Advanced Mathematical Models & Applications
Vol.6, No.2, 2021, pp.89-105

REGULARIZED AND PRECONDITIONED CONJUGATE GRADIENT LIKE-METHODS METHODS FOR POLYNOMIAL APPROXIMATION OF AN INVERSE CAUCHY PROBLEM

Sudad M. Rasheed[1*], **Abdeljalil Nachaoui**[2], **Mudhafar F. Hama**[3], **Adil K. Jabbar**[3]

[1]Department of Mathematics, College of Education, University of Sulaimani, Sulaymaniyah, Iraq
[2]Laboratoire de Mathématiques Jean Leray UMR6629 CNRS/Université de Nantes, Nantes, France
[3]Department of Mathematics, College of Science, University of Sulaimani, Sulaymaniyah, Iraq

Abstract. In this paper, regularization combined with a preconditioning strategy is used to solve the ill-conditioned linear system obtained from the approximation of the inverse Cauchy problem for the Poisson equation in an arbitrary bounded domain. This approximation is based on the polynomial expansion (Liu & Kuo, 2016). The presented numerical results show that the considered methods produced stable numerical solutions. Moreover, the preconditioned regularization algorithms analyzed in this article require a very low computational effort and that therefore represents another advantage of the numerical methods used to solve the Cauchy problem.

Keywords: Inverse Cauchy problems, polynomial expansion, preconditionned linear systems, Conjugate Gradient like-methods.

AMS Subject Classification: 65N21, 41A10, 65F22, 65N35.

Corresponding author: Sudad M. Rasheed, Department of Mathematics, College of Education, University of Sulaimani, Kirkuk Road, 46001 Sulaymaniyah, Kurdistan Region, Iraq, Tel:+964 (770)196 5969,
e-mail: *sudad.rasheed@univsul.edu.iq*

Received: 13 June 2021; Revised: 29 July 2021; Accepted: 13 August 2021; Published: 24 August 2021.

1 Introduction

In this paper, we consider the inverse Cauchy problem for Poisson equation where the solution on one part of the boundary has to be determined from overdetermined data (Cauchy data) on another part is well-known to be severely ill-posed (Hadamard, 1953; Alessandrini et al., 2009; Belgacem, 2007).

This inverse problem has been intensively investigated by several researchers over the last half century, the reader can consult for example the following books (Choulli, 2009; Kabanikhin, 2012; Lavrent'ev, 2013; Isakov, 2017)

One of the first methods introduced as an approximation of the Cauchy problem is the quasi-reversibility method Latté & Lions (1969). Since then it has been widely applied in different fields and several variants have been developed, one can mention for example Bourgois (2006); Klibanov & Santosa (1991). Other methods, based on formulations in the form of an optimal control problems where a functional taking into account one of the conditions on the overdetermined part, have been developed Andrieux et al. (2006); Chakib & Nachaoui (2006); Kabanikhin & Karchevsky (2012). One of the most popular techniques is that of Tikhonov regularization (Berntsson et al., 2017; Cimetière et al., 2001; Chang et al., 2001; Kabanikhin & Karchevsky , 2012; Kabanikhin et al., 2013; Liu & Wei, 2013). Among the many numerical methods, the schemes based on iteration have also been developed previously by Kozlov et al. (1991), Jourhmane & Nachaoui (1996, 1999, 2002). The relaxation JN method introduced in the last papers drastically reduced the number of iterations required to achieve convergence. It was used in elasticity (Ellabib & Nachaoui, 2008; Marin & Johansson, 2010), and recently for Cauchy problem governed by Stocks equation (Chakib et al., 2018) and for the Helmholtz equation (Berdawood et al., 2020, 2021; Berdawood-Nachaoui et al., 2021). This relaxation JN method has been applied the to simulate the two-dimensional non linear elliptic problem (Essaouini et al., 2004; Essaouini & Nachaoui., 2004) and recently (Aboud et al., 2021) combined the domain decomposition method and the relaxation method to solve a Cauchy problems on inhomogenious material. In Nachaoui et al. (2021), efficient iterative domain decomposition like-methods was developed.

In contrast to the methods mentioned above, direct and/or mesh-free methods have been developed over the last two decades. Their particularity is that they can be implemented without resorting to iteration. Their efficiencies are independent of the shape of the mesh. Among these methods, the analytical methods that are highly effective but can only be used for particular forms of the domains in which problems are defined (Liu, 2011; Grigor'ev, 2018; Nachaoui & Nachaoui, 2021). Other mesh-free methods can be used in arbitrary fields, for example the collocation techniques together with the expansions by different basis-functions (Hu et al., 2005; Kuo et al., 2013; Li et al., 2008; Nachaoui et al., 2018; Shen, 2002; Tian et al., 2008) Other methods have been developed for solving Cauchy's problems. The reader can consult for example (Bergam et al., 2019; Berntsson et al., 2017; Ellabib et al., 2021; Isakov, 2017; Juraev, 2019, 2020; Nachaoui, 2003, 2004) and the references therein.

The aim of this paper is to explore a method based on polynomial expansion for the approximation of the Cauchy problem for the multidimensional Poisson equation in an arbitrary bounded domain enclosed by a smooth boundary. We compare some methods for the resolution of the ill-conditioned linear systems resulting from this approximation.

The rest of the paper is organized as follows. In Section 2, we recall Cauchy problem. The approximation method is given in Section 3. We present the techniques for dealing with ill-conditioning of linear systems in section 4. Finally, we present some numerical experiments to demonstrate the efficiency of the method in Section 5.

2 Inverse Cauchy problems

Let us consider $\Omega \subset \mathbb{R}^2$ with its boundary $\partial\Omega = \Gamma_1 \cup \Gamma_2$ where

$$\Gamma_1 = \{(r, \theta) : r = \rho(\theta),\ 0 \le \theta \le \beta\pi\}$$

and

$$\Gamma_2 = \{(r, \theta) : r = \rho(\theta),\ \beta\pi \le \theta < 2\pi\},\ \beta < 1.$$

The inverse Cauchy problem for the Poisson equation that will be considered is as follows: Given the Cauchy data $u(x, y)$ and $\partial_n u(x, y)$, on the accessible part Γ_1 find the unknown function $u(x, y)$ such that

$$\begin{aligned} -\Delta u(x, y) &= F(x, y) \qquad (x, y) \in \Omega, & (1)\\ u(\rho,\, \theta) &= h(\theta) \qquad 0 \le \theta \le \beta\pi, & (2)\\ \partial_n u(\rho,\, \theta) &= g(\theta) \qquad 0 \le \theta \le \beta\pi, & (3) \end{aligned}$$

where F, $h(\theta)$ and $g(\theta)$ are given functions.

Note that the part, Γ_1 is overdetermined (two boudary conditions are specified) while Γ_2 is undetermined (no boudary condition is specified). The inverse problem can therefore be redefined as follows: request an unknown boundary function $f(\theta)$ on the inaccessible part Γ_2 under Eqs. 1), (2) and (3).

