Neural Networks With Multidimensional Transfer Functions

C. Tsitouras

***Abstract*—We present a new type of neural network (NN) where the data for the input layer are the value $x \in \Re$, the vector $\mathbf{y} \in \Re^m$ associated to an initial value problem (IVP) with $\mathbf{y}'(x) = f(\mathbf{y}(x))$ and a steplength h. Then the stages of a Runge–Kutta (RK) method with trainable coefficients are used as hidden layers for the integration of the IVP using f as transfer function. We take as output two estimations $\mathbf{y}^*, \hat{\mathbf{y}}^*$ of IVP at the point $x+h$. Training the RK method at some test problems and counting the cost of the method under the coefficients used, we may achieve coefficients that help the method to perform better at a wider class of problems.**

***Index Terms*—Initial value problem (IVP), orbits, oscillators, Runge–Kutta (RK), vector transfer function.**

NOMENCLATURE

x	independent variable
$\mathbf{y}$	$[^1y, ^2y, \cdots, ^my] \in \Re^m$, depended variable
$\mathbf{y}_n$, $\hat{\mathbf{y}}_n$	$\in \Re^m$, approximations of $\mathbf{y}(x)$ at x_n
f	$\Re \times \Re^m \mapsto \Re^m$, function of x,$\mathbf{y}$
$\mathbf{f}$	$[f_1, f_2, \cdots, f_m] \in \Re^{1\times m}$
$\mathbf{b}^T$, $\hat{\mathbf{b}}^T$, $\mathbf{c}$	$\in \Re^s$, RK coefficients
a_{ij}	coefficients of RK matrix $A \in \Re^{s\times s}$
$\mathbf{a}_i$	$[a_{i1}\ a_{i2}\ \ldots\ a_{i,i-1}\ 0\ \ldots\ 0]$, $\mathbf{a}_i^T \in \Re^s$
h_n	stepsize $x_{n+1} - x_n$
i_λ	number of truncation error coefficients of order i
t_{ij}	truncation error coefficients of order i
$\mathbf{d}_{ij}$	coresponding elementary differentials
$T^{(i)}$	$[t_{i1}, t_{i2}, \ldots, t_{i_\lambda}] \in \Re^{i_\lambda}$
$\mathbf{bc}$	scalar product between $\mathbf{b} \in \Re^{1\times s}$ **and** $\mathbf{c} \in \Re^s$
$\mathbf{b}A\mathbf{c}$	dot product
$\mathbf{k}_i$	$\in \Re^m$, output of $i-$st hidden layer
$\Vert\cdot\Vert_\infty, \Vert\cdot\Vert_2$	maximum absolute norm, Euclidean norm
eff	efficiency measure (better small)
ε	objective function to minimize
p	order of RK method

I. INTRODUCTION

EXPLICIT Runge–Kutta (RK) pairs are widely used for the numerical solution of the initial value problem

$$\mathbf{y}' = f(x, \mathbf{y}), \quad \mathbf{y}(x_0) = \mathbf{y}_0 \in \Re^m, \quad x \in [x_0, x_e]$$

where $f : \Re \times \Re^m \mapsto \Re^m$. These pairs are characterized by the extended Butcher tableau [2], [7]

$$\begin{array}{c|c} \mathbf{c} & A \\ \hline & \mathbf{b} \\ & \hat{\mathbf{b}} \end{array}$$

with $\mathbf{b}^T, \hat{\mathbf{b}}^T, \mathbf{c} \in \Re^s$ and $A \in \Re^{s\times s}$ is strictly lower triangular. The procedure that advances the solution from $(x_n, \mathbf{y}_n)$ to $x_{n+1} = x_n + h_n$ computes at each step two approximations $\mathbf{y}_{n+1}, \hat{\mathbf{y}}_{n+1}$ to $\mathbf{y}(x_{n+1})$ of orders p and $p-1$, respectively, given by

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h_n \sum_{i=1}^{s} b_i \mathbf{f}_{ni}$$

and

$$\hat{\mathbf{y}}_{n+1} = \mathbf{y}_n + h_n \sum_{i=1}^{s} \hat{b}_i \mathbf{f}_{ni}$$

with

$$\mathbf{f}_{ni} = f\left(x_n + c_i h_n, \mathbf{y}_n + h_n \sum_{j=1}^{i-1} a_{ij} \mathbf{f}_{nj}\right) \in \Re^m$$

for $i = 1, 2, \ldots, s \geq p$. From this embedded form (called RK$p(p-1)$) we can obtain an estimate

$$u_{n+1} = \Vert \mathbf{y}_{n+1} - \hat{\mathbf{y}}_{n+1} \Vert_\infty$$

of the local truncation error of the $p-1$ order formula. So the step-size control algorithm

$$h_{n+1} = 0.8 \cdot h_n \cdot \left(\frac{\mathrm{TOL}}{u_{n+1}}\right)^{1/p} \tag{1}$$

is in common use, with TOL being the requested tolerance. The above formula is used even if TOL is exceeded by u_{n+1}, but then h_{n+1} is simply the recomputed current step. See [28] for more details on the implementation of these type of step size policies.

Less experienced readers are refered to [14, p. 173], while [2], [7] are classical for the area of numerical analysis of ordinary differential equations.

