World Bulletin of Social Sciences (WBSS)
Available Online at: https://www.scholarexpress.net
Vol. 41, December 2024
ISSN: 2749-361X

NEW METHODS FOR SOLVING INTEGRAL EQUATIONS.

Mamadaliev Foziljon Abdullaevich
- Doctor of Physics – mathematical sciences,
holder of the title "Excellent Higher Education"
of the Republic of Uzbekistan .
Kokand branch of Tashkent State Technical
University named after Islam Karimov.

Email: fozil . bek .80@ mail . ru
Mobile: +99897 590 98 77

Article history:
Received: 28th August 2024
Accepted: 26th September 2024

Abstract:
This article discusses new methods for solving integral equations, using which many problems are solved; without using this method, there are no optimal ways to find their answers.

Keywords: Integral , integral equations, new solutions, new approach, key role, function.

Integral equations, equations in which the unknown function is under the integral sign, play a key role in various fields of science and engineering, from physics and mechanics to economics and biology. Traditional methods for solving such methods, such as the method of successive approximations or the Fredholm method, are often effective and not always efficient, especially for complex integral methods. In recent years, new approaches have emerged that significantly improve the efficiency and accuracy of solutions. This article will briefly review some of them.

1. Collocation-based methods:

Collocation methods are numerical methods that approximate an unknown function using linear mathematical basis functions. The coefficients are calculated by this method using equations at a set of collocation points. Modern collocation methods often use orthogonal polynomials (e.g., Chebyshev or Legendre polynomials) as basis functions, which ensures high accuracy and stability. The advantages of this approach are the relative simplicity of implementation, the possibility of an efficient solution, and high dimensionality.

The Fredholm integral equation of the second kind is considered:

$$y(x) = x + \lambda\int_0^1 xy(t)\,dt$$

where λ is a parameter and y(x) is an unknown function. We solve this equation by the collocation method using linear approximation.

1. Selection of basis functions:

chosen as basis functions are: $\varphi_1(x) = 1$ and $\varphi_2(x) = x$. Then the approximation solutions can be written in the video:

$y(x) \approx c_1\varphi_1(x) + c_2\varphi_2(x) = c_1 + c_2x$

where c_1 and c_2 are unknown coefficients.

2. Selecting collection points:

You are pregnant with two collocation points: $x_1 = 0$ and $x_2 = 1$. This is the minimum number of points for a linear approximation.

3. Substitution into the equation:

Let us substitute the approximation y(x) into the original equation and evaluate it at the collocation points:

For $x_1 = 0$:

$c_1 + c_2(0) = 0 + \lambda\int_0^1 (0) * (c_1 + c_2t)\,dt = 0 \Rightarrow c_1 = 0$

For $x_2 = 1$:

$c_1 + c_2(1) = 1 + \lambda\int_0^1 t(c_1 + c_2t)\,dt$

Let's substitute $c_1 = 0$:

$c_2 = 1 + \lambda\int_0^1 c_2t^2\,dt = 1 + \lambda c_2[t^3/3]_0^1 = 1 + \lambda c_2/3$

4. Definition of the refinement system:

We obtain the equation for c_2:

$c_2 = 1 + \lambda c_2/3$

$c_2(1 - \lambda/3) = 1$

$c_2 = 1/(1 - \lambda/3) = 3/(3 - \lambda)$

5. Solution:

Thus, the approximate solution of the integral equation has the form:

$y(x) \approx [3 / (3 - \lambda)]x$

Examination:

Let's substitute this solution back into the original equation:

$[3 / (3 - \lambda)]x = x + \lambda\int_0^1 x[3 / (3 - \lambda)]t\,dt$

$[3 / (3 - \lambda)]x = x + \lambda[3 / (3 - \lambda)]x\,[t^2/2]_0^1$

$[3 / (3 - \lambda)]x = x + \lambda[3 / (3 - \lambda)]x\,(1/2)$

$[3 / (3 - \lambda)]x = x\,[1 + \lambda/(2(3-\lambda))]$

This is the ratio of the indices only when approaching equality. The accuracy of the approximation depends on the choice of collocation points and the number of basis functions. To determine the accuracy, you can use a larger number of collocation points and more complex basis functions (for example, higher-order

World Bulletin of Social Sciences (WBSS)
Available Online at: https://www.scholarexpress.net
Vol. 41, December 2024
ISSN: 2749-361X

polynomials). This example is the basic method of collocations. More complex algorithms and software are used to solve problems of determining accuracy.

