Advances in Applied Mathematics and Mechanics
Adv. Appl. Math. Mech., Vol. **1**, No. 2, pp. 257-272 (2009)

April 2009

Solving Delay Differential Equations through RBF Collocation

Francisco Bernal[1,*] and Gail Gutierrez[2]

[1] *Institute for Scientific Computing, Technical University of Dresden 01062 Dresden, Germany,*

[2] *Cir 1 No. 73-34, Instituto de Energia y Termodinamica, Universidad Pontificia Bolivariana, Medellin, Colombia.*

Received 23 October 2008; Accepted (in revised version) 27 December 2008

Available online 17 March 2009

Abstract. A general and easy-to-code numerical method based on radial basis functions (RBFs) collocation is proposed for the solution of delay differential equations (DDEs). It relies on the interpolation properties of infinitely smooth RBFs, which allow for a large accuracy over a scattered and relatively small discretization support. Hardy's multiquadric is chosen as RBF and combined with the Residual Subsampling Algorithm of Driscoll and Heryudono for support adaptivity. The performance of the method is very satisfactory, as demonstrated over a cross-section of benchmark DDEs, and by comparison with existing general-purpose and specialized numerical schemes for DDEs.

AMS subject classifications: 34-04, 65L99

Key words: Meshless method, delay differential equations, radial basis function, multiquadric, adaptive collocation.

1 Introduction

In this work, we present a general numerical approach for solving DDEs based on the RBF collocation method invented by Kansa [19] [20], also known as Kansa's method. Due to its many advantages (which include superior interpolation accuracy, spectral convergence, robustness with respect to the discretization support, and ease of coding), Kansa's method is becoming increasingly popular for the solution of ordinary and partial differential equations (ODEs and PDEs, respectively). Its performance in the solution of DDEs, however, has scarcely been explored, with the exception of a recent paper on the solution of neutral DDEs with multiquadrics [22].

*Corresponding author.
Email: fco_bernal@hotmail.com (F. Bernal), gail.gutierrez@upb.edu.co (G. Gutierrez)

This paper is organized as follows. In Section 2, Kansa's method is adapted to a general formulation of (first order) DDEs. The basic algorithm is further improved by the inclusion of several heuristic observations concerning the tunable shape parameter which appears in the multiquadric RBF, and by the residual subsampling algorithm (RSA) by Driscoll and Heryuodono [8]. The RSA is at the core of the high accuracy attained by the multiquadrics interpolant. Section 2 is closed by some remarks concerning the solution of nonlinear problems with Kansa's method. Section 3 tests the proposed method against a cross-section of benchmark problems taken from the literature. As we shall see, not only does Kansa's method attain excellent results in well-understood (first order) DDEs, but also in the less explored neutral and higher-order DDEs — which may offer an additional tool for looking into this kind of problems. Finally, Section 4 concludes the paper.

2 Solving linear DDEs through Kansa's method

Consider the following linear DDE

$$y'(x) - p(x)y(x) - q(x)y[x-\tau(x)] = s(x), \qquad \text{if } x \in (a,b], \tag{2.1}$$

$$y(x) = h(x), \qquad \text{if } x \le a. \tag{2.2}$$

It will be convenient to split (2.2) into a DDE and an ODE

$$y'(x) - p(x)y(x) - q(x)y[x-\tau(x)] = s(x), \qquad \text{if } x-\tau(x) > a, \tag{2.3}$$

$$y'(x) - p(x)y(x) = q(x)h[x-\tau(x)] + s(x), \qquad \text{if } x-\tau(x) < a, \tag{2.4}$$

$$y(a) = h(a). \tag{2.5}$$

Discretize $[a,b]$ into a set N scattered nodes $\xi=\{x_j, j=1...N\}$ (with $x_1=a$ and $x_N=b$), and consider as well the outside point $x_0=a-\lambda$, $\lambda>0$. We seek an approximate solution to (2.3)-(2.5) in the form of an expansion of $N+1$ RBFs $\phi_j(r)$:

$$y(x) = \sum_{j=0}^{j=N} \alpha_j \phi(\|x-x_j\|). \tag{2.6}$$

The addition of an RBF at x_0 allows to enforce *both* the initial condition *and* the DDE at $x=a$, thus contributing to the accuracy (this is the PDECB strategy discussed in [12]). Once the coefficients α_j are available, the approximate RBF solution can be reconstructed anywhere in $[a,b]$. In order to solve for the coefficients, (2.3)-(2.5) are enforced over (2.6) on a set of collocation N nodes, usually ξ. Notice that no equation is collocated on x_0, but two of them are on $x_1=a$. For $i=1,\ldots,N$, this leads to the linear system of dimension $N+1$,

$$\sum_{j=0}^{j=N} \left\{ \phi_j'(r_{ij}) - p(x_i)\phi_j(r_{ij}) - q(x_i)\phi_j(\|x_i-\tau(x_i)-x_j\|) \right\} = s(x_i), \qquad \text{if } x_i-\tau(x_i) > a, \tag{2.7}$$

$$\sum_{j=0}^{j=N}\Big\{\phi_j'(r_{ij})-p(x_i)\phi_j(r_{ij})\Big\}\alpha_j=q(x_i)h[x_i-\tau(x_i)]+s(x_i),$$

$$\text{if } x_i-\tau(x_i)\leq a, \tag{2.8}$$

$$\sum_{j=0}^{j=N}\alpha_j\phi_j(r_{ij})=h(a), \qquad \text{if } x_i=a, \tag{2.9}$$

where $r_{ij}=\|x_i-x_j\|$. In the remainder of this paper we will restrict ourselves to the well-tested Hardy's multiquadric (MQ),

$$\phi_j(r_j)=\sqrt{\|\ x-x_j\ \|^2+c_j^2}, \tag{2.10}$$

whose derivative is

$$\phi_j'(r_j)=\frac{x-x_j}{\sqrt{\|\ x-x_j\ \|^2+c_j^2}}, \tag{2.11}$$

as the RBF of choice. The shape of the MQ depends on the free parameter c_j (hence the name of *shape parameter* for it). The fact that the MQ has global support leads to fully populated matrices. It is a hallmark of Kansa's method that the best accuracy can only be obtained at the expense of extreme ill-conditioning, as will be discussed next. In order to improve stability, the direct inversion of the linear system (2.7)-(2.9) has been replaced by the use of Penrose's pseudoinverse.