In the next section we will approach the solution of the Cauchy problem using The collocation technique with the polynomial expansion method. It is a simple and inexpensive method in computation but which is rarely used as a major means to solve the PDEs due to its very ill-conditioned behavior. It makes the interpolation by higher-order polynomials not being easy to be numerically implemented. Indeed, the interpolation by polynomials of higher order is difficult to very ill-conditioned linear systems for which that iterative methods struggle to converge towards suitable solutions. We will see how to get around this inconvenience.

Let's start by noting that the normal derivative of u can be expressed in the following form (Liu & Kuo, 2016):

$$\partial_n u(\rho,\, \theta) = \eta(\theta) \left[\frac{\partial u(\rho,\, \theta)}{\partial \rho} - \frac{\rho'}{\rho^2} \frac{\partial u(\rho,\, \theta)}{\partial \theta} \right], \tag{4}$$

where

$$\eta(\theta) = \frac{\rho(\theta)}{\sqrt{\rho^2(\theta) + [\rho'(\theta)]^2}}. \tag{5}$$

On the other hand, we can also express $\partial_n u(x, y)$ in terms of $\partial_x u$ and $\partial_y u$ by

$$\partial_n u = \eta(\theta) \left[\cos(\theta) - \frac{\rho'}{\rho^2} \sin(\theta) \right] \partial_x u \,+\, \eta(\theta) \left[\sin(\theta) - \frac{\rho'}{\rho^2} \cos(\theta) \right] \partial_y u. \tag{6}$$

3 Polynomial expansion

The solution $u(x, y)$ is expanded by

$$u(x, y) = \sum_{i=1}^{m} \sum_{j=1}^{i} c_{ij} x^{i-j} y^{j-1}, \tag{7}$$

where the $n = \frac{m(m-1)}{2}$ coefficients c_{ij} are to be determined. Note that the maximal order of the above polynomial is $m - 1$.

Note that From Eq.(7) it is very easy to obtain

$$\begin{aligned} \partial_x u(x, y) &= \sum_{i=1}^{m} \sum_{j=1}^{i} c_{ij} (i-j) x^{i-j-1} y^{j-1}, & (8) \\ \partial_y u(x, y) &= \sum_{i=1}^{m} \sum_{j=1}^{i} c_{ij} (j-1) x^{i-j} y^{j-2}, & (9) \\ \Delta u(x, y) &= \sum_{i=1}^{m} \sum_{j=1}^{i} c_{ij} \left[(i-j)(i-j-1) x^{i-j-2} y^{j-1} + (j-1)(j-2) x^{i-j} y^{j-3} \right]. & (10) \end{aligned}$$

Firstly, the coefficients c_{ij} in Eq.(7) can be expressed as an $n-$dimensional vector $\mathbf{c}$ with the components c_k, $k = 1, \ldots, n$ where the coefficients c_{ij} are reordered crossing i from 1 to m and j from 1 to i. Then for the generic point in the boundary the term $u(x, y)$ can be expressed as an inner product of a vector $\mathbf{a}$ with $\mathbf{c}$, that is,

$$u(x, y) = [1 \;\; x \;\; y \;\; x^2 \;\; xy \;\; y^2 \;\; x^3 \;\; x^2y \;\; xy^2 \;\; y^3 \;\; \cdots] \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_n \end{bmatrix} = \mathbf{a}^T \mathbf{c}. \tag{11}$$

Including Eqs.(8) and (9) into (6) gives us an expression of $\partial_n u$.

For $(x, y) \in \Gamma_1$ the term $\partial_n u(x, y)$ can be expressed as an inner product of a vector $\mathbf{e}$ with $\mathbf{c}$, where the components e_k is defined by

$$e_k = \eta(\theta)\left[(i-j)x^{i-j-1}y^{j-1}\left(\cos(\theta) - \frac{\rho'}{\rho^2}\sin(\theta)\right) + (j-1)x^{i-j}y^{j-2}\left(\sin(\theta) - \frac{\rho'}{\rho^2}\cos(\theta)\right)\right]. \tag{12}$$

with the coefficients i, j are the same as those used to calculate c_k from $c_{i,j}$. Similarly, for a generic point in the domain the term $\Delta u(x, y)$ can be expressed from (10) as an inner product of a vector $\mathbf{d}$ with $\mathbf{c}$, where the component d_k, $k = 1, \dots, n$ are in the form

$$d_k = (i-j)(i-j-1)x^{i-j-2}y^{j-1} + (j-1)(j-2)x^{i-j}y^{j-3}. \tag{13}$$

Let's begin by choosing n_1 points (x_i, y_i), $i = 1, \dots, n_1$ on the boundary Γ_1 to satisfy the boundary condition(2)-(3), and n_2 points (x_l, y_l), $l = 1, \dots, n_2$ on the domain Ω to satisfy the the equation (1). Then, introduce these equations into Eqs. (1)-(3), we obtain a system of linear algebraic equations to solve the n coefficients c_{ij}. Now, it is suitable to express the resulting linear algebraic equations in terms of matrix-vector product of the form

$$\mathbf{Ac{=}b}, \tag{14}$$

where $\mathbf{b}$ is the vector of of order $n_c = 2n_1 + n_2$ and $\mathbf{A}$ is a the $n_c \times n$ matrix given respectively by

$$\mathbf{b} = \begin{bmatrix} h(\theta_1) \\ \vdots \\ h(\theta_{n_1}) \\ \\ g(\theta_1) \\ \vdots \\ g(\theta_{n_1}) \\ \\ F(x_1, y_1) \\ \vdots \\ F(x_{n_2}, y_{n_2}) \end{bmatrix}, \mathbf{A} = \begin{bmatrix} \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_{n_1}^T \\ \\ \mathbf{e}_1^T \\ \vdots \\ \mathbf{e}_{n_1}^T \\ \\ \mathbf{d}_1^T \\ \vdots \\ \mathbf{d}_{n_2}^T \end{bmatrix}, \tag{15}$$

and $n \ll n_c$.

Then, the inverse Cauchy problem is reduced to solving the over-determined linear system (14) with respect to the chosen collocation points.

4 Solving the lineear system

4.1 The regularization of Tikhonov

Solving the linear system (14) is equivalent to solving the following least square minimization problem:

$$\min_{C \in R^n} ||AC - b|| \tag{16}$$

Tikhonov's regularization makes it possible to determine an approximation of the solution C by replacing the minimization problem (16) by a least squared problem with a penalty form :

$$\min_{C \in \mathbb{R}^n} \{||AC - b||^2 + ||L_\alpha C||^2\} \tag{17}$$

L_α is a matrix called regularization matrix and in general, it is taken equal to αI with I is the identity matrix and $\alpha > 0$ is a regularization parameter to be specified.

