GFPINNs : General Fractional Physics-informed Neural Networks

Pavan Pranjivan Mehta, Gianluigi Rozza [*]
SISSA, International School of Advanced Studies,
Via Bonomea, 265, 34136 Trieste TS, Italy
pavan.mehta@sissa.it, grozza@sissa.it

Abstract

General fractional calculus is a further generalisation of fractional calculus which admits an arbitrary kernel [1, 2, 3]. In the context of numerics for such differential equations, we propose, general fractional physics-informed neural networks (GF-PINNs). In our method, we employ, separation of variable technique by introducing Jacobi convolution polynomial [3], whilst taking advantage of neural network's ability as a non-linear solver. Furthermore, we remark automatic differentiation can now be carried out for integer-order operators, thereby mitigating the need of a splitting scheme. The resulting scheme is both accurate and computationally efficient, since we do not have to compute discrete convolution.

1 Introduction

Fractional Calculus is a generalisation of integer order calculus [4, 5, 6], where the order is defined in $\mathbb{R}$. Paradoxically, for non-integer orders these are non-local operators, which are appropriate mathematical objects to address non-local phenomenon. For instance see [7, 8] for its application in turbulence. In [7, 8], a variable order Caputo fractional derivative was used, where the order of the operators was found as an inverse problem. Particularly in [8] a pointwise version of fractional physics-informed neural networks (fPINNS) was introduced (see also [9]). The advantage of pointwise fPINNS for inverse problems was outlined in [8] as,

- Given that the fractional order exists, it does not require boundary conditions.
- If the fractional order is a discontinuous function or the neural network collapses during training process; a pointwise algorithm still converges.

Furthermore, in [8] pointwise fPINNS was extended to tempered fractional operators, where equivalence was found between operators with different tempering functions. Such an equivalence was found by introducing the "horizon of non-local interactions" defined as,

Definition 1.1 *(see [8]) The horizon of non-local interactions is a distance beyond which no long-range interactions occur.*

Although, fractional operators have found real world applications. One overwhelming question remains for physicist and engineers, *can fractional operators with a fixed kernel describe all processes?* The answer, is clearly no (see [10, 11]); thus invoking the need for a mathematical theory of operators with arbitrary kernels.

With regards to generalisation of fractional operators, Sonine [12] recognised a key property that the convolution of the kernel (of the fractional derivative and integral) is unity, thereafter proposed the condition (1) for any pair $(k(x), \kappa(x))$ for analytical solution.

$$\int_0^x k(x-t)\kappa(t)dt = 1,\ x > 0. \tag{1}$$

[*]Corresponding author: grozza@sissa.it

1st Conference on Applied AI and Scientific Machine Learning (CASML 2024).

Indeed, there is a large class of functions which satisfy the condition (1) (for examples, infer [12, 1, 2]).

Perhaps, the first results within the framework of fractional calculus was done in [1] (now known as general fractional calculus). In [1] the conditions over the kernel was in its Lapace transform. However, working with Laplace transform of the kernel is rather cumbersome, thus in [13, 2, 14], another class of kernels were introduced (Luchko class). Note that the the mathematical theory of general fractional calculus is now complete (with Luchko class of kernels), where in [2] the results are on semi-infinite domain and the work on finite interval in [3] for arbitrary orders.

Given the importance of the subject, in [3] an accurate and efficient Petrov-Galerkin scheme was constructed by introducing Jacobi Convolution Polynomials. A notable property of this basis function, the general fractional derivative of Jacobi convolution polynomial is a shifted Jacobi polynomial. Thus, with a suitable test function it results in diagonal stiffness matrix, hence, the efficiency in implementation.

To aid in numerics of general fractional operators, we introduce general fractional physics-informed neural networks (GFPINNs). The structure of this short paper is as follows,

- Section 2: We introduce general fractional calculus on finite interval [3].
- Section 3: We introduce Jacobi convolution polynomials [3].
- Section 4: We introduce general fractional physics-informed neural networks (GFPINNs).

2 General Fractional Calculus

In this section, we will outline the preliminaries for general fractional calculus for arbitary order on finite interval [3] (and for semi-infinite domain, refer [2]). We start by defining the modified Sonine condition, which the kernel's obey.

