Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Upen2DTool: A Uniform PENalty Matlab tool for inversion of 2D NMR relaxation data

V. Bortolotti [a], L. Brizi [b], P. Fantazzini [b], G. Landi [c], F. Zama [c,*]

[a] *Department of Civil, Chemical, Environmental, and Materials Engineering, University of Bologna, Italy*
[b] *Department of Physics and Astronomy, University of Bologna, Italy*
[c] *Department of Mathematics, University of Bologna, Italy*

A R T I C L E   I N F O

Article history:
Received 5 June 2019
Received in revised form 29 July 2019
Accepted 30 July 2019

Keywords:
2D NMR data inversion
Uniform penalty
Adaptive regularization
SVD filtering

A B S T R A C T

The analysis of Nuclear Magnetic Resonance (NMR) relaxation of ^{1}H nuclei gives fundamental information about the structure of different types of porous media via the determination of 1-dimensional and 2-dimensional (2D) distributions of the typical NMR parameters (longitudinal T_1 and transverse T_2 relaxation times). Despite the great amount of literature available on this topic, only a few open source software are available, usually after a direct request to the authors. This work aims to stimulate the comparison of different approaches by making our open source inversion Matlab Package (Upen2DTool) available, to encourage its application and extension to the various 2D NMR problems. The solution algorithm applies the Uniform PENalty principle and automatically computes locally adapted regularization parameters and approximate solutions. Moreover a Windows binary code is available for testing for non-Matlab users.

Code metadata

Current code version	v.1.0
Permanent link to code/repository used for this code version	https://github.com/fzama63/Upen2DTool.git
Legal Code License	GPL v.3
Code versioning system used	none
Software code languages, tools, and services used	Matlab
Compilation requirements, operating environments & dependencies	none
If available Link to developer documentation/manual	https://site.unibo.it/softwaredicam/en/software/2dupen
Support email for questions	fabiana.zama@unibo.it

1. Motivation and significance

In this paper we present a Matlab package, named Upen2DTool, to compute the 2-dimensional (2D) distribution (2D map) of Nuclear Magnetic Resonance (NMR) relaxation times from NMR experimental data. The structure of different types of porous media can be analyzed through measurements of NMR relaxation of ^{1}H nuclei. Parameters like the longitudinal (T_1) and the transverse (T_2) relaxation times, as well as the self-diffusion molecular coefficient, can be determined and associated to both properties of the saturating fluids and porous media. To obtain the distribution of relaxation times from NMR data is an inverse ill-posed problem modeled by a first kind Fredholm Integral Equation having separable exponential kernel. Upen2DTool applies the Upen2D algorithm (the 2D extension of the Uniform PENalty principle [1]) to compute locally adapted regularization parameters and approximate solutions by solving a sequence of least squares regularized problems. The software can be successfully used on NMR data obtained from samples with pore size distributions ranging from a few nanometers to micrometers. Therefore Upen2DTool can be usefully applied to analyze the structure of various porous media: from biological systems to hydro-carbon bearing sedimentary rocks. Moreover, being an open source code, it can be easily adapted to NMR data acquired by different types of scanners, making it possible to extend the

* Corresponding author.
E-mail address: fabiana.zama@unibo.it (F. Zama).

https://doi.org/10.1016/j.softx.2019.100302

range of problems analyzed by means of the Uniform Penalty approach.

Despite the great amount of literature about 2D NMR inversion algorithms (see for example [2–6] and references therein), only a few of them are available from the authors [3,5], making it difficult to compare the different approaches on specific data sets. A number of commercial products do exists (see http://www.spincore.com/nmrinfo/software_s.shtml for a survey) but to the authors' best knowledge they are linked to specific relaxometers for particular data analysis and are not suitable for algorithm comparisons.

In Section 2 we report a brief description of the problem and numerical method, all the details can be found in the cited papers. The software structure is considered in Section 3 and finally some illustrative examples are reported in Section 4.

