MODEL PHYSICS-INFORMED NEURAL NETWORK WAKTU DISKRIT PADA PERSAMAAN DIFERENSIAL PANAS

SKRIPSI

Lintang Bristian
11160940000034

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UIN SYARIF HIDAYATULLAH JAKARTA
2023/1444 H

PERNYATAAN

DENGAN INI SAYA MENYATAKAN BAHWA SKRIPSI INI BENAR-BENAR HASIL KARYA SAYA SENDIRI YANG BELUM PERNAH DIAJUKAN SEBAGAI SKRIPSI ATAU KARYA ILMIAH PADA PERGURUAN TINGGI ATAU LEMBAGA MANAPUN.

Jakarta, Februari 2023

Lintang Bristian
NIM. 11160940000034

LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini:

Nama : Lintang Bristian

NIM 11160940000034

Program Studi : Matematika Fakultas Sains dan Teknologi

Demi pengembangan ilmu pengetahuan, saya menyetujui untuk memberikan Hak Bebas Royalti Non-Eksklusif *(Non-Exclusive-Free Right)* kepada Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta atas karya ilmiah saya yang berjudul:

"Model Physics-Informed Neural Network Waktu Diskrit Pada Persamaan Diferensial Panas"

Beserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-Eksklusif ini, Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta berhak menyimpan, mengalihmedia/formatkan, mengelolanya dalam bentuk pangkalan data *(database)*, mendistribusikannya, dan menampilkan/mempublikasikannya di internet dan media lain untuk kepentingan akademis tanpa perlu meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta. Segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta karya ilmiah ini menjadi tanggung jawab saya sebagai penulis.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Dibuat di Tangerang Selatan

Pada tanggal: 6 Februari 2023

Yang membuat pernyataan

(Lintang Bristian)

PERSEMBAHAN

Skripsi ini saya persembahkan untuk kedua orang tua saya. Terima kasih telah mendidik dan mengajarkan hal-hal baik sejak awal saya lahir hingga detik ini.

MOTTO

“Lebih baik dikenal menjadi orang baik”

ABSTRAK

Lintang Bristian, Model Physics-Informed Neural Network Waktu Diskrit Pada Persamaan Diferensial Panas dibawah bimbingan **Bapak Taufik Edy Sutanto, M.Sc. Tech., Ph.D** dan **Dr. Gustina Elfiyanti, M.Si**

Seiring dengan perkembangan zaman, *neural network* dapat menyelesaikan sebuah permasalahan persamaan diferensial. Dalam penerapannya, persamaan diferensial dapat memodelkan berbagai bidang fisika yang sering juga ditemukan kondisi jumlah data besar atau bahkan persamaan yang cukup kompleks. Pada permasalahan inilah metode *Physics-informed Neural Network* (PINN) digunakan untuk memecahkan permasalahan tersebut dengan memasukkan persamaan fisika ke dalam arsitektur *neural network*. Metode PINN yang dipilih adalah model waktu diskrit, yang bekerja memprediksi solusi pada langkah waktu selanjutnya dengan asumsi solusi pada langkah waktu t^n diketahui. Pada Skripsi ini meneliti performa PINN pada persamaan differensial panas. Model yang digunakan menghasilkan nilai *error* sebesar 1.541920e-02. Penelitian pendahuluan PINN ini diharapkan dapat menginspirasi penelitian selanjutnya pada penggunaan machine learning dalam menyelesaikan berbagai permasalahan terkait persamaan differensial.

Kata Kunci: Persamaan Panas, Physics-Informed Neural Network**,** Waktu Diskrit.

ABSTRACT

Lintang Bristian, Discrete Time Physics-Informed Neural Network Model in the Heat Equation under the guidance of **Mr. Taufik Edy Sutanto, M.Sc. Tech., Ph.D and Dr. Gustina Elfiyanti, M.Sc**

Along with the times, neural networks can solve a differential equation problem. In practice, differential equations can model various fields of physics where conditions of large amounts of data or even fairly complex equations are often found. It is on this problem that the Physics-informed Neural Network (PINN) method is used to solve this problem by incorporating physical equations into the neural network architecture. The chosen PINN method is the discrete time model, which works to predict the solution in the next time step with the assumption that the solution in the t^n time step is known. This thesis examines the performance of PINN in the heat differential equation. The model used produces an error value of 1.541920e-02. This preliminary research on PINN is expected to inspire further research on the use of machine learning in solving various problems related to differential equations.

Keywords: Heat Equation, Physics-Informed Neural Network, Time Discrete

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillah, puji syukur kehadirat Allah SWT atas rahmat serta karunia-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi yang berjudul **"**Metode *Physics-Informed Neural Network* Waktu Diskrit Pada Persamaan Diferensial Panas**".**

Terselesaikannya skripsi ini tidak lepas dari bantuan, bimbingan, saran dan dukungan dari berbagai pihak. Maka dari itu penulis mengucapkan terima kasih sebesar-besarnya kepada:

1. Bapak Nashrul Hakiem, S.Si., M.T., Ph.D selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta.
2. Ibu Dr. Suma'inna, M.Si, selaku Ketua Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta dan Ibu Irma Fauziah, M.Sc, selaku Sekretaris Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta.
3. Bapak Taufik Edy Sutanto, M.Sc. Tech., Ph.D, selaku pembimbing I dan Ibu Dr. Gustina Elfiyanti, M.Si selaku pembimbing II terimakasih atas saran, arahan, nasihat dan ilmu yang diberikan kepada penulis sehingga skripsi ini dapat terselesaikan.
4. Ibu Dr. Nina Fitriyati, S.Si., M.Kom. selaku penguji I dan Ibu Dr. Nur Inayah, S.Pd., M.Si. selaku penguji II, terima kasih atas kritik dan sarannya kepada penulis selama melaksanakan seminar hasil dan sidang skripsi.
5. Kedua orang tua penulis Bapak Bristiyono Pratikto dan Ibu Ratmi Amikasari yang tiada henti memberikan doa, semangat, kasih sayang. Dan juga kakak, Lintang Briskarisma yang selalu memberikan cerita pengalamannya dalam mengerjakan skripsi.

6. Rekan perjuangan, Azahra Benita yang selalu memberi semangat, motivasi, dan saran sehingga skripsi ini dapat terselesaikan.
7. Maftuh Mahsuri, Akmal Tanjung, Abu jafar Yazid dan segenap rekan kosan yang selalu memberi wawasan dan pengalamannya kepada penulis untuk dapat menyelesaikan skripsi ini.
8. HIMATIKA UIN Jakarta yang sudah memberikan banyak sekali pengalaman dalam berorganisasi kepada penulis.
9. Teman-teman Matematika 2016 yang sudah membantu penulis selama proses perkuliahan yang tidak dapat disebutkan satu-persatu.
10. Seluruh pihak yang telah berkontribusi dalam penyusunan skripsi ini.

Penulis menyadari bahwa skripsi ini masih terdapat banyak kekurangan, oleh karena itu penulis mengharapkan masukan melalui kritik dan saran yang bersifat membangun untuk menyempurnakan penelitian di masa yang akan datang melalui *email* penulis *lintangbristian@gmail.com*. Akhir kata, penulis berharap semoga skripsi ini dapat bermanfaat bagi pembaca.

Wassalamualaikum Warahmatullahi Wabarakatuh.

Jakarta, Februari 2023

Penulis

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Matematika merupakan cabang ilmu pengetahuan yang mempunyai peranan yang penting dalam perkembangan ilmu pengetahuan dan teknologi, baik dalam penerapan di bidang ilmu-ilmu lain maupun dalam pengembangan matematika itu sendiri. Allah SWT berfirman kepada manusia pada surat Al-Alaq ayat 1:

اِقْرَأْ بِاسْمِ رَبِّكَ الَّذِيْ خَلَقَۚ ١

Artinya:
"Bacalah dengan (menyebut) nama Tuhanmu yang telah mencipatakan."

Ayat dalam surat tersebut berarti bahwa manusia telah diperintahkan untuk membaca guna memperoleh berbagai pemikiran dan pemahaman terhadap ilmu. Alam semesta yang diciptakan Allah SWT ini banyak mengandung ilmu pengetahuan. Allah SWT menciptakan alam semesta ini agar dipelajari oleh manusia sebagai suatu ilmu pengetahuan. Allah SWT juga memberikan ilmu pengetahuan kepada manusia sejak awal penciptaan manusia sebagai pembeda dengan makhluk lainnya oleh karena itu sebagai umat islam, sebagaiman pernitah Allah pada ayat tersebut kita harus memperluas wawasan ilmu pengetahuan. Ilmu pengetahuan dan teknologi sangat berkaitan erat dengan matematika. Mulai dari cabang ilmu matematika yang dasar, hingga ke ilmu yang lebih kompleks, matematika sangat luas digunakan. Salah satu contohnya adalah persamaan diferensial.

