Rapport de Stage

Student : **Gregorio Pérez Bernal**
Tutor : **Alejandro Ospina Vargas & Juan Manuel Rodriguez Prieto**
Date : **September - December 2023**

A hybrid model for electrospinning based on the Finite Element Method

Contact Author : **gperezb1@eafit.edu.co**
Contact Tutor : **alejandro.ospina@utc.fr**

Abstract

Electrospinning is a versatile technique used to produce nanofibers out of polymers, with applications in various fields such as tissue engineering, filtration, and drug delivery. This report presents a numerical model based on the finite element method to simulate the electrospinning process. The model considers the coupled electrostatic, fluid flow, solid deformation, and fiber stretching phenomena that govern the process.

The electrostatic field is solved using Poisson's equation, aided by Maxwell's tensor to calculate electrostatic forces. Two techniques were employed to model the polymer, one taking it as a highly elastic solid, for which Maxwell's viscoelastic model was used. As well as that, a second approach was taken, modeling the polymer as a fluid, solving for a two-phase using the level set method.

The developed model is implemented in finite element software called FeniCs, which is an open source, Python package, and validated against experimental data from the literature. The results show some agreement with the experimental tendencies. The model is then used to study the effect of process parameters such as applied voltage, flow rate, and polymer properties on the morphology of the formation of a Taylor Cone, a crucial geometry when modeling electrospinning.

Contents

1. Introduction

The study of nanofibers has gained popularity over previous years, due to the different and unique properties that they show compared to materials with a larger diameter, as well as the distinct applications that said fibers have, ranging from tissue engineering to 3D printing and even to fabrication of electronic and optical devices.(1).

The process of electrospinning has been widely explored over the past decades, as a groundbreaking technique used to produce fibers with small diameters. As described by (2), the fibers that are produced through electrospinning span almost 4 orders of magnitude in diameters, ranging from tens of micrometers down to only a few nanometers, and, due to the nature of their fabrication, are essentially unlimited in length. It is crucial to understand and tune the different parameters that affect electrospinning, to obtain a fiber in the order of magnitude desired.

Electrospinning proves to have a core difference from other techniques used to produce fibers, the fact that, whereas different methods tend to use mechanical artifices to form nanofibers, electrospinning takes advantage of the electric field-induced changes in the geometry of liquid polymers to elongate a material (tends to be a polymer) until obtaining the desired fiber.

The main idea behind electrospinning is to take a polymer in its liquid state and place it inside a needle. When high electric tension is applied to the metallic syringe, the polymer will become polarized, and movement will be induced on it, so it will leave the needle. Upon leaving it, a small fiber will be generated and fall into a collector that electrically acts as ground.

The importance of simulation of electrospinning plays a major role in offering a cost-effective and time-efficient method for predicting nanofiber sizes. In simulating the process, predictions of the fiber's diameter can be obtained, giving a range of values for the diameter, in order to reduce experimentation. Note that, as reported by (3), in electrospinning, the flow rate is around $0.01 cm^3/s$, which can be rather slow, making it reasonable to search for alternatives such as simulation. As well as that, it seems obvious that simulating the process will aid in reducing costs for research, as less physical experimentation would be needed.

2. Context

2.1 Description of the process

Figure **2.1** presents a schematic drawing for the overview behind the assembly of an electrospinning device. The idea behind the process goes as follows: In the laboratory, the test bench is composed of a metallic syringe that contains a liquid phase polymer (which tends to contain a volatile solvent, that will evaporate during the process) and a collector, which is often a cylinder.

The needle and the collector are connected to a high-voltage device, which can vary from a few kilovolts and go up to more than 50 kV (4). When voltage is applied, charge accumulates on the surface of the fluid, due to polarization of the fluid. Posterior repulsion among the charges and effects from external forces such as surface tension lead to the expulsion of the polymer from the needle, and a jet ensues.

Upon leaving the capillary, the electrified polymer, will form a cone-like interphase, known as the Taylor Cone, which will be explained shortly. The jet will move towards the collector, at first in a straight manner, but due to electrostatic interactions, the jet will go through bending perturbations, and will take on a whipping-like motion.

The jet will solidify, making way for the formation of the desired nanofibers, which will continue to go toward the collector, where it will be received. The shape of the collector will vary depending on how the newly formed fibers should be arranged.

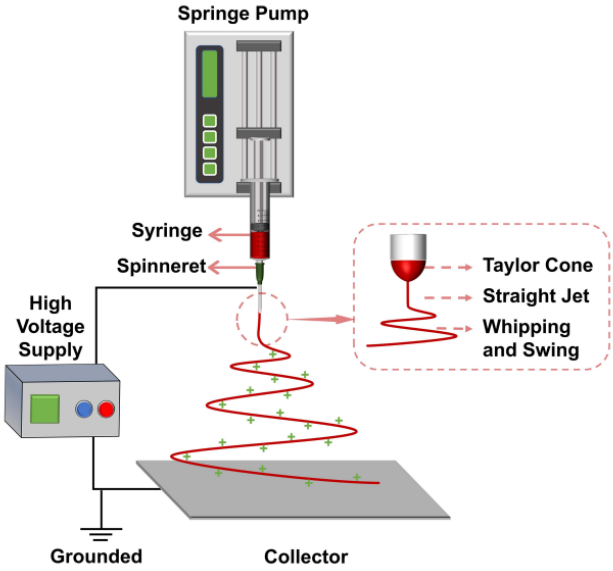


Figure 2.1: Electrospinning Setup. *taken from (5)*

2.2 Key variables for electrospinning

Electrospun fibers can exist in up to four different orders of magnitude, which poses the question of how the process can be tuned so that the desired fiber is obtained. Several key control variables can and should be modified by the user that affect the formation of fibers.

The main properties inherent in the material to be electrospun include the electrical conductivity (K), dielectric constant (ϵ), surface tension coefficient between the air and the polymer (γ), as well as the viscoelastic properties of the polymer, such as viscosity constant (μ) and relaxation time (τ) (1). The density of the polymer (ρ) also plays a role in how fibers move during electrospinning.

The setup for the process must also be calibrated, as several parameters may alter the end radius of the electrospun fiber. Some of them include the infusion rate (Q), the applied voltage (V) which can be either held constant or increased over time, and the distance between the needle and the collector (d).

It is noteworthy that not every mathematical model includes every parameter, and that one of the reasons why every simulation is unique comes from the fact that several values are omitted whilst making assumptions. Models found in the literature that describe the process of electrospinning can include different combinations of parameters, which lead to different results.

Figure **2.2** shows several key relationships that the parameters have between

one another and in relation to the radius of the created fiber.

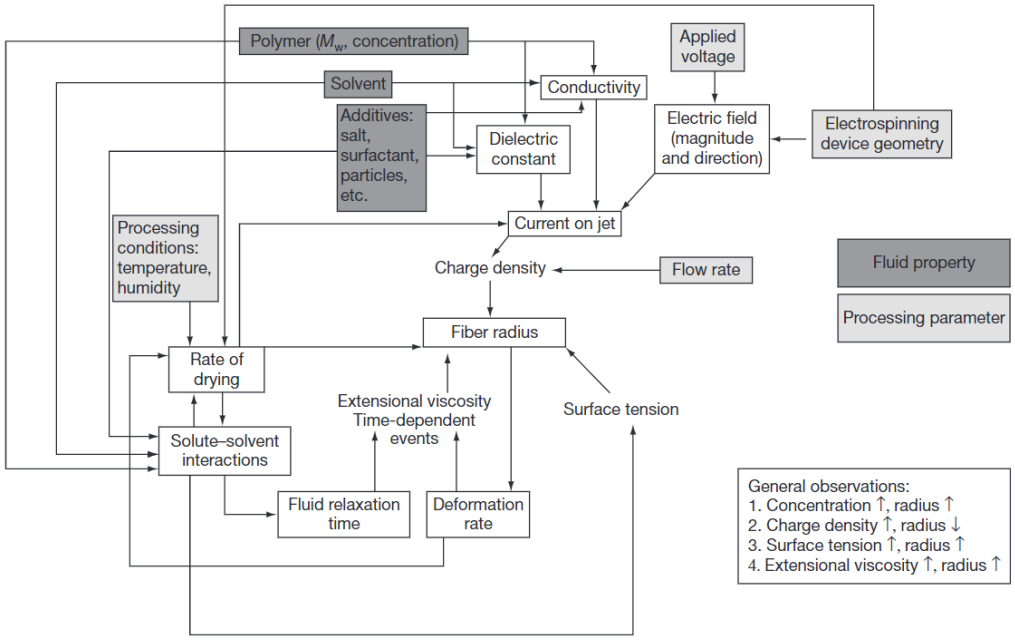


Figure 2.2: Flow chart representing how the key variables of electrospinning affect the resulting fiber. *taken from (2)*

2.3 Stages of electrospinning

As shown in figure **2.1**, there are three main stages during the process of electrospinning, which are, the formation of a Taylor Cone, the expulsion of a Straight Jet and the Whipping and Swing, also o known as whipping instability.

2.3.1 Taylor Cone Formation

Upon leaving the needle, the polymer experiments two main forces, electrostatic contributions and effects due to surface tension. As the electrostatic interactions begin to overcome the surface tension charges, a conical interface begins to form, known as the Taylor Cone. Figure **2.3** clearly represents the different stages a polymer undergoes while forming a Taylor cone.

As elaborated by (6), the formation of a Taylor cone is crucial for the process of electrospinning, and understanding how it is created may help deduce some properties of the electrospun fiber, such as an approximate radius. It was first thought that the critical half angle of the meniscus of a Taylor cone had to be

around **49.3º**, however, after posterior research by (6), it was found that the meniscus during electrospinning can vary a lot in semi angle, as wide varieties of cones have been found.

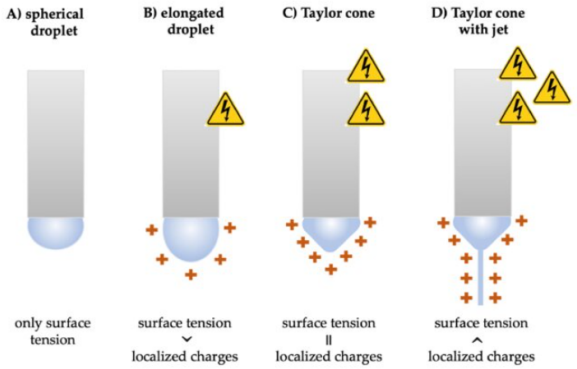


Figure 2.3: Stages of the formation of a Taylor cone. *taken from (7)*

As explored by (2), studying the formation of a Taylor Cone proves to be a very difficult endeavor. This is because there are a lot of variables, that have proven to be sensible under small changes. There has been both qualitative and quantitative insight into how the different properties affect the formation of the conical shape on the meniscus, however, due to the nature of the problem, different values will greatly affect the shape of the cone. Figure **2.4** presents an example of this, showing how the same polymer under different concentrations forms various shapes.

Figure 2.4: Various shapes of the meniscus observed for the same polymer under different concentrations. *taken from (2)*

2.3.2 Straight Jet and Whipping Instability

As observed in figure **2.1**, a straight jet is ejected from the tip of the Taylor cone. It is suggested that the shape of the thinning jet depends significantly on the evolution of the surface charge density and the local electric field, which will continue to increase as the fiber approaches the collector, due to the inverse square

law. (2). This essentially means that as the jet evolves, the radius of the fiber will slowly decrease, until some instabilities arise.

The time that the straight jet takes might not be enough to properly solidify the polymer, which succumbs to the effect of force imbalances that lead to instability. The whipping motion, which can be observed in figure **2.1**, is a result of electrostatic forces acting on the charged liquid jet. When a high voltage is applied to the polymer solution, it becomes charged. As the charged jet moves towards the collector, these forces cause the jet to undergo a whipping or bending instability.

2.4 Applications of electrospinning

Electrospinning has found diverse applications across various industries due to the unique properties of the nanofibers it produces. In biomedical engineering, electrospun fibers are employed for drug delivery systems, wound healing scaffolds, and tissue engineering constructs, capitalizing on their high surface area and porous structure (8; 9). Regenerative tissue building with electrospun fibers has also been studied and explored, due to the wide range of materials that can be used.

In filtering technology, nanofibrous membranes contribute to efficient air and water purification processes (10). As stated by (4), it has been proven that electrospun fibers from certain polymers, under certain conditions, can have diameters small enough to be used as nanofiltration membranes.

The textile industry benefits from nanofiber-based fabrics with enhanced properties such as breathability and moisture management (11). Apart from this, it electrospun fibers can also be used to create protective clothing from chemical warfare agents and even to form composite materials reinforcement in military and construction. (4).

