User Guide for

DigiGel Analysis software

Version 1.1

Developed by Mattan Ze'ev Becker and Maya Georgia Pelah, under the supervision of Prof. Ronen Berkovich, the Department of Chemical Engineering, Ben Gurion University of the Negev, Israel.

Also presented in:

Conformation-driven mechanical duality: viscoelastic and poroelastic switching in protein hydrogels

Mattan Ze'ev Becker[1], Maya Georgia Pelah[1], Ofek Avrahami[1], Ionel Popa[2], Ronen Berkovich[1, 3]

[a] Department of Chemical Engineering, Ben-Gurion University of the Negev, Beer-Sheva 8410501, Israel.

[b] Department of Physics, University of Wisconsin-Milwaukee, Milwaukee, WI 53211, USA.

[c] Ilze Katz Institute for Nanoscience and Technology, Ben-Gurion University of the Negev, Beer-Sheva 8410501, Israel.

(*Acta Biomaterialia*, under review February 2025).

1. ***DigiGel*** software for data analysis – user instructions.

DigiGel is a computerized tool to facilitate data analysis in mechanical characterization of soft materials. Specifically, indentation measurements, followed by a relaxation phase, to describe time-dependent mechanical characteristics of soft materials, namely agreement of the experimental data with either the viscoelastic or poroelastic model. All Theoretical framework required to understand the analysis performed by the ***DigiGel*** software is described in section 2 in the article. The application is suitable for data analysis of both micro- and macro-indentation experiments, using either a spherical or conical indenter. A quick installation file and the full code are available at **github.com/BeckerMattan**.

After installing the software, the icon shown in Figure M1a will appear on the computer desktop. Once launched, the *DigiGel* main screen will open (Figure M1b). Here the user will enter the experiment name, and indicate the number of measured samples, number of trigger forces (set-point)

that were applied at each experiment, indenter type, controller settling time and specify the trigger forces. Then, the user will upload the measured raw data, which will be organized in separate CSV (Excel) files for each trigger force, and phase (indentation/relaxation: the adhesion phase will be added in future iterations). All relaxation data uploaded should have the same relaxation duration, due to the data normalization applied by the software. The github.com/BeckerMattan library contains a folder with exemplary (dummy) CSV files containing data from two experimental samples at three trigger forces. The following figures will relate to data from these files. The experiment, as a numerical entity, will be saved as a .json file, however the analyzed data and figures could be downloaded as Excel and image files. The user can also upload previous experiments (*.json files), for additional work.