Persamaan diferensial merupakan salah satu topik dalam matematika yang penerapannya banyak ditemukan dalam permasalahan kehidupan sehari-hari. Persamaan diferensial dibagi menjadi dua, yaitu persamaan diferensial biasa dan persamaan diferensial Parsial [2]. Dalam penerapannya, terdapat banyak pemodelan matematika pada bidang fisika dan teknik yang ditemukan dalam

persamaan diferensial. Contohnya seperti model persamaan panas, persamaan Laplace, dan persamaan gelombang. Untuk menentukan solusi dari persamaan tersebut, terdapat beberapa metode yang bisa dilakukan, seperti metode pemisahan variabel, metode kanonik, metode d'Alembert, metode transformasi Laplace. Namun dalam penyelesaiannya, terkadang tak jarang juga ditemukan kondisi dimana jumlah data pengukuran yang sangat besar atau bahkan persamaan yang cukup kompleks [3]. Sehingga hal tersebut dapat menjadi kendala karena membutuhkan waktu dan tenaga yang cukup besar untuk menemukan solusi dari persamaan pemodelan tersebut.

Dalam perkembangan teknologi saat ini, para ilmuwan mulai mengeksplor kegunaan *neural network* dalam memecahkan masalah dalam berbagai domain terapan seperti visi komputer, pemrosesan bahasa alami, dan lain-lain. Akhir-akhir ini telah dikembangkan kegunaan dari *neural network* dalam bidang fisika dan teknik untuk memecahkan solusi dari persamaan diferensial, yang dikenal dengan *Physics Informed Neural Network* (PINN).

Gagasan utama dari PINN adalah untuk memecahkan masalah di mana data yang tersedia hanya terbatas. Contohnya, pengukuran yang *noisy* dari suatu percobaan. Sehingga untuk mengimbangi kelangkaan data tersebut, algoritma yang berkaitan dengan PINN mengacu pada aturan hukum fisika yang selanjutnya dijelaskan secara umum pada persamaan diferensial parsial non-linear berikut.

$$\frac{\partial t}{\partial u} + N[u; \lambda] = 0, x \in \Omega, t \in T. \quad (1.1)$$

Di sini, solusi laten $u(t, x)$ bergantung pada waktu $t \in [0, T]$ dan variabel spasial $x \in \Omega$, di mana mengacu pada ruang di R^D. Selanjutnya, $N[u; \lambda]$ merupakan operator diferensial non-linear dengan koefisien λ.

PINN terbagi menjadi dua model yang dapat digunakan sesuai kebutuhannya. Yaitu, model waktu kontinu dan model waktu diskrit. Model waktu kontinu membutuhkan *collocation Points* pada tahapan *feed-forward*. *Collocation Points* adalah titik yang berada di domain pada metode kolokasi untuk menyelesaikan sebuah persamaan diferensial. Model waktu kontinu membutuhkan

Collocation Points pada tahapan *feed-forward*. Model kontinu sendiri hanya membutuhkan beberapa data saja diantaranya batas atas dan batas bawah yang akan digunakan untuk menginterpolasi sehingga menjadi *Collocation Points*. *Collocation Points* tersebut yang dihasilkan kemudian dapat digunakan untuk mengevaluasi model atau algoritma dalam berbagai kondisi yang berbeda. Namun untuk menghindari kebutuhan *Collocation Points*, Raissi mengusulkan sebuah model yang bernama model waktu diskrit untuk masalah *forward* dan *inverse*. Model ini melakukan pendekatan solusi-inferensi alternatif berdasarkan skema *time-stepping* Runge-Kutta untuk memprediksi solusi $u(x)$ pada waktu t^{n+1} berdasarkan waktu ke t^n. Keuntungan dari model ini tidak memerlukan *Collocation Points* karena hanya solusi untuk keadaan awal pada t^n dan kondisi batas yang harus disediakan.

Berdasarkan latar belakang yang telah diuraikan diatas, penelitian ini merupakan studi literatur terkait metode *Physics-Informed Neural Network (PINN)* waktu diskrit berdasarkan penelitian terdahulu yang telah dilakukan oleh Raissi yang berjudul *Physics-informed neural networks*: *A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*. Penelitian ini akan membahas bagaimana *Physics-Informed Neural Network* melakukan pendekatan dalam memprediksi solusi numerik persamaan diferensial panas.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, permasalahan penelitian skripsi ini adalah seberapa akurat model *Physics-Informed Neural Network* waktu Diskrit dalam memprediksi solusi numerik persamaan diferensial panas?

1.3 Tujuan Penelitian

Penelitian ini dilakukan dengan tujuan untuk mengimplementasikan model *Physics-Informed Neural Networks* Waktu Diskrit dalam memprediksi solusi numerik persamaan diferensial panas.

1.4 Batasan Masalah

Dalam penelitian ini, penulis memberikan batasan masalah agar penelitian lebih terarah. Batasan masalah pada penelitian ini sebagai berikut:

1. Persamaan Diferensial yang digunakan adalah persamaan diferensial panas.
2. Solusi eksak dan pendekatan dari persamaan diferensial panas tidak dibahas, dan skripsi hanya fokus pada pendekatan numerik menggunakan *machine learning.*

1.5 Manfaat Penelitian

Penulis berharap dalam penelitian ini dapat mengetahui kegunaan dari *Physic-Informed Neural Network* dalam mencari solusi pada persamaan diferensial panas.

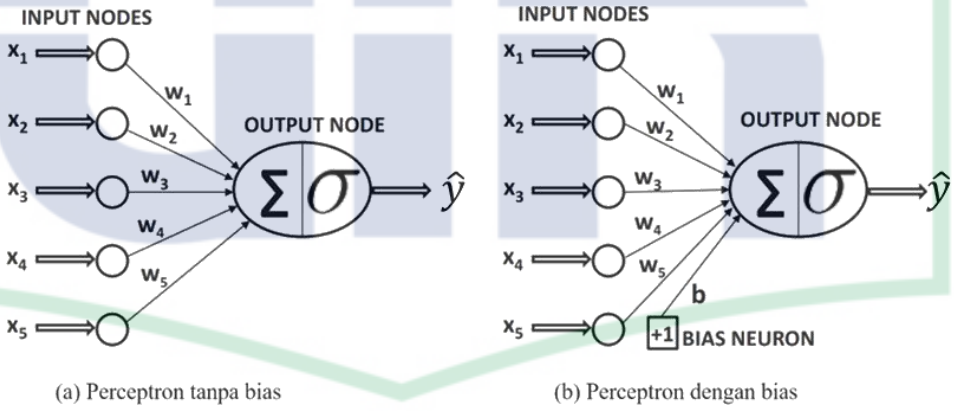
BAB II
LANDASAN TEORI

2.1 Neural Network

Neural network adalah metode dalam kecerdasan buatan yang mengajarkan komputer untuk memproses data yang cara kerjanya terinspirasi dari sistem otak manusia. *Neural network* menciptakan sistem adaptif yang digunakan oleh komputer untuk belajar dari kesalahannya dan memperbaikinya secara terus-menerus. Oleh karena itu, *neural network* belajar dari data yang ada, lalu mencari solusi dari masalah yang dihadapi secara otomatis

Bentuk paling sederhana dari *Neural network* adalah *perceptron*. *Perceptron* berisi sebuah lapisan *input* dan node *output* [1]. Koneksi antara node *input* dan *output* dapat diinterpretasikan sebagai bobot yang mengontrol pentingnya *input* mereka. Arsitektur dasar *perceptron* ditunjukkan pada gambar dibawah.

