Vector-Matrix Implementation of the Integral Method of Recovery of Input Signals of Nonlinear Dynamic Systems

https://doi.org/10.31713/MCIT.2020.13

Anatolii Verlan
Pukhov Institute for Modelling in Energy Engineering
Kyiv, Ukraine
afverlan@gmail.com

Volodymyr Fedorchuk
Department of Informatics
Kamianets-Podilskyi National Ivan Ohiienko University
Kamianets-Podilskyi, Ukraine
fedvlad@kpnu.edu.ua

Vitalii Ivaniuk
Department of Informatics
Kamianets-Podilskyi National Ivan Ohiienko University
Kamianets-Podilskyi, Ukraine
wivanyuk@kpnu.edu.ua

Vadym Ponedilok
Department of Informatics
Kamianets-Podilskyi National Ivan Ohiienko University
Kamianets-Podilskyi, Ukraine
ponedilok.vadym@kpnu.edu.ua

Abstract— **In the article the method of signal reconstruction at the input of nonlinear dynamic objects based on the application of vector-matrix approach to solving polynomial Volterra integral equations of the first kind of the second degree using a differential regularization operator is studied.**

Keywords—signal recovery, nonlinear dynamic objects, polynomial integral operator, Volterra equation of the first kind, vector-matrix method.

I. INTRODUCTION

The inverse problems of the dynamic usually reduced to solving Volterra integral equations of the first kind [2, 3, 6]. In the case of nonlinear dynamical systems, the problem is to determine the input effect $\xi(t)$ based on a given response $y(t)$ and a mathematical model in the form of a polynomial Volterra integral equation of the first kind, in particular the second degree [1, 7, 8, 9]:

$$\int_0^t K_1(t,s)\xi(s)ds + \int_0^t\int_0^t K_2(t,s_1,s_2)\xi(s_1)\xi(s_2)ds_1ds_2 = y(t)\ , \quad (1)$$

where $K_1(t,s)$, $K_2(t,s_1,s_2)$ – Volterra kernel, t — time of transition process.

The classic approach to solving this problem is the use of quadrature methods [2, 5, 8, 9]. Since the problem is incorrect and experimental data are usually given with inaccuracy, it is necessary to use regularization methods. Such methods can be: methods of reduction to a correct mathematical model, in particular the method of constructing the integral Volterra equation of the second kind (it is effective if there is a possibility of exact differentiation of the kernel and the right part); methods of regularization — the introduction of regularization parameters that apply to both the whole model and its parts [4].

It should be noted that the numerical implementation of this approach is accompanied by a significant number of computational procedures [2, 5].

That is, the development of new, computationally efficient algorithms for solving polynomial integral equations (1) based on regularization algorithms for recovery signal at the input of nonlinear dynamic objects is a topical problem.

II. MAIN PART

A. *Regularization of a polynomial equation*

An effective approach to solving the problem of recovery of signals that pass through nonlinear dynamic objects is the use of differential regularization operators [4]. The use of a first-order differential regularization operator to equation (1) gets reduced to the solution of the following integro-differential equation:

$$\begin{gathered}\alpha\frac{dx}{dt} + \int_0^t K_1(t,s)x(s)ds + \\ + \int_0^t\int_0^t K_2(t,s_1,s_2)x(s_1)x(s_2)ds_1ds_2 = y(t),\end{gathered} \quad (2)$$

where α – is a regularization parameter, x – the approximate value of the required function.

B. *Algebraic approximation of a polynomial integro-differential equation*

Applying to (2) the quadrature and difference formulas [4], we obtain:

$$\begin{gathered}
\alpha\frac{x(t_i)-x(t_{i-1})}{h}+A_{1,0}K_1(t_i,t_0)x(t_0)+\\
+\sum_{j=1}^{i-1}A_{1,j}K_1(t_i,t_j)x(t_j)+A_{1,i}K_1(t_i,t_i)x(t_i)+\\
+A_{2,00}K_2(t_i,t_0,t_0)x(t_0)x(t_0)+\\
+\sum_{g=1}^{i-1}A_{2,0g}K_2(t_i,t_{0,},t_g)x(t_0)x(t_g)+\\
+A_{2,0i}K_2(t_i,t_0,t_i)x(t_0)x(t_i)+\\
+\sum_{j=1}^{i-1}A_{2,j0}K_2(t_i,t_{j,},t_0)x(t_j)x(t_0)+\\
+\sum_{j=1}^{i-1}\sum_{g=1}^{i-1}A_{2,jg}K_2(t_i,t_{j,},t_g)x(t_j)x(t_g)+\\
+\sum_{j=1}^{i-1}A_{2,ji}K_2(t_i,t_{j,},t_i)x(t_j)x(t_i)+\\
+A_{2,i0}K_2(t_i,t_i,t_0)x(t_i)x(t_0)+\\
+\sum_{g=1}^{i-1}A_{2,ig}K_2(t_i,t_{i,},t_g)x(t_i)x(t_g)+\\
+A_{2,ii}K_2(t_i,t_i,t_i)x(t_i)x(t_i)=y(t_i),
\end{gathered}\tag{3}$$

where $i=\overline{0..n}$, $h=t_i-t_{i-1}$, A_1 , A_2 – coefficients of quadrature formulas.

It can be seen that expression (3) is determined by the complexity of the description and the significant number of computational actions. To overcome these difficulties, it is advisable to use a vector-matrix approach with the reduction of all operations in (3) to element-by-element addition and multiplication [5].

C. Vector-matrix method

Consider separately homogeneous operators of the first and second degree. The coefficients A_1 , A_2 are represented in vector-matrix form. In the case of the operator of the first degree, the coefficients A_1 are determined by a vector that has a different form, depending on the type of quadrature formula used:

– formula of rectangles:

$$A_1^R=h\begin{pmatrix}1 & 1 & 1 & \dots & 1 & 1 & 0\end{pmatrix};$$

– trapezoid formula:

$$A_1^T=h\begin{pmatrix}\frac{1}{2} & 1 & 1 & \dots & 1 & 1 & \frac{1}{2}\end{pmatrix};$$

– Simpson formula:

$$A_1^S=h\begin{pmatrix}\frac{1}{3} & \frac{4}{3} & \frac{2}{3} & \dots & \frac{2}{3} & \frac{4}{3} & \frac{1}{3}\end{pmatrix}.$$

To approximate the second-degree operator, matrices that determine the coefficients A_2 are obtained:

– formula of rectangles:

$$A_2^R=h^2\begin{pmatrix}1&1&1&\dots&1&1&0\\1&1&1&\dots&1&1&0\\1&1&1&\dots&1&1&0\\\dots&\dots&\dots&\dots&\dots&\dots&\dots\\1&1&1&\dots&1&1&0\\1&1&1&\dots&1&1&0\\0&0&0&\dots&0&0&0\end{pmatrix};$$

– trapezoid formula:

$$A_2^T=h^2\begin{pmatrix}\frac{1}{4}&\frac{1}{2}&\frac{1}{2}&\dots&\frac{1}{2}&\frac{1}{2}&\frac{1}{4}\\\frac{1}{2}&1&1&\dots&1&1&\frac{1}{2}\\\frac{1}{2}&1&1&\dots&1&1&\frac{1}{2}\\\dots&\dots&\dots&\dots&\dots&\dots&\dots\\\frac{1}{2}&1&1&\dots&1&1&\frac{1}{2}\\\frac{1}{2}&1&1&\dots&1&1&\frac{1}{2}\\\frac{1}{4}&\frac{1}{2}&\frac{1}{2}&\dots&\frac{1}{2}&\frac{1}{2}&\frac{1}{4}\end{pmatrix};$$