II. DERIVATION OF RK PAIRS

The derivation of better RK pairs is of continued interest the last 30–40 years [6], [5], [17], [31], [20], [23], [24], [19]. The main framework for the construction of RK pairs is matching Taylor series expansions of $\mathbf{y}(x+h) - \mathbf{y}_{n+1}$ after we have expanded various $\mathbf{f}_{ni}$'s. The final series has the form

$$\begin{aligned} \mathbf{y}(x+h) - \mathbf{y}_{n+1} = {} & h t_{11} \mathbf{d}_{11} + h^2 t_{21} \mathbf{d}_{21} \\ & + h^3 (t_{31} \mathbf{d}_{31} + t_{32} \mathbf{d}_{32}) \\ & + h^4 \begin{pmatrix} t_{41} \mathbf{d}_{41} + t_{42} \mathbf{d}_{42} \\ + t_{43} \mathbf{d}_{43} + t_{44} \mathbf{d}_{44} \end{pmatrix} + \cdots \end{aligned}$$

Manuscript received November 3, 2000; revised June 4, 2001.
The author is with the Department of Applied Mathematical and Physical Sciences, National Technical University of Athens, GR 15780 Athens, Greece (e-mail: tsitoura@math.ntua.gr).
Publisher Item Identifier S 1045-9227(02)00359-4.

In this expansion scalars t_{ij} are the ith order conditions depending explicitly on $A, \mathbf{b}, \mathbf{c}$ and $\mathbf{e} = [1, 1, \ldots, 1]^T \in \Re^s$. e.g.,

$$t_{11} = \mathbf{be} - 1, t_{21} = \mathbf{bc} - \frac{1}{2}$$
$$t_{31} = \frac{1}{2}\mathbf{bc}^2 - \frac{1}{6}, t_{32} = \mathbf{b}A\mathbf{c} - \frac{1}{6},$$
$$t_{41} = \mathbf{bc}^3 - \frac{1}{4}, t_{42} = \mathbf{b}A\mathbf{c}^2 - \frac{1}{12}$$
$$t_{43} = \mathbf{bc}A\mathbf{c} - \frac{1}{8}, t_{44} = \mathbf{b}A^2\mathbf{c} - \frac{1}{24}$$

etc. $\mathbf{d}_{ij}$'s $\in \Re^m$ are elementary differentials of f, e.g., $\mathbf{d}_{31} = (\partial^2 f/\partial \mathbf{y}^2) f^2, \mathbf{d}_{44} = (\partial f/\partial \mathbf{y})^3 f$, etc. See [2, pp. 170–171] for a list of order conditions and the coresponding elementary differentials. So in order to derive a third-order RK method we set $s = 3$ and we have to satisfy four equations $t_{11} = t_{21} = t_{31} = t_{32} = 0$ for the six coefficients $b_1, b_2, b_3, a_{32}, c_2, c_3$ (by default $A\mathbf{e} = \mathbf{c}$, restricts $c_1 = 0, a_{21} = c_2, a_{31} = c_3 - a_{32}$). Once a third-order method is found, nothing can be said about the errors it may produce when applied to some problem since the magnitude of $\mathbf{d}_{ij}, i > 3$ is unpredictable. Some better accuracy is expected if we reduce the norm of the principal truncation error

$$||T^{(4)}||_2 = \sqrt{t_{41}^2 + t_{42}^2 + t_{43}^2 + t_{44}^2}$$

where $T^{(i)} = [t_{i1}, t_{i2}, \ldots, t_{i_\lambda}] \in \Re^{i_\lambda}$, are the conditions for satisfying order i. Values of i_λ for various orders i are given in Table I.

This technique is used widely for derivation of better RK of higher orders too [5], [21], [17], [20], [26]. The order conditions are solved using various simplifying assumptions considering different families of pairs. After we express all the coefficients of the family with respect to some free parameters we continue minimizing $||T^{(p+1)}||_2$ for these parameters. Although minimization of $||T^{(p+1)}||_2$ for a p-order RK method seems the best choice for a general problem, a lot of speculation is raised for problems where it is believed that $\mathbf{d}_{ij}$'s can be handled. Such problems are Hamiltonians [3], orbits [25], periodic [29], [16], [19], Schrödinger [1] and many others.

Unfortunately in most cases analytical consideration of test problems produces complicated algebra and enforces us to proceed with oversimplifications. In other cases we deal with some side properties such as symplectiness, [7, p. 312]. Tenths of symplectic RK methods were appeared last years and no one of them was even competitive to the conventional ones.

An interesting alternative could be the consideration of RK type neural networks (NNs), where the various new families pairs are tested on some model problems to give good predictions for their coefficients.

III. RK Neural Network

The literature combining numerical analysis and especially numerical IVP and NNs is limited. Lagaris *et al.* [13] presented a neural-network approach of solving IVP, but they do not give comparisons with the traditional multistep or RK methods. Multistep methods depending directly and linearly on a set of points give extremely accurate results. In [13] ten points are used and it seems theoretically difficult to compete multistep methods

TABLE I
NUMBER OF ORDER CONDITIONS FOR VARIOUS ORDERS

order i	1	2	3	4	5	6	7	8
i_λ	1	1	2	4	9	20	48	115

with minimizations requiring repeated calls of f and evaluations or even inversions of Jacobians. Recent literature has answered for the most of the claimed there drawbacks of discrete methods. For example RK can be combined with continuous [27] or highly differentiable solution [18]. Perhaps their technique is promising in parallel computers or stiff systems where nonlinear equations has to be solved anyway.