Misal diberikan suatu permasalahan dengan *input* $X = [x_1, x_2 \ldots, x_m]$

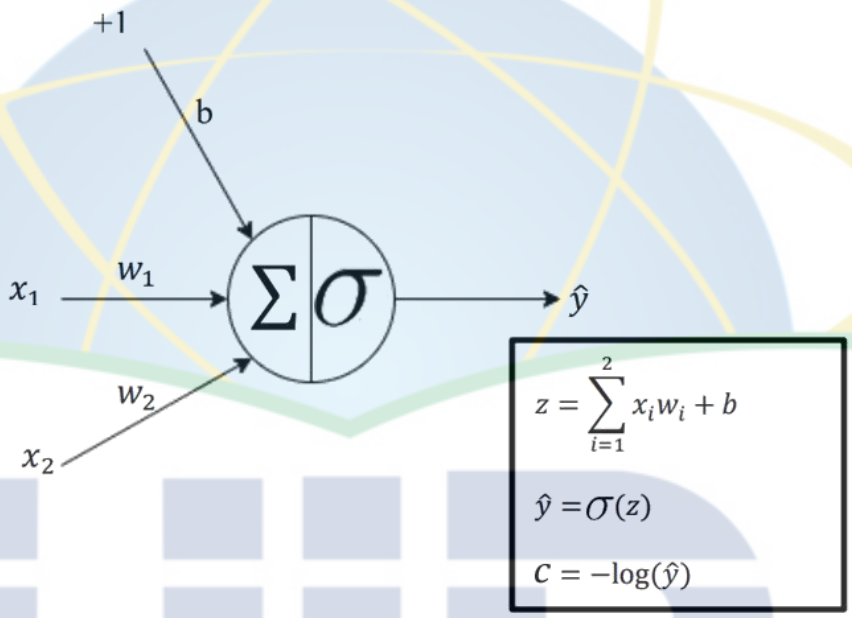


Gambar 2. 1 Arsitektur Dasar dari *Perceptron*

Sumber: Aggarwal C [1]

2.1.2 Feed-Forward Neural Network

Misal diberikan suatu permasalahan dengan *input* $X = [x_1, x_2. . . , x_m]$ kemudian setiap *input* x_i dikali dengan bobot w_i, dimana $W = [w_1, x_2, \dots , w_m]$ berikutnya sebuah fungsi nonlinier f memetakan jumlah dari hasil kali x_i dengan w_i, dan ditambah bias b ke *output* y. Operasi tersebut umum disebut dengan *feed forward* atau *forward propagation*.



Gambar 2. 2 Ilustrasi *feed-forward neural network*

Feed-forward neural network merupakan *neural network* yang koneksi antar node-nya tidak membentuk sebuah siklus. Bentuk algoritma dari *feed-forward neural network* merupakan bentuk algorima paling sederhana, karena informasi hanya diproses dalam satu arah. Dengan kata lain, *neural network feed-forward* hanya memungkinkan sinyal untuk melakukan perjalanan melalui satu jalur saja, yakni dari *input* menuju ke *output*.

Algoritma *Feed-forward neural network* ini sering pula disebut sebagai *multilayer perceptron*. *Multilayer neural network* merupakan *neural network* yang dikembangkan dari algoritma *perceptron* yang terdiri dari beberapa layer komputasi yaitu *input*, *output* dan terdapat *hidden layer* diantara keduanya. Dapat dilihat

bahwa *feed-forward* membentuk lapisannya dengan menggabungkan beberapa *perceptron*. setiap node pada *hidden layer* dan *output* node menggunakan suatu fungsi linear atau nonlinier dan memprosesnya untuk menentukan nilai dari *output* node.

2.1.2 Fungsi Aktivasi

Dalam *neural network*, setiap node pada *hidden layer* dan *output* node menggunakan suatu fungsi linear atau nonlinier yang fungsinya untuk menentukan nilai dari *output* node tersebut. fungsi tersebut dikenal dengan fungsi aktivasi [4]. Tugas dari fungsi aktivasi yaitu menentukan diaktifkan atau tidaknya suatu neuron dengan menghitung jumlah bobot dan menambahkan biasnya. Fungsi aktivasi yang digunakan pada algoritma *perceptron* ini dapat berbeda-beda jenisnya, bergantung pada permasalahan yang sedang diamati. Ada beberapa jenis fungsi aktivasi, Fungsi aktivasi yang paling umum digunakan yaitu fungsi perceptron, fungsi sigmoid, fungsi tanh dan ReLu, namun tentu ada beberapa fungsi aktivasi lainnya.

$$\sigma(z) = z \quad \text{(fungsi linear)} \tag{2.1}$$

$$\sigma(z) = sign(z) \quad \text{(fungsi tangga)} \tag{2.2}$$

$$\sigma(z) = \{z, 0\} \quad \text{(fungsi ReLU)} \tag{2.3}$$

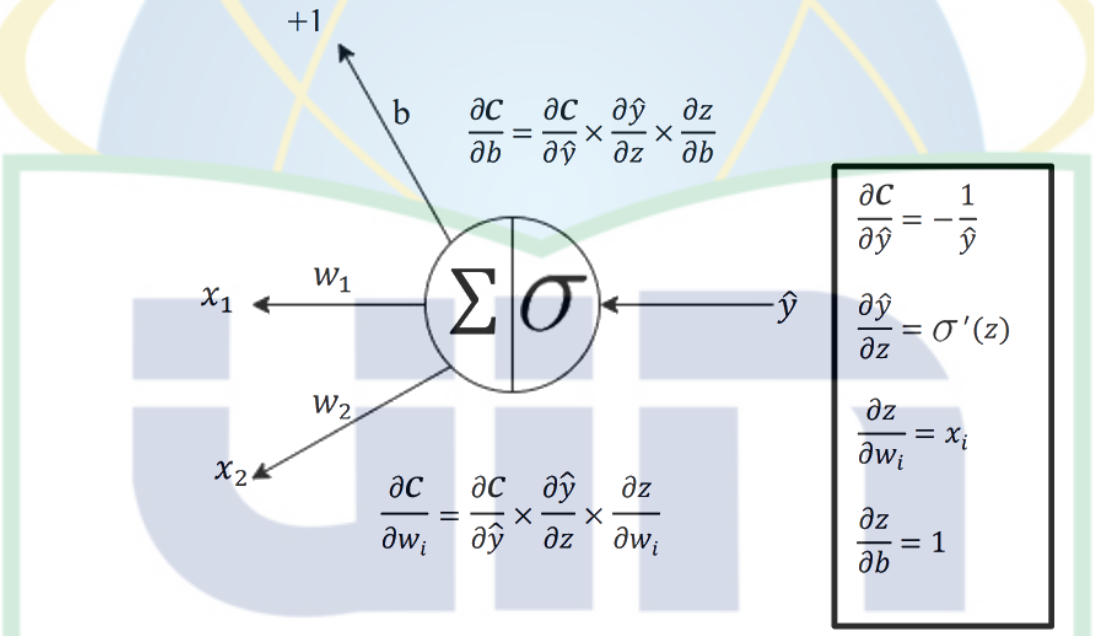
$$\sigma(z) = \{\{z, 1\}, -1\} \quad \text{(fungsi } \textit{hard} \text{ tanh)} \tag{2.4}$$

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad \text{(fungsi sigmoid)} \tag{2.5}$$

$$\sigma(z) = \frac{e^{2z} - 1}{e^{2z} + 1} \quad \text{(fungsi tanh)} \tag{2.6}$$

2.1.3 Backward Propagation

Pada backward Propagation, perhitungan gradient dilakukan dari fungsi *cost* berdasarkan berbagai parameter bobot (w) dan bias (b) setiap layer. Pada tahapan ini gradient dihitung dengan turunan parsial secara mundur, yakni gradient dihitung dari fungsi *cost* (C) hingga input layer dengan memperhatikan bobot dan bias pada layer tersebut. Setelah itu nilai gradient dari fungsi *cost* terhadap bobot atau bias akan digunakan untuk optimasi parameter didalam *neural network* dengan algoritma optimasi berbasis *gradient-descent* [1]. Ilustrasi *backward propagation* pada *neural network* dengan 2 *input* dan sebuah node dapat dilihat pada **Gambar** 2.3



Gambar 2. 3 Illustrasi *backward propagation*

Dalam memperbarui bobot jaringan, perlu dilakukannya penurunan gradien. Penurunan gradien ini bertujuan untuk meminimalkan fungsi *cost* pada jaringan dengan cara "menggeser" parameter sepanjang arah negatif gradien [1]. Gradien tidak lain adalah vektor yang menyimpan semua turunan parsial sehubungan dengan semua parameter fungsi. Selama setiap langkah iterasi atau

epoch, untuk menggunakan terminologi *machine learning*, parameter model diperbarui sesuai dengan aturan berikut:

$$w^* = w_i - \alpha \frac{\partial C}{\partial w_i}, \quad b^* = b_i - \alpha \frac{\partial C}{\partial b_i}. \tag{2.7}$$

Dimana w^* adalah yang telah *diupdate*, b^*adalah bias yang telah *diupdate*, $w_i$dan b_i adalah bobot dan bias terkini atau belum terupdate, α adalah *learning rate* dan C adalah fungsi *cost*. Algoritma pengoptimalan yang digunakan adalah *optimizer Adam* dan optimizer L-BFSG.