2. Algorithm description

Upen2D computes the distribution $F(T_1, T_2)$ of T_1 and T_2 relaxation times from 2D-NMR relaxation data $S(t_1, t_2)$ depending on t_1, t_2 evolution times. Considering, for example, a conventional Inversion-Recovery (IR) or Saturation Recovery (SR) experiment detected by a Carr–Purcell–Meiboom–Gill (CPMG) pulse train [7], where the kernel is separable, the relaxation data can be expressed as:

$$S(t_1, t_2) = \iint_0^\infty k_1(t_1, T_1)k_2(t_2, T_2)F(T_1, T_2)\, dT_1\, dT_2 + e(t_1, t_2) \quad (1)$$

where $e(t_1, t_2)$ represents Gaussian additive noise. The kernels are decaying exponential functions whose expression depends on the specific NMR experiment. The unknown distribution $F(T_1, T_2)$ is known to be greater or equal to a constant $\ell \in \mathbb{R}$. Considering $M_1 \times M_2$ samples of the times $t_1 \times t_2$, and $N_1 \times N_2$ samples of the relaxation times $T_1 \times T_2$, problem (1) is discretized as:

$$(\mathbf{K}_2 \otimes \mathbf{K}_1)\mathbf{f} + \mathbf{e} = \mathbf{s} \quad (2)$$

where $\mathbf{K}_1 \in \mathbb{R}^{M_1 \times N_1}$, $\mathbf{K}_2 \in \mathbb{R}^{M_2 \times N_2}$ are the discretized exponential kernels, $\mathbf{s} \in \mathbb{R}^M$, $M = M_1 \cdot M_2$, is the discrete vector of the measured noisy data, $\mathbf{f} \in \mathbb{R}^N$, $N = N_1 \cdot N_2$, is the vector reordering of the distribution and $\mathbf{e} \in \mathbb{R}^M$ is the vector with the discretized noise. Upen2D uses multiple-parameter Tikhonov regularization and solves the minimization problem

$$\min_{\mathbf{f} \geq \ell} \left\{ \|(\mathbf{K}_2 \otimes \mathbf{K}_1)\mathbf{f} - \mathbf{s}\|^2 + \sum_{i=1}^{N} \lambda_i (\mathbf{L}\mathbf{f})_i^2 \right\} \quad (3)$$

where $\|\cdot\|$ is the L_2 norm, $\mathbf{L} \in \mathbb{R}^{N \times N}$ is the discrete Laplacian operator and λ_i are the regularization parameters, $i = 1, \dots, N$. The numerical method is an iterative procedure where, at each iteration, suitable values for the λ_i's are determined by imposing that all the non-null products $\lambda_i (\mathbf{L}\mathbf{f})_i^2$ have the same constant value (Uniform PENalty principle [1]) and an approximated distribution is obtained by solving a problem of the form (3) with Newton's Projection method.

Upen2DTool allows us to apply Singular Value Decomposition (SVD) filter, in order to improve the quality and the efficiency of the reconstructions.

2.1. SVD filter

The SVD filter allows reducing the data dimensions by projecting $\mathbf{s}$ onto a lower-dimensional subspace spanned by the first left singular vectors of the kernel matrices $\mathbf{K}_1$ and $\mathbf{K}_2$. Let $\mathbf{K}_1 = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T$ and $\mathbf{K}_2 = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T$ be the SVD of $\mathbf{K}_1$ and $\mathbf{K}_2$, where $\mathbf{U}_1 \in \mathbb{R}^{M_1 \times M_1}$, $\mathbf{V}_1 \in \mathbb{R}^{N_1 \times N_1}$, $\mathbf{\Sigma}_1 \in \mathbb{R}^{M_1 \times N_1}$ and $\mathbf{U}_2 \in \mathbb{R}^{M_2 \times M_2}$, $\mathbf{V}_2 \in \mathbb{R}^{N_2 \times N_2}$, $\mathbf{\Sigma}_2 \in \mathbb{R}^{M_2 \times N_2}$, with $\mathbf{U}_i, \mathbf{V}_i$ orthogonal matrices and $\mathbf{\Sigma}_i$ diagonal matrices of the singular values, $i = 1, 2$. The data fitting term of (3) can be written as follows:

$$\|(\mathbf{K}_2 \otimes \mathbf{K}_1)\mathbf{f} - \mathbf{s}\|^2 = \|(\mathbf{\Sigma}_2\mathbf{V}_2^T \otimes \mathbf{\Sigma}_1\mathbf{V}_1^T)\mathbf{f} - (\mathbf{U}_2^T \otimes \mathbf{U}_1^T)\mathbf{s}\|^2. \quad (4)$$

Considering only the singular values greater than a given threshold $\tau > 0$ and substituting $\mathbf{K}_1$ and $\mathbf{K}_2$ by their truncated SVDs, we obtain the following SVD compressed least squares problem [8]:

$$\min_{\mathbf{f} > \ell} \left\{ \|(\hat{\mathbf{K}}_2 \otimes \hat{\mathbf{K}}_1)\mathbf{f} - (\hat{\mathbf{U}}_2^t \otimes \hat{\mathbf{U}}_1^t)\mathbf{s}\|_2^2 + \sum_{i=1}^{N} \lambda_i (\mathbf{L}\mathbf{f})_i^2 \right\} \quad (5)$$

where

$$\hat{\mathbf{K}}_1 = \hat{\mathbf{\Sigma}}_1\hat{\mathbf{V}}_1^t, \quad \hat{\mathbf{K}}_2 = \hat{\mathbf{\Sigma}}_2\hat{\mathbf{V}}_2^t \quad (6)$$

The matrices $\hat{\Sigma}_i$ have the first $\rho_i \ll \min(M_i, N_i)$ diagonal elements of Σ_i greater than τ and $\hat{\mathbf{U}}_i, \hat{\mathbf{V}}_i$ are made by the first ρ_i columns of $\mathbf{U}_i, \mathbf{V}_i$ respectively ($i = 1, 2$), therefore the components of the truncated SVD are:

$$\hat{\mathbf{U}}_1 \in \mathbb{R}^{M_1 \times \rho_1}, \hat{\mathbf{V}}_1 \in \mathbb{R}^{N_1 \times \rho_1}, \ \hat{\mathbf{\Sigma}}_1 \in \mathbb{R}^{\rho_1 \times \rho_1}$$
$$\hat{\mathbf{U}}_2 \in \mathbb{R}^{M_2 \times \rho_2}, \hat{\mathbf{V}}_2 \in \mathbb{R}^{N_2 \times \rho_2}, \ \hat{\mathbf{\Sigma}}_2 \in \mathbb{R}^{\rho_2 \times \rho_2}$$

