文章编号:1000-0887(2019)02-0115-12

几类微分-代数方程的神经网络求解法*

杨 钊[1], 兰 钧[2], 吴勇军[1]

(1. 上海交通大学 工程力学系, 上海 200240;
2. 卫宁健康科技集团股份有限公司, 上海 200072)

摘要: 在非线性科学中,寻求微分方程的近似解析解一直是重要的研究课题和研究热点.利用人工神经网络原理,结合最优化方法,研究了几类微分-代数方程的近似解析解,包括指标 1,2,3 型 Hessenberg 方程及指标 3 型 Euler-Lagrange 方程,得到了方程近似解析解的表达式.通过与精确解或 Runge-Kutta(龙格-库塔)数值计算结果对比,表明神经网络方法的结果有很高的精度.

关 键 词: 人工神经网络; 微分-代数方程; 近似解析解; 最优化方法

中图分类号: O193; O322 **文献标志码:** A DOI: 10.21656/1000-0887.390122

引 言

在科学与工程中常常会碰到微分-代数方程(differential-algebraic equation, DAE).例如,在线性或非线性电路、化学工程、电力工程、机械工程、多体动力学、社会经济系统等领域中,微分-代数方程均有广泛的应用.随着科学的发展,对微分-代数方程的研究也越来越多.目前,对微分-代数方程的研究大多集中于数值计算方面,即如何用数值计算方法求解微分-代数方程[1-4],而定性的理论分析方面的研究还不是很多[5-7].众所周知,对常微分方程(ODE)的求解,已经发展了许多非常成熟及有效的数值计算方法.对于微分-代数方程,成熟的数值计算方法相对较少.1971 年,Gear 提出微分-代数方程的概念及 BDF(backward differentiation formulae)求解法[8];之后,Cash 进一步发展了这种方法,提出了修正扩展 BDF(modified extended backward differentiation formulae)法[9].1982 年,Petzold 指出微分-代数方程不同于常微分方程[10];之后,Hairer 等提出了隐式 Runge-Kutta 法[11].2006 年,Guzel 等应用 Padé 近似方法求解指标 2 的微分-代数方程,将精度进一步提高[12].在文献[13]中,Karta 和 Çelik 应用变分迭代法求解指标 1、指标 2 和高指标的微分代数方程,表现出很好的优化、拟合能力.新近发展的求解微分-代数方程的方法还有伪谱法(pseudo-spectral method)[14]、Adomian 分解法(Adomian decomposition method)[15]、指数积分法(exponential integration method)[16]、李群法(Lie-group method)[17]等.

微分-代数方程的一个重要应用领域是非完整力学,例如多体动力学[2,18].对多体动力学中的微分-代数方程已有丰富的研究.在文献[19]中,潘振宽等对多体系统动力学微分-代数方

* **收稿日期:** 2018-04-18; **修订日期:** 2018-11-11
基金项目: 国家自然科学基金(11772293;11272201)
作者简介: 杨钊(1995—),男,硕士生;
吴勇军(1978—),男,副研究员(通讯作者. E-mail: yj.wu@sjtu.edu.cn).

程的数值积分方法和新的理论做了综述和介绍.2002 年,赵维加、潘振宽[20]讨论了存在相关约束 Euler-Lagrange 方程解的存在唯一性条件,给出了解的存在唯一性定理.2004 年,吴永[21]利用辛算法并结合 Runge-Kutta 法,求解了 Euler-Lagrange 方程.2008 年,王加霞[22]利用谱积分和 Newton-Krylov 方法提出了对 Euler-Lagrange 方程的高精度计算方法,可以达到任意精度并具有 A-稳定性.2010 年,刘颖、马建敏[23]将 ε 嵌入处理方式与隐式 Runge-Kutta 法相结合,提出了直接满足位移约束条件的多体系统动力学方程的无违约算法,避免了约束违约问题.2017 年,马秀腾等[24]采用 HHT 直接积分方法求解降指标后的指标 2 运动方程,针对违约问题,采用基于 Moore-Penrose 广义逆理论的违约校正方法,消除位置约束方程的违约,在规模小时具有更快的运算速度.

以上介绍的求解代数-微分方程的方法属于传统数值方法.近年来,受自然界启发(nature inspired),基于神经网络、生物进化、生物遗传等模型的智能计算方法迅速发展[25-28].利用智能计算方法,可以求解常(偏)微分方程[26-28].其最大的优点是能获得微分方程的近似精确解的显式表达式,为后续分析带来便利.

本文将求解常微分方程的人工神经网络法加以推广,用以研究几类典型的微分-代数方程的近似解析解,包括 Hessenberg 型、Euler-Lagrange 型,为微分-代数系统的进一步理论分析提供支持.

1 人工神经网络法的求解步骤

这里用一个一般形式的微分方程来描述该方法:

$$F(\boldsymbol{x},\Psi(\boldsymbol{x}),\nabla\Psi(\boldsymbol{x}),\nabla^2\Psi(\boldsymbol{x}))=0, \tag{1}$$

$$f(\boldsymbol{x}_0,\Psi(\boldsymbol{x}_0))=0, \tag{2}$$

其中 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^{\mathrm{T}}\in R^n$ 表示定义域为 D 的系统自变量.$\Psi(\boldsymbol{x})$ 是待求值,$\boldsymbol{x}_0$ 是初始或边界值.方程(2)表示微分方程满足的初始或边界条件.

