambar 3.

Dari Gambar 3 dapat dijelaskan bahwa *neural network* terdiri dari tiga jenis lapisan unit pemrosesan yaitu *input layer, hidden layer* dan *output layer*[13]. *Input layer* merupakan lapisan yang berkomunikasi dengan lingkungan eksternal dengan menerima sinyal dan data dari luar yang menyajikan pola ke *neural network* [14]. *Input layer* memiliki pola yang kemudian akan diolah oleh neuron-neuron atau *node* yang berada pada *hidden layer* yang kemudian memberikan *output* ke lapisan berikutnya dan seterusnya, sehingga menghasilkan *output layer* dengan pola lain [13].



Gambar 3. Arsitektur *Neural Network*

Hidden layer merupakan lapisan unit pemrosesan yang terletak diantara *input layer* dan *output layer* dan tidak dapat diamati di luar sistem [11]. *Hidden layer* memiliki fungsi untuk menerapkan bobot pada *input layer* dan mengarahkannya melalui fungsi aktivasi sebagai *output*. Setiap lapisan pada *hidden layer* memiliki satu atau banyak neuron, dan masing-masing neuron akan menghitung sebuah fungsi kecil (misalnya, fungsi aktivasi) [13]. Setiap neuron yang tersimpan pada *hidden layer* harus dihitung. Jika data dipisahkan secara linier, tidak diperlukan penggunaan *hidden layer* untuk fungsi aktivasi. Karena *hidden layer* melakukan transformasi secara non-linier dari *input* yang dimasukkan ke dalam jaringan. Unit pemrosesan pada *hidden layer* bervariasi bergantung pada fungsi *neural network*, yakni bergantung pada bobot yang terkait. Nilai-nilai *node* atau neuron pada *hidden layer* dihitung dengan menggunakan penjumlahan total dari nilai-nilai node *input* dikalikan dengan bobot yang ditetapkan. *Hidden layer* merupakan lapisan unit pemrosesan yang penting, karena pada *hidden layer* jaringan saraf dapat menangkap hubungan yang sangat kompleks antara *input layer* dan *output layer* dan mencapai kinerja yang menarik. *Hidden layer* mengubah fitur *input* menjadi fitur yang diproses dan kemudian diklasifikasikan dengan benar pada *output layer*.

Output layer merupakan lapisan unit pemrosesan terakhir dalam *neural network* yang menyajikan pola dari *hidden layer* ke lingkungan eksternal yang menghasilkan prediksi akhir. Jumlah neuron yang keluar harus terkait secara langsung dengan pekerjaan yang dilakukann oleh *neural network*. Dalam menentukan jumlah neuron yang akan digunakan pada *output* layer maka hal yang harus dilakukan yaitu mempertimbangkan tujuan penggunaan *neural network* tiruan. Apabila akan mengklasifikasikan item ke dalam kelompok menggunakan *neural network*, maka setiap kelompok yang ditugaskan ke dalam *input* memiliki satu neuron keluaran. Jika *neural network* melakukan pengurangan *noise* pada sinyal, maka jumlah neuron akan sama dengan jumlah neuron *output.* Dalam *neural network* pola-pola pada *output layer* memiliki pola yang sama dengan *input*.

Selain terdapat unit-unit pemrosesan, pada *neural network* juga terdapat komponen penting lainnya yang digunakan untuk mengevaluasi kinerja model dan memandu proses optimasi yaitu *loss function. Loss function* digunakan untuk menghitung kuantitas yang ingin diminimalkan oleh model. *Loss function* pada *neural network* yang umum digunakan yaitu *Mean Squared Error* (MSE), *cross-entropy loss, log loss,* dan *hinge loss*. Pemakaian *loss function* berdasarkan pada jenis tugas dan model yang digunakan. Pemilihan *loss function* bergantung dengan aktivasi yang digunakan *output layer* pada *neural network*.

