HyEIT : Iterative Reconstruction Methods in MATLAB for Hybrid Inverse Problems in Electrical Impedance Tomography [1]

A.D. Garnadi[23a,b], Mokhamad Fakhrul Ulum[4d], Deddy Kurniadi[51], Utriweni Mukhaiyar[6f], Lina Choridah[7c], Nurhuda Hendra Setyawan[8c], Khusnul Ain[9g]

[a] *Institute for Globally Distributed Open Research and Education*
[b] *IPB University, Indonesia*
[c] *Faculty of Medicine, Public Health, and Nursing Universitas Gadjah Mada, Yogyakarta, Indonesia*
[d] *Faculty of Veterinary Medicine, IPB University, Indonesia*
[e] *Instrumentation and Control Research Division, Faculty of Industrial Technology, Institut Teknologi Bandung, Indonesia*
[f] *Statistics Research Division, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung, Indonesia*
[g] *Faculty of Sciences and Technology, Airlangga University, Surabaya, Indonesia*

Abstract

We present a MATLAB-based, open source reconstruction software for Hybrid Imaging problem arising from Current Density Impedance Imaging (CDII) and Acoustic-Electrical Impedance Tomography (AEIT), utilizing mixed finite element method solver of Darcy toolbox `DarcyLite` .

For a formulation of hybrid inverse problems in impedance tomography the successive substitution iterative schemes are adapted and an reconstruction algorithm is developed. The problem formulation is a class of hybrid

[1]Funded by "Riset Kolaborasi Indonesia 2019","Pengembangan Tomografi Elektrik Sebagai 'Contrasting Agent' Magnetic Resonance Imaging" AND by "Riset Kolaborasi Indonesia 2020", 'Pengembangan Tomografi Hibrida : Electrical Impedance Tomography + Ultrasonography Dalam Bidang Biomedika'

[2]e-mail address:agah.garnadi@gmail.com
[3]home address: Gg Bungur IV 107, Perumahan Cijerah II, Melong, Cimahi Selatan, Indonesia 40534
[4]e-mail address: ulum@ipb.ac.id; OrcID: http://orcid.org/0000-0003-2092-9645
[5]e-mail address: kurniadi@fti.itb.ac.id
[6]e-mail address:utriweni@math.itb.ac.id
[7]e-mail address: linachoridah@ugm.ac.id
[8]e-mail address: nurhuda.hendra.s@ugm.ac.id
[9]e-mail address: k_ain@fst.unair.ac.id

imaging modality which is current density impedance imaging or Acoustic-Electrical Impedance Tomography. The proposed algorithm is implemented numerically in two dimensions.

Keywords: Hybrid inverse problems, Iterative reconstruction methods, Impedance tomography, Current density impedance imaging, Magnetic resonance imaging, electrical impedance tomography, Ultrasound modulated electrical impedance tomography

---

Required Metadata

Current code version

Ancillary data table required for subversion of the codebase. Kindly replace examples in right column with the correct information about your current code, and leave the left column as it is.

Nr.	**Code metadata description**	**Please fill in this column**
C1	Current code version	Version 0.0
C2	Permanent link to code/repository used for this code version	NA
C3	Code Ocean compute capsule	
C4	Legal Code License	CC-BY NC 4.0 International
C5	Code versioning system used	For example svn, git, mercurial, etc. put none if none
C6	Compilation requirements, operating	MATLAB/GNU OCTAVE for the future
C7	Compilation requirements, operating environments & dependencies	`DarcyLite` http://www.math.colostate.edu/ liu/DarcyLite.zip
C8	If available Link to developer documentation/manual	NA
C9	Support email for questions	agah.garnadi[at]gmail.com

Table 1: Code metadata (mandatory)

The permanent link to code/repository or the zip archive should include the following requirements:

README.txt and LICENSE.txt.

Source code in a src/ directory, not the root of the repository.

Tag corresponding with the version of the software that is reviewed.

Documentation in the repository in a docs/ directory, and/or READMEs, as appropriate.

