scientific reports

Check for updates

OPEN

Machine learning based on a generative adversarial tri-model

Song Wang[1]✉, Ning Xi[1] & Zhengfang Zhou[2]

This paper proposes a novel machine learning paradigm called the generative adversarial tri-model (GAT) to incorporate analytical knowledge into neural networks through a unique positive-sum game strategy. The motivation is to solve the problem that pure machine learning models fail to obey the fundamental governing laws of physics in engineering fields. The GAT method is successfully implemented to solve ODE (ordinary differential equation) problems with various constraints. A strict error bound is proven for initial-constraint problems, which certifies its reliability. The real-world significance of the GAT method is reflected by its application to a human body oscillation recovery problem, based on balance sensor measurements, which is critical for human balancing evaluation, yet unresolved after massive precedent research work. Further human experiment results prove the effectiveness of the GAT method. Both theoretical and experimental studies demonstrate that the GAT method is useful and reliable. It envisions great scalability for wider applications and adaptions prospect.

Keywords GAT method, PINN, DEDS, Nonlinear ODE, Human balancing evaluation

Human balance is defined as the body's ability to maintain its center of gravity within the base of support formed by feet[1], which is a basic physiological function. Being able to keep balance is fundamental for us to perform daily activities, such as standing and walking. Maintaining balance requires sensory inputs from our visual system, proprioceptive system and vestibular system[2], as well as the coordination within our motion control system comprised of brain, spine, muscle, etc. Lots of factors can lead to deterioration of balance ability, such as aging[3], alcohol[4], drugs[5], limb pain[6] and brain injury[7]. Poor balance can make our everyday life difficult and even induce falls, which account for a variety of injuries and disabilities. Especially for the elderly, falls incur billions of medical cost every year[8].

Balance test has been widely used to evaluate human balance ability in medical diagnosis activities which can check for potential balance disorders, and what is causing it. For example, it can act as a non-invasive tool to provide important information for diagnosing neural diseases, such as Parkinson's disease[9] and stroke[10]. For other diseases diagnosis like migraine[11], Meniere's disease[12] and so on, it can play a helpful role as well. In addition, it can assist in detecting mild cerebral injury[13] without obvious physical lesion like concussion[14] which cannot be diagnosed with CT scan. The detection of subtle abnormalities in balance ability is necessary not only for early diagnosis but for accurate assessment of rehabilitation progress[15]. Apart from medical usages, balance test can also be used in law-enforcement scenarios, like identifying drunk drivers and drug addicts[16]. When it comes to sports, balance test can also serve many purposes, such as risk identification of ankle sprains[17] and sports talent identification and selection[18].

With all these benefits of balance test, researchers have already developed various balancing evaluation methods, which can be roughly categorized as either human-scoring or machine-scoring. Typical human-scoring methods include Romberg test[19,20], OLST method[21], Tinetti test[22] and so on. A common characteristic of these methods is that there must be a human evaluator to give scores for testees' performance on each task, so the result is hard to avoid being subjective. Machine-scoring methods are based on balance testers which originate from 1976[23]. With a balance tester, the pressure distribution under feet can be recorded when testee standing. Traditionally, most human balancing evaluations are based on CoP (center of pressure) measurements[24], such as the mean velocity and root mean square distance of CoP displacements, as recommended by the International Society of Posturography[25]. However, the CoP variations do not directly reflect the mechanical motion of human body. Hence, some researchers have tried to measure the CoG (center of gravity) variations of human body using methods like cinematography and optoelectrical imaging systems, etc.[26] However, these methods are also indirect and involve anthropometric approximations[27]. Some researchers even tried to estimate CoG from CoP, such as by filtering the time history data of CoP[28]. Nevertheless, these methods fail to consider the kinetic

[1]Department of Data and Systems Engineering, The University of Hong Kong, Pok Fu Lam, Hong Kong. [2]Department of Mathematics, Michigan State University at East Lansing, East Lansing, USA. ✉email: songwang@connect.hku.hk

properties of the human body. To fill this research gap, this paper proposes a new approach called the GAT method to recover the CoG motion process from CoP measurements, without losing the kinetic properties of human body.

In fact, a standing human body can be regarded as an inverted pendulum[29] which is under control from the cerebellum for keeping balance[30]. Therefore, this paper simplifies a standing human body as a nonlinear two-degree-of-freedom inverted pendulum and establishes its dynamic model, which consists of two second-order nonlinear ODEs. The inputs of these equations are the CoP measurements from a balance sensor. If given proper constraints, it should be able to numerically solve the human body motion and hence the CoG motion process from this dynamic model. Traditionally, there are lots of methods to generate numerical solutions for ODEs, such as the well-known Euler method and Runge–Kutta method, but they only work for initial/boundary-constraint problems. Other classical methods include the finite difference method and shooting method, but they only work for linear problems. Unfortunately, no initial constraint can be found for solving human body motion from the established nonlinear dynamic equations. Instead, a set of integral constraints can be obtained heuristically. For nonlinear integral-constraint problems, the weighted residual method can be used, but the constraints cannot be strictly satisfied.

Interestingly, from a novel perspective, an ODE solving problem can be treated as a function fitting task, where the target function is its solution. Therefore, it can be solved through machine learning methods like neural networks. Physics-informed neural networks (PINN)[31] methods approximate ODE solutions with deep neural networks. During training, its loss function includes two terms, which respectively measure the residual of differential equations and the residual of constraints. However, the constraints cannot be strictly satisfied, either. Moreover, competitive loss terms during network training can lead to conflicting gradients[32], which may impair training stability and cause PINN hard to accurately learn the underlying ODE solution.

Exact constraint satisfaction is very important for a variety of problems in scientific domains, particularly when confidence in the constraint information is high[33]. For example, the constraints may imply certain symmetries or conservation laws[34]. Theory of functional connections (TFC) methods derive constrained expressions as trial solutions of ODE problems, which contain a free function and always satisfy a set of linear constraints no matter what the free function is[35]. Optimization methods are then leveraged to determine the unknown parameters in the free function. TFC methods can solve both linear and nonlinear differential Eqs. [36,37] subject to initial, boundary and multi-value constraints. The free function used in TFC methods can either be a linear combination of orthogonal basis functions, or extreme learning machines[38], or deep neural networks[33,39,40]. However, integral-constraint differential equations can only be solved by TFC methods based on the first two kinds of free functions[41,42], while TFC with deep neural networks can only solve initial/boundary-constraint problems. Another limitation of the TFC methods is that they cannot tackle nonlinear constraints.

Although solving ODE problems with neural networks offer potential benefits compared with traditional numerical methods, they still face the challenge of very limited convergence theory. In addition, the PINN methods and TFC methods pose ODE problems as optimization problems, so they inevitably share all the difficulties with optimization problems, with the major one being getting stuck in local optima[43]. Furthermore, current optimization-based methods can only solve ODE problems from scratch. They cannot take advantage of approximate solutions from other methods.

There lacks of a unified framework which can solve both linear and nonlinear ODE problems with arbitrary constraints. More importantly, it must guarantee exact satisfaction of all constraints and provide convergence theory. Besides, it would be preferable if it can leverage approximate solutions from other methods and provide mechanisms to jump out of local optima. The GAT method is able to fill this research gap, which ingeniously decouples the trial solutions with constraints and treats the constraints as extra analytical knowledge that the trial solutions generated by neural networks should further satisfy. This idea leads to a two-stage positive-sum strategy, which is the essence of the GAT method. The GAT method not only can solve ODE problems with arbitrary constraints. In fact, it is a general framework to fuse two originally mutual-exclusive ways to describe the world, i.e., data-driven connectionism and traditional symbolism.

Results

Concept of the GAT method

Schematic diagram of the GAT method is shown in Fig. 1. Explicitly, there are two types of models, namely the machine learning model and analytical model. The machine learning model is always used to represent some functional relationship and can be independently trained. But the generated result may be unsatisfactory or fail to conform to some necessary constraints because the training data and loss function may not contain all the necessary information to describe the desired function. However, if some extra knowledge can be obtained to describe the desired function, such as some physical equation, it can be used to adjust the machine learning outputs. Here the analytical model represents a mathematical description of the extra knowledge. But this analytical model must also be inadequate to determine the desired function. Otherwise, there will be no need to apply machine learning models. Therefore, the adjusted outputs must be further refined in the machine learning model. The method is to re-initialize the machine learning model based on the adjusted outputs and train it again. The whole process forms a loop and will converge when the machine learning model outputs conform to the analytical model to some acceptable level or vice versa.