Recently Wang and Lin [32] proposed the so called RK NNs. Their approach is from system identification point of view and they are interested in estimating the function f by an NN. They used a classical RK method [12] of fourth order with constant stepsize because it is easier to prove some theoretical results. From practical consideration we might observe better results when using newer higher order methods with variable stepsize implementation. Perhaps some modification is needed for the learning algorithms reported there, since the simplification of dealing with scalar problems does not work for RK of orders exceeding 3 [14, p. 173].

In this paper we neither indent to solve IVP nor to verify the function f. We are interested in deriving better RK pairs of a prescribed order $p(p-1)$ using s stages. Thus we introduce a feedforward NN consisted from s hidden layers and each one contains m neurons.

INPUT:

$$x, h \in \Re, \mathbf{y} \in \Re^m$$

and the function

$$f : \Re \times \Re^m \mapsto \Re^m.$$

First hidden layer: $\mathbf{k}_1 = f(x, \mathbf{y})$.
Second hidden layer: $\mathbf{k}_2 = f(x + c_2 h, \mathbf{y} + h a_{21} \mathbf{k}_1)$.
Third hidden layer: $\mathbf{k}_3 = f(x + c_3 h, \mathbf{y} + h(a_{31} k_1 + a_{32} \mathbf{k}_2))$.
...
sth hidden layer: $\mathbf{k}_s = f(x + c_s h, \mathbf{y} + h \sum_{j=1}^{i-1} a_{sj} \mathbf{k}_j)$.

OUTPUT:

$$\mathbf{y}^* = \mathbf{y} + h \sum_{i=1}^{s} b_i \mathbf{k}_i.$$

The corresponding drawing of the above NN is shown in Fig. 1.

The NN we introduced is more general than a common multilayer NN since at each neuron acts a different component of the multidimensional function f. This NN can be trained for a variety of inputs $(x_q, h_q, \mathbf{y}_q, f_q), q = 1, 2, \ldots H$ and furnish the proper coefficients $c_2, c_3, \ldots c_s, b_1, \ldots, b_s, a_{32}, \ldots, a_{s,s-1}$ achieving the desirable minimization of the value

$$\varepsilon = \sum_{q=1}^{H} \frac{1}{h_q^{p+1}} ||\mathbf{y}_q^* - \mathbf{y}(x_q + h_q)||^2 = \sum_{q=1}^{H} \frac{e_q^2}{h_q^{p+1}}.$$

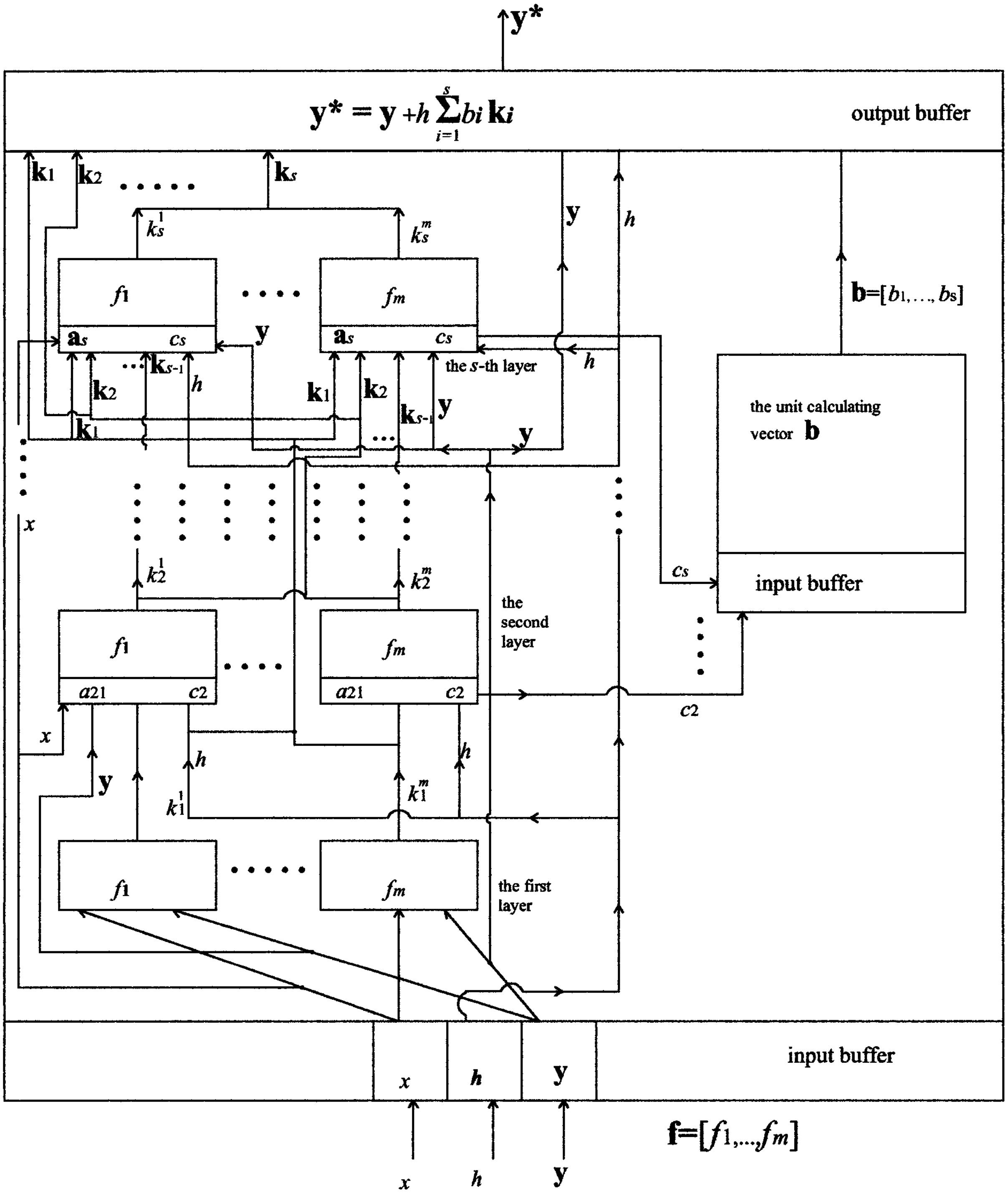

Fig. 1. The neural system calculating R-K coefficients.