– Simpson formula:

$$A_2^S=h^2\begin{pmatrix}\frac{1}{9}&\frac{4}{9}&\frac{2}{9}&\dots&\frac{2}{9}&\frac{4}{9}&\frac{1}{9}\\\frac{4}{9}&\frac{16}{9}&\frac{8}{9}&\dots&\frac{8}{9}&\frac{16}{9}&\frac{4}{9}\\\frac{2}{9}&\frac{8}{9}&\frac{4}{9}&\dots&\frac{4}{9}&\frac{8}{9}&\frac{2}{9}\\\dots&\dots&\dots&\dots&\dots&\dots&\dots\\\frac{2}{9}&\frac{8}{9}&\frac{4}{9}&\dots&\frac{4}{9}&\frac{8}{9}&\frac{2}{9}\\\frac{4}{9}&\frac{16}{9}&\frac{8}{9}&\dots&\frac{8}{9}&\frac{16}{9}&\frac{4}{9}\\\frac{1}{9}&\frac{4}{9}&\frac{2}{9}&\dots&\frac{2}{9}&\frac{4}{9}&\frac{1}{9}\end{pmatrix}.$$

In fig. 1 and fig. 2 a structural representation of the initial data of these operations for different homogeneous operators [5] is given. The program analogy of such a representation for the implementation of the operator of the first degree has the form: `sum(A.*K(1:j).*X1)`, where `A` is the vector that determines the coefficients of the quadrature formula, according to the above; `K` is the vector of kernel values according to the discretization of the time variable; `X1` is the vector of input influence values. This approach is used for the numerical implementation of the following components (3):

$$\sum_{j=1}^{i-1} A_{1,j}K_1(t_i,t_j)x(t_j),\ \sum_{g=1}^{i-1} A_{2,0g}K_2(t_i,t_{0,},t_g)x(t_0)x(t_g),$$

$$\sum_{j=1}^{i-1} A_{2,j0}K_2(t_i,t_{j,},t_0)x(t_j)x(t_0),\ \sum_{j=1}^{i-1} A_{2,ji}K_2(t_i,t_{j,},t_i)x(t_j),$$

$$\sum_{g=1}^{i-1} A_{2,ig}K_2(t_i,t_{i,},t_g)x(t_g).$$

To implement the second degree operator, the program analogy has the form `sum(sum(A.*K(1:j,1:j) .*X1.*X2))`, where `A` and `K` are matrices, `X1` is a matrix consisting of identical rows of vector *x*; `X2` is a matrix consisting of identical columns of the vector *x*. The following representation is used for numerical implementation:

$$\sum_{j=1}^{i-1}\sum_{g=1}^{i-1} A_{2,jg}K_2(t_i,t_{j,},t_g)x(t_j)x(t_g).$$

This approach greatly simplifies the software implementation of multidimensional operators, as it allows easy generalization to the multidimensional case [5].

$$\sum \boxed{\text{A}}\boxed{\text{K}}\boxed{\text{X1}}$$

Figure 1. Structural representation of the vector-matrix approach for the first degree operator

$$\sum\ \sum \boxed{\text{A}}\boxed{\text{K}}\boxed{\text{X1}}\boxed{\text{X2}}$$

Figure 2. Structural representation of the vector-matrix approach for the second degree operator

The suggested approach is studied in model experiments. Applying of the formula of the left rectangles leads to inaccurate results, as in the vector-matrix representation in the *i*-th place there is zero, which significantly affects the final result. Simpson's formula is the most accurate, but it requires more calculations than the rectangles and trapezoids formulas, which involve additional partitioning to find intermediate values at the interpolation points. The trapezoidal formula is the best in terms of "accuracy – complexity of implementation".

It should be noted that the accuracy of the implementation of integrated models depends only on the selected formula, modeling step, kernel type, and does not depend on the dimension of the operator. Therefore, the choice of the best method should be based primarily on the analysis of the kernels of Volterra's integral polynomial equation.

Depending on the type of kernel, it is possible to build computational algorithms based on the use of different quadrature formulas, which are applied separately to each dimension of the multidimensional integral operator. Moreover, not only the considered formulas (rectangles, trapezoids, Simpson) can be used, but also the formulas are built on the basis of a combination of quadrature formulas of Newton-Cotes of higher orders. This approach will allow to obtain different cubature formulas and expand the set of algorithms for approximation of integral models with finite sums and will allow you to choose the best formula depending on the original problem.

The suggested approach makes it possible to parallelize computational algorithms, which significantly speeds up the numerical implementation of integrated operators. Table 1 shows the order of complexity of the numerical implementation of polynomial integral operators depending on the number of possible parallel flows.