In the GAT method, the desired functional relationship is the ultimate purpose, which is generated by the machine learning model, so the machine learning model acts as a generator, represented by the "G" letter. The machine learning model and analytical model can optimize the desired function in turn in an adversarial way, which is the meaning of the "A" letter. The two models are in competition because the machine learning model wants its generated function to conform to its training data and loss function in terms of minimal loss value, while the analytical model wants its adjusted function to be closer to its contained knowledge. However, this

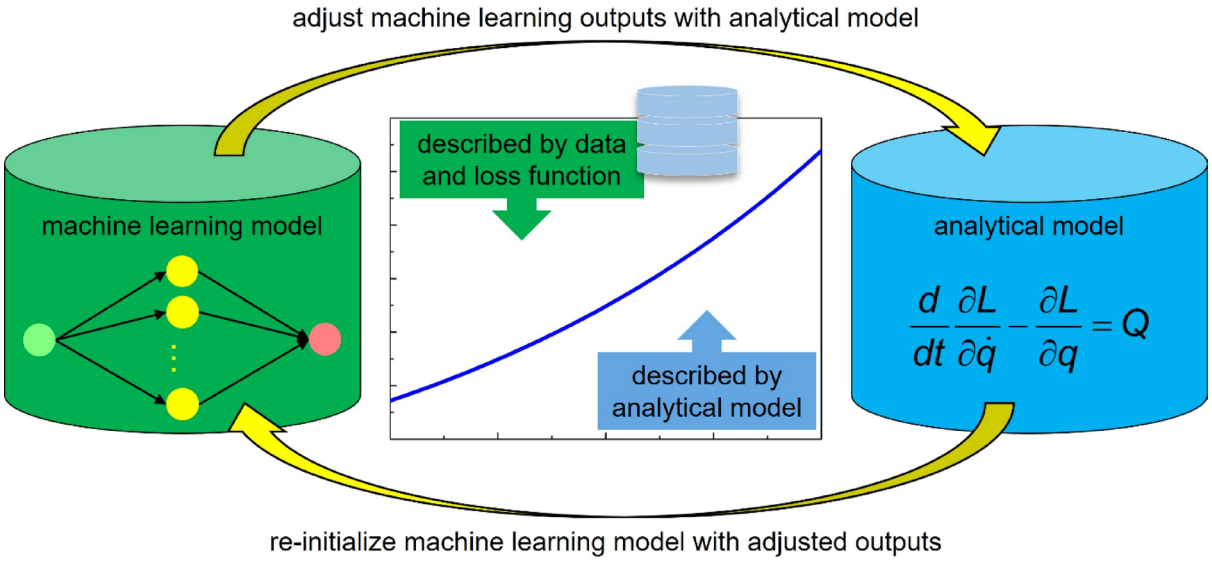

Fig. 1. Schematic diagram of the GAT method.

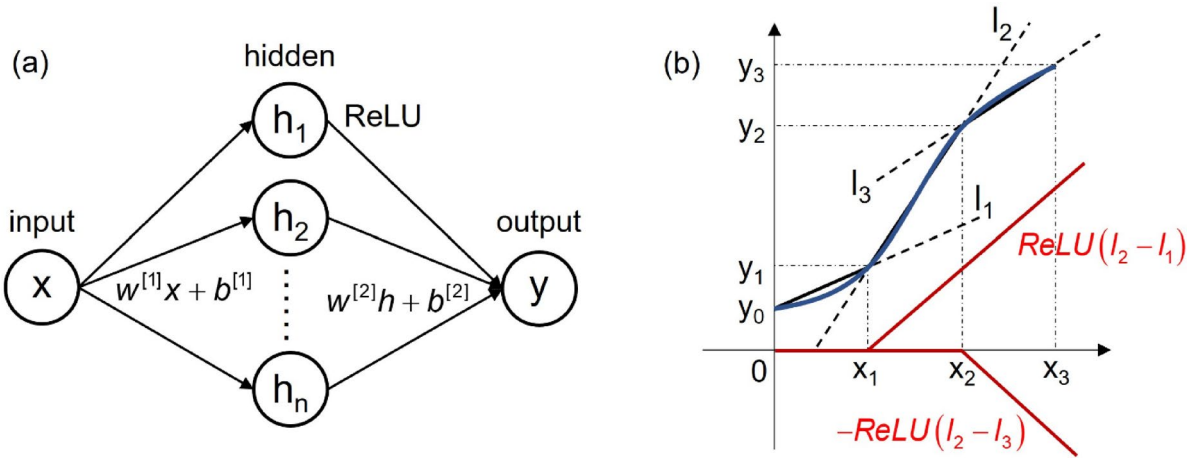

Fig. 2. Elaboration of the neural networks used to realize the GAT method. (**a**) The weights of the hidden layer and output layer are represented by $w^{[1]}$ and $w^{[2]}$, respectively. The biases of the hidden layer and output layer are represented by $b^{[1]}$ and $b^{[2]}$, respectively. The functional relationship of these networks can be expressed as $y = \sum_{i=1}^{n}\left[w_i^{[2]} ReLU\left(w_i^{[1]}x + b_i^{[1]}\right)\right] + b^{[2]}$. (**b**) Illustration of the procedure to reproduce an arbitrary function with these networks.

competition is positive-sum rather than zero-sum as what is in the GAN model[44], because at the convergence moment, the generated function is expected to conform to both the loss function and analytical model. In fact, there is another model implied in the running process of the GAT method. Its state transitions during running can be described by a DEDS (discrete event dynamic system) model[45], which is a logic model built upon non-deterministic finite automata. The machine learning model, analytical model and DEDS model construct the tri-model, denoted by the "T" letter. The most vital conjunction point in the GAT method is how to re-initialize the machine learning model with adjusted outputs. This paper provides a feasible approach with respect to connectionist models, i.e., neural networks.

Realization of the GAT method with neural networks

It is well-known that fully connected neural networks with just one hidden layer can approximate all continuous functions, regardless of whether the activation function is sigmoid[46,47] or ReLU[48]. Therefore, ReLU-activated fully connected neural networks with one hidden layer are employed to realize the GAT method, as shown in Fig. 2a. These networks can approximately reproduce any function in a very simple procedure, as illustrated in Fig. 2b. For an arbitrary function (indicated by the blue curve), a series of points $(0,\ y_0)$, $(x_1,\ y_1)$, $(x_2,\ y_2)$

and $(x_3,\ y_3)$ can be sampled from it. The original function can be approximated by a piecewise linear function consisting of the line segments connecting every two adjacent points. The prolongations of these line segments are represented by l_1, l_2 and l_3. For the specific case in Fig. 2b, the piecewise linear function can be expressed by $l_1 + ReLU\,(l_2 - l_1) - ReLU\,(l_2 - l_3)$, which naturally conforms to the functional relationship represented by these networks.

For a general case, if the sampled data points are $(x_0,\ y_0)\,,\ (x_1,\ y_1)\,,\ \ldots,\ (x_n,\ y_n)$, a formulaic algorithm has been developed to set every parameter in a network with n hidden nodes, allowing it to reproduce the piecewise linear function connecting these sampled points. This algorithm, named as "Algorithm 1", is expressed with pseudocode as follows, where $sgn\,(\cdot)$ is the sign function.

Algorithm 1:

$$w_1^{[1]} \leftarrow \left|\frac{y_1 - y_0}{x_1 - x_0}\right|$$

$$b_1^{[1]} \leftarrow -w_1^{[1]} \cdot x_0$$

$$w_1^{[2]} \leftarrow \mathrm{sgn}\left(\frac{y_1 - y_0}{x_1 - x_0}\right)$$

for i from 2 to n

$$w_i^{[1]} \leftarrow \left|\frac{y_i - y_{i-1}}{x_i - x_{i-1}} - \frac{y_{i-1} - y_{i-2}}{x_{i-1} - x_{i-2}}\right|$$

$$b_i^{[1]} \leftarrow -w_i^{[1]} \cdot x_{i-1}$$

$$w_i^{[2]} \leftarrow \mathrm{sgn}\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}} - \frac{y_{i-1} - y_{i-2}}{x_{i-1} - x_{i-2}}\right)$$

end

$$b^{[2]} \leftarrow y_0$$

The process of running "Algorithm 1" to make the network reproduce the piecewise linear function connecting those sampled points can be vividly described with the following metaphor. Assume there is a wire which initially coincides with the x-axis. The problem is how to bend and move this wire so that it can coincide with the piecewise linear function. The first step is to bend this wire at position x_0 so that the wire segment within $[x_0, x_1]$ is parallel to the first line segment of the piecewise function. The second step is to further bend this wire at position x_1 so that the wire segement within $[x_1, x_2]$ is parellel to the second line segment of the piecewise function. This process continues until the wire segment within $[x_{n-1}, x_n]$ is parallel to the last line segment of the piecewise function. Lastly, the wire is vertically moved as a whole, with moving distance being y_0. The final shape and position of the wire will exactly coincide with the piecewise linear function from x_0 to x_n. The network output includes a summation of a series of ReLU activated linear functions, which can be used to imitate this sequential bending process. Specifically, the bending angles are indicated with the elements of $w^{[1]}$, while the elements of $b^{[1]}$ are determined accordingly so as to ensure the bending positions are exactly x_0, x_1, x_2, …, x_{n-1}. The bending directions are indicated with the elements of $w^{[2]}$, where +1 means bending upward and −1 means bending downward. The final vertical movement is indicated with the value of $b^{[2]}$. With the "Algorithm 1", any adjusted outputs from analytical models can be used to re-initialize the neural network so that the adjusted outputs can be refined with further training.

The idea of the GAT method has already been proven to be effective in sensors calibration[48,49], which are supervised learning problems. Neural networks were used to fit the desired sensor property functions. However, experiment data was hard to obtain, hence the amount of data was rather noisy and deficient. Direct training on the limited dataset gave very poor results. At the same time, extra qualitative physical models describing the sensor property functions were established under ideal assumptions. Although they diverge from reality, the physical knowledge they contain greatly helped the network converge from both the perspective of speed and accuracy.

Adaption of the GAT method for solving ODEs

Here, the GAT method is applied to unsupervised learning problems. Specifically, it is adapted to numerically solve ODEs. Using the GAT method to solve an ODE problem, the differential equation and its constraint will be decoupled. The neural network in the GAT method only focuses on decreasing the residual of the corresponding difference equations of the original differential equation, while the analytical model guarantees the constraint being satisfied. All kinds of constraints can be handled by this paradigm.