2.2. Iterative Upen2D algorithm

Let $p_\mu^{(k)}, c_\mu^{(k)}$ denote the value of the gradient and Laplacian of the reconstructed map $\mathbf{f}^{(k)}$ at index μ (see [1] for detailed explanation) the steps of the Upen2D algorithm can be summarized as follows:

```
1. Using $\tau > 0$ compute the truncated SVDs of the matrices $\mathbf{K}_1$ and $\mathbf{K}_2$ (6)
2. Compute $\hat{\mathbf{s}} = (\hat{\mathbf{U}}_2^t \otimes \hat{\mathbf{U}}_1^t)\mathbf{s}$
3. Set $k = 0$. Using the tolerance parameter $Tol_{GP}$ and the Gradient Projection algorithm, compute a starting guess:
       $\mathbf{f}^{(0)} = \arg\min_{\mathbf{f} \geq \ell} \|(\hat{\mathbf{K}}_2 \otimes \hat{\mathbf{K}}_1)\mathbf{f} - \hat{\mathbf{s}}\|_2^2.$
4. repeat
    (a) Compute
        $\lambda_i^{(k)} = \dfrac{\|(\hat{\mathbf{K}}_2 \otimes \hat{\mathbf{K}}_1)\mathbf{f}^{(k)} - \hat{\mathbf{s}}\|_2^2}{N\left(\beta_0 + \beta_p \max_{\mu \in I_i}(p_\mu^{(k)})^2 + \beta_c \max_{\mu \in I_i}(c_\mu^{(k)})^2\right)}$
        $i = 1, \dots, N$    (7)
    (b) Using tolerance parameters $Tol_{NP}, Tol_{CG}$ for Newton Projection and Conjugate Gradient algorithms, compute
        $\mathbf{f}^{(k+1)} = \arg\min_{\mathbf{f} \geq \ell} \|(\hat{\mathbf{K}}_2 \otimes \hat{\mathbf{K}}_1)\mathbf{f} - \hat{\mathbf{s}}\|_2^2 + \sum_{i=1}^{N} \lambda_i^{(k)} (\mathbf{L}\mathbf{f})_i^2$
    until $\|\mathbf{f}^{(k+1)} - \mathbf{f}^{(k)}\| \leq Tol_{UPEN} \|\mathbf{f}^{(k)}\|$
```

The Kronecker products are efficiently computed as matrix-matrix products, using the following property:

$$(\hat{\mathbf{K}}_2 \otimes \hat{\mathbf{K}}_1)\mathbf{x} = vec(\hat{\mathbf{K}}_1 \mathbf{X} \hat{\mathbf{K}}_2^T)$$

where $vec(V)$ is obtained by vector reordering of the matrix V and the matrix $\mathbf{X}$ is obtained by reorganizing the elements of the vector $\mathbf{x}$.

Table 1
Default tolerance parameters.

Tol_{CG}	Tol_{NP}	Tol_{GP}	Tol_{UPEN}
10^{-5}	10^{-5}	10^{-2}	10^{-3}

```
 -----------------------------------------------------------
UPen2D Input Parameters
 upen_tol=1.000000e-03,
 Projected Gradient Tol =1.000000e-03
 Projected Newton Tol=1.000000e-05
 Conjugate Gradient Tol =1.000000e-05
 SVD Threshold =1e-04
 Data size= 23 x 20
-----------------------------------------------------------
Number of Inversion channels:  horizontal 80, vertical  80
Final Relative Residual Norm =2.4942e-02
Total UPen2D Iterations = 9
Total CG Iterations = 60250
Total Projected Newton Iterations = 106
Computation Time = 54.11000 s.
------------------------------------------------------------
```

Fig. 1. Content of file `Parameters.txt` for the `T1-T2` test.

2.3. Upen2D parameters

The algorithm depends on several parameters that can be grouped as follows:

- Regularization Parameters: β_{00}, β_0, β_p, β_c. These parameters control the proper choice of the regularization parameters λ_i (7). The value β_{00} is a scale parameter that depends on the type of sample measured, see [1,9] and [8] for a detailed discussion.
- Tolerance Parameters used to stop the iterative methods: Tol_{CG}, Tol_{NP}, Tol_{GP}, Tol_{UPEN}. The default values are reported in Table 1
- SVD threshold parameter τ relative to the filtering of the input data [8,9]. Values in the range $[10^{-6}, 10^{-4}]$ guarantee good quality reconstructions and very high data compression factors ($> 90\%$).