为了获得上述微分方程的解,假设近似解 $\Psi_{\mathrm{a}}(\boldsymbol{x},\boldsymbol{p})$ 满足上述方程,其中 $\boldsymbol{p}$ 为一组可调节的待定参数,那么所求问题可以转换为

$$E(\boldsymbol{p})=\min_{\boldsymbol{p}}\sum_{\boldsymbol{x}\in D_i}[F(x_i,\Psi_{\mathrm{a}}(x_i,\boldsymbol{p}),\nabla\Psi_{\mathrm{a}}(x_i,\boldsymbol{p}),\nabla^2\Psi_{\mathrm{a}}(x_i,\boldsymbol{p}))]^2\approx 0, \tag{3}$$

$E(\boldsymbol{p})$ 被称为损失函数.注意到在方程(3)中,定义域 D 被离散为多个子域.为求 $\Psi_{\mathrm{a}}(\boldsymbol{x},\boldsymbol{p})$,假设其表达式由一个前馈神经网络(见图 1)得到,激活函数取为如下形式:

$$s=\frac{1}{1+\mathrm{e}^{-x}}. \tag{4}$$

对于满足方程初始或边界条件的近似解,假设近似解由两部分组成:

$$\Psi_{\mathrm{a}}(\boldsymbol{x},\boldsymbol{p})=A(\boldsymbol{x})+G(\boldsymbol{x},\boldsymbol{N}(\boldsymbol{x},\boldsymbol{p})), \tag{5}$$

$$N_i(\boldsymbol{x},\boldsymbol{p})=\sum_{k=1}^{H}v_{ik}s(w_{ik}x+b_{ik})=\sum_{k=1}^{H}v_{ik}\left(\frac{1}{1+\mathrm{e}^{-w_{ik}x-b_{ik}}}\right), \tag{6}$$

此处 $\boldsymbol{p}$ 为一组待定的参数 v_{ik}, w_{ik} 及 b_{ik},表示神经网络结构中的权值和偏值.$N_i(\boldsymbol{x},\boldsymbol{p})$ 是参数为 $\boldsymbol{p}$,输入为 $\boldsymbol{x}$ 的前馈神经网络的输出值.$A(\boldsymbol{x})$ 部分不包含可调节参数,且满足方程(2).H 是隐藏层的隐藏单元数.

这样就建立了微分方程的神经网络近似解表达式,可以通过训练神经网络的权值和偏值

来得到满足微分-代数方程组的近似解析解.由于针对此模型的梯度优化方法的不足[29],在求解最优化问题时本文运用单纯形方法进行训练[30].

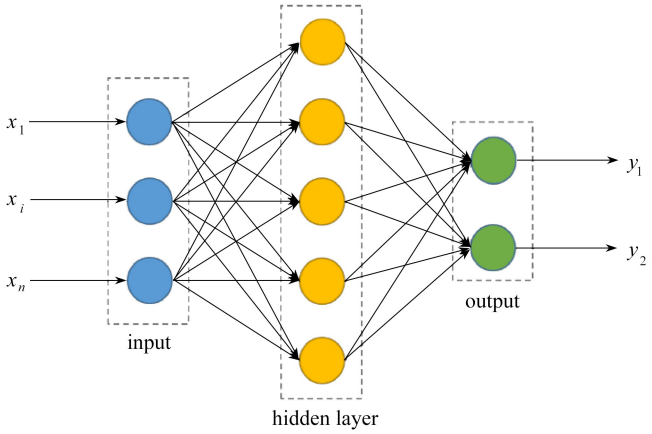

图 1 前馈人工神经网络结构图

Fig. 1 The structure of a feedforward artificial neural network

2 Hessenberg 型微分-代数方程的求解

一般形式的微分-代数方程 $F(t,\boldsymbol{y},\boldsymbol{y}')=0$,有时不能很好地表达某些数学模型,不便于直接数值求解.不过实际中的大多数高指标的微分-代数方程可以表示成耦合的常微分方程组与几何约束组的组合.在这样的系统中,高指标的微分-代数方程由代数变量和微分变量明确确定.原则上,代数变量可以由微分代数方程经过多次(与指标次数相同)微分后消除.这样的微分-代数方程形式称为 Hessenberg 型微分-代数方程[12-13].下面给出 3 种形式的 Hessenberg 微分-代数方程,并用神经网络方法计算近似解析解.

2.1 Hessenberg 指标 1 型

考虑如下形式的微分-代数方程:

$$\begin{cases}\boldsymbol{x}'=\boldsymbol{f}(t,\boldsymbol{x},\boldsymbol{z}),\\ \boldsymbol{0}=\boldsymbol{g}(t,\boldsymbol{x},\boldsymbol{z}),\end{cases} \tag{7}$$

这里的 Jacobi(雅可比)矩阵 $\boldsymbol{g}_z$ 在任何时候都是非奇异的.方程(7)为指标 1 的半显性微分-代数方程的一般形式.具体地,考虑如下指标 1 的微分-代数方程:

$$\begin{bmatrix}2 & 3 & 2\\ 1 & 0 & -2\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1'\\ x_2'\\ x_3'\end{bmatrix}+\begin{bmatrix}18 & 14 & 10\\ 0 & 1 & 2\\ -27 & -21 & -15\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}=\boldsymbol{0}, \tag{8}$$

初始条件为 $\boldsymbol{x}(0)=[1/3 \quad 1 \quad -2]^{\mathrm{T}}$,方程的精确解析解为

$$\begin{bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{bmatrix}=\begin{bmatrix}1/6+1/6\cdot e^{2t/3}\\ -1/3+4/3\cdot e^{2t/3}\\ 1/6-13/6\cdot e^{2t/3}\end{bmatrix}. \tag{9}$$

