Towards a two dimensional model of surface piezoelectricity

Author: Oscar Monge Villora
Supervisor: Irene Arias Vicente
External Supervisor: Amir Abdollahi

Escola Superior d'Enginyeria de Telecomunicació
Universitat Politècnica de Catalunya
Barcelona, June 2015
Contents

Abstract 1

1. Introduction 3

1.1 Introduction to flexoelectricity............................ 3
1.2 A little of history.................................................... 4
1.3 Presentation of our work and objectives............. 6

2. Theory 8

2.1 Continuum model of flexoelectricity theory........ 8
  2.1.1 Phenomenological theory................................. 8
  2.1.2 Experimental study........................................ 14
  2.1.3 Applications .................................................. 16
2.2 Background of finite elements......................... 19
  2.2.1 Galerkin formulation in 1D scalar problem........... 19
  2.2.2 Galerkin formulation in 2D-3D vectorial problem 26
2.3 Background of B-Splines..................................... 31
  2.3.1 Definition and properties of B-Splines basis
        functions................................................................ 31
  2.3.2 Derivatives of B-Splines Basis functions............. 32
  2.3.3 B-Spline curves............................................... 34
  2.3.4 B-Spline surfaces.......................................... 36

3. Simulation method 38

3.1 Theoretical development of our model.............. 38
3.2 Numerical methodology of the model............... 46

4. Results that validates the model 55
  4.1 The size effect of a cantilever beam beanding..... 55
  4.2 An additional model of surface piezoelectricity.... 57

5. Conclusions and future work 59

6. Appendix 63
ABSTRACT

Nowadays, it's necessary to understand the electromechanical conversion materials in order to develop the next-generation electromechanical transducers. The first materials that comes to mind when we listen the word "electromechanical" are the piezoelectric materials. Piezoelectricity is revealed on materials that become electrically polarized when are subjected to a mechanical stress: the ferroelectric materials, that are non-centrosymmetric. We take advantage of the piezoelectric effect in a lot of applications, but basically in the production of piezoelectric sensors, actuators and energy harvesters. These can be found from crash sensors in cars, to activate the airbags when we suffer a collision, until applications as a loudspeaker or a switch to turn on the light. In the present work we focus on a related but far less understood electromechanical coupling, namely flexoelectricity.

The flexoelectric effect is based on a strain gradient that generates a polarization. It's an universal effect because it can be produced in any dielectric material in contrast to piezoelectricity which is only possible in a specific class of materials. As a consequence, we can realise the potential importance of flexoelectricity in order to develop new applications in areas like nanomaterials, soft biomaterials, biocompatible sensors and actuators, to name a few.

However, there exist some controversies and difficulties that prevent to progress in this field. Any dielectric material at small scales can produce the flexoelectric effect even centrosymmetric materials. It's the external perturbation, the strain gradient, that breaks the symmetry of the sample (microscopically) and as a consequence induce a polarization. However, any flexoelectric sample is finite, so despite we are treating with a centrosymmetric material, at the surface we break the symmetry. This produces the appearance of the surface piezoelectricity, that consist in inducing a polarization due to the intrinsic symmetry loss of the material. Therefore, we have two electric contributions and it's very complicate to distinguish them.

This controversies motivate this work and in order to contribute to the investigation on this field, we propose to model surface piezoelectricity through a zero-thickness effective piezoelectric layer in a simple rectangular geometry (2 dimensions). The response of this model can be compared against that of a flexoelectric sample to extract conclusions about the physics of both phenomena. As a first step towards this goal, in this work we implement a rich continuum model based on strain gradient elasticity. Since standard \( C^0 \) finite elements are not applicable to the high-order partial differential equations governing strain-gradient elasticity, we use here a Galerkin method with smooth shape functions based on B-splines. Although we have only implemented the mechanical part in this work, our code can be easily extended to deal with the electromechanical problem and in particular with the high-order partial differential equations of flexoelectricity in simple geometries. For validation purposes, we check that we are able to describe the well-known size effect in strain gradient elasticity theories for the case of a cantilever beam under a point force.
1. INTRODUCTION

1.1 Introduction to flexoelectricity

Electromechanical conversion materials have been very important for the progress of the technology. Understanding them is necessary to develop the next-generation electromechanical transducers. When we think on a type of material that exhibit a no-weakness electromechanical coupling nature, the first thing that comes to mind is a piezoelectric material. Piezoelectricity is revealed on materials that become electrically polarized when are subjected to a mechanical stress. They exhibit the converse effect too, in which they experience a strain in response to an applied electric field. [1] Although we are more familiar with piezoelectricity, our work treats with another electromechanical effect, the flexoelectricity, that in the last decades has caused an increasing interest of the scientific community. While piezoelectricity is only notable in crystalline structures with no inversion symmetry, that delimites a lot the type of material used, flexoelectricity does not carry this inconvenient, because can be revealed in any material, it's an universal property. This is one of some advantages respect piezoelectricity that will be announced. But, first of all, what's flexoelectricity?

Flexoelectricity is the polarization response of an insulating material to a strain gradient. When we talk about strain gradient we are referring to a gradient of elastical deformation that can be represented by a bending. Therefore, the flexoelectric effect is the coupling between polarization and strain gradient. This is the main difference with piezoelectricity, which couples the polarization with an homogenius strain (deformation). Piezoelectricity is an effect that is exhibited when we apply an homogenous strain on a non-centrosymmetric material. Since piezoelectrics lack inversion symmetry, deformations can generate internal polarization due to noncompensation of ionic charges. The most part of piezoelectrics are ferroelectrics materials, i.e., materials that have a spontaneous electrical polarization that can be reversed by the application of an external electric field.

If we apply an homogenius strain on a centrosymmetric material it won't induce any polarization because this material will continue being centrosymmetric and it won't be any preferred direction of induced polarization (figure 1) [2]. For this reason, the symmetric materials can't be piezoelectrics.

![Figure 1](image-url)  
**Figure 1.** Cartoon illustrating the microscopic mechanism of flexoelectricity. 
(a) Uniform strain. (b) Gradient strain.
By contrast, in flexoelectricity a strain gradient does break centrosymmetry: applying it on a symmetric plate, their top and bottom surfaces differs in longitude and this fact define a sense of direction for the induced polarization vector (figure 1b). Microscopically, considering the ionic lattice of this figure 1b, a vertical gradient may cause the central cation to be squeezed up breaking the local centrosymmetry and inducing polarity. Therefore, the main idea is that in piezoelectrics, the polarization thus occurs due to the low symmetry of the material rather than due to the symmetry-breaking effect of the perturbation that occurs in flexoelectrics materials.

