8. ULUSAL HAVACILIK VE UZAY KONFERANSI **UHUK-2022-054**
14-16 Eylül 2022, İzmir Ekonomi Üniversitesi, İzmir

SIKIŞTIRILABİLİR EULER DENKLEMLERİNİN FİZİKLE ÖĞRENEN YAPAY SİNİR AĞLARI İLE ÇÖZÜMÜ

Atakan Aygun* ve Ali Karakus†
Orta Doğu Teknk Üniversitesi,
Ankara

ÖZET

Bu çalışmanın amacı fizikle öğrenen yapay sinir ağları ile sıkıştırılabilir Euler denklemlerinin çözümlerinin elde edilebildiğini ve süreksizliklerin yakalanabildiğini göstermektir. Öncül bir çalışma olarak bir boyutlu zamana bağlı bir problemde oluşan şokun ilerleyişi tahmin edilmiştir. Hesaplamalar TensorFlow paketi ve içerdiği otomatik türev alabilme özelliğiyle yapılmıştır. Sinir ağları akışın hızı, basıncı ve yoğunluğunu tahmin etmekte kullanılmıştır ve yoğunlukta oluşan süreksizliği doğru bir şekilde yakalayabilmiştir.

GİRİŞ

Kısmi diferansiyel denklemlerinin çözümleri mühendislik çalışmalarında önemli bir yer oluşturmaktadır. Son dönemlerde, bu denklem sistemlerinin çözümünde makine öğrenmesi yöntemleri, geleneksel nümerik çözümlerin yanı sıra kullanılmaya baslamıştır [Raissi vd., 2017a,b]. Raissi ve ark. tarafından [Raissi vd., 2019] fizikle öğrenen yapay sinir ağlarının tanıtılmasıyla, ileri (forward) ve ters (inverse) problemlerin çözümüne farklı bir yaklaşım gelişmiştir. Bu sinir ağları, herhangi bir geometrik ağ yapısına ihtiyaç duymadan, otomatik türevlenme [Baydin vd., 2017] kullanarak, diferansiyel denklemlerin artıklarını hata fonksiyonuna ekleyerek minimize etmeye çalışmakta olup, yaygın bir şekilde akış ve ısı transferi uygulamalarında kullanılmaya başlanmış [Rao, Sun ve Liu , 2020; Jin vd., 2021; Cai vd., 2022, 2021] ve bu yöntem kullanılarak bazı yazılımlar geliştirilmiştir [Hennigh vd., 2021; Lu vd., 2021; Zubov vd., 2021; Koryagin vd., 2019; Chen vd., 2020].

Bu çalışmada fizikle öğrenen yapay sinir ağları sıkıştırılabilir Euler denklemlerinin çözümü için uygulanmıştır. Öncül bir çalışma olarak bir boyutlu ve zamana bağlı bir şokun ilerleyişi incelenmiştir. Çözüm için oluşturulan hesaplama alanının ve yapay sinir ağlarının detayları sunulmuş, elde edilen sonuçlar gerçek çözümlerle kıyaslanmıştır.

*Araştırma Görevlisi, Makina Müh. Böl., E-posta: atakana@metu.edu.tr
†Doktor Öğretim Üyesi, Makina Müh. Böl., E-posta: akarakus@metu.edu.tr

YÖNTEM

Fizikle Öğrenen Yapay Sinir Ağları

Aşağıdaki gösterilen genel bir kısmi diferansiyel denklemi düşünelim.

$$\mathbf{u}_t + \mathcal{N}[u] = 0, \qquad \mathbf{x} \in \Omega,\ t \in [0,T] \tag{1a}$$

$$\mathbf{u}(\mathbf{x},0) = f(\mathbf{x}), \qquad \mathbf{x} \in \Omega \tag{1b}$$

$$\mathbf{u}(\mathbf{x},t) = g(\mathbf{x},t), \qquad \mathbf{x} \in \partial\Omega,\ t \in [0,T] \tag{1c}$$

$\mathcal{N}$ doğrusal veya doğrusal olmayan bir genelleştirilmiş diferansiyel operatör, $x \in \mathbb{R}^d$ ve t uzaysal ve zamansal koordinatlardır. Ω ve $\partial\Omega$ sırasıyla hesaplama alanını ve sınırları temsil eder. $\mathbf{u}(\mathbf{x},t)$ diferansiyel denklem çözümünü, $f(\mathbf{x})$ başlangıç koşulunu, $g(\mathbf{x},t)$ ise sınır şartlarını temsil eder.