It is supposed that f_q's belong to the same parametric type of functions and most of the times are identically the same. If we do not know the analytical solution of the IVP (to be avoided for model problems) we may estimate the true values of the sample points $\mathbf{y}(x_q + h_q)$ by an accurate integration with higher order method at very stringent tolerance.

Let us illustrate now a paradigm for the derivation of the gradient directions that can be used to derive learning algorithms. For the third-order RK method $(p = 3)$ we take $s = 3$. We can easily derive a two parameter general family of coefficients depending on the values c_2, c_3. So we have [2, p. 174]

$$b_1 = \frac{2 - 3c_3 + c_2(-3 + 6c_3)}{6c_2c_3}$$
$$b_2 = \frac{2 - 3c_3}{6c_2^2 - 6c_2c_3}$$
$$b_3 = \frac{2 - 3c_2}{-6c_2c_3 + 6c_3^2}$$

$$a_{32} = \frac{(c_2 - c_3)c_3}{c_2(-2+3c_2)}$$
$$\hat{b}_1 = 1 - \frac{1}{2c_2}$$
$$\hat{b}_2 = \frac{1}{2c_2}$$
$$\hat{b}_3 = 0$$

and we need to evaluate only

$$\frac{\partial e_q^2}{\partial c_2} = 2\left(\mathbf{y}_q^* - \mathbf{y}(x_q + h_q)\right)\frac{\partial\left(\mathbf{y}_q^* - \mathbf{y}(x_q+h_q)\right)}{\partial c_2},$$

with

$$\frac{\partial\left(\mathbf{y}_q^* - \mathbf{y}(x_q+h_q)\right)}{\partial c_2} = \frac{h_q \partial \sum_{j=1}^s b_j \mathbf{k}_j}{\partial c_2}$$

and similarly

$$\frac{\partial e_q^2}{\partial c_3} = 2\left(\mathbf{y}_q^* - \mathbf{y}(x_q + h_q)\right)\frac{h_q \partial \sum_{j=1}^s b_j \mathbf{k}_j}{\partial c_3}.$$

If we expand $\sum_{j=1}^s b_j \mathbf{k}_j$ in power series of h_q, we get

$$\frac{h_q \partial \sum_{j=1}^s b_j \mathbf{k}_j}{\partial c_2} = h_q^4 \cdot \left(\frac{\partial t_{41}}{\partial c_2}\mathbf{d}_{41} + \frac{\partial t_{42}}{\partial c_2}\mathbf{d}_{42}\right) + O\left(h_q^5\right)$$
$$\frac{h_q \partial \sum_{j=1}^s b_j \mathbf{k}_j}{\partial c_3} = h_q^4 \cdot \left(\frac{\partial t_{41}}{\partial c_3}\mathbf{d}_{41} + \frac{\partial t_{43}}{\partial c_3}\mathbf{d}_{43}\right) + O\left(h_q^5\right)$$

where

$$\frac{\partial t_{41}}{\partial c_2} = \left(\frac{1}{18} - \frac{1}{12}c_3\right), \frac{\partial t_{41}}{\partial c_3} = \left(\frac{1}{18} - \frac{1}{12}c_2\right)$$
$$\frac{\partial t_{42}}{\partial c_2} = \frac{1}{12}, \frac{\partial t_{43}}{\partial c_3} = \frac{1}{6}$$
$$\frac{\partial t_{42}}{\partial c_3} = \frac{\partial t_{43}}{\partial c_2} = \frac{\partial t_{44}}{\partial c_2} = \frac{\partial t_{44}}{\partial c_3} = 0$$

and

$$\mathbf{d}_{41} = \frac{\partial^3 f}{\partial \mathbf{y}^3} f^3, \mathbf{d}_{44} = \left(\frac{\partial f}{\partial \mathbf{y}}\right)^3 f$$
$$\mathbf{d}_{42} = \frac{\partial f}{\partial \mathbf{y}} \cdot \frac{\partial^2 f}{\partial \mathbf{y}^2} f^2 \neq \mathbf{d}_{43} = \frac{\partial^2 f}{\partial \mathbf{y}^2} f \cdot \frac{\partial f}{\partial \mathbf{y}} f.$$

Evaluating the elementary differentials $\mathbf{d}$ at $(x_q, \mathbf{y}_q)$ is supposed to be an easy task for a model problem. They are also evaluated once at the beginning of training only. Notice $\mathbf{d}_{42} = \mathbf{d}_{43}$ in the scalar case, but this is not true for systems of equations since $(\partial f/\partial \mathbf{y}), (\partial^2 f/\partial \mathbf{y}^2)$ and f are matrices then.