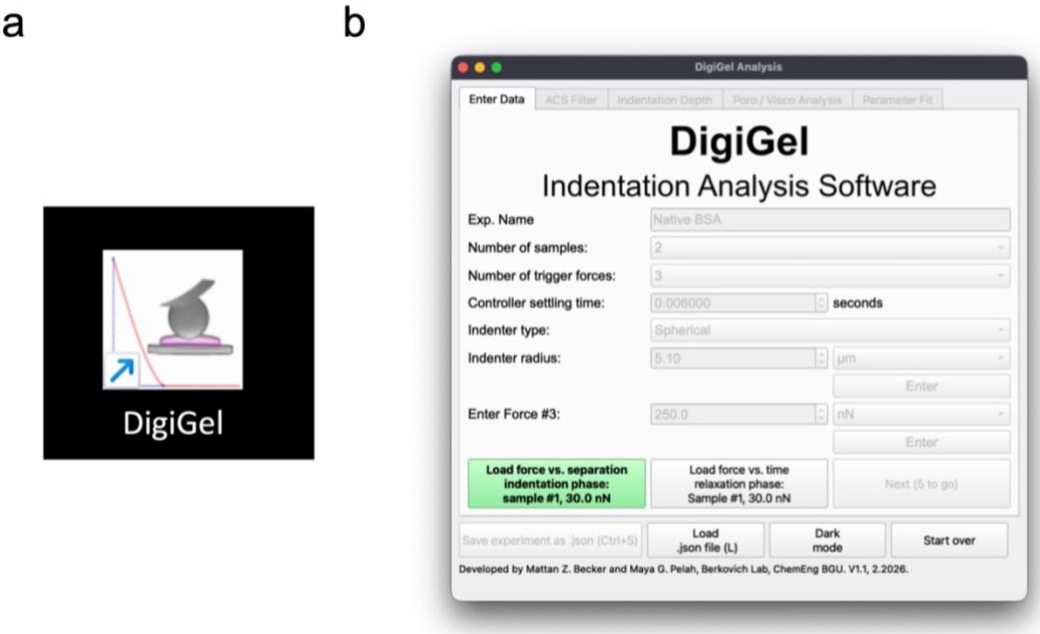

Figure M1. *DigiGel* analysis software. a. Desktop icon. b. Initial screen in which the experiment is defined and its raw data uploaded as CSV files.

Once the data is uploaded, and the experiment was saved, an initial pre-processing step is performed, in which relaxation curves are time-aligned to the start of the hold segment, and resampled to a common time grid prior to clustering. Then, the user will continue to the next step, in which the data can be clustered to identify different populations, and filtering it using the "ACS filter" tab. The user has two options: manual filtering with individual trace selection, or automatic filtering using the automated curve screening (ACS) filter algorithm (discussed further in the subsequent section). For the latter, the user will press the "Calculate cluster number" button and a graph will appear, presenting the Elbow method graph for the current data (Figure M2a). In this mode, the software determines the optimal cluster number using the Elbow method, though users can override this value if they believe a different number would yield better results. Next, the "Apply clustering" button is pressed, and the user interface presents the different traces according to the clustering performed by the ACS algorithm. The user can view individual traces and select either to filter out individual traces or entire

clusters (Figure M2b). If sub-populations are apparent, the user can choose to save raw data of specific cluster for further analysis. Following filtration of all samples and force triggers, additional traces are randomly removed to ensure equal measurement counts across all force triggers and samples, thereby preventing statistical bias. The user can choose to save a filtering report which specifies all the measurements that were filtered out alongside the reason for filtering. Next, the user will continue to the "Indentation depth" tab, where the indentation depth is calculated for each individual trace using the force vs. separation data. The software automatically detects the point of contact, by binary segmentation using *ruptures* python library [1], and calculates the indentation depths (Figure M2c). The user can choose one of four calculation algorithms: sum of squared deviations cost ("l2"), Gaussian-likelihood cost ("normal"), kernelized mean-change cost ("rbf") and rank-based cost ("rank"). After the automatic calculation, the user can browse the results visually and change individual measurements using the user interface.

Figure M2. ACS filtration and indentation depth calculation in DigiGel analysis software. a. Elbow method graph for ACS. b. Filtering by the user interface after clustering. c. Indentation depth calculation tab.

Finally, the user will continue to the results tabs. The first tab that presents results is the "Poro / Visco analysis" tab, which shows the full graphic representation of the relaxation results (Figure M3a) and results relevant for distinguishing between viscoelastic and poroelastic materials (Figure M3b and M3c). The graphs generated correspond to the results presented in Figures 4 and 5 in the manuscript. Here, the user can choose to save the graphs and the filtered and analyzed data.

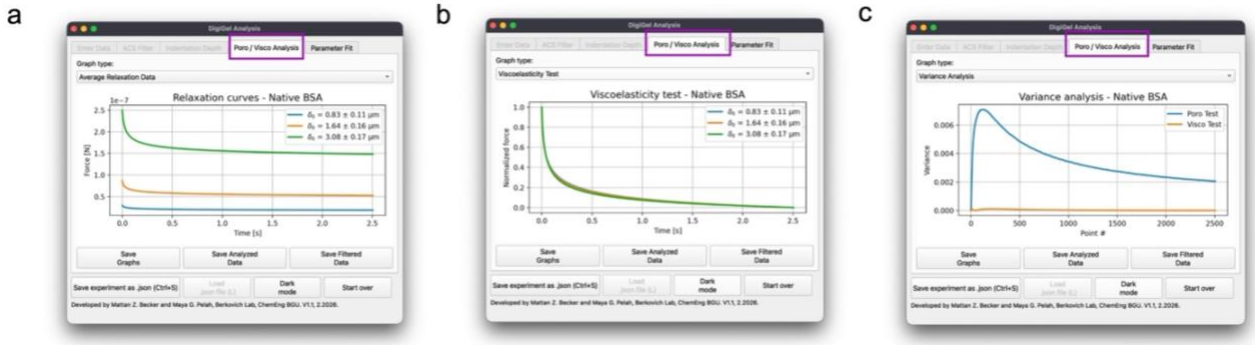

Figure M3. The "Poro / Visco analysis" tab in *DigiGel* analysis software. a. Force-relaxation results for all force triggers. b. Normalized graph according to the viscoelasticity test, where overlapping

