NUMERICAL ANALYSIS OF THE DROP SHAPE DURING WICKING IN POROUS MEDIA

Daciana BOTTA[1,2], Corneliu BĂLAN[3]

A numerical algorithm for interface analysis of wicking drops was developed. The code identifies the interface points and subsequently calculates the diameter, volume and contact angle of the drop. As the code can work with multiple frames, it can evaluate the evolution in time of these parameters and calculate the spreading velocity and the flow rate. The performance of the code was tested with solutions of poly(ethylene oxide) of 0.5% and 5% (w/v) concentration. The mean execution time was 0.5 seconds per frame. Therefore, the code can greatly improve the analysis of fluids wicking in porous media, where numerous frames are involved.

Keywords: image processing, numerical analysis, drop interface, wicking, porous media.

1. Introduction

The imbibition of liquids in porous media has importance in many industries, such as textile manufacturing, industrial coating of fibers, ink-jet printing, filtration of liquid aerosols, and the development of paper-based sensing devices [1]. The final stain area that a drop will make is an important parameter in the aforementioned domains, therefore it is a necessity to be able to accurately predict the behaviour of fluids in porous media.

The interaction between fluids and porous media has been divided in two processes. Wetting is the term used to describe the initial displacement of air by the fluid entering the porous matrix, whereas wicking represents the transport of fluid along the fibers, by capillary action [2-5]. Moreover, the wicking phenomenon takes place in two phases of different kinetics [6,7]. In the first phase, the liquid spreads on the surface and penetrates the porous substrate, while in the second phase, the fluid is fully contained in the substrate and continues to spread as an effect of capillary force. Both phases were taken into account in previous studies [8-11], but most works were focused on studying the phenomenon from above, which allows for observing only the evolution of stain area.

[1] Phd. Student, REOROM Laboratory, University POLITEHNICA of Bucharest, Romania
[2] Research Assistant, National Institute of Materials Physics, Romania, e-mail: daciana.botta@infim.ro
[3] Professor, REOROM Laboratory, University POLITEHNICA of Bucharest, Romania, e-mail: corneliu.balan@upb.ro

The wicking process is usually analysed by recording the phenomenon and extracting information from singular frames. However, as the phenomenon takes place very fast, high-speed cameras are required, leading to a numerous quantity of frames to be analysed. As this process is tedious, methods for automating the analysis procedure were implemented, with the use of Photoshop and MATLAB [2]. However, this was done as well mostly for droplets seen from above, where information can only be obtained from the stain area.

For the study of drop interfaces techniques such as the axisymmetric drop-shape analysis (ADSA), have been developed to measure the interfacial tensions and contact angles of pendant drops, sessile drops, and bubbles [12,13]. This technique involves a numerical method to fit a theoretical Laplacian curve with known surface tension values to the experimental profile of a drop. The value of surface tension is therefore found as the best match between the theoretical and experimental profiles of the drop [14].

In this work, we propose a numerical code for the analysis of the first phase of wicking. The purpose of the code is to identify the interface of the drop and calculate parameters such as diameter, volume and contact angle. The novelty of this work is that it allows the study of drops whose interfaces change in time, instead of sessile droplets. In comparison with other techniques, no information about the drop or the fluid used have to be known beforehand, as all the parameters will be calculated from the shape of the drop. Moreover, the use of dyes, which can influence the surface tension of the fluid, is not needed, because the proposed code can easily detect the edges of clear fluids, even if these reflect the surrounding environment.

2. Experimental section

2.1 Materials

Poly(ethylene oxide) (PEO), Mv ~ 1,000,000, and Whatman 1 Chromatography Paper were purchased from Sigma-Aldrich (St. Louis, USA). A Nikon camera with a long focal distance lens (Tamron 90 mm) were used for recording the wicking phenomenon. Two light sources were used: Erbauer Lewo 2400-Li LED and Fiber-Lite MI-150 (Dolan Jenner Industries).

2.2 Measurement procedure

Solutions of PEO dissolved in water were prepared with two concentrations: 0.5% and 5% (w/v). Drops with volumes ranging between 1 and 2 μl were placed on the paper with the use of a micropipette.