Another remarkable difference between piezoelectricity and flexoelectricity is that the first one is revealed at an ordinary scales of the material dimensions while the second effect is more important in reduced dimensions. As more reduced, we obtain larger gradients. So, flexoelectricity is very connected with nanotechnology, a growing science that has very impact in the scientific interest. Therefore, this characteristic can be considered a stimulus to study flexoelectricity, as important as the universality property of appearing in any material that we have commented before. Curiously, despite this universality, the materials that exhibit a better flexoelectric behaviour are these which are piezoelectric, the ferroelectrics. Mathematically, the flexoelectricity can be viewed as a high-order electromechanical phenomenon with respect to the piezoelectric effect, which is a linear response. Another thing to comment is that the flexoelectric effect is controlled by a fourth-rank tensor while the piezoelectric is controlled by a third-rank tensor.

Also, we can realise the potential importance of flexoelectricity in order to develop new applications in scientific areas like nanomaterial, soft biomaterials, etc. Some of these applications are strain gradient engineering, mechanical polarization switching and nanodevices that will talk about in our work.

1.2 A little of history [3]

Once we know what's the flexoelectricity, the reasons that justify its importance to study it and which are the differences with piezoelectricity, we have to continue with a little of history of this "new" effect. The flexoelectric effect in solids was first identified theoretically by Maskevich and Tolpygo in 1960, based on their study on lattice dynamics in crystals. It was be Kogan at 1964, who did the first phenomenological framework of this effect, in the context of electron-phonon coupling in centrosymmetric crystals. The first microscopic calculations of the coefficients involved in the control of the flexoelectricity response was made by Askar in 1970 for different materials. However, all the information recollected about this phenomenon in solids was insufficient. This same year, Bursian E.V., and his coworkers adress flexoelectricity in ferroelectrics. They characterized flexoelectricity in the classical ferroelectric BaTiO3 and demonstrated switching of spontaneous polarization driven by a strain gradient, an application that opened new trend lines of research. Also, they develop a phenomenological theory of the flexoelectric effect in a finite plate of a ferroelectric
and realised the linear relation between the relative permittivity with the flexoelectric coefficients. All this is referred to the theory research of flexoelectricity. In the experimental progress, Axe J.D. identify an important manifestation of the flexoelectric effect based on the analysis of the phonon spectra. However, until 1980, people though that it was a non-local piezoelectricity and the word “flexoelectricity” didn’t appear in any paper. In this year, Tagantsetv demonstrate using the phenomenological and microscopic approaches that this effect is more complicated and there are non-trivial dynamic and surface contributions to the flexoelectric response, having no analogs in piezoelectricity. Early 2000s, the flexoelectric response was characterized in a number of ferroelectric ceramics and single crystals. These experimental studies motivate the interest of theorist.

First of all, the theorists began to study the microscopic theories using the framework that Tagantsetv made, enabling the calculations of the flexoelectric coefficients. The results in this microscopic area were called first-principle calculations and the concept behind them had been defined by Martin and Resta in 1972. So, Hong and Vanderbit obtain the first-principle calculations of the purely electronic contribution in a number of crystals, including perovskites. Secondly, it was studied the flexoelectricity in a finite sample. A phenomenological approach of this was made by Tagantsetv and Yurkov and they remark the importance of the surface effects in the flexoelectric behaviour. In 2007, the researcher group of Eric Cross composite materials made of non-piezoelectrics but exhibiting the macroscopic piezoelectric response (due to flexoelectricity and local strain gradients). The effective piezoelectric coefficient were comparable with those of commercial piezoelectrics. All this progress was traduced into a fast growth of the publications about this topic.

![Figure 2](image)

*Figure 2. The number of publication on flexoelectricity per year from database “webknowledge.com”*

Nowadays, there are made a lot of experiments on flexoelectricity-driven phenomena. Much of them are focused on the key feature of the flexoelectric effect, that consists in that a strain gradient may work as an electric field, inducing poling, switching and rotation of polarization. Also, it can create a voltage offset of hysteresis loops.
1.3 Presentation of our work and objectives

The flexoelectricity is a modern research field that has still some controversies or unresolved issues to confront. One of them is that some part of the community thinks that in a finite sample the flexoelectric effect is due to the surface piezoelectric contribution and the bulk contribution while the other part believes there is only surface piezoelectric contribution. This controversy maybe has born in the difficulty of separate these contributions on the experimental results. Another controversy we can find is the fact that the theoretical values differ from these obtained experimentally due for instance to finite temperature effects or to finite sample effects, as the piezoelectric surface contribution before mentioned. The experimental values greatly exceed the theoreticals. This discrepancy has lead to a controversy in the field of flexoelectricity, demanding further investigations into the quantification of the flexoelectric response.

When we talk about the theoretical values, we are referring to the values calculated with the microscopic theory of flexoelectricity, also called the first-principles calculations. This theory is based on a density-functional perturbation theory. It consists in derive the complete flexoelectric tensor in terms of the microscopic linear response to atomic displacements. On the other hand, the experimental values are based on experimental set-up's to reproduce the flexoelectric effect. In order to translate an experimental measurement into a material parameter estimation, one must resort to a model. The model more used for flexoelectricity is the continuum theory of flexoelectricity and it will be explained in the next chapter.

We want to understand the behaviour of flexoelectricity and surface piezoelectricity and distinguish them in order to go deep into the controversies. This motivate the construction of a model of surface piezoelectricity. The model proposed is a two-dimensional model that integrates the electromechanical equations that include the elastic, dielectric, piezoelectric and flexoelectric effect on a rectangular sample. As the flexoelectric and the surface piezoelectric effects appear on thin films (at the nanoscale) it requires a rich continuum model based on strain gradient elasticity too (we have to consider a second order effect). The high-order partial differential equations of electromechanics can't be solved with standard $C^0$ finite elements because of the second order effects included. Therefore, our method will consist in using a galerkin method with smooth shape functions based on B-Splines.

As a first step to understand the surface piezoelectricity we implement the mechanical part of our model, i.e., considering the elastic effect term and the gradient elasticity term. In order to do that we encode from zero the numerical method using MATLAB. We arrive to a very important result about the elasctical behaviour of a beam (sample): the size effect of cantilever beam beanding. This result, that is known, will validates the development of our flexoelectricity model and the correct realization of the numerical method involved.
In conclusion, the objective of my work is to learn how to make a surface piezoelectricity model, that requires learning and having assimilated some knowledge about continuum flexoelectricity theory, the background theory of finite element methods and the background of B-splines. In order to accomplish that, we have made the numerical code necessary to carry out this model, solving for the mechanical part of the system and obtaining an important result, the size effect of cantilever beam beanding, that can be used as a final validation of our code and, thus, it validates the flexoelectric model that will explain. Furthermore, we have analysed some flexoelectric results using another numerical method (the local maximum-entropy (LME) approximants) to solve the electromechanical equations.
2. Theory

A fundamental challenge is separating the bulk flexoelectric response from surface piezoelectric contributions. In order to do that we have to study the continuum model of flexoelectricity:

2.1 Continuum model of flexoelectricity theory

2.1.1 Phenomenological theory of flexoelectricity[2]

We can separate the study of the flexoelectricity effects in the continuum model depending on whether they are produced at the bulk of the material, at the surface (finite samples) or appear in thin films (ferroelectric material). These classification can be visualized in the sequent diagram and they will be explained in the next sections:

![Diagram of type of flexoelectric effects.](image-url)
2.1.1.1 The bulk flexoelectric response

A) Constitutive equations of the static response

We are looking for the equations that describe the flexoelectric response on the bulk. To simplify we only treat the one-dimensional case. We consider a thermodynamic potential density (1D):

\[ \Phi_G = \frac{1}{2} c P^2 + \frac{c}{2} u^2 - \phi P u - f_1 P \frac{\partial u}{\partial x} - f_2 u \frac{\partial P}{\partial x} - PE - u \sigma, \]  

(1)

that is a generalization of the thermodynamic potential density of the description of the piezoelectric response, now including the flexoelectric couple terms that describes the bulk flexoelectric response (fourth and fifth terms). The third term represents the linear piezoelectric response. The main variables that we can distinguish in this expression are: \( P \), the polarization of the material; \( u \), the strain; \( \sigma \), the stress; \( E \), the electric field and \( X \), the electric susceptibility (\( c, v, f_1 \) and \( f_2 \) are some constants). As we know, the strain is a measurement of deformation that indicates the displacement between particles in the body relative to a reference length and the stress is the force per unit area on a body that provokes a change of shape.

\[ u = \frac{d l}{l_0}, \quad \sigma = \frac{F}{A} \]  

(2)

Continuing with the thermodynamic potential, we minimize the expression respect \( P \) and \( u \), first, without considering the flexoelectric terms. We arrive to the piezoelectric fundamental equations:

\[ P = \chi E + e u, \]
\[ \sigma = -e E + c E u, \]  

(3)

where \( e = \chi v \) is the strain-charge piezoelectric coefficient and \( c^E = c - \chi v^2 \) is the elastic constant at constant electric field. The first term of the first equation represents the linear term that varies with the electric field, considered in a linear, homogenous and isotropic medium. The second term represents the piezoelectric effect in which a strain induces a polarization. Examining the second equation, we understand the stress is induced by a converse piezoelectric effect (the first term) and by the Hook’s law term (the second one). It is called converse piezoelectric effect.

Figure 4. Cartoon illustrating the different types of stresses. Tensile, compressive and shear stresses, respectively.
because we understand the electric field on the material produces an stress that induces a deformation, so, it's the opposite to the piezoelectric effect, in which, the mechanical energy of a deformation induces an electric field.

Once we familiarized with these equations, we neglect now the piezoelectric term in equation (1) imposing $v=0$ and we rewrite it calling $f = f_1 - f_2$ the flexocoupling coefficient. We present the potential $\Phi_G$ as the sum of two contributions:

$$
\Phi_G = \Phi - \frac{f_1 + f_2}{2} \left( \frac{\partial u P}{\partial x} \right),
$$

$$
\Phi = \frac{1}{2} \chi P^2 + \frac{e}{2} u^2 - \frac{f}{2} \left( P \frac{\partial u}{\partial x} - u \frac{\partial P}{\partial x} \right) - PE - u\sigma.
$$

The first contribution of the potential represents the bulk potential while the second term, is the superficial potential. Integrating $\int \Phi_G dV$ and applying Euler's equations we obtain the constitutive equations of flexoelectricity:

$$
P = \chi E + \mu \frac{\partial u}{\partial x},
$$

$$
\sigma = e u + \frac{\mu}{\chi} \frac{\partial P}{\partial x},
$$

$$
\mu = \chi f.
$$

As a difference with the piezoelectric response, in which the polarization is caused linearly by the strain, in the flexoelectric response, the polarization is induced by a gradient of the strain. We realize that the flexoelectric response and the converse flexoelectric response, this last called to the phenomenon in which a gradient of the polarization induce a stress, are proportional to the flexoelectric coefficient $\mu$. This coefficient in three dimensions can be defined as a fourth-rank tensor with the next formula (obvious to obtain it using the first one of the constitutive equations):

$$
\mu_{ijkl} = \left( \frac{\partial P_j}{\partial (\partial u_{kl}/\partial x_i)} \right)_{E=0}
$$

Another important observation is that the flexoelectric coefficient $\mu$ is proportional to the relative permittivity (also called dielectric constant) because $\chi_e = \epsilon_r - 1$ and we ignore the unity for high number of $\epsilon_r$.

We can simplify the second one of the constitutive equations considering that the flexoelectric effect is relatively weak, introducing $P = \epsilon_e \chi E$ in it. Doing this, we
In these equations we realize again about the direct and the converse flexoelectric effect and comparing with the equations of piezoelectricity we differenciate now an asymmetry in which a gradient of strain generates a constant electric field but we need a non uniform electric field to generate the mechanical stress.

B) Constitutive equations of the dynamic response

Now, in this case, we reproduce the motion of an acoustic wave. To do that we have to consider the kinetic energy density of the problem that is equal to one half times the mass density times the square of the velocity (first term of $T$) and we add it in the action we will minimize using the Euler-Lagrange equations:

$$
\int\int (T - \Phi + u\sigma) d\Omega dt \quad \text{with} \quad T = \frac{1}{2} \rho \dot{U}^2 + M\dot{P}
$$

where $U$ is the acoustic displacement, $\rho$ is the density of the material, $P$ is the polarization, $M$ is not the mass but a constant and the dot represents the time derivative. So, we consider the temporal dependence of $P$ and $U$. Then, minimizing the action we arrive to the equations of the left:

$$
P = \chi E + \mu \frac{\partial u}{\partial x}, \quad \rho \dot{U} = \varepsilon \frac{\partial u}{\partial x} - M\dot{P} + \frac{\mu}{\chi} \frac{\partial^2 P}{\partial x^2}.
$$

The first equation is so similar to the static case but now we have a new term, the last, that represents the dynamic flexoelectric effect, which is the polarization due to the acceleration of the medium. The second equation is not so similar to the case of before. The two last terms are involved in lattice dynamics, controlling the shape of the dispersion curve of phonons ("vibrations of the matter"). Considering macroscopic situations, we can neglect these second and third term and substituing $\dot{U}$ in the first equation we obtain the equations of the right. These are simplified and we can distinguish clearly the static contribution that contains the static flexoelectric coefficient $\mu$ and the dynamic contribution that contains the dynamic flexoelectric coefficient $\mu_d$ and all too are linear response to the spatial derivative of the strain.

Also, it's important to know that the dynamic flexoelectric coefficient depends on the

$$
P = \chi E + (\mu + \mu_d) \frac{\partial u}{\partial x},
$$

$$
\mu_d = -\varepsilon \chi M/\rho
$$
susceptibility like the static coefficient and they have the same magnitude order. They
don't depend on the wave frequency unlike the strain gradient, that depends on it. In
quasi-static regime (f_{wave} < f_{material}) we can neglect the dynamic contribution. We
can't forget that "\chi", "c", and "f" and the flexoelectric coefficients are tensors in the
three-dimensional case. "f" and the flexoelectric coefficients are fourth rank tensors
symmetric with respect to the permutation of the first pair of suffixes.