由初始条件建立神经网络假设解的表达形式:

$$x_i(t)=x_i(0)+N_i(t,\boldsymbol{p})t. \tag{10}$$

损失函数为

$$E(\boldsymbol{p}) = E(w,v,b) = \sum_{j=1}^{n} \boldsymbol{K}_j^{\mathrm{T}} \boldsymbol{K}_j, \tag{11}$$

$$\boldsymbol{K}_j = \begin{bmatrix} 2 & 3 & 2 \\ 1 & 0 & -2 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1'(t_j) \\ x_2'(t_j) \\ x_3'(t_j) \end{bmatrix} + \begin{bmatrix} 18 & 14 & 10 \\ 0 & 1 & 2 \\ -27 & -21 & -15 \end{bmatrix} \begin{bmatrix} x_1(t_j) \\ x_2(t_j) \\ x_3(t_j) \end{bmatrix}, \tag{12}$$

n 为样本数.

将神经网络的隐含层数设为 1,单元数设为 10,并在[0,1] s 时间区域内随机取 20 个样本点来训练神经网络,得到近似解及误差.限于篇幅,3 组参数 v_{ik}, w_{ik} 及 b_{ik} 此处未给出.图 2 给出了一组结果,表 1~3 定量地比较了精确解与近似解析解的相对误差.可以看到,应用前馈人工神经网络的方法所得的结果与精确解相当吻合.可见把前馈人工神经网络应用在 Hessenberg 指标 1 型微分-代数系统的求解上是可行的.

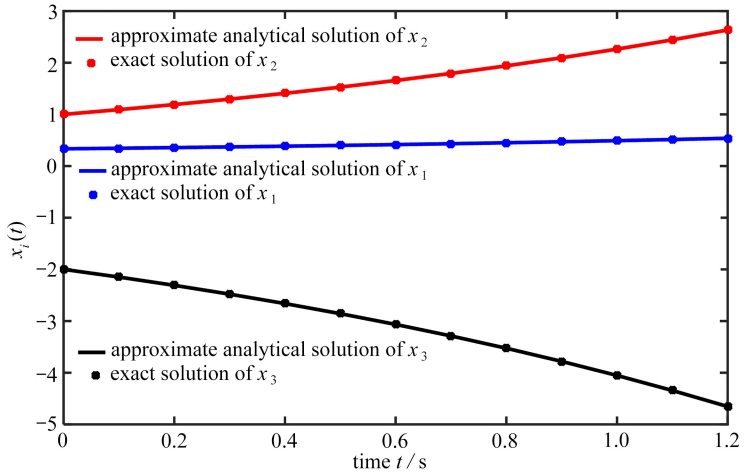

图 2 方程(8)的近似解析解与精确解比较

Fig. 2 Comparison between approximate analytical solutions and exact solutions to eq. (8)

表 1 方程(8)的近似解析解与精确解比较: $x_1(t)$

Table 1 Comparison between approximate analytical solutions and exact solutions to eq. (8): $x_1(t)$

t/s	$x_1(t)$		
	approximate analytical solution	exact solution	relative error ($\delta \times 10^{-3}$) /%
0.1	0.344 823 496 235 311	0.344 823 184 291 208	0.090 464 944 864 246
0.2	0.357 094 927 076 269	0.357 105 135 299 287	2.858 604 374 063 68
0.3	0.370 222 116 573 598	0.370 233 793 026 695	3.153 805 329 767 39
0.4	0.384 259 836 853 412	0.384 267 528 677 492	2.001 684 635 420 36
0.5	0.399 264 636 660 499	0.399 268 737 514 348	1.027 091 145 420 60
0.6	0.415 299 649 200 321	0.415 304 116 273 545	1.075 614 964 821 27
0.7	0.432 436 152 152 065	0.432 444 959 704 719	2.036 687 549 938 79
0.8	0.450 753 682 964 502	0.450 767 477 553 792	3.060 245 021 493 15
0.9	0.470 338 502 470 639	0.470 353 133 398 418	3.110 626 195 622 90
1	0.491 280 835 953 784	0.491 289 006 842 446	1.663 153 164 031 76

表 2 方程(8)的近似解析解与精确解比较：$x_2(t)$

Table 2 Comparison between approximate analytical solutions and exact solutions to eq. (8): $x_2(t)$

t/s	$x_2(t)$		
	approximate analytical solution	exact solution	relative error ($\delta \times 10^{-3}$)/%
0.1	1.091 921 493 550 53	1.091 918 807 663 00	0.245 978 686 101 078
0.2	1.190 190 363 551 59	1.190 174 415 727 63	1.339 956 879 224 020
0.3	1.295 219 496 307 96	1.295 203 677 546 89	1.221 333 859 869 390
0.4	1.407 484 882 672 60	1.407 473 562 753 27	0.804 272 253 104 553
0.5	1.527 492 700 941 73	1.527 483 233 448 12	0.619 809 985 630 914
0.6	1.655 777 931 140 93	1.655 766 263 521 69	0.704 665 839 405 033
0.7	1.792 909 504 093 23	1.792 893 010 971 09	0.919 916 695 579 609
0.8	1.939 494 592 696 78	1.939 473 153 763 67	1.105 399 838 443 210
0.9	2.096 181 441 210 65	2.096 158 400 520 68	1.099 186 490 910 720
1	2.263 661 854 582 88	2.263 645 388 072 90	0.727 433 283 719 295