3. Software structure

The Upen2DTool package consists of the following files and folders:

- `DATA`. A folder used to store the input data and parameters files. The software has three examples whose data are stored in the `T1-T2`, `T2-T2` and `T1_T2_Synth` subfolders. For a detailed description of the files please refer to the included software documentation.
- `output_files`. A folder containing the following output files produced by the computations:
  - `2D_Distribution.txt`. Map of relaxation times.
  - `residual.txt` Matrix containing the components of the residual vector $\mathbf{r} = (\mathbf{K}_2 \otimes \mathbf{K}_1)\mathbf{f} - \mathbf{s}$ where $\mathbf{f}$ is the computed map of the relaxation times.
  - `Parameters.txt`. Relevant output parameters such as computation time, iteration numbers, and the value of the relative residual norm Rr, given by the ratio between the residual and the data 2−norms:

    $$Rr = \frac{\|\mathbf{r}\|_2}{\|\mathbf{s}\|_2}$$
  - `T1.txt`,`T2.txt`. Files containing the vectors T_1 and T_2 in (1)
  - `t1.txt`, `t2.txt`. Files containing the vectors t_1 and t_2 in (1)
- Matlab functions that implement the Upen2D method, graphical utility functions and input–output functions (see the included manual for further details.)

The `main.m` script can be used to run all the included examples.

Once the computation is finished, four images are plotted: the surface and contours of the relaxation times map and the projection onto the horizontal and vertical dimensions.

3.1. Software functionalities

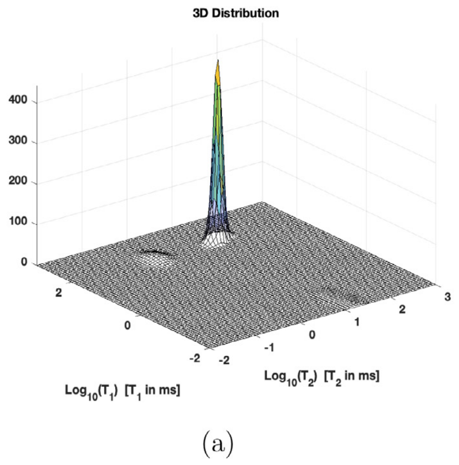
We show here the different functionalities that can be activated by setting the proper parameters:

- *Implemented models.* It is possible to choose between the $T1 - T2$ or $T2 - T2$ cases by setting the parameter `FL_typeKernel` to the values 1 or 4 respectively. In the $T1-T1$ case the kernels have the following expression:

  $$k_1(t_1, T_1) = 1 - 2\exp(-t_1/T_1), \quad k_2(t_2, T_2) = \exp(-t_2/T_2),$$

  while in the $T2 - T2$ case the kernels have the following expression:

  $$k_1(t_1, T_{21}) = \exp(-t_1/T_{21}), \quad k_2(t_2, T_{22}) = \exp(-t_2/T_{22}).$$



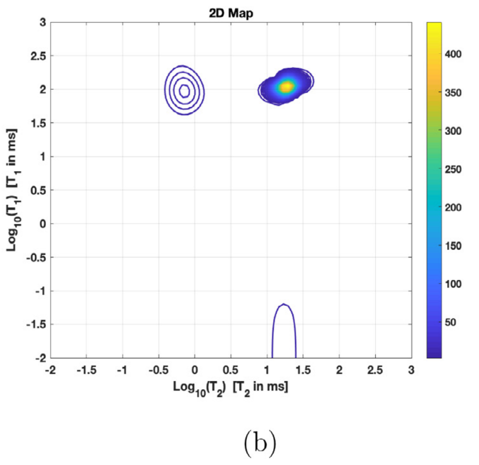


Fig. 2. T1–T2 experiment. 3D map (a) and 2D contour map (b) of the 2D reconstructed relaxation time map.