Dalam *neural network, loss function* yang umum digunakan adalah *Mean Squared Error* (MSE). *Mean Squared Error* (MSE) digunakan untuk mengukur rata-rata perbedaan kuadrat antara nilai aktual dan nilai prediksi. Pada *neural network* MSE digunakan sebagai fungsi untuk mengukur kinerja jaringan menurut rata-rata kesalahan kuadrat dengan rumusan sebagai berikut

$$MSE = MSE_f + MSE_u \tag{3}$$

dengan

$$MSE_f = \frac{1}{N_f}\sum_{i=1}^{N_f}\left(\frac{dN(x)}{dx}|_{x=x_i} - f(x_i)\right)^2 \tag{4}$$

dan

$$MSE_u = \frac{1}{N_f}\sum_{i=1}^{N_u}\left(\frac{dN(x)}{dx}|_{x=x_i} - y(x_i)\right)^2 . \tag{5}$$

Pada persamaan (4) dan (5), MSE_f merupakan persamaan diferensial sedangkan MSE_u berhubungan dengan data awal dan batas.

2. Metode

Persamaan diferensial pada *bungee jumping* biasa didapatkan dari penurunan persamaan gerakan melalui persamaan Lagrange dan dapat dibuktikan dengan hukum kekekalan energi pada rumuasan berikut

$$L = T - V \tag{6}$$

dimana energi kinetik (T) dan energi potensial (V). Untuk persamaan gerak pada *bungee jumping* dapat dilihat pada persamaan (1). Dari persamaan (1) dapat diselesaikan menggunakan analisis numerik berupa metode Euler. Setelah hasil numerik dengan metode Euler tervalidasi kemudian diimplikasikan menggunakan pada Odeint dan *neural network* menggunakan *Phyton*. Dengan menggunakan berbagai metode tersebut didapatkan solusi yang akurat dari persamaan gerak pada *bungee jumping*.

3. Hasil Penelitian

Dari penyelesaian persamaan diferensial *bungee jumping* diperoleh jawaban secara numerik dengan parameter pada Tabel 1.

Tabel 1. *Parameter*

No.	Parameter	Dimensi	Satuan
1.	g	Percepatan gravitasi	9,8 ms^{-2}
2.	L	Panjang Tali	$1m$
3.	m	Massa Tali	$0{,}05kg$
4.	M	Massa Benda	$1\ kg$
5.	μ	Koefisien gesek	$0{,}1\ kgm/s$
8.	h	Lebar grid	0.01

Solusi persamaan diferensial secara numerik didapatkan dari parameter pada Tabel 1. Penyelesaian secara numerik pada kasus *bungee jumping* didapatkan persamaan gerak berupa

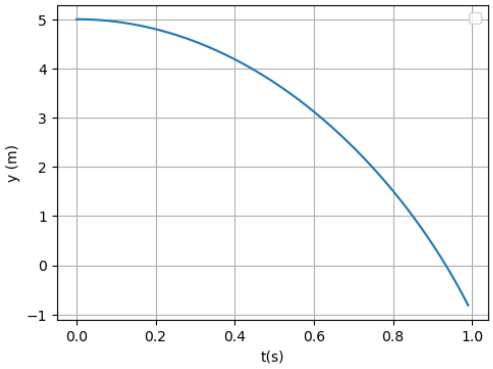
$$\frac{d^2y}{dt^2} = g + \frac{\frac{1}{2}\mu\dot{y}^2}{\mu(L-y)+2L} \tag{7}$$

dari persamaan gerak dapat diselesaiakan menggunakan metode Euler dengan persamaan yang di dapat