表 3 方程(8)的近似解析解与精确解比较：$x_3(t)$

Table 3 Comparison between approximate analytical solutions and exact solutions to eq. (8): $x_3(t)$

t/s	$x_3(t)$		
	approximate analytical solution	exact solution	relative error ($\delta \times 10^{-3}$)/%
0.1	−2.149 371 587 698 14	−2.149 368 062 452 37	0.164 013 127 185 800
0.2	−2.309 042 002 423 14	−2.309 033 425 557 40	0.371 448 314 569 160
0.3	−2.479 711 608 701 59	−2.479 705 976 013 70	0.227 151 442 215 732
0.4	−2.662 147 832 254 82	−2.662 144 539 474 06	0.123 689 030 024 420
0.5	−2.857 164 588 020 16	−2.857 160 254 353 19	0.151 677 420 357 427
0.6	−3.065 626 431 208 13	−3.065 620 178 222 75	0.203 971 301 414 077
0.7	−3.288 458 102 677 48	−3.288 451 142 828 02	0.211 645 214 162 307
0.8	−3.526 650 961 109 52	−3.526 643 874 865 97	0.200 934 480 531 830
0.9	−3.781 265 151 206 24	−3.781 257 400 846 10	0.204 967 800 885 515
1	−4.053 429 702 543 21	−4.053 423 755 618 46	0.146 713 620 486 725

2.2 Hessenberg 指标 2 型

考虑如下形式的微分-代数方程：

$$\begin{cases} \boldsymbol{x}' = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{z}), \\ \boldsymbol{0} = \boldsymbol{g}(t, \boldsymbol{x}), \end{cases} \tag{13}$$

这里的 Jacobi 矩阵乘积 $\boldsymbol{g}_x \boldsymbol{f}_z$ 在任何时刻都是非奇异的，通过两次微分变换可将其化成 ODEs. 方程(13)为指标 2 型微分-代数方程的一般形式.具体地，考虑一个如下形式的 Hessenberg 指标 2 型微分-代数方程：

$$\begin{bmatrix} 1 & x_3 & -x_2 - 1 \\ x_3 + 1 & x_1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1' \\ x_2' \\ x_3' \end{bmatrix} = \begin{bmatrix} \sin t - x_1 + 1 \\ -\mathrm{e}^{-t} \\ x_1 x_2 x_3 - \mathrm{e}^{-t} \sin t \cos t \end{bmatrix}, \tag{14}$$

初始条件为 $\boldsymbol{x}(0) = [1 \quad 0 \quad 1]^{\mathrm{T}}$.通过微分变换得到 $\boldsymbol{x}'$ 的初始值 $\boldsymbol{x}'(0) = [-1 \quad 1 \quad 0]^{\mathrm{T}}$.

建立如下神经网络近似解的表达形式：

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} 1 - t \\ t \\ 1 \end{bmatrix} + t^2 \begin{bmatrix} N_1(t, \boldsymbol{p}) \\ N_2(t, \boldsymbol{p}) \\ N_3(t, \boldsymbol{p}) \end{bmatrix}. \tag{15}$$

这里的损失函数为

$$E(\boldsymbol{p}) = E(w,v,b) = \sum_{j=1}^{n} \boldsymbol{K}_j^{\mathrm{T}} \boldsymbol{K}_j, \tag{16}$$

$$\boldsymbol{K}_j = \begin{bmatrix} 1 & x_3(t_j) & -x_2(t_j) - 1 \\ x_3(t_j) + 1 & x_1(t_j) & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1'(t_j) \\ x_2'(t_j) \\ x_3'(t_j) \end{bmatrix} + \begin{bmatrix} \sin t_j - x_1(t_j) + 1 \\ -\mathrm{e}^{-t_j} \\ x_1(t_j)x_2(t_j)x_3(t_j) - \mathrm{e}^{-t_j}\sin t_j \cos t_j \end{bmatrix}. \tag{17}$$

原方程的精确解析解为

$$\boldsymbol{x}(t) = [\mathrm{e}^{-t} \quad \sin t \quad \cos t]^{\mathrm{T}}. \tag{18}$$

以下是与上例相同的隐藏层的训练结果,图 3 给出了精确解与近似精确解的比较图,表 4 给出了相对误差.根据对比可以得出结论,应用前馈人工神经网络的方法,结合微分得到的初始条件求出的解与精确解吻合.可见,把前馈人工神经网络应用在 Hessenberg 指标 2 型微分-代数系统的求解上是可行的.

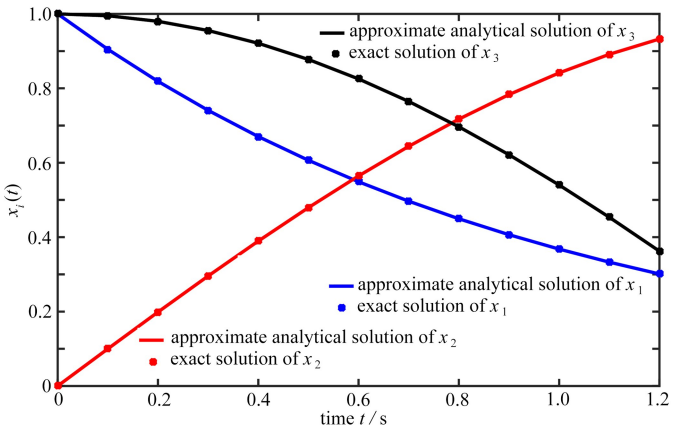

图 3 方程(14)的近似解析解与精确解对比

Fig. 3 Comparison between approximate analytical solutions and exact solutions to eq. (14)

