Bernal, F. (2012). Solving Non-smooth Delay Differential Equations with Multiquadrics. In: Günther M., Bartel A., Brunk M., Schöps S., Striebel M. (eds), *Progress in Industrial Mathematics at ECMI 2010.* Mathematics in Industry. (The European Consortium for Mathematics in Industry), vol. 17. Springer, Berlin, Heidelberg. Pp. 603-609.

DOI: https://doi.org/10.1007/978-3-642-25100-9_70

Solving Non-Smooth Delay Differential Equations with Multiquadrics

Francisco Bernal

Instituto Superior Tecnico, UTL Lisboa
Av. Rovisco Pais 1, 1049-001 Lisbon (Portugal) francisco.bernal@ist.utl.pt

Summary. We put forward a discretization scheme for the numerical solution of neutral differential equations (NDEs). The solution to the NDE in an interval $I = [a, b]$ is approximated by a multiquadric (MQ) interpolant, whose coefficients are found by collocation on a set of N nodes in I. This approach, also known as Kansa's method, enjoys an exponential rate of convergence and great flexibility regarding the location of the nodes, as long as the solution to the differential equation is smooth. However, the critical difficulty posed by NDEs is precisely that they propagate, forward in time and without damping, low-order discontinuities of the history function. Here, we exploit the sensitivity of the MQ interpolant to discontinuities in order to detect them in computing time. This allows for a partition of I into smooth subintervals, which are then sequentially solved by Kansa's method.

1 Introduction

Differential equations with delayed arguments are increasingly being used for modeling phenomena appearing in biology, economics, ecology, and engineering, to name but a few relevant fields of application [2]. The sophistication of many such models calls for accurate and reliable numerical solvers [9]. In turn, the growing availability of general-purpose numerical software encourage researchers to incorporate delay effects into their mathematical models.

Delayed differential equations (DDEs), however, are difficult to solve numerically. They inherit all of the potential challenges posed by ordinary differential equations (which can, in fact, be seen as a subset of DDEs), such as stiffness, instability, etc., but also have some difficulties of their own. Two of them are particularly relevant. In DDEs, there is often the need to interpolate the delayed argument between discretization nodes, which makes necessary a good interpolant. Moreover, DDEs often propagate discontinuities forward in time. We stress the fact that the arising discontinuities are part of the solution, not a numerical artifact. In order to understand why, consider the following simple DDE:

$$\begin{cases} y'(t) = y(t-\tau), & t \geq 0 \\ y(t) = h(t), & t < 0. \end{cases} \tag{1}$$

where $h(t)$ is the *history function* and τ is the *delay*. If $h(0) \neq y(0)$, a discontinuity occurs at $y(0)$. The solution progresses smoothly until $t = \tau$, when $y'(\tau^-) = h(0)$ but $y'(\tau^+) = y(0)$. This brings about a discontinuity in y' -which, in turn, will propagate as discontinuities in $y''(2\tau), y'''(3\tau)$..., and so on. Neutral differential equations are a type of delayed differential equation of the form:

$$y'(t) = F[t, y(t), y(t-\tau_1), y'(t-\tau_2)] \tag{2}$$

Because of the presence of the delayed argument in the derivative (which is the hallmark of NDEs), singularities may propagate without 'smoothing out', as was the case in the previous example. The correct tracking of such discontinuities make NDEs especially challenging from a numerical point of view, since the performance of most numerical methods degrade when they step over discontinuities. If the NDE is linear and has a constant delay, the location of the derivative jumps can be predicted, and measures which preserve the accuracy of the numerical scheme can be prepared in advance -typically, splitting the interval by the jump. For non-linear NDEs (or non-constant delays), however, this may not be possible, and discontinuities must be detected in computing time. Therefore, a discontinuity detector is a critical ingredient in a NDE solver.