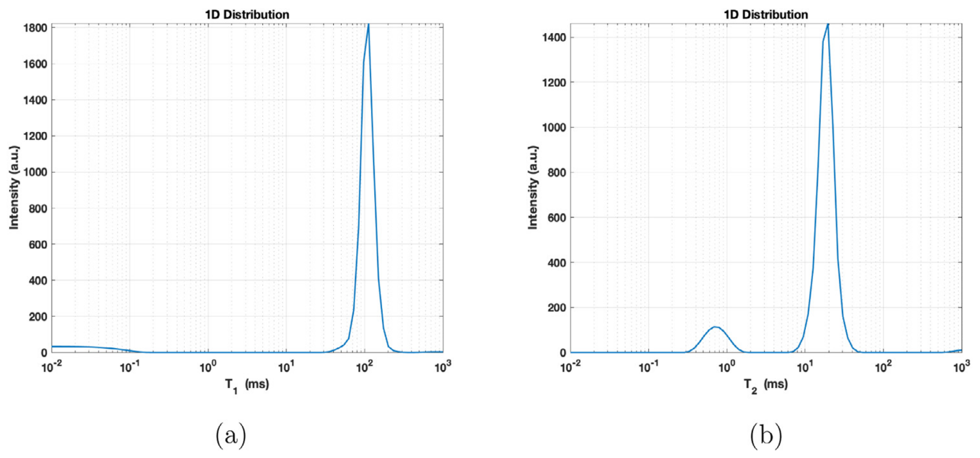


Fig. 3. T1–T2 experiment: Projections along the T_2 (a) and T_1 (b) dimension.

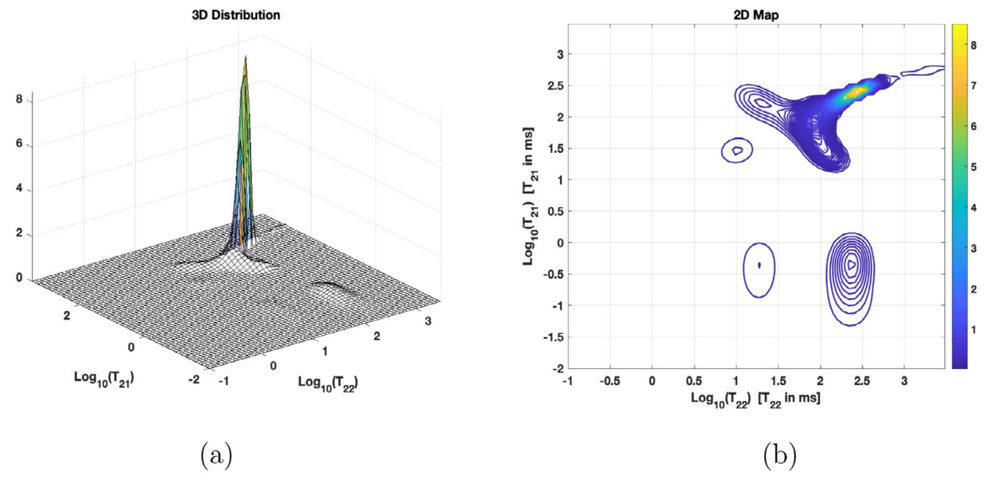


Fig. 4. T2–T2 experiment. 3D map (a) and 2D contour map (b) of the 2D reconstructed relaxation map.