Representation of quadrature and cubature formulas in vector-matrix form provides opportunities to take advantage of matrix-oriented application packages (Matlab, Octave, Scilab), which feature a high speed of matrix operations.

TABLE I. DIFFICULTY OF POLYNOMIAL INTEGRATED OPERATORS IMPLEMENTATION

	First degree operator	**Second degree operator**
Usual approach	$O(n)$	$O^2(n)$
Vector-matrix	$\frac{O(n)}{kp}$	$\frac{O^2(n)}{kp}$
n – number of discretization points, *kp* – number of parallel flows		

D. *Method for solving a polynomial integro-differential equations*

Rewrite (3), grouping the coefficients relative to the degrees of the desired $x(t_i)$:

$$\begin{gathered}
A_{2,ii}K_2(t_i,t_i,t_i)x(t_i)x(t_i)+ \\
+A_{1,i}K_1(t_i,t_i)x(t_i)+A_{2,0i}K_2(t_i,t_0,t_i)x(t_0)x(t_i)+ \\
+\sum_{j=1}^{i-1} A_{2,ji}K_2(t_i,t_{j,},t_i)x(t_j)x(t_i)+ \\
+A_{2,i0}K_2(t_i,t_i,t_0)x(t_i)x(t_0)+ \\
+\sum_{g=1}^{i-1} A_{2,ig}K_2(t_i,t_{i,},t_g)x(t_i)x(t_g)+\alpha\frac{x(t_i)}{h}+ \\
+A_{1,0}K_1(t_i,t_0)x(t_0)+\sum_{j=1}^{i-1} A_{1,j}K_1(t_i,t_j)x(t_j)+ \\
+A_{2,00}K_2(t_i,t_0,t_0)x(t_0)x(t_0)+ \\
+\sum_{g=1}^{i-1} A_{2,0g}K_2(t_i,t_{0,},t_g)x(t_0)x(t_g)+ \\
+\sum_{j=1}^{i-1} A_{2,j0}K_2(t_i,t_{j,},t_0)x(t_j)x(t_0)+
\end{gathered}$$

$$+\sum_{j=1}^{i-1}\sum_{g=1}^{i-1} A_{2,jg}K_2(t_i,t_j,t_g)x(t_j)x(t_g)+ \\ +\alpha\frac{x(t_{i-1})}{h}-y(t_i)=0. \quad (4)$$

We introduce the notation:

$$F_{2,i}=A_{2,ii}K_2(t_i,t_i,t_i), \quad (5)$$

$$F_{1,i}=A_{1,i}K_1(t_i,t_i)+A_{2,0i}K_2(t_i,t_0,t_i)x(t_0)+ \\ +\sum_{j=1}^{i-1}A_{2,ji}K_2(t_i,t_j,t_i)x(t_j)+A_{2,i0}K_2(t_i,t_i,t_0)x(t_0)+ \\ +\sum_{g=1}^{i-1}A_{2,ig}K_2(t_i,t_i,t_g)x(t_g), \quad (6)$$

$$F_{0,i}=A_{1,0}K_1(t_i,t_0)x(t_0)+\sum_{j=1}^{i-1}A_{1,j}K_1(t_i,t_j)x(t_j)+ \\ +A_{2,00}K_2(t_i,t_0,t_0)x(t_0)x(t_0)+ \\ +\sum_{g=1}^{i-1}A_{2,0g}K_2(t_i,t_0,t_g)x(t_0)x(t_g)+ \\ +\sum_{j=1}^{i-1}A_{2,j0}K_2(t_i,t_j,t_0)x(t_j)x(t_0)+ \\ +\sum_{j=1}^{i-1}\sum_{g=1}^{i-1}A_{2,jg}K_2(t_i,t_j,t_g)x(t_j)x(t_g)-y(t_i)=0, \quad (7)$$

Then (4) taking into account the notation (5), (6), (7) has the form:

$$F_{2,i}x_i^2+F_{1,i}x_i+F_{0,i}=0. \quad (8)$$

The quadratic equations of system (8) are solved sequentially, and the following algorithm is used to select one of the roots: if $|x_{i-1}-x_{i_1}|\geq|x_{i-1}-x_{i_2}|$, then $x_i=x_{i_2}$, otherwise $x_i=x_{i_1}$. In the application of iterative methods, the root of the previous equation is taken as the initial approximation [4].