The problem is now written as follows:

$$\min_{C\in\mathbb{R}^n}\{||AC-b||^2+\alpha^2||C||^2\} \tag{18}$$

Like any problem of minimization without constraints, the minimum is reached for a zero gradient of the criterion of minimsation. Consequently, the minimum sought C_μ satisfies:

$$< \nabla J_\mu(C_\mu), \phi = 0 \forall \phi \in R^n \tag{19}$$

with $J_\mu(x) = ||Ax-b|| + \mu||C||^2,\ \mu = \alpha^2$

Indeed :

$$\begin{aligned} J_\mu(x) &= ||Ax-b|| + \mu||C||^2 && (20)\\ &= < Ax-b, Ax-b > +\mu < x, x > && (21)\end{aligned}$$

As a result:

$$\begin{aligned} < \nabla J_\mu(C_\mu), \phi > &= 2 < AC_\mu - b, A\phi > +2\mu < C_\mu, A\phi > \forall \phi \in R^n \\ &= 2 < A^T\{AC_\mu - b\}, \phi > +2\mu < A^T C_\mu, \phi > && (22)\end{aligned}$$

By applying equation (19), any calculation done gives a linear system to solve of the form:

$$\left[A^T A + \mu I\right] C_\mu = A^T b \tag{23}$$

Note that if $\mu = 0$ this system is equivalent to solving the following normal equation:

$$Dc = b_1, \tag{24}$$

where $b_1 = A^T b$ and $D = A^T A > 0$. D is a symmetric positive definite matrix, thus the conjugate grdiant (CG) method can be used to solve this last linear system.

To solve 23 or (24) with CG one must take care during the calculations. An example is never to calculate the matrix $D = A^T A$ because this leads to unnecessary inaccuracies. The procedure of solving the normal equations has many variations, but experience has shown the method denoted CGLS (Algorithm 1) to be the best choice in strong competition with the close cousin LSQR based on Lanczos bidiagonalization Paige & Saunders (2002).

Algorithm 1: Least Square Conjugate Gradients (CGLS). Solve $A^T Ax = A^T b$, $A \in \mathbb{R}^{n,m}, n > m$

```
1: $r_{(0)} = b - Ax_{(0)}$
2: $d_{(0)} = A^T r_{(0)}$
3: $i \leftarrow 0$
4: while $r_{(i)} \neq 0$ do
5:     $\beta_{(i)} = \frac{(A^T r_{(i)})^T A^T r_{(i)}}{(Ad_{(i)})^T Ad_{(i)}}$
6:     $x_{(i+1)} = x_{(i)} + \beta_{(i)} d_{(i)}$
7:     $r_{(i+1)} = r_{(i)} - \beta_{(i)} Ad_{(i)}$
8:     $\gamma_{(i+1)} = \frac{(A^T r_{(i+1)})^T A^T r_{(i+1)}}{(A^T r_{(i)})^T A^T r_{(i)}}$
9:     $d_{(i+1)} = A^T r_{(i+1)} + \gamma_{(i+1)} d_{(i)}$
10:    $i \leftarrow i + 1,$
11: end while
```

4.2 Preconditioned and two-side squential operations

It is well known that the rate of convergence of iterative methods for solving (23) is strongly influenced by the spectral properties of A. Preconditioning amounts to transforming the original system into one having the same solution but more favorable spectral properties, such as a clustering of the eigenvalues around 1. A preconditioner is a matrix that can be used to accomplish such a transformation. If M is a nonsingular matrix which approximates $A^{-1}(M \sec A^{-1})$, the transformed linear system

$$MAx = Mb \tag{25}$$

will have the same solution as system (23) but the convergence rate of iterative methods applied to (25) may be much higher. Problem (25) is preconditioned from the left, but right preconditioning is also possible. Preconditioning on the right leads to the transformed linear system

$$AMy = b. \tag{26}$$

Once the solution y of (26) has been obtained, the solution of (23) is given by $x = My$. The choice between left or right preconditioning often depends on the choice of the iterative method, and on properties of the coefficient matrix (the side of the preconditioner can be important for nonsymmetric problems).

Now, we want to reduce the condition number $\kappa(A)$ by scaling of A. The importance of scaling of linear algebraic equations is needed to equilibrate our system and it has a long history of development. If the norm of all rows or columns in a matrix are equals, then that matrix is called equilibrated. The problem is to find a suitable diagonal matrix Q or P such that the condition number of AP, QA, or QAP is reduced considerable possible. Liu & Wei (2013) has proposed a simple procedure to find P and Q; summarized as follows:

Let us start with the right- and left- conditioner diagonal matrices P and Q, respectively as:

$$P = \begin{bmatrix} p_1 & 0 & \cdots & 0 & 0 \\ 0 & p_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & p_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & p_n \end{bmatrix} \tag{27}$$

and

$$Q = \begin{bmatrix} q_1 & 0 & \cdots & 0 & 0 \\ 0 & q_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & q_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & q_n \end{bmatrix} \tag{28}$$

where p_k and q_k defined by the following formula:

$$p_k = \gamma \left(\frac{\sum_{i=1}^{n} A_{i1}^2}{\sum_{i=1}^{n} A_{ik}^2} \right)^{(1/2)}, \qquad k = 1, \dots, n, \tag{29}$$

$$q_k = \delta \left(\frac{\sum_{j=1}^{n} A_{1j}^2}{\sum_{j=1}^{n} A_{kj}^2} \right)^{(1/2)}, \qquad k = 1, \dots, n. \tag{30}$$

Where γ and δ are amplification factors used to more reduce the condition number. When $\gamma = 1 = \delta$, the resultant matrices QA and AP have the same norm of each row and column respectively.

Let us define two two operators in Eqs.(27) and (28), respectively, by:

$$P = P_\gamma(A), \quad Q = Q_\delta(A)$$

and we construct a sequence of P_k and Q_k, $k = 1, \ldots, M$ by

$$P_1 = P_\gamma(A), \quad A_1 = AP_1 \tag{31}$$

$$Q_k = Q_\delta(A_{2k-1}),\ k = 1, \ldots, M \quad P_k = P_\gamma(A_{2k-2}),\ k = 2, \ldots, M \tag{32}$$

where

$$A_{2k-1} = A_{2k-2}P_k,\ k = 2, \ldots, M \quad \text{and} \quad A_{2k} = Q_kA_{2k-1}, \quad k = 1, \ldots, n. \tag{33}$$

Consider the right-preconditioner P and left-preconditioner Q given by

$$P = P_1P_2P_3 \cdots P_n \quad \text{and} \quad Q = Q_nQ_{n-1}Q_{n-2} \cdots Q_1. \tag{34}$$

Now, Eq.(23) will be solved in section 5 with

1. **Right-**preconditioner $By = b_1$ where $\mathbf{c} = Py$ and $B = AP$,
2. **Left-**preconditioner $B\mathbf{c} = b_0$ where $b_0 = Qb_1$ and $B = QA$,
3. **Two-sides** preconditioner $By = b_0$ where $\mathbf{c} = Py$, $b_0 = Qb_1$ and $B = QAP$.