The technique we used is obligatory since the analytic evaluation of $(\partial\varepsilon/\partial c_2), (\partial\varepsilon/\partial c_3)$ is difficult, while the presented algorithms are valid even for higher order methods. Then there are a little more parameters (e.g., for a fifth-order method there are only four parameters), and the calculations are straightforward following RK theory.

A very interesting realization of the above algorithm may test the reliability of the error estimator u combining many NNs in line, where the output of the first is input for the second just like we proceed with RK steps when integrating an IVP. In this case a small modification is required since the input data must be determined dynamically. The tolerance TOL and the region $[x_0, x_e]$ where we want to test the RK pair determine the whole line of NN. The output of each NN is $\mathbf{y}^*$ and $u^* = h\sum_{i=1}^s (b_i - \hat{b}_i)\mathbf{k}_i$ in order to evaluate a new step h^* based on (1). Then we may proceed to the next NN in line with inputs $x + h, h^*, \mathbf{y}^*$ and of course the transfer function f.

The number of inputs is dynamically determined each time we backpropagate for another training and the value we get as final output is the efficiency of the pair which is to be optimized. It is worth to notice that there might be rejections of some NN from this line if u^* is not less than TOL. The cost of evaluating these NN must be also considered in the final evaluation of the efficiency eff. So here we simply choose

$$(x_{q+1}, h_{q+1}, \mathbf{y}_{q+1}, f_{q+1}) = \left(x_q + h_q^*, h_q^*, \mathbf{y}_q + h\sum_{i=1}^s b_i \mathbf{k}_i, f_q\right).$$

The cost of the pair for the integration we made is not simply the true global error ge at grid points but it has to involve the total number of f-evaluations N_f needed. A measure of the efficiency is presented in [22]

$$\text{eff} = N_f \cdot \left(\frac{ge}{\text{TOL}}\right)^{1/p}.$$

The smaller the eff the better the efficiency, so we indent to minimize eff for some test IVPs hoping for a better performance in a wider class of problems.

In the latter case the values of the form $(\partial\varepsilon/\partial c_2)$ and $(\partial\varepsilon/\partial c_3)$ of the paradigm concerning third-order RK methods are difficult to evaluate because the various elementary differentials are changing every time. Denoting by $\mathbf{d}_{41}^{[q]}$ the corresponding elementary differential of the q-th input we calculate

$$\frac{\partial\varepsilon}{\partial c_2} = \sum_{q=1}^H \frac{1}{h_q^4} 2\left(\mathbf{y}_q^* - \mathbf{y}(x_q + h_q)\right)\frac{h_q \partial \sum_{j=1}^s b_j \mathbf{k}_j}{\partial c_2}$$
$$= 2\sum_{q=1}^H \left(\mathbf{y}_q^* - \mathbf{y}(x_q + h_q)\right)$$
$$\times \left(\frac{\partial t_{41}}{\partial c_2}\mathbf{d}_{41}^{[q]} + \frac{\partial t_{42}}{\partial c_2}\mathbf{d}_{42}^{[q]} + O(h_q)\right).$$

Since $t_{4i}, i = 1,2,3,4$ that depend on the RK method remain constant during the integration, we may admit some averaging value of the form

$$\frac{\partial\varepsilon}{\partial c_2} \approx \frac{\partial\|T^{(4)}\|_2}{\partial c_2}, \frac{\partial\varepsilon}{\partial c_3} \approx \frac{\partial\|T^{(4)}\|_2}{\partial c_3}$$

with

$$\|T^{(4)}\|_2 = \frac{1}{5184} \cdot \begin{pmatrix} 9 + 9\cdot(1-2c_2)^2 + 9\cdot(3-4c_3)^2 \\ +(3 - 4c_3 + c_2\cdot(-4+6c_3))^2 \end{pmatrix}.$$

This means that we follow the direction of minimizing the principal truncation error coefficients of the RK formula hoping that will lead us to a better choice.

In alternative we may evaluate

$$\frac{\partial\varepsilon}{\partial c_2} \approx \frac{\partial\|T^{(4)}\|_2}{\partial c_2} + \bar{h}\frac{\partial\|T^{(5)}\|_2}{\partial c_2}$$
$$\frac{\partial\varepsilon}{\partial c_3} \approx \frac{\partial\|T^{(4)}\|_2}{\partial c_3} + \bar{h}\frac{\partial\|T^{(5)}\|_2}{\partial c_3}$$

with $\bar{h}$ the average step size.

IV. Numerical Results for Pairs of Orders 5(4)

The initial idea of producing coefficients without satisfying any order condition was a total failure. Actually our technique has reason in existing families where most of the coefficients are expressed with respect to a small number of them served as free parameters. Here, we will try to produce more efficient RK5(4) pairs.

Pairs of orders 5(4) are the most popular ones and can easily found in the literature. Matlab [15], has included the standard function *ode45*, which uses perhaps the most famous such pair DP5(4) due to Dormand and Prince [5]. For this pair we have $s = 7$ but the last stage of each step may reused as the first stage in the next step (FSAL). So only six evaluations of the transfer function are wasted every step since the last layer of the some NN in line is placed again as first layer for the next NN. We name this property last layer as first (LLAF).

In [17], we may find an algorithm where all the coefficients of a 5(4) pair may expressed explicitly based on the parameters c_2, c_3, c_4, c_5 and $\hat{b}_7$. Reproducing a learning algorithm here we have to evaluate the 20 truncation error values $t_{61} = \mathbf{b}\mathbf{c}^5 - 1/6, \ldots, t_{6,20} = \mathbf{b}A^4\mathbf{c} - 1/720$ and differentiate them with the free parameters.