Definition 2.1 *(see [3, 2]) The pair (k_n, κ_n) satisfy the left modified Sonine condition for $n \in \mathbb{N}$ on an interval $(a, b]$, where $a, b \in \mathbb{R}$, is given by,*

$$\int_a^x k_n(x-t)\kappa_n(t)dt \;=\; \frac{(x-a)^{n-1}}{(n-1)!} \;=: \{1\}_l^n,\; a < x \le b,\; n \in \mathbb{N}. \tag{2}$$

where $\{1\}_l$ is a function uniformly equal to one over the interval and

$$\{1\}_l^n \;:=\; \underbrace{\{1\}_l * \{1\}_l * \cdots * \{1\}_l}_{n-terms}.$$

Note the the kernels belong to the, C_α, function spaces (3), first introduced in [15] and also used in [14] for general fractional calculus.

Definition 2.2 *(see [15]) For $\alpha \ge -1$ and $n \in \mathbb{N}$, the function spaces are defined as,*

$$C_\alpha^n(a,b] \;=\; \Big\{ f : f^{(n)} \in C_\alpha(a,b] \Big\}, \tag{3}$$

where,

$$C_\alpha(a,b] \;=\; \Big\{ f : (a,b] \to \mathbb{R} : f(x) = (x-a)^p f_1, p > \alpha, f_1 \in C[a,b] \Big\},$$

Note that, space C_{-1} is inadequate to exclude all non-singular functions (see also [16]). However, we recognise that there is no need to define a function space explicitly to eliminate non-singular functions, rather satisfying the modified Sonine condition will lead to singular functions [3]. Thus, we will look for kernels belonging to $\mathbb{L}_n$ (Luchko class) as,

$$\mathbb{L}_n(a,b] = \Big\{ k_n, \kappa_n \in C_{-1}^n(a,b] : \int_a^x k_n(x-t)\kappa_n(t)dt = \frac{(x-a)^{n-1}}{(n-1)!}, \\ n \in \mathbb{N}, a < x \le b \in \mathbb{R} \Big\} \tag{4}$$

Now, we define the general fractional integral and derivatives.

Definition 2.3 *(see [3]) If (k_n, κ_n) are a Sonine kernel from $\mathbb{L}_n(a,b]$, then, we define,*

(a) *The left-sided general fractional integral is defined with the kernel, κ_n as,*

$$ {}_a\mathcal{I}_x^{(\kappa_n)} f(x) := \int_a^x \kappa_n(x-s) f(s) ds \tag{5} $$

(b) *The left-sided general fractional Riemann–Liouville derivative is defined with the kernel, k_n as,*

$$ {}_a^{RL}\mathcal{D}_x^{(k_n)} f(x) := \frac{d^n}{dx^n} \int_a^x k_n(x-s) f(s) ds. \tag{6} $$

We state the first fundamental theorem of calculus for left-sided operators.

Theorem 2.1 *(see [3]) (First fundamental theorem of calculus for left-sided operators) If pair $(k_n, \kappa_n) \in \mathbb{L}_n(a,b]$ satisfy the left modified Sonine condition for $n \in \mathbb{N}$, where $a < b \in \mathbb{R}$, then, the left-sided general fractional Riemann–Liouville derivative is defined with the kernel, k_n is a left inverse of left-sided general fractional integral defined with the kernel, κ_n*

$$ {}_a^{RL}\mathcal{D}_x^{(k_n)} {}_a\mathcal{I}_x^{(\kappa_n)} f(x) = f(x) \ , \ a < x \leq b. $$

For the results on general fractional Caputo derivative, second fundamental theorem of calculus, right-sided opertaors and genereal fractional integration by parts, refer [3].

3 Jacobi Convolution Polynomials

For construction of an efficient scheme for fractional operators, Jacobi Poly-fractonomials were introduced in [17], which forms a basis functions (see also [18]). Since, the kernel is arbitrary in our case, we make use of Jacobi convolution polynomial [3] defined as below (7). Note that, since fractional calculus is a special case of general fractional calculus, Jacobi convolution polynomial are applicable to fractional operators as well.