2.1.1.2 Flexoelectricity in finite samples

A) Surface piezoelectricity

In centrosymmetric materials, due to symmetry-breaking impact of the surface, a thin
surface layer of thickness \( \lambda \) generally becomes piezoelectric. We consider a finite
sample connected into an electric circuit, in which we apply a gradient of strain. We
can distinguish two zones, the surface zone and the bulk. Due to surface piezoelectric
effect, it is generated a polarization \( P_\lambda \) in the direction that the figure indicates. The
normal component of the electric displacement

\[
D_z = P_\lambda + \varepsilon_0 E_\lambda = P_b + \varepsilon_0 E_b
\]

must be conserved, so

So the polarization of the surface induces some internal electrical fields in the sample, polarizing it.

We can obtain an effective formula for the electric displacement:

\[
D = \varepsilon \lambda, b \varepsilon_b \frac{\partial \mu_{11}}{2 \varepsilon_b + b \varepsilon_\lambda} \partial \chi_3
\]

and considering thin-enough surface layers

\( \lambda/b \ll \varepsilon_\lambda/\varepsilon_b \)

we obtain an effective flexoelectric coefficient:

\[
\mu_{1113}^{\text{eff}} = \varepsilon \lambda \frac{\varepsilon_b}{\varepsilon_\lambda}
\]

The constants \( \lambda, \varepsilon_\lambda, h \) and \( E_r \) are the thickness and the relative permittivity of the
surface layer and the bulk respectively. This flexoelectric coefficient don't depend on
the surface/volume ratio but in the permittivity ratio. We do the next estimation of the
flexocouple coefficient:

\[
f_{\text{eff}} = \mu_{\text{eff}}/\chi_b \approx \varepsilon \lambda/\varepsilon_\lambda
\]

Figure 5. Surface piezoelectricity. Upon bending, as shown on the left, the tensile/compressive strains in
the top/bottom surface layers give rise to a polarization \( P \lambda \) in the piezoelectric surface layers of
thickness \( \lambda \).
Putting some values we obtain this coefficient is a few volts, like the flexoelectric coefficient involved in bulk flexoelectricity. So we can conclude that surface piezoelectricity compete with bulk flexoelectricity.

**B) Polarization-Induced Bending**

Some researchers show that it isn’t achieve the asymmetric condition seen some pages before: a material will be bend when we apply an uniform electric field. The reason is a local effect that predicts the thermodynamics and it can be understood knowing that at the surface we have a polarization gradient. This polarization gradient appears because of the polarization have to decrease since a value $P$ in the material until a zero value at the air. Using the equation know of the stress originated by a gradient of polarization we predict some forces appeare applied to the opposite surfaces of the plate. By measuring the curvature $\zeta$ induced by the electric field $E$, one can also get an estimate of the flexoelectric coefficient, which is given by:

$$\tilde{\mu}_{12} = \frac{\zeta}{E} \cdot \frac{Gd^2}{12(1 - v^2)}$$  \hspace{1cm} (14)

where $G$ is the Young’s modulus, $d$ is the crystal thickness, and $v$ is the Poisson ratio.

**2.1.1.3 Flexoelectricity on thin films**

**A) Surface flexoelectricity**

Most of the reported effects of flexoelectricity on thin films have involved ferroelectric materials. The reason is the flexoelectric coefficients are high due to the susceptibilities are high too and we know $\mu \equiv \chi f$.

Stresses between the substrate and thin films are caused by having diferent “a” (lattice constants or different thermal coefficients and the result is a substrate-induced strain that will be homogenous for films sufficiently thins but if we exceed a minimum value we will have some relaxation mechanisms of stress like dislocations and twinning that are inhomogeneous effects. As a consequence, the mismatch strain will be relax exponentially through the film. This is called exponential strain relaxation. This phenomen was considered the origin of the strain gradient. So, finally we obtain surface gradient-induced flexoelectricity.
B) Flexoelectric poling

In ferroelectrics, strain gradients can lead to asymmetric polarization-field hysteresis loops due to the presence of an internal field. This effect is expected to be particularly appreciable in ferroelectric thin films in which the misfit strain due to the underlying substrate is relaxed though the formation of dislocations, giving rise to large strain gradients. The effective electric field due to this gradients lead to the reorientation of defect dipoles, that results in a shift in the hysteresis loop. Another interesting thing is that we can use the strain gradients to switch the polarization of a ferroelectric material.

2.1.2 Experimental study of flexoelectricity

The static flexoelectric response is generally measured by two methods: using a cantilever or using a truncated pyramid-shaped sample. The objective of the experimental study is to obtain the flexoelectric tensor $\mu$.

A) The cantilever bending method

This method consist in dynamically bending a material in a cantilever beam to generate a strain gradient like is shown in the picture. We connect the cantilever to an electrical circuit in order to measure the electrical response of the strain gradient. We can measure the flexoelectric polarization measuring the displacement current between the two plates. Then, knowing the strain gradient produced we can calculate the effective transverse flexoelectric coefficient $\tilde{\mu}_{12}$ through the next formula:

$$P_f = \tilde{\mu}_{12} \frac{\partial u_{11}}{\partial x_3}$$

(15)

Therefore, we have a longitudinal strain that varies along the $x_3$ axis generating the strain gradient in this direction. The section of the beam is modified by the anticlastic bending generating other directional strains $u_{22}$ and $u_{33}$ that are related to $u_{11}$ by the Poisson ratio $v$. We obtain the next relation between flexoelectric coefficient:

$$\tilde{\mu}_{12} = -v \mu_{11} + (1 - v)\mu_{12}$$

(16)
B) The pyramid compression method

This method is based on the uniaxial compression of a truncated pyramid-shaped sample. We apply the same force at both surfaces of the pyramid. Due the fact that one of them have more area than the other, the stress \( \sigma_{33} = \frac{F}{A} \) will be lower in that one and this will generate a strain gradient that will cause a flexoelectric effect. In this case, the strain gradient has the same direction than the strain and using the following formula we can extract the effective flexoelectric coefficient \( \tilde{\mu}_{11} \).

\[
P_f = \tilde{\mu}_{11} \frac{\partial u_{33}}{\partial \chi_3} \quad \text{with the next relation:} \quad \tilde{\mu}_{11} = \mu_{11} - 2\nu \mu_{12}
\]  

(17)

C) Other methods and experimental aspects

There exist other methods that allow us to obtain an experimental result for the flexoelectric coefficient. For example, we can generate a non uniform \( E \) or gradient of polarization and this will generate strain taking advantage of the converse flexoelectric effect, which can be measured using some interferometric techniques. In this method it is important to separate the strain generated by the converse flexoelectric effect with this generated by electrostriction, a dielectric effect of changing the shape under the effect of an electrical field in a quadratic form.

It is important to remark that there are experimental measurements of the flexoelectric response as a function of the temperature that confirm the expected scaling of \( \mu \) with \( \chi \) (\( \mu \equiv \chi f \)).