These applications underscore the versatility of electrospinning technology in addressing a spectrum of challenges and advancing innovations across various scientific and industrial domains. Precisely for this reason, it is important to attempt to grasp the main concepts that cause electrospun fibers to behave as they do.

3. Objectives

3.1 General Objective

Develop an algorithm based on the finite element method, backed by a mathematical model built with partial differential equations, capable of simulating the formation of a Taylor Cone, to simulate the process of electrospinning.

3.2 Specific Objectives

1. Formulate a mathematical model based on the Navier Stokes and Maxwell equations, able to describe the process of electrospinning, using partial differential equations.

2. Develop an appropriate distribution for how the nodes interact, such as when the problem is discretized and can be further solved via the finite element method.

3. Track the two-phase flow between the polymer-air interface, using an appropriate mathematical tool.

4. Construct a model, under the assumption that the electrospun fiber will act as a deformed, viscoelastic solid, with a highly low Young Modulus, as opposed from a highly viscous fluid.

5. Compare and contrast the results obtained with the fluid model and the solid assumption.

4. Physical and Mathematical background

This chapter provides technical descriptions for different physical and mathematical tools that will be used during the creation of the mathematical models in question.

4.1 The Finite Element Method (FEM)

The Finite Element Method is a rather complex subject that can take several thousand pages to correctly explain. However, a brief description of how a problem is solved using FEM is going to be presented. For further information, refer to (12).

4.1.1 Key concepts

Take the following definitions:

1. C_0^∞: Functions defined from $\Omega \subseteq \mathbb{R}^n \implies \mathbb{R}$, that are infinitely derivable (C^∞), whose support in Ω is compact (closed and bounded). This functions are referred to as *Test Functions.* (12).

2. $L^2(\Omega)$: The space of functions that consists of measurable functions (in the Lebesgue sense), whose square is integrable over a domain Ω. This means that if $f \in L^2(\Omega)$, $\int_\Omega f^2 d\Omega < \infty$. (13).

3. $H^1(\Omega)$: This function space, also known as a Sobolev space, consists of the set of functions that belong to L^2, such that its derivative also belongs to L^2. Mathematically, $H^1(\Omega) = \{f \in L^2(\Omega) : \nabla f \in L^2(\Omega)\}$(13).

4. $H_0^1(\Omega)$: This functional space has an analogous definition to H^1, however, if $f \in H_0^1$, this means that f must also be annulled on the boundary of Ω.(13).

5. Weak derivative: Let D be an open set in $\mathbb{R}^d$, Let $u, v \in L^1(D)$ (12). Let $i \in \{1 : d\}$. We say that v is the weak derivative of u in the direction of i if:

$$\int_D u\partial_i \varphi dx = -\int_D \nu\phi dx \quad \forall \varphi \in C_0^\infty \tag{4.1}$$

6. Inner product: Let f, g be functions defined over Ω. The inner product of f and g is defined as $(f, g) = \int_\Omega f g d\Omega$.

4.1.2 Toy problem solution

In order to grasp the key concepts of the finite element method, the following toy problem will be solved. Consider the following model: Find $u \in L^2([0, 1])$, such that:

$$\begin{cases} \frac{d^2u}{dx^2} = f(x) \\ u(0) = 0 = u(1) \end{cases} \tag{4.2}$$

Where $f \in L^2([0, 1])$ is a known function.

The first step towards solving any FEM problem is to find the variational (weak) form of the problem. Basically, this means to use test functions v, which live on a chosen functional space, and use them in order to transform **4.2** into a system of equations, using integrals over the domain Ω. Mathematically speaking, the following steps are done:

1. Multiply by test functions v and integrate over the domain. This would give:

$$-\int_0^1 u_{xx} v(x) dx = \int_0^1 f(x) v(x) \tag{4.3}$$

2. Integrate by parts. This is done to reduce the order of the derivative of u by adding derivatives into v. The process is as follows:

$$-\int_0^1 u_{xx} v dx = \int_0^1 u_x v_x - [u_x v|_0^1] \tag{4.4}$$

$$-\int_0^1 u_x v_x dx - [u_x v|_0^1] = \int_0^1 f(x) v(x) dx \tag{4.5}$$

Evaluating the boundary conditions, we get:

$$\int_0^1 u_x v_x dx = \int_0^1 f(x) v(x) dx \tag{4.6}$$

Equation **4.6** is known as the variational form of equation **4.2**. Note that a first derivative is needed to exist for v, which is why v must be in H_0^1. This space may

change as a function of the largest ordered derivative for the test function. There are many different functions that can act as test functions. The more used are *Lagrange Polynomials* which are usually denoted as $\mathbb{P}_n$. Lagrnge order 1 polynomials refer to linear hat functions, that go through the domain.

Next, the domain must be discretized into n elements, in order to create the space $V_h := \{u \in C(\Omega) : u|_{[x_j, x_{j+1}] \in \mathbb{P}_1}\}$. This means that for every element, there is an associated test function. Figure **4.1** illustrates the discretization a domain Ω and Lagrange $\mathbb{P}_1$ polynomials, denoted as φ.

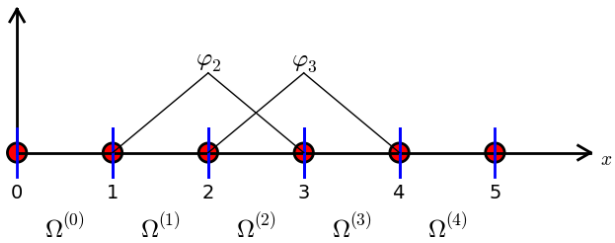


Figure 4.1: Discretization of the domain and generation of test functions. *taken from (14)*

This, inherently means that a system of equations will be created, one for each piecewise element. The approximate solution can be written as the following linear combination:

$$u_h(x) = \sum_{j=0}^{n} \alpha_j v_j \quad v_j \in \mathbb{P}_1 \tag{4.7}$$

Taking this value and substituting it into 4.6, the following expression is created, where every function is known, except from α.

$$\int_0^1 \frac{d}{dx}\left[\sum_{j=0}^{n} \alpha_j v_j\right] \cdot \frac{dv_i}{dx} dx = \int_0^1 f(x) \cdot v_i dx \tag{4.8}$$

Note that from equation **4.8**, a linear system of equations can be formed, in order to obtain a vector of αs. Obtaining the values for said vector is the ultimate goal of the finite element method. Fortunately, many programs available are able to solve a FEM problem, given the test functions and the variational forms. This is precisely why the approaches for solving the system of equations are not included here.

4.2 Navier Stokes Equations

In the way in which Newton's laws describe the movement of solids, the Navier Stokes equations are a set of partial differential equations that model the movement of every fluid, be it a liquid or a gas. For a general case, this equations do not have a known analytical solution. This is precisely why numerical methods are employed to grasp an approximate solution for the PDE.

The general case of the incompressible Navier Stokes equations read as follows:

$$\begin{cases} \nabla \cdot \vec{u} = 0 \\ \rho \frac{D\vec{u}}{Dt} = -\nabla P + \rho \vec{g} + \mu \nabla^2 \vec{u} + \vec{F} \end{cases} \tag{4.9}$$

Where $\vec{u}$ refers to the velocity vector field and $\vec{P}$ to the pressure field, which are the two fields to be solved for. μ and ρ refer to the viscosity and the density of the fluid and $\vec{g}$ is the acceleration due to gravity. $\vec{F}$ refers to the external forces that act upon the fluid, and will be very important to model electrospinning, as the electrostatic forces will be included in the Navier Stokes equations through this variable.

The first equation is known as the mass equation, and it is accountable for the conservation of mass during flow, as well as the incompressibility, whereas the second equation is known as the momentum equation and it accounts for the conservation of momentum. It acts just like Newton's second law acts on solids.

4.2.1 Solving the Navier Stokes equations using FEM

In order to simulate the Navier Stokes equations, the problem must be reduced, in order to get to a variational form, as described by (15). Chorin's scheme proposes to divide the solution in three main points. First, to compute a tentative velocity by neglecting the pressure in the momentum equation and then to project the velocity onto the space of divergence free vector fields.

Mathematically speaking, this means that Chorin's scheme first solves the following momentum equation:

$$\frac{\mathbf{u}^* - \mathbf{u}^h}{\Delta t} = -\left(\mathbf{u}^h \cdot \nabla\right) \mathbf{u}^h + \mu \nabla^2 \mathbf{u}^h \tag{4.10}$$

Where u^h refers to the velocity in an h^{th} time and u^* acts as temporary velocity. Note that equation **4.10** provides a discretization for the temporal derivative presented in equation **4.9**. The second half of the algorithm is in charge of taking the pressure in order to correct u^* by solving equation **4.11**.

$$\frac{\mathbf{u}^{h+1} - \mathbf{u}^*}{\Delta t} = -\frac{1}{\rho} \nabla P^{h+1} \tag{4.11}$$

With this scheme, it must be clear that the algorithm is really just an operator-splitting approach in which one considers the different contributions in different steps. Figure **4.2** shows the algorithm proposed by (15) that must be followed to solve the Navier Stokes equations via Chorin's protection. Note that v, q refer to test functions that must be in appropriate function spaces. (15) recommends using **Taylor-Hood** elements, which in nature correspond to using the second order Lagrange Polynomials (P^2) to solve for velocity and first order Lagrange Polynomials (P^1) to solve for pressure.

Scheme 1: Chorin's projection method

1. Compute a tentative velocity $u_h^\star$ by solving

$$\langle D_t^n u_h^\star, v\rangle + \langle \nabla u_h^{n-1}\, u_h^{n-1}, v\rangle + \langle \nu \nabla u_h^\star, \nabla v\rangle = \langle f^n, v\rangle \quad \forall\, v \in V_h,$$

   including any boundary conditions for the velocity.

2. Compute the corrected pressure p_h^n by solving

$$\langle \nabla p_h^n, \nabla q\rangle = -\langle \nabla \cdot u_h^\star, q\rangle / k_n \quad \forall\, q \in Q_h,$$

   including any boundary conditions for the pressure.

3. Compute the corrected velocity u_h^n by solving

$$\langle u_h^n, v\rangle = \langle u_h^\star, v\rangle - k_n \langle \nabla p_h^n, v\rangle \quad \forall\, v \in V_h,$$

   including any boundary conditions for the velocity.

Figure 4.2: Algorithm for the Chorin protection scheme. *taken from (15)*

4.3 Maxwell's viscoelastic model

A viscoelastic material is one that exhibits both elastic and viscous properties. This inherently means that, when subject to stress, a viscoelastic material shows some recoverable deformations (elastic behavior) and some non-recoverable deformations (viscous behavior). This means that viscoelastic materials have to be modeled with temporal dependence. the rate and magnitude of deformation depend not only on the applied stress but also on the duration of the stress application. Examples of viscoelastic materials include certain polymers, biological tissues, and certain types of fluids like some types of oils and lubricants.

There are several models used to model viscoelastic behavior. One of the pioneers on the subject is Maxwell's model, which models a material as a configuration of a spring-damper system. The spring in the model would act as the elastic portion, which has a reversible effect, and the damper would act as the viscous portion, presenting unrecoverable effects.

Figure **4.3** shows the configuration for Maxwell's model, placing the damper and the spring on a series configuration. The parameters of E and η would be

analogous to the elastic and viscous constants. It can also be seen the response that the system presents under a time-dependent force, noting that there are non-recoverable deformations when the force stops acting upon the model.

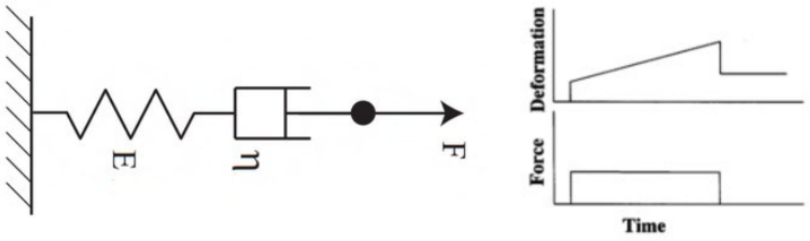


Figure 4.3: Overview for Maxwell's viscoelastic model configuration and response to a stress. *taken from (16)*

The equation that models the viscoelastic behavior of a material is analogous to a spring-damper equation. Let E, η be the material's Young's modulus and viscous coefficient. The deformations ϵ dependent on stress applied (σ) can be expressed as:

$$\sigma + \frac{\eta}{E}\dot{\sigma} = \eta\dot{\epsilon} \tag{4.12}$$

Note, by figure **4.3** and by equation **4.12**, that the act of viscous deformation only occurs upon relaxation, as it is the role of the damper to account for those effects.