2.1 Choosing the shape parameters c_j

Although the accuracy of the interpolant (2.6) is largely influenced by the values c_j, $j=0,\dots,N+1$, theoretical results regarding the choice of an 'optimal' set of values are not yet available, and heuristic rules must be used instead, which mostly address the homogeneous case $c_j=c$. In this case, the convergence rate of the error of the interpolant (2.6) has been proven to go as $\Lambda^{c/h}$ in interpolation problems [23], and has been shown to obey $\Lambda^{\sqrt{c}/h}$ in elliptic PDEs [6], where $0<\Lambda<1$ and h is the distance between nodes. Therefore, the accuracy could seemingly be improved at no computational cost by increasing c. In practice, however, this spectral regime of convergence only takes place for extremely smooth RBFs, *i.e.* at the cost of shrinking the interpolation space. Moreover, as $c\to\infty$, the MQ profile becomes increasingly flatter and the collocation system (2.7)-(2.9) becomes extremely ill-conditioned, dictating in practice a limit for the accuracy attainable at a given resolution h and machine precision. A trade-off principle arises between accuracy and stability, which is actually common to all parameter-dependent RBFs, not only MQs [28]. Optimal results are obtained by pushing c as large as possible before incurring in numerical instability. Since in the approximation of differential equations the exact solution is unknown, other estimators are used instead, which replicate the behavior of the error curves with c and are

available in run time. Examples are the 'leave-one-out' strategy [27] [11], or the residual to the ODE/PDE [6]. For instance, in (2.3)-(2.5),the pointwise residual is defined as

$$R(x) = s(x) - \sum_{j=0}^{j=N} \alpha_j \phi_j'(x) + p(x) \sum_{j=0}^{j=N} \alpha_j \phi_j(x) + q(x) \sum_{j=0}^{j=N} \alpha_j \phi_j'[x - \tau(x)]. \tag{2.12}$$

The case where c is center-dependent has been less investigated, although it may outperform MQ collocation with constant c, as shown in a numerical investigation by Kansa and Carlson [21]. Carlson and Foley showed that c_j is related to the curvature of the function to be interpolated at $x \approx x_j$ [4]. In [15], Hon and Mao let $c_j = Mj + b$, where j is the center index and M and b are chosen so that the condition number κ is about 10^{16}. In [31], Wertz *et al.* reported improved accuracy in a 2D problem if $c_j \gg c_0$ for $(x_j, y_j) \in \partial\Omega$ and $c_j = \mu(1 + \gamma(-1)^j)$ if $(x_j, y_j) \in \Omega$, for some constants μ and γ. These findings were confirmed in the 1D case in a later work by Fornberg and Zuev [13]. Another common strategy has been to set c proportional to the distance to the closest node in the point set, *v.g.* in [8].

2.2 Extension to nonlinear DDEs

In the case that the DDE is nonlinear, or that the lagged argument is a function of the solution itself (*a state-delay DDE*), the collocation of the interpolant (2.6) leads to a system of nonlinear algebraic equations for the unknowns $\alpha_0, \ldots, \alpha_N$. Let us write this system as

$$\begin{aligned} F_0(\alpha_0, \ldots, \alpha_N) &= 0, \\ &\vdots \\ F_N(\alpha_0, \ldots, \alpha_N) &= 0. \end{aligned} \tag{2.13}$$

In order to solve $\vec{F} = \vec{0}$, a gradient-based method may be used. In the MATLAB routine *fsolve*, the user can choose between providing the analytical Jacobian J to the solver

$$J = \begin{pmatrix} \frac{\partial F_0}{\partial \alpha_0} & \cdots & \frac{\partial F_0}{\partial \alpha_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_N}{\partial \alpha_0} & \cdots & \frac{\partial F_N}{\partial \alpha_N} \end{pmatrix}, \tag{2.14}$$

or allowing it to construct J based on finite differences. In order to keep the implementation of Kansa's method as simple and general as possible, we have only explored the latter possibility. However, there is a practical drawback: while it is well known that the convergence of Newton-type methods is very sensitive to the condition number of the Jacobian, RBF interpolation needs to push κ for the best accuracy, often beyond

the ill-condition threshold (which is $\kappa \approx 10^{14}$ in our MATLAB environment). Consequently, we have used instead a trust-region method (that of Powell's [26]) in our numerical experiments, with good results. Nevertheless, the condition number must be kept lower than in linear DDEs in order to guarantee convergence, which is likely to prevent optimal accuracy as well.

2.3 Adaptive selection of nodes

Another important yet open issue in Kansa's method is the optimal number and location of RBF centers/collocation nodes. We will restrict ourselves to the case where both point sets are identical (save for the extra RBF center at x_0 added in order to enforce the equation at $x = a$). In 1D problems, regular grids are often preferred for simplicity, although experimental evidence suggests that the optimal placement of nodes is problem-dependent, *i.e.* is determined by the function to be interpolated. We will be using an algorithm introduced by Driscoll and Heryuodono [8] which works well in practice, both for interpolation and differential equations and not only in 1D. The idea is to monitor the residual R to the differential equation at midpoints and iteratively refine the point set until R drops below some user-defined threshold. The reader is referred to the original paper for details. Here, we present a slightly modified version of the algorithm which we have preferred.