2.2 Persamaan Diferensial

Persamaan diferensial adalah persamaan yang memuat turunan-turunan dari satu atau lebih variabel tak bebas terhadap satu atau lebih variabel bebas [2].. Persamaan diferensial dibagi menjadi dua, yaitu Persamaan diferensial biasa dan Persamaan diferensial parsial [2]. Salah satu contoh bentuk persamaan diferensial adalah sebagai berikut:

$$\frac{dx}{dy} = x + y \tag{2.8}$$

Berdasarkan persamaan diatas, x menyatakan variabel bebas dan y menyatakan variabel tak bebas/terikat. Persamaan diferensial dibagi menjadi 2 jenis, yaitu persamaan diferensial biasa dan persamaan diferensial parsial. Persamaan diferensial biasa adalah suatu persamaan yang memuat turunan-turunan dari satu atau lebih variabel tak bebas terhadap satu variabel bebas [2]. Sedangkan, Persamaan diferensial parsial adalah persamaan yang memuat turunan-turunan parsial dari satu atau lebih variabel tak bebas terhadap lebih dari satu variabel bebas [2]. Berdasarkan hubungan terhadap variabel tak bebasnya, persamaan diferensial terbagi menjadi dua, yaitu persamaan linear dan non-linear [2]. Persamaan diferensial dikatakan memiliki bentuk linear jika memenuhi syarat-syarat berikut ini:

1. Derajat dari variabel tak bebas dan turunan-turunannya adalah satu.

2. Tidak ada perkalian antara variabel tak bebas dengan turunan-turunannya maupun perkalian antara turunan dengan turunannya.

3. Tidak ada fungsi transenden dari variabel-variabel tak bebas

Sehingga berdasarkan uraian diatas, apabila Persamaan diferensial yang tidak memenuhi ketiga syarat tersebut dapat dikatakan sebagai persamaan diferensial non linear.

Dalam menentukan solusi dari sebuah Persamaan diferensial, dibutuhkan masalah nilai awal dan syarat batas. Khususnya dalam mencari solusi dari Persamaan Diferensial, yang memerlukan dua bentuk kondisi tertentu karena terdapat lebih dari satu variabel bebas. Menurut Strauss, untuk memecahkan solusi tersebut dibutuhkan kondusi awal dan kondisi batas [7].

2.3 Metode Runge-Kutta

Metode Runge-Kutta adalah suatu metode numerik yang digunakan untuk menyelesaikan masalah nilai awal atau masalah nilai batas pada persamaan differensial linear atau nonlinear [12]. Bentuk umum metode Runge-Kutta orde-n didefinisikan pada persamaan (2.9):

$$y_{r+1} = y_r + a_1k_1 + a_2k_2 + ... + a_n\, k_n \tag{2.9}$$

dimana nilai $a_{1,}\, a_{2,\ldots,}\, a_n$ merupakan konstanta dan semua nilai k berhubungan secara rekurensi, yang artinya nilai k sebelumnya akan muncul di persamaan k selanjutnya:

$$k_1 = hf\,(x_r\,,\, y_r) \tag{2.10}$$

$$k_2 = hf\,(x_r + p_1h,\, y_r + p_1k_1) \tag{2.11}$$

$$k_3 = hf\,(x_r + p_2h,\, y_r + p_2k_2) \tag{2.12}$$

.

.

$$k_n = hf\,(x_r + p_{n-1}h,\, y_r + p_{n-1}k_{n-1}) \tag{2.13}$$

dimana h adalah langkah atau *step*, p adalah konstanta, x_r langkah waktu saat ini dan y_r adalah solusi pada langkah waktu saat ini. Metode Runge-Kutta sendiri dapat dibedakan berdasarkan jumlah ordenya.

Contoh

Untuk mendapatkan solusi dari suatu persamaan diferensial parsial pada x = 0,2 dengan step 0,2 dengan nilai awal telah ditentukan yaitu $y(0) = 1$

$$y' - y = x \tag{2.14}$$

Maka berdasarkan persamaan (2.14), diperoleh persamaan (2.15)

$$f(x_r , y_r) = x + y \tag{2.15}$$

karena persamaan ini merupakan Runge-Kutta berorde 2, maka persamaan yang digunakan adalah persamaan (2.16)

$$y_{r+1} = y_r + {}^1/_2 (k_1 + k_2) \tag{2.16}$$

Selanjutnya dapat ditentukan nilai dari k_1 dan k_2, dimana dengan melihat persamaan (2.10) dan (2.11) didapatkan

$$k_1 = (0,2)\ (0 + 1) = 0,2$$

$$k_2 = (0,2)\ (0,1 + 1,2) = 0,26$$

Sehingga dengan nilai x = 0,2, persamaan (2.16) menjadi

$$y(0,2) = y(0) + {}^1/_2 (k_1 + k_2)$$

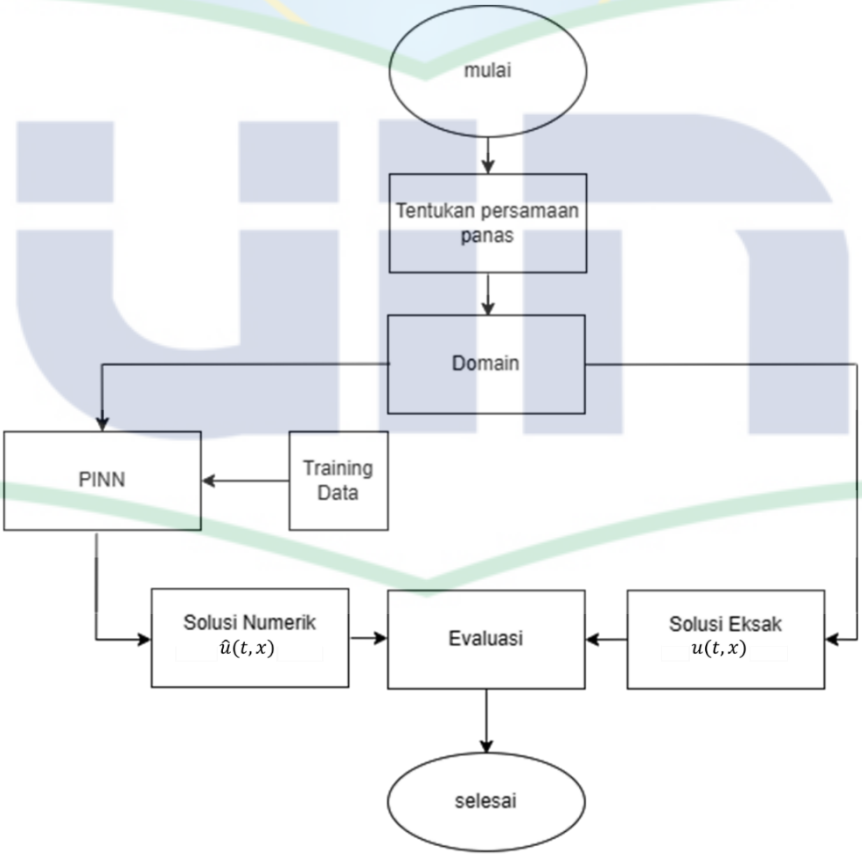
$$= 1 + {}^1/_2 (0,2 + 0,26)$$

$$= 1,23$$

Maka didapatkanlah solusi persamaan diferensial (2.14) diatas sebesar 1,23.

BAB III
PHYSICS-INFORMED NEURAL NETWORK

PINN (*Physics-Informed Neural Network*) waktu diskrit dilakukan dengan beberapa langkah, dimulai memilih PDE, memilih domain spasial dan temporal, membuat data latih, membangun arsitektur, melatih *neural network*, mengevaluasi performa. Arsitektur PINN terdiri dari *neural network feed-forward* yang menggunakan koordinat spasial dan temporal sebagai input dan mengeluarkan nilai yang diprediksi dari solusi, serta melibatkan *backpropagation* melalui PDE untuk menghitung gradien terhadap koordinat input. Setelah jaringan dilatih, dapat membandingkan solusi prediksi dengan solusi eksak atau solusi numerik. Berikut adalah alur penelitian yang yang digambarkan pada **Gambar 3.1.**



Gambar 3. 1 Alur Penelitian

1. Persamaan Panas

Telah ditentukan persamaan diferensial parsial yang digunakan adalah persamaan panas, yaitu:

$$c\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}(\kappa \frac{\partial u}{\partial x}) - s = 0, t \in [0,0.5], x \in [0,1] \quad (3.1)$$

Dengan $c(u)$ kapasitas panas dan $\kappa(u)$ konduktivitas termal yang didefinisikan sebagai

$$c(u) = 1/2000u^2 + 500$$
$$\kappa(u) = 1/100u + 7 \quad (3.2)$$

pada kondisi batas *Dirichlet* yang homogen

$$u(0,t) = u(1,t) = 0 \quad (3.3)$$

dan masalahnya tunduk pada kondisi awal

$$u(0,x) = u0 \quad (3.4)$$

2. Training Data

Dalam memecahkan solusi pada kasus model waktu diskrit digunakan *time-stepping* Runge-kutta. Pada metode hanya diprediksi pada langkah waktu tertentu dengan asumsi solusi pada langkah waktu t^n diketahui, maka metode yang diusulkan dapat mampu memprediksi solusi pada langkah waktu selanjutnya $t^{n+1} = t^n + \Delta t$ dimana Δt menunjukkan ukuran langkah model [9].