graphs represent results of a viscoelastic material. c. Variance analysis between the viscoelasticity and the poroelasticity test.

The last tab ("Parameter fit", Figure M4) aids in numerically characterizing the tested material. After determining whether viscoelasticity or poroelasticity prevails in the previous tab, the user can choose to fit the data to either the poroelastic model, where G, ν and D_p are yielded, or to the fractional viscoelastic model, which yields β and C_β values. Curve fitting is performed using the *curve_fit* function from Python's *scipy* optimize package [2].

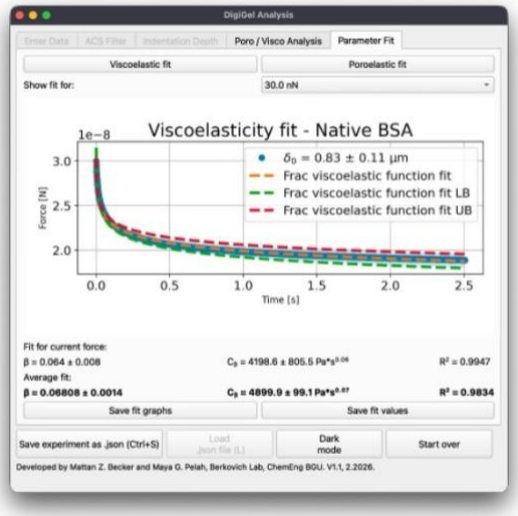

Figure M4. The "Parameter fit" tab in *DigiGel* analysis software. The data can be fitted to either the poroelastic or the fractional viscoelastic model.

2. Automated Curve Screening (ACS) for filtering faulty measurements in force spectroscopy data.

As described above, the first step of analysis within the *DigiGel* software is quality control, meaning filtering out faulty data points. We use an ACS algorithm that leverages K-means clustering to streamline the initial data classification process, thereby enhancing efficiency and reducing human intervention. This algorithm is implemented in Python, using the *scikit-learn* python library [3], and identifies patterns within the force spectroscopy data without prior labeling, utilizing the Elbow Method to determine the optimal number of clusters. This method effectively balances model complexity and the explained variance, allowing for more efficient and unbiased data analysis.
The ACS step uses standard unsupervised clustering as a data-organization tool. In this context, unsupervised methods are used only to cluster relaxation curves based on overall shape similarity, without using labeled training data. Importantly, ACS is used solely to assist automated quality screening and does not introduce model-based inference or identification of distinct mechanical populations. All retained curves are subsequently pooled and analyzed using the same viscoelastic and poroelastic fitting procedures.

K-means clustering is one of the most widely used unsupervised learning algorithms. It aims to partition a dataset into K distinct, non-overlapping subsets (or clusters), where each data point belongs to the cluster with the nearest mean. The algorithm follows an iterative process to minimize the within-cluster variance, commonly referred to as the sum of squared distances between data points and their respective cluster centroids [4,5].

The K-Means algorithm is a clustering technique that begins by determining the number of clusters K and randomly initializing centroids from the dataset. It then calculates the distance between each data point and all centroids, assigning points to their nearest centroid to form distinct clusters. The algorithm continues by computing new centroids as the mean of all data points within each respective cluster. This process repeats, recalculating distances, reassigning points, and updating centroids, until convergence is reached, which occurs when the centroids no longer change significantly between iterations.