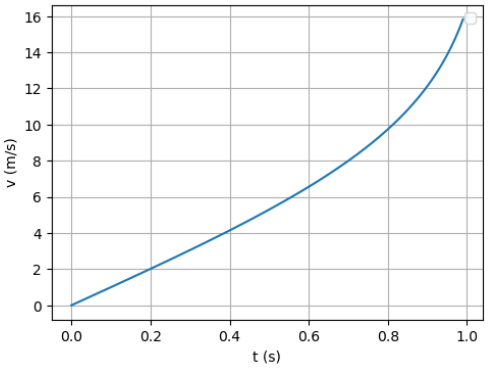
$$y_{t+h} = y_t + h\dot{y} \tag{8}$$

$$\dot{y}_{t+h} = \dot{y}_t + h\ddot{y}. \tag{9}$$

Selain menggunakan metode Euler, *bungee jumping* juga dapat diselesaikan dengan menggunakan odeint. Odeint merupakan fungsi yang digunakan untuk menyelesaikan sistem persamaan diferensial biasa dimana odeint menyimulasikan gerakan benda yang jatuh dari waktu ke waktu.



(a) y terhadap t

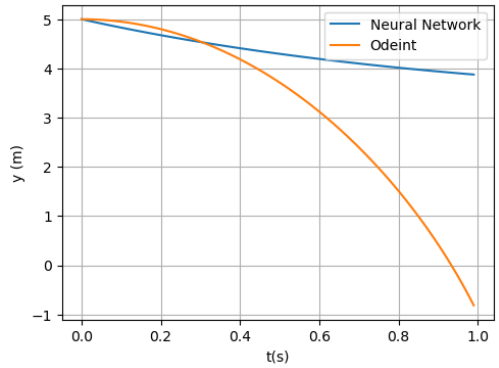


(b) v terhadap t

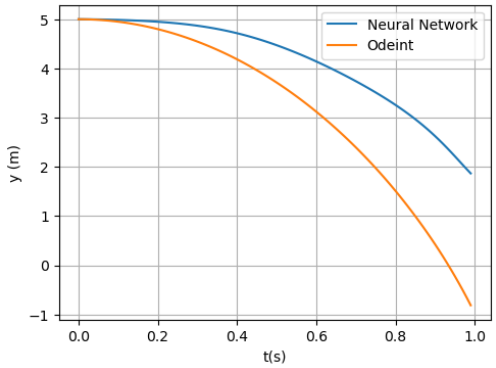
Gambar 4. Grafik Odeint

Setelah menyelesaikan persamaan gerak pada *bungee jumping* dengan solusi numerik dan Odeint, selanjutnya persamaan gerak dapat diprediksi menggunakan *neural network* untuk meminimalisir kesalahan. Hasil dari Odeint yang sudah tervalidasi dijadikan acuan terhadap metode *neural network*.

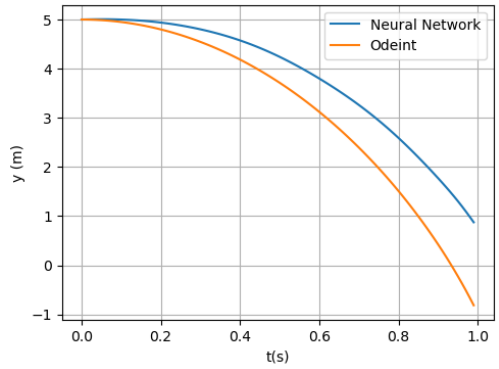
Untuk membuktikan bahwa *neural network* mempunyai nilai yang akurat dan mendekati nilai Odeint maka dapat ditinjau dengan melakukan perubahan epochs algoritma numerik pada persamaan diferensial *bungee jumping* seperti pada Gambar 5. Jumlah epochs yang besar dapat mengurangi kesalahan dalam perhitungan.