Kısmi diferansiyel denklemin çözümü tam bağlantılı bir sinir ağı ile tahmin edilebilir. Fizikle öğrenen yapay sinir ağları [Raissi vd., 2019], uzaysal ve zamansal koordinatları girdi olarak aldıktan sonra saklı katmanlardan geçirerek bir çözüm ortaya çıkarır. Bu saklı katmanlar bir önceki katmandaki değişkenleri girdi olarak alıp doğrusal olmayan bir aktivasyon fonksiyonu ile bir sonraki saklı katmanın girdisi için bir çıktı üretir. Bu aktivasyon fonksiyonu $\sigma(\cdot)$:

$$y_j = \sigma(\omega_{i,j} x_j + b_j) \tag{2}$$

$\omega_{i,j}$ ve b_j sırasıyla ağırlık ve bias değerlerini göstermektedir ve eğitilebilir parametrelerdir. Bu parametreler kompozit bir hata fonksiyonunu minimuma indirgemek için değiştirilir. Bu hata fonksiyonu,

$$\mathcal{L} = \omega_D \mathcal{L}_D + \omega_R \mathcal{L}_R + \omega_{BC} \mathcal{L}_{BC} + \omega_{IC} \mathcal{L}_{IC}, \tag{3}$$

şeklinde yazılabilir. Her bir hata terimi genelde hataların karelerinin ortalaması alınarak yazılabilir:

$$\mathcal{L}_D = \frac{1}{N_D} \sum_{i=1}^{N_D} |\mathbf{u}(x^i,t^i) - \mathbf{u}^i|^2 \tag{4a}$$

$$\mathcal{L}_R = \frac{1}{N_R} \sum_{i=1}^{N_R} |\mathbf{u}_t + \mathcal{N}[\mathbf{u}(\mathbf{x}^i,t^i)]|^2 \tag{4b}$$

$$\mathcal{L}_{BC} = \frac{1}{N_{BC}} \sum_{i=1}^{N_{BC}} |\mathbf{u}(\mathbf{x}^i,t^i) - g(\mathbf{x}^i,t^i)|^2 \tag{4c}$$

$$\mathcal{L}_{IC} = \frac{1}{N_{IC}} \sum_{i=1}^{I_{BC}} |\mathbf{u}(\mathbf{x}^i,0) - f(\mathbf{x}^i)|^2. \tag{4d}$$

Bu hata terimindeki yüksek doğruluktaki veriden sapma hatası ($\mathcal{L}_D$), sınır şartları hatası ($\mathcal{L}_{BC}$) ve başlangıç koşulu hatasının ($\mathcal{L}_{IC}$) yanı sıra kısmi diferansiyel denklemin artık değerinin de ($\mathcal{L}_R$) sıfıra yaklaşması istenir. Bunun için gerekli türev alma ise otomatik türevlendirmeyle yapılır [Baydin vd., 2017]. TensorFlow [Abadi vd., 2016] ve PyTorch [Paszke vd., 2019] gibi derin öğrenme paketlerinde bu özellik hazır olarak bulunmaktadır.N_D, N_R, N_{BC} ve N_{IC} farklı kayıp terimleri için kullanılan nokta sayısını, ω_D, ω_R, ω_{BC} ve ω_{IC} ise bu hata terimlerinin toplam hata fonksiyonlarındaki ağırlıklarını belirtmektedir.

Euler Denklemleri

Sıkıştırılabilir ve viskoz olmayan akışta kütle, momentum ve enerji korunumu Euler denklemleri ile modellenebilir. Bu denklemler şu şekilde yazılabilir:

$$\partial_t U + \nabla \cdot f(U) = 0,\ x \in \Omega \subset \mathbb{R}, \quad t \in [0,T] \tag{5}$$