MQ interpolants have advantageous interpolation properties such as exponential spatial convergence [7]. In exchange, they behave poorly in the presence of singularities. In this work, we benefit from the sensitivity of MQ interpolants to non-smooth features of the solution in a twofold way: to detect them, first, and to approximate the smooth solution of the NDE in each of the singularity-free subintervals which lie in between.

This paper is organized as follows. In Section 2, the features of MQ interpolation which are of interest for this work are reviewed. In Section 3, Kansa's method is adapted to linear NDEs with singularities. Section 4 concludes the paper with a numerical example.

2 MQ interpolation of 1D functions

Consider $f : [a, b] \to \mathbb{R}$ and N scattered nodes $\Xi = \{x_1 < x_2 < ... < x_{N-1} < x_N\}$. Hardy's multiquadric (MQ) centered at x_j is defined as:

$$\phi_j^c(x - x_j) = \sqrt{(x - x_j)^2 + c_j^2} \tag{3}$$

whose derivative is

$$\phi_j^c(x - x_j)' = \frac{x - x_j}{\sqrt{(x - x_j)^2 + c_j^2}} \tag{4}$$

The shape of the MQ depends on the free parameter c_j (hence the name of *shape parameter* for it). Notice that it can be thought of as a characteristic distance. An MQ interpolant of f can be written as

$$f(x) \approx \sum_{j=1}^{N} \alpha_j \phi_j^c(x - x_j) \tag{5}$$

where the coefficients are found by collocation, i.e. by solving the linear system

$$\begin{bmatrix} \phi_1^c(0) & \dots & \phi_N^c(x_1 - x_N) \\ \vdots & \ddots & \vdots \\ \phi_1^c(x_N - x_1) & \dots & \phi_N^c(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{bmatrix} \tag{6}$$

The fact that the MQ has global support leads to fully populated matrices. It is a hallmark of this method that the best accuracy can only be obtained at the expense of extreme ill-conditioning [8]. Notice also that, since the interpolation space is made up of infinitely smooth basis functions $\phi_j^c(x), j = 1...N$, it cannot be expected to capture well non-smooth features in f. Let us assume that the N nodes are equispaced in $[a, b] = [x_1, x_N]$, and that $c_1 = ... = c_N = c$. We are interested in the case when f is piecewise smooth with a single jump discontinuity δ, located at $a < x_0 < b$ (i.e., $|f(x_0^+) - f(x_0^-)| = \delta$). Then, the MQ interpolant will exhibit oscillations around x_0 -the well known Gibbs phenomenon. The problem just described was studied in detail in [4]. Let us define $\alpha_{max} = \max_{j=1...N} |\alpha_j|$, and x_{max} is the center of the MQ with coefficient α_{max}. Here, we will focus on the following observations from that paper:

- For fixed N, the magnitude of the spurious oscillations grow with increasing c and δ.
- If $\delta = 0$, α_{max} takes place in the neighbourhood of the boundaries.
- If $\delta > 0$, α_{max} takes place in the neighbourhood of x_0.

We will exploit these observations to detect jump discontinuities in piecewise smooth functions, based on $x_{max} \approx x_0$. Since the $\{|\alpha_j|\}$ tend to form a 'peak' around the discontinuity, the full-width at half maximum ($FWHM$) of it can be used to estimate the error:

$$|x_0 - x_{max}| \lesssim FWHM \tag{7}$$

We can devise a scheme to improve x_{max} iteratively:

MQ Discontinuity Detection Algorithm

- Define $I_0 = [a_0, b_0] := [a, b]$, N equispaced MQs $\Xi_0 = \{a_0 = x_1^{(0)}, ..., x_N^{(0)} = b_0\}$, and $c = c_0 > 0$; and compute and store $[\phi^{c_0}]_0^{-1}$ (the matrix in (6)).
- At iteration k:

1. Find $\alpha_{max}^{(k)} = \max_{j=1,...,N} |[\phi^{c_k}]_k^{-1}[f(x_1^{(k)}), ..., f(x_N^{(k)})]^T|$.
2. Estimate $FWHM_k$.
3. If $x_{max}^{(k)} - a_k < FWHM_k$ or $b_k - x_{max}^{(k)} < FWHM_k$, then let $x_0 \approx x_{max}$. END.
4. Let $a_{k+1} = \max(a_k, x_{max}^{(k)} - FWHM_k)$ and $b_{k+1} = \min(b_k, x_{max}^{(k)} + FWHM_k)$.
5. Let $\beta_{k+1} = (b_{k+1} - a_{k+1})/(b_k - a_k), 0 < \beta_{k+1} < 1$, and $c_{k+1} = \beta_{k+1} c_k$.
6. Then, $[\phi^{c_{k+1}}]_{k+1}^{-1} = [\phi^{c_k}]_k^{-1}/\beta_{k+1}$.
7. Let $I_{k+1} = [a_{k+1}, b_{k+1}]$ and $\Xi_{k+1} = \{a_{k+1} = x_1^{(k+1)}, ..., x_N^{(k+1)} = b_{k+1}\}$
8. Iterate until the desired precision is reached or $FWHM_k \approx (b_k - a_k)/2$. (The condition number is the same throughout the iterations).

3 Solving linear NDEs by Kansa's method

Kansa modified the MQ interpolation scheme to solve PDEs [5],[6]. Kansa's method was adapted to smooth DDEs in [1]. Consider the following linear NDE:

$$y'(x) - p(x)y(x) - q(x)y[x - \tau(x)] - r(x)y'[x - \tau(x)] = s(x) \text{ if } x \in [a, b] \quad (8)$$

$$y(x) = h(x) \text{ if } x \leq a \quad (9)$$

It will be convenient to split (9) into a NDE and an ODE

$$y'(x) - p(x)y(x) - q(x)y[x - \tau(x)] - r(x)y'[x - \tau(x)] = s(x) \quad \text{if } x - \tau(x) > a \quad (10)$$

$$y'(x) - p(x)y(x) = q(x)h[x - \tau(x)] + r(x)h'[x - \tau(x)] + s(x) \quad \text{if } x - \tau(x) < a \quad (11)$$

$$y(a) = h(a) \quad (12)$$

We seek an approximate solution to (10)-(12) in the form:

$$y(x) = \sum_{j=1}^{N} \alpha_j \phi_j^c(x - x_j) \quad (13)$$

In order to solve for the coefficients, (10)-(12) are enforced over (13). For $i = 1, ..., N$:

$$\sum_{j=1}^{N} \{\phi_j^{c\prime}(x_i - x_j) - p(x_i)\phi_j^c(x_i - x_j) - q(x_i)\phi_j^c(x_i - \tau(x_i) - x_j) - \quad (14)$$
$$-r(x_i)\phi_j^{c\prime}(x_i - \tau(x_i) - x_j)\}\alpha_j = s(x_i) \text{ if } x_i - \tau(x_i) > a$$

$$\sum_{j=1}^{N}\{\phi_j^{c\prime}(x_i - x_j) - p(x_i)\phi_j^c(x_i - x_j)\}\alpha_j = \quad (15)$$

$$= q(x_i)h[x_i - \tau(x_i)] + r(x_i)h'[x_i - \tau(x_i)] + s(x_i) \text{ if } x_i - \tau(x_i) \leq a$$

$$\sum_{j=1}^{N}\alpha_j\phi_j^c(x_i - x_j) = h(a) \text{ if } x_i = a \quad (16)$$

This scheme can be extended, in a straighforward manner, to the case of instances of y and y' being present with different delays. For nonlinear NDEs, the system of collocation equations above is nonlinear and a nonlinear solver must be used to find a root. Finally, an important yet open issue in Kansa's method is the optimal number and location of RBF centers/collocation nodes. A recommended approach is the residual subsampling algorithm (RSA) from [3] (see also [1]).