Quantify the flexoelectric tensor has ever been challenging. We use some of the experimental methods before explained to measure the flexocoupling coefficient of perovskite ceramics (common example, as \( \text{BaTiO}_3 \), with high flexoelectric coefficients). We notice the results, that are about 20-800 V, greatly exceed the theoretical values calculated by Kogan with the formula:

\[
f' \approx \frac{q}{4\pi \varepsilon_0 a} = 1-10 \text{ V}
\]  

(17)

If we analyze single crystals we see the values of \( f \) adjust better to the theoretical values. Therefore, we notice there exist very large discrepancies between the experimental data of ceramic materials and single crystals and equally large between
the theoretical values obtained by using different techniques. The challenge of quantify correctly the flexoelectricity depends on distinguish the surface piezoelectric contribution and dynamic flexoelectricity of the bulk static flexoelectricity. Understanding this is the key to can solve the meaning of this discrepancies.

2.1.3 Applications of flexoelectricity

The main important applications that used a flexoelectric effect are mechanical polarization switching, piezoelectric metamaterials and nanodevices and strain gradient engineering.

2.1.3.1 Mechanical polarization and switching

Flexoelectricity can acts like an external electric field enabling the control and switching the material polarization. As we have mention before, in 1969 Burssian and his coworkers showed that the bending of a few-micrometers-thick plate can result in the reversal of the sign of its pyroelectric coefficient: it was the beginning of the this research. The flexoelectric switching of polarization has a useful advantage in that we don't need to apply larger electric fields in order to get a higher intensity and this can avoid problemas as leakage and breakdown. At the next image it is possible to see how an atomic force microscope generate a strain gradient in a specific part of the layer changing the polarization.

Figure 9. Highly concentrated stress fields under an atomic force microscope tip pressing on the sample surface produce strain gradients that are equivalent to an electric field sufficient to reverse the polarization of a ferroelectric thin film.

2.1.3.2 Piezoelectric metamaterials and nanodevices

The idea consists in utilise a nanocomposite with built-in shape gradients to take advantage of the flexoelectric effect in order to produce an effective piezoelectric
response, despite its components aren't piezoelectrics. An example of this, made by Cross and coworkers in 2007 consists in a device made of an array of truncated pyramids with high flexoelectric coefficient set in another medium as air, all these placed between two metallic plates. When we compress the plates, we generate a strain gradient in each pyramid, giving us an effective piezoelectric response. This nanodevice can be visualized in the figure 30a. Another device we can explain, is studied by the Sharma group. It consists in a nonpiezoelectric elastic matrix with nanoscales inhomogeneities, called nanoinclusions, that allow us to generate local strain gradients. For the purpose of generate a flexoelectric effect, the nanoinclusions distribution must be in a non-centrosymmetric way. In this way, we avoid to cancel the local polarizations induced. Finishing with the examples, a nanodevice studied too consists in a piezoelectric composite that collects the charge generated by bending the flexoelectric material throughout several metallic strips located at positions where the strain gradient is maximum. This nanodevice can be visualized in the figure 30b.

The enhancement in flexoelectric nanodevices have stimulated the theoretical work on these dispositives. Group of reseachers as the Sharma group, go deep into this topic and has done some atomistic calculations about flexoelectricity in different nanostructures as nanocantilevers, nanocomposites and superlattices. Other groups work with the nanodevices before explained changing some configurational aspects or they work with another nanostructures as nanowires, buckled nanoribbons, etc.

2.1.3.1 Strain gradient engineering

It consists in being able to generate strain gradients using some sophisticated techniques. For this purpose, we use epitaxial strain, that allow us to modify the properties of thin films; it is possible to modify the critical temperature of some superconductors or converting a material as SrTiO₃ into a room temperature ferroelectric, as an examples. When we talk about expitaxial strain we refer to the strain generated in a expitaxial film, i.e., a crystalline thin film deposited on a crystalline substrate.
There are some possible mechanisms to achieve macroscopic piezoelectricity, breaking the inversion symmetry. In order to do that we can put a third component layer to form the so-called tricolor superlattices, for example, or having a net compositional gradient between the top and bottom layers. So the effective piezoelectric response is conditioned by the action of the flexoelectric effect and the inhomogeneity of the local polarization induced and strain. Another way to create a strain gradient is based on growing films with ferroelastic twins, that are domains with alternate directions of spontaneous polarization. In fact, twinning is a mechanism caused by strain relaxations in thin films, i.e., the substrate mismatch allow us to control the size and the orientation of the twins.
2.2 Background of finite elements

This section is referred to the book of the reference [4]. In order to develop the numerical method that simulates the electromechanical response of dielectric solids, that involucrate the flexoelectricity and piezoelectricity effects, we have to apply a galerkin method. This method is based on the finite element method to treat and discretize the problem with the only difference that it uses a smooth meshfree basis function, in our case, the B-splines basis function. Therefore, in order to learn about the Galerkin method we have to know the finite element method.

The main parts of a finite element method for the solution of a boundary-value problem are two:

a) The weak statement of the problem.

b) The approximate solution of the variational equations through the use of "finite element functions". In our case they will be the B-Splines basis functions.

Therefore, we will explain the Galerkin formulation with the finite element method point of view.

2.2.1 Galerkin formulation in 1D scalar problem

A) Strong and weak form
We begin with an example of problem in 1D with two point boundary-values.

\[
\begin{align*}
{u}_{xx} + l &= 0 \quad \text{On } \Omega \\
{u}(1) &= g \\
-u, x(0) &= h
\end{align*}
\]

Where \( l \) is a function that depends on \( x \), \( g \) and \( h \) are some known constants and \( u(x) \) is the solution we want to find. The differential equation is valid for all \( x \) living in the domain \( \Omega \). This presentation of the problem with the differential equation to solve and the two boundary conditions is called the strong form of the problem.

Since the strong form, we want to find the weak form, a different way to consider the problem. We want to remark the difference the strong form and the weak form. In a strong formulation for an approximate solution, we need to assume displacement functions that are 2nd order differentiable. In a weak formulation, we create linear and bilinear forms and then search for a particular function (an approximate solution) that satisfy the weak statement.

In order to do find the weak form, we define two collection of functions by the letters \( \delta \) and \( \nu \). The first one are the collection of trial solutions \( u \) that are \( H^1 \) functions,
i.e, square-integrable and have to satisfy the first boundary condition. The second collection of functions are the weighting functions, or variations that have to be square-integrable too and must satisfy the boundary condition \( w(1) = 0 \).