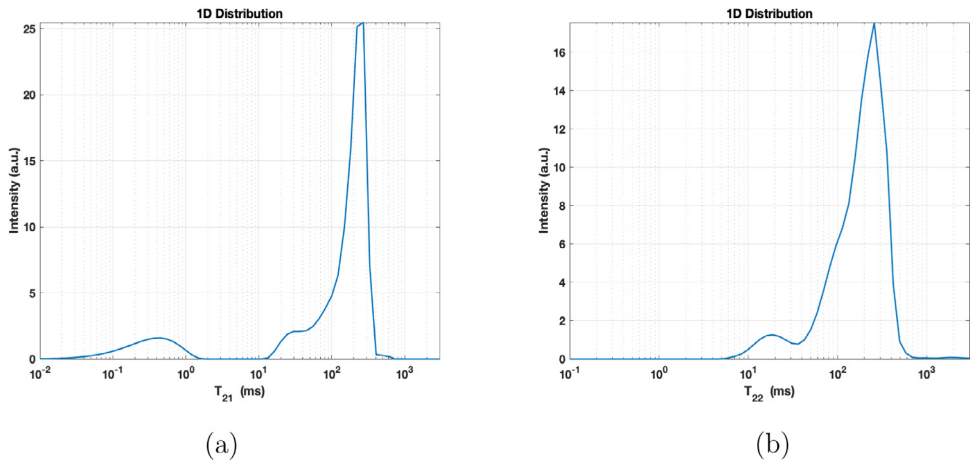


Fig. 5. T2–T2 experiment. Projections along the T_{21} (a) and T_{22} (b) dimension.

- *SVD filter*. The SVD filter is applied by setting the parameters in file `FilePar.par`: `par.svd.svd` to 1 and `par.svd.soglia` to a positive value $\tau > 0$, suggested values are in the range $[10^{-6}, 10^{-2}]$, see [8,9] for details.

4. Illustrative examples

We report here two examples where the different functionalities of the software are represented. We added also a synthetic test problem in the folder `T1_T2_Synth`. The details are discussed in the included software manual.

```
 -----------------------------------------------
 Upen2D Input Parameters
 upen_tol=1.000000e-03,
 Projected Gradient Tol =1.000000e-03
 Projected Newton Tol=1.000000e-05
 Conjugate Gradient Tol =1.000000e-05
 SVD Threshold =1e-06
 Data size= 20 x 30
 -----------------------------------------------
Number of Inversion channels:  horizontal 64, vertical  64
Final Relative Residual Norm =9.8456e-03
Total UPen2D Iterations = 14
Total CG Iterations = 9837
Total Projected Newton Iterations = 70
Computation Time = 10.83000 s.
 ------------------------------------------------
```

Fig. 6. Content of file `Parameters.txt` for the T2-T2 test.

4.1. T1-T2 test

The data represent a test on fresh cement pastes prepared by filling a 10 mm external diameter glass NMR tube with White Portland Cement mixed with distilled water for 2 min at 1600 r.p.m. (w/c ratio 0.5). The NMR measurements were performed at 25 °C by a homebuilt relaxometer based on a PC-NMR portable NMR console (Stelar, Mede, Italy) and a 0.47 T Jeol electromagnet (Jeol Ltd. Tokyo, Japan). The IR-CPMG 2D data has size 48 $\times$ 1000 and the reconstructed map $F \in \mathbb{R}^{N_1 \times N_2}$ 80 $\times$ 80 ($N_1 \times N_2$). The lower bound $\ell = 0$ is used. The SVD threshold ($\tau = 10^{-4}$) gives about 99% compression factor reducing the data to the following size $460 = 23 \times 20$. The reconstructed 3D map and 2D contour map are reported in Fig. 2 while the projections onto the T_1 and T_2 axes are reported in Fig. 3. The output file `Parameters.txt` represented in Fig. 1 reports information about the computation cost in terms of iteration numbers and time on an Intel i7 processor with 16 GB Ram using Matlab 2018a.

Observing the T_2 projection in Fig. 3(b) we can see two peaks at 1 ms and 20 ms: the position and height are consistent to the results obtained by [2](Victoria University implementation). A small artifact is visible in the contour map (Fig. 2(b)) as well as in the T_1 projection (Fig. 3(a)).

4.2. T2-T2 test

In this example a sample of Maastricht stone saturated with water was measured with a CPMG–CPMG 2D sequence. During the CPMG–CPMG preparation period of the sequence, the echo time (TE) was 6.61 ms and the number of echoes was 128. The mixing time lasts 100 s. During the CPMG–CPMG detection period, the TE was of 300 µs and the number of echoes was 2800. The repetition time was 3 s and number of scans 16. As expected we observe in Fig. 4(b) the asymmetric out of diagonal peaks due possible to mass exchange in the population spins.