4.3 Stopping criterion

The simplest and most common stopping criteria are based on the absolute and relative residual:

$$\|r_i\| < Tol \tag{35}$$

$$\|r_i\|/\|b\| < Tol \tag{36}$$

where Tol is a user-provided error tolerance. Since onen usually prefers relative stopping criterion in practice, we will focus on (36). Note that criterion 3 can be obtained from 4 by an appropriate scaling on 4. Other effcient stopping criterion can be found in Ashby et al. (2006).

4.4 Initial guess

Both the CGM and the CGLS described in Section 4.1 require an initial guess to be specified for u, on the underspecified boundary Γ_2. This initial guess is improved at every iteration and approaches the exact solution. Therefore, the rate of convergence and the accuracy of these methods clearly depend on how close the initial guess is to the exact solution. In all the examples presented here, we have taken a zero vector as the initial iteration.

5 Numerical results and discussion

It is the purpose of this section to present and compare the numerical results for the Cauchy problem considered in this study which have been obtained using the methods described in Section 4.2.

To illustrate the ability of the present inverse algorithms in estimating the unknown boundary boudary condition on Γ_2 from the knowledge of the specified conditions on Γ_2, we consider in following some analysis models where an exact solution is used to calculate the function F, its trace H and its normal derivative g on $Gamma_1$. By using these exact given data and zero initial data, one is asked to reconstruct the exact boundary using the CGM like-méthods descrived in section 4.2. Under this consideration the left term in Stopping criterion (36) may decrease to a very small number since there exists an exact solution. Thus one may take a very small Tol.

Example 1. *In this example we suppose that the closed-form solution is $u(x, y) = x^2 - y^2 - r^2$, the domain is bounded by $\rho(\theta) = 1$ and Γ_1 is defined taking $\beta = 0.5$. The number of boundary collocation used for discretising the boundary is taken to be $n_1 = 11$ and the number of internal collocation points $n_2 = 88$. The inverse analysis is then performed by varying m from 1 to 10. For both algorithms CGM and CGLS, we take $Tol = 10^{-10}$.*

Table 1

m	Error by using CGM	Iteration
2	3.43767386515131	3
3	6.43902176532603e-12	7
4	6.16624977783573e-11	17
5	1.02578575221883e-08	39
6	5.72754328572817e-09	94
7	0.00214552046363989	134
8	0.00459485878112826	195
9	0.00738105992314577	434
10	0.00853877435842424	789

(a) CGM method

m	Error by using CGLS	Iteration
2	3.43767386515131	3
3	5.58249858283589e-13	7
4	1.18318715677388e-12	17
5	4.09398738778544e-10	37
6	1.15626872999672e-11	84
7	0.00214551490191763	113
8	0.00459488563614370	153
9	0.00738168803743888	300
10	0.00854067598588999	544

(b) CGLS Method

The results are presented in the (1a) and (1b). we observe that from $m = 3$ we obtain a very good approximation, which conforms to the data since the axacte solution is a polynomial function of degree 2. We also note that the best approximation is obtained for $m = 3$, that is to say a polynomial of degree 2. Which shows that the approximation is exact when the solution is a plolynomial. We also observe that the CGLS method is more accurate than the CGM and that even with equal accuracy CGLS is faster.

Table 2

m	Error by using CGM	Iteration
2	3.08098845306835	3
3	2.06501482580279e-14	9
4	7.73714425861272e-13	21
5	3.02635694282571e-11	44
6	8.89843532192458e-10	104
7	2.56322399971864e-08	258
8	8.51758197195629e-06	843
9	0.00189369340274298	1445
10	0.00450677718451753	3865

(a) CGM method

m	Error by using CGLS	Iteration
2	0.452826479542826	3
3	7.77156117237610e-16	9
4	8.99280649946377e-15	18
5	2.60569343879524e-13	41
6	3.95239396766556e-13	82
7	1.30903621275991e-10	176
8	7.00823010735263e-11	428
9	0.00188057609507458	787
10	0.00446554140209410	1300

(b) CGLS Method

We take the same data and we take a smaller tolerance, $Tol = 10^{-15}$. The results in tables(2a) and (2b) confirm the first remarks. With this smaller tolerance accuracy is improved for all approximation order m. We obtain almost an exact approximation with always a better approximation for CGLS with a smaller number of iteration.

Example 2. *In this example we take a polynomial solution with a higher degree $u(x, y) = 6x^2y^2 - x^4 - y^4$. The domain is bounded by $\rho(\theta) = 0.5$ and Γ_1 is defined taking $\beta = 0.5$. The number of boundary collocation used for discretising the boundary is taken to be $n_1 = 11$ and the number of internal collocation points $n_2 = 88$. The inverse analysis is then performed by varying m from 1 to 10. For both algorithms CGM and CGLS, we take $Tol = 10^{-10}$.*

We also notice from tables(3a) and (3b) that, the accuracy is always good but deteriorates when m increases. This is explained by the fact that, since the degree is high, a higher m is *needed to have a better approximation and therefore a larger number of collocation points is necessary, but we kept the same points as in the first case.*

Table 3

m	Error by using CGM	Iteration
2	0.452826479542826	3
3	3.55687071904056	7
4	3.73406976901455	17
5	2.54404769231325e-10	40
6	1.48116175324710e-08	94
7	3.76989889757884e-06	309
8	0.0331786183717281	414
9	0.0645458932329768	659
10	0.0140685031061766	1622

(a) CGM method

m	Error by using CGLS	Iteration
2	0.452826479542826	3
3	3.55687071903923	7
4	3.73406976899618	17
5	3.42080141649189e-10	37
6	7.29394158641365e-11	85
7	9.50189633141289e-08	204
8	0.0331790188448867	311
9	0.0645466728425814	448
10	0.0140703156276937	1089

(b) CGLS Method

We increase the number of collocation points, we take $n_1 = 60$ and $n_2 = 720$, we then observe an improvement in the results: the error is multiplied by 10^{-1} everywhere with always the same advantage for the CGLS method (see from tables(4a) and (4b)).

Table 4

m	Error by using CGM	Iteration
2	0.802436693874813	3
3	9.7596071887227	7
4	10.2269890251551	17
5	3.98794661270787e-9	42
6	1.56532750250490e-08	94
7	1.16663086470551e-06	311
8	0.0931954381763555	417
9	0.0114750589626180	1015
10	0.0258746121939944	1546

(a) CGM method

m	Error by using CGLS	Iteration
2	0.802436693874812	3
3	9.75969071886996	7
4	10.22698902249560	17
5	2.11913586435851e-10	38
6	4.6731159384812e-11	85
7	8.17991998152097e-8	206
8	0.0931957783481953	276
9	0.0114987827121759	589
10	0.0258937249121710	816

(b) CGLS Method

The results of the tables (5a) and (5b) are obtained with the same data as the for tables (3a) and (3b), only the tolerance is smaller $Tol = 10^{-15}$. We observe a clear improvement in the results, this time the error is multiplied by 10^-2. We therefore obtain better results without increasing the number of collocation points.