We proceed to obtain the weak form. First of all, we multiply the differential equation plus \( w \) and we integrate it.

\[
0 = \int_0^1 w(u_{xx} + l) \, dx
\]

Applying integration by parts we obtain:

\[
0 = \int_0^1 w_{,x} u_{,x} \, dx - \left[ w \cdot l \, dx - w u_{,x} \right]_0^1
\]

We substitute the boundary conditions and we finally have the weak form formulation of the problem:

\[
\left\{ \begin{array}{l}
\text{Given } l, g \text{ and } h, \text{ find } u \in \delta \text{ such that for all } w \in \mathcal{V} : \\
\int_0^1 w_{,x} u_{,x} \, dx = \int_0^1 w \cdot l \, dx + w(0) h
\end{array} \right.
\]  \tag{19}

It is common to express the equation of before as:

\[
a(w, u) = (w, l) + w(0) h
\]  \tag{20}

with

\[
a(w, u) = \int_0^1 w_{,x} u_{,x} \, dx
\]  \tag{21}

\[
(w, l) = \int_0^1 w \cdot l \, dx
\]  \tag{22}

where \( a(\cdot, \cdot) \) and \( (\cdot, \cdot) \) are examples of symmetric and bilinear forms. Now, we have to use the \textit{galerkin's approximation method}.

\textbf{B) The Galerkin's approximation method and the matrix equations}

The first step of the method consist in constructing finite-dimensional approximations of \( \delta \) and \( \mathcal{V} \), denoted by \( \delta^h \) and \( \mathcal{V}^a \), considered as subsets of the first
function spaces. The $h$ superscript refers to the association of the two subsets with a mesh or discretization of $\Omega$.

The solution $u \in \delta^h$ and $w^h \in \mathcal{V}^h$ have to satisfy the boundary condition too:

$$\begin{align*}
   u^h(1) &= g \\
   w^h(1) &= 0
\end{align*}$$

(23)

Now we continue with the second step of the Galerking method. It consists in construct a function that verify:

$$u^h = v^h + g^h$$

(24)

where $v^h \in \mathcal{V}^h$ and $g^h$ is a given function that satisfies the essential boundary condition $g^h(1) = g$. Thus, $u^h$ satisfies the requisite boundary condition too. The resulting equation of the weak form, now is in terms of $w^h$ and $u^h$:

$$a(w^h, u^h) = (w^h, l) + w^h(0) h$$

(25)

and applying the decomposition of $u^h$ we obtain the usually called Galerkin equation, where now the unknown we want to solve is $v^h$:

$$a(w^h, v^h) = (w^h, l) + w^h(0) h - a(w^h, g^h)$$

(26)

The third step of the Galerkin's method consist in go deep into the structure of the functions of $\mathcal{V}^h$. They are linear combinations of given function denoted by $N_A$ where $A = 1, 2, \ldots, n$; so the space is said to have dimension $n$. Therefore:

$$w^h = \sum_{A=1}^{n} c_A N_A = c_1 N_1 + c_2 N_2 + \ldots + c_n N_n$$

(27)

We require that each $N_A$ satisfies: $N_A(1) = 0$, $A = 1, 2, \ldots, n$. These $N_A$'s are called shape, basis or interpolation functions. We introduce another shape function with the property $N_{n+1}(1) = 1$ to specify $g^h$. So, we define:

$$g^h = g N_{n+1}$$

and as a consequence, $u^h = v^h + g^h = \sum_{A=1}^{n} d_A N_A + g N_{n+1}$

(28)

where $d_A$'s are the constants we want to find and having them we we'll have the solution of the problem. Now, we substitute the expressions of $g^h$, $v^h$ and $w^h$ in the Galerkin equation:
\[ a\left(\sum_{A=1}^{n} c_{A} N_{A}, \sum_{B=1}^{n} d_{B} N_{B}\right) = \left(\sum_{A=1}^{n} c_{A} N_{A}, l\right) + \left[\sum_{A=1}^{n} c_{A} N_{A}(0)\right] h - a\left(\sum_{A=1}^{n} c_{A} N_{A}, g N_{N+1}\right) \]

Using the bilinearity of \(a(\cdot,\cdot)\) and \(\langle \cdot, \cdot \rangle\) we organize this expression with the sequent form:

\[ 0 = \sum_{A=1}^{n} c_{A} G_{A} \text{ where } G_{A} = \sum_{B=1}^{n} a(N_{A}, N_{B}) d_{B} - (N_{A}, l) - N_{A}(0) h + a(N_{A}, N_{n+1}) g \]

Since the \(c_{A}\)'s are arbitrary because we want to get a solution that verifies the Galerkin equation for any \(w^{h}\), it follows that all \(G_{A}\) must be zero. This fact, drive us to the next system of \(n\) equations in \(n\) unknowns:

\[ \sum_{B=1}^{n} a(N_{A}, N_{B}) d_{B} = (N_{A}, l) + N_{A}(0) h - a(N_{A}, N_{n+1}) g \]  

We can write it in a more compact form:

\[ \sum_{B=1}^{n} K_{AB} d_{B} = F_{A} \quad \text{for } A=1,2,\ldots,n \]

These system can be expressed in a efficient way using a matrix notation. We define the stifness matrix, the force vector and the displacement vector, respectively:

\[ K = [K_{AB}] = \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1n} \\ K_{21} & K_{22} & \cdots & K_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ K_{n1} & K_{n2} & \cdots & K_{nn} \end{bmatrix} \quad d = \{d_{B}\} = \begin{bmatrix} d_{1} \\ d_{2} \\ \vdots \\ d_{n} \end{bmatrix} \quad F = \{F_{A}\} = \begin{bmatrix} F_{1} \\ F_{2} \\ \vdots \\ F_{n} \end{bmatrix} \]

Thus, the system is written:

\[ K d = F \]
Therefore we calculate the stiffness matrix and the force vector and doing the inverse of the first matrix, we obtain the displacement vector: \( d = K^{-1}F \). Once we have the \( d \)'s, we finally obtain the solution of our problem:

\[
\begin{align*}
\mathbf{u}^h(x) &= \sum_{A=1}^{n} \mathbf{d}_A N_A + \mathbf{g} N_{n+1} \\
&= f(x) + \mathbf{g} N_{n+1}
\end{align*}
\]

It is important to comment some details about this method. The first thing to comment are the properties of \( K \). The stiffness matrix is a symmetric, banded and positive-definite matrix. The symmetry comes from the operator \( a(\cdot,\cdot) \) and it is banded cause of \( K_{AB} \) are zero outside a neighborhood of node \( A \).

The second thing to comment is what type of shape functions we have to use. The common used in finite elements are associated to a piecewise linear finite element space and have the next expression:

\[
N_A(x) = \begin{cases} 
\frac{(x-x_{A-1})}{h_{A-1}}, & x_{A-1} \leq x \leq x_A \\
\frac{(x_A-x)}{h_A}, & x_A \leq x \leq x_{A+1} \\
0, & \text{elsewhere}
\end{cases}
\]

where the \( x_A \)s are called nodal points or nodes; the subintervals are called finite element domains or elements; and the \( h_A \)s are the lengths of the elements, defined as \( h_A = x_{A+1} - x_A \), that are not required to be equal. We can plot these shape functions:

![Figure 11. Basis functions for the piecewise linear finite element space.](image)

We notice about the local character of the shape function, satisfying this condition:

\[
N_A(x_B) = \delta_{AB}
\]

However, we aren’t going to work with these shape functions but with a special group of shape functions called B-splines that we will define in another chapter of this work. The reason of not apply them is because of the nature of the flexoelectric and
piezoelectric differential equations. In them, there appear some discontinuities that the piecewise linear function can't confront them in order to interpolate.