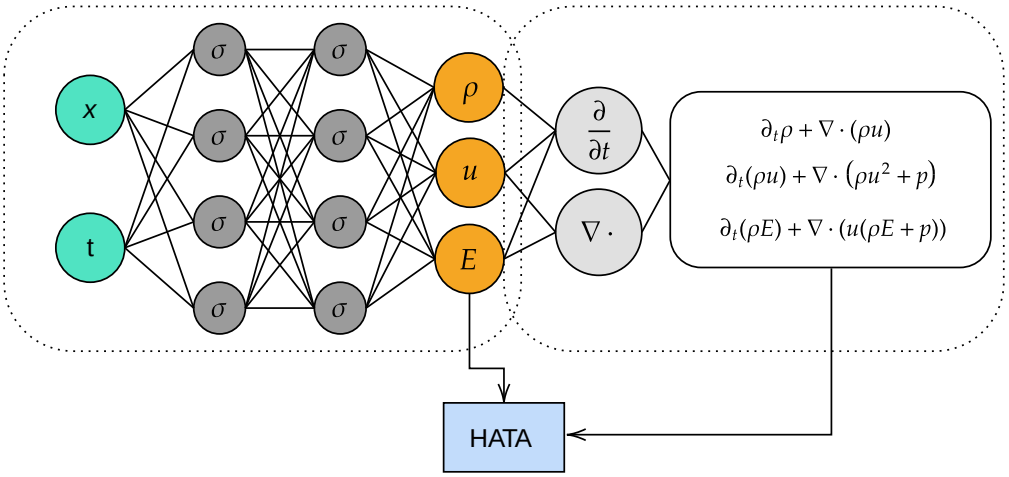
Buradaki terimler ise

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho E \end{pmatrix}, \quad f(U) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(\rho E + p) \end{pmatrix}$$

şeklinde gösterilir. ρ yoğunluğu, u bir boyuttaki hızı, p basıncı ve E toplam enerjiyi temsil etmektedir. Bunlar haricinde bir de hal denklemine ihtiyaç duyulmaktadır. Bu denklem ise

$$p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho u^2\right)$$

şeklinde gösterilir. Hava için $\gamma = 1.4$ olarak alınabilir.



Şekil 1: Fizikle Öğrenen Yapay Sinir Ağlarının Şematiği

Şekil 1'de görüldüğü gibi fizikle öğrenen yapay sinir ağları uzaysal ve zamansal koordinatları girdi olarak aldıktan sonra saklı katmanlardan geçirerek sıkıştırılabilir Euler denklemleri için gerekli olan yoğunluk (ρ), hız (u) ve enerji (E) terimlerini tahmin eder. Bu değişkenleri sınır şartlarında ve başlangıç koşulunda yerlerine koyarak oradaki kayıpları hesaplamasının ardından türevlerini otomatik türevlenebilme ile alarak sıkıştırılabilir Euler denklemlerini sağlaması için yerlerine koyar. Toplam hata fonksiyonu bu hataların toplamından oluşur ve bu hata gerekli optimizasyon algoritmaları ile sıfıra yaklaştırılmaya çalışılır. Yeterli bir toleransa ulaşıldığında sonuç elde edilir.

UYGULAMALAR VE DEĞERLENDİRME

Bu çalışmada bir boyutlu zamana bağlı Euler denklemleri fizikle öğrenen yapay sinir ağları ile çözülmüştür. Hesaplama alanı $[0, 1]$ arasındadır ve başlangıç şartı olarak $x = 0.5$ noktasında bir şok yer almaktadır. Şokun sol ve sağ tarafındaki başlangıç değerleri ise

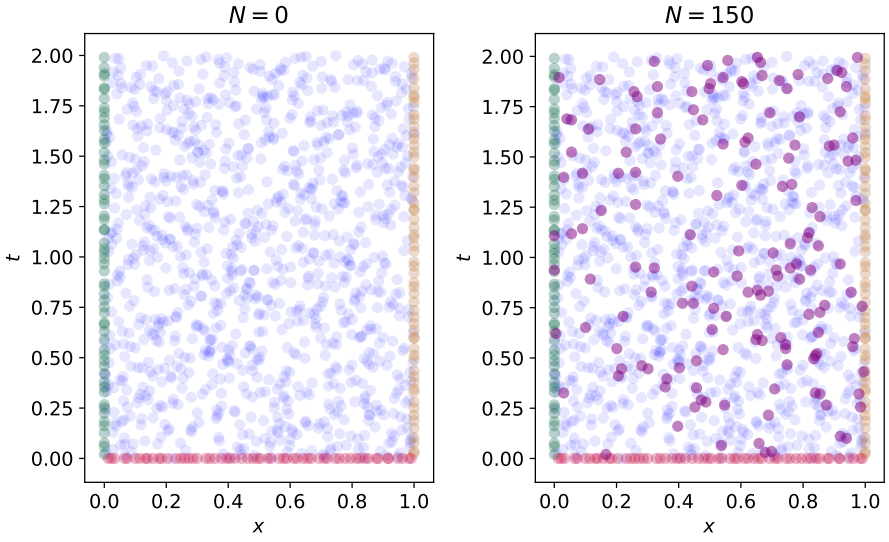
$$(\rho_L, u_L, p_L) = (1.4, 0.1, 1.0), \quad (\rho_R, u_R, p_R) = (1.0, 0.1, 1.0)$$