Given an ODE problem, suppose the differential equation and constraint are expressed with Eq. (1), where $\Psi\,[\cdot]$ denotes a functional of the solution $y\,(x)$. If numerical solutions of x within [0, 1] are considered, step size h is used to discretize the interval [0, 1] and the number of hidden nodes n in the neural network is set to $1/h$. The inputs of the network are $0,\ h,\ 2h,\ \cdots,\ 1$ and the corresponding outputs of the network are denoted with $Y_0, Y_1, Y_2, \ldots, Y_n$, which are regarded as the numerical solutions of the ODE problem. Therefore, the constraint with respect to these numerical solutions should be rephrased as Eq. (2). Two types of loss functions are designed, which are called Euler loss function and Runge–Kutta loss function, respectively, as expressed by

Eqs. (3) and (4). What the loss functions actually calculate is the mean square residual in different orders of the corresponding difference equations of the original differential equation.

$$\begin{cases} \dfrac{dy}{dx} = f(x, y) \\ \Psi[y(x)] = 0 \end{cases} \tag{1}$$

$$\Psi(Y_0, Y_1, Y_2, \ldots, Y_n) = 0 \tag{2}$$

$$L_{Euler} = h \sum_{i=0}^{n-1} \left[\frac{Y_{i+1} - Y_i}{h} - f(ih, Y_i) \right]^2 \tag{3}$$

$$L_{RK} = h \sum_{i=0}^{n-1} \left[\frac{Y_{i+1} - Y_i}{h} - \frac{1}{6}(k_{i1} + 2k_{i2} + 2k_{i3} + k_{i4}) \right]^2 \tag{4}$$

$$\begin{aligned} where \quad k_{i1} &= f(ih, Y_i) \\ k_{i2} &= f\left(ih + \frac{h}{2}, Y_i + \frac{h}{2}k_{i1}\right) \\ k_{i3} &= f\left(ih + \frac{h}{2}, Y_i + \frac{h}{2}k_{i2}\right) \\ k_{i4} &= f(ih + h, Y_i + hk_{i3}) \end{aligned}$$

The whole running process of the GAT method for solving the above ODE problem is illustrated in Fig. 3. The neural network can either be initialized randomly or with the "Algorithm 1" to reproduce approximate numerical solutions generated by other methods, such as the finite difference method. With inputs and loss function definitions, the network can now be trained with back propagation until convergence to minimize the loss value. After convergence, the network outputs $Y_0, Y_1, Y_2, \ldots, Y_n$ may not satisfy the constraint Eq. (2). Then, in the analytical model, translation and scaling can be used to adjust these outputs to make Eq. (2) satisfied. In a general way, this can be achieved by solving an optimization problem so that the modification to network outputs can be minimized, as expressed by Eq. (5), where C_1 and C_2 denote the scaling parameter and translation parameter, respectively, and $C_1Y_0 + C_2, C_1Y_1 + C_2, \ldots, C_1Y_n + C_2$ are the adjusted network outputs. However, the constraint is not always convex, so this optimization problem may not be solved conveniently. In many cases, it is not necessary to solve Eq. (5), but heuristic methods can be used to directly generate a pair of C_1 and C_2, without much influence on the running of the GAT method.

$$\begin{aligned} &\min_{C_1,\, C_2} \sum_{i=0}^{n} (C_1Y_i + C_2 - Y_i)^2 \\ &s.t. \quad \Psi(C_1Y_0 + C_2, C_1Y_1 + C_2, \ldots, C_1Y_n + C_2) = 0 \end{aligned} \tag{5}$$

A new loss value can be calculated for these adjusted network outputs, and if it is less than a preset threshold ε, the original ODE problem can be considered solved, and these adjusted network outputs are the corresponding

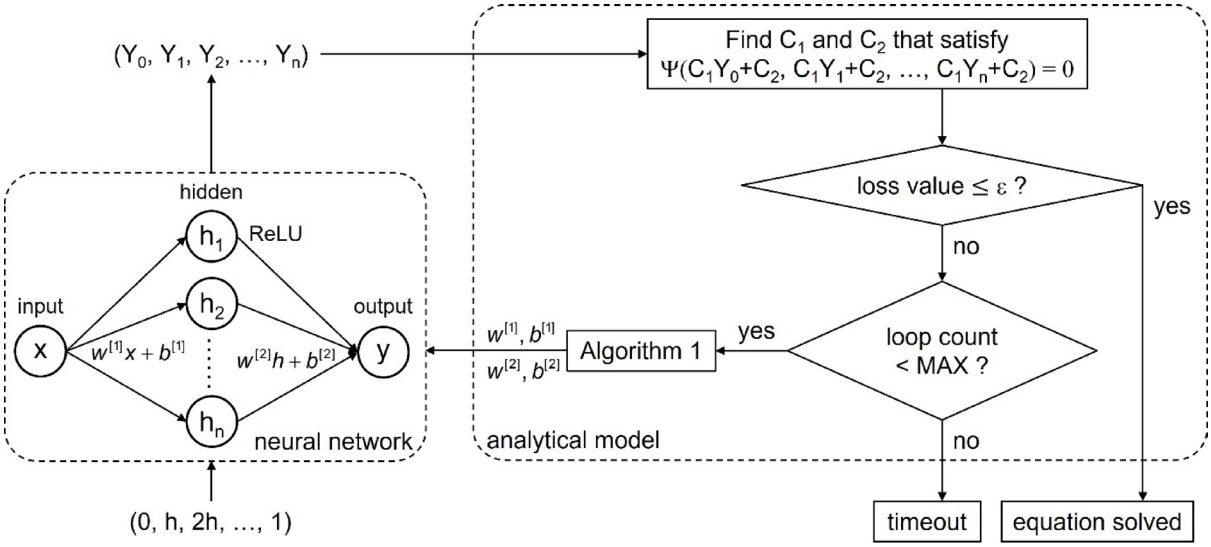

Fig. 3. Adaption of the GAT method for solving ODEs.

numerical solutions. Otherwise, the "Algorithm 1" can be used to re-initialize the network to reproduce $(0,\ C_1Y_0 + C_2)$, $(h,\ C_1Y_1 + C_2)$, $\cdots$, $(1,\ C_1Y_n + C_2)$ so that the neural network can jump out of local optima and be trained again to lower the loss value. In ideal circumstances, the GAT method will converge to a series of outputs that both give zero loss value and satisfy the constraint. If an initial constraint and the Euler loss function are used, it can be easily proven that the GAT method outputs at the ideal convergence moment are the same as the numerical solutions generated by Euler method. If the Runge–Kutta loss function is used, the outputs will be the same as those generated by Runge–Kutta method[50]. Hence, these two loss functions are named as such. Besides, a "MAX" value is preset to limit the loop number between the neural network and analytical model to avoid a dead loop.

Notably, the reason to use translation and scaling for adjustments is that these two operations do not change the relative relationships between adjacent numerical solutions, so they can retain the knowledge of derivatives learned from differential equations. Meanwhile, translation and scaling towards meeting constraints will further add the knowledge of constraints to numerical solutions. That is the reason why the GAT method is positive-sum. This positive-sum characteristic of GAT method will finally aggregate all the knowledge contained in differential equations and constraints into numerical solutions, which means the eventual convergence of the GAT method. Now that both the neural network and analytical model tend to modulate the GAT method outputs towards the unique convergence state, the GAT method will naturally tend to converge, which means it is intrinsically stable.

For the network outputs, two indexes can be defined as follows: whether the gradients of the network parameters in the loss function are zero (or whether the network converges), and whether the constraint Eq. (2) is satisfied. The two indexes can form three states. In state 1, the gradients of network parameters are not zero and the constraint is not satisfied, which corresponds to the state after network initialization and during network training. In state 2, the gradients of network parameters are zero, but the constraint is not satisfied, which corresponds to the state after network convergence. In state 3, the gradients of network parameters are not zero, but the constraint is satisfied, which corresponds to the state after network being re-initialized by the "Algorithm 1". The end of the GAT method is defined as state 4. The state transitions between the four states are triggered by different events. The above description is exactly a DEDS model[45], as illustrated in Fig. 4, where event α represents that convergence is achieved during network training; event β represents that the "Algorithm 1" is run to make network outputs satisfy the constraint; event γ represents training the network; event η represents that the GAT method is terminated. Here, event γ and η are controllable according to a criterion u, which represents that the loss value $\leq$ ε or loop number $\geq$ MAX. Event η happens when u is satisfied. Otherwise, event γ happens.

The formal definitions of the quadruple and functions in the above DEDS model are detailed as follows: X = {state 1, state 2, state 3, state 4} is the finite set of states. Σ = {α, β, γ, η} is the finite set of possible events. U = {γ, η} is the set of admissible control inputs, which is a subset of Σ. Γ = {α, β, γ, η} is the set of observable events. The functions that specify the set of possible events at each state are d(state 1) = {α}, d(state 2) = {β}, d(state 3) = {γ, η}, d(state 4) = Ø. The functions that specify the set of events that cannot be disabled at each state are e(state 1) = {α}, e(state 2) = {β}. The state transition functions are f(state 1, α) = {state 2}, f(state 2, β) = {state 3}, f(state 3, γ) = {state 1}, f(state 3, η) = {state 4}. Here, in order to ensure there is no dead loop, state 4 is identified as the "good" state E in the DEDS model. Since the loop number is restricted by "MAX", state 4 will definitely be reached, which means the DEDS model is stable.