1. Motivation and significance

In this note we develop and analyse an iterative reconstruction method for the following hybrid inverse problem:

Reconstruct the scalar function σ from measurements of the form

$$\mathcal{A} = \sigma|\nabla u|^p \text{ in } \Omega, p = 1, 2 \tag{1}$$

where u is the solution to the linear PDE problem

$$\left.\begin{array}{rcl} \nabla(-\sigma\nabla u) & = & 0 \text{ in } \Omega, \\ u|_{\partial\Omega} & = & u_D, \end{array}\right\} \tag{2}$$

Here Ω is a smooth open bounded domain in $R^n, n \in \{2, 3\}$, and σ is bounded from above and from below by positive constants in Ω. We will henceforth assume that $|\nabla u|$ is uniformly bounded from below by a positive constant in Ω. The problem formulation and the reconstruction methods are independent of the spatial dimension, here we only works on $n = 2$, for prototyping. This inverse problem can be identified as a hybrid inverse problem or, equivalently, as a coupled-physics inverse problem, since it models an imaging modality utilizing coupled different physical phenomena.

The functions (1) model measurements from a couple of hybrid inverse problems of Electrical Impedance Tomography, i.e.

1. In the case $p = 1$, the functions (1) model measurements of the internal electrical current density and the tomographic method is known as Current Density Impedance Imaging (CDII); the same model appears in certain formulations of Magnetic Resonance Electrical Impedance Tomography (MREIT). [12], [13],[14]
2. While for $p = 2$, coming from AEIT (Acoustics Induce Electric Impedance Tomography). [15], [16] [17], [18]

The only package we are aware of with fully fledged capabilities was [21], implemented in PDE suite FENICS, written in C++.

2. Software description

In this section we derive an iterative reconstruction algorithms for the presented inverse problem. It is common feature of all the presented algorithms, that they are based on a transformation of a non-linear problem into a series of linear problems for which the corresponding solutions, under appropriate conditions, converge to a solution to the original non-linear problem. In this section, we explain how to use the successive iterative scheme to reformulate the inverse problem, develop a general iterative reconstruction algorithm and implement it numerically.

A way to reformulate (1)-(2) is to recast the set of generalized Laplace problems (2) as non-linear PDE problems. The unknown function σ can be eliminated from the PDEs using the internal measurements, since

$$\sigma = \frac{\mathcal{A}}{|\nabla u|^p}, p = 1, 2.$$

This gives the non-linear PDE problems

$$\left.\begin{array}{rcl} \nabla - \frac{\mathcal{A}}{|\nabla u|^p}\nabla u &=& 0 \text{ in } \Omega, p = 1, 2, \\ u|_{\partial\Omega} &=& u_D, \end{array}\right\} \tag{3}$$

In this setting a procedure for solving the inverse problem would be to first solve the non-linear problems for u and then determine σ by (1). One of the simplest approaches to approximate solutions to such a non-linear PDE problem is to use a fixed-point iteration or successive iterative scheme, also known as the method of successive substitutions [4]. The idea behind the successive substitution scheme is basically to make an initial guess on the non-linear coefficient in the PDE problems (3); then the now linear PDE problem is solved, and we get an approximation of u which is used to update the estimate of the coefficient. Let the initial guess be denoted by σ_0. Throughout this paper, we use a superscript to denote the iteration number. The successive substitution scheme, applied to (3), generates a series of solving linear problems of the type

$$\left.\begin{array}{rcl} \nabla(-\sigma^k\nabla u^k) &=& 0 \text{ in } \Omega, k = 0; 1; ...; \\ u^k|_{\partial\Omega} &=& u_D, \end{array}\right\} \tag{4}$$

which are solved successively for (u^k). The coefficient σ^{k+1} is defined by the relation

$$\sigma^{k+1} = \frac{\mathcal{A}}{|\nabla u^k|^p}$$

It is important to notice that the transformation of the non-linear problem into a series of linear problems is not a result of any algebraic or symbolic manipulations; it simply a consequence of a substitution made directly in the non-linear PDE. The choice of stopping criteria for the successive iterations depends on the application. For problems like ours, where the objective is to estimate the PDE coefficient σ, and not to the functions u, it is natural to choose a stopping criteria based on the tolerance $\|\sigma^{k+1} - \sigma^k\|$ in some suitable function space and the iteration number k.

The performance of the successive substitution relies on an initial guess that provides a sufficiently good starting point, such that the successive approximation of the nonlinear coefficients is improved in each iteration. The convergence properties of the successive substitution scheme can be studied using contraction maps and the application of the Banach fixed point theorem, however the practicality of applying this theory on a general non-linear PDE problem seems rather limited [4]. Nonetheless, for our case, the convergence properties of the successive substitution have been analysed [2, 6].