Algorithm 2.1: Residual Subsampling Algorithm (RSA)

a) Initially, discretize $[a,b]$ into a grid of $N^{(0)}$ nodes with spacing $\Delta=(b-a)/(N^{(0)}-1)$. Define $x_j=a+(j-1)\Delta$, $j=1,\dots,N^{(0)}$, $\xi^{(0)}=\{x_j\}$, and $x_0=a-\Delta$. The $N^{(0)}+1$ starting MQ centers are the set $x_0 \cup \xi^{(0)}$. Define the values of the adjustable parameters $\lambda>0$, $\mu>0$, $\gamma>0$, $\eta>0$, $\theta_{max}>\theta_{min}>0$, and *itmax*.

b) For $k=0,\dots$ until $\max|R_j^{(k)}|<\theta_{max}$ or $k>$*itmax*

- Distribute the shape parameters as $c_0=c_{N^{(k)}}=\lambda\mu d_1$, and $c_j=\mu d_j[1+\gamma(-1)^j]$, $j=1,\dots,N^{(k)}-1$, where d_j is the distance to the closest collocation node from x_j.
- Compute set of midpoints $z_j=(x_j+x_{j+1})/2$, $j=1\dots N^{(k)}-1$.
- Solve the DDE through Kansa's method with $y(x)=\sum_{j=0}^{j=N^{(k)}}\alpha_j\phi_j(\|x-x_j\|)$.
- Compute the residuals $\{R_j^{(k)}\}$ to the differential equation at midpoints.
- Set $\Theta^{(k)}=\max\left(\theta_{max},\max_{j=1\dots N^{(k)}-1}|R_j^{(k)}/\eta|\right)$.
- Define point set $\Xi^{(k)}=\xi^{(k)}\cup\{z_j \text{ such that } |R_j^{(k)}|>\Theta^{(k)}\}$.
- Delete points x_i, $i=2\dots N^{(k)}-1$ such that $|R_{i-1}^{(k)}|<\theta_{min}>|R_{i+1}^{(k)}|$ from $\Xi^{(k)}$.
- Let $\xi^{(k+1)}=\Xi^{(k)}$ and $\{x_1,\dots,x_{N^{(k+1)}}\}=\xi^{(k+1)}$.
- Update $\{d_j\}$ for $j=1\dots N^{(k+1)}$.
- Consider the set of $1+N^{(k+1)}$ MQs centered at $x_0\cup\xi^{(k+1)}$ and iterate.

In the above algorithm, the shape parameters are adjusted after each iteration in order to prevent the condition number from skyrocketing. As further nodes are in-

cluded, however, the onset of instability will be eventually reached and the accuracy of the MQ approximation begins to deteriorate. The only tweakings to the original RSA in [8] are: PDECB, the use of the recipe in [31] in the distribution of c's, and the subtitution of θ_{max} by $\Theta^{(k)}$ on enlargement of the point set.

3 Numerical examples

In the remainder of the paper, we will refer to the method described in section 2 as MQCM (multiquadric collocation method). The MQCM is coded in MATLAB 7 running on a laptop with 1.8 GHz CPU and 1 GB RAM. In this section, the MQCM is tested against a cross-section of benchmark DDEs taken from the literature. The performance of the MQCM is compared with that of MATLAB built-in general-purpose routines DDE23 [29] by Shampine and Thompson, or DDESD [30] by Shampine, which are both based on Runge-Kutta-type schemes. DDE23 is restricted to constant delays, while the more recent DDESD can handle variable- and state-delay equations as well. For Examples 4 and 5, DDENSD has been used instead of DDESD, which is a routine based on DDESD for DDEs of neutral type. The fact that these three programs are written in MATLAB allows for a direct comparison of error estimates and CPU times with MQCM. In particular, the root mean squared error is defined as

$$RMS(\epsilon) = \sqrt{\frac{\sum_{i=1}^{i=N_{ev}} [u_{NUM}(z_i) - u_{EX}(z_i)]^2}{N_{ev}}}, \tag{3.1}$$

where u_{EX} is the exact solution, u_{NUM} the approximation yielded by the considered numerical scheme, ϵ is the point-wise error, and z_i, $i = 1, \ldots, N_{ev}$=103 is a set of equispaced evaluation points in $[a, b]$. In some of the examples presented, published results of some specialized method for the kind of DDE considered have been included as further reference. In such cases, not all the estimators are available for comparison. CPU times, in particular, cannot be directly compared - which is denoted by adding an * to the corresponding entry.

In all of the numerical examples which follow, the working parameters for the RSA have been set to

$$\lambda = 10, \quad \mu = \sqrt{40/N^{(0)}}, \quad \gamma = 0.1, \quad \eta = 10, \quad \theta_{max} = 10^{-13}, \quad \theta_{min} = 10^{-14}, \tag{3.2}$$

except in Example 5 where $\mu = \sqrt{25/N^{(0)}}$. The initial discretization is $N^{(0)}$=6 in Examples 1-4, and $N^{(0)}$=10 in Examples 5 and 6.