Sebelum melakukan training data, terdapat beberapa tahapan yang diperlukan, yaitu:

a) Inisialisasi sampel data *training* $\{x^i, u^{n,1}\}_{i=1}^{Nn}$ pada t^n

Data sampel yang digunakan untuk melatih PINN ditentukan pada (t) interval waktu [0, 0.5] dan (x) *domain space* [0, 1] Dengan batas *Dirichlet* {x | x = 0, x = 1}. Berikut adalah data yang digunakan:

Tabel 3. 1 Data (t) interval waktu

t
0
0.0025
0.005
…
0.495
0.4975
0.5

Tabel 3. 2 Data (x) *domain space*

x
0
0.00392157
0.00784314
…
0.99215686
0.99607843
1

b) Menentukan ukuran langkah waktu Δt

Ukuran langkah waktu dihitung dengan mengurangkan waktu awal t^0 dari waktu terakhir t^1. Ukuran langkah waktu digunakan saat menentukan model PINN, yang memungkinkan model memperhitungkan diskritisasi domain waktu saat membuat prediksi. Ditentukan $t^0 = 20$ bernilai 0,05 dan $t^1 = 120$ bernilai 0,3.

c) Menentukan jumlah tahapan q

q adalah jumlah tahapan atau jumlah bagian diskrit dalam suatu interval waktu menentukan jumlah langkah *intermediate* yang digunakan PINN untuk menghitung solusi pada langkah waktu berikutnya. Dalam penelitian ini, nilai q ditetapkan menjadi 50.

d) Menentukan arsitektur *neural network*

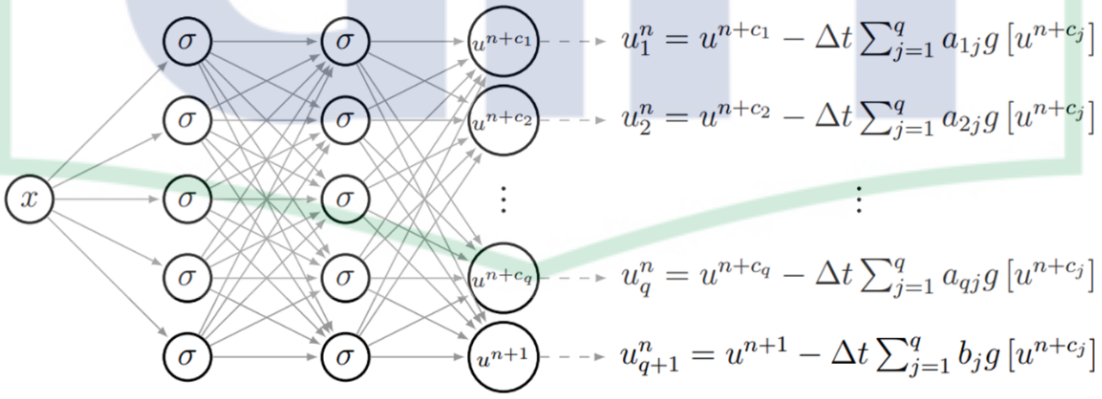
Neural network ini, memiliki empat lapisan (termasuk lapisan *input* dan lapisan *output*). Lapisan pertama memiliki satu neuron *input*, lapisan kedua, ketiga, dan keempat yaitu *hidden layer* masing-masing 20 neuron, dan lapisan kelima memiliki 51 neuron *output* karena telah ditentukan $q + 1$ neuron dengan nilai q berjumlah 50. Artinya jaringan tersebut memiliki total 1 + 20 + 20 + 20 + 51 = 112 neuron.

e) Inisialisasi parameter *neural network* (bias, bobot dan fungsi aktivasi)

Parameter *neural network* diwakili oleh bobot dan bias. Nilai awal bobot dan bias ditentukan oleh modul *pytorch* secara otomatis dan nilai bobot akan terus menerus diperbaharui sehingga *output* jaringan sedekat mungkin dengan *output* yang diinginkan untuk satu set input yang diberikan. Selain itu juga terdapat fungsi aktivasi diseluruh *hidden layer* yang ada dan menggunakan fungsi aktivasi $tanh$ pada persamaan (2.6).

f) Inisialisasi lapisan output dengan $q + 1$ neuron

Secara keseluruhan, jaringan belajar untuk memprediksi solusi pada waktu t^{n+1} berdasarkan solusi yang diketahui pada waktu t^n dan kondisi batas dalam interval waktu $[t^n, t^{n+1}]$ dengan meminimalkan fungsi *cost* yang sesuai.



Gambar 3. 2 Struktur neural network dengan satu node *input* dan $\boldsymbol{q + 1}$ *output*

Pada **Gambar 3.2**, *feed-forward neural network* menghasilkan prediksi u^{n+1} dan solusi *intermediate* u^{n+ci} , i $= 1, \ldots,$ q , untuk q tahap, yang kemudian digunakan dalam Persamaan (3.10) dan persamaan (3.11) untuk menghitung *output* yang dapat dibandingkan dengan solusi u^n pada waktu awal t^n.

Dengan mengasumsikan *neural network feed-forward* u^{n+1}_{NN} mampu memprediksi solusi u^{n+1} pada waktu t^{n+1} dan solusi antara u^{n+ci} pada semua tahap $i = 1, \ldots, q$ dari *input* x, maka *output*nya dapat ditulis sebagai:

$$[u^{n+c1}(x), \ldots, u^{n+cq}(x), u^{n+1}(x)] = u^{n+1}_{NN} \tag{3.7}$$

Sehingga kita dapat mendefinisikan *output* dari PINN u^n_{NN} untuk *input* x sebagai berikut:

$$[u^n_i(x), \ldots, u^n_q(x), u^n_{q+1}(x)] = u^n_{NN} \tag{3.8}$$

Sejauh ini, kondisi awal dan batas telah diberlakukan secara lemah dalam fungsi *cost*. Namun, juga memungkinkan untuk menerapkan kendala dalam arti yang kuat. Untuk melakukannya, solusi u dimodifikasi untuk memenuhi kondisi batas untuk setiap input yang diberikan. Mengikuti pendekatan umum Lagaris dkk [12]. untuk penerapan yang kuat, *output* jaringan dikalikan dengan $(1 - x)x$ untuk mempertimbangkan kondisi batas *Dirichlet* yang homogen seperti yang ditentukan dalam Persamaan (3.14). Karena solusi diskrit hanya tergantung pada variabel spasial x, output dapat ditulis sebagai

$$[\hat{u}^{n+c1}, \ldots, \hat{u}^{n+c4}, \hat{u}^{n+1}(x)] = (1 - x)xu^{n+1}_{NN}(x; \Theta) i = 1, \ldots, q \tag{3.9}$$

g) Menentukan *hyperparameter*

Selanjutnya menentukan *hyperparameter* utuk melatih jaringan menggunakan optimizer Adam dan L-BFGS. Untuk mengoptimasi suatu jaringan, diperlukan

adanya *hyperparameter*, salah satunya adalah *learning rate*. Learning rate ini menentukan ukuran langkah untuk pembaruan bobot yang dilakukan oleh *optimizer*. Secara umum, nilai *learning rate* yang lebih besar dapat menghasilkan konvergensi yang lebih cepat, tetapi juga dapat menyebabkan *optimizer* jauh daru titik maksimum global atau bahkan menyimpang. Sebaliknya, *Learning rate* yang lebih kecil bisa lebih stabil, tetapi juga bisa membuat proses optimasi lebih lambat. Dalam penelitian ini laju nilai *learning rate* untuk *optimizer* Adam adalah 0,001. Sedangkan nilai *learning rate* pada *optimizer* L-BFGS menggunakan nilai *default* yaitu 1.

Epoch juga ditentukan untuk mengoptimasi suatu *neural network*. Dalam hal ini, *epoch* pada *optimizer* Adam ditentukan, yaitu 10000 dan *epoch* pada *optimizer* L-BFGS ditentukan sebesar 10. Artinya, model akan dilatih dengan total 10000 + 10 = 10100 *epoch*.