表 4 方程(14)近似解析解与精确解的相对误差

Table 4 Relative errors between approximate analytical solutions and exact solutions to eq. (14)

t / s	relative error ($\delta \times 10^{-2}$) /%		
	$x_1(t)$	$x_2(t)$	$x_3(t)$
0.1	0.126 832 861 540 007	1.428 826 090 327 65	0.000 575 894 464 878 154
0.2	0.302 849 501 333 754	1.557 249 078 913 52	0.015 849 287 973 979 8
0.3	0.338 329 927 716 987	0.884 183 883 889 322	0.086 882 479 955 363 8
0.4	0.166 844 614 644 368	0.189 281 804 753 055	0.254 027 207 109 693
0.5	0.188 622 813 796 571	1.343 506 509 351 15	0.536 033 607 114 447
0.6	0.640 611 966 020 120	2.323 690 752 274 56	0.921 443 430 987 687
0.7	1.054 271 385 660 24	2.923 266 657 055 27	1.367 693 232 675 09
0.8	1.258 481 317 711 55	2.967 574 586 049 43	1.804 431 813 369 00
0.9	1.052 349 619 200 93	2.298 115 588 706 85	2.138 008 102 711 91
1	0.206 749 500 226 202	0.757 626 047 097 973	2.253 655 453 454 50

2.3 Hessenberg 指标 3 型

考虑如下一般形式的微分-代数方程：

$$\begin{cases} \boldsymbol{x}' = \boldsymbol{f}(t,\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}), \\ \boldsymbol{y}' = \boldsymbol{g}(t,\boldsymbol{x},\boldsymbol{y}), \\ \boldsymbol{0} = \boldsymbol{h}(t,\boldsymbol{y}), \end{cases} \tag{19}$$

此处的 Jacobi 矩阵乘积 $\boldsymbol{h}_y\boldsymbol{g}_x\boldsymbol{f}_z$ 在任何时候都是非奇异的.对该方程需做 3 次微分变换才能完全化为 ODEs,即方程(19)为指标 3 型微分-代数方程.具体地,考虑一个如下形式的微分-代数方程：

$$\begin{cases} x_2' + x_1 - 1 = 0, \\ tx_2' + x_3' + 2x_2 - 2t = 0, \\ tx_2 + x_3 - \mathrm{e}^t = 0, \end{cases} \tag{20}$$

$\boldsymbol{x}(0) = [0 \quad -1 \quad 1]^{\mathrm{T}}$,$\boldsymbol{x}'(0) = [1 \quad 1 \quad 2]^{\mathrm{T}}$ 为给定的初始条件.

建立如下神经网络近似解的表达形式：

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} t \\ t-1 \\ 1+2t \end{bmatrix} + t^2 \begin{bmatrix} N_1(t,\boldsymbol{p}) \\ N_2(t,\boldsymbol{p}) \\ N_3(t,\boldsymbol{p}) \end{bmatrix}. \tag{21}$$

损失函数为

$$E(\boldsymbol{p}) = E(w,v,b) = \sum_{j=1}^{n} \boldsymbol{K}_j^{\mathrm{T}}\boldsymbol{K}_j, \tag{22}$$

$$\boldsymbol{K}_j = \begin{bmatrix} x_2'(t_j) + x_1(t_j) - 1 \\ t_j x_2'(t_j) + x_3'(t_j) + 2x_2(t_j) - 2t_j \\ t_j x_2(t_j) + x_3(t_j) - \mathrm{e}^{t_j} \end{bmatrix} m. \tag{23}$$

原方程的精确解析解为

$$\boldsymbol{x}(t) = [\mathrm{e}^t - 1 \quad 2t - \mathrm{e}^t \quad (1+t)\mathrm{e}^t - 2t^2]^{\mathrm{T}}. \tag{24}$$

以下是与上例相同的隐藏层的训练结果,将求得的近似解析解与精确解对比如下,表 5 及图 4 给出了精确解与近似精确解的比较.

表 5 方程(20)近似解析解与精确解的相对误差

Table 5 Relative errors between approximate analytical solutions and exact solutions to eq. (20)

t/s	relative error δ/%		
	$x_1(t)$	$x_2(t)$	$x_3(t)$
0.1	0.755 649 750 471 465	0.003 008 272 663 055 41	0.016 936 281 950 228 1
0.2	1.227 508 066 625 07	0.004 273 905 483 367 99	0.038 353 875 622 227 9
0.3	1.447 707 721 652 91	0.043 907 766 628 710 9	0.041 899 108 697 408 0
0.4	1.451 291 231 288 24	0.129 871 390 360 823	0.022 170 473 906 022 9
0.5	1.276 237 229 041 10	0.260 735 422 823 970	0.016 630 437 421 319 6
0.6	0.963 428 405 629 209	0.413 633 469 756 345	0.064 610 040 768 728 9
0.7	0.556 531 576 549 882	0.544 961 824 200 906	0.109 180 390 944 694
0.8	0.101 756 637 666 012	0.602 606 389 304 657	0.137 852 627 350 028
0.9	0.352 539 946 466 488	0.547 492 774 805 822	0.140 835 684 652 440
1	0.756 439 256 358 322	0.372 920 440 502 510	0.113 166 136 116 771