To determine the optimal number of clusters in K-means clustering, the Elbow Method is used. This method involves running the K-means algorithm with different values of K and plotting the sum of squared distances against the number of clusters. The point where the decrease starts to slow down, resembling an "elbow," suggests the appropriate number of clusters [8,9]:

$$WCSS = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - \mu_i\|^2 \quad (1)$$

where K is the number of clusters, x is a data point, C_i is the i^{th} cluster, m_i is the centroid of the i^{th} cluster, and $||x - m_i||^2$ is the squared Euclidian distance between x and m_i.

The Elbow Method is a heuristic procedure for determining the optimal number of clusters in K-means clustering by systematically executing the algorithm across a range of K values. For each iteration, the Within-Cluster Sum of Squares (WCSS) is calculated, quantifying the sum of squared distances between data points and their assigned cluster centroids. These values are then visualized in a plot of WCSS versus K, where the researcher identifies the "elbow point" – the critical juncture at which the rate of WCSS reduction diminishes significantly. This inflection point represents the optimal cluster quantity, balancing model complexity and explanatory power by indicating where additional clusters would yield diminishing returns in reducing intra-cluster variance. In all, the Elbow Method helps in selecting a K value that balances model complexity and the amount of explained variance, avoiding both overfitting and underfitting.

The clustering was applied directly to the force-time relaxation curves. Each curve was processed to obtain an equal length time-series representation suitable for comparison across measurements and then normalized to emphasize similarities and differences in the relaxation profile rather than absolute force values. This representation preserves the temporal characteristics of the relaxation process, which is essential for distinguishing between different curve behaviors. The optimal number of clusters for each dataset was determined using the Elbow method, as described above.

To illustrate this approach, relaxation experiments on BSA-PBH were conducted with three set-points (SP) at different force levels: 50 nN, 120 nN, and 200 nN. Figure M5 illustrates the classification results of the force-time relaxation curves (N = 1064) with SP of 200 nN, categorized into 4 clusters.

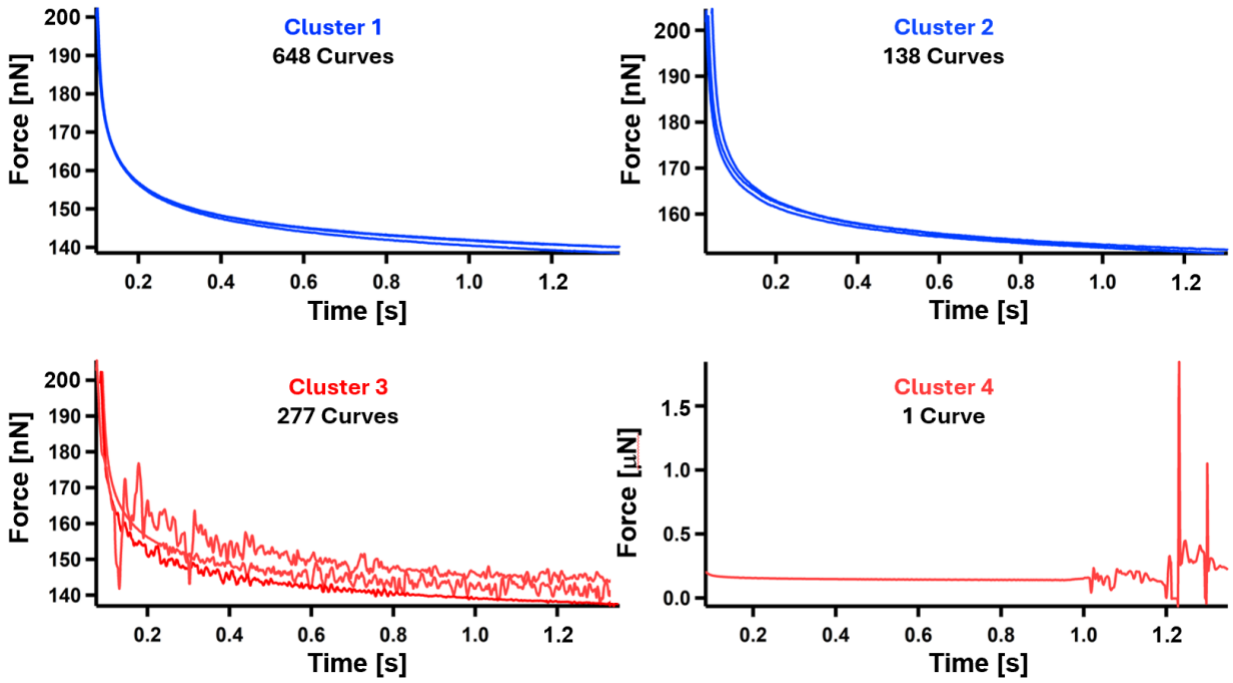

Figure M5. Exemplary measured force relaxation traces. Results shown at set point of 200 nN representing the 4-cluster categorization classification identified by the ACS algorithm.