The flow was recorded from sideview at 60 fps. For ensuring a good quality of the images, with low noise, the paper was illuminated from two sides: one light source was placed behind the drop, and the other below the paper (Figure 1).

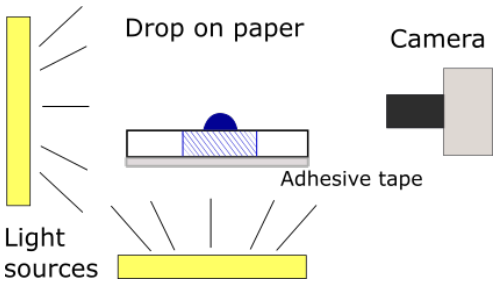

Fig. 1. Experimental set-up for recording the drop wicking.

Singular frames were extracted from the recordings and further analyzed in MATLAB R2022a, which was used on a laptop with a 1.80 GHz processor and 16 GB RAM for processing time testing. The computation time was measured for each frame.

2.3 Numerical code algorithm

The code assumes from the start that the image is in RGB mode, so it is first converted to grayscale, and subsequently to binary. The binarization threshold was maintained the same (x > 151) for all the images analyzed in this study, and it can be manually adjusted for different lighting conditions. In order to find the interface between the fluid and the surrounding air, a for-loop was performed on the binary matrix, which identifies the first non-zero element (white pixel) from each column. For each element, its position on rows and columns was stored in two vectors, which therefore, contained the X and Y coordinates for each point of the interface.

Using a reference image for scale, the vectors for the X and Y coordinates were multiplied with a correction factor, so that they would contain the information in millimeters, instead of pixels. The X-vector was then translated, so that the peak would be centered on X = 0. In order to accurately describe the interface shape, a seventh order polynomial function was used to fit the points. Having the interface in this position, the drop volume could be approximated as the solid of revolution around the Y-axis:

$$V(x) = 2\pi \int_0^r x \cdot f(x)\,, \tag{1}$$

where r is the drop radius, and it is taken as the X coordinate of the last interface point. Having this value, the diameter can also be calculated as $D = 2r$.

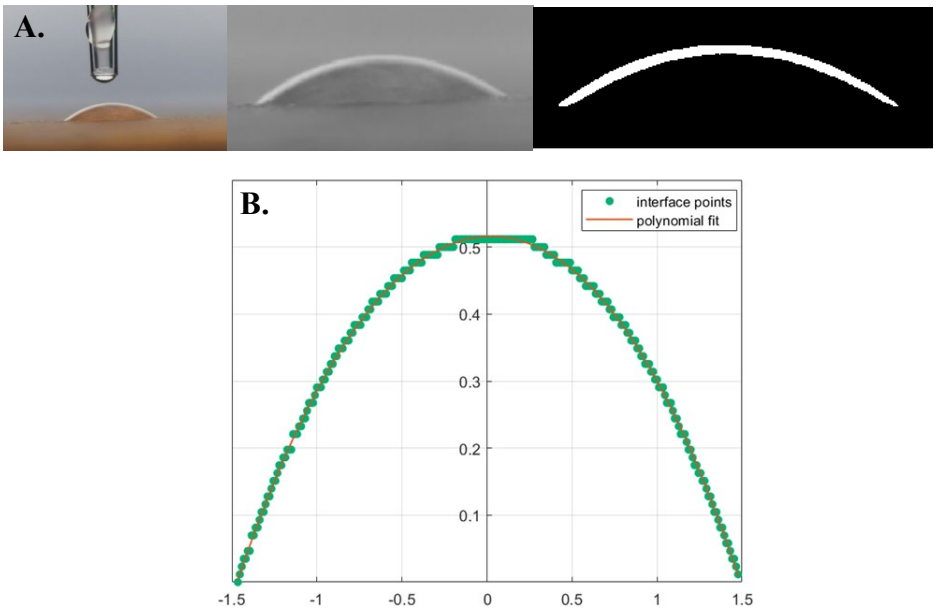

Fig. 2. The steps involved in the image processing procedure: **A.** transforming the image from the original RGB-mode to binary and **B.** plotting and fitting the drop interface

Moreover, using the fitting curve, the contact angle of the drop can be obtained as the arctangent of the slope. Using the first 30 points of the interface, the first derivative of the curve was evaluated, and the value of the contact angle, θ, was calculated as

$$\theta = \tan^{-1} f'(x). \tag{2}$$

The interface analysis was performed in a for-loop, which ran for each frame found in the specified folder. The values for the frame number, diameter, volume and contact angle were saved as vectors. The vector containing the frame numbers was afterwards used to transform the information to seconds, therefore containing the timescale for the recorded event. Finally, the wicking velocity, the fluid flow and the variation of contact angle in time were calculated as the first derivatives of the stored vectors.