As there is only one neural network and one analytical model in the ODE solving case, the DEDS model seems trivial here. However, the GAT method has great scalability. For example, multi machine learning models and analytical models could be involved. Evaluating the outputs of the GAT method may involve lots of indexes, and hence, lots of states. The state transitions between these states will become very complex. Under such circumstance, a helpful tool like the DEDS model is necessary to analyze the running process of the GAT method and control its stability. The operation that the GAT method takes different actions in some states according to certain criteria resembles the policy iterator in learning automata or reinforcement learning. However, the GAT method is different since it involves no Markov stochastic process and there is no reward from outside environment for updating policy. The GAT method is a closed system that runs independently with a preset policy.

Application of the GAT method in solving ODEs

Five examples are given to demonstrate the ability of GAT method to solve ODE problems with various constraints, including two initial-constraint problems, one boundary-constraint problem, one integral-constraint problem and one nonlinear-constraint problem. Each example has its analytical solution for reference purpose. In all five

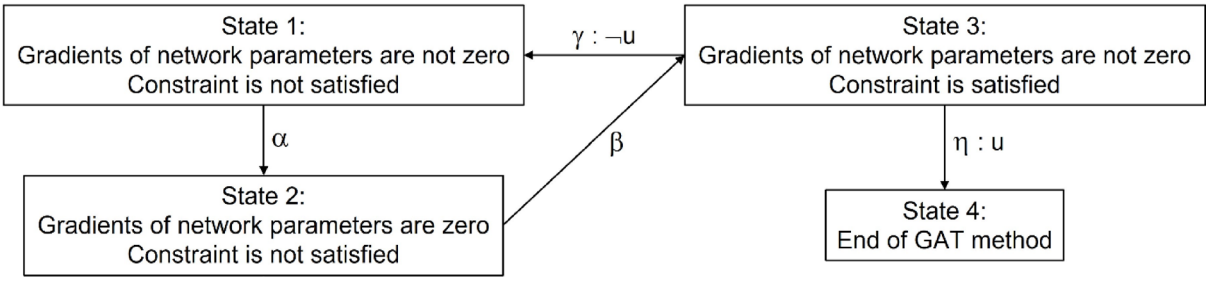

Fig. 4. DEDS description of the state transitions in the GAT method for solving ODEs.

examples, neural networks are used in the GAT method to give numerical solutions with a step size being 0.01, which are initialized randomly and trained under both the Euler loss function and Runge–Kutta loss function. For comparison purpose, PINN method is also employed to solve these five examples. For the sake of fairness, the networks used in the PINN method are the same as those used in the GAT method. The Euler and Runge–Kutta loss functions are also used to measure the residual of differential equations, as part of the PINN loss functions. The GAT method and PINN method share the same convergence criteria during network training.

$$\text{Problem 1}: \quad \begin{cases} \dfrac{dy}{dx} = -\dfrac{xy}{x+2}, & x \in [0,1] \\ y(0) = 4 \end{cases}$$

$$\text{Problem 2}: \quad \begin{cases} \dfrac{dy_1}{dx} = -2y_1 + 4y_2, & x \in [0,1] \\ \dfrac{dy_2}{dx} = -y_1 + 3y_2 \\ y_1(0) = 5 \\ y_2(0) = 2 \end{cases}$$

$$\text{Problem 3}: \quad \begin{cases} \dfrac{dy}{dx} = \dfrac{1 - y\sin x}{\cos x}, & x \in [0,1] \\ y(1) - y(0) = \sin 1 + \cos 1 - 1 \end{cases}$$

$$\text{Problem 4}: \quad \begin{cases} \dfrac{dy}{dx} = \dfrac{y \ln y}{x+1}, & x \in [0,1] \\ \displaystyle\int_0^1 y(x)\,dx = e^2 - e \end{cases}$$

$$\text{Problem 5}: \quad \begin{cases} \dfrac{dy}{dx} = y^2 - y, & x \in [0,1] \\ \displaystyle\int_0^1 \frac{1}{y(x)}dx = e^3 - e^2 + 1 \end{cases}$$

The analytical solutions for the five problems are $y = e^{-x}(x+2)^2$, $\begin{cases} y_1 = e^{2x} + 4e^{-x} \\ y_2 = e^{2x} + e^{-x} \end{cases}$, $y = \sin x + \cos x$, $y = e^{x+1}$ and $y = \frac{1}{1+e^{x+2}}$, respectively. To illustrate the running process of the GAT method for solving ODE problems, the state transitions in the first three loops for solving Problem 1 are illustrated (refer to Supplementary Materials Section S1). The generated numerical solutions by both methods with both loss functions are depicted in Fig. 5. The final numerical solutions by the GAT method are found to be very accurate for all kinds of constraints. For initial-constraint problems, it is also found that the performance of Runge–Kutta loss function is slightly superior to that of Euler loss function. This is unsurprising since the error bound of the GAT method with Runge–Kutta loss function is tighter than that with Euler loss function for the same loss value (refer to the following theorem and corollary). It is clear that the GAT method can generally generate more accurate and smoother results than PINN method. This is because the training of PINN is prone to getting stuck in local optima. Fortunately, in the GAT method, once networks converge to local optima, which means the GAT method reaches state 2, the "Algorithm 1" will be run to re-initialize networks to make their outputs conform to constraints. An amazing byproduct of this procedure is that it can help networks jump out of local optima. Meanwhile, the drawback of PINN methods that they cannot strictly satisfy constraints is obvious from the results of the first two initial-constraint problems. In contrast, the solutions generated by the GAT method can always strictly satisfy constraints. Note that the GAT method can solve multiple first-order differential equations like the Problem 2. It must also be able to solve higher order ODEs, because an n-order differential equation can be transformed into n first-order differential equations.

It is worth emphasizing that on top of being able to surpass PINN methods, the concept of GAT method is not confined to just being an ODE solver. Because PINN methods choose to incorporate physical knowledge through manipulating loss functions, the physical knowledge must be quantitative. However, the GAT method can even handle qualitative physical knowledge with unknown parameters, as has been demonstrated by its applications in sensors calibration[48,49]. The GAT method characterizes an interaction process between machine learning models and analytical models. The knowledge contained in experiment data and analytical models can both be incomplete on their own, but the GAT method is able to aggregate all the knowledge together to reach an optimal result.

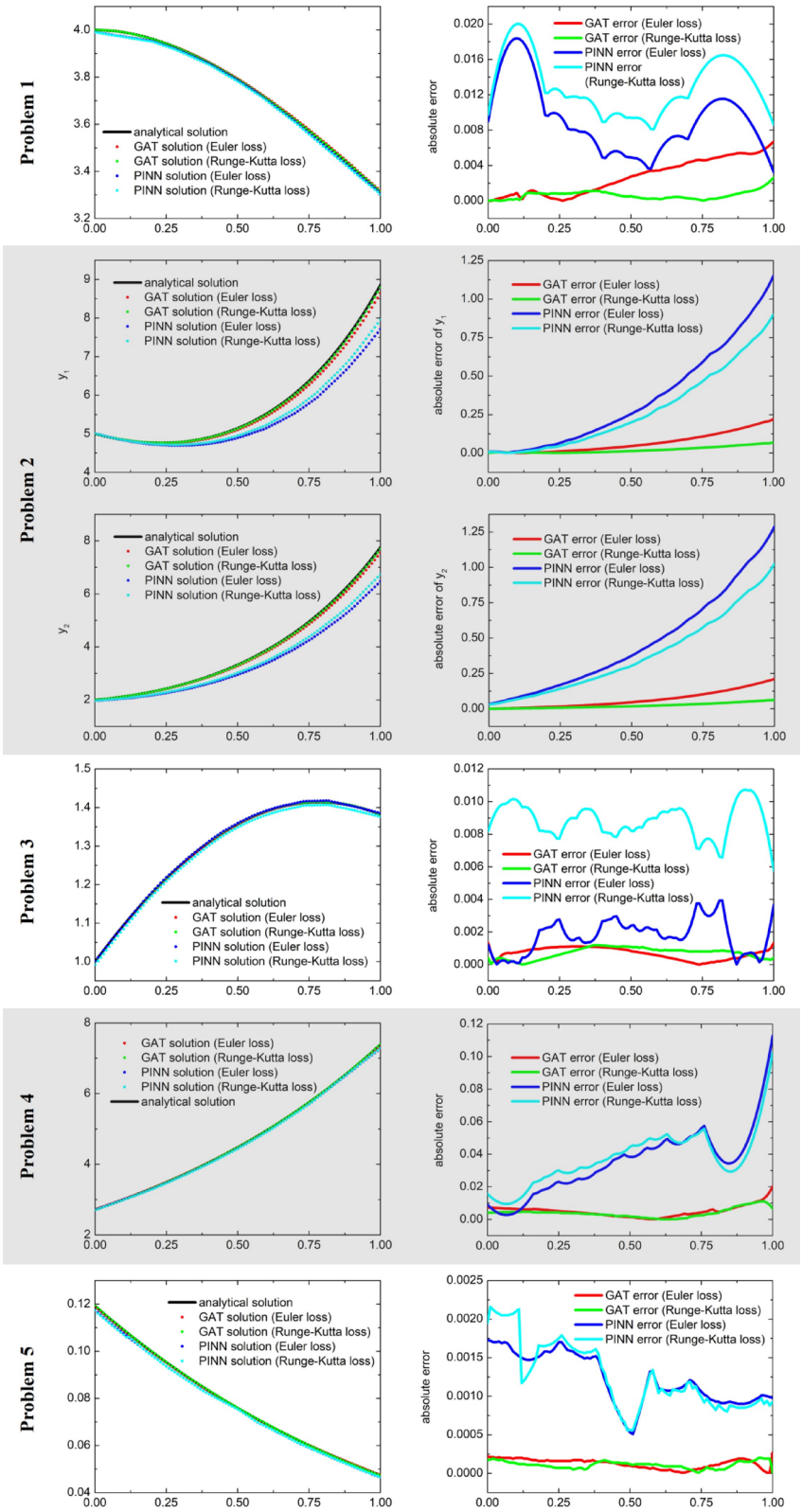

Fig. 5. Results of the ODE examples solved by both the GAT method and PINN method. The numerical solutions are drawn on the left, in comparison to every analytical solution. The absolute errors of numerical solutions with respect to analytical solutions are drawn on the right. The Euler and Runge–Kutta loss functions are employed in both methods for comparison.