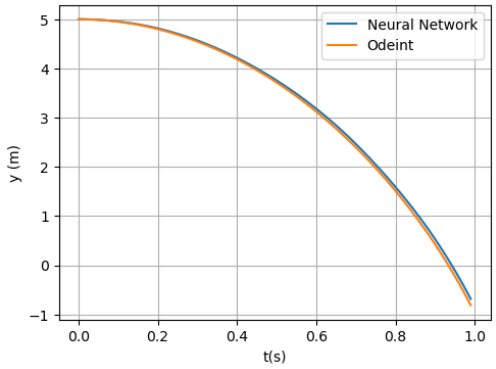
(a) 100 epochs



(b) 500 epochs



(c) 1000 epochs

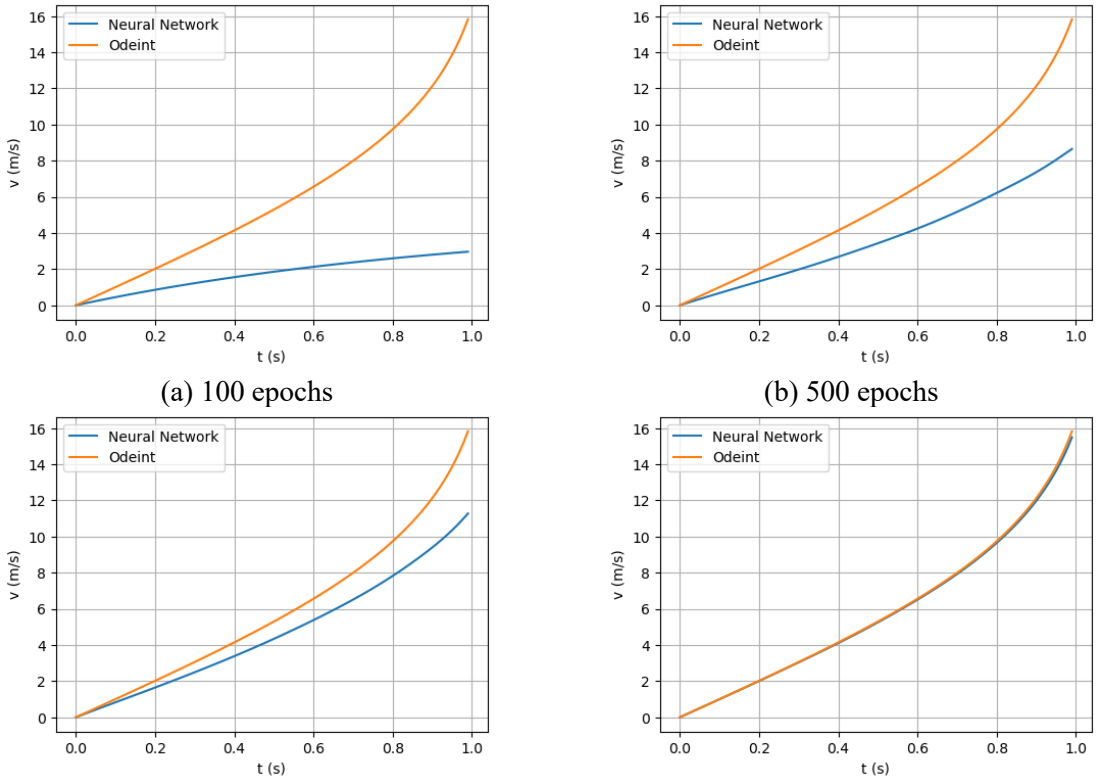


(d) 5000 epochs

Gambar 5. Grafik Perbandingan y Neural Network dan Odeint

Dari hasil Gambar 6 merupakan penyelesaian y terhadap t menggunakan *neural network* dengan menggunakan epochs yang berbeda-beda. Pada gambar (a), (b), (c) menggunakan epochs 100, 500, dan

1000 dengan hasil jauh mendekati grafik y pada Odeint. Pada gambar (d) diberikan epochs 5000, dan hasil grafik y mendekati hasil dari grafik Odeint.