3. Results and discussions

3.1 Algorithm evaluation

In order to verify the correctitude of the interface representation and the subsequent calculated values for diameter and volume, the numerical code was

tested with images of basic mathematical shapes. The shapes used for this purpose were a parabola of equation $y = -\frac{x^2}{4} + 5$ and circle with the radius R = 20 mm. The values obtained from the code were compared with the theoretical ones and the measurement error was calculated (Table 1).

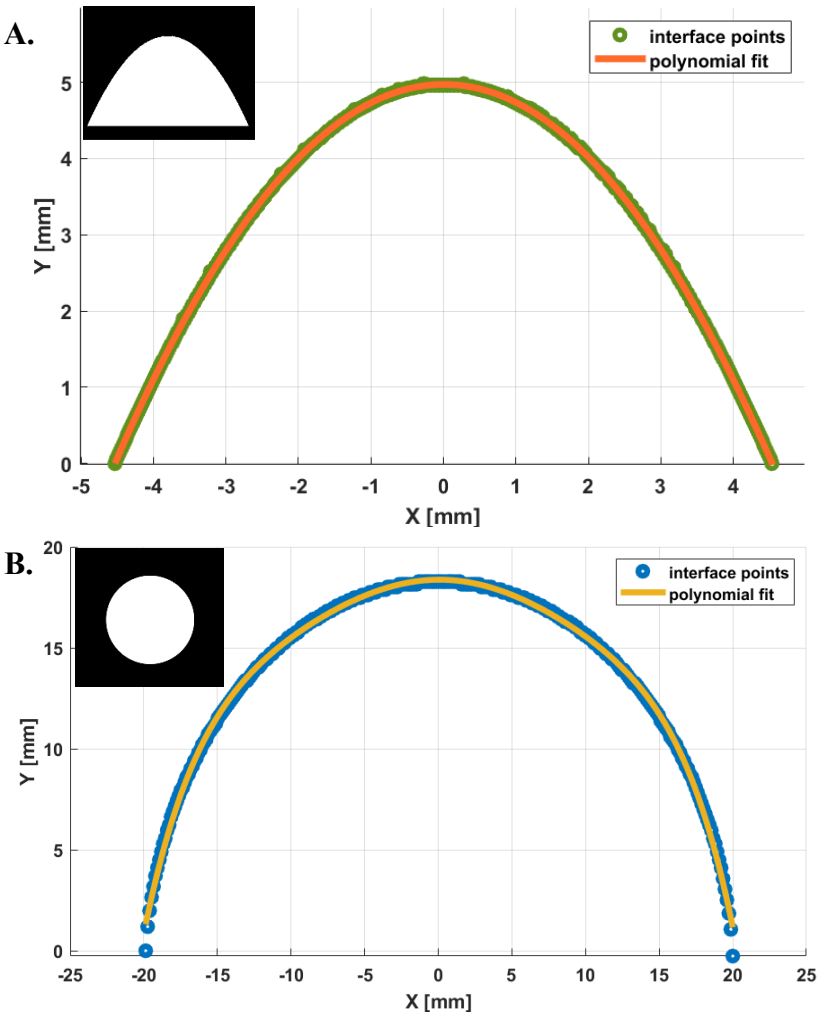

Fig. 3. Interface points obtained for images representing: **A.** a parabola of equation $y = -\frac{x^2}{4} + 5$, and **B.** a circle of radius R = 20 mm. Both interfaces were fitted with a 7th order polynomial. Insets: the images used for edge detection.

As expected, the interface obtained from the image of the parabola is well fitted and the points were retrieved and represented correctly. This can be seen when comparing the computed volume of the shape with the theoretical one, where the error between the two is of 3.7%.