3.1 Example 1: Stiff DDE

Consider the following DDE with a stiffness parameter p (Example 1 in [17]).

$$\begin{cases} y'(x) = Ay(x) + y\left(x - \frac{3\pi}{2}\right) - A\sin(x), & x \in [0, 13], \\ y(x) = e^{px} + \sin(x), & x \in \left[-3\pi/2, 0\right], \end{cases} \tag{3.3}$$

Table 1: Comparison to other methods (Example 1).

p	x	ϵ_{MQ}	ϵ_{DDE23}	ϵ_{SPC}
	$3\pi/4$	5.1(-15)	1.9(-12)	2.6(-6)
	$3\pi/2$	6.2(-14)	1.5(-12)	7-9(-8)
-0.1	$9\pi/4$	9.7(-14)	1.4(-12)	1.0(-5)
	3π	3.5(-14)	8.2(-13)	3.1(-7)
	$15\pi/4$	1.6(-13)	8.6(-13)	8.4(-7)
DoF		261	71072	27
RMS(ϵ)		9.4(-14)	2.3(-12)	
CPU		16	891	0.009 *
	$3\pi/4$	1.9(-13)	2.5(-11)	8.3(-9)
	$3\pi/2$	7.0(-14)	1.3(-11)	7.6(-7)
-1	$9\pi/4$	4.2(-14)	6.6(-12)	1.5(-8)
	3π	6.6(-14)	2.0(-14)	4.2(-7)
	$15\pi/4$	6.4(-14)	1.9(-11)	2.0(-7)
DoF		254	82258	33
RMS(ϵ)		6.0(-14)	3.9(-11)	
CPU		26	1358	0.017 *
	$3\pi/4$	9.3(-14)	2.0(-10)	1.3(-10)
	$3\pi/2$	1.4(-13)	1.2(-10)	1.1(-9)
-2	$9\pi/4$	4.2(-14)	2.0(-10)	2.1(-10)
	3π	1.8(-14)	1.0(-12)	1.1(-9)
	$15\pi/4$	8.1(-14)	1.0(-10)	2.1(-10)
DoF		281	151122	51
RMS(ϵ)		1.4(-13)	2.1(-10)	
CPU		19	5850	0.036 *

where $A=p-e^{-3\pi p/2}$. The exact solution is given by $y_{EX}(x)=e^{px}+\sin(x)$. For $p<0$, the solution consists of a short transient of exponential decay, followed by periodic sinusoidal oscillations (see Fig. 1).

Since the parameter p also enters the equation exponentially, its effect on the stiffness of the problem is dramatic. Table 1 compares the performance of the MQCM with that of DDE23 and with that of the spectral method in [17] (SPC). In Table 1, an entry

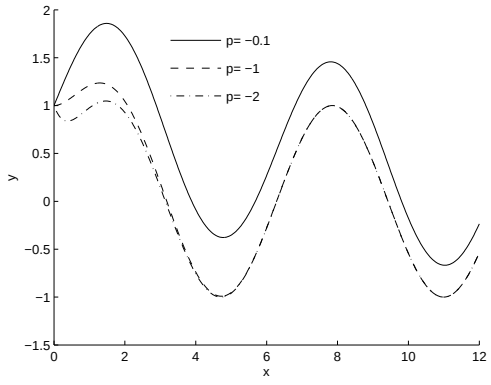

Figure 1: Plots of the exact solution of Example 1.

Table 2: RSA iterations (Example 1).

	$p=-0.1$			$p=-1$			$p=-2$		
it	DoF	RMS(ϵ)	Condition ♯	DoF	RMS(ϵ)	Condition ♯	DoF	RMS(ϵ)	Condition ♯
0	7	0.77	3.3(+10)	7	0.94	1.9(+14)	7	1.04	3.9(+17)
1	12	0.0040	3.2(+14)	10	0.0112	3.4(+14)	11	0.02	1.2(+17)
2	15	1.6(-6)	4.3(+13)	14	4.0(-5)	5.3(+16)	13	1.6(-4)	7.3(+17)
3	22	7.5(-8)	1.7(+16)	23	1.0(-7)	4.3(+17)	15	3.0(-4)	2.3(+17)
4	27	3.5(-7)	4.1(+18)	27	6.2(-8)	4.2(+17)	23	2.1(-6)	7.5(+16)
5	51	6.4(-11)	1.1(+18)	45	1.6(-10)	1.8(+19)	27	1.3(-7)	4.0(+17)
6	94	5.5(-12)	2.2(+18)	67	3.2(-10)	7.5(+18)	43	1.3(-9)	8.9(+17)
7	110	4.0(-12)	5.9(+18)	73	1.3(-11)	2.1(+18)	69	1.1(-10)	9.1(+18)
8	146	1.8(-12)	9.7(+18)	95	1.1(-11)	1.5(+19)	70	2.3(-11)	1.6(+19)
9	153	2.3(-13)	3.3(+18)	120	1.5(-12)	1.4(+19)	77	6.4(-12)	4.9(+18)
10	261	9.4(-14)	1.4(+19)	127	5.8(-13)	7.6(+18)	132	1.8(-12)	2.2(+18)
11				203	3.3(-13)	3.1(+19)	219	2.6(-12)	1.3(+19)
12				239	1.7(-13)	6.7(+19)	281	1.4(-13)	3.2(+19)
13				254	6.0(-14)	1.4(+19)			

like $5.1(-15)$ means 5.1×10^{-15}, and so on. DoF (degrees of freedom) stands for the size of support of the given discretization scheme -the number of MQ centers in the MQCM. The listed results for DDE23 are the best within a reasonable computing time and/or memory restrictions.

The MQCM is barely affected, if anything, by the increasing stiffness of the problem. In fact, the advantages of the MQCM in dealing with stiff ODEs were already reported in [15]. In terms of efficiency, the MQMC outperforms DDE23. The inversion of full matrices required by the MQCM is made up for by the gain in the size of the discretization support.

On the other hand, the accuracy of the SPC can be improved by increasing the order of the scheme, as happens in Table 1 for different p. The SPC is more efficient than the MQCM, but is affected by the increasing value of p (see discussion in [17]). Moreover, it is restricted to constant delays.

Table 2 shows the performance of the RSA throughout the iterations for this prob-

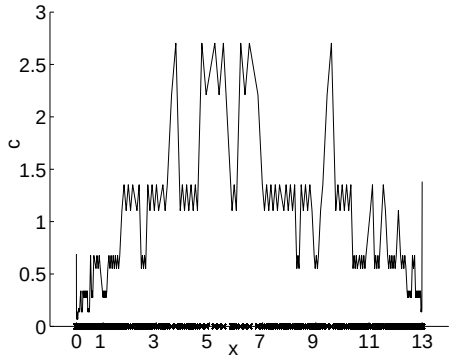

Figure 2: Effect of RSA in Example 1, $p=-2$ (last iteration, DoF=281): Shape-parameter and node distributions.