Error bound of the GAT method for initial-constraint ODE problems

For an ODE problem $\begin{cases} \frac{dy}{dx} = f(x,y) \\ y(0) = Y_0 \end{cases}$, $x \in [0,1]$, the step size is chosen to be h. $n = 1/h$ and $x_i = ih,\ i = 0,1,\ldots,n$. Y_i is used to approximate the value of $y(x_i)$. Assume there is a method F of order m that satisfies $\left| \frac{y(x_i) - y(x_{i-1})}{h} - F[x_i, h, y(x_{i-1})] \right| \leq O(h^m)$. The used numerical method is $Y_i \approx Y_{i-1} + hF(x_i, h, Y_{i-1})$. Consider the loss value given by L2 norm $L = \sum_{i=1}^{n} h \left[\frac{Y_i - Y_{i-1}}{h} - F(x_i, h, Y_{i-1}) \right]^2$. The following theorem can be proven.

Theorem: Let $\{Y_i\}_{i=0}^n$ satisfy $Y_0 = y(0)$ and $L = \sum_{i=1}^{n} h \left[\frac{Y_i - Y_{i-1}}{h} - F(x_i, h, Y_{i-1}) \right]^2$, then $|Y_i - y(x_i)| \leq e^{M_1} M_2 h^m + e^{M_1}\sqrt{L}$, where $Y_i \approx Y_{i-1} + hF(x_i, h, Y_{i-1})$ with F being a method of order m. M_1 and M_2 are two constants.

Corollary: As is known, $m = 1$ for the Euler method, and $m = 4$ for the Runge–Kutta method[50]. Therefore, if the Euler loss function is used in the GAT method, the error bound is $e^{M_1} M_2 h + e^{M_1}\sqrt{L}$. Alternatively, if the Runge–Kutta loss function is used, the error bound is $e^{M_1} M_2 h^4 + e^{M_1}\sqrt{L}$. As $h \to 0$ and $L \to 0$, the error also converges to zero. If the loss value decreases to zero, it can be found that the order of the error bounds of the GAT method stay consistent with the Euler and Runge–Kutta methods.

Proof of the error bound for initial-constraint ODE problems

Let $g_i = \frac{Y_i - Y_{i-1}}{h} - F(x_i, h, Y_{i-1})$ so that

$$Y_i - Y_{i-1} = hF(x_i, h, Y_{i-1}) + hg_i \quad ①$$

Note that

$$y(x_i) - y(x_{i-1}) = hF[x_i, h, y(x_{i-1})] + C_i h^{m+1} \quad ②$$

Let $\varepsilon_i = Y_i - y(x_i)$, ①–② yields

$$\begin{aligned} \varepsilon_i - \varepsilon_{i-1} &= h\{F(x_i, h, Y_{i-1}) - F[x_i, h, y(x_{i-1})]\} - C_i h^{m+1} + hg_i \\ &= hF_y(x_i, h, \omega_i)\varepsilon_{i-1} - C_i h^{m+1} + hg_i \end{aligned} \quad ③$$

Here the mean value theorem $F(x_i, h, Y_{i-1}) - F[x_i, h, y(x_{i-1})] = F_y(x_i, h, \omega_i)\varepsilon_{i-1}$ with $\omega_i \in [y(x_{i-1}), Y_{i-1}]$ is used. Let $\beta_i = F_y(x_i, h, \omega_i)$, ③ can be rewritten as.

$$\varepsilon_i = (1 + h\beta_i)\varepsilon_{i-1} - C_i h^{m+1} + hg_i, \quad i = 1,2,\ldots,n \quad ④$$

Assume $|\beta_i| \leq M_1$ and $|C_i| \leq M_2$ for all $i = 1,2,\cdots,n$, so that

$$\begin{aligned} |\varepsilon_i| &\leq (1 + hM_1)|\varepsilon_{i-1}| + M_2 h^{m+1} + h|g_i| \\ &\leq (1 + hM_1)\left[(1 + hM_1)|\varepsilon_{i-2}| + M_2 h^{m+1} + h|g_{i-1}|\right] + M_2 h^{m+1} + h|g_i| \\ &= M_2 h^{m+1}[1 + (1 + hM_1)] + h[|g_i| + (1 + hM_1)|g_{i-1}|] + (1 + hM_1)^2|\varepsilon_{i-2}| \end{aligned}$$

Continue this process and use $\varepsilon_0 = 0$, the following estimate can be derived.

$$|\varepsilon_i| \leq M_2 h^{m+1} \sum_{j=0}^{i-1} (1 + hM_1)^j + h \sum_{j=0}^{i-1} |g_{i-j}|(1 + hM_1)^j \quad ⑤$$

Next, for any $j = 0,1,\ldots,n-1$, $(1 + hM_1)^j = e^{j\ln(1+hM_1)} \leq e^{jhM_1} \leq e^{M_1}$, ⑤ can be estimated by

$$
\begin{aligned}
|\varepsilon_i| &\le i e^{M_1} M_2 h^{m+1} + e^{M_1} h \sum_{j=1}^{i} |g_j| \\
&\le e^{M_1} M_2 h^m + e^{M_1} h \sqrt{\sum_{j=1}^{i} |g_j|^2} \sqrt{\sum_{j=1}^{i} 1} \qquad (Cauchy\text{'s } inequality) \\
&= e^{M_1} M_2 h^m + e^{M_1} \sqrt{ih} \sqrt{h \sum_{j=1}^{i} |g_j|^2} \\
&\le e^{M_1} M_2 h^m + e^{M_1} \sqrt{L}
\end{aligned}
$$

The proof of error bound for initial-constraint ODE problems is provided as an example to demonstrate the mathematical rigor of the GAT method. In fact, the proof method can also be extended to other constraint types. The problem setting is similar except only the constraint is different. Because the GAT method can always guarantee exact satisfaction of constraints for numerical solutions, various constraints can thereby be transformed into corresponding relationships between each ε_i based on the definition $\varepsilon_i = Y_i - y(x_i)$. For example, initial constraints like the one in Problem 1 can be transformed into $\varepsilon_0 = 0$. Boundary constraints like the one in Problem 3 can be transformed into $\varepsilon_0 = \varepsilon_n$. Integral constraints like the one in Problem 4 can be transformed into $\sum_{i=0}^{n-1} \varepsilon_i + \sum_{i=1}^{n} \varepsilon_i = 0$. If only the relationships between each ε_i can be properly used, the proof of error bound for other constraint types can be derived accordingly.

Application of the GAT method in human balancing evaluation

For a standing person, a coordinate system is established, as shown in Fig. 6. The human body is approximated by three rigid hinged bars, with the hip joints and ankles considered as hinges. When viewed from the back, the trunk and arms are considered as a single bar which always keeps upright. The two legs are considered as two bars which can swing in the xz-plane. The rotation of the two legs in the xz-plane are always the same. When viewed from the right side, the body can swing in the yz-plane as a whole. The rotation of the trunk and the two legs in the yz-plane are always the same. The corresponding dynamic equations describing the human body motion during standing are shown in Eq. (6), with the establishment process of this dynamic model reported in Supplementary Materials Section S2. The right sides of the equations are related to CoP variations during standing. A balance sensor has been developed based on the principle of frustrated total internal reflection[51]. This balance sensor can record high-resolution pressure distribution variations under human feet in video form. The sensor properties between pressure and pixel intensities were calibrated[48,49], so the pressure distribution variations can be quantitatively determined, and from which, CoP variations can be calculated.