4.4 Methods used to track two-phase flow

When attempting to model the process of electrospinning, a key factor is to comprehend that the phenomenon is occurring between interfaces. This is because as the polymer moves out of the needle, the polymer-air interface begins to move. Upon consideration that air and the electrospun polymer have different properties, a mathematical framework must be established in order to correctly model the movement of interfaces, which inherently will describe the position of the polymer, as opposed to the Navier Stokes equations, which model velocities. Two methodologies are described, the Level Set Methods and Cahn Hilliard's equation for phase separation.

4.4.1 Level Set methods

Take a domain $\Omega \subset \mathbb{R}^n$ and Ω_1, Ω_2 to be disjoint subsets of *Omega*. Let us define *Sigma* as the internal boundary of Ω_1 and Ω_2. The main idea behind Level Set formulations is to define a signed distance function ϕ, such that the sign of ϕ changes as a function of Ω, inherently tracking the position of the interphase. Figure **4.4** shows the idea behind a level set formulation.

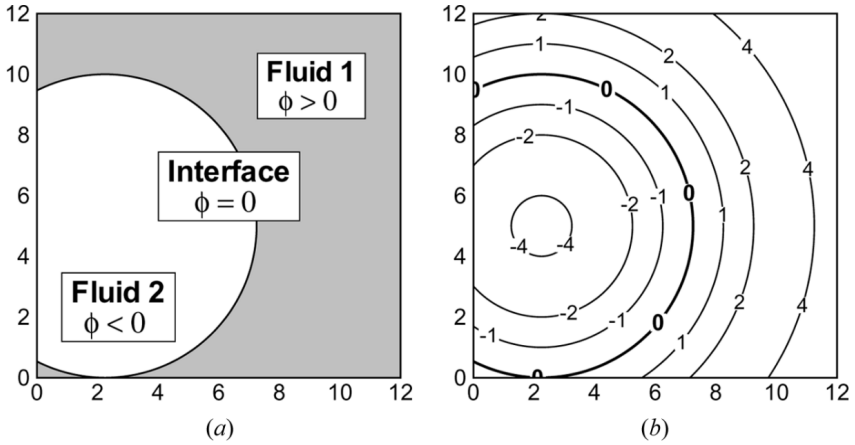


Figure 4.4: Interphase behind a Level Set Formulation. *taken from (17)*

Note that an important property of ϕ is that $\Gamma = \{x : \phi(x) = 0\}$.

Some classic signed functions used to initialize a level set formulation are the following (with $x \in \mathbb{R}^n$):

$$\phi(x) = \frac{2}{1 + e^{\frac{x}{\epsilon}}} - 1 \tag{4.13}$$

$$\phi(x) = \tanh\left(\frac{x}{\epsilon}\right) \tag{4.14}$$

$$\phi(\mathbf{x}) = \begin{cases} \min_{\mathbf{x}_i \in \Gamma} \|\mathbf{x} - \mathbf{x}_i\|, & \mathbf{x} \in \Omega_1 \\ -\min_{\mathbf{x}_i \in \Gamma} \|\mathbf{x} - \mathbf{x}_i\|, & \mathbf{x} \in \Omega_2 \end{cases} \tag{4.15}$$

Level set Advection and reinitialization

Consider a fluid that must move through an interface. Consider a velocity flow field $u \in \mathbb{R}^n$. Tracking the interface is a matter of solving the following transport equation (18):

$$\phi_t + u \cdot \nabla\phi = 0 \tag{4.16}$$

Note that ϕ must change its value after every iteration, the field ϕ changes its value, which means that ϕ will not retain the property of being a signed function. This is a drawback, so it is recommended to consider a reinitialization, so ϕ retains its signed properties.

In order to reinitialize, the following differential equation must be solved:

$$\phi_\tau(\mathbf{x},\tau) + \nabla \cdot [\phi(\mathbf{x},\tau)(1-\phi(\mathbf{x},\tau))\hat{n}] = \varepsilon \nabla \cdot [\hat{n}(\nabla\phi(\mathbf{x},\tau)\cdot\hat{n})] \tag{4.17}$$

Where $\hat{n}$ defines the unit normal to the interface, defined as:

$$\hat{n} = \hat{n}(\mathbf{x}) = \frac{\nabla\phi(\mathbf{x},\tau=0)}{|\nabla\phi(\mathbf{x},\tau=0)|} \tag{4.18}$$

In equation **4.17** τ refers to a different time as t. Given a $\phi_0 = \phi(x,t_n)$, an iteration must be done to equation **4.17** until a steady state with respect to τ is reached.

It is clear the the reinitialization equation should be solved under homogeneous Neumann boundary conditions, saying that there is no net transport of mass into or from the system during the reinitialization (18).

FEM for the level set method

The variational form of equation **4.16** is (18):

$$(\phi_t + \nabla\cdot(\phi u), w) = 0 \tag{4.19}$$

and using the divergence theorem, can be expressed as:

$$(\phi_t, w) = (\phi, \nabla w \cdot u) \tag{4.20}$$

In order to solve the equation, a temporal discretization must be done, to remove the time derivative. Applying a Crank-Nicholson discretization of equation **4.20**, the following equation is obtained:

$$\frac{1}{\Delta t}\left(\phi^{n+1}-\phi^n, w\right) = \frac{1}{2}\left(\phi^{n+1}+\phi^n, \nabla w\cdot\mathbf{u}\right). \tag{4.21}$$

Rewriting terms, the following bilinear form is obtained:

$$\left(\phi^{n+1}, w\right) - \frac{\Delta t}{2}\left(\phi^{n+1}, \nabla w\cdot\mathbf{u}\right) = (\phi^n, w) + \frac{\Delta t}{2}\left(\phi^n, \nabla w\cdot\mathbf{u}\right) \tag{4.22}$$

In order to fatherly stabilize the solution given, the dominant method is the Streamline Upwind Petrov-Galerkin (SUPG) (19). Let r be the residual of the advection equation, written as

$$r = \phi_t + u\cdot\nabla\phi \tag{4.23}$$

and let the parameter

$$s = \frac{h}{2||u||} u \cdot \nabla w \tag{4.24}$$

where h refers to the local element size. The stabilized weak form of the Level Set equation can be written as:

$$(\phi_t, w) + (\nabla \cdot (\phi \mathbf{u}), w) + (s, r) = 0 \tag{4.25}$$

Note that this equation can be solved using FEM after a temporal discretization, solving for ϕ^{n+1}.

4.4.2 Cahn - Hilliard's equation

As described by (20), Cahn Hilliard's equation is born as a tool used to provide the evolution of phase separation in materials, capturing how distinct phases gradually form and organize over time. As opposed to the level set method, Cahn Hilliard's equation is described in terms of energies and diffusive properties, modeling only phase separation. Figure **4.5** demonstrates the way in which two phases separate due to diffusion over time.

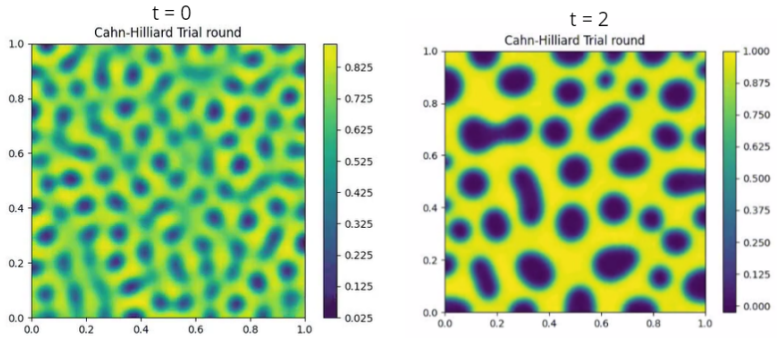


Figure 4.5: Implementation of Cahn-Hilliard's equation on two different times.

The derivation done by (20) follows the following structure. Surface motion can be derived as the dissipation of a phase-field's energy functional, which can be written as:

$$\phi_t = \frac{\delta E_\epsilon}{\delta \phi} \tag{4.26}$$

Where the operator δ refers to the functional derivative. Taking ϵ to be a parameter that defines the width of the interphase. There are several ways to

describe the energy E_ϵ. A simple definition of an energy functional, used by (20) and (15) reads as follows:

$$E_\epsilon = \int_\Omega \left(\frac{1}{2} |\nabla \phi|^2 + \frac{1}{\epsilon^2} F(\phi) \right) \tag{4.27}$$

F is called a double-well potential, and represents the tendency of a system to have two stable phases. F tends to be a $4th$ order polynomial. Under most cases, it is written as:

$$F(\phi) = \frac{1}{4} \left(\phi^2 - 1 \right)^2 \longrightarrow f(\phi) = F'(\phi) = \left(\phi^2 - 1 \right) \phi \tag{4.28}$$

Let γ be a relaxation constant. Noting the different definitions given, Cahn Hilliard's equation is expressed as:

$$\phi_t = \gamma \nabla^2 \left(-\nabla^2 \phi + \frac{1}{\epsilon^2} f(\phi) \right) \tag{4.29}$$

Note from equation **4.29** that the Cahn-Hilliard equation models phase separation due to a concentration gradient and does not include a velocity. This means that the procedure to couple the Navier-Stokes equations with equation **4.29** is not as trivial as it is when using the level set method.

As well as that, consider that due to the appearance of two Laplacians, the Cahn-Hilliard equation is a fourth-order differential equation, which is challenging to solve, both analytically and numerically. As a result of this, most numerical solvers introduce the auxiliary variable of the chemical potential $w = -\nabla^2 \phi + \frac{1}{\epsilon^2} f(\phi)$, to substitute on equation **4.29** to get the following system of equations:

$$\begin{cases} \phi_t = \gamma \nabla^2 w \\ w = -\nabla^2 \phi + \frac{1}{\epsilon^2} f(\phi) \end{cases} \tag{4.30}$$

Note that equation **4.30** is the equation that most solvers deal with. The derivation for the weak form can be found on (20).

5. State of the art

Electrospinning has been widely modeled, using different mathematical and physical tools. One of the most common ways to separate how the phenomenon is modeled is by describing the models based on Leaky Dielectric Models, and the models based on tracking interfaces.

5.1 Leaky Dielectric Model

In the realm of electrohydrodynamics, the Leaky Dielectric model stands as a pivotal concept, shedding light on the behavior of insulating materials crucial in the design and analysis of electrical systems. Unlike ideal insulators, dielectric materials in this model exhibit a certain level of electrical conductivity, allowing a subtle current flow when subjected to an applied voltage (21).

The Leaky Dielectric model was purposed originally by Taylor and Melcher (21) and encompasses several key variables essential for understanding the behavior of insulating materials within electrical systems. Central to this model is the dielectric constant, representing the material's ability to store electrical energy when subjected to an electric field. Additionally, the conductivity of the dielectric material itself plays a crucial role, dictating the extent to which current can flow through the insulator. Furthermore, factors such as temperature and frequency can significantly impact the conductivity and dielectric properties, and are often included on leaky dielectric models.

One of the most acclaimed models done using the Leaky dielectric model is the one done by Reznik et al (3). Reznik's model takes the polymer as a perfect conductor, and using the Stokes (not the Navier Stokes) equations for fluid flow, describes the velocity of the polymer. As well as that, they find the electric field that affects the polymer by solving the Laplace equation.

Reznik's model also creates an experimental setup in order to validate the simulations for the model created, which were solved using the Kutta–Merson method. Figure **5.1** shows the results obtained by Reznik, both from the experimental setup (left side of the figure) and the simulated case (right side of the figure). Note from

this, the creation of a Taylor cone. This results will be furtherly compared with the simulations generated.

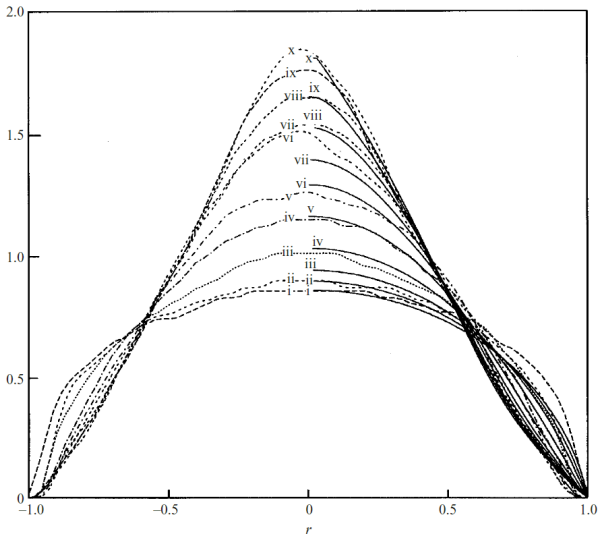


Figure 5.1: Experimental and theoretical results obtained from Reznik's model. *taken from (3)*

5.2 Models based on tracking the movement of an interface

There are several models available on the literature that model the process of electrospinning. Chen et al (22) purposed a methodology to simulate electrospinning, using a variation from the level set equation, modeling the Navier Stokes equations, with the external forces from gravity, surface tension, and electrostatic forces modeled into it. Figure **5.2** shows the Taylor cone obtained by simulating this model, using *COMSOL Simulation*.