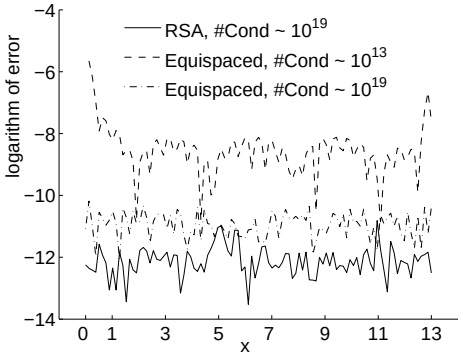

Figure 3: Effect of RSA in Example 1, $p=-2$ (last iteration, DoF=281): Pointwise errors with and without RSA.

lem. While it converges on average, the scheme is clearly not monotone. It is surprising that the convergence can be sustained at so high condition numbers, even considering that in our implementation of Kansa's method the ill-conditioning problem has been -at least partially- ameliorated by the use of the pseudoinverse (instead of the direct inversion of the matrix). We point out that this phenomenon of high accuracy at very high condition numbers has already been reported when smooth functions are interpolated with MQs [15].

Fig. 2 illustrates the effect of the node- and shape-parameter distributions according to the RSA, for the last iteration in the case $p=-2$. Due to the many DoFs (281), it is difficult to make out the individual nodes. Notice, however, that the node density essentially varies inversely proportional to c. For the same number of collocation nodes, but evenly arranged in $[0,13]$, the accuracy is much worse, with an RMS of $2.3(-7)$. Indeed, it is also due to the fact that the shape parameters are generally lower, as the condition number of roughly 10^{13} reflects. Therefore, we increased μ to yield a comparable condition number to that of the RSA. While the RMS drops to $2.0(-11)$, it is still some 100 times higher than the accuracy attained by the RSA, therefore showing the importance of node adaptivity (Fig. 3).

3.2 Example 2: Pantograph DDE

Consider the following pantograph differential equation (see also [3]).

$$\begin{aligned} &y'(x) = -y(x) + \frac{q}{2}y(qx) - \frac{q}{2}e^{-qx}, \qquad 0 \le x \le T, \\ &y(0) = 1, \qquad\qquad\qquad\qquad\qquad 0 < q < 1, \end{aligned} \tag{3.4}$$

whose solution is $y_{EX}(x)=e^{-x}$.

Numerical methods for DDEs like (3.4) are a topical subject of research because of two features associated to a proportional delay of the form

$$\tau(x) = (1-q)x, \qquad 0 < q < 1,$$

namely: it vanishes at $x=0$ and becomes unbounded as $x\to\infty$. The former one leads to difficulties in carrying out the integration of the first step, while the latter entails the need for a vast amount of computer memory if long term integration ($T\gg 0$) is required. In what follows we set $T=10$. Table 3 compares the MQCM with DDESD and with the specialized reference method (REF) in [3], which works on a specific (*geometric*) kind of mesh in order to attain superconvergence. The listed results for DDESD are not the best attainable, but those for which the CPU time is comparable to that of the MQCM.

A recent improvement to the reference method [3] is [16] , more efficient than the former in case that long integration times T are required. For (3.4) with $T=10$ and $q=0.5$, it attains $|y(x=T)-y_{EX}(x=T)|=1.8(-13)$ with 1280 nodes. The results of the MQCM throughout the first 11 iterations of the RSA are shown in Table 4. A fifth column has been added that lists the errors (in absolute value) of the MQCM solution at $x=T$.

3.3 Example 3: DDE with discontinuity propagation

In the event that the solution $y(x)$ to the DDE has discontinuities or low-order derivative singularities, Kansa's method performs relatively poorly. The reason is that nonsmooth features do not belong to the interpolation space, which is made up of infinitely derivable MQs that cannot possibly capture them accurately. Any attempt to do so will bring about Gibbs' oscillations around the singularities, whose amplitude will not be damped by letting $N\to\infty$. An interesting approach is to include MQs with $c=0$ close to the singularities as in [18]. However, although the oscillations are indeed reduced, we have not been able to recover the high convergence rate attained with smooth solutions. In order to solve DDEs with piecewise smooth solutions, Kansa's method can still be applied sequentially if the domain is partitioned into subintervals which are C^∞. For instance, assume that it can be predicted that the only three singularities take place at $a<x_1<x_2<x_3<b$. First, the DDE is solved in the subdomain $a\leq x<x_1$ to yield $y^{(1)}_{app}(x)$. Then, $y^{(1)}_{app}(x)$ is used as history function for the second subdomain $x_1<x<x_2$ yielding $y^{(2)}_{app}(x)$, and so on.

The next example ([17], Example 4, also [25] 1.1.12), deals with a DDE having piecewise C^∞ initial function:

$$\begin{aligned} &y'(x) = y(x) + y(x-1), \\ &y(x) = \begin{cases} 0, & x\in[-1,-1/3), \\ 1, & x\in[-1/3,0]. \end{cases} \end{aligned} \tag{3.5}$$

The analytical solution for $x\in[0,8/3]$ is given by

$$y_{EX}(x) = \begin{cases} e^x, & x\in[0,2/3], \\ -1+C_1e^x, & x\in[2/3,1], \\ xe^{x-1}+C_2e^x, & x\in[1,5/3], \\ 1+C_1xe^{x-1}+C_3e^x, & x\in[5/3,2], \\ (x^2/2-x)e^{x-2}+C_2xe^{x-1}+C_4e^x, & x\in[2,8/3], \end{cases}$$

$$\begin{aligned} C_1 &= 1+e^{-2/3},\ \ C_2 = -2e^{-1}+C_1, \\ C_3 &= \frac{5}{3}e^{-1}+C_2-e^{-5/3}-C_1\frac{5}{3}e^{-1}, \\ C_4 &= e^{-2}+2C_1e^{-1}+C_3-2C_2e^{-1}. \end{aligned}$$

Notice that the discontinuity of the initial function propagates in x, giving rise to singularities of order k at points $x_k=-1/3+k$, $k\geq 0$. For both the MQCM and the SPC

Table 3: Comparison to other methods (Example 2).