In the case of the circle, as the code retrieves the coordinates of the first non-zero element of each column, a slight reduction in the height of the interface happens. The computed height of the interface is 18.37 mm, instead of the 20 mm of the radius, meaning an 8% reduction. This is due to the fact that at the middle section of the circle, the interface has more than one point placed at the same position on the X-axis. However, the code was designed to only find the point placed the highest on the Y-axis. On the other hand, no errors occurred in the radius computation. Also, as the interface was generated by the equation of a circle, a polynomial is not a perfect fit, as it does not account for the lowest values on the Y-axis. This further leads to the propagation of the error in the volume calculus, leading to a 12% decrease in the computed volume, compared to the theoretical one.

Table 1

Comparison between theoretical and computed values

Shape	Theoretical volume (mm^3)	Computed volume (mm^3)	Error (%)
Parabola	157.08	160.80	3.7
Circle	1.68×10^4	1.47×10^4	12

However, in reality, a fluid drop placed on a porous hydrophilic surface will always have an angle lower than 90°, and the shape of the interface will rather resemble a parabola, than a circle. Thus, for the analysis of the wicking phenomenon, it is considered that the measurement error will be under 5% for the volume. In the measurement of the diameter, an error would not occur in this case for sharp interfaces, but it is possible that it could appear if the interface is noisy or blurred.

3.2 Drop wicking analysis

The interface detection was tested for the drops of PEO wicking on the paper substrate (Fig. 4). The experimental points of the interface were plotted and fitted with a seventh order polynomial function. For each frame, the diameter, the volume and the contact angle of the drop were computed by the numerical code, as previously described. Their evolution was plotted over time and for each parameter, a fitting function was employed from the standard options provided in MATLAB.

Polynomial functions proved to be the best fit for the evolution of the diameter in time. It was observed that as the viscosity of the fluid increased, a higher order polynomial function was needed for a good fit. Therefore, for the drops of the 0.5% solution a second order polynomial was used, whereas for the drops of 5% PEO a fourth order polynomial provided the best fit (Fig. 5). Subsequently, the

spreading velocity was calculated as the absolute value of the first derivative evaluated for the fitting functions used.

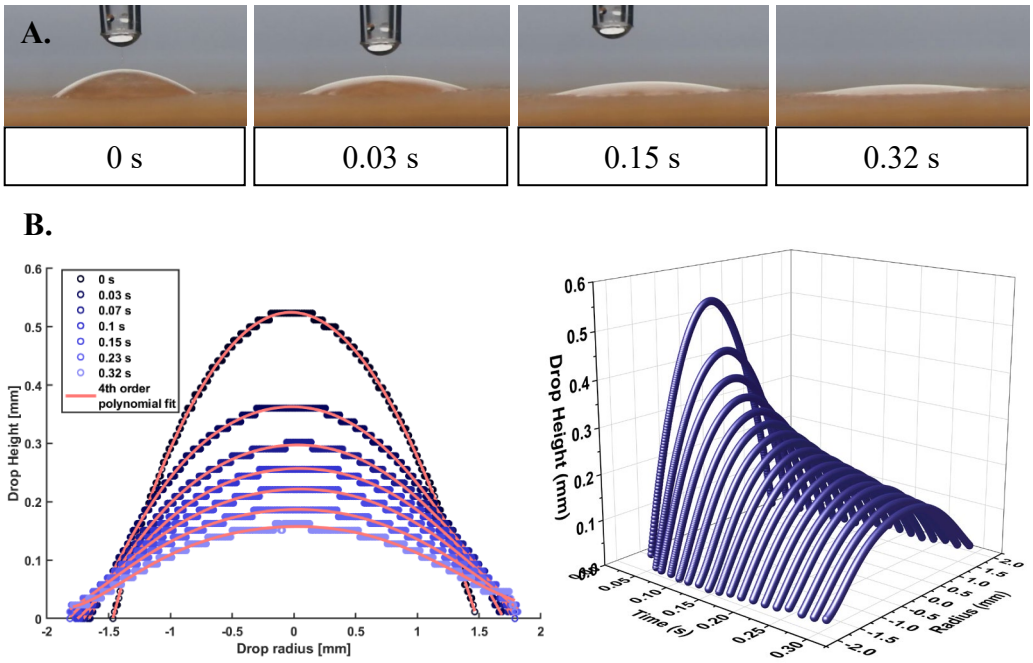

Fig. 4. Drop interface evolution over time. **A.** Frames extracted from the recording of a PEO 0.5% drop wicking on paper. **B.** Evolution over time of the detected interface by the numerical code, shown in 2D (computed points and polynomial fit) and 3D (only the function fit).