Table 5

m	Error by using CGM	Iteration
2	0.452826479542826	3
3	3.55687071904009	11
4	3.73406976907850	25
5	1.98182892995909e-10	50
6	1.28474817444136e-08	135
7	1.28065266389299e-06	440
8	3.08965163546447e-05	1840
9	0.00142812522068457	9385
10	0.0721122655173159	14500

(a) CGM method

m	Error by using CGLS	Iteration
2	0.452826479542826	3
3	3.55687071904010	12
4	3.73406976908181	21
5	2.41388734534916e-14	50
6	3.28126843826970e-12	104
7	8.32238364465459e-11	252
8	9.13505304008724e-09	729
9	2.893499240629375e-08	2438
10	0.00667144887609868	2655

(b) CGLS Method

5.1 The effect of noise

In real-world inverse problems, the known boundary data are measured and, therefore, contaminated by inherent measurement errors. For all examples investigated in this paper, the simulated noisy data are generated using the following formula:

$$h(\theta) = uex(\rho(\theta), \theta) + w\sigma$$

where σ is the standard deviation of measurement errors which is assumed to be the same for all measurements, and w is the Gaussian distributed random error. σ determines the noise level, it takes values of 0.001, 0.010.05 and 0.1.

We now discuss the influence of the measurement errors on the inverse solutions in predicting $f(\theta)$ on Γ_2 using the the data of example 1 for various amounts of noise are presented in figures

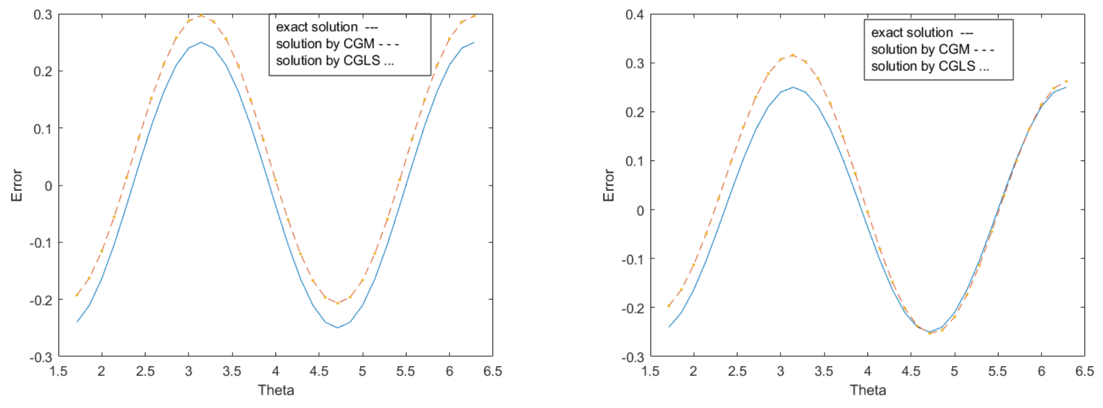

Figure 1: Results for the analysis model Left: $\sigma = 0.1$, right; $\sigma = 0.05$

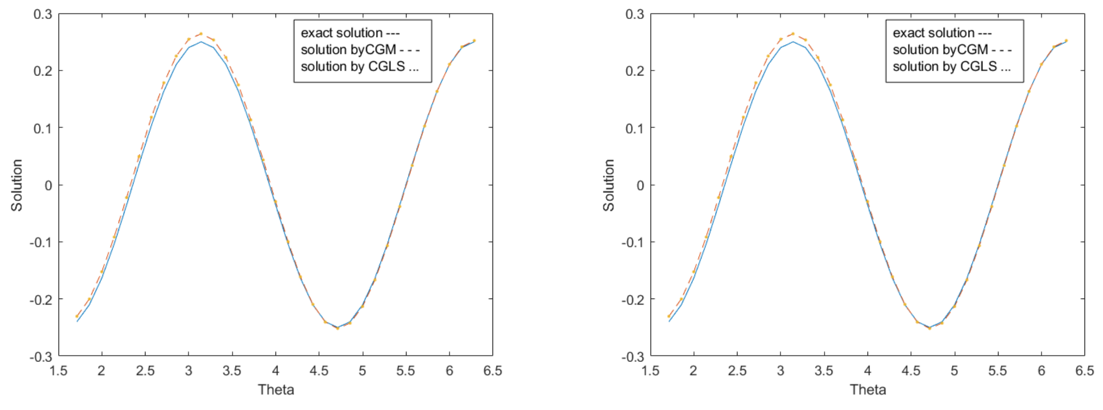

Figure 2: Results for the analysis model Left: $\sigma = 0.01$, right; $\sigma = 0.001$

Figures 1 and 2 present the numerical solution the boundary Γ_2, retrieved using CGM and CGLS algorithms in comparison with the exact solution. It can be seen from these figures that, as σ decreases, the numerical solution approximates better the exact solution while remaining stable. The numerical results obtained by both algorithms are équivalent. They are still a reasonably good approximation to the exact solution of the problem, even when the boundary data being polluted with a 10%($\sigma = 0.1$) relative random noise, since we have solved an ill-conditioned problem (Condition number about 1.39×10^2).

However, when we solve a very ill-posed problem as in example 4 and 5 (Condition number about 3.44×10^7 and 7.7×10^9 respectively) with noise in the data, the recovery of the approximate solution on the underspecified boundary Γ_2 then becomes not so good. The results remain unsatisfactory even when the two methods are used with regularization and preconditioning.

5.2 The effect of regularization en preconditioning

Example 3. *To see the impact of regularization as well as the contribution of preconditioning, we applied the different methods presented in section 4.2.*

Here, we suppose that the closed–form solution is $u(x, y) = \exp(x) * \cos(y)$. The domain is bounded by $\rho(\theta) = \frac{1}{2}$, and Γ_1 defined taking $\beta = 0.5$. The number of points $n_1 = 50$ and $n_2 = 500$. the best results are obtained for $m = 9$. The tolerance is taken $Tol = 10^{-15}$.

In what follows NP denotes the number of matrix product in the equation (34), γ is the parameter of the preconditioning in equations (29-29)), α is the parameter of regularization and Iter denotes the number of iterations necessary for convergence.