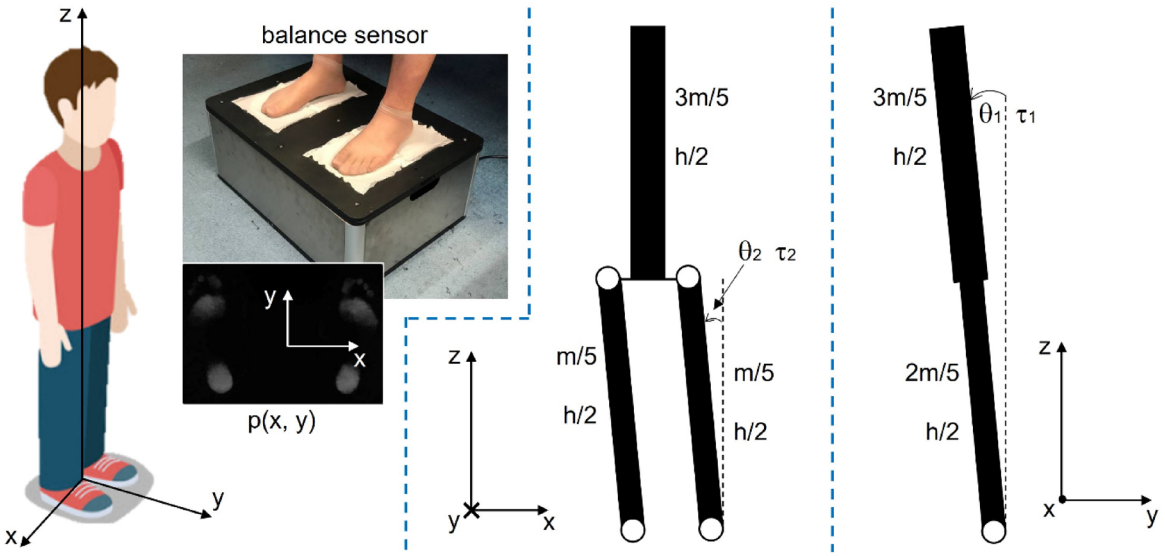

Fig. 6. Mechanical model of human body. The mass of the human body is m. It is approximately considered 3m/5 for the trunk and m/5 for each leg. The human height is h, with approximately h/2 for each leg and the trunk. The whole model has two degrees of freedom. The first is θ_1, indicating body rotation in the yz-plane, with positive direction being anti-clockwise. The second is θ_2, indicating rotation of both legs in the xz-plane, with positive direction being clockwise under right-hand rule. The trunk always keeps upright. The pressure distribution under human feet is represented by p(x, y), which can be measured by the balance sensor.

$$\begin{cases} \frac{23}{60}\frac{h^2}{g}\ddot{\theta}_1 - \frac{31}{60}\frac{h^2}{g}\dot{\theta}_1\dot{\theta}_2\theta_2 - \frac{11}{20}h\theta_1 = COP_y - y_0 \\ \frac{11}{60}\frac{h^2}{g}\ddot{\theta}_2 + \frac{31}{120}\frac{h^2}{g}\dot{\theta}_1^2\theta_2 - \frac{2}{5}h\theta_2 = COP_x - x_0 \end{cases} \tag{6}$$

Here, g is the gravitational acceleration. (x_0, y_0) is defined as the coordinate of the middle point between two ankles, and (CoP_x, CoP_y) is defined as the coordinate of the CoP under feet, namely, $\left(\frac{\iint p(x,y)xdxdy}{\iint p(x,y)dxdy}, \frac{\iint p(x,y)ydxdy}{\iint p(x,y)dxdy}\right)$. Because the swing of human body is around its equilibrium position, (x_0, y_0) can be approximated by the mean value of (CoP_x, CoP_y) across a long period of measurement by the balance sensor[24].

Equation (6) is a nonlinear ODE problem, without analytical solutions. Even though only numerical solutions are considered, there still lacks constraints. Fortunately, hueuristic constraints can be obtained from the knowledge that people do not fall during standing on the balance sensor. This means that θ_1 and θ_2 must always oscillate around 0. The angular velocities $\dot{\theta}_1$ and $\dot{\theta}_2$ must also always oscillate around 0. Therefore, across a long period [0, T], Eq. (7) is approximately satisfied, which can be used as the integral constraints for solving Eq. (6).

$$\begin{cases} \int_0^T \theta_1 dt = 0 \\ \int_0^T \theta_2 dt = 0 \\ \int_0^T \dot{\theta}_1 dt = 0 \\ \int_0^T \dot{\theta}_2 dt = 0 \end{cases} \tag{7}$$

Even though the real human body motion does not exactly satisfy Eq. (7), it can be argued that using Eq. (7) as constraints for solving Eq. (6) can still generate very accurate solutions for most of the measurement period (refer to Supplementary Materials Section S3). This conclusion is further validated by the following experiment.

In order to verify whether the GAT method can solve human body motion from Eq. (6) with constraints being Eq. (7), two experiments were conducted using the balance sensor. In the first experiment, the testee swung front and back deliberately when standing on the balance sensor. In the second experiment, the testee swung left and right during standing. For each experiment, data was collected for 10 s with sampling frequency being 30 Hz. Human body motion is successfully solved with the GAT method. The coordinate of CoG can be calculated from the solved θ_1 and θ_2 by $(CoG_x, CoG_y) = \left(-\frac{2}{5}h\theta_2, -\frac{11}{20}h\theta_1\right)$ according to the human body model. The variations of CoP and CoG in the two experiments are shown in Fig. 7a,b. The trajectories of CoP and CoG variations are shown in Fig. 7c. It can be found that when the testee swung front and back, the solved CoG also has larger amplitude in the direction of front and back. When the testee swung left and right, the solved CoG also oscillates left and right. The solved CoG synchronizes perfectly with CoP, which proves the correctness of the solutions generated by the GAT method. The phenomenon that the excursions of CoP oscillate to either side of the CoG excursions and have higher amplitude and frequency also coincides with the results from other researchers[52,53].

Now that the human body oscillations during standing can be solved from Eq. (6) with Eq. (7) using the GAT method, human balance ability can now be assessed based on CoG variations. Another six experiments were further conducted to verify the balancing evaluation ability of the balance sensor with the help of the GAT method. The six experiments were conducted for three testees in their normal and drunk states, respectively. These testees kept standing still on the balance sensor without any intentional motion this time. Data was also collected for 10 s with sampling frequency being 30 Hz. The GAT method is used to solve their body oscillations, with the results shown in Fig. 8. It is obvious that the amplitudes of body oscillations in drunk states greatly exceed that in normal states for both directions. Quantitatively, the CoG trajectories length of the three testees in drunk states magnified by 9.1, 15.6 and 3.5 times, respectively, compared to their normal states. All these indexes clearly indicate the deterioration of their balance abilities after drinking.

Discussion

This paper proposes a new machine learning framework called the GAT method. It can take advantage of analytical knowledge to enhance machine learning outputs through a positive-sum game strategy. Fully connected neural networks are employed to realize the GAT method and it is demonstrated by solving ODE problems. Two types of loss functions are designed, namely the Euler and Runge–Kutta loss functions. The GAT method successfully extends the ability of classical Euler method, Runge–Kutta method and finite difference method to ODE problems with arbitrary constraints. Traditionally, loss value is only qualitatively related to the accuracy of network predictions. Everyone knows smaller loss value means higher prediction accuracy, but no one knows how small the loss value should be for a desired accuracy. However, for ODE solving problems using the GAT method, at least for initial-constraint problems, the loss value is now proven to be explicitly related

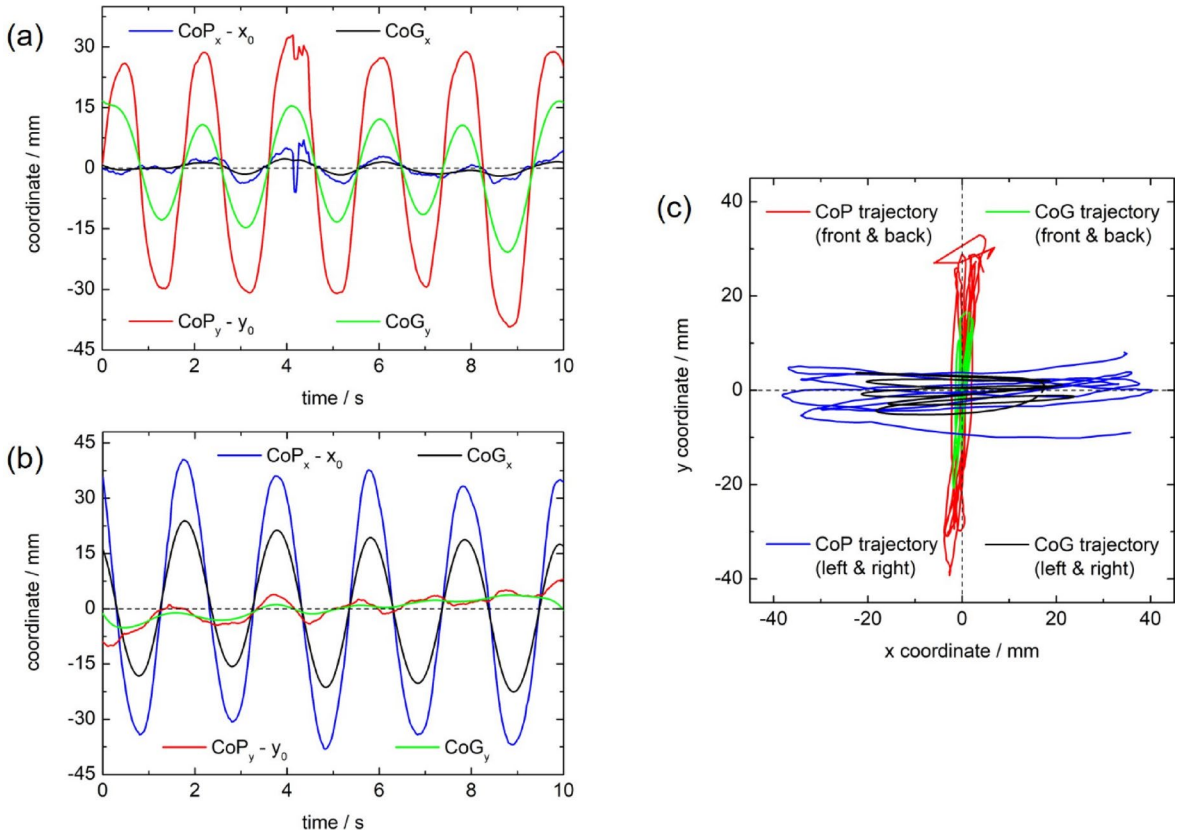

Fig. 7. Human experiment results. (**a**) Variations of CoP and CoG when the testee swung front and back. (**b**) Variations of CoP and CoG when the testee swung left and right. (c) Trajectories of CoP and CoG in the two swing experiments.

to the accuracy of network outputs, thereby providing greater transparency into the network performance and reducing its "black box" nature for these problems.