In the case of volume evolution, the best fit was found to be a smoothing spline function. The same smoothing parameter could be used for both solutions, and the optimal value was found to be 0.97423. Using the first derivative of the fitting functions, the flow rate of the drops into the porous substrate could be calculated (Fig. 6).

The contact angle of the drops was found to decrease exponentially for the drops of both solutions (Fig. 7). It was also concluded that the first 30 points of the interface were enough in order to accurately calculate this parameter.

All the analysed parameters had similar trends over time. This led to the conclusion that the viscosity of the fluid does not influence considerably the phenomenon, but it leads to a decrease in the velocity of the fluid front in the porous substrate.

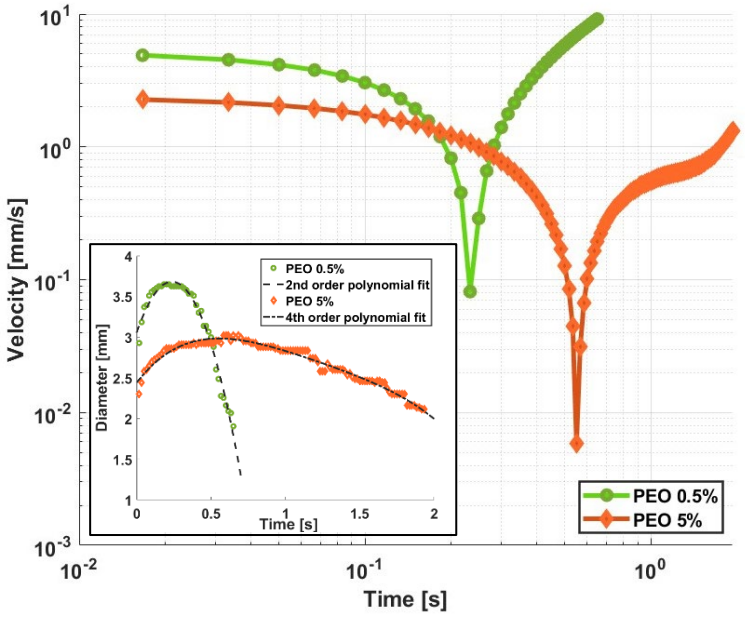

Fig. 5. Spreading velocity as a function of time, calculated as the first derivative of the diameter evolution over time. Inset: Drop diameter evolution over time, with the associated polynomial fittings.

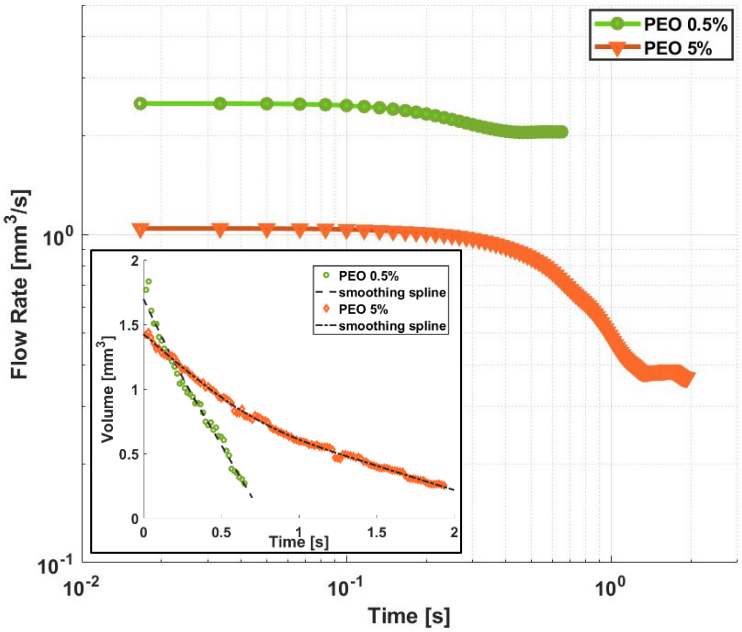

Fig. 6. Flow rate evolution over time derived from the evolution of drop volume. Inset: Drop volume as function of time, with the associated smoothing spline fittings.