A. Kepler Orbital Problem

The problem has the form

$$\begin{aligned} {}^1y' &= {}^3y \\ {}^2y' &= {}^4y \\ {}^3y' &= -\frac{{}^1y}{\sqrt{{}^1y^2 + {}^2y^2}^3} \\ {}^4y' &= -\frac{{}^2y}{\sqrt{{}^1y^2 + {}^2y^2}^3} \end{aligned}$$

where $x \geq 0, \mathbf{y}(0) = [1-\epsilon, 0, 0, \sqrt{(1+\epsilon)/(1-\epsilon)}]^T$, with ϵ the eccentricity of the orbit. The left superscript denotes the component of $\mathbf{y}$. It is known that the energy

$$E = \frac{1}{2}({}^3y^2 + {}^4y^2) - \frac{1}{\sqrt{{}^1y^2 + {}^2y^2}} = -\frac{1}{2}$$

remains constant since the problem is conservative. So we considered as ge the maximum observed value of $|E + 1/2|$ over all the grid points.

We applied the NN described above with inputs $\epsilon = 0.5, x_e = 2\pi$ and TOL $= 10^{-6}$. Simulating NN with DP5(4) coefficients we found $\mathrm{eff}_{\mathrm{DP}} = 1059.5$. Then we proceed training the NN line obtaining finally $\mathrm{eff} = 620.8$ for the new method which means that it was almost 72% less expensive than DP5(4). The optimal efficiency was found for

$$c_2 = \frac{5}{22}, c_3 = \frac{133}{400}, c_4 = \frac{33}{34}, c_5 = \frac{174}{175}, \hat{b}_7 = \frac{1}{18}.$$

The coefficients of the new pair NEW5(4)a can be derived applying the algorithm in [17], and are listed in Table II.

We test the result with simulation for a wide class of regions, eccentricities, and tolerances. We include here three simulations.

a) $\epsilon = 0.1, 0.2, \ldots, 0.9, x \in [0, 20], \mathrm{TOL} = 10^{-3}, 10^{-4}, \ldots, 10^{-9}$. We recorded the 63 values of eff and found that in average $\mathrm{eff}_{\mathrm{DP}} \approx 1527.6$ while for the new pair we have $\mathrm{eff} \approx 1150.1$, reducing the cost 33%.

b) $\epsilon = 0.1, 0.2, \ldots, 0.9, x \in [0, 40], \mathrm{TOL} = 10^{-3}, 10^{-4}, \ldots, 10^{-9}$. We recorded again the 63 values of eff and found that in average $\mathrm{eff}_{\mathrm{DP}} \approx 3277.6$ while for the new pair we have $\mathrm{eff} \approx 2426.8$, reducing the cost 35%.

c) $\epsilon = 0.05, 0.15, \ldots, 0.95, x \in [0, 20], \mathrm{TOL} = 10^{-3}, 10^{-4}, \ldots, 10^{-9}$. We recorded again the 70 values of eff and found that in average $\mathrm{eff}_{\mathrm{DP}} \approx 1617.6$ while for the new pair we have $\mathrm{eff} \approx 1206.4$, reducing the cost 34%.

So the pair we derived optimizing a simple integration has some hidden property that helps it to perform much better than conventional pairs for every test in the family of two body (Kepler) problems. We can not obtain this hidden property with a simple minimization of $||T^{(6)}||_2$. In [20] a new optimal RK pair of orders 6(5) was found with minimized truncation error, but when tested on Kepler problem it was 30% less efficient than older methods for some eccentricities, while there were eccentricities where it was almost 50% more efficient. Here the observed deviation from the mean value of efficiency is very small.

Gaining in average 10–15% for methods of the same order is the usual improvement of RK pairs through the years passed [6], [5], [17]. Gaining here more than 30% is worth mention.

B. Periodic Problems

When dealing with periodic or oscillatory problems it is constructive to consider the test equation

$$y' = \sqrt{-1}\omega y, y(x_0) = y_0 \in \Re^m, x \in [x_0, x_e], \omega \in \Re.$$

The theoretical solution of this problem is

$$y(x) = (\cos\omega x + \sqrt{-1}\sin\omega x) \cdot y_0.$$

In many applications we are interested on the argument of the solution which is the phase of the angle ωx, i.e., $\arg y = \tan\omega x$, [29], [19]. Monitoring this value through the integration we ensure that we remain in phase. So we considered as ge the maximum observed value of $|\tan\omega x - \arg y|$ over all the grid points.

We applied the NN described above with inputs $\omega = 1, x_e = 2\pi$ and TOL $= 10^{-6}$. Simulating NN with DP5(4) coefficients we found $\mathrm{eff}_{\mathrm{DP}} = 140.9$. Then we proceed training the NN line obtaining finally $\mathrm{eff} = 15.9$ for the new method which means that it was almost 800% less expensive than DP5(4). The optimal efficiency was found for

$$c_2 = \frac{157}{792}, c_3 = \frac{457}{1393}, c_4 = \frac{171}{215}, c_5 = \frac{1104}{1241}, \hat{b}_7 = \frac{1}{20}.$$

The coefficients of the new pair NEW5(4)b can be derived applying the algorithm in [17], and are listed in Table III.