	REF		MQ			DDESD		
q	DoF	$MAX(\epsilon)$	DoF	$MAX(\epsilon)$	CPU	DoF	$MAX(\epsilon)$	CPU
0.9	1600	8.8(-13)	179	1.7(-13)	9.7	4136	3.7(-14)	11.8
0.5	1600	1.2(-11)	135	2.8(-13)	5.0	4136	3.5(-14)	11.2
0.2	1600	1.5(-11)	192	2.0(-13)	9.3	4136	3.3(-14)	11.0

Table 4: RSA iterations (Example 2).

it	DoF	RMS(ϵ)	Condition ♯	$\epsilon(x=T)$
0	7	0.01	1.9(+11)	0.02
1	12	2.2(-5)	3.3(+15)	5.3(-6)
2	14	2.6(-5)	1.7(+14)	1.3(-4)
3	16	4.3(-7)	1.1(+14)	7.3(-7)
4	23	1.4(-8)	4.2(+16)	3.5(-9)
5	30	6.0(-10)	2.2(+17)	3.0(-10)
6	49	2.3(-10)	1.1(+18)	1.2(-10)
7	64	2.6(-11)	7.3(+18)	1.8(-10)
8	65	3.2(-11)	3.3(+18)	3.5(-11)
9	77	1.7(-12)	1.2(+18)	2.4(-12)
10	135	1.3(-13)	3.9(+18)	2.8(-13)
11	177	2.2(-13)	8.5(+18)	8.7(-15)
CPU	7.8			

Table 5: Comparison to other methods (Example 3).

x	ϵ_{MQ}	ϵ_{DDE23}	ϵ_{SPC}
0.25	2.0(-14)	6.0(-15)	9.2(-13)
0.5	4.4(-14)	2.2(-14)	1.0(-13)
0.75	4.0(-14)	1.9(-13)	5.6(-15)
1	6.2(-14)	2.6(-13)	1.3(-15)
1.25	1.0(-13)	3.6(-13)	1.1(-11)
1.5	7.7(-14)	4.9(-13)	1.2(-11)
1.75	1.3(-13)	6.3(-13)	5.8(-14)
2	1.8(-13)	8.4(-13)	3.3(-15)
2.25	4.8(-13)	1.5(-12)	7.3(-11)
2.5	6.6(-13)	2.4(-12)	7.9(-11)
DoF	342	38386	45
RMS(ϵ)	3.2(-13)	9.3(-13)	
CPU	11	246	

to cope with this problem, the integration domain $[a,b]=[0,8/3]$ must be divided into the 5 smooth subintervals in (3.6). The results of MQCM, SPC and DDE23 are listed in Table 5. DDE23 has a *'Jumps'* option which has been set to a vector that contains the locations of the discontinuities.

3.4 Example 4: Neutral state-delay DDE

Neutral DDEs (which involve lagged derivatives) are considered tougher to handle with numerical methods than retarded ODEs and are an active research field. In the following problem, taken from [24] (see also [25] 2.3.4), the delay is a function of the solution itself, and therefore the DDE is nonlinear. The MQCM tackles it with Powell's method, implemented by the option *'dogleg'* of MATLAB nonlinear solver *fsolve*.

$$\begin{cases} y'(x) = -y'(y(x)-2), & x \geq 0 \\ y(x) = 1-x, & x \leq 0. \end{cases} \tag{3.6}$$

The exact solution is $y_{EX}(x)=1+x,\ 0 \leq x \leq 1$.

Table 6: RSA iterations (Example 4).

It	DoF	RMS(ϵ)	Condition	NL iter
0	7	0.0452	7.5(+11)	f
1	13	1.3(-11)	5.1(+13)	22
2	14	1.2(-11)	2.9(+15)	16
3	18	1.9(-13)	1.1(+14)	19
4	24	2.0(-14)	2.5(+15)	15
CPU	7.5			

The initial guess of $y(x)$ required to trigger Powell's method is $y_{GUESS}(x)$=0. The entry f in Table 6 means that Powell's method fails to converge in the maximum number of iterations allowed (set to 30). Nevertheless, it yields an approximation good enough to be used as a guess for the nonlinear solution with 13 MQs (whose solution is in turn used as a guess for the next RSA iteration, and so on). For reference, DDENSD yields $RMS(\epsilon_{REF})$=2.2(-9) in 50.1 s. CPU time.

3.5 Example 5: Vanishing state-delay DDE

This example is a nonlinear neutral differential equation with vanishing state delay. It was first proposed in [9] as a modification of a problem originally considered in [5]:

$$\begin{cases} y'(x) = \cos(x)\left[1 + y\left(xy^2(x)\right)\right] + cy(x)y'\left(xy^2(x)\right) + g(x), & 0 \le x \le \pi, \\ g(x) = (1-c)\sin(x)\cos\left(x\sin^2(x)\right) - \sin\left(x + x\sin^2(x)\right), \\ y(0) = 0. \end{cases} \tag{3.7}$$

For every choice of the parameter c, the exact solution is $y_{EX}(x)$=$sin(x)$. Because the delay vanishes at x=$0, \pi/2, 3\pi/2, ...$, the numerical solution of (3.7) by Runge-Kutta methods causes some difficulties. For the MQCM, the main difficulty is that the condition number must be kept low enough (below 10^{14}) for the nonlinear solver to converge in a reasonable number of nonlinear iterations (again Powell's algorithm in the *fsolve* routine). Therefore, we have set $N^{(0)}$=11 and μ=$\sqrt{20/N^{(0)}}$. The initial hint of the solution is y_{GUESS}=1/2. As reference results (REF), we have taken those of [14] (example 2), where (3.7) is solved by the Radau-type code RADAR5 (Table 7). In the case c=1, there is a singularity at x=$\pi/2$ -in the sense that $y'(\pi/2)$ is not well defined- and the MQCM with default parameters fails.