We first present the results obtained by CGM Then the same system is solved using the

Table 6: CGM Method

	Precondition	γ	NP	Error	α	Iter
Without preconditioning and without regularization				0.008988		3869
Regularization without preconditioning				0.008872	$\alpha = 10^{-12}$	3788
Regularization with preconditioning	Right			0.005888	$\alpha = 10^{-12}$	2766
	Left			0.007731	$\alpha = 10^{-13}$	2170
	Two-sides			0.006863	$\alpha = 10^{-14}$	1872
Regularization with preconditioning	Right	0.2		0.004002	$\alpha = 10^{-15}$	2144
	Left	0.6		0.007825	$\alpha = 10^{-12}$	1694
	Two-sides	0.5		0.007145	$\alpha = 10^{-14}$	1835
Regularization with preconditioning	Right	0.9	2	0.008357	$\alpha = 10^{-14}$	2088
	Left	0.2	1	0.006437	$\alpha = 10^{-16}$	1295
	Two-sides	0.5	1	0.003953	$\alpha = 10^{-13}$	1818

(CGLS).

In Table 6 and 7 we present the accuracy errors and the number of iteration obtained using the regularization methods described in section 4.2. We notice that after 3869 *iteration CGM converges with a precision of the order of* 10^{-2}. *The regularization does not bring any improvement in accuracy and we only gain* 2% *of time (*81 *iterations).*

On the other hand the various types of preconditioning reduce the error as well as the number of iterations necessary to reach convergence. The best result is obtained for a regullarization with a two-sided multiple preconditioner for which the error is divided by 10 *and the number of iterations is divided by* 2.

The accuracy obtained with the CGLS method is comparable to that obtained with CGM, however, it is much faster. The number of iterations is reduced by 70% *without regularization or preconditioning. Again, the best result is obtained by the regularized and preconditioned method with a multiple two-sided preconditioner.*

Example 4. *In this example, we suppose that the closed–form solution is $u(x, y) = \exp(x) * \cos(y)$. The domain is bounded by $\rho(\theta) = \exp(\sin(\theta))\sin^2(2\theta) + \exp(\cos(2\theta))\cos^2(2\theta)$, and Γ_1*

Table 7: CGLS Method

	Precondition	γ	NP	Error	α	Iter
Without preconditioning and without regularization				0.0090856		1162
Regularization without preconditioning				0.0090860	$\alpha = 10^{(-12)}$	1068
Regularization with preconditioning	Right			0.0090395	$\alpha = 10^{(-14)}$	793
	Left			0.00767280	$\alpha = 10^{(-15)}$	811
	Two-sides			0.00696777	$\alpha = 10^{(-12)}$	693
Regularization with preconditioning	Right	0.4		0.00908275	$\alpha = 10^{(14)}$	790
	Left	0.2		0.00780252	$\alpha = 10^{(-13)}$	805
	Two-sides	0.2		0.00738994	$\alpha = 10^{(-16)}$	680
Regularization with preconditioning	Right	0.4	1	0.00879906	$\alpha = 10^{(-14)}$	790
	Left	0.6	1	0.00658269	$\alpha = 10^{-8}$	769
	Two-sides	0.2	2	0.00128111	$\alpha = 10^{-19}$	661

defined taking $\beta = 0.5$. The number of points $n_1 = 50$ and $n_2 = 1000$. the best results are obtained for $m = 9$. The tolerance is taken $Tol = 10^{-15}$.

We first present the results obtained by CGM

Table 8: CGM Method

	Precondition	γ	NP	Error	α	Iter
Without preconditioning and without regularization				0.261835		1243
Regularization without preconditioning				0.132279	$\alpha = 10^{-5}$	1197
Regularization with preconditioning	Right			0.096146	$\alpha = 10^{-8}$	756
	Left			0.074738	$\alpha = 10^{-3}$	1455
	Two-sides			0.036945	$\alpha = 10^{-3}$	910
Regularization with preconditioning	Right	0.1		0.096146	$\alpha = 10^{-10}$	744
	Left	0.4		0.064063	$\alpha = 10^{-3}$	1116
	Two-sides	0.2		0.033322	$\alpha = 10^{-6}$	882
Regularization with preconditioning	Right	0.1	1	0.096146	$\alpha = 10^{-10}$	744
	Left	0.4	2	0.09692	$\alpha = 10^{-6}$	1017
	Two-sides	0.3	1	0.038629	$\alpha = 10^{-13}$	869

Then the same system is solved using the CGLS method.

Note that the linear system obtained for this example est very ill-conditioned (Condition number about 3.44×10^7). From Table 8 we see that after 1243 iteration CGM converges with a error of 26%. The regularization divide the error by 2 but stay above 13% with 46 iteration. Again, the preconditioning postivement act on accuracy and time of execution.

The results presented in Table 9 confirm the performances of the CGLS method already observed for the previous example.

Example 5. *In this last example, we consider data calculated from the following solution $u(x, y) = cos(5(x^2 + y^2))$ The domain is bounded by $\rho(\theta) = 0.5$ and Γ_1 defined taking $\beta = 1$. The number of points $n_1 = 64$ and $n_2 = 2048$. the best results are obtained for $m = 14$. The tolerance is taken $Tol = 10^{-15}$.*

It should be noted that the example considered in here is a very severe example for iterative methods since the exact solution is very far from the most natural guess available, the matrix is a large matrix which is very ill-conditioned (Condition number about 7.72×10^9).

Table 9: CGLS Method

	Precondition	γ	NP	Error	α	Iter
Without preconditioning and without regularization				0.261831		977
Regularization without preconditioning				0.132260	$\alpha = 10^{-5}$	877
Regularization with preconditioning	Right			0.096146	$\alpha = 10^{-8}$	563
	Left			0.074731	$\alpha = 10^{-3}$	864
	Two-sides			0.023686	$\alpha = 10^{-4}$	664
Regularization with preconditioning	Right	0.1		0.096145	$\alpha = 10^{-10}$	554
	Left	0.9		0.099474	$\alpha = 10^{-4}$	855
	Two-sides	0.4		0.053762	$\alpha = 10^{-4}$	537
Regularization with preconditioning	Right	0.1	1	0.096145	$\alpha = 10^{-8}$	554
	Left	0.5	3	0.091908	$\alpha = 10^{-6}$	791
	Two-sides	0.1	2	0.035928	$\alpha = 10^{-18}$	526

Table 10: CGM Method

	Precondition	γ	NP	Error	α	Iter
Without preconditioning and without regularization				-		-
Regularization without preconditioning				-	-	-
Regularization with preconditioning	Right			0.004362	$\alpha = 10^{-11}$	17753
	Left			0.008014	$\alpha = 10^{-18}$	37737
	Two-sides			0.005971	$\alpha = 10^{-12}$	10477
Regularization with preconditioning	Right	0.9		0.006194	$\alpha = 10^{-11}$	15795
	Left	0.5		0.009419	$\alpha = 10^{-13}$	34757
	Two-sides	0.3		0.006867	$\alpha = 10^{-13}$	9686
Regularization with preconditioning	Right	0.7	1	0.009247	$\alpha = 10^{-10}$	15493
	Left	0.2	2	0.053826	$\alpha = 10^{-11}$	23013
	Two-sides	0.1	1	0.012130	$\alpha = 10^{-18}$	7087