The successive substitution scheme was the main idea behind one of the first reconstruction algorithms for the hybrid inverse problems expressed in the form of the non-linear PDE problems given by (3). The resulting algorithm was denoted the J-substitution algorithm, due to the fact that the functions $\mathcal{A}$ models measurements of the magnitude of the current density vector field, often denoted by $\mathcal{A}$ [3]. Note that the original formulation of the J-substitution algorithm considered the corresponding Neumann problem. A similar implementation of the Dirichlet problem for a single internal measurement has also been used to solve the equivalent inverse problem from CDII [5, 6].

2.1. Software Architecture

The successive iterative scheme can be applied directly to the non-linear PDE formulation of the inverse problem to produce the following algorithm:

- Algorithm. Successive algorithm

- Define σ^0, maximum number of iteration K and tolerance level T.
- Set $k = 0$ and $\varepsilon = 2T$.
- while $k < K$ and $\varepsilon > T$ do
  1. Solve for $(u^k, \mathbf{p}^k)$ in

$$\begin{aligned} \mathbf{p}^k - \sigma^k \nabla u^k &= 0 \quad \text{in } \Omega \\ \nabla \cdot \mathbf{p}^k &= 0 \quad \text{in } \Omega \\ u^k &= u_D \quad \text{on } \partial\Omega, \end{aligned}$$

  2. update

$$\sigma^{k+1} = \frac{\mathcal{A}}{|\mathbf{p}^k/\sigma^k|^p}$$

3. Set $\varepsilon = |\sigma^{k+1} - \sigma^k|$ and $k = +1$

- end while

We assume that σ^k is uniformly bounded above and below in Ω for all k, such that the PDE problems for u^k are all well-posed.

2.2. Software Functionalities

The successive substitution Algorithm is fairly simple to implement, since each iteration only involves some arithmetic operations on known functions and the solution of a linear PDE problem.

For σ^k, we want to solve for ∇u^k in

$$\begin{aligned} \nabla(-\sigma^k \nabla u^k) &= 0 \text{ in } \Omega, \\ u^k &= u_D, \text{ on } \partial\Omega, \end{aligned} \tag{5}$$

We use a mixed formulation, such that we get a problem for the gradient ∇u^k, which is crucial for the reconstruction algorithm. In this way, we do not need to numerically differentiate u^k, which can introduce additional errors.

$$\begin{aligned} -\sigma^k \nabla u^k + \mathbf{p}^k &= 0 \text{ in } \Omega, \\ \nabla \cdot \mathbf{p}^k &= 0 \text{ in } \Omega, \\ u^k &= u_D, \text{ on } \partial\Omega, \end{aligned} \tag{6}$$

We fully utilize DarcyLite [10], which provide solver for equation 6 in MATLAB using Mixed Finite Element Method.

3. Illustrative Examples

We illustrate an example to test the implementation for $p = 1$, i.e. Current Density Impedance Imaging, the case for $p = 2$, can be implemented by modification a view lines within the main MATLAB script given below. Here, we consider the reconstruction of a piece-wise constant conductivity [11],

$$\sigma = \begin{cases} 1.0, & 0 < x < 0.5, 0 < y < 1 \\ 1.4, & 0.5 < x < 0, 0 < y < 1 \end{cases}$$

For simplicity, the exact u_σ in $\Omega = [0,1]^2$ is set to be

$$u_\sigma = cos(x - 0.5) * exp(y).$$

The forward problem is sought successively iterated on a uniform triangulation of the domain $\Omega = [0,1] \times [0,1]$.