olarak belirtilmiştir. Gerçek çözümdeki değerler Dirichlet sınır şartı olarak kullanılmıştır. Gerçek çözüm

$$\rho(x,t) = \begin{cases} 1.4 & x < 0.5 + 0.1t, \\ 1.0 & x > 0.5 + 0.1t, \end{cases} \quad u(x,t) = 0.1, \quad p(x,t) = 1.0$$

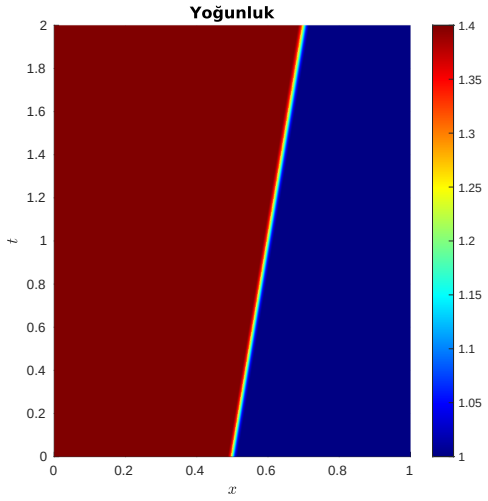
şeklinde verilmiştir.

Yapay sinir ağlarının eğitim aşaması için sınırlarda 60'ar sınır noktası, 60 başlangıç noktası ve 1000 hesaplama alanı içinde yer alan nokta, latin hiperküp örneklemesi ile seçilmiştir. Ayrıca farklı bir hesaplamada yapay sinir ağına gerçek çözümlerden 150 değer de eklenmiştir. Bu seçim Şekil 2'de görülmektedir ve mor renkle belirtilen noktalar gerçek değerlerin rastgele bir şekilde seçilmesiyle elde edilmiştir. Ağ yapısında 4 saklı katman ve her saklı katmanda 40 sinir hücresi yer almaktadır. Optimal parametrelerin hesaplanması için Adam [Kingma ve Ba, 2017] yönteminin 0.001 öğrenme oranı ve 20000 yinelemeyle kullanılmasının ardından BFGS (Broyden–Fletcher–Goldfarb–Shanno) yöntemi kullanılmıştır. Aktivasyon fonksiyonu olarak tanh fonksiyonu kullanılmıştır. Çözücü Python üzerinde, TensorFlow paketi kullanılarak geliştirilmiştir.



Şekil 2: Hesaplama alanının oluşturulması

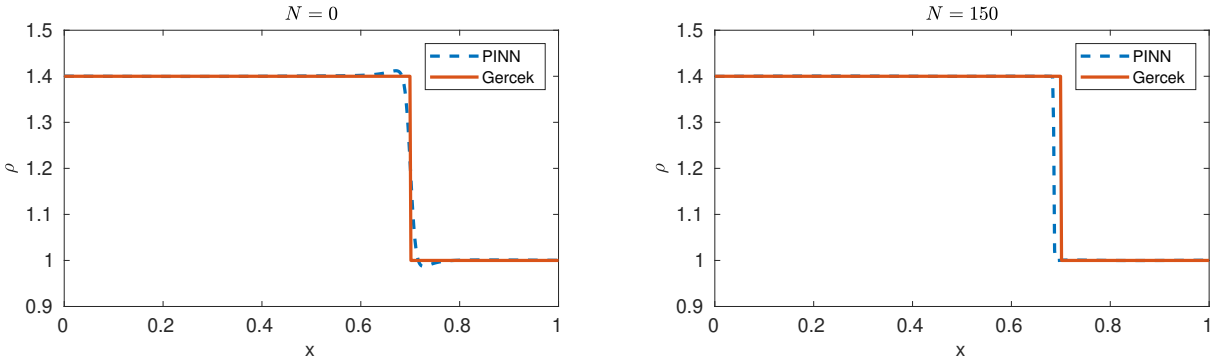
Eğitim aşamasından sonra 500×500 noktalık bir alanda tahmin gerçekleştirilmiştir. Şekil 3'te görüldüğü gibi şokun ilerleyişi ve yeri doğru bir şekilde tahmin edilebilmiştir.