Definition 3.1 *(Left Jacobi convolution polynomials [3]) We define left Jacobi convolution polynomials $(\phi_n(x))$ as,*

$$ \phi_n(x) := \int_0^x \kappa(x-t) \tilde{P}_n^{\alpha,\beta}(t) dt, \ x \in [0,1], \ \alpha, \beta > -1, \forall n \in \mathbb{N} \cup \{0\} \tag{7} $$

where $\kappa \in \mathbb{L}_m(0,1]$ satisfies the left modified Sonine condition and $\tilde{P}_n^{\alpha,\beta}(x)$ is the shifted Jacobi polynomial.

By virtue of the construction, we have [3],

$$ \phi_n(0) = \lim_{x \to 0} \int_0^x \kappa(x-t) \tilde{P}_n^{\alpha,\beta}(t) dt = 0, \ \forall n \in \mathbb{N} \cup \{0\} \tag{8} $$

Note that, the left-sided general fractional derivative of left Jacobi convolution polynomial is a shifted Jacobi polynomial, shown as [3],

$$ {}_0^{RL}D_x^k \phi_n = \frac{d}{dx} k * \kappa * \tilde{P}_n^{\alpha,\beta}(x) = \frac{d}{dx}\{1\} * \tilde{P}_n^{\alpha,\beta}(x) = \tilde{P}_n^{\alpha,\beta}(x), x \in (0,1] \tag{9} $$

Similarly, a right Jacobi convolution polynomials are defined in [3].

Remark 1 *Jacobi convolution polynomials forms a basis function in infinite dimensional Hilbert space [3]*

4 General Fractional Physics-informed Neural Networks (GFPINNs)

In this section, we outline the development of General Fractional Physics-informed Neural Networks (GFPINNs). For the details of Physics-informed Neural Networks (PINNs), refer [19]. This work was extended for fractional case in [9] followed by application to turbulence in [7] and development of pointwise algorithm for both fractional and tempered fractional operators in [8].

Let's consider the below boundary value problem (10) to outline the development of GFPINNs.

$$\begin{aligned} {}^{RL}_{0}D^{(k)}_{x} f(x) \;&=\; g(x),\; x \in (0,1), \\ f(0) = 0,\; f(1) &= b. \end{aligned} \tag{10}$$

Recall that, in classical approach of PINNS, the loss function has two components: (a) PDE residual (b) boundary term.

We start by obtaining the residual (R_N) from the equation (10). Since the Jacobi convolution polynomial (7) is a basis function, we seek an approximation of type (11), where $\hat{f}_n$ are the coefficients.

$$f_N(x) \;=\; \sum_{n=0}^{N-1} \hat{f}_n \phi_n(x) \;=\; \sum_{n=0}^{N} \hat{f}_n \left(\kappa * \tilde{P}^{\alpha,\beta}_n\right)(x) \tag{11}$$

Plugging (11) into (10) and using (9), we get the residual (R_N) as (12).

$$R_N \;:=\; {}^{RL}_{0}D^{(k)}_{x} f_N \;-\; g(x) \;=\; \sum_{n=0}^{N} \hat{f}_n \tilde{P}^{\alpha,\beta}_n(x) \;-\; g(x) \tag{12}$$

In [3] a variational method was constructed by computing a weighted inner product of the above residual and a suitable test function such that orthogonality holds.

However, in GFPINNs, we shall not follow a variational approach in favour of avoiding errors from numerical integration and by reconsigning the system (9) is the classical separation of variables. Such a system can be solved by any non-linear solves and a neural network being the best candidate.