Furthermore, now the human body oscillations during standing can be solved using the GAT method, which can more convincingly reflect human balance ability. By referring to those CoP-based measures[24] and replacing CoP with CoG, a series of CoG-based measures can be created, which brings a new perspective for human balancing evaluation. Since the CoG variations represent the actual physical oscillations of human body, while CoP can be considered as the control force applied from human body to maintain balance, the variations of CoP and CoG can be compared to further analyze the balance control ability of human body, which is a direct indicator of the physiological ability to keep balance.

The GAT method has seen great success in sensors calibration and solving ODEs. It still shows much broader applications prospect. For example, it is natural to consider solving partial differential equations (PDEs) using the GAT method. If taking the solving process of differential equations as a special case of sequential data generation, the GAT method is potential to be applied to more general sequential data generation tasks. The most difficult problem is how to adapt the GAT method for various application cases. Moreover, currently the GAT method can only be implemented for regression tasks using shallow neural networks. The future research will also target to integrate deep neural networks into the GAT method and extend it to more diverse tasks.

Methods

Five examples have been provided to demonstrate the ODE solving ability of the GAT method for different kinds of constraints. The step size is chosen to be 0.01 for all five examples. For Problem 1, after network convergence, translation is used heuristically to make its initial constraint satisfied. Specifically, C_1 is chosen to be 1 and C_2 is chosen to be $4 - Y_0$. For Problem 2, two neural networks are used to output the numerical solutions of y_1 and y_2, respectively, which are trained jointly. The loss functions are still in the form of Eqs. (3) and (4), except that Y_i and $f\,(ih, Y_i)$ become two-dimensional vectors. After network convergence, translation can also be used heuristically to make its initial constraints satisfied. For Problem 3, after network convergence, translation and scaling are used jointly and heuristically to make its boundary constraint satisfied. Specifically, C_1 is chosen to be $\frac{\sin 1+\cos 1-1}{Y_{100}-Y_0}$ and C_2 is chosen to be $\left(1-\frac{\sin 1+\cos 1-1}{Y_{100}-Y_0}\right)\frac{Y_0+Y_{100}}{2}$. For Problem 4, after network convergence, scaling is used heuristically to make its integral constraint satisfied. Specifically, C_1 is chosen to be

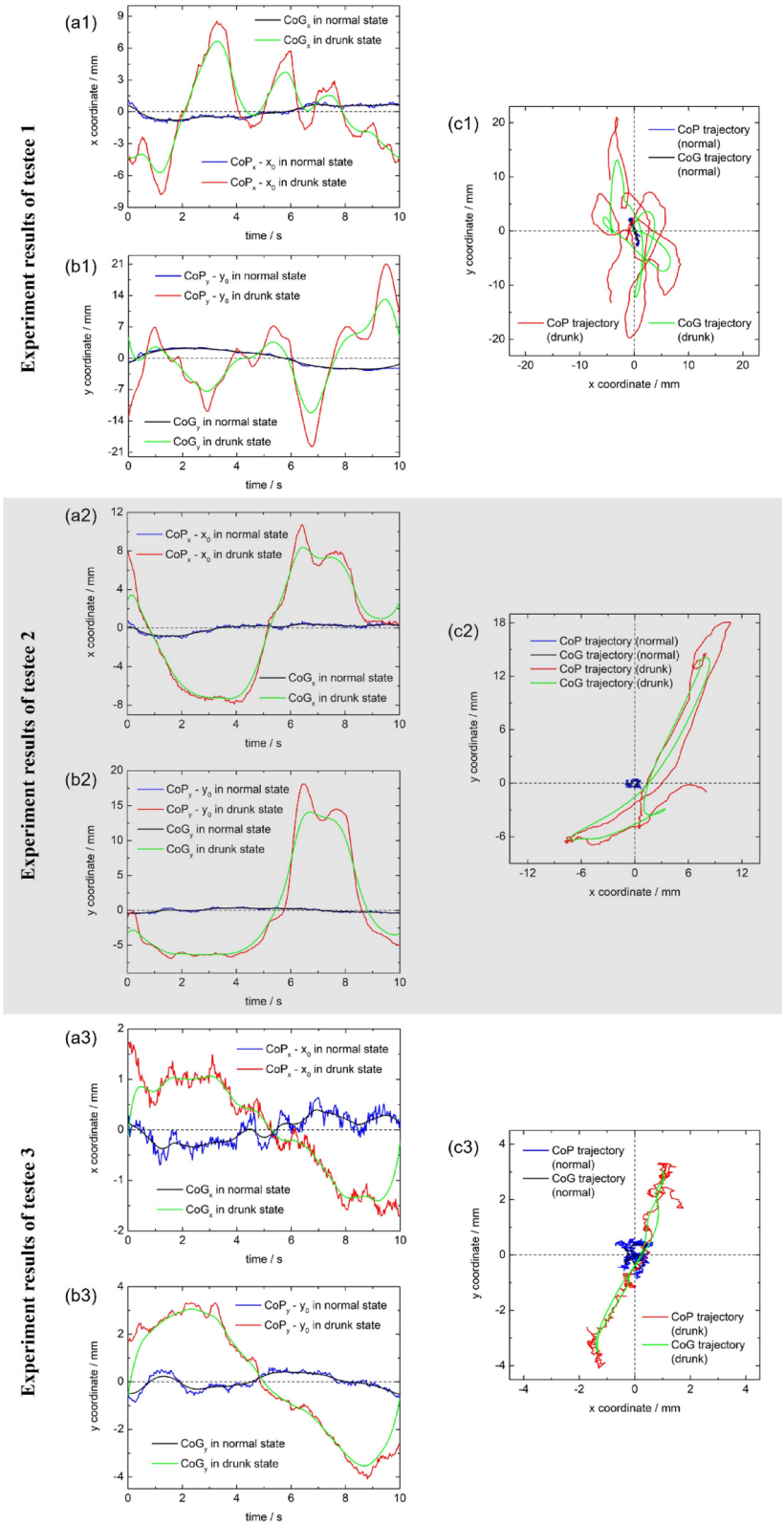

Fig. 8. Human experiment results. (**a1**) (**a2**) (**a3**) Lateral variations of CoP and CoG for testee 1, 2, 3 in normal and drunk states. (**b1**) (**b2**) (**b3**) Longitudinal variations of CoP and CoG for testee 1, 2, 3 in normal and drunk states. (**c1**) (**c2**) (**c3**) Trajectories of CoP and CoG for testee 1, 2, 3 in normal and drunk states.

$\frac{e^2-e}{\left(\sum_{i=0}^{99} Y_i+\sum_{i=1}^{100} Y_i\right)\times\frac{0.01}{2}}$ and C_2 is chosen to be 0. For Problem 5, after network convergence, scaling is also used heuristically to make its nonlinear constraint satisfied. Specifically, C_1 is chosen to be $\frac{\left(\sum_{i=0}^{99}\frac{1}{Y_i}+\sum_{i=1}^{100}\frac{1}{Y_i}\right)\times\frac{0.01}{2}}{e^3-e^2+1}$ and C_2 is chosen to be 0.

Ethics approval was obtained for all human experiments from the Human Research Ethics Committee (HREC) of The University of Hong Kong. The HREC's reference number is EA1904010. All experiments were conducted in accordance with relevant named guidelines and regulations. Informed consents were obtained from all participants.

When using the GAT method to solve human body motion, Eq. (6) is first transformed into four first-order ODEs and four neural networks with 300 hidden nodes for each are employed to output the numerical solutions of θ_1, θ_2, $\dot{\theta}_1$, $\dot{\theta}_2$, respectively, which are then trained jointly using the Runge–Kutta loss function. The step size is chosen to be 1/30. After network convergence, translation is used heuristically to make Eq. (7) satisfied. In order to help the GAT method converge, these networks are initialized with approximate solutions generated by the finite difference method, rather than random initialization. Specifically, Eq. (6) is approximately linearized by directly discarding those nonlinear terms so that its finite difference equations can be linearly solved very easily to get approximate solutions for network initialization.

Data availability

The raw data from human experiments and all related source codes can be found in the GitHub repository (https://github.com/swang-libra/Generative_Adversarial_Tri-Model). There is no restriction on data availability.