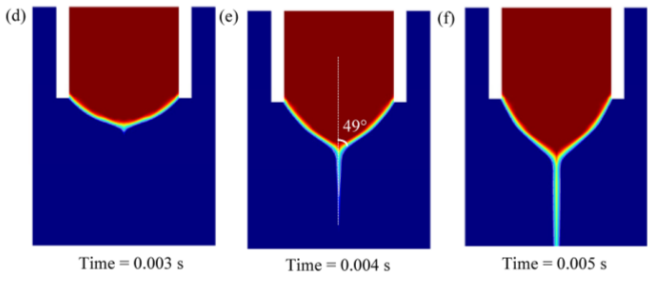


Figure 5.2: Theoretical results obtained from Cehn's model. *taken from (22)*

Rafieli et al (23) also built a model for electrospinning, however, in order to simulate the movement of the polymer air interface, they opted to solve the Cahn-Hilliard equation, coupled with the Navier Stokes equation. As well as the inclusion of Flick's law of mass diffusion, used to describe the rate of diffusion of one substance through another, they modeled the electric field around the set up using Maxwell's equations.

This leads the model built to be a system of 6 differential equations, which is solved using FeniCs, a Python package used to solve partial differential equations using the Finite Element Method. Figure **5.3** shows the simulations done for the Rafieli et al model, noting that the formation of a theoretical Taylor cone is not as evident as in the previous models exposed.

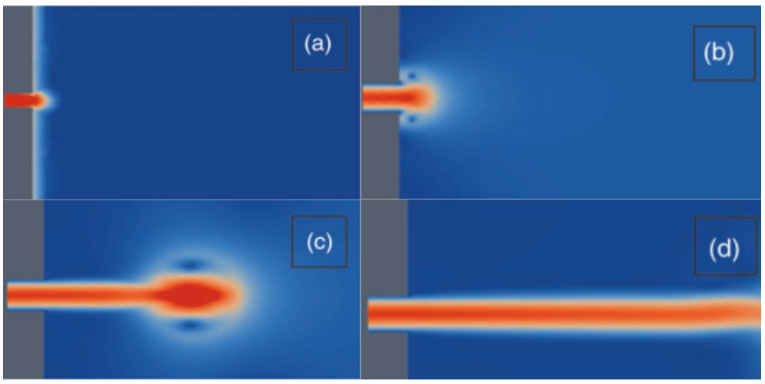


Figure 5.3: Theoretical results obtained from Rafieli's model. *taken from (23)*

6. Mathematical model

The model done for electrospinning contains the triple interaction between electrostatic forces, surface tension effects and gravity. The objective of this chapter is to thoroughly explain how each force is calculated, and the different ways in which the polymer that will move during the electrospinning process, which will also be discussed in this section.

6.1 Electrostatic Interactions

In order to form a Taylor cone, the interaction between the applied voltage and the polymer is one of the most important stages of electropinning. In order to model the electrostatic forces that cause movement between the polymer, several assumptions must be made. First and foremost, the assumption that the polymer is a perfect conductor, rather than a leaky dielectric is made, assuming that the electrostatic forces will only alter the polymer on the boundary.

In order to find the electrostatic forces, the electric potential has to be first encountered. In order to do that, using FEM, the classic Poisson equation for electric potential must be solved, as follows:

$$\begin{cases} \frac{\partial^2 \phi}{\partial t^2} = 0 & x \in \Omega \\ \Gamma_n = V & x \in \partial\Omega \quad , x \in \text{ needle} \\ \Gamma_g = 0 \end{cases} \tag{6.1}$$

Equation **6.1** Has the boundary conditions that express that the voltage is to be equal to the voltage applied in the polymer part, and for it to be 0 in $y = 0$, which is analogous to ground. Figure **6.1** shows an initial trial simulation for a symmetric surface, so that $5kV$ are applied towards the needle. Note that upon solving Equation **6.1** with FEM, it is clear that a value for the electrostatic potential can be obtained for every node.

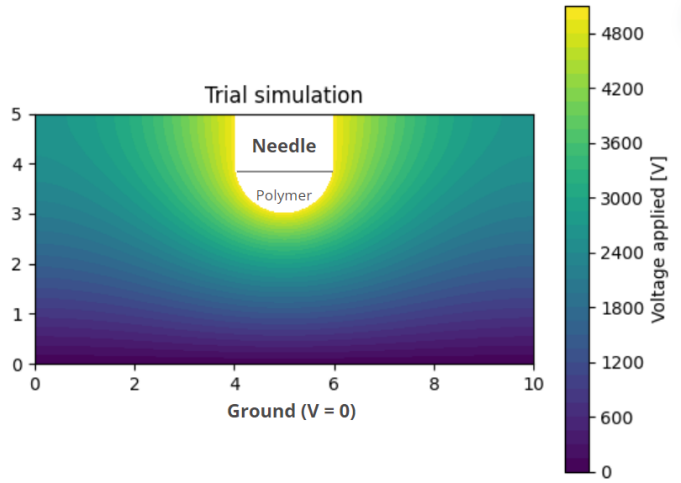


Figure 6.1: Preliminary simulation for voltage induced in electrospinning setup, for $V = 5kV$

Consider the definition for an electric field, given an electric potential ϕ, as follows:

$$\vec{E} = -\nabla\phi \tag{6.2}$$

Note that projecting the results obtained in figure **6.1** into an appropriate Vector space and taking its gradient, a vector expression for the electric field on every node can be easily derived. Figure **6.2** shows how a resultant electric field would look, using the visualization software, *Paraview*.

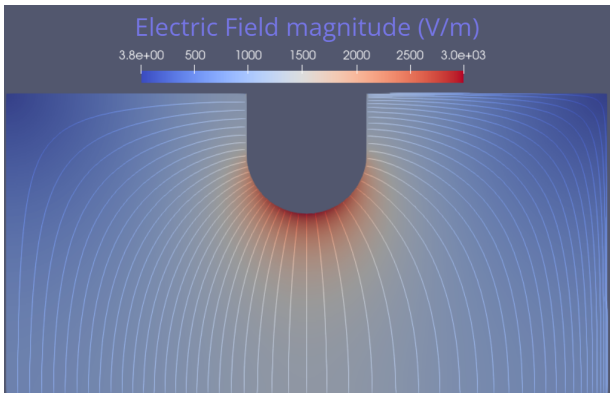


Figure 6.2: Preliminary simulation for Electric field induced in electrospinning setup, for $V = 5kV$

Now, once the electric field is obtained, the electrostatic force is defined as

the flux of the Maxwell stress tensor (M) through a surface S, ignoring magnetic effects, Such that:

$$F = \int_S M dS \tag{6.3}$$

The electrostatic interactions can be calculated using the definition for the Maxwell Stress tensor, expressed by Riva (24), which follows as:

$$F = \epsilon \int_S (E\hat{n}) \cdot E - \frac{1}{2} E^2 \hat{n} dS \tag{6.4}$$

Where $\hat{n}$ refers to a vector normal to the surface and ϵ refers to the electric permittivity. Now, let the following, $E = [E_x, E_y] \quad \hat{n} = [n_n, n_y]$. Taking the definition given by equation **6.4**, and having the following identity $E^2 := \langle E, E \rangle$, the following expression is obtained.

$$\frac{1}{\epsilon} F = \int_S (E_x n_x + E_y n_y) \cdot E dS - \frac{1}{2} \int_S \left(E_x^2 + E_y^2\right) \cdot \hat{n} dS \tag{6.5}$$

Removing the integral sign, in order to have the expression formed for every node, rather than for a continuum and expression for the electrostatic force affecting the polymer is obtained. Figure **6.3** shows the electrostatic force around the boundary of the polymer for a trial example. Note that the value of the force only makes sense in the context of the boundary of the polymer, as it is the only area where there is a vector normal to the surface.

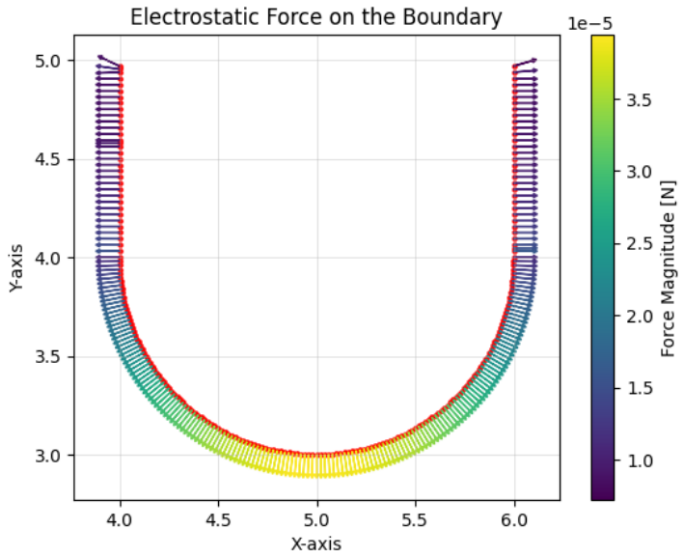


Figure 6.3: Preliminary simulation for Electric force induced in electrospinning setup, for $V = 5kV$

The normal vector to the interface can be obtained by solving the following problem. Find n, such that $n = n_0$ on $\partial\Omega$ where n refers to a normal vector. The variational problem is then written as:

$$\int_{\partial\Omega} n_h \cdot v ds = \int_{\partial\Omega} n \cdot v ds \tag{6.6}$$

The normal vector can also be calculated by taking a segment that unites every two nodes and finding a segment proportional to it. These applications on how different normal vectors can be obtained are discussed in further chapters.

6.2 Surface tension

Surface tension is a measure of the cohesive forces between molecules at the surface of a liquid, leading to the formation of a thin layer with higher energy compared to the bulk. It's crucial in electrospinning modeling as it governs the formation of the jet during the process. Surface tension plays a pivotal role in maintaining the stability of the Taylor cone and controlling the diameter of the resultant fibers, influencing the overall efficiency and quality of the electrospun material.

The definition for surface tension purposed by (25) reads as follows:

$$\int_{\Omega} f_\sigma = \oint_A^B \sigma dt = \oint_A^B \sigma\kappa\hat{n} dS = \int_{\Omega} \sigma\kappa\hat{n}\delta_s \tag{6.7}$$

Where κ refers to the curvature, $\hat{n}$ is a normal vector to the surface, σ is the surface tension coefficient and δ_s is a Dirac's delta function that is equal to one in the surface and is equal to zero everywhere else. This, in order to guarantee that as the name says it, surface tension only has values on the surface.

Now, considering that this model is to be solved using the finite element method, its recommended to remove the integral in equation **6.7**, as the value desired is the value on each node rather than the value on the total surface. This results on the following definition for the surface tension on every node:

$$F_i = \sigma\kappa\hat{n}\delta_s \tag{6.8}$$

Note that the only parameter that must be found is the value for the curvature κ, which (25) presents different ideas into how to model it.

A nearly horizontal interface can be simply described by a height function $y = h_y(x)$. The curvature can be described by the following differential equation.

$$\kappa = \frac{h_y''}{\sqrt{1 + h_y'^2}} \tag{6.9}$$

Applying a finite differences scheme to equation **6.9**, the following discrete equation is derived:

$$\kappa_i = \frac{(h_{i-1} - 2h_i + h_{i+1})/\Delta^2}{\sqrt{1 + [(h_{i+1} - h_{i-1})/(2\Delta)^2]}} \tag{6.10}$$

Where Δ just refers to a timestep and h is the height function. Note that this approximation is only valid if the curvature is nearly horizontal, which means that other approaches might have to be explored. As well as that, the curvature estimation is quite sensible, so impressions can easily be encountered.

Surface tension calculation through Level set methods

A method that has been explored theoretically but not algorithmically, is by using a level set function in order to model surface tension.

In the case of level set methods, let ϕ be a level set function. The following expression, helpful for calculating surface tension, can be derived:

$$\hat{n} = \nabla\phi \quad \text{and} \quad \kappa = \nabla \cdot \hat{n} \tag{6.11}$$

6.3 Formulation for gravity effects

As mentioned by (6), most model don't include the effects of gravity effects. However, for reasons that may become apparent, an inclusion of gravity might be necessary. Take the definition presented by (22), in which gravity is taken nodally, equally distributed between all the nodes in the mesh. Take the volumetric definition for gravity:

$$F_g = \rho g \tag{6.12}$$

Where ρ describes the density of the polymer. The discretization scheme taken is quite simple and intuitive. Let:

$$F_{gi} = \rho \frac{g}{n} \tag{6.13}$$

Where n describes the number of nodes that will be affected by gravitational effects. Figure **6.4** presents an scheme of the loads distributed on every node.