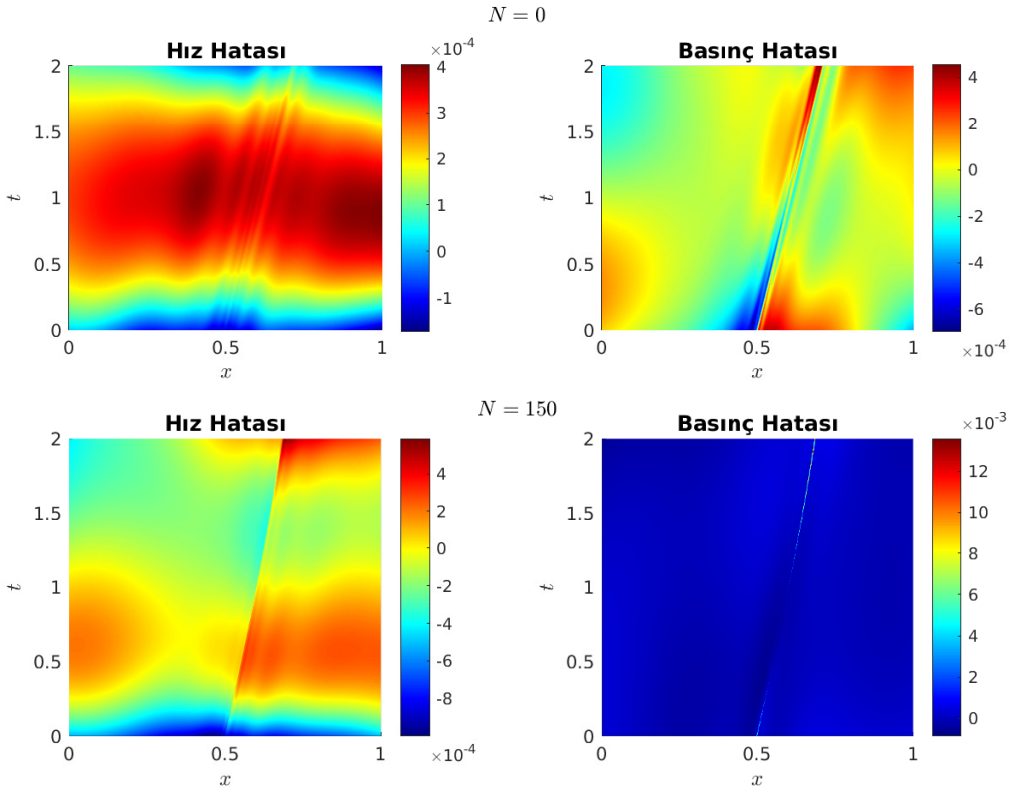
Şekil 3: 1 Boyutlu şokun ilerleyişinin yapay sinir ağıyla tahmini

Eğitim aşamasının ardından yapılan tahminlerde yoğunluk, hız ve basınç değerleri elde edilmiştir. Şekil 4 yapılan tahminlerin $t = 2.0$s zamanında gerçek çözümlerle olan karşılaştırılmasını

göstermektedir. Sağ tarafta yer alan şekil 150 gerçek çözümle birlikte eğitilen sinir ağıyla yapılan tahmini göstermektedir. Yoğunluk fonksiyonunda kontak süreksizliği yer almasından dolayı yapay sinir ağları bu fonksiyonu tahmin etmekte hız ve basınç fonksiyonuna kıyasla daha az başarılı olabilmektedir. Yapay sinir ağlarına gerçek çözümün eklenmesi ise bu başarı oranını artırmaktadır. Şekil 5'te görüldüğü gibi hız ve basınç değerlerindeki hatalar çok daha küçüktür. Şeklin alt kısmında görüldüğü gibi gerçek verinin eklenmesi ise bu hataları iyice azaltmaktadır.



Şekil 4: Yoğunluk, hız ve basınç değerlerinin tahminlerinin $t = 2.0$s zamanında gerçek çözümlerle kıyaslanması.



Şekil 5: PINN çözümlerinin gerçek çözümlerle olan farkları. Üst sıradaki şekiller herhangi bir veri olmadan alınan çözümlerin hatası, alt sıradaki şekiller ise 150 verinin verilmesiyle elde edilen sonucun hatasıdır.