We test the result with simulation for a wide class of regions, frequencies, and tolerances. We include here three simulations.

a) $\omega = 1, 2, \ldots, 10, x \in [0, 2\pi], \mathrm{TOL} = 10^{-3}, 10^{-4}, \ldots, 10^{-9}$. We recorded the 70 values of eff and found

TABLE II
COEFFICIENTS OF NEW5(4)A (ACCURATE AT 16 DIGITS)

0							
$\frac{5}{22}$	$\frac{5}{22}$						
$\frac{133}{400}$	$\frac{71421}{800000}$	$\frac{194579}{800000}$					
$\frac{33}{44}$	$\frac{592292276}{418440609}$	$-\frac{76869243}{13068580}$	$\frac{472517100}{86906057}$				
$\frac{174}{175}$	$\frac{5826402054}{3540418421}$	$-\frac{1802051930}{263259491}$	$\frac{2198919976}{354178997}$	$-\frac{9453649}{641339334}$			
1	$\frac{344811583}{202675472}$	$-\frac{209300311}{29580530}$	$\frac{1269016359}{198536683}$	$-\frac{9310471}{918210297}$	$-\frac{2741836}{370671635}$		
1	$\frac{12085}{120582}$	0	$\frac{356137626}{708647483}$	$\frac{766889822}{60568101}$	$-\frac{3676531250}{68192253}$	$\frac{22241}{534}$	
b	$\frac{12085}{120582}$	0	$\frac{356137626}{708647483}$	$\frac{766889822}{60568101}$	$-\frac{3676531250}{68192253}$	$\frac{22241}{534}$	
$\hat{b}$	$\frac{31681789}{351068635}$	0	$\frac{86354954}{160893035}$	$\frac{1469236840}{280170657}$	$-\frac{2761387937}{153579530}$	$\frac{1705684531}{130667931}$	$\frac{1}{18}$

TABLE III
COEFFICIENTS OF NEW5(4)B (ACCURATE AT 16 DIGITS)

0							
$\frac{157}{792}$	$\frac{157}{792}$						
$\frac{457}{1393}$	$\frac{17242153}{304650493}$	$\frac{82704204}{304650493}$					
$\frac{171}{215}$	$\frac{164974215}{165204568}$	$-\frac{1126675043}{381969438}$	$\frac{507877261}{184925378}$				
$\frac{1104}{1241}$	$\frac{5895014254}{341488313}$	$-\frac{1061597689}{19324293}$	$\frac{6801715378}{163955985}$	$-\frac{1238630440}{423868997}$			
1	$\frac{185064709}{85647086}$	$-\frac{3076588154}{474989365}$	$\frac{1084016133}{207648103}$	$\frac{49457902}{332207679}$	$-\frac{4296453}{81166066}$		
1	$\frac{5577289}{54489024}$	0	$\frac{125285369}{263408437}$	$\frac{66135053}{167483031}$	$-\frac{9418625}{376769549}$	$\frac{588307}{11284416}$	
b	$\frac{5577289}{54489024}$	0	$\frac{125285369}{263408437}$	$\frac{66135053}{167483031}$	$-\frac{9418625}{376769549}$	$\frac{588307}{11284416}$	
$\hat{b}$	$\frac{31349362}{306706991}$	0	$\frac{47609631}{99976703}$	$\frac{99593296}{255216031}$	$-\frac{3289939}{176951650}$	$-\frac{6995}{118486508}$	$\frac{1}{20}$

that for the new pair we have in average $\mathrm{eff} \approx 411.7$ while for DP5(4) we have $\mathrm{eff}_{\mathrm{DP}} \approx 1419.6$, and this means that it is 245% more expensive.

b) $\omega = 1, 2, \ldots, 10, x \in [0, 10\pi]$, TOL $= 10^{-3}, 10^{-4}, \ldots, 10^{-9}$. We recorded again the 70 values of eff and found that in average for the new pair we have $\mathrm{eff} \approx 2826$, while for DP5(4) we have $\mathrm{eff}_{\mathrm{DP}} \approx 10\,622$ i.e., 275% more expensive.

c) $\omega = 2, 4, \ldots, 20, x \in [0, 10\pi]$, TOL $= 10^{-3}, 10^{-4}, \ldots, 10^{-9}$. We recorded again the 70 values of eff and found that in average for the new pair we have $\mathrm{eff} \approx 6526$, while for DP5(4) we have $\mathrm{eff}_{\mathrm{DP}} \approx 24\,461$ i.e., again about 275% more expensive.

So we again derived a pair optimizing a simple integration of the test periodic problem. We did not get better results by using a more analytical approach [19]. Besides the NN technique may expand to inhomogeneous problems where the analysis is very difficult, e.g., [30]

$$y'' = -\omega^2 y + c e^{\sqrt{-1}\delta \cdot t}, \omega^2 \neq \delta^2, c, \omega \in \Re \setminus \{0\}$$

or even to nonlinear ones.

V. CONCLUSION

A new type of artificial NN design was proposed in this paper for the derivation of the coefficients of Runge-Kutta pairs for the numerical solution of initial value problems.

The system described in this paper is more general than multilayer NN considered in the classic articles on approximation problems [4] and [8]–[11]. We also show an example of a system for which is not true that one hidden layer is sufficient to solve the problem.

The main features of the new proposal are:

1) vector transfer functions;
2) dynamic changing of input data;
3) LLAF;
4) backpropagating in the direction of minimizing principal truncation error of the numerical scheme.

A first interesting result is that we can produce better IVP solvers for various types of problems. This was achieved by training them in a test problem and optimizing their coefficients.

ACKNOWLEDGMENT

The author wishes to thank the anonymous referee for drawing Fig. 1.