Figure M6 shows the results of applying the elbow method to the data. A distinct elbow is not clearly observed, since the sample statistics is relatively low (thousands of measurements only), and a distinct elbow is expected to increase for higher statistics. The elbow graph is automatically presented in the *DigiGel* software, for the user's convenience. The user can manually override the number of clusters for clustering by the K-means algorithm.

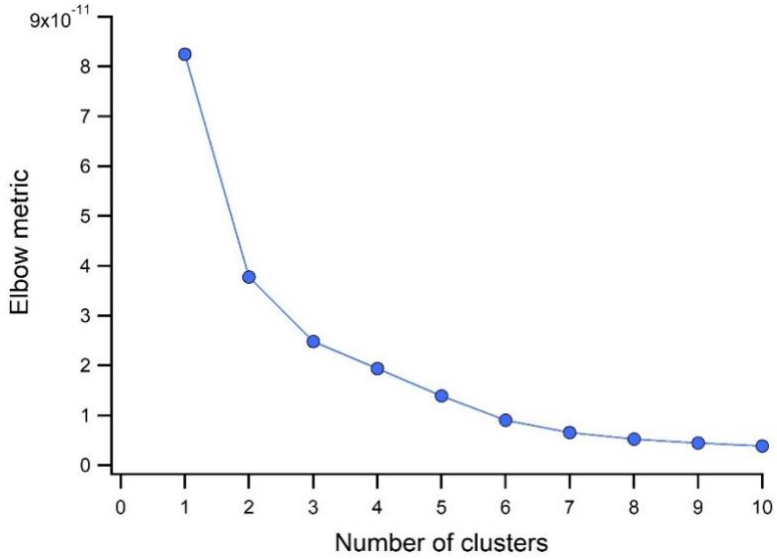

Figure M6. Determination of the best cluster using the Elbow method. The results represent force relaxation traces at set point of 200 nN.

The classification algorithm successfully categorized the force-time relaxation curves into distinct clusters, significantly reducing analysis. The distribution of clusters at various force set-points demonstrates the algorithm's robustness and accuracy, with most data classified into the expected clusters. Clusters 3 and 4 show intentionally disrupted measurements that were proactively inflicted to test the capability of the classification algorithm. These disruptions represent characteristic disturbances that occasionally arise during AFM measurements on hydrogels (e.g., a tip-height adjustment during a force map), typically producing a single disrupted trace similar to the curve shown in Cluster 4 of Figure M5. Figure M7 summarizes the clustering of the data at all three forces. Following the clustering the relaxation force traces are normalized as described above, averaged, and fitted with the relevant model.

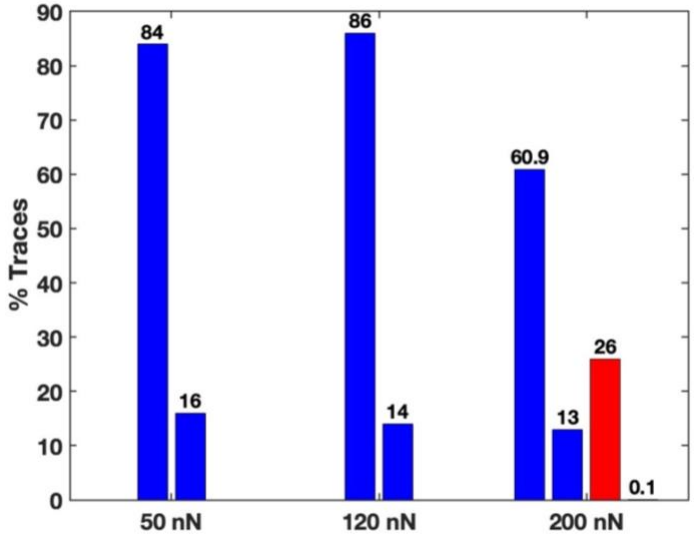

Figure M7. Distinguishing between faulty and valid data using ACS. The graph shows the percentage of traces in each group at different force set-points. The numbers above each bar indicate the percentage of traces in the cluster division at different force set-point.