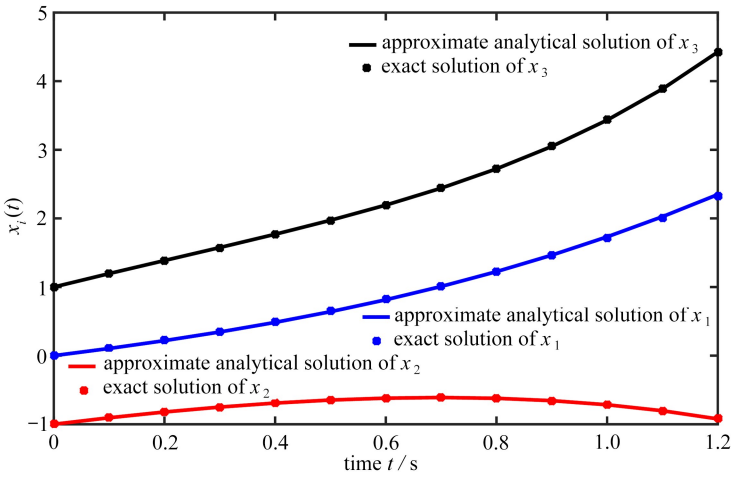

图 4 方程(20)的近似解析解与精确解对比

Fig. 4 Comparison between approximate analytical solutions and exact solutions to eq. (20)

3 Euler-Lagrange 方程

这里以平面运动的单摆为例,如图 5 所示,建立直角坐标系 xOy 下的动力学方程,几何约束方程为 $x^2 + y^2 = l^2$.引入未定乘子,得到如下含 Lagrange 乘子 λ 的 Lagrange 函数:

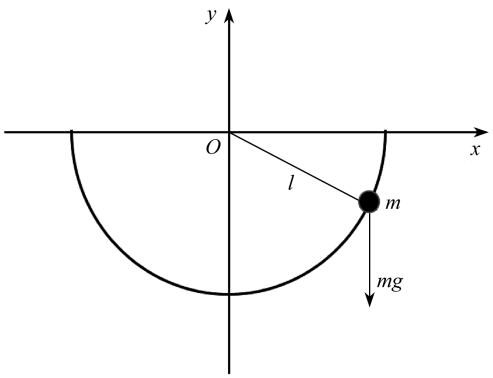

图 5 单摆模型图

Fig. 5 A simple pendulum model

$$L = \frac{1}{2} m(\dot{x}^2 + \dot{y}^2) - mgy - \lambda(x^2 + y^2 - l^2). \tag{25}$$

代入第二类 Lagrange 方程

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = 0, \tag{26}$$

得到如下的动力学方程:

$$\begin{cases} m\ddot{x} + 2\lambda x = 0, \\ m\ddot{y} + 2\lambda y + mg = 0, \\ x^2 + y^2 - l^2 = 0, \end{cases} \tag{27}$$

这是指标 3 的微分-代数系统.引入变量 u, v, 将其转化为一阶微分-代数方程组:

$$\begin{cases} \dot{x} = u, \\ \dot{y} = v, \\ m\dot{u} = -2\lambda x, \\ m\dot{v} = -2\lambda y - mg, \\ x^2 + y^2 - l^2 = 0. \end{cases} \tag{28}$$

其损失函数可以表示为

$$E(\boldsymbol{p}) = E(w, v, b) = \sum_{j=1}^{n} (\boldsymbol{K}_j^{\mathrm{T}} \boldsymbol{K}_j + \mu f^2), \tag{29}$$

式中 μ 为惩罚因子,此处取 20,

$$\boldsymbol{K}_j = \begin{bmatrix} x'(t_j) - u(t_j) - 1 \\ y'(t_j) - v(t_j) \\ mu'(t_j) + 2\lambda x(t_j) \\ mv'(t_j) + 2\lambda x(t_j) + mg \end{bmatrix}, \tag{30}$$

$$f = x^2 + y^2 - l^2. \tag{31}$$

取摆杆水平向右的静止状态为初始状态,质量 $m = 1$ kg,杆长 $l = 2$ m.初始状态设为

$$\begin{cases} [x(0) \quad y(0) \quad u(0) \quad v(0)]^{\mathrm{T}} = [2 \quad 0 \quad 0 \quad 0]^{\mathrm{T}}, \\ [\dot{x}(0) \quad \dot{y}(0) \quad \dot{u}(0) \quad \dot{v}(0)]^{\mathrm{T}} = [0 \quad 0 \quad 0 \quad -1 \quad 0]^{\mathrm{T}}. \end{cases} \tag{32}$$

对约束方程微分,求得加速度初始条件.建立如下神经网络近似解的表达形式:

$$\begin{bmatrix} x(t) \\ y(t) \\ u(t) \\ v(t) \end{bmatrix} = \begin{bmatrix} 2 \\ 10(\cos t - 1) \\ 0 \\ -10\sin t \end{bmatrix} + t^2 \begin{bmatrix} N_1(t,\boldsymbol{p}) \\ N_2(t,\boldsymbol{p}) \\ N_3(t,\boldsymbol{p}) \\ N_4(t,\boldsymbol{p}) \end{bmatrix}, \tag{33}$$

$$\lambda = \sum_{k=1}^{H} v_{5k} s(w_{5k} x + b_{5k}) = \sum_{k=1}^{H} v_{5k} \left(\frac{1}{1 + \mathrm{e}^{-w_{5k}x - b_{5k}}} \right). \tag{34}$$

代入方程(30)得到损失函数.

神经网络的隐含层数设为3,单元数设为10.在时间区域[0, 2] s内随机取20个样本点来训练神经网络,得到如下结果.将求得的近似解析解与Runge-Kutta法数值解对比,如图6所示.根据对比可以得出结论,应用前馈人工神经网络的方法,可以求出指标3的Euler-Lagrange方程的解.算例数值分析还表明,前馈神经网络可以在不违约的情况下求得Euler-Lagrange方程近似解析解.这对于多体系统动力学方程的数值求解和稳定性分析具有一定的参考价值.