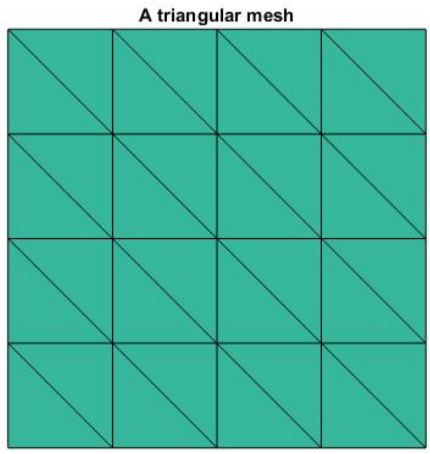

Figure 1: Domain Triangulation

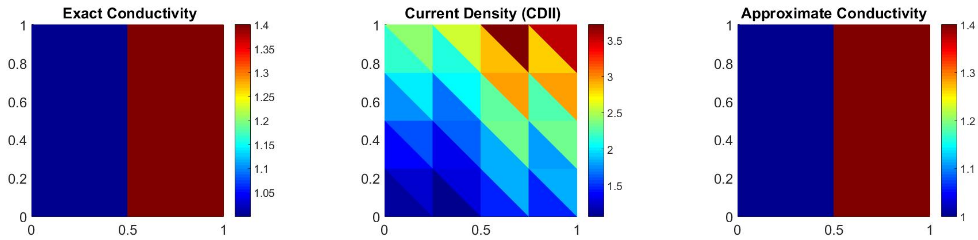

Figure 2: Exact Conductivity is on the left column, while on the right column shows the approximate conductivity after the iteration satisfied the given tolerance with rate convergence given in table 2. In the middle column, given the current density as artificial measurement data.

4. Impact

Despite the clear relevance of Hybrid inverse in multiple research facets, to the best of our knowledge, there are no open-source for Hybrid inverse packages available. To that end, HyEIT is developed using a MATLAB - based platform to solve the non-linear hybrid inverse problem.

k'th-iterates	$\frac{\|\sigma_{k+1}-\sigma_k\|_2}{\|\sigma_{k+1}\|_2}$	$\frac{\|\sigma_{k+1}-\sigma_k\|_2}{\|\sigma_{exact}\|_2}$
1	0.718569590056506	0.718569590056506
2	5.31152248204223e-16	5.31152248204223e-16

Table 2: Stopping error value criteria at each iterate

Due to the rapid advances with the field of Hybrid Inerse over the recent years by numerous researchers (for example, [22],[23] [24], [25]), we decided to initiate releasing the package HyEIT as an open-source communal project, which will allow users to continually improve HyEIT.

Potential near - term advances improving the applicability and performance of HyEIT over the coming years may include

(i) additional prior models, for example Bounded Variation [19], which is easily tested using Weak-Galerkin Element in `DarcyLite` as [20] studied for Electrical Impedance Tomography, and pointed out some estimates in Boundary Variation.

(ii) estimation of unknown boundaries and Uncertainty Quantification on boundary values,

(iii) nonGaussian noise models, and

(iv) different optimization approaches, such as full Newton, quasi - Gauss, gradient descent, primal dual interior point methods, and more[21].

Moreover, we look forward to extending the 2D HyEIT to a 3D version.

5. Conclusions

In this Note, we presented the MATLAB-based Hybrid Electrical Impedance Tomography Imaging (HyEIT) open-source package. The present version of the packages only provide a basic successive iterations solver by users, but open to the developer to adds-on for HyEIT problems with different prior models, constraints, optimization parameters, geometries, boundary conditions, and more. In the future, we hope to engage other researchers to further develop and improve the capabilities of HyEIT for a large suite of potential applications.

6. Conflict of Interest

No conflict of interest exists: We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

The lead author, would like to thank the Department of Mathematics, Institut Pertanian Bogor, who give exempt of teaching duties enjoying the pleasure of research.

References

[1] Garnadi, A.D. and Bahriawati, C., 2020. A mixed $RT_0 - P_0$ Raviart-Thomas finite element implementation of Darcy Equation in GNU Octave. https://doi.org/10.31223/osf.io/wjeq8

[2] Kim, Y. J., Kwon, O., Seo, J. K., & Woo, E. J. (2003). Uniqueness and convergence of conductivity image reconstruction in magnetic resonance electrical impedance tomography. 19(5), 1213–1225. doi:10.1088/0266-5611/19/5/312.

[3] Kwon, O., Woo, E. J., Yoon, J.-R., & Seo, J. K. (2002). Magnetic resonance electrical impedance tomography (MREIT): Simulation study of Jsubstitution algorithm. IEEE Transactions on Biomedical Engineering, 49(2), 160–167. doi:10.1109/10.979355.

[4] Langtangen, H. P. (2003). Computational partial differential equations: Numerical methods and diffpack programming. Texts in computational science and engineering. Berlin, Heidelberg: Springer. ISBN: 9783540434160.

[5] Nachman, A., Tamasan, A., & Timonov, A. (2011). Current density impedance imaging. Tomography and Inverse Transport Theory. doi:10.1090/conm/559/11076.

[6] Nachman, A., Tamasan, A., & Timonov, A. (2009). Recovering the conductivity from a single measurement of interior data. Inverse Problems, 25(3), 035014. doi:10.1088/0266-5611/25/3/035014.