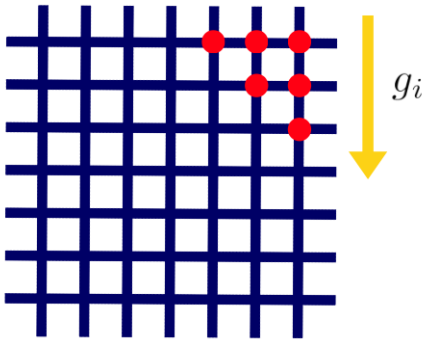


Figure 6.4: Scheme for gravitational effects on a mesh

6.4 Formulation for a solid model

In order to model the deformation of the polymers on electrospinning, due to the forces defined previously, two different approaches were taken. The first model consists of working under the assumption that the fluid in question acts as a highly viscoelastic solid, with an extremely low elasticity constant. This assumption aids to model the effect of forces, as solids are significantly easier to model than fluids. Some of the arguments in favor of taking this modelling approach are as follows:

1. A material's Young's module (elasticity coefficient) describes elongation and also compression, however, compression will never be included on models done, as a force normal to the surface is applied at all times. This also means that if the viscoelastic model proposed by Maxwell is the one used to model, there would be no damper, because there is no relaxation, so the viscous terms would be negligible.

2. Polymers in liquid states are highly viscous, so their movement is more similar to that of a solid, so the model wouldn't be as far from the truth.

3. Electrospinning occurs in a solid/liquid state anyway, as polymers must solidify during the process.

6.4.1 Definition of a solid problem

The problem to be solved if the polymer is considered as a solid reads as follows. Take E be a material's Young modulus, which represents its stiffness or resistance to deformation under tensile or compressive stress. and ν its Poisson's coefficient,

which represents its ratio of lateral strain to longitudinal strain when subjected to stress.

Let the following parameters to be defined in terms of E, ν, known as Lamé's parameters.

$$\mu = \frac{E}{2(1+\nu)} \qquad \lambda = \frac{E \cdot \nu}{(1+\nu)(1-2\nu)} \tag{6.14}$$

Letting ϵ be the strain (deformations) and σ to account for the stress (force per unit area), both written in tensor form, the following relation is defined:

$$\sigma = \lambda \cdot \mathrm{tr}(\epsilon) \cdot I + 2\mu\epsilon \tag{6.15}$$

Where I defines the identity tensor. Take equation **6.15**, and adding the external forces purposed, whose sum will be represented as F, the following differential equation that relates all of the components is given, as follows,

$$\nabla \cdot \sigma + F = 0 \tag{6.16}$$

The weak form for equation **6.16** can be derived to obtain the following equation, letting v be a test function that lives on an appropriate function space:

$$\langle \sigma(v), \epsilon \rangle = \langle F, \epsilon \rangle \tag{6.17}$$

Recall that the forces must be included in the problem element by element, however, the methodology used to do that will be discussed on the next chapter. This is a classic elasticity problem, that is commonly solved in the context of FEM analysis, using Lagrange $\mathbb{P}^1$, however the order can be increased if seen necessary, in order to improve presición.

6.4.2 From Viscosity to elasticity

One major drawback of modeling viscosity using elastic properties is how to go from a viscous constant, such as the one that liquid polymers present, into representing it, as a Young modulus. This setback is addressed by taking the definition of σ, the stress tensor, and ϵ, the strain tensor, based in a viscous and in an elastic regime, which are defined as follows:

$$\sigma = E \cdot \epsilon \quad \sigma = \eta \cdot \dot{\epsilon} \tag{6.18}$$

Making the said equations equal through σ, the following differential equation is formed:

$$E \cdot \epsilon = \eta \cdot \dot{\epsilon} \tag{6.19}$$

Consider that the strain over time tends to be linear, which inherently means that, given an auxiliary constant γ:

$$\epsilon = \gamma t \quad \dot{\epsilon} = \gamma \tag{6.20}$$

By combining equations **6.19** and **6.20**, the following relation is obtained:

$$E \cdot t = \eta \tag{6.21}$$

which under constant time becomes:

$$E \cdot \tau = \eta \tag{6.22}$$

This serves as evidence that a material's Young modulus and its viscous coefficient present a linear relation.

6.5 Formulation for a fluid model

Take the Navier Stokes equations, which, lets recall from Equation **4.9**, are written as follows:

$$\begin{cases} \nabla \cdot \vec{u} = 0 \\ \rho \frac{D\vec{u}}{Dt} = -\nabla P + \rho \vec{g} + \mu \nabla^2 \vec{u} + \vec{F} \end{cases} \tag{6.23}$$

Note that all of the parameters are given, so the only thing that would be needed is to solve for $\vec{u}, P$. Now, a level set function, like the one explored on equation **4.16** is then solved to see how, based on the values obtained from $\hat{u}$ in the Navier Stokes equations make the interface ϕ change position.

Note that as ϕ describes the interface change between air and polymer, parameters, such as viscosity and density must change depending on the value of ϕ. This is why, the following artificial values for μ and ρ need to be defined, as follows.

$$\mu = \mu_{polymer} + \frac{1}{2}(\mu_{air} - \mu_{polymer}) \cdot (1 + \tanh(2\phi)) \tag{6.24}$$

$$\rho = \rho_{polymer} + \frac{1}{2}(\rho_{air} - \rho_{polymer}) \cdot (1 + \tanh(2\phi)) \tag{6.25}$$

The argument included in the hyperbolic tangent refers to the thickness of the interface that one desires. Upon probing with different thicknesses, 2 was shown to be the most appropriate. Figure **6.5** shows the way in which the density described on equation **6.25** behaves.

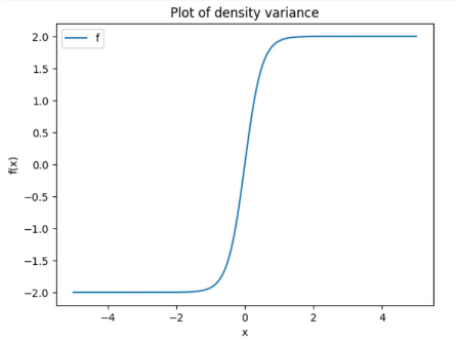


Figure 6.5: Density variations due to ϕ

In order to solve the Navier Stokes equations, (15) recommends using a pair of elements known as *Taylor Hood Elements*, which basically refer to solve for the pressure with a Lagrange polynomial of order n and for velocity with a Lagrange polynomial of order $n+1$. (18) suggest to use the same order of element for velocity and for the function ϕ, so the following elements $(\mathbb{P}^1, \mathbb{P}^2, \mathbb{P}^2)$ were used to solve for Pressure, velocity and ϕ respectively.

7. Description of the algorithms implemented

7.1 Computational Tools

All of the models that are presented in the form of partial differential equations (PDE) were solved numerically, with the help of legacy *FeniCs*, a Python and C based library, used to solve PDEs using FEM. The methodology behind FeniCs (15) is to generate a mesh, give a variational problem (with the boundary conditions, if needed), define the test functions to be in a desired space, and solve the bilinear form given by the discretization presented.

FeniCs doesn't run on a Windows operating system and presents several difficulties when being installed on an Ubuntu system. This is why *FEM for all* (26), a project developed to include all of the tools provided by FeniCs, with the benefit of having it all inside a *Google Colab* notebook, was used to create all of the scripts. All of the codes relating to this work can be found on (27).

In order to create some meshes, the mesh editor that comes with Fenics didn't show to be enough in terms of geometry, so the mesh generation software *GMSH* was used to create .geo() mesh files, which would be later read and operated by FeniCs.

FeniCs does not come with a visualization tool, so the Python library *Matplotlib* can be used to see the resultant functions. However, Matplotlib was evidently not created to plot vector functions in complex geometries, especially when there is a time component included in the PDE. This is why it was found that Matplotlib committed several mistakes and incoherences when plotting. *Paraview* is a data visualization, open source, software designed to handle large datasets and supports a variety of visualization techniques to help users explore and analyze their data, such as streamlines for vector fields, which makes it much more appealing for visualizing material.

7.2 Case for validation

As a tool to compare the model built, all of the models were compared to the model, simulation, and experiments built by Reznik et al (3). On their experiment, they worked with the polymer **PLC2**, and the main parameters that they used are as follows.

1. Voltage applied: 13.5 kV
2. Distance from needle to collector: 5 cm
3. Radius of the needle: 1 mm
4. Fluid Viscosity: 212 Pa · s
5. Fluid Density: 1.32 g/cm^3
6. Surface Tension Coefficient: 28.9 g/cm^2
7. Fluid permittivity: 21.3
8. Flux out the needle: $0.003 m/s$

Note that several other properties that are included on Reznik's model, such as conductivity, are not used on the model built, so are ignored for the moment. Figure **7.1** presents different forms of the results obtained by Reznik, which will be used to compare the simulations done.

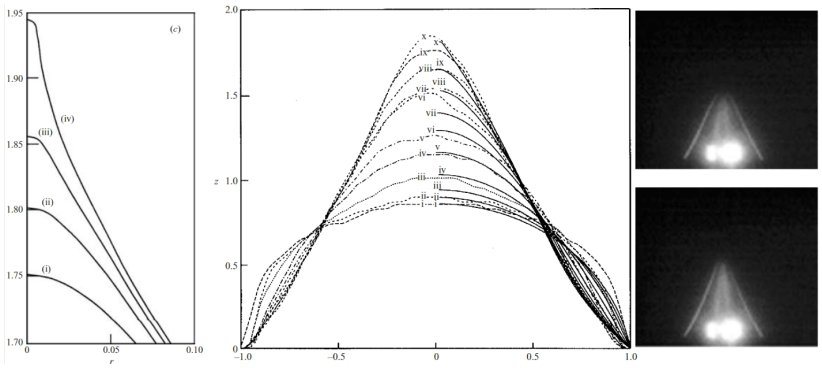


Figure 7.1: Results reported by Reznik's model and Simulation. *Taken from (3)*

7.3 Algorithm for electrostatic force calculation

Note that the electrostatic solver takes a geometry as an input, such as the one shown in figure **7.2**, where the main objective is to find the electrostatic forces on the boundary between Ω_1 and Ω_2.

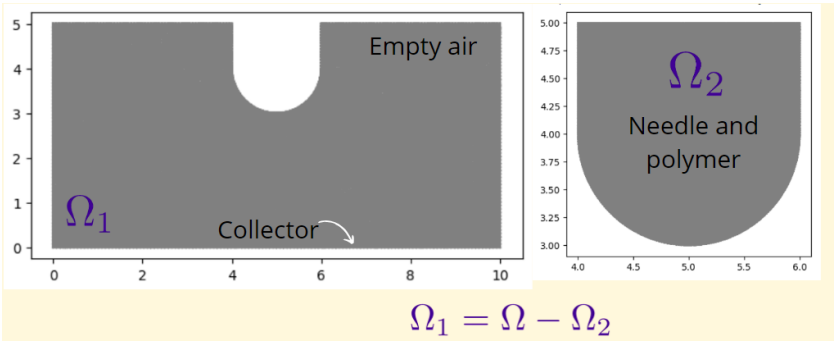


Figure 7.2: Initial mesh received by electrostatic solver

Two different, but analogous approaches were taken towards finding the electrostatic forces once an electric field is defined, as shown in figure **6.2**.

7.3.1 Calculating the forces on each point

Once an electric field is defined, the first approach taken is to export the values of the E at each node (E_x and E_y), and operate with them out of FeniCs, in order to get the Forces. The code can be seen on (27). The main idea would be to take the values for E_x and E_y for two consecutive points, and using the equation of the line, define a segment between the two points.

Taking the multiplicative inverse slope of said segment, a perpendicular line will be created, that of a normal vector to the surface. Making the vector unitary, a form such as the one shown in Figure **7.3** results.

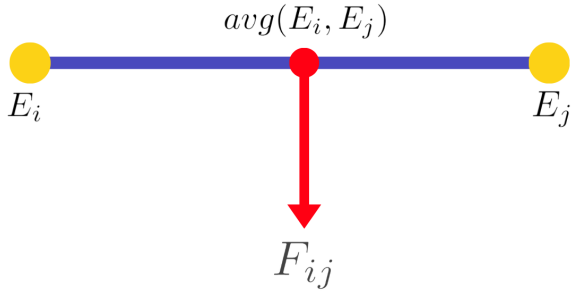


Figure 7.3: Normal vector Calculation

These gives the components n_x, n_y for a unit normal vector to the surface. Once the force is calculated, taking Maxwell's definition, every average point will have a force value associated to it. Now, a projection would have to be made, in order to have a force value associated to each node.