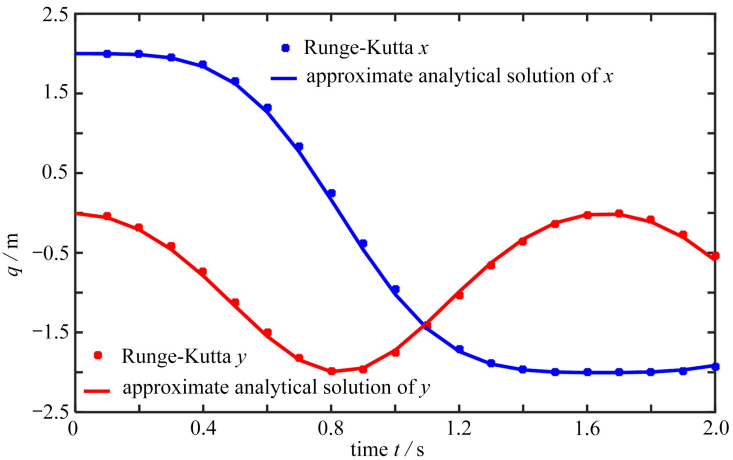

图6 方程(27)的近似解析解与数值解对比

Fig. 6 Comparison between approximate analytical solutions and numerical solutions to eq. (27)

4 结论与讨论

本文研究了一种求解微分-代数方程的智能算法:人工神经网络法.该方法用神经网络建立模型并应用数值最优化方法对方程进行求解.应用该方法,求解了3类Hessenberg型微分-

代数方程(指标1, 2, 3型)和Euler-Lagrange方程,得到了近似解析解.将所得的近似解析解与原方程精确解或Runge-Kutta法数值解结果进行比较,可以看到神经网络方法所得的结果有较高的精度.该方法的一个显著特点是可直接得到近似解析解的显式表达式,为后续研究提供了便利.

注意到本文选取的激活函数为Sigmoid型,即方程(4)的形式.在神经网络分析中还有其他形式的激活函数,如三角函数型、双曲正切型(tanh)、整流线性单元型等.对于本文Euler-Lagrange型微分-代数方程的求解,计算表明:Sigmoid型精度最高,三角函数型、双曲正切型精度较低;对于Hessenberg指标2型微分-代数方程的求解,Sigmoid型、三角函数型、双曲正切型精度都很高.而整流线性单元模型均不宜用于上述两类微分-代数方程的求解.可见,Sigmoid型激活函数能最大程度地保证求解精度,但采用Sigmoid型激活函数的计算时间最长,三角函数型激活函数的计算时间最短.

参考文献(References):

[1] DAI L. *Singular Control Systems* (*Ser Lecture Notes in Control and Information Sciences*) [M]. Berlin: Springer, 1989.

[2] SIMEON B. *Computational Flexible Multibody Dynamics: a Differential-Algebraic Approach* [M]. Berlin: Springer, 2013.

[3] ILCHMANN A, REIS T. *Surveys in Differential-Algebraic Equations II*[M]. London: Springer, 2015.

[4] BENNER P, BOLLHROFER M, KRESSNER D, et al. *Numerical Algebra, Matrix Theory, Differential-Algebraic Equations and Control Theory*[M]. London: Springer, 2015.

[5] BERGER T. The zero dynamics form for nonlinear differential-algebraic systems[J]. *IEEE Transactions on Automatic Control*, 2017, **62**(8): 4131-4137.

[6] SCARCIOTTI G. Steady-state matching and model reduction for systems of differential-algebraic equations[J]. *IEEE Transactions on Automatic Control*, 2017, **62**(10): 5372-5379.

[7] MOSTACCIUOLO E, VASCA F, BACCARI S. Differential algebraic equations and averaged models for switched capacitor converters with state jumps[J]. *IEEE Transactions on Power Electronics*, 2018, **33**(4): 3472-3483.

[8] GEAR C W. Simultaneous numerical solution of differential-algebraic equations[J]. *IEEE Transactions on Circuit Theory*, 1971, **18**(1): 89-95.

[9] CASH J R. Modified extended backward differentiation formulate for the numerical solution of stiff initial value problems in ODEs and DAEs[J]. *Computers and Mathematics With Applications*, 2000, **125**(1/2): 117-130.

[10] PETZOLD L. Differential/algebraic equations are not ODEs[J]. *Journal on Scientific and Statistical Computing*, 1982, **3**(3): 367-384.

[11] HAIRER E, LUBICH C, ROCHE M. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*[M]. Berlin: Springer, 1989.

[12] GUZEL N, BAYRAM M. Numerical solution of differential-algebraic equations with index-2[J]. *Applied Mathematics and Computation*, 2006, **174**(2): 1279-1289.

[13] KARTA M, ÇELIK E. On the numerical solution of differential-algebraic equations with Hessenberg index-3[J]. *Discrete Dynamics in Nature and Society*, 2012, **2012**: 147240. DOI: 10.1155/2012/147240.

[14] SARAVI M, BABOLIAN E, ENGLAND R, et al. System of linear ordinary differential and differential-algebraic equations and pseudo-spectral method[J]. *Computers and Mathematics With Applications*, 2010, **59**(4): 1524-1531.

[15] HOSSEINI M M. Adomian decomposition method for solution of differential-algebraic equations [J]. *Journal of Computational and Applied Mathematics*, 2006, **197**(2): 495-501.

[16] NEWMAN C K. Exponential integrators for the incompressible Navier-Stokes equations[D]. PhD Thesis. Blacksburg: Virginia Polytechnic Institute, 2003.

[17] LIU C S, CHEN W, LIU L W. Solving mechanical systems with nonholonomic constraints by a Lie-group differential algebraic equations method[J]. *Journal of Engineering Mechanics*, 2017, **143**(9): 04017097.