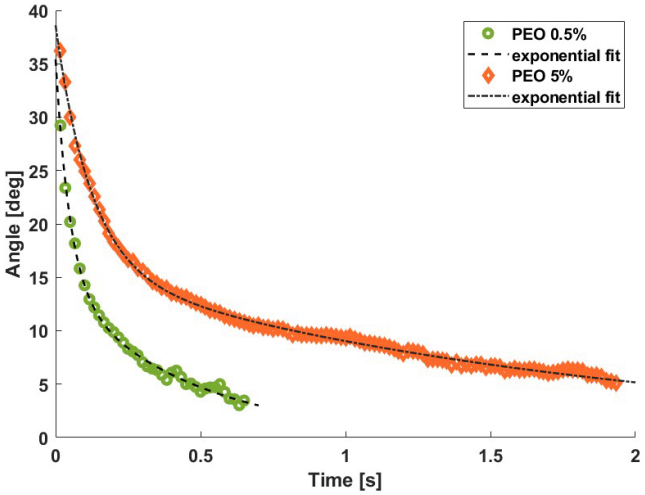

Fig. 7. Contact angle evolution over time, calculated as the arctangent of the first derivative evaluated for the linear fit of the first 30 interface points.

The execution time of the program was found to vary between the frames. The images of the initial stage of the wicking process employ a larger drop, with a higher number of interface points, so the first frames required approximately 1 second for running the code. As the interface decreases, the average running time per frame was found to decrease to 0.5 seconds. The wicking cases analysed had 50 frames for the 0.5% PEO solution and 120 frames for the 5% solution. This means that the wicking phenomenon of a single droplet can be evaluated in under 1 minute of running time.

4. Conclusions

In this work, a MATLAB image processing code was developed in order to simplify the analysis of wicking drops in porous media. The phenomenon was analyzed for solutions of poly(ethylene oxide) with two different concentrations. The program can accurately detect the interface of the drop and further evaluate the diameter, volume and contact angle. From the evolutions over time of these parameters, the spreading velocity and flow rate could be calculated. The study is still under development, but the results can be used to understand how the wicking process determines the final stain diameter. This is especially important in the paper-based biosensors domain, where the area of the fluid drop influences the electroactive surface. Moreover, as porous media is inhomogeneous, diffusion is a complex process, for which a general law has not been yet defined. The numerical code can be used to better understand how the material properties of fluids influence this phenomenon.

5. Acknowledgements

The authors acknowledge the financial support of CHIST-ERA – 19 – XAI – 009 MUCCA project, by the founding of EC and The Romania Executive Agency for Higher Education, Research, Development, and Innovation Funding - UEFISCDI, grant COFUND-CHIST-ERA MUCCA no. 206/2019.

The authors acknowledge the support from the Executive Agency for Higher Education, Research, Development and Innovation Funding (UEFISCDI), Romania, Project code: PN-III-P4-PCE-2021-1006, the Romanian Ministry of Research, Innovation and Digitalization through Core Program PN19-03 (contract no. 21 N/08.02.2019).