[7] Bahriawati, C.& Carstensen, C.,(2005), Three MATLAB implementations of the lowest-order Raviart-Thomas MFEM with a posteriori error control, Computational Methods in Applied Mathematics Comput. Methods Appl. Math., 5,4,333–361,De Gruyter

[8] Adler A., Lionheart W.R.,(2006), Uses and abuses of EIDORS: an extensible software base for EIT Physiological Measurement, 27 (5) , p. S25

[9] Hoffmann, Kristoffer and Knudsen, Kim, Iterative reconstruction methods for hybrid inverse problems in impedance tomography, Sensing and Imaging, 15(1), 96, 2014, Springer

[10] Liu, Jiangguo and Sadre-Marandi, Farrah and Wang, Zhuoran, DarcyLite: A Matlab toolbox for Darcy flow computation, Procedia Computer Science, 80, 1301–1312, 2016, Elsevier

[11] Faouzi Triki and Tao Yin., 2020,Inverse conductivity equation with internal data,*arXiv*, eprint= 2003.13638

[12] Bal, G., 2012, Hybrid inverse problems and internal functionals, Inverse Problems and Applications: Inside Out Vol - II

[13] Kuchment, P. (2012). Mathematics of hybrid imaging: A brief review. In: Springer Proceedings in Mathematics, 183–208. doi:10.1007/978-88-470-1947-812.

[14] Widlak, T., & Scherzer, O. (2012). Hybrid tomography for conductivity imaging. Inverse Problems, 28(8), 084008. doi:10.1088/0266-5611/28/8/084008.

[15] Ammari, H., Bonnetier, E., Capdeboscq, Y., Tanter, M., & Fink, M. (2008). Electrical impedance tomography by elastic deformation, SIAM Journal on Applied Mathematics, 68(6), 1557–1573. doi:10.1137/070686408.

[16] Capdeboscq, Y., Fehrenbach, J., de Gournay, F., & Kavian, O. (2009). Imaging by modification: Numerical reconstruction of local conductivities from corresponding power density measurements, SIAM Journal on Imaging Sciences, 2(4), 1003–1030, doi:10.1137/080723521.

[17] Gebauer, B., & Scherzer, O. (2008). Impedance-acoustic tomography. SIAM Journal on Applied Mathematics, 69(2), 565–576. doi:10.1137/080715123.

[18] Kuchment, P., & Kunyansky, L., 2011, 2D and 3D reconstructions in acousto-electric tomography, Inverse Problems, 27(5), 055013. doi:10.1088/0266-5611/27/5/055013.

[19] Charalambopoulos, Antonios and Markaki, Vanessa and Kourounis, Drosos, The inverse conductivity problem via the calculus of functions of bounded variation, Mathematical Methods in the Applied Sciences, 43,8,5032–5072, 2020

[20] Liang, Ying and Zou, Jun, Weak Galerkin Method for Electrical Impedance Tomography,arXiv preprint arXiv:2011.04991, 2020

[21] Hoffmann, Kristoffer and Knudsen, Kim, Iterative reconstruction methods for hybrid inverse problems in impedance tomography, Sensing and Imaging, 15,1,96,2014

[22] Hubmer, Simon and Knudsen, Kim and Li, Changyou and Sherina, Ekaterina, Limited-angle acousto-electrical tomography, Inverse Problems in Science and Engineering, 27, 9, 1298–1317, 2019

[23] Adesokan, Bolaji James and Jensen, Bjørn and Jin, Bangti and Knudsen, Kim, Acousto-electric tomography with total variation regularization, Inverse Problems, 35, 3, 035008, 2019

[24] Li, Changyou and An, Kang and Zheng, Kuisong and Lesselier, Dominique, A complete framework for acousto-electric tomography with numerical examples, IEEE Access, 2020

[25] Li, Changyou and An, Kang and Zheng, Kuisong, The Levenberg–Marquardt Method for Acousto-Electric Tomography on Different Conductivity Contrast, Applied Sciences, 10, 10, 3482, 2020