For the boundary term ($\mathcal{B}_N$), a Tau approach is implemented at $x = 1$ (13). Note that, since $\phi_n(0) = 0, \forall n \in \mathbb{N} \cup \{0\}$ the no additional boundary term is required at $x = 0$.

$$\mathcal{B}_N \;:=\; \sum_{n=0}^{N-1} \hat{f}_n \phi_n(1) - b \tag{13}$$

At this point, one may construct a loss function by using (12) and (13). However, we take one more step further, by invoking another neural network (*phiNet*) with input as $\{x_m\}_{m=0}^{M} \in [0,1]$ and output being the basis function. The corresponding loss term for phiNet is (14), where ϕ_n^{NN} denotes the neural network output of basis function.

$$\Phi_N \;:=\; \phi_n - \phi_n^{NN} \tag{14}$$

Such a neural network, namely, phiNet can be pre-trained separately and then basis functions can be obtained on the fly. Furthermore, we may chose the replace phiNet with a neural operator [20]. By virtue of a neural operator for phiNet, we could then provide arbitrary kernels to compute convolutions to compute basis functions on the fly. We leave this investigation for our future full paper.

In our present construction, we invoke two neural networks with the joint loss function (15). The first neural networks input is a vector $n = \{0, 1, \ldots N-1\}$ and output being the coefficients ($\hat{f}_n$). The second neural network is phiNet. In (15), $\{x_m\}_{m=0}^{M} \in [0,1]$ are the spatial points and $\lambda_e, \lambda_b, \lambda_p$ are the hyper-parameters to balance each term.

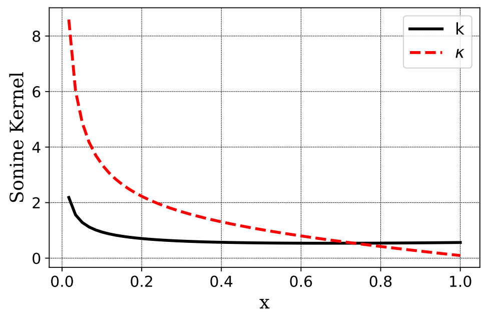
$$loss \;=\; \frac{\lambda_e}{M}\sum_{m=0}^{M}|R_N|^2 \;+\; \frac{\lambda_b}{M}\sum_{m=0}^{M}|B_N)|^2 \;+\; \frac{\lambda_p}{M}\sum_{m=0}^{M}|\Phi_N|^2 \tag{15}$$

Example: As an example we choose, $f(x) = x^6$. The Sonine kernel (see [12]) is constructed as (16). For $\alpha = 0.5$ and $a = \{0.5, 0.25, 0.25\}$ results, $b = \{2, -1, -0.83333\}$ using the relationship (16). We plot the Sonine pair in fig. 1a, note that these kernel have singularity at $x = 0$, however, ϕ_n is a non-singular function. In this example, for our construction of Jacobi convolution polynomial, we chose $\alpha = \beta = 0$ in (7) corresponding to Legendre polynomials. Indeed, we observe a good convergence to the true solution (fig. 1b).

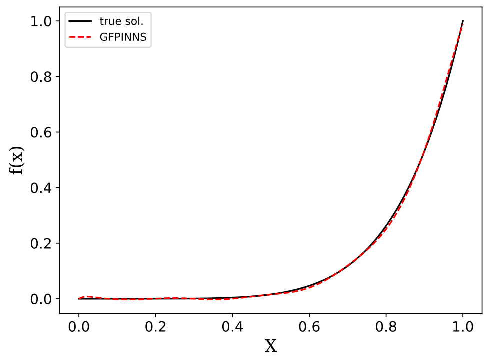
$$k(x) = \frac{x^{-\alpha}}{\Gamma(1-\alpha)}\sum_{k=0}^{N} a_k x^k, \quad \kappa(x) = \frac{x^{\alpha-1}}{\Gamma(\alpha)}\sum_{k=0}^{N} b_k x^k$$

where, the coefficients follow the relationship, (16)

$$a_0 b_0 \;=\; 1,\; k=0, \quad \sum_{k=1}^{N}\Gamma(k+1-\alpha)\Gamma(\alpha+N-k)a_{N-k}b_k = 0,\; k = \{1,2,\ldots,N\}$$



(a) Sonine Kernel obtained using (16) for $\alpha = 0.5$ and $a = \{0.5, 0.25, 0.25\}$ results in $b = \{2, -1, -0.83333\}$ are singular functions with singularity at $x = 0$.



(b) Comparison of the results of GFPINNs (red dashed line) with the true solution (black solid line).