Table 11: CGLS Method

	Precondition	γ	NP	Error	α	Iter
Without Precondition and without regularization				-		-
Regularization without preconditioning				-	-	-
Regularization with preconditioning	Right			0.004671886	$\alpha = 10^{-11}$	11332
	Left			0.007536478	$\alpha = 10^{-14}$	50717
	Two-sides			0.006039663	$\alpha = 10^{-13}$	7676
Regularization with preconditioning	Right	0.2		0.009809684	$\alpha = 10^{-12}$	10927
	Left	0.2		0.018132786	$\alpha = 10^{-18}$	44702
	Two-sides	0.3		0.006861981	$\alpha = 10^{-12}$	6935
Regularization with preconditioning	Right	0.1	2	0.003153701	$\alpha = 10^{-12}$	9601
	Left	0.1	1	0.049899417	$\alpha = 10^{-11}$	41066
	Two-sides	0.7	1	0.012127153	$\alpha = 10^{-14}$	5694

We first present the results obtained by CGM.

We observe from table Table 10 that the CGM does not produce any solution for this problem even for the regularized system. This confirms the ill-conditioning character of the systems resulting from the polynomial expantion in particular when the degree of the approximation polynomial is quite high. We also observe that CGM produces a fairly good solution with decent precision when applying preconditioning. This shows the interest of preconditioning the systems isus of the approximation of the ill-posed problems.

We then used CGLS to solve this extremely ill-conditioned linear system. The results presented in Table 11 confirm the performances of the CGLS method already observed for the previous example.

6 Conclusion

We solve the inverse Cauchy problem of Poisson equations in an arbitrary domaine for recovering unknown data on a part of the boundary from the over-specified Cauchy boundary conditions given on an other inaccessible part. We have transformed the inverse Cauchy problem to solve a direct problem, using polynomial expansion. A regularization combined with a preconditioning strategy is used to reduce the number of conditions of the linear system in order to determine the coefficients of expansion. Several numerical examples are presented to show that the method can overcome the very ill-posed property of the inverse Cauchy problem. We have found that the Cauchy problem for the Poisson equation can be regularized by one of the two methods considered as all of them produced a stable numerical solution.

However, the numerical solutions obtained by these methods differ in terms of number of iteration. It has been found that the CGLS method outperforms the CGM. We note that for the severe test example considered, none of the two methods succeeds to converge without a regularization and a strategy of preconditioning.

References

Aboud, F., Nachaoui, A. & Nachaoui, M. (2021). On the approximation of a Cauchy problem in a non-homogeneous medium, *J. Phys.: Conf. Ser.* 1743 012003.

Alessandrini, G., Rondi, L., Rosset, E. & Vessella, S. (2009). The stability for the Cauchy problem for elliptic equations, *Inverse problems, 25,* 123004

Andrieux, S., Baranger, T. N. & Ben Abda, A. (2006). Solving Cauchy problems by minimizing an energy-like functional, *Inverse Probl., 22,* 115-133.

Ashby, S. F., Holst, M.J., Manteuffel, T.A., Saylor, P. E. (2001). The role of the inner product in stopping criteria for conjugate gradient iterations. *BIT 41*(1), 26-52.

Belgacem, F.B. (2007). Why is the Cauchy problem severely ill-posed?, *Inverse Problems, 23,* 823-836.

Berdawood, K.A., Nachaoui, A., Saeed, R., Nachaoui, M. & Aboud, F. (2020). An alternating procedure with dynamic relaxation for Cauchy problems governed by the modified Helmholtz equation. *Advanced Mathematical Models & Applications, 5*(1) 131-139.

Berdawood, K.A., Nachaoui, A., Saeed, R., Nachaoui, M. & Aboud, F. (2021). An efficient D-N alternating algorithm for solving an inverse problem for Helmholtz equation. *Discrete & Continuous Dynamical Systems - S*, doi: 10.3934/dcdss.2021013

Berdawood, K.A., Nachaoui, A., Nachaoui, M. & Aboud, F. (2021). An effective relaxed alternating procedure for Cauchy problem connected with Helmholtz Equation. *Numer. Methods Partial Differential Eq.,* 1-27. https://doi.org/10.1002/num.22793

Bergam, A., Chakib, A., Nachaoui, A. & Nachaoui, M. (2019). Adaptive mesh techniques based on a posteriori error estimates for an inverse Cauchy problem. *Applied Mathematics and Computation*, *346*, 865-878.

Berntsson, F., Kozlov, V.A., Mpinganzima, L. & Turesson, B.O. (2017). Iterative Tikhonov regularization for the Cauchy problem for the Helmholtz equation. *Comput. Math. Appl.,* *73*(1) 163-172.

Bourgois, L. (2006) Convergence rates for the quasi-reversibility method to solve the Cauchy problem for Laplace's equation. *Inverse Probl., 22* , 413-430.

Burger, M. (2001). A level set method for inverse problems. *Inverse Probl., 17,* 1327-1355.

Cimetière, A., Delvare, F., Jaoua, M. & Pons, F. (2001) Solution of the Cauchy Problem Using Iterated Tikhonov Regularization, *Inverse Problems, 17*, 553-570.

Chakib, A. & Nachaoui, A. (2006). Convergence analysis for finite element approximation to an inverse Cauchy problem. *Inverse Problems, 22*(4), 1191–1206.

Chakib, A., Nachaoui, A., Nachaoui, M. & Ouaissa, H. (2018). On a fixed point study of an inverse problem governed by Stokes equation. *Inverse Problems*, *35*(1), 015008.

Chang, J.R., Yeih, W. & Shieh, M.H. (2001) On the Modified Tikhonov's Regularization Method for the Cauchy Problem of the Laplace Equation. *Journal of Marine Science and Technology, 9*, 113-121.

Chen, J.T. & Chen, K.H. (1998). Analytical Study and Numerical Experiments for Laplace Equation with Overspecified Boundary Conditions. *Applied Mathematical Modelling, 22*, 703-725.

Chi, C.C., Yeih, W. & Liu, C.S. (2009) A Novel Method for Solving the Cauchy Problem of Laplace Equation Using the Fictitious Time Integration Method. *CMES-Computer Modeling in Engineering and Sciences, 47*, 167-190.

Choulli, M. (2009). *Une introduction aux problèmes inverses elliptiques et paraboliques.* Springer-Verlag, Berlin.

Ellabib, A. & Nachaoui, A. (2008). An iterative approach to the solution of an inverse problem in linear elasticity, *Mathematics and Computers in Simulation*, *77*(2-3), 189-201.

Ellabib, A., Nachaoui, A. & Ousaadane A. (2021). Mathematical analysis and simulation of fixed point formulation of Cauchy problem in linear elasticity, *Mathematics and Computers in Simulation*, *187*, 231-247.

Engl, H., Hanke M. & Neubauer A. (1996). *Regularization of Inverse Problems,* Kluwer, Dordrecht.