Take, for instance, two forces, like the ones shown in figure **7.4**. The model designed in order to find the value of F_3, takes the distances from each middle segment and each node, to find the x, y components of the force, as explored on equation **7.1**.

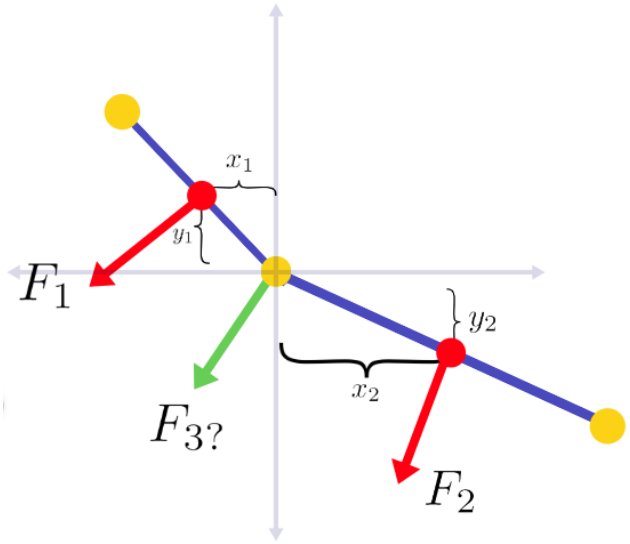


Figure 7.4: Normal vector Calculation

$$\begin{aligned} F_{3x} &= \left(1 - \frac{x_1}{x_1 + x_2}\right) F_{1x} + \left(1 - \frac{x_2}{x_1 + x_2}\right) F_{2x} \\ F_{3y} &= \left(1 - \frac{y_1}{y_1 + y_2}\right) F_{1y} + \left(1 - \frac{y_2}{y_1 + y_2}\right) F_{2y} \end{aligned} \tag{7.1}$$

With this algorithm, it is guaranteed to obtain a Force value for every node on the surface.

7.3.2 Calculating the forces as a function

The second approach taken is using the fact that Equation **6.6** provides a weak formulation used to find a normal vector. That being said, this equation was applied in order to find a vector field with all of the normal vectors, which was later operated through FeniCs to get to the Force.

Remembering that the Force is defined as the flux of the electrostatic Maxwell tensor, the following operation can be applied to E to get M (The stress tensor):

1. M = -0.5 * epsilon * dot(E, E) * Identity(2) + epsilon*outer(E, E)

This computational definition matches with the definition provided by (24). The force on every node is the resulting dot product between M and $\hat{n}$.

This approach, as opposed to the previous one, gives the electrostatic forces, not one by one, but as a FeniCs function containing every force. Mathematically speaking, the two approaches taken are equivalent, but computationally speaking, both have their benefits and disadvantages, which is why both algorithms were kept.

7.4 Algorithm for a Solid Model

Figure **7.5** Shows an overview of how a solid model should behave, starting with the application of forces and proceeding with the deformation of the polymer, solving the viscoelastic model, repeating this process until the formation of a Taylor Cone (or until desired).

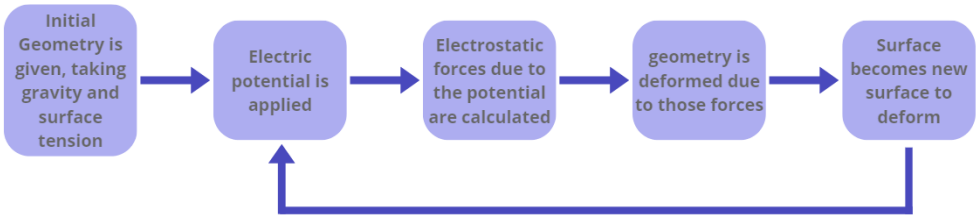


Figure 7.5: Diagram that explores algorithmically, the behaviour of a electrostatic model

The algorithm must only take the polymer part of the mesh as an input, so if the mesh described in Figure **7.2** acts as the initial, Ω_2 would act as the domain for the elastic solver, making sure to calculate the deformations on every node from the boundary.

The electrostatic forces and surface tension effects are calculated and must be introduced in the solid model, however, note by the definition for surface tension, that a curvature is needed, so if the initial configuration for the polymer is a rectangle interphase, one iteration must be held, in which only the values for gravity are considered. This, in order to form a curvature, as the surface tension solver needs it to work, and the electrostatic solver acts as an effect from the inverse square law, so for it to make some sense, the solver is imperative. Figure

7.6 shows a pseudocode of how the solid solver might act, including the disclaimer that gravity effects might be necessary to obtain results.

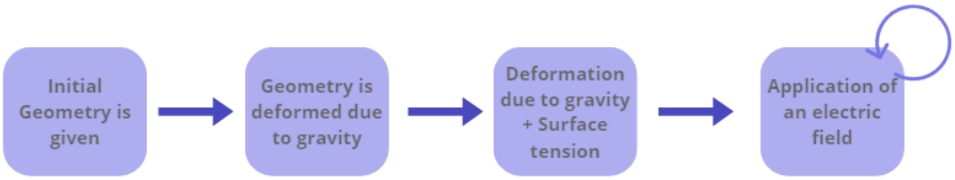


Figure 7.6: Diagram that explores algorithmically, the behaviour of a solid model

Once the value of the force that must be applied on every node is given, two different approaches are discussed for applying said forces on the variational form.

7.4.1 Dirac Delta

Consider that the forces were defined so that each node has one force applied to it. Knowing this, it would be appropriate to use a method that takes a resized impulse on every node, which acts as a force, and then do the same thing for all of the nodes. Figure **7.7** shows an idea of how this could work, by taking a horizontal and vertical impulse and reshaping them in order to match the force value.

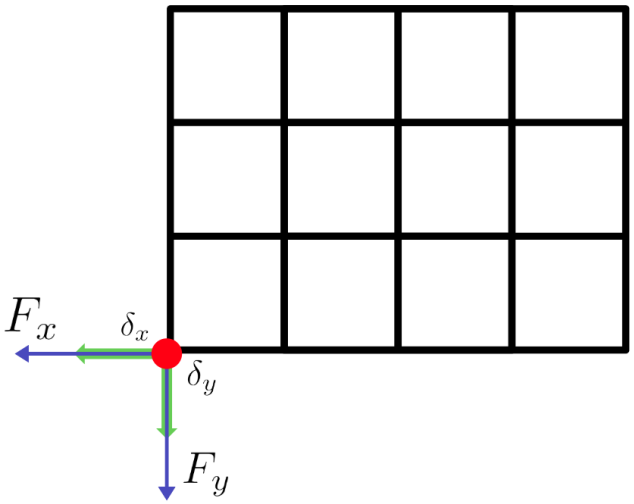


Figure 7.7: Diagram that explores the behavior of Dirac Delta impulse

FeniCs does not come with a definition for Diracs's Delta, so the following

continuous form for an impulse was defined, for an impulse located in a point x_0, making the disclaimer that x_0 can be both a scalar or a vectorial quantity:

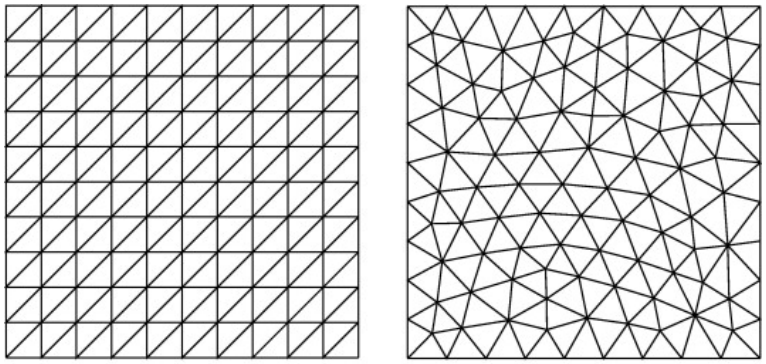
$$\delta(x) = \frac{|x - x_0|}{\pi(x - x_0)^2} \tag{7.2}$$

The main issue that comes with this is that computing each Delta, inherently means that when a δ function is defined, it must be defined in the whole domain, even if it has the value of zero for almost all of the points. This makes that the program can't take more than a set quantity of nodes without running for days at a time, which means that an optimization routine to find an inclusion of inputs may be needed as a future work. The codes for this algorithm can be found on (27).

7.4.2 Subdomains on structured meshes

FeniCs has a built-in function for the creation of subdomains, which permits the creation of subdomains over a mesh that can be operated individually. This being said, one subdomain can be defined for every node in the mesh, and an individual force can be applied on every node, then summing the contributions to arrive to a global deformation.

This approach seems to be perfect, however, subdomains are defined geometrically, which means that in order to have squares of equal size as subdomains, the mesh created must be structured. The creation of a structured mesh can sometimes be a difficult process and must be done outside of FeniCs. This occurs precisely because in a structured mesh, it is guaranteed that all of the nodes are located at the same distance one from another, as shown in Figure **7.8**. If this algorithm is performed in an unstructured mesh, problems defining the subdomains will arise, as not every subdomain will be covered by the same amount of nodes.



(a) Structured mesh (b) Unstructured mesh

Figure 7.8: Comparison of structured and unstructured mesh. *Taken from (28)*

This approach results much faster than implementing Dirac's Delta, so if calibrated correctly, can be a better tool. All the code is found in (27).

7.4.3 Mesh Post-Processing

Once a deformation is achieved, the loop must be repeated, which means that the new, deformed geometry must be passed through the electrostatic solver in order to take the new electrostatic forces that affect the new model. This creates the need to generate a new mesh, one that has the same outline that the one shown in figure **7.2**, but where the form of Ω_2 is defined by the form of the previously deformed surface.

This means that a new mesh must be generated, one that FeniCs does not have the technical capabilities to create, so GMSH must be used. To do so, take the boundary values for the deformed mesh and sort them due to a point, in terms of their angle in respect to it, as shown in Figure **7.9**.

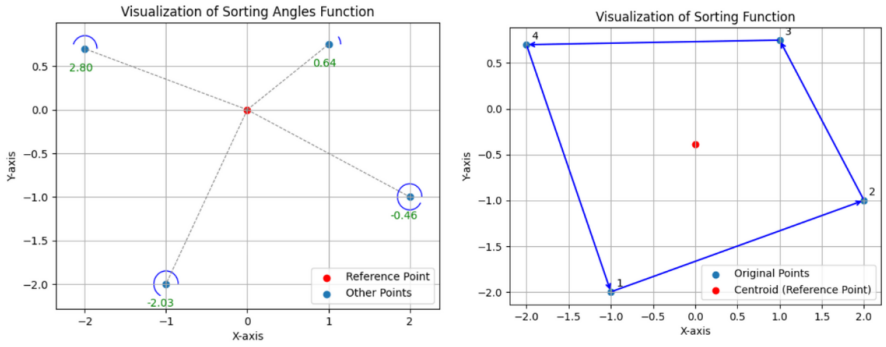


Figure 7.9: Visualization for sorting algorithm

Once the points are sorted, a Python program can be executed to create a **.geo** geometry file, by taking the sorted points and creating segments between them. This file can be read by GMSH and a new mesh can be easily generated. To guarantee some continuity between the data meshes, it is recommended for every mesh to have the same number of nodes.

7.4.4 Experimentation

Take the experiments done by Reznik et al (3), which can be visualized in figure **5.1**. The solid model was calibrated with the same parameters, to attempt to see how it would be deformed under electrospinning. Figure **7.10** shows different iterations for the deformation that the solid polymer presents as a function of the

forces applied. It can be seen that, as expected, the center of the polymer is the one that elongates the most (due to the inverse square law), and that a semi-conical shape is indeed obtained.