Received: 24 July 2024; Accepted: 2 June 2025
Published online: 01 July 2025

References

1. Shumway-Cook, A., Anson, D. & Haller, S. Postural sway biofeedback: its effect on reestablishing stance stability in hemiplegic patients. *Arch. Phys. Med. Rehabil.* **69**, 395–400 (1988).
2. Gribble, P. A. & Hertel, J. Effect of lower-extremity muscle fatigue on postural control. *Arch. Phys. Med. Rehabil.* **85**, 589–592 (2004).
3. Chui, K. K. & Schmitz, T. Examination of sensory function. *Phys. Rehabil.* **87**, 64 (2013).
4. Cho, S.-H. & Choi, Y.-S. The effects of alcohol on static balance in university students. *J. Phys. Ther. Sci.* **24**, 1195–1197 (2012).
5. Mets, M. A., Berend, L. M. D. S. D., Olivier, E. R. V. & Verster, J. C. Effects of hypnotic drugs on body balance and standing steadiness. *Sleep Disord.* **5**, 513–537 (2008).
6. Kim, D., Park, G., Kuo, L.-T. & Park, W. The effects of pain on quadriceps strength, joint proprioception and dynamic balance among women aged 65 to 75 years with knee osteoarthritis. *BMC Geriatr.* **18**, 1–6 (2018).
7. Lee, J.-Y. & Roh, H.-L. Comparison of balance ability between stable and unstable surfaces for chronic stroke patients. *J. Korea Acad. Ind. Cooper. Soc.* **12**, 3587–3593 (2011).
8. Stevens, J. A., Corso, P. S., Finkelstein, E. A. & Miller, T. R. The costs of fatal and non-fatal falls among older adults. *Inj. Prev.* **12**, 290–295 (2006).
9. Dibble, L. E., Christensen, J., Ballard, D. J. & Foreman, K. B. Diagnosis of fall risk in Parkinson disease: An analysis of individual and collective clinical balance test interpretation. *Phys. Ther.* **88**, 323–332 (2008).
10. Hill, K. D., Bernhardt, J., McGann, A. M., Maltese, D. & Berkovits, D. A new test of dynamic standing balance for stroke patients: Reliability, validity and comparison with healthy elderly. *Physiother. Can.* **48**, 257–262 (1996).
11. Carvalho, G. F. et al. Balance impairments in different subgroups of patients with migraine. *Headache J. Head Face Pain* **57**, 363–374 (2017).
12. Havia, M., Kentala, E. & Pyykkö, I. Postural instability in Meniere's disease. *J. Vestib. Res.* **14**, 37–46 (2004).
13. Riemann, B. L. & Guskiewicz, K. M. Effects of mild head injury on postural stability as measured through clinical balance testing. *J. Athlet. Train* **35**, 19–25 (2000).
14. Guskiewicz, K. M. Balance assessment in the management of sport-related concussion. *Clin. Sports Med.* **30**, 89–102 (2011).
15. De Aune, W., Jackson, R. T. & Epstein, C. M. The enhancement of balance testing. *J. Rehabil. Res. Dev.* **30**, 303 (1994).
16. Tzambazis, K. & Stough, C. The SFST and driving ability. Are they related? In *Proceedings International Council on Alcohol, Drugs and Traffic Safety Conference* **2002**, 397–400 (2002).
17. Trojian, T. H. & McKeag, D. B. Single leg balance test to identify risk of ankle sprains. *Br. J. Sports Med.* **40**, 610–613. https://doi.org/10.1136/bjsm.2005.024356 (2006).
18. Ko, B. Sports talent identification and selection in Korea. *Int. J. Appl. Sports Sci.* **26**, 2 (2014).
19. Black, F. O., Wall, C. III., Rockette, H. E. Jr. & Kitch, R. Normal subject postural sway during the Romberg test. *Am. J. Otolaryngol.* **3**, 309–318 (1982).
20. Agrawal, Y., Carey, J. P., Hoffman, H. J., Sklare, D. A. & Schubert, M. C. The modified Romberg balance test: Normative data in US adults. *Otol. Neurotol.* **32**, 1309–1311 (2011).
21. Michikawa, T., Nishiwaki, Y., Takebayashi, T. & Toyama, Y. One-leg standing test for elderly populations. *J. Orthop. Sci.* **14**, 675–685 (2009).
22. Köpke, S. & Meyer, G. D. Tinetti-test–Babylon Im geriatrischen assessment: Babylon in geriatric assessment. *Z. Gerontol. Geriatr.* **39**, 288–291 (2006).
23. Terekhov, Y. Stabilometry as a diagnostic tool in clinical medicine. *Can. Med. Assoc. J.* **115**, 631 (1976).
24. Prieto, T. E., Myklebust, J. B., Hoffmann, R. G., Lovett, E. G. & Myklebust, B. M. Measures of postural steadiness: Differences between healthy young and elderly adults. *IEEE Trans. Biomed. Eng.* **43**, 956–966 (1996).
25. Directions, I. Standardization in platform stabilometry being a part of posturography. *Agressologie* **24**, 321–326 (1983).
26. Hasan, S. S., Robin, D. W. & Shiavi, R. G. Drugs and postural sway: Quantifying balance as a tool to measure drug effects. *IEEE Eng. Med. Biol. Mag.* **11**, 35–41 (1992).
27. Riley, P. O., Mann, R. W. & Hodge, W. A. Modelling of the biomechanics of posture and balance. *J. Biomech.* **23**, 503–506 (1990).
28. Benda, B. J., Riley, P. O. & Krebs, D. E. Biomechanical relationship between center of gravity and center of pressure during standing. *IEEE Trans. Rehabil. Eng.* **2**, 3–10 (1994).
29. Gage, W. H., Winter, D. A., Frank, J. S. & Adkin, A. L. Kinematic and kinetic validity of the inverted pendulum model in quiet standing. *Gait Posture* **19**, 124–132 (2004).
30. Ghez, C. & Fahn, S. The cerebellum. In Kandek, E. R. & Schwartz, J. H. *Principles of Neural Science* 502–552 (1985).

31. Raissi, M., Perdikaris, P. & Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707 (2019).
32. Yu, T. et al. Gradient surgery for multi-task learning. *Adv. Neural. Inf. Process. Syst.* **33**, 5824–5836 (2020).
33. Leake, C. & Mortari, D. Deep theory of functional connections: A new method for estimating the solutions of partial differential equations. *Mach. Learn. Knowl. Extract.* **2**, 37–55 (2020).
34. Wang, S., Teng, Y. & Perdikaris, P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM J. Sci. Comput.* **43**, A3055–A3081 (2021).
35. Mortari, D. The theory of connections: Connecting points. *Mathematics* **5**, 57 (2017).
36. Mortari, D. Least-squares solution of linear differential equations. *Mathematics* **5**, 48 (2017).
37. Mortari, D., Johnston, H. & Smith, L. High accuracy least-squares solutions of nonlinear differential equations. *J. Comput. Appl. Math.* **352**, 293–307 (2019).
38. Schiassi, E. et al. Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations. *Neurocomputing* **457**, 334–356 (2021).
39. Lagaris, I. E., Likas, A. & Fotiadis, D. I. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Trans. Neural Netw.* **9**, 987–1000 (1998).
40. Chiaramonte, M. & Kiener, M. Solving differential equations using neural networks. *Mach. Learn. Project* **1**, 23 (2013).
41. Johnston, H. & Mortari, D. In *Proceedings of the 2018 AAS/AIAA Astrodynamics Specialist Conference, Snowbird, UT, USA* 19–23.
42. De Florio, M., Schiassi, E., D'Ambrosio, A., Mortari, D. & Furfaro, R. Theory of functional connections applied to linear ODEs subject to integral constraints and linear ordinary integro-differential equations. *Math. Comput. Appl.* **26**, 65 (2021).
43. Rout, S., Dwivedi, V. & Srinivasan, B. Numerical approximation in CFD problems using physics informed machine learning. arXiv preprint arXiv:2111.02987 (2021).
44. Goodfellow, I. et al. Generative adversarial networks. *Commun. ACM* **63**, 139–144 (2020).
45. Sobh, T. M. Discrete event dynamic systems: An overview. *Technical Reports (CIS)* 388 (1991).
46. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **2**, 303–314 (1989).
47. Funahashi, K.-I. On the approximate realization of continuous mappings by neural networks. *Neural Netw.* **2**, 183–192 (1989).
48. Wang, S. & Xi, N. Calibration of haptic sensors using transfer learning. *IEEE Sens. J.* **21**, 2003–2012 (2020).
49. Wang, S. & Xi, N. In *2019 IEEE Sensors* 1–4. (IEEE).
50. Sewell, G. *The Numerical Solution of Ordinary and Partial Differential Equations* Vol. 75 (Wiley, 2005).
51. Wang, S. & Xi, N. In *2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)* 807–812. (IEEE).
52. Winter, D. A., Prince, F., Frank, J. S., Powell, C. & Zabjek, K. F. Unified theory regarding A/P and M/L balance in quiet stance. *J. Neurophysiol.* **75**, 2334–2343 (1996).
53. Winter, D. A. Human balance and posture control during standing and walking. *Gait Posture* **3**, 193–214 (1995).

Acknowledgements

The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. T42-717/20-R, C7174-20G, C7100-22G, 17209521, 17212922, 17207323).

Author contributions

Song Wang is mainly responsible for conceptualization, data curation, formal analysis, investigation, methodology, software, visualization and writing the original draft. Ning Xi is partially responsible for the conceptualization and mainly responsible for funding acquisition, project administration, resources, supervision, review and editing of the original draft. Zhengfang Zhou is mainly responsible for the generation of the theorem in "Error bound of the GAT method for initial-constraint ODE problems" section and its corresponding proof in "Proof of the error bound for initial-constraint ODE problems" section.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-025-05320-6.

Correspondence and requests for materials should be addressed to S.W.