Appendix – MATLAB code

```
path = 'E:\..\*.jpg'; % the folder in which the images exist
jpgFiles = dir(path);
numFiles = length(jpgFiles);

% Preallocate space for every array needed
dropVolume = zeros(1,numFiles);
stainArea = zeros(1,numFiles);
diameter = zeros(1,numFiles);
time = zeros(1,numFiles);
spreadingVelocity = zeros(1,numFiles);
flowRate = zeros(1,numFiles);
contactAngle = zeros(1,numFiles);

for k = 1:numFiles    % loop through each img
   filename = strcat('E:\..\',jpgFiles(k).name);
   I = imread(filename);

    % Prepare the image for processing by converting to grayscale,
cropping to a smaller area of interest and flipping the image to be in
the right direction of the Y axis
    I = im2gray(I);
    I = imcrop(I,[396.8 725.6 784.4 109.7]);
    I = flip(I);

    % Binarize the image, remove the artifacts (innacurate small clusters
of pixels) and convert the logical matrix to double
    [BW, mask] = segmentImage (I);
    BW = bwareaopen (BW, 50);
    BW = double(BW);

    % Find the coordinates of the first and last white pixels
    [rowFirst, colFirst] = find(BW, 1, "first");
    [rowLast, colLast] = find(BW, 1, "last");
```

```
    % Prealocating space for the row and column arrays
    row = zeros(1, colLast);
    col = zeros(1, colLast);

    % The for loop must run only on the area where white pixels are
present to avoid errors
   for kk=colFirst:colLast
      % Find the last non-zero element from each column and put it in row
array
      [row(kk)] = find(BW(:,kk),1,"last");
      % Construct the column array by incrementing each iteration
      col(kk) = kk;
   end

    % Remove the 0 elements from arrays, which represent the black space
    col(col==0) = [];
    row(row==0) = [];

    % Rescale the arrays so that interface starts from (0;0). Rescale the
arrays from px to mm. Then center the interface peak on 0.
    col = col - colFirst;
    col = col .* (1/86);
    row = row .* (1/86);
    if k == 1
        rowInitial = row(1);
    end
    row = row - rowInitial;
    col = col - col(round(end/2));
    colLast = col(end);

    % Call the fitting function and apply it to the arrays col and row
    fittingFunction = 'poly7';
    [fitresult1, gof1] = createFit(col, row, fittingFunction);

    % Calculate the volume as the integral of fit from 0 to the last
column (implies rotation around Y axis). Fit stores the handle for
y=fitresult(x). Calculate the stain area also. Finally put the calculated
values in arrays and change the time from frames to seconds.
    fit1=@(col) feval(fitresult1,col);
    middleCol = colLast/2;
    vol_around_y = 2*pi*integral(@(col) fit1(col) .* col, 0, colLast,
'ArrayValued',true);
    stain_area = pi*colLast^2;

    dropVolume(k) = vol_around_y;
    stainArea(k) = stain_area;
    diameter(k) = colLast*2;
    time(k) = k * (1/60);
```

```
    % Find the first 30 points of the interface
      linearcontactPointsX = col(1:30);
      contactPointsY = fit1(linearcontactPointsX);
      contactPointsY = contactPointsY';
      [linearX,  linearY]  =  prepareCurveData(  linearcontactPointsX,
contactPointsY );
      fitLinearInterface = fit(linearX,linearY,'poly1');

     [dlinTheta1,~] = differentiate(fitLinearInterface,contactPointsX);
     contactAngleLin(k) = atan(mean(dlinTheta1));
     contactAngleLin(k) = contactAngleLin(k) .* (180/pi);

     C{k} = col;
     interfacePoints = fit1(col);
     R{k} = interfacePoints;
end

% find the longest vector
lA = max(cellfun(@(x) length(x), C));
lB = max(cellfun(@(x) length(x), R));
A = zeros(length(C), lA);
B = zeros(length(C), lB);
for i=1:length(C)
    A(i, 1:length(C{i})) = C{i};
    B(i, 1:length(C{i})) = R{i};
end

plot3(A,B,time,'o');

figure("Name","Contact Angle Linear vs Time")
    scatter(time,contactAngleLin);
    hold on
    [smoothTime,   smoothAngleLin]   =   prepareCurveData   (   time,
contactAngleLin );
    fitAngleLin = fit(smoothTime,smoothAngleLin,'exp2');
    plot(fitAngleLin);
    legend({'experimental data','exp2'},'Location','northeast')

%fitting the diameter and calculating the spreading velocity as the 1st
%derivative
[fitTime, fitDiameter] = prepareCurveData( time, diameter );
smoothDiameter = fit(fitTime,fitDiameter,'poly4');
[smooth_dD1,~] = differentiate(smoothDiameter,time);
smooth_dD1 = abs(smooth_dD1);

 figure("Name","Diameter vs Time")
    scatter(time,diameter);
    hold on
```

```
    plot(smoothDiameter);

figure("Name","Spreading velocity (function derivative) vs Time")
     plot(time,smooth_dD1);
     hold on
     scatter(time,smooth_dD1);

%fitting the volume
[fitTime, fitVolume] = prepareCurveData( time, dropVolume );
smoothVolume     =     fit(fitTime,     fitVolume,     'smoothingspline',
'SmoothingParam', 0.97423);
[smooth_dV1,~] = differentiate(smoothVolume,time);
smooth_dV1 = abs(smooth_dV1);

figure("Name","Drop Volume vs Time")
    scatter(time,dropVolume);
    hold on
    plot(smoothVolume);
    legend({'experimental   data',   'smooth   function'},   'Location',
'northeast')

figure("Name","Flow rate vs Time")
    scatter(time,smooth_dV1);
    hold on
    plot(time,smooth_dV1);

%fitting the stain area
[fitTime, fitArea] = prepareCurveData( time, stainArea );
smoothArea = fit(fitTime,fitArea,'poly4');

figure("Name","Stain Area vs Time")
    scatter(time,stainArea);
    hold on
    plot(smoothArea);
    hold on

function [fitresult, gof] = createFit(X, Y, fittingFunction)
[xData, yData] = prepareCurveData( X, Y );
ft = fittype( fittingFunction ); % Set up fittype and options.
[fitresult, gof] = fit( xData, yData, ft ); % Fit model to data.
end

function [BW,maskedImage] = segmentImage(X)
BW = X > 153; % Threshold image - manual threshold
maskedImage = X;
maskedImage(~BW) = 0;
end
```