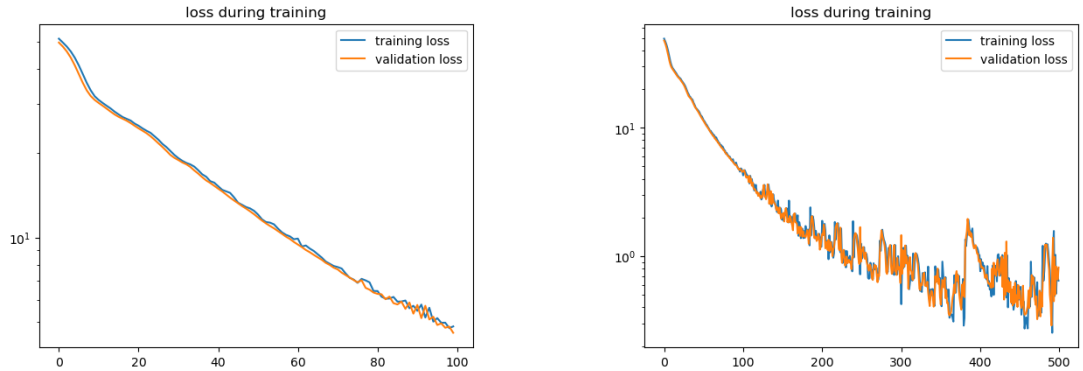
(a) 100 epochs (b) 500 epochs

(c) 1000 epochs (d) 5000 epochs

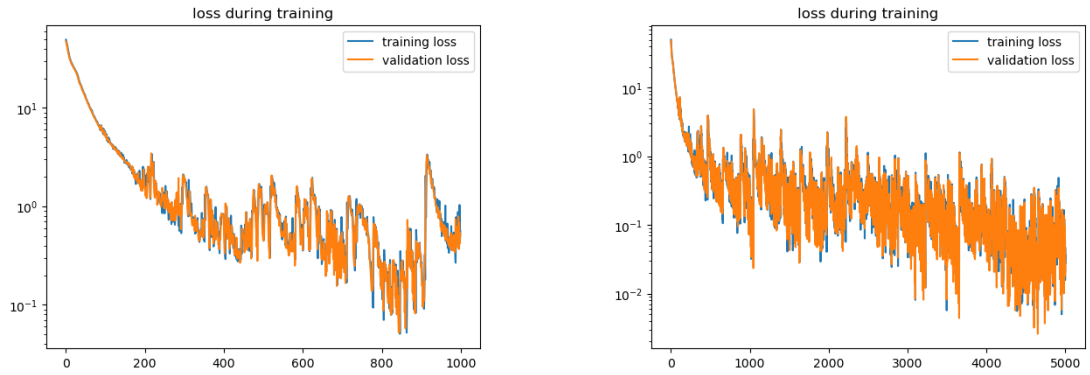
Gambar 6. Grafik Perbandingan v Neural Network

Sama halnya pada Gambar 5 dan 6 merupakan penyelesaian v terhadap t *neural network* dengan epochs yang berbeda-beda. Pada gambar (d) dengan epochs 5000 grafik *neural network* membentuk garis lurus yang mendekati grafik Odeint.

Jumlah epochs merupakan parameter penting yang dapat mempengaruhi kinerja model. Namun jumlah epochs yang sedikit dapat menyebabkan *underfitting*, sedangkan jumlah epochs yang terlalu banyak dapat menyebabkan *overfitting*. Untuk itu kita dapat menggunakan epochs dengan jumlah yang tepat. Dalam meningkatkan akurasi dan mengurangi kesalahan, dapat meningkatkan jumlah epochs, namun membutuhkan proses pelatihan yang lama.



(a) 100 epochs (b) 500 epochs



(c) 1000 epochs (d) 5000 epochs

Gambar 7. Grafik *Training loss* dan *Validation loss*

Training loss dan *validation loss* digunakan untuk menilai kemampuan dari *neural network. Training loss* merupakan kesalahan pada *dataset* setiap pelatihan [15]. *Validation loss* merupakan kesalahan pada dataset validasi yang terpisah dan dihitung pada setiap epochs dan digunakan untuk mengevaluasi seberapa baik model menggeneralisasi data baru yang tidak terlihat[16]. Analisis *training loss* dan *validation loss* dapat menunjukkan kesetabilan dan generalisasi pada *neural network* dalam simulasi persamaan gerak pada *bungee jumping*. Untuk menilai kemampuan dan keakuratan *neural network* dalam simulasi dalam persamaan diferensial biasa pada bungee jumping diperlukan perbandingan grafik dan mengevaluasi loss dengan mengubah berbagai epochs. Semakin besar epochs yang diterapkan akan menghasilkan *training loss* yang kecil [17].