2. Methods based on Galerkin approximations:

Galerkin methods, similar to collocation methods, also use approximation of the unknown function using basis functions. However, instead of two equations at the bottom points, they minimize the remainder of the equation in the integral sense. This leads to a system of linear algebraic models that can be solved by standard methods. The use of orthogonal basis functions makes this method very effective for solving integral models of various types.

Let us solve the Fredholm integral equation of the second kind using the Galerkin method :

$$y(x) = 1 + \lambda\int_0^1 (x + t)y(t)\,dt$$

where λ is a parameter and $y(x)$ is an unknown function.

1. Selection of basis functions:

For simplicity, we choose the basis functions as polynomials: $\varphi_1(x) = 1$ and $\varphi_2(x) = x$. We write the approximate solution as:

$y(x) \approx c_1\varphi_1(x) + c_2\varphi_2(x) = c_1 + c_2x$

2. Formulation of the problem in weak form:

We multiply the original equation by the weight functions (in the Galerkin method, the weight functions coincide with the basis functions) and integrate over the interval [0, 1]:

$\int_0^1 (y(x) - 1 - \lambda\int_0^1 (x + t)y(t)\,dt\,)\varphi_i(x)\,dx = 0$, where $i = 1, 2$

3. Substitution of approximate solution:

Let us substitute the approximate solution $y(x) \approx c_1 + c_2x$ into the equation:

$\int_0^1 (c_1 + c_2x - 1 - \lambda\int_0^1 (x + t)(c_1 + c_2t\,)\,dt\,)\varphi_i(x)\,dx = 0$

4. Solution for $i = 1$ ($\varphi_1(x) = 1$):

$\int_0^1 (c_1 + c_2x - 1 - \lambda\int_0^1 (x + t)(c_1 + c_2t\,)\,dt\,)\,dx = 0$

Let's divide the integrals:

$\int_0^1 (c_1 + c_2x - 1)\,dx - \lambda\int_0^1 \int_0^1 (x + t)(c_1 + c_2t\,)\,dtdx = 0$

Let's calculate the integrals:

$[\,c_1x + c_2x^2/2 - x]_0^1 - \lambda\int_0^1 [\,xc_1 + xc_2/2 + tc_1 + tc_2/2]_0^1\,dx = 0$

$c_1 + c_2/2 - 1 - \lambda\int_0^1 (c_1 + c_2/2 + c_1 + c_2t\,)\,dt = 0$

$c_1 + c_2/2 - 1 - \lambda[c_1 + c_2/2 + c_1 + c_2/2] = 0$

$c_1 + c_2/2 - 1 - \lambda(2c_1 + c_2) = 0$

5. Solution for $i = 2$ ($\varphi_2(x) = x$):

$\int_0^1 (c_1 + c_2x - 1 - \lambda\int_0^1 (x + t)(c_1 + c_2t\,)\,dt\,)\,xdx = 0$

After similar calculations of integrals (more complicated), we obtain the second equation of the system. The details of the calculations are omitted here due to their cumbersomeness.

6. Solution of the system of equations:

As a result, we obtain a system of two linear algebraic equations with two unknowns c_1 and c_2. Having solved this system (for example, by the Cramer or Gauss method), we find the values of c_1 and c_2.

7. Approximation of the solution:

Substituting the found values of c_1 and c_2 into $y(x) \approx c_1 + c_2x$, we obtain an approximate solution of the integral equation.

Important to note: Calculating the integrals in step 5 is quite complex, and it is better to use a mathematical package (for example, Mathematica , Maple, or Python with the SymPy library) to obtain accurate values. This example shows the general approach of the Galerkin method. In more complex problems, the choice of basis functions and the number of equations in the system will be determined by the complexity of the problem and the required accuracy of the solution.

3. Methods based on neural networks:

Recently, there has been interest in using neural networks to solve integral methods. This approach is based on the ability of neural networks to approximate complex neural functions with an upper bound. The neural network is trained on examples of solving an integral equation, and after training, it can predict the solution for new input data. This method is especially effective for mathematical solutions, while other methods are not accurate.

Unfortunately, it is very difficult to directly *solve* the integral equation problem using neural networks. Solving with neural networks is an iterative process that requires specialized software (e.g. TensorFlow , PyTorch) and computing resources. I do not have access to such an environment.