Table 7: RSA iterations (Example 5).

c	DoF	Condition	CPU	RMS(ϵ)	$\epsilon_{MQ}(x=\pi)$	$\epsilon_{REF}(x=\pi)$
-1.0	65	1.8(13)	58.3	4.7(-9)	3.3(-9)	1.8(-9)
-0.7	44	2.5(11)	20.4	3.2(-8)	9.5(-9)	4.2(-9)
-0.3	44	9.1(10)	9.8	3.2(-8)	7.5(-8)	1.7(-10)
0.0	69	6.8(12)	30.0	3.0(-8)	1.6(-8)	1.2(-9)
0.3	46	9.9(10)	17.1	4.3(-9)	2.5(-9)	1.0(-9)
0.7	49	3.0(11)	36.2	1.1(-9)	7.2(-10)	5.3(-9)
1.0	f					4.3(-8)

3.6 Example 6: Second order DDE

The last example illustrates the ability of the MQCM to accurately solve higher-order DDEs. Since equations of this type are less common in the literature, most solvers are not designed to handle them. In order to compare, we have transformed a system of two state-delay DDEs into a second-order DDE:

$$\begin{cases} y''(x) = \big(\exp[1-y(x)]-x\big)y\big(x-\exp[1-y(x)]\big)y'(x)^2, & x \geq 1, \\ y(x) = \log(x), & 0 < x \leq 1, \end{cases} \tag{3.8}$$

which is obtained by differentiation of $y_2(x)$ and insertion into $y_1''(x)$ in

$$\begin{cases} y_1(x) = \log(x), & 0 < x \leq 1, \\ y_2(x) = 1/x, & 0 < x \leq 1, \\ y_1'(x) = y_2(x), & x \geq 1, \\ y_2'(x) = \big(\exp[1-y_1(x)]-x\big)y_2(x)(x-\exp[1-y_1(x)])y_2^2(x), & x \geq 1, \end{cases} \tag{3.9}$$

(see [1] and [25], 1.4.17). The exact solution is $y_{EX}(x)$=$y_{1,EX}(x)$=$\log(x)$.

We consider the interval $[a,b]$=$[1,5]$. In this problem, the second derivative of the multiquadric (2.10) is required, as well as *two* extra MQ centers for PDEBC (since y,y', and y'' are enforced at x=a). Such centers are placed at x_0=$a-\Delta$ and x_{-1}=$a-2\Delta$. Results are shown in Table 8.

An indirect reference is provided by DDESD which solves (3.9) with RMS(ϵ_{REF}) = 2.3(−13) (for $y_1(x)$) in 2.1 s. Notice that ill-conditioning must be kept lower in order to ensure convergence of the nonlinear solver, thus limiting accuracy. A possible alternative would be to use a Newton-type routine with the analytical Jacobian to the given DDE. The good performance shown by MQCM relies on the accuracy with which numerical derivatives are reproduced in Kansa's method. While not every system of m DDEs can be transformed into a single DDM of order m, there are many cases where this transformation can be an advantageous alternative for the solution of DDE systems with the MQCM.

Table 8: RSA iterations (Example 6)

It	DoF	RMS(ϵ)	Condition	NL iter
0	12	0.103	2.4(+10)	*f*
1	18	2.9(-5)	2.0(+10)	*f*
2	21	3.1(-7)	6.8(+11)	1
3	28	3.3(-8)	8.7(+10)	1
4	37	3.7(-9)	5.5(+13)	1
5	49	2.1(-11)	2.5(+13)	1
6	63	6.2(-11)	1.7(+15)	1
7	72	8.6(-12)	3.5(+14)	1
CPU	48.5			

4 Conclusions

A novel numerical method for the solution of DDEs has been presented. It relies on the multiquadric collocation method introduced by Kansa combined with the RSA algorithm by Driscoll and Heryudono for node adaptivity, which uses the residual as the refinement criterion. As long as the solution of the DDE is smooth (or piecewise smooth, with the position of the singularities being known in advance), the present method can accurately handle a large variety of such problems, including state-delay, neutral, and high-order DDEs. Moreover, the scheme is straightforward to code and enjoys spectral convergence. Because in this paper the stress is placed on simplicity, nonlinearities are fed to a general-purpose solver, without attempting to optimize. Possible improvements include: providing an analytical Jacobian, and linearizing the DDE along the lines of [10] [2].

The presented method has as well a number of shortcomings. Kansa's scheme is limited to problems in a bounded domain, and in its current formulation, it is unable to address DDEs in the semiinfinite domain -whose solution with the use of geometric grids is a topic of research in the field on numerical DDEs. Regarding the handling of discontinuities, we point out that infinitely smooth RBFs -and MQs in particular- have been used to detect low-order singularities [7]. This ability might be exploited to allow the MQCM to cope with a more general class of DDEs.

References

[1] A. N. AL-MUTIB, *An Explicit One-Step Method of Runge-Kutta Type for Solving Delay Diff. Eqns.*, Utilitas Math., 31 (1987), pp. 67-80.

[2] F. BERNAL AND M. KINDELAN, *A meshless solution to the p-Laplace equation*, in Progress on Meshless Methods, A. J. M. Ferreira, E. J. Kansa, G. E. Fasshauer, V. Leitão (eds.), Springer 17-35 (2008).