Tabel 3. 3 Nilai *Hyperparameter*

Hyperparameter/Optimizer	**Optimizer Adam**	**Optimizer L-BFGS**
Learning Rate	0,001	1
Epoch	10000	10

Setelah Semua tahap yang diperlukan untuk mentraining data telah dipenuhi, mulai dari inisialisasi sampel data *training*, menentukan ukuran langkah waktu Δt, Menentukan jumlah tahapan q, Menentukan arsitektur *neural network*, Inisialisasi parameter *neural network*, Inisialisasi lapisan output dengan $q + 1$ neuron, dan Menentukan *hyperparameter*, Pelatihan data dilakukan. Pertama-tama yaitu melakukan komputasi *feed-forward* pada *neural network.*

$$[\hat{u}_1^n(x^i), \dots, \hat{u}_q^n(x^i), \hat{u}_{q+1}^n(x^i)] = u_{NN}^n(x^i; \Theta)\}_{i=1}^{N^n} \tag{3.10}$$

Seperti yang sudah dijelaskan, *feed-forward* akan menghasilkan output $\hat{y}$ Tahapan selanjutnya dapat dihitung nilai *cost*. Karena dalam model waktu diskrit tidak

dibutuhkan *collocation points* untuk melatih PINN. Selain itu, pengenalan penegakan batas yang kuat meniadakan istilah batas MSE_b sebagai bagian dari MSE_u. Sebagai hasilnya, fungsi *cost* pada model waktu diskrit yaitu

$$C = MSE_n \tag{3.11}$$

dimana

$$MSE_n = \sum_{j=1}^{q+1} \sum_{i=1}^{N_n} \left(\hat{u}_j^n(x^i) - u^{n,i}\right)^2 \tag{3.12}$$

Istilah MSE_n menghitung kesalahan prediksi dari PINN pada N_n titik sampel acak $\{x^{n+1}, u^{n,1}\}_{i=1}^{Nn}$ dari solusi pada waktu awal t^n. Ini bisa menjadi kondisi awal dari masalah yang sedang dihadapi atau cuplikan lain dari solusi. kombinasi *optimizer* Adam dan metode L-BFGS digunakan untuk meminimalkan fungsi *cost* pada (3.11).

Setelah mendapatkan nilai *cost*, apabila nilai yang didapatkan belum konvergen ke nilai minimum, maka harus dilakukan pembaruan parameter bobot dan bias pada *neural network*. Nilai *cost* dihitung untuk mengevaluasi perbedaan antara output yang diprediksi dan hasil yang sebenarnya. Nilai *cost* kemudian digunakan untuk memperbarui parameter model melalui *backpropagation* (**Gambar 2.3**) dan algoritma optimasi Adam atau L-BFGS. Tujuannya adalah untuk meminimalkan Nilai *cost* dan meningkatkan akurasi model.

Secara umum pada proses *Training data* ini terjadi proses *feed-forward*, *backpropagation* dan optimasi. Seperti yang telah dijelaskan, pada *feed-forward* terjadi proses mengolah data *input* dan memperoleh hasil akhir atau *output* melalui *neural network*. Proses ini melibatkan perhitungan dari lapisan *input* menuju lapisan *outpu*t. Sedangkan *backpropagation* adalah proses meng*update* bobot dan bias dalam *neural network* dengan mengukur selisih antara hasil aktual dan hasil yang diharapkan pada *output*. Proses ini memanfaatkan *gradient descent* untuk mengurangi *error* dalam *neural network*. Kedua proses ini berlangsung secara bergantian dalam pelatihan *neural network*, hingga *error* menjadi minimal dan *neural network* dapat memberikan hasil yang diharapkan.

Fungsi solusi eksak pada penelitian ini ditentukan, yaitu:

$$u(x,t) = \exp\ (-\frac{(x-p)^2}{2\sigma^2}) \tag{3.13}$$

Dimana nilai $\sigma = 0{,}02$, $x \in [0,1]$, dan p didefinisikan pada persamaan (3.14)

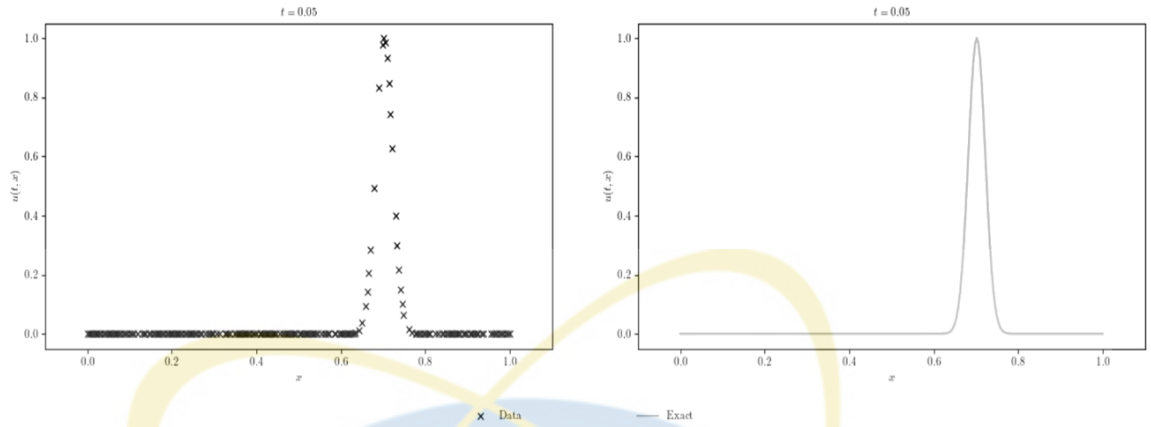
$$p(t) = \frac{1}{4}\cos\ (\frac{2\pi t}{t_{max}}) + \frac{1}{2} \tag{3.14}$$

Setelah melatih model *neural network* dengan data pelatihan, tes data digunakan untuk mengevaluasi kinerja model yang telah dilatih. Data yang digunakan pada tes data ditentukan, yaitu:

Tabel 3. 4 Data pada data tes

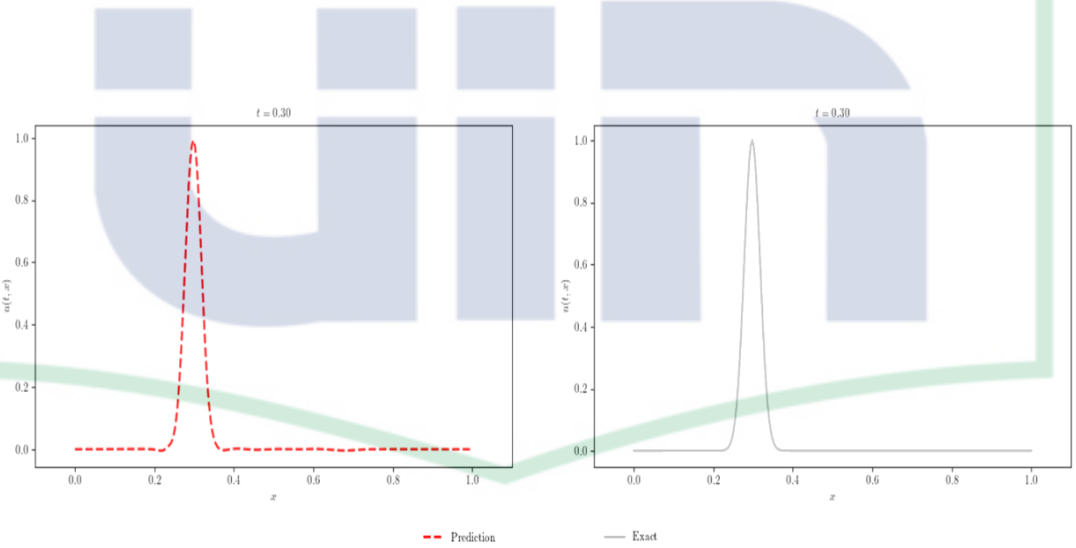
$x*$
0.0000
0.0039
0.0078
…
0.9922
0.9961
1.0000

Hal ini dilakukan untuk membuat prediksi yang akurat pada data yang baru dan data yang belum pernah dilihat sebelumnya. Selama evaluasi, model dijalankan pada set data uji dan prediksinya dibandingkan dengan solusi eksak.



Gambar 3. 3 Hasil $\hat{u}(x,t)$ **dan** $u(x,t)$ **pada** $t = 0,05$

Pada **gambar 3.3** diatas, dihasilkan 2 plot, yaitu plot data pada $t = 0,05$ dan juga didapatkan plot eksaknya. Plot tersebut merupakan $\hat{u}(x,t)$ yang diperoleh dari pemodelan PINN . Jika dilihat dibandingkan kedua plot tersebut, pola pada data dan eksak sama.