For a general NDE containing several discontinuities in a finite interval, we first try to find an estimate of the two leftmost discontinuities by looking at the MQ coefficients. Next, we define a subinterval which contains just the first one (x_0^I), and apply the Algorithm in Section 2 to accurately locate it. Then Kansa's method is used to approximate the discontinuity-free solution up to x_0^I. We then repeat the procedure to bracket and find the second discontinuity x_0^{II} -whereby we use the previous MQ approximation as 'history function'- and so on.

4 A Numerical Example

The following NDE is also used as a demo in MATLAB's NDE solver *ddeNsd* [10]. It is a crafted example with a non-trivial exact solution and known discontinuities which allows us to check our method.

$$\begin{cases} y'(t) = y'(t-2), & t \geq 0 \\ y(t) = h(t) = (t+1)^5, & t < 0. \end{cases} \quad (17)$$

The exact solution is $y_{EX}(t) = [t] + (t - [t])^5$, $t \geq 0$. It has a jump discontinuity at $t = 0$ and first-derivative discontinuities at $t = n$ for $n \geq 0$. For simplicity, (17) was solved in each discontinuity-free subinterval using $N = 80$ equispaced nodes. Equispaced nodes are a suboptimal distribution, and results can be expected to improve by orders of magnitude using the RSA adaptive algorithm, as shown in [1].

References

1. F. Bernal and G. Gutierrez, *Solving delay differential equations through RBF collocation*. Adv. Appl. Math. Mech. **1** 257-272 (2009).

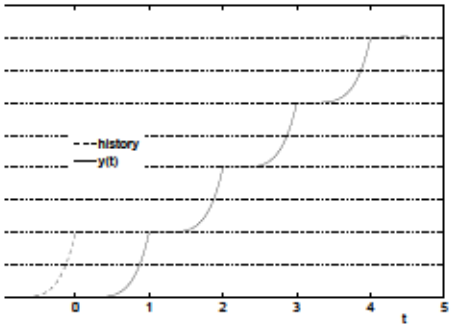

Fig. 1. The exact solution of the NDE (17). The MQ approximation cannot be distinguished with the naked eye.

2. T. Erneux, *Applied delay differential equations*. Springer (2009).
3. T. A. Driscoll and A. Heryudono, *Adaptive residual subsampling methods for radial basis function interpolation and collocation problems*, Computers Math. Appl. **53**, 927-939 (2007).
4. J.H. Jung, *A note on the Gibbs phenomenon with multiquadric radial basis functions*, Applied Numerical Mathematics **57**, Issue 2, 213-229 (2007).
5. 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**, 127-145 (1990).
6. 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**, 147-161 (1990).
7. W.R. Madych, *Miscellaneous error bounds for multiquadric and related interpolators*, Comput. Math. Appl. **24**, 121-38 (1992).
8. R. Schaback, *Error estimates and condition numbers for radial basis function interpolation*, Adv. Comput. Math. **3**, 251-264 (1995).
9. L.F. Shampine, *Solving ODEs and DDEs with residual control*, Appl. Numer. Math., **52**, 113-127 (2005).
10. L.F. Shampine, *Dissipative approximations to neutral DDEs*, Appl. Math. Comput. **203**, 641-648 (2008).

Table 1. Approximate discontinuities, error estimations and accuracy for the example. $RMS(\epsilon)$ stands for the root mean square of the error to the exact solution

Discontinuity	Exact location	MQ Approximation	$FWHM$	Subinterval	RMS(ϵ)
I	1.0	0.99999703	2.0×10^{-6}	$[a, x_0^I)$	6.50×10^{-7}
II	2.0	1.99999931	1.2×10^{-6}	(x_0^I, x_0^{II})	7.46×10^{-6}
III	3.0	3.00000021	1.0×10^{-6}	(x_0^{II}, x_0^{III})	1.73×10^{-5}
IV	4.0	4.00000070	9.9×10^{-7}	(x_0^{III}, x_0^{IV})	2.95×10^{-5}
				$(x_0^{IV}, b]$	4.36×10^{-7}