[3] H. BRUNNER, Q. HU AND Q. LIN, *Geometric meshes in collocation methods for Volterra integral equations with proportional delays*, IMA J. Numer. Anal., 21 (2001), pp. 783-798.

[4] R. E. CARLSON AND T. A. FOLEY, *The parameter R2 in multiquadric interpolation*, Comput. Math. Appl., 21 (1991), pp. 29-42.

[5] R. N. CASTLETON, L. J. GRIMM, *A first order method for differential equations of neutral type*, Math. Comp., 27 (1973), pp. 571-577.

[6] A. H. D. CHENG, M. A. GOLBERG, E. J. KANSA AND T. ZAMMITO, *Exponential convergence and h-c multiquadric collocation method for partial differential equations*, Numer. Methods Part. Differ. Equat., 19 (2003), pp. 571-594.

[7] C.K. CHUI, J. STOECKLER, J.D. WARD, *Analytic wavelets generated by radial functions*, Adv. Computational Math., 5 (1996), pp. 95-123.

[8] T. A. DRISCOLL AND A. HERYUDONO, *Adaptive residual subsampling methods for radial basis function interpolation and collocation problems*, Computers Math. Appl., 53 (2007), pp. 927-939.

[9] W. H. ENRIGHT AND H. HAYASHI, *A delay differential equation solver based on a continuous Runge-Kutta method with defect control*, Numer. Alg., 16 (1998), pp. 349-364.

[10] G. E. FASSHAUER, *Nonsymmetric Multilevel RBF Collocation within an Operator Newton Framework for Nonlinear PDEs*, in Trends in Approximation Theory, K. Kopotun, T. Lyche, and M. Neamtu (eds.), Vanderbilt University Press, (2001), pp. 103-112 .

[11] G.E. FASSHAUER AND J. ZHANG, *On Choosing "Optimal" Shape Parameters for RBF Approximation*, Numerical Algorithms, 45 (2007), pp. 345-368 .

[12] A. I. FEDOSEYEV, M. J. FRIEDMAN AND E. J. KANSA, *Continuation for Nonlinear Elliptic Partial Differential Equations Discretized by the Multiquadric Method*, Int. J. Bifur. Chaos, 10 (2000), pp. 481-492.

[13] B. FORNBERG AND J. ZUEV, *The Runge phenomenon and spatially variable shape parameters in RBF interpolation*, Comput. and Math. with Applic, 54 (2007), pp. 379-398.

[14] N. GUGLIELMI AND E. HAIRER, *Implementing Radau IIA Methods for Stiff Delay Differential Equations*, Computing, 67(1) (2000), pp. 1-12.

[15] Y. C. HON AND X. Z. MAO, *A multiquadric interpolation method for solving initial value problems*, Sci. Comput., 12 (1997), No.1, pp. 51-55.

[16] E. ISHIWATAA AND Y. MUROYAB, *Rational approximation method for delay differential equations with proportional delay*, Applied Mathematics and Computation, 187 (2007), issue 2, pp. 741-747 .

[17] K. ITO, H. T. TRAN AND A. MANITIUS, *A Fully-Discrete Spectral Method for Delay Differential Equations*, SIAM J. Numerical Analysis, 28 (1991), pp. 1121-1140.

[18] J. H. JUNG, *A note on the Gibbs phenomenon with multiquadric radial basis functions*, Applied Numerical Mathematics 57 (2007), Issue 2, pp. 213-229.

[19] E. J. KANSA, *Multiquadrics - a scattered data approximation scheme with applications to computational fluid-dynamics. I. Surface approximations and partial derivative estimates*, Comput. Math. Appls., 19 (1990), pp. 127-145.

[20] E. J. KANSA, *Multiquadrics - a scattered data approximation scheme with applications to computational fluid-dynamics. II. Solutions to parabolic, hyperbolic and elliptic partial differential equations*, Comput. Math. Appls., 19 (1990), pp. 147-161.

[21] E.J. KANSA AND R. E. CARLSON, *Improved accuracy of multiquadric interpolation using variable shape parameters*, Comput. Math. Appl., 24 (1992), pp. 99-120.

[22] S. KARIMI VANANI AND A. AMINATAEI, *Multiquadric approximation scheme on the numerical solution of delay differential systems of neutral type*, Mathematical and Computer Modelling, In Press (2008).

[23] W. R. MADYCH, *Miscellaneous error bounds for multiquadric and related interpolators*, Comput. Math. Appl., 24 (1992), pp. 121-38.

[24] S. B. NORKIN AND L. E. EL'SGOL'TS, *Introduction to the Theory and Applications of Differential Equations with Deviating Arguments*, Math. in Sci. and Eng., 105 (1973), pp. 44-45.

[25] C. A. H. PAUL, *A test set of functional differential equations*, Technical Report 243, University of Manchester (1992).

[26] M.J.D. POWELL, *A Fortran Subroutine for Solving Systems of Nonlinear Algebraic Equations*, Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Ch.7 (1970).

[27] S. RIPPA, *An algorithm for selecting a good value for the parameter c in radial basis function interpolation*, Adv. Comput. Math., 11(2-3) (1999), pp. 193-210.

[28] R. SCHABACK, *Error estimates and condition numbers for radial basis function interpolation*, Adv. Comput. Math., 3 (1995), pp. 251-264.

[29] L. F. SHAMPINE AND S. THOMPSOM, *Solving DDEs in Matlab*, Appl. Numer. Math., 37 (2001), pp. 441-458.

[30] L. F. SHAMPINE, *Solving ODEs and DDEs with residual control*, Appl. Numer. Math., 52 (2005), pp. 113-127.

[31] J. WERTZ, E. J. KANSA AND L. LING, *The role of the multiquadric shape parameters in solving elliptic partial differential equations*, Computers and Mathematics with Applications, 51(8) (2006), pp. 1335-1348.