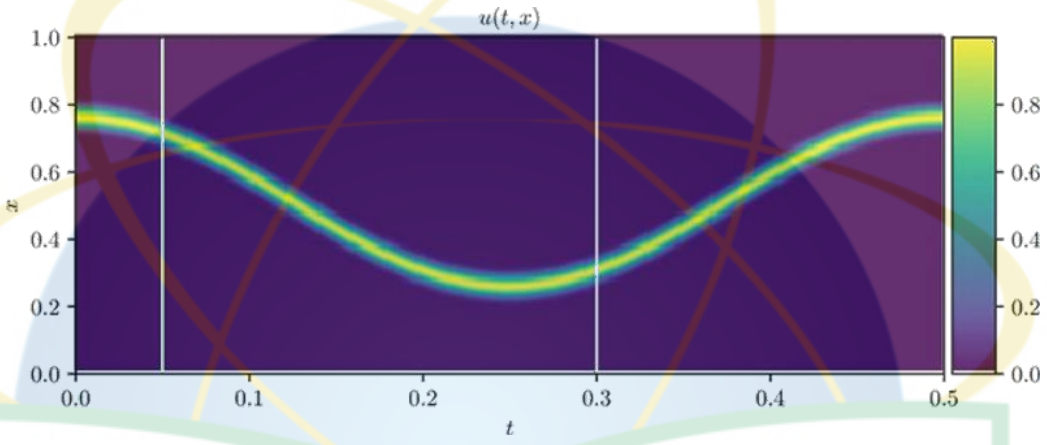


Gambar 3. 4 Plot prediksi dan eksak pada $t = 0,3$

Juga didapatkan prediksi pada $t = 0{,}3$ dibandingkan dengan eksak $t = 0{,}3$ pada tersebut polanya menyerupai eksak.

BAB IV
EKSPERIMEN NUMERIK

Setelah melakukan seluruh proses pada alur penelitian, dengan melakukan komputasi, maka akan didapatkan sebuah plot $u(t, x)$

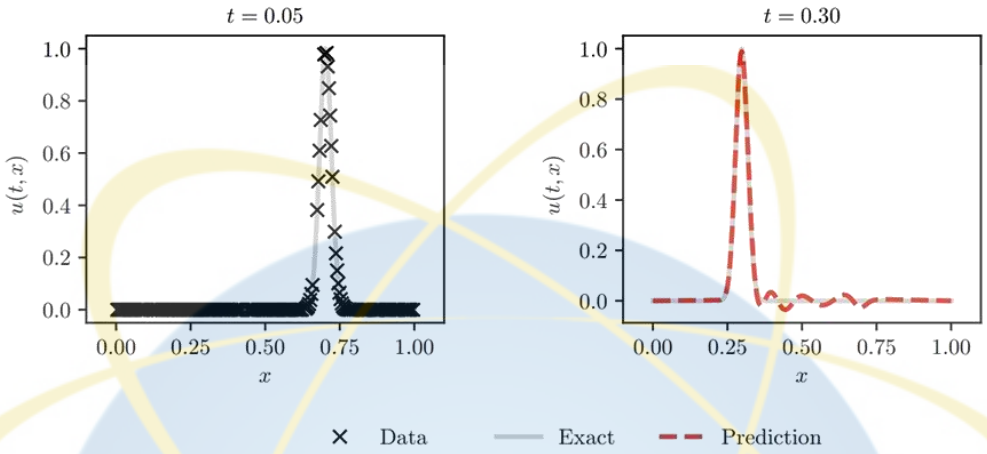


Gambar 4. 1 Plot $\boldsymbol{u(t, x)}$

Pada gambar diatas, nilai dari $u(t,\ x)$ dapat dipresentasikan melalui indikator warna yang ada di sebelah kanan plot. Semakin besar nilai dari $u(t,\ x)$ maka warna akan semakin terang dengan warna kuning. Pada plot ini, titik *x* berada pada interval 0 hingga 1. Dicontohkan, Jika kita melihat pada $x \approx 0{,}3$ dan $t = 0{,}2$ sehingga didapatkan nilai dari $u(t, x)$ berwarna kekuningan yaitu mendekati 1. Jika kita melihat pada $x = 0{,}1$ dan $t = 0{,}4$ maka didapatkan nilai dari $u(t,\ x)$ berwarna ungu gelap yaitu bernilai 0.

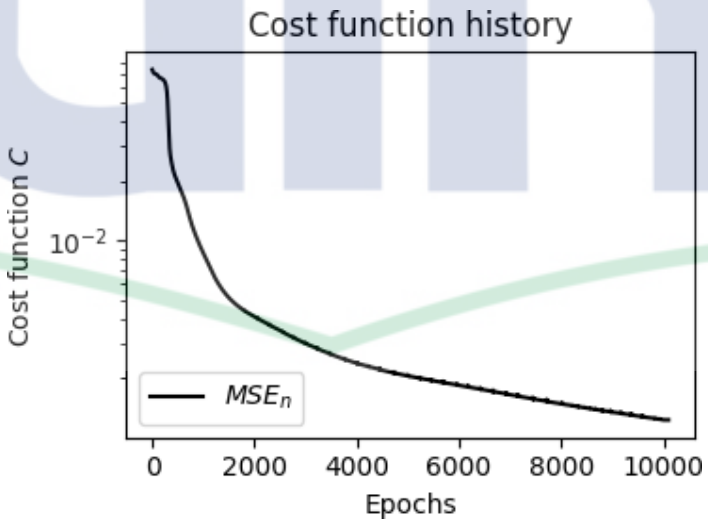
Selanjutnya juga didapatkan plot $u(x,\ t)$. Pada plot disebelah kiri pada **Gambar 4.2,** Didapatkan plot data *training* dari $Nn = 200$ titik data acak pada pada $t^n = 0{,}05$. Data training ini polanya menyerupai eksak. *Neural network* yang telah dilatih pada kumpulan data pelatihan dapat membuat prediksi yang akurat untuk sebagian besar contoh dalam kumpulan pelatihan. Pada plot disebelah kanan, didapatkan model prediksi pada $t^{n+1} = 0{,}3$. yang mana model prediksi tersebut polanya menyerupai eksak, sehingga penggunaan model tersebut dapat

membuat prediksi yang akurat bahkan ketika dihadapkan pada data input yang baru dan berbeda.



Gambar 4. 2 Plot $\boldsymbol{u(x,t)}$

Selain itu, plot *cost function history* juga didapatkan. Seperti **Gambar 4.3**, menunjukan bahwa pada *epoch* 0, nilai dari nilai *cost* masih cukup besar. Namun selama *epoch* dijalankan, nilai *cost* itu sendiri makin mengecil. Banyak sedikitnya jumlah *epoch* dapat mempengaruhi suatu pelatihan *neural network*. Sehingga pada *epoch* ke 10100, didapatkan nilai *cost* sebesar 1.541920e-02.



Gambar 4. 3 Riwayat *Cost Function*

BAB V
PENUTUP

5.1 Kesimpulan

Berawal dari menentukan persamaan panas, fungsi solusi eksak, hingga melakukan seluruh proses training data menggunakan model *physics-informed neural network* dalam waktu diskrit, dan melalukan tes data, diperoleh sebuah plot $u(t,x)$ yang nilainya dapat dipresentasikan melalui indikator warna.

Didapatkan pula gambar $u(x,\ t)$, yang mana terdapat training dari $Nn = 200$ titik data acak pada pada $t^n = 0{,}05$ dimana data *training* ini polanya menyerupai eksak. Pada gambar **Gambar 4.2** disebelah kanan, didapatkan model prediksi pada $t^{n+1} = 0{,}3$. yang mana model prediksi tersebut polanya menyerupai eksak, sehingga penggunaan model tersebut dapat membuat prediksi yang akurat bahkan ketika dihadapkan pada data input yang baru dan berbeda.

Selanjutnya dapat disimpulkan bahwa setelah melakukan *training* sebanyak 10100 *epoch* yang terdiri dari optimasi Adam sebanyak 10000 *epoch* dan L-BFGS sebanyak 100 *epoch* dalam jangka waktu 2467.2421 s, menghasilkan nilai *cost* dari plot *cost function history* yang dapat dilihat pada **Gambar 4.3** bahwa nilai *cost* terus mengecil disaat *epoch* bertambah dan berhenti pada nilai sebesar 1.541920e-02. Sehingga model PINN waktu disrit untuk persamaan diferensial panas dapat dikatakan akurat karena solusi numerik mendekati solusi eksak.

5.2 Saran

Dari hasil penelitian, penulis memberikan saran yang diharapkan berguna untuk penelitian selanjutnya dengan menambahkan jumlah *epoch* yang lebih besar dengan melihat nilai *cost*nya. Selanjutnya penulis menyarankan untuk menggunakan persamaan diferesial lainnya pada model *physics-informed neural network* dalam waktu diskrit.

REFERENSI

[1] C. C. Aggarwal, *Neural Networks and Deep Learning*, 1st ed. Cham: Springer International Publishing, 2018.

[2] S. L. Ross, Differential Equations. 1964.