Current executable software

Main MATLAB Script

```
%%
%% run_HyEIT_CDII_MFEM_v0
%%
%% ...
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ...
   %%
%% Algorithm 1 Successive Iteration algorithm ...
                                %%
%% Define:  \sigma^0,                            %%
%%          maximum number of iteration K and ...
                         %%
%%          tolerance level T. ...
                                      %%
%% Set k = 0 and \varepsilon = 2 T. ...
                                 %%
%% while k < K and \varepsilon > T do ...
                               %%
```

```
%% 1 ...                                                              ...
   %%
%% 2  Solve for u^k in ...
                                                   %%
%% 3           \nabla (- \sigma^k  nabla u^k) = 0 in \Omega ...
        %%
%% 4            u  =  u_D   on \partial \Omega , ...
                     %%
%% 5      update ...
                                                        %%
%% 6           \sigma^{k+1} = a/|\nabla u^k| ...
                        %%
%% 7      Set \varepsilon = |\sigma^{k+1} - \sigma^k| and k ...
   =+1  %%
%% end while ...
                                                            %%
%% ...
   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ...
   %%
%%
%% Modified from :
%% run_Darcy_MFEM_TriRT0P0.m
% Assuming CondKs is an elementwise constant scalar
% James Liu, ColoState; 2012/07--2017/02
% DarcyLite
%
% Modified by A.D. Garnadi, Institut Pertanian Bogor; ...
   2019/03-2020/12
% For Riset Kolaborasi Indonesia - 2019/2020
% Hybrid Imaging : EIT+MRI and USG+EIT
%
% Conductivity := CondKs
%
%
% clc;
clear all;  close all;
format compact;  format short e;

%% Setting up the problem
xa = 0;  xb = 1;  yc = 0;  yd = 1;
BndryDescMat = [xa,yc,xb,yc, 0,-1;...
                xb,yc,xb,yd, 1, 0;...
                xb,yd,xa,yd, 0, 1;...
                xa,yd,xa,yc,-1, 0];
% Setting up the Darcy equation & boundary conditions on ...
   four sides:
% 1 for Dirichlet, 2 for Neumann, 0 for coding convenience ...
   in Matlab
```

```
BndryCondType = [0;1;1;1;1]; % Dirichlet Data
% EqnBC = EqnBC_Poisson_Ex00Linear;         % Case 0
EqnBC = EqnBC_Poisson_ExactTrikiYin;      % Case 1
% EqnBC = EqnBC_Poisson_KelloggIntfc;     % Case 2
% % Domain for case 2 : [-1,1]x[-1,1]
%
%
% BndryCondType = [0;1;1;1;1];

%

%% Discretization: Mesh generation and preparation
% Modified later to accept the size of an image,
k_powerExperiment = [0:4]; % Experiment index
k_power = k_powerExperiment(3)
n = 2^k_power
%
nx = n;  ny = n;
TriMesh = RectDom_TriMesh_GenUnfm(xa,xb,nx,yc,yd,ny,2);
TriMesh = TriMesh_Enrich2(TriMesh,BndryDescMat);
TriMesh = TriMesh_Enrich3(TriMesh,BndryDescMat);

%% Setting up quadratures
GAUSSQUAD = SetGaussQuad(5,25,13);

%% Sampling the conductivity/permeability (as a scalar)
fxnKs = EqnBC.fxnKs ;
CondKs = Darcy_SmplnPermSca_TriMesh(fxnKs, TriMesh, GAUSSQUAD);

%% Sorting out boundary edges: Dirichlet, Neumann
DirichletEdge = find(BndryCondType(TriMesh.BndryEdge+1)==1);
NeumannEdge   = find(BndryCondType(TriMesh.BndryEdge+1)==2);
%%
show_TriMesh(TriMesh,25)
%% TriMesh: Show the mesh in a specified figure window
%
% show_TriMesh_labels(TriMesh,26,1,1,1)
%show_TriMesh_labels(TriMesh,fwn,ch0,ch1,ch2)
%% TriMesh: Show node, element, edge labels in a specified ...
   window
% fwn: figure window number
% ch0: choice for showing node labels (0 or 1)
% ch1: choice for showing edge labels (0 or 1)
% ch2: choice for showing element labels (0 or 1)
%
%% Assembling and solving...
% tic;
[sln,MatA,MatB,GlbRHS] = Darcy_MFEM_TriRT0P0s_AsmSlv(...
  EqnBC,TriMesh,CondKs,DirichletEdge,NeumannEdge,GAUSSQUAD);
```

```
% toc

%% Computing numerical ElectricalPotential, velocity, flux, etc.
[NumElPotEm,NumElCurrEmCntr,NumerFlux,LMCR,FluxDscp] = ...
  Darcy_MFEM_TriRT0P0_PresVelFlux(TriMesh,sln,BndryDescMat);
%
% NumElPotEm(i)      = p_h(i), 1≤ i ≤ NumEl
% NumElCurrEmCntr(i) = u_h(i,[1 2]), 1≤ i ≤ NumEl