III. Model experiments

The study of this approach effectiveness was carried out using the method of model experiments. The model is considered as an example

$$\alpha\frac{dx(t)}{dt}+\int_0^t e^{-s}x(t-s)ds+ \\ +\int_0^t\int_0^t e^{-(s_1+s_2)}x(t-s_1)x(t-s_2)ds_1ds_2=y(t).$$

The results of computational experiments are presented in fig. 3. (graphs of registered, restored and accurate signals).

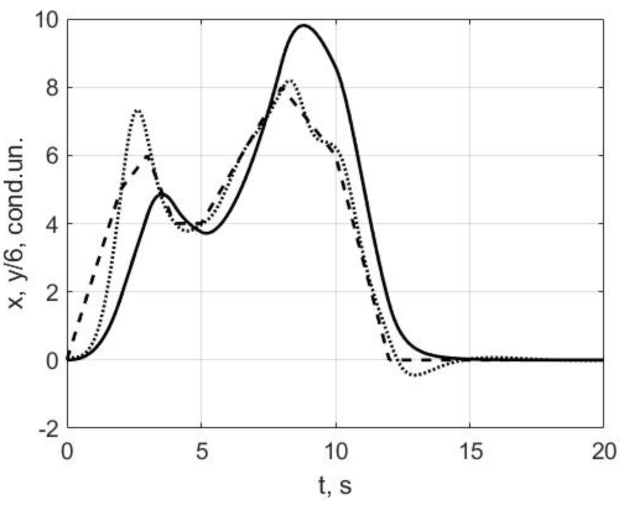

Figure 3. Graphs of the input (– – – – – accurate, ········· – calculated) and output (———) signals

Conclusion

Thus, to increase the noise immunity of the signal recovery process at the input of nonlinear dynamic objects and reduce the number of computational procedures, a regularization method for solving polynomial Volterra integral equations using a vector-matrix approach and a differential regularization operator is developed.

References

[1] Apartsin A. S. Multilinear Volterra integral equations of the first kind: elements of theory and numerical methods. ISU News, mathematics series. 2007. № 1. P. 13–42.

[2] Verlan A. F., Sizikov V. S. Integral equations: methods, algorithms, programs. Kyiv, 1986. 544 p.

[3] Doyle F. J., Pearson R. K., Ogunnaike B. A. Identification and control using Volterra models. Germany, 2002. P. 314.

[4] Ivaniuk V., Ponedilok V. Method of restoration of input signals of nonlinear dynamic object with destributed parameters. Mathematical and computer modeling. Series: Technical Sciences: Coll. scientific works. Kamianets-Podilskyi, 2018. Issue. 18. P. 65–73.

[5] Ivanyuk V. A., Fedorchuk V. A. Vector-matrix method of numerical implementation of the polynomial integral Volterra operators. Mathematical and computer modeling. Series: Technical Sciences: Coll. scientific works. Kamianets-Podilskyi, 2019. Issue. 20. P. 40–50.

[6] Pupkov K. A., Kapalin V. I., Yushchenko A. S. Functional series in the theory of nonlinear systems. Moscow, 1976. 448 p.

[7] Solodusha S. V. Numerical modeling of heat transfer dynamics by the modified quadratic Volterra polynomial. Computing technologies. 2013. Vol. 18. № 2. P. 84–94.

[8] Solodusha S. V. Modeling Heat Exchangers by Quadratic Volterra Polynomials. Automation and Remote Control. 2014. Vol. 75. № 1. P. 87–94.

[9] Solodusha S. V., Yaparova N. M. Numerical Solving an Inverse Boundary Value Problem of Heat Conduction Using Volterra Equations of the First Kind. Numerical Analysis and Applications. 2015. Vol. 18. № 3. P. 267–274.