The reconstructed map F has size $N_1 \times N_2$, where $N_1 = N_2 = 64$, the lower bound $\ell = 0$ is used. The SVD threshold ($\tau = 10^{-6}$) gives about 99% compression factor reducing the data to the following size $400 = 20 \times 30$. The reconstructed 3D map and 2D contour map are reported in Fig. 4 while the projections onto the T_1 and T_2 axes are reported in Fig. 5. The content of the output file `Parameters.txt`, represented in Fig. 6, reports among the others things, information about the computation cost in terms of iteration numbers and time on an Intel i7 processor with 16 GB Ram using Matlab 2019a.

5. Impact

Despite the importance of this research field and the great variety of applications, ranging from biological tissues to rocks or cement pastes, only a few open-source Matlab based codes are available. By releasing Upen2DTool not only did we mean to extend the range of applications of the Upen2D method, but we also intended to offer a tool where other methods could be included for extensions as well as comparison purposes.

Due to the rapid advances within this field Upen2DTool could be regarded as a general framework where users can give their contribution with extensions and improvements. Potential near term advances could be the inclusion of compressed sensing techniques for the inversion of 2D-NMR data when it is necessary to reduce the conventional relaxation data acquisition times (i.e. in vivo studies) [10].

6. Conclusions

In this paper we present the Matlab-based 2D Uniform Penalty Tool: Upen2DTool. The software promotes the application of the Upen2D algorithm to compute the 2D distribution of NMR relaxation times.

The present version includes two data sets relative to the $T1 - T2$ and $T_2 - T_2$ experiments. In the future we hope to involve more researchers to develop and improve the capabilities of Upen2DTool by including different NMR experiments. Finally binary codes will be soon delivered for Windows, Linux and Mac Operating Systems to enable non-Matlab users to experiment the potentialities of the Upen2D method.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been partially supported by the Italian Institute GNCS - INdAM.

References

[1] Bortolotti V, Brown RJS, Fantazzini P, Landi G, Zama F. Uniform penalty inversion of two-dimensional NMR relaxation data. Inverse Problems 2016;33(1):19.
[2] Venkataramanan L, Song Y-Q, Hurlimann MD. Solving fredholm integrals of the first kind with tensor product structure in 2 and 2.5 dimensions. IEEE Trans Signal Process 2002;50(5):1017–26.
[3] Urbańczyk M, Bernin D, Koźmiński W, Kazimierczuk K. Iterative thresholding algorithm for multiexponential decay applied to PGSE NMR data. Anal Chem 2013;85(3):1828–33.
[4] Mitchell J, Chandrasekera T, Gladden L. Numerical estimation of relaxation and diffusion distributions in two dimensions. Prog Nucl Magn Reson Spectrosc 2012;62:34–50.
[5] Teal P, Eccles C. Adaptive truncation of matrix decompositions and efficient estimation of NMR relaxation distributions. Inverse Problems 2015;31(4):045010.
[6] Campisi-Pinto S, Levi O, Benson D, Cohen M, Resende MT, Saunders M, Linder C, Wiesman Z. Analysis of the regularization parameters of primal–dual interior method for convex objectives applied to 1H low field nuclear magnetic resonance data processing. Appl Magn Reson 2018;49(10):1129–50.
[7] Blümich B. Essential NMR. Springer-Verlag; 2005.
[8] Bortolotti V, Brizi L, Fantazzini P, Landi G, Zama F. Filtering techniques for efficient inversion of two-dimensional nuclear magnetic resonance data. J Phys Conf Ser 2017;904(1):012005.
[9] Bortolotti V, Brown RJS, Fantazzini P, Landi G, Zama F. I2DUPEN: Improved 2DUPEN algorithm for inversion of two-dimensional NMR data. Microporous Mesoporous Mater 2018;269:195–8.
[10] Bai R, Cloninger A, Czaja W, Basser PJ. Efficient 2D MRI relaxometry using compressed sensing. J Magn Reson 2015;255:88–99.