Essaouini, M., Nachaoui, A. & El Hajji, S. (2004). Numerical method for solving a class of nonlinear elliptic inverse problems, *Journal of Computational and Applied Mathematics*, *162*(1), 165-181.

Essaouini, M., Nachaoui, A. & El Hajji, S. (2004). Reconstruction of boundary data for a class of nonlinear inverse problems, *Journal of Inverse and Ill-Posed Problems*, *12*(4), 369-385.

Fang, W. & Lu, M. (2004) A Fast Collocation Method for an Inverse Boundary Value Problem, *International Journal for Numerical Methods in Engineering, 21*, 1563-1585.

Grigor'ev, Y. (2018). An analytical method for the inverse Cauchy problem of Lame equation in a rectangle. *J. Phys.: Conf. Ser.,* 991, 012029.

Hadamard, J. (1953). *Lectures on Cauchy's problem in linear partial differential equations.* Dover Publications, New York.

Hu, H.Y., Li, Z.C., & Cheng, A.H.D. (2005). Radial basis collocation methods for elliptic boundary value problems. *Computers & Mathematics with Applications*, *50*(1-2), 289-320.

Huang, C.H., & Chen, W.C. (2000). A three-dimensional inverse forced convection problem in estimating surface heat flux by conjugate gradient method. *International Journal of Heat and Mass Transfer, 43*(17), 3171-3181.

Isakov, V. (2017). *Inverse problems for partial differential equations.* Springer, Cham.

Jourhmane, M. & Nachaoui, A. (1996). A relaxation algorithm for solving a Cauchy problem, *Proc. of the 2nd Internat. Conf.*, 151-158, *Foundation Editor.*

Jourhmane, M. & Nachaoui, A. (2002). Convergence of an alternating method to solve the Cauchy problem for Poisson's equation, *Applicable Analysis, 81*(5), 1065-1083.

Jourhmane, M. & Nachaoui, A. (1999). An alternating method for an inverse Cauchy problem, *Numerical Algorithms*, *21*(1-4), 247-260.

Juraev, D.A. (2019). On a regularized solution of the Cauchy problem for matrix factorizations of the Helmholtz Equation. *Advanced Mathematical Models & Applications. 4* (1), 86-96.

Juraev, D.A. (2020). The solution of the ill-posed Cauchy problem for matrix factorizations of the Helmholtz equation. *Advanced Mathematical Models & Applications. 5*(2), 205-2021.

Kabanikhin, S.I. (2012). *Inverse and ill-posed problems.* Walter de Gruyter GmbH & Co. KG, Berlin.

Kabanikhin, S.I., Gasimov, Y.S., Nurseitov, D.B., Shishlenin, M.A., Sholpanbaev, B.B., Kasemov, S. (2013). Regularization of the continuation problem for elliptic equation. *J. Inverse Ill-Posed Probl., 21*(6), 871-874.

Kabanikhin, S.I. & Karchevsky A.L. (1995) Optimizational method for solving the Cauchy problem for an elliptic equation. *J. Inverse Ill-Posed Probl., 3*(1), 21-46.

Klibanov M., & Santosa, V. (1991). A computational quasi-reversibility method for Cauchy problems for Laplace's equation, *SIAM J. Appl. Math., 51*(6), 1653–1675.

Kozlov, V.A., Maz'ya, V.G. & Fomin, A.V. (1991). An iterative method for solving the Cauchy problem for elliptic equations. *Zh. Vychisl. Mat. i Mat. Fiz.*, *31*(1), 64-74.

Kuo C.L., Chang J.R., & Liu C.S. (2013). The modified polynomial expansion method for solving the inverse heat source problems. *Numer Heat Transf B: Fundam*, *63*, 357-70.

Lattès, R.; Lions, J.-L. (1969). The method of quasi-reversibility. Applications to partial differential equations. *Modern Analytic and Computational Methods in Science and Mathematics, 18* American Elsevier Publishing Co., Inc., New York.

Lavrentiev, M.M. (2013). *Some improperly posed problems of mathematical physics.* Springer Science & Business Media.

Li, Z.C., Lu, T.T., Huang, H.T., Cheng, A.H.D. (2008). *Trefftz and collocation methods.* Southampton: WIT Press.

Liu, C.-S. (2011). An Analytical Method for the Inverse Cauchy Problem of Laplace Equation in a Rectangular Plate. *J. Mech., 27*(4), 575–584.

Liu C.-S. & Atluri S. N. (2009). A highly accurate technique for interpolations using very high-order polynomials, and its applications to some ill-posed linear problems. *Computer Modeling in Engineering & Sciences*, *43* (1), 253-276.

Liu, C.S. & Kuo, C.L. (2016). A multiple-scale Pascal polynomial triangle solving elliptic equations and inverse Cauchy problems. *Engineering Analysis with Boundary Elements*, *62*, 35-43.

Liu, J.-C. & Wei, T. (2013) A quasi-reversibility regularization method for an inverse heat conduction problem without initial data. *Appl. Math. Comput., 219* (23), 10866–10881.

Marin, L. & Johansson, B. T. (2010). A relaxation method of an alternating iterative algorithm for the Cauchy problem in linear isotropic elasticity. *Computer Methods in Applied Mechanics and Engineering*, *199*(49-52), 3179-3196.

Nachaoui, A. (2003). An improved implementation of an iterative method in boundary identification problems. *Numer. Algorithms*, *33* (1-4), 381-398.

Nachaoui, A. (2004). Numerical linear algebra for reconstruction inverse problems A Nachaoui *Journal of Computational and Applied Mathematics, 162*(1), 147-164.

Nachaoui, A . Al-Rawi, E.S. & Qasim A.F.(2018) Solving Three Dimensional and Time Depending PDEs by Haar Wavelets Method. *Open Access Library Journal, 5*(5), 1-18.

Nachaoui, A. & Nachaoui, M., Gasimov, B. (2021). Parallel numerical computation of an analytical method for solving an inverse Problem. *Advanced Mathematical Models & Applications, 6*(2), 162-173.

Nachaoui, A. , Nachaoui, M. Chakib, A. & Hilal, M.A. (2021) Some novel numerical techniques for an inverse Cauchy problem. *Journal of Computational and Applied Mathematics, 381*, 113030.

Paige, C.C. & Saunders M.A. (1982). LSQR: Sparse linear equations and least squares problems. *ACM Transactions on Mathematical Software, 8*, 195-209 .

Shen, S.N.A.S. (2002). The meshless local Petrov-Galerkin (mlpg) method: a simple & less-costly alternative to the finite element and boundary element methods, *Computer Modeling in Engineering & Sciences*, *3*(1), 11-51.

Tian, H.Y., Reutskiy, S., & Chen, C.S. (2008). A basis function for approximation and the solutions of partial differential equations. *Numer Methods Part Differ Equ*, *24*, 1018-36.