Figure 1: Results of GFPINNs

5 Summary

In this short paper, we outlined the key idea of handling general fractional operators (a further generalisation of fractional calculus, which admits an arbitrary kernel) within the framework of PINNs. Needless to mention they are more generally applicable.

- By introducing separation of variable via Jacobi convolution polynomial (7) [3] results in a more accurate scheme, since we avoid errors from numerical integration required for variational scheme [3]. Such a system can be solved by neural network, which are non-linear solvers.
- It also results in an computationally efficient scheme, since we do not compute a discrete convolution, as opposed to to fPINNs [9, 7, 8], where an $L1$ scheme was used.
- In GFPINNs, we propose two using neural networks, one for the basis functions (phiNet) and other for coefficients for the approximation of type (11), since we recognise phiNet can be pre-trained or equally be replaced by neural operators. Thereby, the flexibility of obtaining basis functions on the fly. We shall report our finding this direction in our future full paper.
- Integer-order operators can now take advantage of automatic differentiation (of phiNet), thereby mitigating the need of developing a splitting scheme.

Declaration of interest

The authors declare no conflict of interests.

Funding

This work is funded by PNRR digitalisation, Itlay.

References

[1] Anatoly N Kochubei. General fractional calculus, evolution equations, and renewal processes. *Integral Equations and Operator Theory*, 71(4):583–600, 2011.

[2] Yuri Luchko. General fractional integrals and derivatives of arbitrary order. *Symmetry*, 13(5):755, 2021.

[3] Pavan Pranjivan Mehta and Gianluigi Rozza. Jacobi convolution polynomial for petrov-galerkin scheme and general fractional calculus of arbitrary order over finite interval. *arXiv:2411.08080*, 2024.

[4] Kai Diethelm. *The analysis of fractional differential equations, volume 2004 of Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2010.

[5] Igor Podlubny. *Fractional differential equations*. Academic, 1999.

[6] Keith Oldham and Jerome Spanier. *The fractional calculus theory and applications of differentiation and integration to arbitrary order*. Elsevier, 1974.

[7] Pavan P. Mehta, Guofei Pang, Fangying Song, and George Karniadakis. Discovering a universal variable-order fractional model for turbulent couette flow using a physics-informed neural network. *Fractional Calculus and Applied Analysis*, 22(6):1675–1688, 2019.

[8] Pavan Pranjivan Mehta. Fractional and tempered fractional models for reynolds-averaged navier–stokes equations. *Journal of Turbulence*, 24(11-12):507–553, 2023.

[9] Guofei Pang, Lu Lu, and George Em Karniadakis. fpinns: Fractional physics-informed neural networks. *SIAM Journal on Scientific Computing*, 41(4):A2603–A2626, 2019.

[10] Fujihiro Hamba. History effect on the reynolds stress in turbulent swirling flow. *Physics of Fluids*, 29(2):025103, 2017.

[11] Robert Kraichnan. Eddy viscosity and diffusivity: exact formulas and approximations. *Complex Systems*, 1(4-6):805–820, 1987.

[12] N Sonine. Sur la généralisation d'une formule d'abel. 1884.

[13] Yuri Luchko. General fractional integrals and derivatives with the sonine kernels. *Mathematics*, 9(6), 2021.

[14] Mohammed Al-Refai and Yuri Luchko. The general fractional integrals and derivatives on a finite interval. *Mathematics*, 11(4):1031, 2023.

[15] Ivan Dimovski. Operational calculus for a class of differential operators. *CR Acad. Bulg. Sci*, 19(12):1111–1114, 1966.

[16] Arran Fernandez and Mohammed Al-Refai. A rigorous analysis of integro-differential operators with non-singular kernels. *Fractal and Fractional*, 7(3), 2023.

[17] Mohsen Zayernouri and George Em Karniadakis. Fractional sturm–liouville eigen-problems: theory and numerical approximation. *Journal of Computational Physics*, 252:495–517, 2013.

[18] Sheng Chen, Jie Shen, and Li-Lian Wang. Generalized jacobi functions and their applications to fractional differential equations. *Mathematics of Computation*, 85(300):1603–1638, 2016.

[19] Maziar Raissi, Paris Perdikaris, and George Karniadakis. 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, 2019.

[20] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.