[18] HAUG E J. An index 0 differential algebraic equation formulation for multibody dynamics nonholonomic constraints[J]. *Mechanics Based Design of Structures and Machines*, 2018, **46**(1): 38-65.

[19] 潘振宽, 赵维加, 洪嘉振, 等. 多体系统动力学微分/代数方程组数值方法[J]. 力学进展, 1996, **26**(1): 28-40.(PAN Zhenkuan, ZHAO Weijia, HONG Jiazhen, et al. On numerical algorithms for differential/algebraic equations of motion of multibody systems[J]. *Advances in Machines*, 1996, **26**(1): 28-40.(in Chinese))

[20] 赵维加, 潘振宽. 存在相关约束 Euler-Lagrange 方程解的存在唯一性[J]. 力学与实践, 2002, **24**(3): 49-54.(ZHAO Weijia, PAN Zhenkuan. Existence and uniqueness of solutions of Euler-Lagrange equation with dependent constrains[J]. *Mechanics in Engineering*, 2002, **24**(3): 49-54.(in Chinese))

[21] 吴永. 约束多体系统动力学方程的辛算法[J]. 重庆大学学报, 2004, **27**(6): 102-105.(WU Yong. Symplectic methods of the dynamic equations of constrained multibody systems[J]. *Journal of Chongqing University*, 2004, **27**(6): 102-105.(in Chinese))

[22] 王加霞. Euler-Lagrange 方程的高精度计算方法[J]. 青岛大学学报(自然科学版), 2008, **21**(1): 22-26.(WANG Jiaxia. A algorithm of high precision for Euler-Lagrange equations[J]. *Journal of Qingdao University* (*Natural Sciences*), 2008, **21**(1): 22-26.(in Chinese))

[23] 刘颖, 马建敏. 多体系统动力学方程的无违约数值计算方法[J]. 计算力学学报, 2010, **27**(5): 942-947.(LIU Ying, MA Jianmin. Precise numerical solution for multi-body system's equations of motion based on algorithm without constraint violation[J]. *Chinese Journal of Computational Mechanics*, 2010, **27**(5): 942-947.(in Chinese))

[24] 马秀腾, 翟彦博, 谢守勇. 多体系统指标 2 运动方程 HHT 方法违约校正[J]. 力学学报, 2017, **49**(1): 175-181.(MA Xiuteng, ZHAI Yanbo, XIE Shouyong. HHT method with constraints violation correction in the index 2 equations of motion for multibody systems[J]. *Chinese Journal of Theoretical and Applied Mechanics*, 2017, **49**(1): 175-181.(in Chinese))

[25] CHOU J S, NGO N T, CHONG W K. The use of artificial intelligence combiners for modeling steel pitting risk and corrosion rate[J]. *Engineering Applications of Artificial Intelligence*, 2017, **65**: 471-483.

[26] LAGARIS I E, LIKAS A, FOTIADIS D I. Artificial neural networks for solving ordinary and partial differential equations[J]. *IEEE Transactions on Neural Networks*, 1998, **9**(5): 987-1000.

[27] ROSTAMI F, JAFARIAN A. A new artificial neural network structure for solving high order linear fractional differential equations[J]. *International Journal of Computer Mathematics*, 2018, **95**(3): 528-539.

[28] PARIPOUR M, FERRARA M, SALIMI M. Approximate solutions by artificial neural network of hybrid fuzzy differential equations[J]. *Advances in Mechanical Engineering*, 2017, **9**(9): 1-9.

[29] BISHOP C M. *Neural Networks for Pattern Recognition*[M]. Oxford: Oxford University Press, 1995.

[30] 龚纯, 王正林. 精通 MATLAB 最优化计算[M]. 北京: 电子工业出版社, 2009.(GONG Chun, WANG Zhenglin. *Proficient in MATLAB Optimization Calculation*[M]. Beijing: Publishing House of Electronics Industry, 2009.(in Chinese))

On Solutions to Several Classes of Differential-Algebraic Equations Based on Artificial Neural Networks

YANG Zhao[1], LAN Jun[2], WU Yongjun[1]

(1. *Department of Engineering Mechanics*, *Shanghai Jiao Tong University*, *Shanghai* 200240, *P.R.China*;
2. *Winning Health Technology Group Co.*, *Ltd.*, *Shanghai* 200072, *P.R.China*)

Abstract: In nonlinear science, it is always an important subject and research focus to find the approximate analytical solutions to differential equations. The artificial neural network and the optimization method were combined to solve 2 special classes of differential-algebraic equations (DAEs). The 1st 3 numerical examples, namely, the Hessenberg DAEs with indices 1, 2, 3, fell into a category of pure mathematical problems. Then the 2nd example related to Euler-Lagrange DAEs with indices 3, i.e. a pendulum without external force, arising from the background of nonholonomic mechanics. The approximate analytical solutions to the above 4 examples were obtained and compared with the exact solutions and the results from the Runge-Kutta method. High accuracy of the proposed method was demonstrated.

Key words: artificial neural network; differential-algebraic equation; approximate analytical solution; optimization method

Foundation item: The National Natural Science Foundation of China(11772293;11272201)

引用本文/Cite this paper:

杨钊, 兰钧, 吴勇军. 几类微分-代数方程的神经网络求解法[J]. 应用数学和力学, 2019, **40**(2): 115-126.

YANG Zhao, LAN Jun, WU Yongjun. On solutions to several classes of differential-algebraic equations based on artificial neural networks[J]. *Applied Mathematics and Mechanics*, 2019, **40**(2): 115-126.