As we have commented before, the NA's have a local support, i.e., they are zero outside a certain neighborhood. The finite elements method take advantage of this observation and it allows to us to construct the K and F matrices of the matrix equations, constructing some matrices and vectors defined in a neighborhood (called elemental matrices and vectors) and summing the contributions of all of them. This procedure is used to do an efficient finite element computer program and sometimes is called the direct stiffness method and it will be explained in the next section.

C) Element Stiffness matrix and Force vector. Assembly

We want to compute a method of finite elements in an efficient way. In order to do that, we want to know about the element stiffness matrix and force vector.

First of all, we separate our model in 1D in \( n_e \) elements and we take \( e \) as the variable index for these elements, as is shown in the next picture:

![Figure 12. Element numbers (e)](image)

Now, we divide the integrals over \([0,1]\) that are used in order to construct the stiffness matrix and the force vector, in sums of integrals over the element domains, and as a consequence we have:

\[
K = \sum_{e=1}^{n_e} K^e, \quad K^e = [K^e_{AB}]
\]

\[
F = \sum_{e=1}^{n_e} F^e, \quad F^e = [F^e_{AB}]
\]

where

\[
K^e_{AB} = a (N_A, N_B)^e \int_{\Omega^e} N_{A,x} N_{B,x} dx
\]

\[
F^e_A = (N_A, l)^e + \delta_{e1} \delta_{A1} h - a (N_A, N_{n+1})^e \int_{\Omega^e} N_{A,x} N_{x_{n+1}} dx
\]

(34)

We are considering our N_a's are the piecewise linear shape functions, shown before. For this reason we observe that \( N_A(x_B) = \delta_{AB} \). \( \Omega^e = [x_1^e, x_2^e] \) is the domain of the eth element.

By the definitions of the N_a's for the piecewise linear finite element space (local support), we know:
Therefore, the elemental matrices and vectors in which we divide our stiffness matrix and force vector will have a lot of identically zero elements. The unique elements that will be not zero are shown in the picture:

![Diagram](image)

Figure 13. X's indicate nonzero terms; all others terms are zero.

Thinking a little, we understand the necessity of define for an efficient purpose, the **element stiffness matrix** \( k^e \) and **element force vector** \( f^e \) as the matrices and vectors that only contain the non-zero components:

\[
\begin{align*}
  k^e &= \begin{bmatrix} k_{ab}^e \end{bmatrix}, & f^e &= \begin{bmatrix} f_a^e \end{bmatrix} \\
  2 \times 2 & & 2 \times 1 \\
  k_{ab}^e &= a (N_a, N_b)^e = \int_{\Gamma^e} N_{a,x} N_{b,x} \, dx \\
  f_a^e &= \int_{\Gamma^e} N_a \ell \, dx + \begin{cases} \\
    \delta_{ai} k & e = 1 \\
    0 & e = 2, 3, \ldots, n_{el} - 1 \\
    -k_{a2}^e \varphi & e = n_{el} \\
  \end{cases}
\end{align*}
\]

Locally, in the previous expressions, the local nodes "a" and "b" can be 1 or 2. We want to construct directly the stiffness matrix \( K \) and the force vector \( F \) since the element stiffness matrix \( k^e \) and element force vector \( f^e \). This process is called **Assembly**. In order to do that we have to construct the LM array that for a given degree of freedom number ("a") and a given element number ("e"), and it returns the corresponding global equation number:
Once we construct the LM array the assembly is made by the next procedure, that only put the eth element matrix $k^e$ at the corresponding global position in the global stiffness matrix (equal for $f^e$).

$$K_{ee} \leftarrow K_{ee} + k_{11}^e$$
$$K_{e,e+1} \leftarrow K_{e,e+1} + k_{12}^e$$
$$K_{e+1,e} \leftarrow K_{e+1,e} + k_{21}^e$$
$$K_{e+1,e+1} \leftarrow K_{e+1,e+1} + k_{22}^e$$

We have to remark again, that this section is focused on doing a numerical finite element method and to do it is necessary all these definitions and procedures.

Finally, we want to extrapolate all this learned of finite elements method for scalar 1D problem to vectorial 2D or 3D problem. The reason is that our work is based on a two-dimensional vectorial boundary-value problem.

### 2.2.1 Galerkin formulation in 2D-3D vectorial problem

We will see one example of 2D-3D vectorial problem and we will proceed as the previous case but generalizing some aspects. In order to don’t repeat all thing we will be more concise.

A) Strong and weak form

The problem we will treat will be the classical linear electostastic problem. The statement of the strong form of the boundary-value problem is:

$$\begin{align*}
\sigma_{ij,j} + I_j &= 0 \quad \text{in} \quad \Omega \\
u_i &= g_i \quad \text{in} \quad \Gamma_{gi} \\
\sigma_{ij} n_j &= h_i \quad \text{in} \quad \Gamma_{hi}
\end{align*}$$

(37)

where $\sigma_{ij}$ is the stress tensor defined by the generalized Hooke's law as:

$$\sigma_{ij} = c_{ijkl} \varepsilon_{kl}$$

where the $c_{ijkl}$ are the elastic coefficients and $\varepsilon_{ij}$ is the strain tensor, defined to be the symmetric part of the displacement gradients, i.e.,
\[ \varepsilon_{ij} = u_{(i,j)} = \frac{u_{i,j} + u_{j,i}}{2} \]  

(38)

A part of it, the functions \( g_i \) and \( h_i \) are called the prescribed boundary displacements and tractions, respectively.

We proceed, as we do in scalar 1D problem, in order to find the weak form of the problem. We multiply the first Eq. (37) by the weight functions \( w_i \) and we integrate them:

\[ \int_{\Omega} w_i \sigma_{ij} \, d\Omega = - \int_{\Omega} w_i l_i \, d\Omega \]

Now, we use integration by parts in various dimensions that corresponds to the next rule:

\[ \int_{\Omega} f \, g \, d\Omega = - \int_{\Omega} f \, g \, d\Omega + \int_{\Gamma} f \, g \, n_i \, d\Omega \]  

(39)

So, applied in our case:

\[ - \int_{\Omega} w_i \sigma_{ij} \, d\Omega + \int_{\Gamma} \sigma_{ij} w_i n_j \, d\Gamma = - \int_{\Omega} w_i l_i \, d\Omega \]

We substitute \( w_{i,j} = w_{(i,j)} \), where the term of the right is defined as the symmetric part of the tensor. This equality can be proved; it's due the fact that \( w_{i,j} \) is a nonsymmetric tensor that multiply a symmetric tensor \( \sigma_{ij} \).