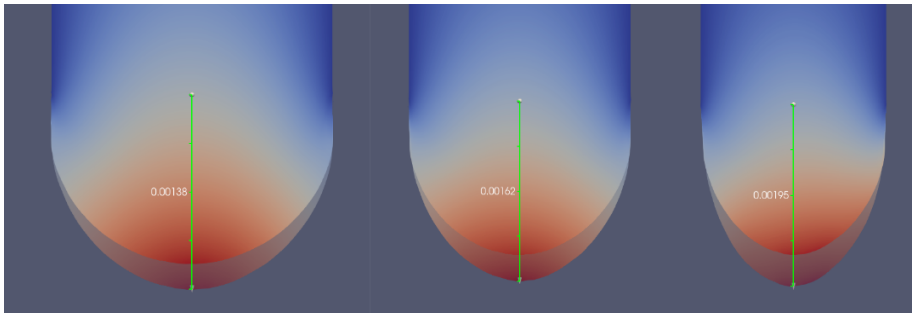


Figure 7.10: Iteration for a solid model

Figure **7.11** shows a side by side comparison between Reznik's results and the results obtained. Note that the elongation of the reported simulation is longer and wider, which might be an effect of the fact that the solid does not have the properties for viscosity, which makes that Reznik model take a more conical interface. In terms of length, the simulated cone has a length of **2.33 mm**, somewhat longer than the one reported by Reznik et al. The biggest difference between the two results is the angle, where Reznik reports a half angle on the tip of 28^o, whereas the simulated solid shows an angle of 23^o.

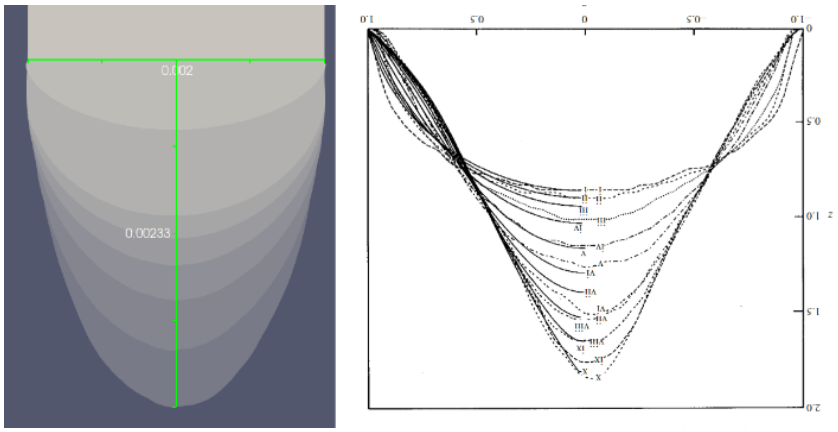


Figure 7.11: Comparison between solid model and Reznik's experiment

This approach towards modeling helps create a visual into how the Taylor Cone looks, however, due to the lack of modeling power, and to the assumption that the material dealt with has a solid nature, the scope of this model does not go past this

stage on the process of electrospinning, failing to simulate correctly the moment when the region of the straight jet ejection should occur. This is why, it is not recommended to treat electrospinning as a solid problem, as the solution can be quite general.

7.5 Algorithm for a Fluid Model

Based on the model discussed, a simulation for a fluid, based on the Navier Stokes equations and the Level Set formulation, iterating due to forces acting upon the boundary between air and polymer must be built. Figure **7.12** shows the overview of how the model works.

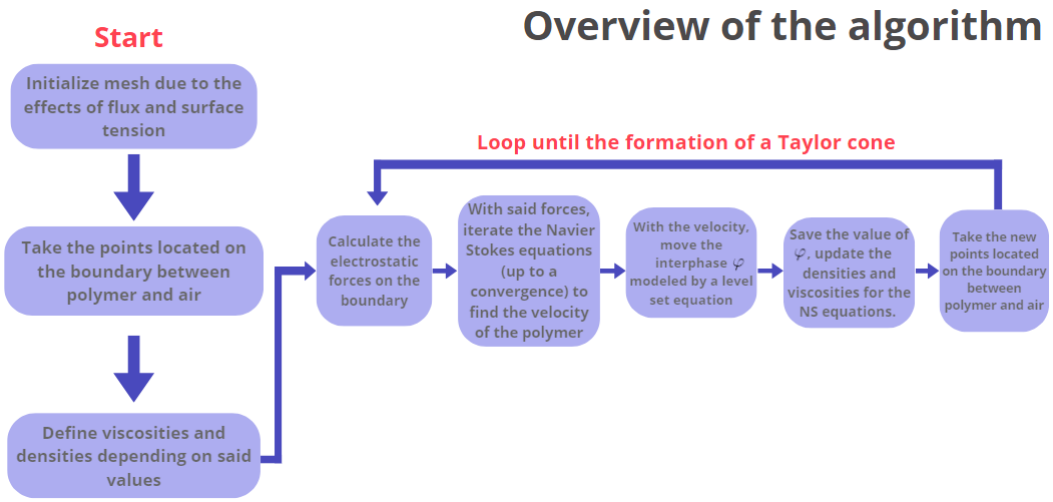


Figure 7.12: Overview for fluid model

Several differences come from modeling this program that were not present in the formulation of a solid model. First, the mesh that is dealt with is one structured square mesh, as the geometry of the polymer will be defined by the Level Set function. As well as that, during the electrospinning process, there is a parameter that can be set, that is the flux coming out of the needle, which is defined by how the needle is programmed. This parameter cannot be included upon considering a solid model, however, a fluid model admits this change.

7.5.1 Initial mesh generation

The interest of this study is not to see what occurs when the polymer is still inside the needle, but to see what happens when the polymer leaves it. This is why, for

simplicity, to avoid the numerical problems found on corners, and the enforcement of boundary conditions on level set solutions (especially given that there is still no algorithm for the reinitialization), the model will start on the moment where the polymer leaves the needle. Figure **7.13** shows the idea behind the model, and a zoom of the implementation of it coded in FeniCs, remembering that the concentration value presented (red for polymer, blue for air) is precisely the level set function that is to be iterated, to see how the interface moves.

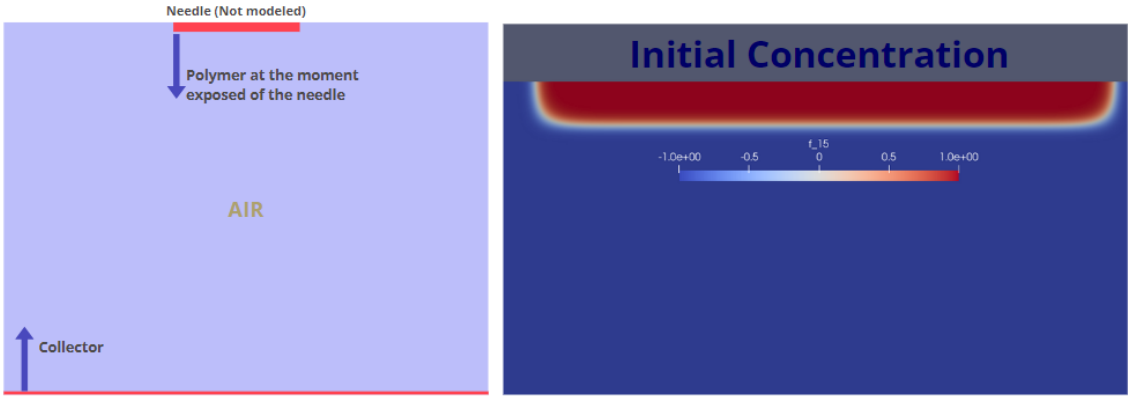


Figure 7.13: Initial mesh in a fluid approach

Geometrically speaking, the approach into the formation of a rectangle-like interface, the following definition for a smooth Heaviside function is taken, to obtain an initial curvature.

$$H = \frac{2}{1 + e^{\frac{[(x-x_0)^p + (y-y_0)^p]^{1/p} - r}{\delta}}} \tag{7.3}$$

Where δ defines the thickness of the interface, r is the radius of the geometry and x_0, y_0 define where the interface must be located. Note that if the value of $p = 2$, the geometry created would be a circle, however, the more the value of p is augmented, the closer it becomes to a square. The ideal case would be to have $p \implies \infty$, however, for computational reasons *(overflow due to giant numbers that numpy cannot understand)* a number close to $p = 20$ was assigned, a number that guarantees a polymer-air interface smooth enough.

It is noteworthy that the thickness of the interface must not be smaller than the separation between nodes, to ensure continuity and that the part of the mesh that will account to fluid flow must be structured, to obtain a well defined boundary interface boundary.

7.5.2 Initialization of the model

Note that the level set function is already initialized, which, as seen in figure **7.13**, every node has a level set number associated to it. As a way to find the interface between air and polymer, note that it is the place where $\varphi = 0$. Let ϵ be a small enough number, the values for $\varphi < \epsilon$ will be found, and those will account for the polymer-air interface. The value of ϵ must be calibrated on every iteration, by hand, as it is hard to predict the exact value for ϵ that will produce only one isoline, that will simulate the boundary.

Figure **7.14** Shows the isolines for where the nodes that account for the polymer-air interface are located. This plot shows essentially, the importance of working with structured meshes, as an unstructured mesh will lead to a disorganized interface.

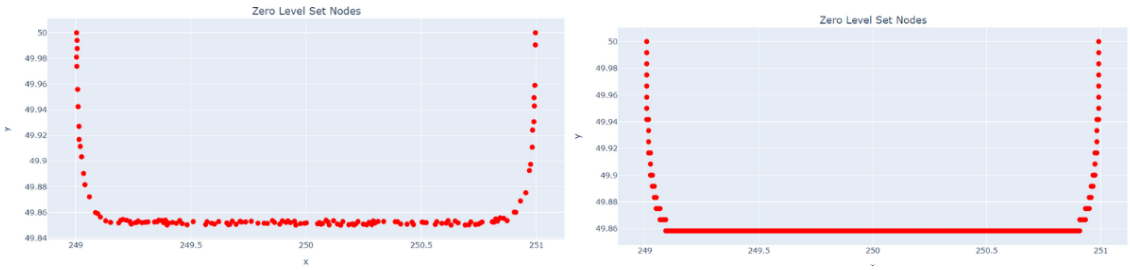


Figure 7.14: Overview for fluid model

Due to the inclusion of flow into the model, the effects of gravity can now be discarded from the solver, as an initial curve can be modeled only through interactions as a boundary condition, defined only on the top of the mesh, exactly on the points within the boundary in which the polymer is defined. As well as that, it is noteworthy that as gravity was to be incorporated into every node (not just on the boundary), so it was rather costly (numerically speaking) to compute, especially taking into account that gravity tends to be a negligible force.

7.5.3 Iteration of the Navier-Stokes equations

The first value that must be calculated is the effect due to electrostatic and surface tension forces, which clearly depend on the geometry of the boundary. That being said, the values for which $\varphi < \epsilon$ will be taken, and a *.geo* geometry will be created with the said points. Take said geometry and mesh its outside, so that the electrostatic solver finds the forces on the boundary, analogous to how it is done in the solid model.

Once the forces are obtained, the Navier-Stokes equations must be solved to calculate the velocity vector field in which the polymer flows. The first step is

to define the values for density and viscosity (ρ, μ), which must be modeled as a hyperbolic tangent, as described by equation **6.24**. The electrostatic and surface tension forces must be added as external forces in the weak form of the Navier-Stokes equations, which can be done by taking the values for each force, and the coordinate in which they are to be defined, and then defining a FeniCs vector function that only contains said forces. The implementation for this vector can be found on (27).

The Navier-Stokes equations take several iterations to reach a steady state and tend to present an overshoot during the first iterations due to the external force term. To avoid this, the Navier Stokes equations must be iterated several times to try and find some sort of convergence. The convergence criterion defines takes two consecutive values for the velocity $\vec{u_n}, \vec{u_{n-1}}$, and use the L^2 norm operator to define convergence as follows, letting $\epsilon > 0$ be a small number, around the order of 10^{-6}:

$$||\vec{u_n} - \vec{u_{n-1}}|| < \epsilon \tag{7.4}$$

The fluid flow algorithm will be stopped when said criterion is fulfilled. Figure **7.15** shows the ways in which the polymer flows, which results similarly (in terms of shape) to the vector fields shown by (22).

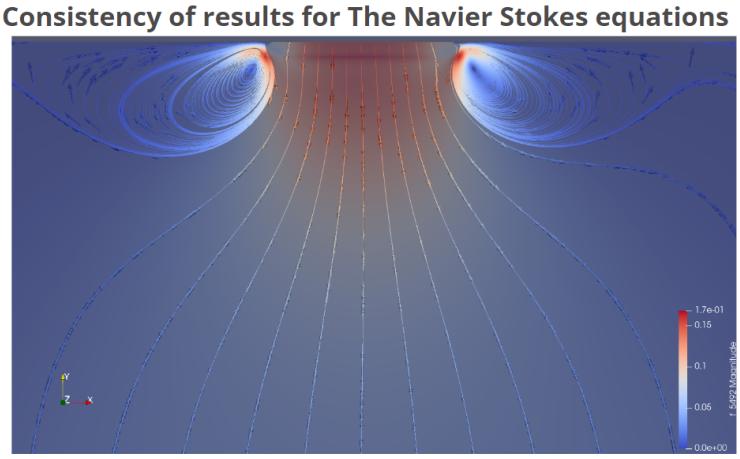


Figure 7.15: Navier-Stokes solution consistency

Problems with the Navier-Stokes implementation

Considering the complexity in both time and memory that comes with solving the Navier-Stokes equations, several issues were identified. As the value of the force changes for every node, FeniCs must create n subdomains to solve each iteration, so the program quickly runs out of RAM. As the code has been elaborated in Colab,

whose servers have limited RAM, this nourishes the problem. At the moment, having 200 nodes on the surface is enough to make the algorithm crash.