%             u_h = - K grad p_h
% NumerFlux(i,[1 2 3]) = u_h(i,[1 2])'(n(i,[1 2]))
%
%% GENERATE SYNTHETIC DATA :
%
%
NumCurrPowerDensity  = zeros(size(CondKs)) ;
%
for ix=1:length(NumCurrPowerDensity)
    absJix = norm(NumElCurrEmCntr(ix,:),2) ;
    %% For CDII, sigma |\nabla u|:
    %  NumCurrPowerDensity(ix) = absJ;
    NumCurrPowerDensity(ix) = absJix;
    %% For AEIT, sigma |\nabla u|^2:
    %  NumCurrPowerDensity(ix) = (1/CondKs(ix))*absJix^2;
end
% NumElPotEm      = Elementwise Numerical ElectricalPotential
% NumElCurrEmCntr = Elementwise Numerical Velocity/Current ...
   (at the centre of Element)
%% Presenting numerical & graphical results
%
show_TriMesh_ScaVecEm_mix(TriMesh, NumCurrPowerDensity, ...
   NumElCurrEmCntr, ...
  23, 'MFEM:  Current Density (CDII) and Current', 1);
%
%% show_TriMesh_ScaVecEm_mix(TriMesh, NumCurrPowerDensity2, ...
   NumElCurrEmCntr, ...
%%  24, 'MFEM:  Current Density (AEIT) and Current', 1);
%%
% DATA for Successive Iteration
%
%% CDII:
%       element coordinate NumCurrPowerDensity
%% AEIT:
%       element coordinate NumCurrPowerDensity2
%
%%
%% Define:  \sigma^0,                       ...
                         %
Sigmak = ...
   mean(NumCurrPowerDensity)*ones(size(NumCurrPowerDensity)) ;
```

```
% Sigmak = CondKs ;
%
absGradU = zeros(size(NumCurrPowerDensity)) ; % Memory ...
    allocation
%%maximum number of iteration K
%
Kmax = 10 ;
Norm2OfRatio = zeros(1,10)'    ; % Place holder
Norm2OfRatioRel = zeros(1,10)' ; % Place holder
NumGradEmCntr = zeros(size(NumElCurrEmCntr(:,1))) ;% Place ...
    holder
%%          tolerance level T. ...
                                    %%
TolLevel = (max(NumCurrPowerDensity) - ...
    min(NumCurrPowerDensity))/2 ;
%% Set k = 0 and \varepsilon = 2 T. ...
                               %%
kIteration = 0 ; VarEpsilon = 2*TolLevel ;
%% while k < K and \varepsilon > T do ...
                             %%
while (kIteration < Kmax) & (VarEpsilon >TolLevel)
%% 1 ...
                                                              ...
    %%
%% 2  Solve for u^k in ...
                                           %%
%% 3           \nabla (- \sigma^k  nabla u^k) = 0 in \Omega ...
         %%
%% 4            u  =  u_D   on \partial \Omega , ...
                      %%
%% Assembling and solving...
% tic;
[sln,MatA,MatB,GlbRHS] = Darcy_MFEM_TriRT0P0s_AsmSlv(...
  EqnBC,TriMesh,Sigmak,DirichletEdge,NeumannEdge,GAUSSQUAD);
% toc
%%
%% Computing numerical ElectricalPotential, velocity, flux, etc.
[NumElPotEm,NumElCurrEmCntr,NumerFlux,LMCR,FluxDscp] = ...
  Darcy_MFEM_TriRT0P0_PresVelFlux(TriMesh,sln,BndryDescMat);
%
%% 5      update ...
                                                %%
%% 6           \sigma^{k+1} = a/|\nabla u^k| ...
                         %%
% absGradU = sqrt(NumerFlux(:,1).^2 + NumerFlux(:,2).^2) ;
NumGradEmCntr(:,1) = NumElCurrEmCntr(:,1)./Sigmak ;
NumGradEmCntr(:,2) = NumElCurrEmCntr(:,2)./Sigmak ;
absGradU = sqrt(NumGradEmCntr(:,1).^2 + ...
    NumGradEmCntr(:,2).^2) ;
```

```
Sigmak1  = NumCurrPowerDensity./absGradU ;
%% 7      Set \varepsilon = |\sigma^{k+1} - \sigma^k| and k ...
   =+1  %%
kIteration = kIteration + 1;
%% Computing errors :
%% Compute \Vert Sigmak1 - Sigmak \Vert_(L^2)/\Vert Sigma ...
   \Vert_(L^2)
% [L2ErrPotential] = ...
   Darcy_MFEM_TriRT0P0_Err(EqnBC,TriMesh,sln,GAUSSQUAD);
Norm2OfRatioRel(kIteration) = norm((Sigmak1 - ...
   Sigmak),2)/norm(Sigmak1,2) ;
Norm2OfRatio(kIteration) = norm((Sigmak1 - ...
   Sigmak),2)/norm(CondKs,2) ;
%
VarEpsilon = norm((Sigmak1 - Sigmak),1) ;
Sigmak =  Sigmak1 ;
%% end while ...
                                                              %%
end
show_TriMesh_ScaEm_ClrImg(TriMesh,CondKs,24,'   Exact ...
   Conductivity  ')
show_TriMesh_ScaEm_ClrImg(TriMesh,NumCurrPowerDensity,26,'  ...
   Current Density (CDII) ')
show_TriMesh_ScaEm_ClrImg(TriMesh,Sigmak1,27,'  Approximate ...
   Conductivity  ')
disp('==========FINISH=========')
return;
%
```