REFERENCES

[1] C. Truong, L. Oudre, N. Vayatis, Selective review of offline change point detection methods, Signal Processing 167 (2020) 107299. https://doi.org/10.1016/j.sigpro.2019.107299.

[2] P. Virtanen, R. Gommers, T.E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S.J. van der Walt, M. Brett, J. Wilson, K.J. Millman, N. Mayorov, A.R.J. Nelson, E. Jones, R. Kern, E. Larson, C.J. Carey, İ. Polat, Y. Feng, E.W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E.A. Quintero, C.R. Harris, A.M. Archibald, A.H. Ribeiro, F. Pedregosa, P. van Mulbregt, A. Vijaykumar, A. Pietro Bardelli, A. Rothberg, A. Hilboll, A. Kloeckner, A. Scopatz, A. Lee, A. Rokem, C.N. Woods, C. Fulton, C. Masson, C. Häggström, C. Fitzgerald, D.A. Nicholson, D.R. Hagen, D. V. Pasechnik, E. Olivetti, E. Martin, E. Wieser, F. Silva, F. Lenders, F. Wilhelm, G. Young, G.A. Price, G.-L. Ingold, G.E. Allen, G.R. Lee, H. Audren, I. Probst, J.P.

Dietrich, J. Silterra, J.T. Webber, J. Slavič, J. Nothman, J. Buchner, J. Kulick, J.L. Schönberger, J.V. de Miranda Cardoso, J. Reimer, J. Harrington, J.L.C. Rodríguez, J. Nunez-Iglesias, J. Kuczynski, K. Tritz, M. Thoma, M. Newville, M. Kümmerer, M. Bolingbroke, M. Tartre, M. Pak, N.J. Smith, N. Nowaczyk, N. Shebanov, O. Pavlyk, P.A. Brodtkorb, P. Lee, R.T. McGibbon, R. Feldbauer, S. Lewis, S. Tygier, S. Sievert, S. Vigna, S. Peterson, S. More, T. Pudlik, T. Oshima, T.J. Pingel, T.P. Robitaille, T. Spura, T.R. Jones, T. Cera, T. Leslie, T. Zito, T. Krauss, U. Upadhyay, Y.O. Halchenko, Y. Vázquez-Baeza, SciPy 1.0: fundamental algorithms for scientific computing in Python, Nat. Methods 17 (2020) 261–272. https://doi.org/10.1038/s41592-019-0686-2.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830. https://doi.org/10.5555/1953048.2078195.

[4] A.K. Jain, Data clustering: 50 years beyond K-means, Pattern Recognit. Lett. 31 (2010) 651–666. https://doi.org/10.1016/j.patrec.2009.09.011.

[5] K. Kandali, L. Bennis, O. El Bannay, H. Bennis, An Intelligent Machine Learning Based Routing Scheme for VANET, IEEE Access 10 (2022) 74318–74333. https://doi.org/10.1109/ACCESS.2022.3190964.

[6] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Trans. Acoust. 26 (1978) 43–49. https://doi.org/10.1109/TASSP.1978.1163055.

[7] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, E. Keogh, Experimental comparison of representation methods and distance measures for time series data, Data Min. Knowl. Discov. 26 (2013) 275–309. https://doi.org/10.1007/s10618-012-0250-5.

[8] R.L. Thorndike, Who Belongs in the Family?, Psychometrika 18 (1953) 267–276. https://doi.org/10.1007/BF02289263.

[9] T.M. Kodinariya, P.R. Makwana, Review on determining number of Cluster in K-Means Clustering, International Journal of Advance Research in Computer Science and Management Studies 1 (2013). www.ijarcsms.com.