[3] Maziar Raissi , Paris Perdikaris , and George Em Karniadakis, Physics Informed Deep Learning (Part II): Data-driven Solutions of Nonlinear Partial Differential Equations, 2017

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning (Adaptive Computation and Machine Learning series)*. Massachusetts London: MIT press, 2016.

[5] M. Raissi , P. Perdikaris , and G.E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations.* 2019

[6] Ross, *L.Differential Equations. 3rd*. New York. Springer. 1984

[7] Dong C. Liu and Jorge Nocedal,*On the limited memory BFGS method for large scale optimization*,1989

[8] Stefan, dkk. *Deep Learning in Computational Mechanics*. 2021

[9] Iserles,Arieh, *A First Course in the Numerical Analysis of Differential Equations Second edition,* 2009

[10] Maziar Raissi , Paris Perdikaris , and George Em Karniadakis, *Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations,* 2017

[11] Hagni Wijayanti, dkk. (2011). Metode Runge-Kutta Dalam Penyelesaian Model Radang Akut. Bogor: UNPAK

[12] Isaac Elias Lagaris, Aristidis Likas, Member, IEEE, and Dimitrios I Fotiadis, *Artificial Neural Networks for Solving Ordinary and Partial Differential Equations.* 1998

LAMPIRAN

Lampiran 1. Data (t) interval waktu [0, 0.5]

[0. , 0.0025, 0.005 , 0.0075, 0.01 , 0.0125, 0.015 , 0.0175,
0.02 , 0.0225, 0.025 , 0.0275, 0.03 , 0.0325, 0.035 , 0.0375,
0.04 , 0.0425, 0.045 , 0.0475, 0.05 , 0.0525, 0.055 , 0.0575,
0.06 , 0.0625, 0.065 , 0.0675, 0.07 , 0.0725, 0.075 , 0.0775,
0.08 , 0.0825, 0.085 , 0.0875, 0.09 , 0.0925, 0.095 , 0.0975,
0.1 , 0.1025, 0.105 , 0.1075, 0.11 , 0.1125, 0.115 , 0.1175,
0.12 , 0.1225, 0.125 , 0.1275, 0.13 , 0.1325, 0.135 , 0.1375,
0.14 , 0.1425, 0.145 , 0.1475, 0.15 , 0.1525, 0.155 , 0.1575,
0.16 , 0.1625, 0.165 , 0.1675, 0.17 , 0.1725, 0.175 , 0.1775,
0.18 , 0.1825, 0.185 , 0.1875, 0.19 , 0.1925, 0.195 , 0.1975,
0.2 , 0.2025, 0.205 , 0.2075, 0.21 , 0.2125, 0.215 , 0.2175,
0.22 , 0.2225, 0.225 , 0.2275, 0.23 , 0.2325, 0.235 , 0.2375,
0.24 , 0.2425, 0.245 , 0.2475, 0.25 , 0.2525, 0.255 , 0.2575,
0.26 , 0.2625, 0.265 , 0.2675, 0.27 , 0.2725, 0.275 , 0.2775,
0.28 , 0.2825, 0.285 , 0.2875, 0.29 , 0.2925, 0.295 , 0.2975,
0.3 , 0.3025, 0.305 , 0.3075, 0.31 , 0.3125, 0.315 , 0.3175,
0.32 , 0.3225, 0.325 , 0.3275, 0.33 , 0.3325, 0.335 , 0.3375,
0.34 , 0.3425, 0.345 , 0.3475, 0.35 , 0.3525, 0.355 , 0.3575,
0.36 , 0.3625, 0.365 , 0.3675, 0.37 , 0.3725, 0.375 , 0.3775,
0.38 , 0.3825, 0.385 , 0.3875, 0.39 , 0.3925, 0.395 , 0.3975,
0.4 , 0.4025, 0.405 , 0.4075, 0.41 , 0.4125, 0.415 , 0.4175,

0.42 , 0.4225, 0.425 , 0.4275, 0.43 , 0.4325, 0.435 , 0.4375,
0.44 , 0.4425, 0.445 , 0.4475, 0.45 , 0.4525, 0.455 , 0.4575,
0.46 , 0.4625, 0.465 , 0.4675, 0.47 , 0.4725, 0.475 , 0.4775,
0.48 , 0.4825, 0.485 , 0.4875, 0.49 , 0.4925, 0.495 , 0.4975,
0.5]]),

Lampiran 2. Data (x) *domain space* [0, 1]

[0. , 0.00392157, 0.00784314, 0.01176471, 0.01568627,
0.01960784, 0.02352941, 0.02745098, 0.03137255, 0.03529412,
0.03921569, 0.04313725, 0.04705882, 0.05098039, 0.05490196,
0.05882353, 0.0627451 , 0.06666667, 0.07058824, 0.0745098 ,
0.07843137, 0.08235294, 0.08627451, 0.09019608, 0.09411765,
0.09803922, 0.10196078, 0.10588235, 0.10980392, 0.11372549,
0.11764706, 0.12156863, 0.1254902 , 0.12941176, 0.13333333,
0.1372549 , 0.14117647, 0.14509804, 0.14901961, 0.15294118,
0.15686275, 0.16078431, 0.16470588, 0.16862745, 0.17254902,
0.17647059, 0.18039216, 0.18431373, 0.18823529, 0.19215686,
0.19607843, 0.2 , 0.20392157, 0.20784314, 0.21176471,
0.21568627, 0.21960784, 0.22352941, 0.22745098, 0.23137255,
0.23529412, 0.23921569, 0.24313725, 0.24705882, 0.25098039,
0.25490196, 0.25882353, 0.2627451 , 0.26666667, 0.27058824,
0.2745098 , 0.27843137, 0.28235294, 0.28627451, 0.29019608,
0.29411765, 0.29803922, 0.30196078, 0.30588235, 0.30980392,
0.31372549, 0.31764706, 0.32156863, 0.3254902 , 0.32941176,
0.33333333, 0.3372549 , 0.34117647, 0.34509804, 0.34901961,

0.35294118, 0.35686275, 0.36078431, 0.36470588, 0.36862745, 0.37254902, 0.37647059, 0.38039216, 0.38431373, 0.38823529, 0.39215686, 0.39607843, 0.4 , 0.40392157, 0.40784314, 0.41176471, 0.41568627, 0.41960784, 0.42352941, 0.42745098, 0.43137255, 0.43529412, 0.43921569, 0.44313725, 0.44705882, 0.45098039, 0.45490196, 0.45882353, 0.4627451 , 0.46666667, 0.47058824, 0.4745098 , 0.47843137, 0.48235294, 0.48627451, 0.49019608, 0.49411765, 0.49803922, 0.50196078, 0.50588235, 0.50980392, 0.51372549, 0.51764706, 0.52156863, 0.5254902 , 0.52941176, 0.53333333, 0.5372549 , 0.54117647, 0.54509804, 0.54901961, 0.55294118, 0.55686275, 0.56078431, 0.56470588, 0.56862745, 0.57254902, 0.57647059, 0.58039216, 0.58431373, 0.58823529, 0.59215686, 0.59607843, 0.6 , 0.60392157, 0.60784314, 0.61176471, 0.61568627, 0.61960784, 0.62352941, 0.62745098, 0.63137255, 0.63529412, 0.63921569, 0.64313725, 0.64705882, 0.65098039, 0.65490196, 0.65882353, 0.6627451 , 0.66666667, 0.67058824, 0.6745098 , 0.67843137, 0.68235294, 0.68627451, 0.69019608, 0.69411765, 0.69803922, 0.70196078, 0.70588235, 0.70980392, 0.71372549, 0.71764706, 0.72156863, 0.7254902 , 0.72941176, 0.73333333, 0.7372549 , 0.74117647, 0.74509804, 0.74901961, 0.75294118, 0.75686275, 0.76078431, 0.76470588, 0.76862745, 0.77254902, 0.77647059, 0.78039216, 0.78431373, 0.78823529, 0.79215686, 0.79607843, 0.8 , 0.80392157, 0.80784314, 0.81176471, 0.81568627, 0.81960784, 0.82352941, 0.82745098, 0.83137255, 0.83529412, 0.83921569,

0.84313725, 0.84705882, 0.85098039, 0.85490196, 0.85882353,
0.8627451 , 0.86666667, 0.87058824, 0.8745098 , 0.87843137,
0.88235294, 0.88627451, 0.89019608, 0.89411765, 0.89803922,
0.90196078, 0.90588235, 0.90980392, 0.91372549, 0.91764706,
0.92156863, 0.9254902 , 0.92941176, 0.93333333, 0.9372549 ,
0.94117647, 0.94509804, 0.94901961, 0.95294118, 0.95686275,
0.96078431, 0.96470588, 0.96862745, 0.97254902, 0.97647059,
0.98039216, 0.98431373, 0.98823529, 0.99215686, 0.99607843,
1.]]