However, I can describe the steps needed to solve such a problem using neural networks:

1. Formulation of the problem:

Let us have an integral equation:

$y(x) = f(x) + \int_0^1 K(x, t)y(t)dt$

Where:

- $y(x)$ is an unknown function;
- $f(x)$ is a known function;
- $K(x, t)$ are known sources of the integral equation.

2. Approximation of the solution:

We will approximate the solution $y(x)$ using a neural network. There are different types of neural networks that can be used, for example:

- **Multilayer Perceptron (MLP):** A Simple and Versatile Choice.
- **Recurrent Neural Networks (RNN):** Suitable for solving problems with time series or persistent groups.

World Bulletin of Social Sciences (WBSS)
Available Online at: https://www.scholarexpress.net
Vol. 41, December 2024
ISSN: 2749-361X

- **Convolutional neural networks (CNN):** can be effective if the source K(x,t) has some structure.

In our example, it is preferable to use MLP. The input network will be the value x, the output will be the approximate value y(x).

3. functional losses:

It is necessary to determine the loss of the function that will be minimized during neural network training. It should reflect the difference between the left and right parts of the integral equation. Possible option:

$L = \int_0^1 (y\_pred (x) - f(x) - \int_0^1 K(x, t)\, y\_pred (t)\, dt)^2 dx$

where y_pred (x) is the output of the neural network.

4. Training the neural network:

Training a neural network to support minimization of the L function loss using gradient descent methods (e.g. Adam, RMSprop). Calculating the gradient will require countable integration.

5. Checking the result:

After training the neural network, it is necessary to check the quality of the obtained solution by comparing it with classical analytical solutions (if any) or other listed methods.

Code example (Python with TensorFlow / Keras - conceptual):

```
import tensorflow as tf

# Definition of neural network
model = tf.keras.Sequential ([
  tf.keras.layers.Dense (64, activation=' relu ', input_shape =(1,)),
  tf.keras.layers.Dense (64, activation=' relu '),
  tf.keras.layers.Dense (1)
])

# Definition of loss function (simplified version)
def loss_function ( y_true , y_pred ):
  # Here we need to implement numerical integration
  # ...
  return tf.reduce_mean ( tf.square ( y_true - y_pred ))
# Compilation models
model.compile (optimizer=' adam ', loss= loss_function )
# Generate training data
# ...
# Model training
model.fit ( x_train , y_train , epochs=100)
# Model checking
# ...
```

This code is just an example. The actual implementation will be much more complex and will require a deeper understanding of neural networks and complex integration methods. To solve a specific problem, you will need to adapt this code to a specific integral equation and select the appropriate neural network architecture.

I hope this description will help you understand how to solve an integral equation using neural networks. Remember that this is a complex task that requires significant computational resources and specialized knowledge.

4. Transformation-based methods:

The use of integral transforms (such as the Laplace or Fourier transform) allows the integral equation to be reduced to an algebraic or differential equation that is easier to solve. This approach is particularly effective for analysis with special types of kernels.

We solve the Volterra integral equation of the second kind using the Laplace transform:

$$\mathbf{y(x) = x + \int_0^x (x - t)y(t)\, dt}$$

1. Laplace transform:

Let's write the Laplace transform for both parts of the equation:

$\mathcal{L}\{y(x)\} = \mathcal{L}\{x\} + \mathcal{L}\{\int_0^x (x - t)y(t)\, dt \}$

Let $Y(s) = \mathcal{L}\{y(x)\}$. Then :

$Y(s) = 1/s^2 + \mathcal{L}\{\int_0^x (x - t)y(t)dt\}$

2. Convolution theorem:

The Laplace transform of the convolution of two functions is equal to the product of their Laplace transforms. In our case:

$\mathcal{L}\{\int_0^x (x - t)y(t)\, dt \} = \mathcal{L}\{x\} * \mathcal{L}\{y(x)\} = (1/s^2) * Y(s)$

3. Substitution and solution:

Let's substitute this into the equation:

$Y(s) = 1/s^2 + (1/s^2) * Y(s)$

Let us solve this equation for Y(s):

$Y(s) * (1 - 1/s^2) = 1/s^2$

$Y(s) * (s^2 - 1) / s^2 = 1/s^2$

$Y(s) = 1 / (s^2 - 1) = 1 / [(s - 1)(s + 1)]$

4. Inverse Laplace transform:

Let's decompose Y(s) into simple fractions:

$Y(s) = A / (s - 1) + B / (s + 1)$

Let's find A and B:

$A = \lim (s\to 1) [(s - 1)Y(s)] = \lim (s\to 1) [1 / (s + 1)] = 1/2$

$B = \lim (s\to -1) [(s + 1)Y(s)] = \lim (s\to -1) [1 / (s - 1)] = -1/2$

Thus:

$Y(s) = 1/2 * [1 / (s - 1) - 1 / (s + 1)]$

Now let's find the inverse Laplace transform:

$y(x) = \mathcal{L}^{-1}\{Y(s)\} = 1/2 * [e^x - e^{-x}] = \sinh (x)$

5. Solution:

Therefore, the solution to the integral equation is:

$y(x) = \sinh (x)$

World Bulletin of Social Sciences (WBSS)
Available Online at: https://www.scholarexpress.net
Vol. 41, December 2024
ISSN: 2749-361X

This example demonstrates the effectiveness of the transform method for solving certain types of integral equations. It is important to remember that the applicability of this method depends on the type of the kernel of the integral equation and the possibility of finding the inverse Laplace transform.

5. Hybrid methods:

Hybrid methods are also being developed that combine the advantages of different approaches. For example, combining the collocation method with optimization methods allows for efficient solution of problems with nonlinear integral equations.

Solving an integral equation analytically using hybrid methods is extremely difficult, since a "hybrid method" implies a combination of several methods, the choice of which depends on the specific equation. An analytical solution is usually possible only for simple equations and specific combinations of methods.

Therefore, I will demonstrate the concept of the hybrid method with an example using a combination **of the collocation method** and **the iteration method** . The full numerical solution will require software code and computing power.

Task:

Let us consider the Fredholm integral equation of the second kind:

$y(x) = x^2 + \int_0^1 (x + t)y(t)\, dt$

Hybrid Method (Collocation + Iteration):

1. **Iterative Method:** Let's start with the iterative process for an approximate solution. The iterative formula is:
   $y_{n+1}(x) = x^2 + \int_0^1 (x + t)y_n(t)\, dt$
   Let's start with the zero approximation: $y_0(x) = x^2$
2. **Collocation:** Instead of calculating the integral at each iteration, we will use the collocation method. We will choose two collocation points: $x_1 = 0$ and $x_2 = 1$. At each iteration, we will substitute the approximate solution into the equation at these points.
3. **Linear approximation:** Assume that the solution can be approximated by a linear function:
   $y_n(x) \approx a_n + b_n x$
4. **Iterations with collocation:**
   - **Iteration 1:** $y_0(x) = x^2$
     - $y_1(0) = 0^2 + \int_0^1 (0 + t)t^2 dt = 1/4$
     - $y_1(1) = 1^2 + \int_0^1 (1 + t)t^2 dt = 1 + 7/12 = 19/12$ We solve the system of linear equations: $a_1 + 0b_1 = 1/4$; $a_1 + b_1 = 19/12 \Rightarrow a_1 = 1/4$; $b_1 = 5/6$. $y_1(x) \approx 1/4 + 5/6x$
   - **Iteration 2:** use $y_1(x)$ to calculate $y_2(x)$, etc.
5. **Termination of iterations:** We continue the iteration process until the difference between successive approximations becomes small enough (the required accuracy is achieved).

Notes:

- This example demonstrates the concept of a hybrid method. To achieve high accuracy, more collocation points, more complex basis functions (not just linear ones), and probably a more sophisticated iterative method would be needed.
- Numerical integration will be required to calculate the integrals at each iteration, even using the collocation method.
- The solution of this problem in analytical form is impossible. The numerical result will be obtained after several iterations.
- More sophisticated hybrid methods may include adaptive collocation methods, Runge-Kutta methods for numerical integration, and other numerical optimization methods.

This example illustrates the basic idea of the hybrid approach. To obtain a numerical solution, an implementation in a programming language (e.g. Python with NumPy and SciPy) is required using numerical integration methods.

Conclusion:

The development of computer technology and mathematical methods has led to the emergence of new effective solutions to integral models. The above methods represent only a small part of the available approaches, and the choice of method depends on the type of integral equation and the requirements for the accuracy and efficiency of solutions. Further research in this area is aimed at developing even more effective and universal methods for solving integral algorithms capable of converters with increasingly complex problems in various scientific and engineering fields.