Test Case

```
function EqnBC = EqnBC_Poisson_ExactTrikiYin
%% EqnBC: EqnBC_Poisson_ExactTrikiYin
% Exact Solution of
%
%  Inverse conductivity equation with internal data,
%  Triki, Faouzi and Yin, Tao,
%  arXiv preprint arXiv:2003.13638},
%  2020
%
% Let $\Omega_1$ be the unit square [0,1]^2
% Set
% p(x,y) =  cos(x-0.5).*exp(y);
% Ks = \sigma = 1.0,    0<x<0.5, 0<y<1
%                1.4,    0.5<x<0, 0<y<1
%
% Now, define the boundary fluxes as
```

```
% f_n = u
% Observe, that¬:
% p satisfies \[\nabla k \nabla p =0,\]
% with Dirichlet data
% g_D = p =
%
%
EqnBC = struct('fxnKs',@fxnKs, 'fxnK',@fxnK, 'fxnf',@fxnf, ...
  'fxnpD',@fxnpD, 'fxnuN',@fxnuN, ...
  'fxnp',@fxnp, 'fxnpg',@fxnpg, 'fxnu',@fxnu);

% Diffusion coeff. or permeability as a scalar
function Ks = fxnKs(pt)
  x = pt(:,1);
  y = pt(:,2);%
  Ks = ones(size(x) ;
  %Ks = \sigma = 1.0,     0<x<0.5, 0<y<1
  %             1.4,     0.5<x<0, 0<y<1
  ix = (x => 0.5) ;
  Ks(ix) = 1.4 ;
end

% Diffusion coeff. or permeability as 2x2 SPD matrix
function K = fxnK(pt)
  x = pt(:,1);
  NumPts = size(pt,1);
  ix = (x => 0.5) ;
  K = zeros(NumPts,2,2);
%
  K(:,1,1) = 1;
  K(:,2,2) = 1;
%
  K(ix,1,1) = 1.4;
  K(ix,2,2) = 1.4;
end

% The right-hand side function in the Poisson/Darcy equation
function f = fxnf(pt)
  x = pt(:,1);  %y = pt(:,2);
  f = zeros(size(x)) ;
end

% Dirichlet boundary condition
function pD = fxnpD(pt)
  % x = pt(:,1);  y = pt(:,2);
  pD = fxnp(pt);
end

% Neumann boundary condition
```

```
% None

% Known exact "pressure" solution
function p = fxnp(pt)
  x = pt(:,1);  y = pt(:,2);
  % u_exact = cos(x - 0.5)*exp(y)
  p = cos(x-0.5).*exp(y);
end

% Known gradient of the exact "pressure" solution
function pg = fxnpg(pt)
  x = pt(:,1);  y = pt(:,2);
  pg = [sin(x-0.5).*exp(y), cos(x-0.5).*exp(y)];
end

% Known exact solution for "velocity"
function u = fxnu(pt)
  x = pt(:,1);  y = pt(:,2);
  u1 = - sin(x-0.5).*exp(y) ;
  u2 = - cos(x-0.5).*exp(y);
  u = [u1,u2];
end

end
```