REFERENCES

[1]. *B. Mullins, R. Braddock and I. Agranovski.* Fibre Wetting Processes in Wet Filtration, 2003.
[2]. *D. Raja, G. Ramakrishnan, V.R. Babu, M. Senthilkumar and M. Sampath.* Comparison of different methods to measure the transverse wicking behaviour of fabrics. Journal of Industrial Textiles. 2014;43(3):366-382.
[3]. *E. Kissa. Wetting and wicking.* Text Res J 1996; 66(10): 660–668.
[4]. *K. Ghali, B. Jones and J. Tracy.* Experimental techniques for measuring parameters describing wetting and wicking in fabrics. Text Res J 1994; 64(2): 106–111.
[5]. *K. Ghali, B. Jones and J. Tracy*. Modeling moisture transfer in fabric. Exp Therm Fluid Sci 1994; 9: 330–336.
[6]. *T. Gillespie.* The spreading of low vapour pressure liquids in paper. J Colloid Interf Sci 1958; 13: 32–32.
[7]. *T. Kawase, S. Sekoguchi, T. Fujii and M. Minagawa.* Spreading of liquids in textile assemblies: Part I: Capillary spreading of liquids. Text Res J 1986; 56: 409–414.
[8]. *M.J. Hertaeg, R.F. Tabor, J.D. Berry and G. Garnier.* Dynamics of stain growth from sessile droplets on paper. J Colloid Interface Sci. 2019 Apr 1; 541:312-321.
[9]. *M.J. Hertaeg, R. F. Tabor and G. Garnier.* Effect of Protein Adsorption on the Radial Wicking of Blood Droplets in Paper. 2018. Journal of Colloid and Interface Science. 528: 116-123.
[10]. *D. Botta, I. Magos and C. Balan, Influence of Viscosity on Radial Diffusion of Fluids in Paper Substrates. 2021. 1-5. 10.1109/ATEE52255.2021.9425224.*
[11]. *D. Botta, I. Magos and C. Balan, "The Influence of Surface Tension on Radial Wicking in Paper," 2021 10th International Conference on ENERGY and ENVIRONMENT (CIEM), 2021, pp. 1-5*
[12]. *M. Hoorfar, M.A. Kurz, A.W. Neumann,* Evaluation of the surface tension measurement of axisymmetric drop shape analysis (ADSA) using a shape parameter, Colloids and Surfaces A: Physicochemical and Engineering Aspects. 2005. 260: 1–3, 277-285.
[13]. *Y.Y. Zuo, M. Ding, A. Bateni, M. Hoorfar*, *A.W. Neumann*, Colloids Surf. A: Physicochem. Eng. 250, 2004, 233
[14]. *M. Hoorfar, A. Neumann.* Axisymmetric Drop Shape Analysis (ADSA) for the determination of surface tension and contact angle. Journal of Adhesion. 2004. 80. 727-743.