Several methods can be explored that reduce the complexity of algorithms, some of which are incorporated in FeniCs. FeniCs has a built-in solver called **Bic-Stab** which is capable of solving linear systems ($Ax = b$) faster with a preconditioner, known as Hypre-amg. The idea is to gradually approximate the result of the system of equations that FEM solves. The main steps of the process are as follows.

1. Begin with an initial guess for the solution x_n.
2. Multiply the matrix A by the current guess x_n to obtain an estimate of the solution Ax_n.
3. Calculate the residual $r_n = b - Ax_n$,which represents the error in our current estimate.
4. Apply a preconditioner to the residual to make it more amenable to solving. This step helps improve the convergence speed.
5. Use conjugate directions to determine the next search direction. This ensures that each iteration is orthogonal to the previous ones, aiding in convergence.
6. Update solutions until convergence

Further exploration in the process of optimizing FEM algorithms is recommended.

7.6 Level set consideration

Consider the formulation of the level set function, as defined by (18). The main idea is to take the vector field $\vec{u}$ obtained by the Navier-Stokes equations. The process is iterated for an artificial time τ. As stated by (18), level set equations must not include Dirichlet boundary conditions, but Neumann (transport to be cero over the boundary) as they may alter the solution's posedness. The only Dirichlet BC imposed was over the slot where flux from the needle was coming out of, to guarantee always having polymer there.

Figure **7.16** shows how the surface evolves after several iterations of the level set method. Note that as a reinitialization process is not done, some errors get in the way, as seen in the black circle.

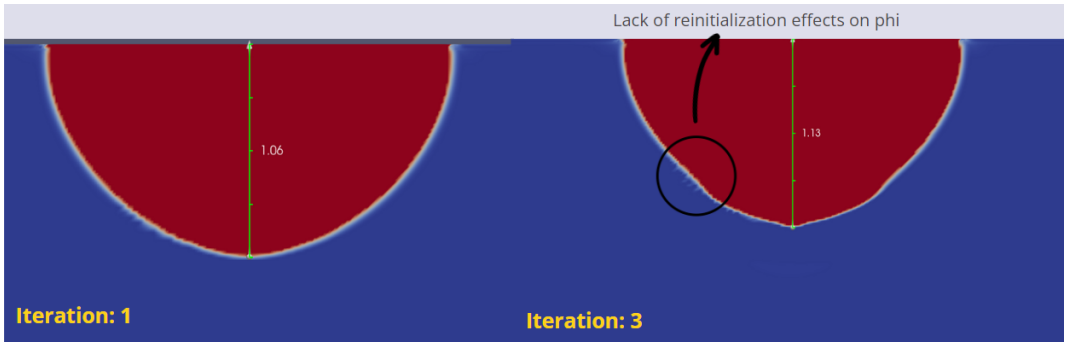


Figure 7.16: Overview for fluid model

Afer iterating the level set equation, the values for density and viscosity must be updated in order to model the movement of the polymer over the interface. A problem arises When extracting the points of the new interface, more movement is found, and less nodes are found when applying the criteria. This means that the tip of the cone tends to not be as refined. Figure **7.17** shows precisely the way in which a conical interface is evidently formed, however, due to the lack of reinitialization and the absence of nodes due to the poor computing power.

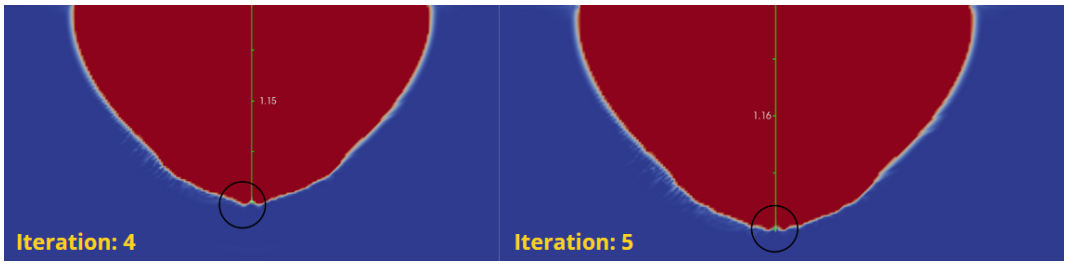


Figure 7.17: Overview for fluid model

8. Final Remarks

8.1 Improvements towards the model

The implementation of the reinitialization process for the level set equation ensures that it conserves its properties as a signed function. This helps avoid error propagation and relaxes the timestep needed for the level set iterative method.

To improve the visualization of the electrospinning process, a more robust calculation with a more appropriate mesh is proposed. This would provide a clearer and more accurate representation of the process. Furthermore, a full-scale validation of Reznik's (or any other relevant) model is recommended to ensure its accuracy and reliability. It is noteworthy that an alternative to Colab would be needed, as the RAM is not enough to clearly define a mesh as fine as needed, so other alternatives to where FeniCs must be installed are to be looked into. FeniCs provide versatility when writing weak forms for equations, so it is recommended to continue to use it as a modeling tool.

To achieve a more accurate simulation of surface tension, it is suggested to use a definition provided by Peng (22). This, is because the first assumption provided by the model used is that the surface must be nearly horizontal, which clearly is not the case when iterating. The definition for the surface tension ties the level set function to the level set, by defining F_{st} as follows:

$$F_{st} = \sigma\delta\kappa\hat{n} + \delta\nabla_s\sigma \tag{8.1}$$

Where the following definitions for $\vec{n}$ and κ are used:

$$\hat{n} = \frac{\nabla\phi}{|\nabla\phi|} \quad \kappa = -\nabla\cdot\hat{n} \tag{8.2}$$

Where ϕ is the level set function.

Another way to direct the project would be by modeling the interface evolution using Cahn Hilliard's equation, as described on the Mathematical background. This equation presents a physical meaning towards the problem of phase separation, as opposed to the level set formulation, which is only a transport equation.

8.2 Conclusion

A test model, both mathematical and numerical, has been built, capable of generating the expected geometries upon the formation of a Taylor cone. The use of FeniCs proved to be an extremely helpful tool, which permitted the shaping of PDE problems in the way needed, as well as the choice of the test elements. More research is needed in the fields of optimization are required to constitute this research into a valid reference towards electrospinning modeling.

9. Acknowledgements

I am deeply grateful to my advisors, Alejandro Ospina Vargas and Juan Manuel Rodriguez, for their invaluable guidance, support, and encouragement throughout this internship. Alejandro, your ability to think critically and propose ideas outside the box has been instrumental in shaping this work. Juan Manuel, thank you for being an emotional anchor, always ready to listen and offer advice, especially during challenging times.

I would like to extend my sincere thanks to Eafit University, particularly Claudia Palacio, for providing me with the opportunity to pursue this research in France. This experience has been incredibly enriching and has significantly contributed to my personal and professional growth.

I am also thankful to the Université de Technologie de Compiègne for their support, both financially and emotionally. The resources and facilities provided by the university have been essential for the successful completion of this thesis. Additionally, I would like to express my gratitude to the warm and welcoming people I have met at Roberval's laboratory, who helped me shape how I pursue research.

Finally, I want to express my heartfelt gratitude to my family, especially my mother, for their unwavering love and support throughout this journey. I would also like to thank my Colombian friends, who have been a source of encouragement and joy during the months that I spent in my internship.

Bibliography

[1] D. Li and Y. Xia, "Electrospinning of nanofibers: Reinventing the wheel?," *Advanced Materials*, pp. 1151 – 1170, July 2004.

[2] P. Bhattacharjee and G. Rutledge, "Electrospinning and polymer nanofibers: Process fundamentals," *Elsevier*, 2017.

[3] S. Reznik, A. Yarin, A. Theron, and E. Zussman, "Transient and steady shapes of droplets attached to a surface in a strong electric field," *Cambridge University Press*, 2004.

[4] T. D. Brown, P. D. Dalton, and D. W. Hutmacher, "Melt electrospinning today: An opportune time for an emerging polymer process," *Progress in Polymer Science*, pp. 117 – 166, January 2016.

[5] P. Zhao, W. Chen, Z. Feng, Y. Liu, P. Liu, Y. Xie4, and D.-G. Yu, "Electrospun nanofibers for periodontal treatment: A recent progress," *Progress in Polymer Science*, pp. 4137 – 4162, 2022.

[6] A. L. Yarim, B. Pourdeyhimi, and S. Ramakrishna, *Fundamentals and applications of Micro- and Nanofibers*. United Kingdom: Cambridge, 2014.

[7] L. Éva Uhljar and R. Ambrus, "Electrospinning of potential medical devices (wound dressings, tissue engineering scaffolds, face masks) and their regulatory approach," *Pharmaceutics*, pp. 1 – 27, January 2023.

[8] J. Li and A. Smith, "Electrospun nanofibers for biomedical applications," *Journal of Biomaterials Science*, vol. 42, no. 3, pp. 123–135, 2018.

[9] Q. Zhang and B. Wang, "Wound healing scaffolds utilizing electrospun fibers," *Advanced Materials*, vol. 28, no. 7, pp. 456–468, 2019.

[10] X. Yuan and Y. Chen, "Nanofibrous membranes for filtration applications," *Journal of Environmental Science*, vol. 35, no. 5, pp. 789–801, 2020.

[11] H. Zhang and K. Liu, "Nanofiber-based fabrics in textile industry," *Textile Research Journal*, vol. 25, no. 4, pp. 210–223, 2017.

[12] A. Ern and J.-L. Guermond, *Finite Elements I: Approximation and interpolation*. Springer, 2021.

[13] P. Ciarlet and Éric Lunéville, *La méthode des éléments finis, De la théorie à la pratique*. Paris: ENSTA ParisTech, 2009.

[14] "Finite element basis functions." "https://hplgit.github.io/num-methods-for-PDEs/doc/pub/approx/html/approx.html".

[15] A. Logg, K.-A. Mardal, and G. N. Wells, *Automated Solution of Differential Equations by the Finite Element Method*. Oslo and Cambridge, 2011.

[16] N. Shah, H. Kanchwala, and B. Bhattacharya, "Development of a novel viscoelastic nanocomposite and investigation of its damping capacity for large frequency band," *ICOVP, 13th International Conference on Vibration Problems*.

[17] A. S. Vinesh H. Gada, "On derivation and physical interpretation of level set method–based equations for two-phase flow simulations," *Numerical Heat Transfer*, vol. 2, p. 309, 2009.

[18] D. Lindbo, "Finite element computations for a conservative level set method applied to two-phase stokes flow," Master's thesis, Royal Institute of Technology, 2006.

[19] T. H. A. Brooks, "On derivation and physical interpretation of level set method–based equations for two-phase flow simulations," *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, pp. 199–259, 1982.

[20] G. Tierra and F. Guillen-González, "Numerical methods for solving the cahn-hilliard equation and its applicability to related energy-based models," *Deposito de investigación Universidad de Sevilla*, 2013.

[21] D. A. Saville, "Electrohydrodynamics: The taylor-melcher leaky dielectric model," *Annual Review of Fluid Mechanics*, vol. 29, no. 1, pp. 27–64, 1997.

[22] P. Chen, Q. Zhou, G. Chen, Y. Wang, and J. Lv, "Numerical simulation and experimental research of electrospun polyacrylonitrile taylor cone based on multiphysics coupling," *e-Polymers*, January 2023.

[23] S. Rafiei, B. Noroozi, L. Heltai, and A. K. Haghi, "An authenticated theoretical modeling of electrified fluid jet in core-shell nanofibers production," *Journal of Industrial Textiles*, p. 1791 – 1811, 2018.

[24] S. Riva, "Computation of electrostatic forces by the virtual work method," *Journal of the Serbian Society for Computational Mechanics*, pp. 1–15, 2007.

[25] S. Popinet, "Numerical models of surface tension," *Annu. Rev. Fluid Mech*, pp. 1–28, 2018.

[26] "Fem on colab." "https://fem-on-colab.github.io/".

[27] G. P. Bernal, "Codes for electrospinning simulation," 2023. "https://github.com/gperezb12/Electrospinning".

[28] 2020. https://glisbronzi.it/structured-vs-unstructured-mesh-k.html.