4. Simpulan

Persamaan gerak *unstretch bungee jumping* dapat diselesaikan menggunakan solusi numerik dengan metode Euler dan Odeint. Hasil dari metode Euler divalidasi menggunkan metode Odeint dan diimplementasikan pada *neural network.* Pada *neural network* dilakukan perubahan epochs, dimana semakin tinggi epochs yang digunakan maka hasilnya lebih akurat dan mendekati dari hasil Odeint.

Daftar Pustaka

[1] Menz P G 1993 The physics of bungee jumping *Phys Teach* **31**(8) p 483–487 doi: 10.1119/1.2343852.
[2] Heck A and Uylings P 2010 Understanding the physics of bungee jumping [Online] Available: www.iop.org/journals/physed
[3] Heck A and Uylings P 2010 Understanding the physics of bungee jumping [Online] Available: www.iop.org/journals/physed
[4] Kagan D and Kott A 1996 The greater-than- g acceleration of a bungee jumper *Phys Teach* **34**(6) p 368–373
[5] Heck A and Uylings P 2020 A Lagrangian approach to bungee jumping *Physics Education* **55** 0250099
[6] Lawson D A 1995 Potential Perils of Euler's Method
[7] Biswas B N, Chatterjee S, Mukherjee S P, and Pal S 2013 A Discussion On Euler Method: A Review [Online] Available: http://ejmaa.6te.net/
[8] Abiodun O I, Jantan A, Omolara A E, Dada K V, Mohamed N A, and Arshad H 2018 State-of-the-art in artificial neural network applications: A survey *Heliyon* **4** p. e00938
[9] Kloeden P E and Pearson R A 1977 The numerical solution of stochastic differential equations *The Journal of the Australian Mathematical Society. Series B. Applied Mathematics* **20**(1) p 8–12

[10] Brown S D, Ratcliff R, and Smith P L 2006 Evaluating methods for approximating stochastic differential equations *J Math Psychol* **50**(4) p 402–410 doi: 10.1016/j.jmp.2006.03.004.
[11] Wu Y C and Feng J W 2018 Development and Application of Artificial Neural Network *Wirel Pers Commun* **102**(2) p 1645–1656
[12] Bulsari A 1993 Some analytical solutions to the general approximation problem for feedforward neural networks *Neural Networks* **6**(7) p 991–996
[13] S. Karsoliya 2012 Approximating Number of Hidden layer neurons in Multiple Hidden Layer BPNN Architecture *International Journal of Engineering Trends and Technology* [Online] Available: http://www.internationaljournalssrg.org
[14] Balcázar J L R, Gavaldà, and Siegelmann H T 1997 Computational Power of Neural Networks: A Characterization in Terms of Kolmogorov Complexity
[15] Kristal A dan Harintaka 2022 Analisis Kehandalan Ekstraksi Garis Tepi Bangunan dari Data Foto Udara Menggunakan Pendekatan *Deep Learning* Berbasis *Mask* R-CNN *Journal of Geodesy and Geomatics* **17**(2) p 273-285
[16] Watanabe S 2010 Asymptotic Equivalence of Bayes Cross Validation and Widely Applicable Information Criterion in Singular Learning Theory *Journal of Machine Learning Research 11* 3571-3594.
[17] Takase T, Oyama S, and Kurihara M 2018 Effective Neural Network Training with Adaptive Learning Rate based on Training Loss *Neutral Networks 101* p 68-78