REFERENCES

[1] G. Avdelas, T. E. Simos, and J. Vigo-Aguiar, "An embedded exponentially fitted Runge-Kutta method for the numerical solution of the Schrödinger equation and related periodic initial value problems," *Comput. Phys. Commun.*, vol. 131, pp. 52–67, 2000.

[2] J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations*. New York: Wiley, 1987.

[3] M. P. Calvo and J. M. Sanz-Serna, "High order symplectic Runge-Kutta-Nyström methods," *SIAM J. Sci. Comput.*, vol. 14, pp. 107–114, 1993.

[4] G. Cybenko, "Approximation by superpositions of a sigmoid function," *Math. Contr. Signals, Syst.*, vol. 2, pp. 303–314, 1989.

[5] J. R. Dormand and P. J. Prince, "A family of embedded Runge-Kutta formulae," *J. Comput. Appl. Math.*, vol. 6, pp. 19–26, 1980.

[6] E. Fehlberg, "Low order classical runge-kutta formulas with stepsize control and their application to some heat-transfer problems,", TR R-287, NASA, 1969.

[7] E. Hairer, S. P. Nrsett, and G. Wanner, *Solving Ordinary Differential Equations I*, 2nd ed. Berlin, Germany: Springer-Verlag, 1993.

[8] R. Hecht-Nielsen, "Counterpropagation networks," in *Proc. Int. Conf. Neural Networks*. New York, 1987, pp. 19–32.

[9] ——, "Kolmogorov's mapping neural network existence theorem," in *Proc. Int. Conf. Neural Networks*. New York, 1987, pp. 11–13.

[10] ——, *Neurocomputing*. Reading, MA: Addison-Wesley, 1990.

[11] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.

[12] W. Kutta, "Beitrag zur nahenrungsweisen integration totaler differentialgleichungen," *ZAMP*, vol. 46, pp. 435–453, 1901.

[13] I. E. Lagaris, A. Likas, and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Trans. Neural Networks*, vol. 9, pp. 987–1000, 1998.

[14] J. D. Lambert, *Numerical Methods for Ordinary Differential Equations*. New York: Wiley, 1991.

[15] *Matlab*, 1999.

[16] G. Papageorgiou, C. Tsitouras, and S. N. Papakostas, "Runge-Kutta pairs for periodic initial value problems," *Computing*, vol. 51, pp. 151–163, 1993.

[17] S. N. Papakostas and G. Papageorgiou, "A family of fifth order Runge-Kutta pairs," *Math. Comput.*, vol. 65, pp. 1165–1181, 1996.

[18] S. N. Papakostas and C. Tsitouras, "Highly continuous interpolants for one-step ODE solvers and their application to Runge-Kutta methods," *SIAM J. Numer. Anal.*, vol. 34, pp. 22–47, 1997.

[19] ——, "High algebraic order, high phase-lag order Runge-Kutta and Nyström pairs," *SIAM J. Sci. Comput.*, vol. 21, pp. 747–763, 1999.

[20] S. N. Papakostas, C. Tsitouras, and G. Papageorgiou, "A general family of explicit Runge-Kutta pairs of orders 6(5)," *SIAM J. Numer. Anal.*, vol. 33, pp. 917–936, 1996.

[21] P. J. Prince and J. R. Dormand, "High order embedded Runge-Kutta formulae," *J. Comput. Appl. Math.*, vol. 7, pp. 67–75, 1981.

[22] L. F. Shampine, "Some practical Runge–Kutta formulas," *Math. Comput.*, vol. 46, pp. 135–150, 1986.

[23] C. Tsitouras, "High order zero dissipative Runge-Kutta-Nyström methods," *J. Comput. Appl. Math.*, vol. 95, pp. 157–161, 1998.

[24] ——, "High algebraic order, high dissipation order Runge-Kutta methods," *Comput. Math. Applicat.*, vol. 36, pp. 31–36, 1999.

[25] ——, "A tenth-order symplectic Runge-Kutta-Nyström method," *Cele. Mech. Dyn. Astron.*, vol. 74, pp. 223–230, 1999.

[26] ——, "Optimal Runge-Kutta pairs of orders 9(8)," *Appl. Numer. Math.*, vol. 38, pp. 123–134, 2001.

[27] C. Tsitouras and G. Papageorgiou, "Runge Kutta interpolants based on values from two successive integration steps," *Computing*, vol. 43, pp. 255–266, 1990.

[28] C. Tsitouras and S. N. Papakostas, "Cheap error estimation for Runge-Kutta pairs," *SIAM J. Sci. Comput.*, vol. 20, pp. 2067–2088, 1999.

[29] P. J. Van der Houwen and B. P. Sommeijer, "Explicit Runge-Kutta (-Nystrom) methods with reduced phase-errors for computing oscillating solutions," *SIAM J. Numer. Anal.*, vol. 24, pp. 407–442, 1987.

[30] P. J. Van der Houwen, B. P. Sommeijer, K. Strehmel, and R. Weiner, "On the numerical integration of second order initial value problems with periodic forcing function," *Computing*, vol. 37, pp. 195–218, 1986.

[31] J. H. Verner, "High order explicit Runge-Kutta pairs with low stage order," *Appl. Numer. Math.*, vol. 22, pp. 345–357, 1996.

[32] Y.-J. Wang and C.-T. Lin, "Runge-Kutta neural network for identification of dynamical systems in high accuracy," *IEEE Trans. Neural Networks*, vol. 9, pp. 294–307, 1998.