SONUÇ

Bu çalışmada fizikle öğrenen yapay sinir ağlarının sıkıştılabilir Euler denklemlerinin çözümünde nasıl kullanılabileceği anlatılmıştır. Bir boyutlu zamana bağlı bir problemin çözümü bu ağların kullanımı için öncül bir çalışma olarak ele alınmıştır.Oluşan sonuçlarda, yoğunlukta oluşan süreksizliğin yakalanabildiği gözlemlenmiştir. Hız ve basınç fonksiyonları ise sürekli fonksiyonlar olduğundan bunlar için yapılan tahminlerdeki başarı daha yüksektir. Bu tahminin doğruluğunu artırmak için rastgele gerçek çözümlerin seçilmesi ve bu noktalardaki hesaplamanın bu değerlerle kıyaslanması denenmiştir. Yapay sinir ağlarının hata fonksiyonuna eklenen bu fazladan hata terimi tahminlerin doğruluğunu artırmış, kontak süreksizliğinin daha keskin bir şekilde yakalanması sağlanmıştır.

Kaynaklar

Abadi M., Barham P., Chen J., Chen Z., Davis A., Dean J., Devin M., Ghemawat S., Irving G., Isard M., Kudlur M., Levenberg J., Monga R., Moore S., Murray D. G., Steiner B., Tucker P., Vasudevan V., Warden P., Wicke M., Yu Y. ve Zheng X., 2016. *TensorFlow: A system for large-scale machine learning.*, 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), 265–283

Baydin A. G., Pearlmutter B. A., Radul A. A. ve Siskind J. M., 2017. *Automatic differentiation in machine learning: a survey.*, The Journal of Machine Learning Research, 18(1):5595–5637

Cai S., Wang Z., Wang S., Perdikaris P., ve Karniadakis G., 2021. *Physics-Informed Neural Networks (PINNs) for Heat Transfer Problems.*, Journal of Heat Transfer, 143.

Cai S., Mao Z., Wang Z., Yin M. ve Karniadakis G., 2022. *Physics-informed neural networks (PINNs) for fluid mechanics: a review.*, Acta Mechanica Sinica, 1614-3116.

Chen, F. vd. 2020. *NeuroDiffEq: A python package for solving differential equations with neural networks.* J. Open Source Softw. 5, 1931.

Hennigh, O., Narasimhan, S., Nabian, M. A., Subramaniam, A., Tangsali, K., Fang, Z., Rietmann, M., Byeon, W., and Choudhry, S. 2021. *Nvidia SimNet™: An ai-accelerated multi-physics simulation framework*, In International Conference on Computational Science, pages 447–461. Springer.

Jin X., Cai S., Li H. ve Karniadakis G., 2021. *NSFnets (Navier-Stokes Flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations.*, Journal of Computational Physics, 426:109951

Kingma, D. P. ve Ba, J. 2017. *Adam: A method for stochastic optimization.* arXiv:1412.6980 [cs].

Koryagin, A., Khudorozkov, R. and Tsimfer, S. 2019. *PyDEns: a Python framework for solving differential equations with neural networks.* arXiv 1909.11544

Lu, L., Meng, X., Mao, Z., and Karniadakis, G. E. 2021. *DeepXDE: A deep learning library for solving differential equations*, SIAM Review, 63(1):208–228

Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Kopf A., Yang E., DeVito Z., Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J. ve Chintala S., 2019. *PyTorch: An Imperative Style, High-Performance Deep Learning Library.*, Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc.

Raissi M., Perdikaris P., ve Karniadakis G. E., 2017a. *Inferring solutions of differential equations using noisy multi-fidelity data.*, Journal of Computational Physics, 335:736–746.

Raissi M., Perdikaris P., ve Karniadakis G. E., 2017b. *Machine learning of linear differential equations using Gaussian processes.*, Journal of Computational Physics, 348:683–693.

Raissi M., Perdikaris P., ve Karniadakis G. E., 2019. *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations.*, Journal of Computational Physics, 378:686–707.

Rao C., Sun H. ve Liu Y., 2020. *Physics-informed deep learning for incompressible laminar flows.*, Theoretical and Applied Mechanics Letters, 10(3):207–212.

Zubov, K., McCarthy, Z., Ma, Y., Calisto, F., Pagliarino, V., Azeglio, S., Bottero, L., Luján, E., Sulzer, V., Bharambe, A., Vinchhi, N., Balakrishnan, K., Upadhyay, D., and Rackauckas, C. 2021. *NeuralPDE: Automating physics-informed neural networks (PINNs) with error approximations.* arXiv:2107.09443.