\[ - \int_{\Omega} w_{(i,j)} \sigma_{ij} \, d\Omega + \int_{\Gamma} \sigma_{ij} w_i n_j \, d\Gamma = - \int_{\Omega} w_i l_i \, d\Omega \]

After that, we replace the boundary condition \( \sigma_{ij} n_j = h_i \) and we obtain the statement of the weak form of our problem that is:

\[ \text{Given } l_i : \Omega \rightarrow \mathbb{R}, \quad g_i : \Gamma_{gl} \rightarrow \mathbb{R} \text{ and } h_i : \Gamma_{hi} \rightarrow \mathbb{R}, \text{ find } u_i \in \delta_i \text{ such that for all } w_i \in V_i, \]

\[ \int_{\Omega} w_{(i,j)} \sigma_{ij} \, d\Omega = \int_{\Omega} w_i l_i \, d\Omega + \sum_{i=1}^{n} \left( \int_{\Gamma_i} w_i h_i \, d\Gamma \right) \]

where \( \sigma_{ij} \) is defined in terms of \( u_i \)

(40)
We can write the weak form in the abstract notation:

\[ a(w, u) = (w, l) + (w, h) \]  \hspace{1cm} (41)

where \( a(w, u) = \int_{\Omega} w_{(i,j)} c_{ijkl} u_{(k,l)} \, d\Omega \) (we define it in terms of \( u_i \))

\[ (w, l) = \int_{\Omega} w_l \, d\Omega \] \hspace{0.5cm} and \hspace{0.5cm} \[ (w, h) = \sum_{i=1}^{n_d} \left( \int_{\Gamma_i} w_i \, h \, d\Gamma \right) \]

When we define the symmetric part of \( u \) and \( w \), i.e., \( w_{(i,j)} \) and \( u_{(k,l)} \) we are defining the next vectors:

\[ \varepsilon(u) = \begin{bmatrix} u_{1,1} \\ u_{2,2} \\ u_{1,2} + u_{2,1} \end{bmatrix} \quad \varepsilon(w) = \begin{bmatrix} w_{1,1} \\ w_{2,2} \\ w_{1,2} + w_{2,1} \end{bmatrix} \]  \hspace{1cm} (42)

Continuing since the weak form written in abstract notation, we apply the galerkin method (using finite dimensional approximations):

\[ u^h = v^h + g^h \]  \hspace{1cm} (43)

as we have done in the one dimensional case. Now the unknown variable is \( v^h \). We introduce the previous expressions on the weak form and we obtain the galerkin formulation of our problem:

\[ a(w^h, v^h) = (w^h, l) + (w^h, h) - a(w^h, g^h) \]  \hspace{1cm} (44)

We show the explicit representation of \( u^h, v^h \) and \( g^h \):

\[ v^h = v^h_i e_i, \quad w^h = w^h_i e_i, \quad g^h = g^h_i e_i \]

where

\[ v^h_i = \sum_{A \in \eta_i} N_A d_{iA}, \quad w^h_i = \sum_{A \in \eta_i} N_A c_{iA}, \quad v^h_i = \sum_{A \in \eta_i} N_A g_{iA} \]  \hspace{1cm} (45)

("A" is the global node number and "i" is the degree of freedom number that indicates the vectorial component). We introduce these expressions in the Galerkin formulation and we arrive to the next expression:
\[ \sum_{j=1}^{\text{ndof}} \left( \sum_{B \in \eta - \eta_i} a(N_A e_i, N_B e_j) d_{jB} \right) = (N_A e_i, l) + (N_A e_i, l)_i - \sum_{j=1}^{\text{ndof}} \left( \sum_{B \in \eta - \eta_i} a(N_A e_i, N_B e_j) g_{jB} \right) \]

\[ A \in \eta - \eta_{gi} \quad 1 \leq i \leq \text{ndof} \]  

(46)

This is equivalent to the matrix equation:

\[ K d = F \]

where

\[ K = [K_{PQ}] \quad d = [d_Q] \quad F = [F_P] \]

\[ K_{PQ} = a(N_A e_i, N_B e_j) \]

\[ F_P = a(N_A e_i, l) + (N_A e_i, h)_i - \sum_{j=1}^{\text{ndof}} \left( \sum_{B \in \eta - \eta_i} a(N_A e_i, N_B e_j) g_{jB} \right) \]

in which

\[ P = \text{ID}(i,A), \quad Q = \text{ID}(j,B) \]  

(47)

ID is the ID array defined as a generalization of the 1D case:

\[ \text{ID}(i,A) = \begin{cases} 
P & \text{if } A \in \eta - \eta_{gi} \\
0 & \text{if } A \in \eta_{gi} 
\end{cases} \]  

(48)

where we recall that \( \eta \) denotes the set of global node numbers. As we have done with the symmetric part of the tensor \( u \), the symmetric part of \( N_A e_i \) can be defined as:

\[ \varepsilon(N_A e_i) = B_A e_i \quad \text{where} \]

\[ B_A = \begin{bmatrix} N_{A,1} & 0 & 0 \\ 0 & N_{A,2} & 0 \\ N_{A,2} & N_{A,1} \end{bmatrix} \quad \text{for ndof=2}; \]

\[ B_A = \begin{bmatrix} N_{A,1} & 0 & 0 & 0 \\ 0 & N_{A,2} & 0 & 0 \\ 0 & 0 & N_{A,3} & 0 \\ N_{A,3} & 0 & N_{A,1} & 0 \\ N_{A,2} & 0 & 0 & 0 \end{bmatrix} \quad \text{for ndof=3} \]  

(49)
Knowing that, the $K_{PQ}$ will be:

\[
K_{PQ} = e_i^T \int_{\Omega} B_A^T D B_B d\Omega e_j
\]  

Therefore, we have obtained the system of equation that we have to solve, specifying the meaning of each component. However, we are treating with integrals over all the domain. When we compute the method, we have to proceed as in the 1D case and encoding the element stiffness matrix and force vector. The last step would be assembly these matrices into the global matrix. We don’t focus on explain that because it will be explained in the development of our element finite method code.
2.3. Background of B-Splines

2.3.1) DEFINITION AND PROPERTIES OF B-SPLINES BASIS FUNCTION

In this section, we are going to talk about the shape function we use in our work in order to solve the flexoelectric and piezoelectric system of differential equations, the B-Splines. This section is mainly referred to the book of the reference [5]. They are so different respect to these defined in a piecewise linear finite element space, in the background of finite elements. We define $U=\{u_0, \ldots, u_m\}$ as the knot vector with a nondecreasing sequence of real numbers, i.e., $u_i \leq u_{i+1}$ for $i=0, \ldots, m-1$. These $u_i$ are called knots and there can exist some of them in $U$ having the same value. We can define the $i$th B-Spline basis function of $p$ degree, $N_{i,p}(u)$, as:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

$N_{i,0}(u)$ is a step function and $N_{i,p}(u)$ is a linear combination of two $(p-1)$-degree basis functions. They are piecewise polynomial too. Since the definition, we can see that, in order to compute a set of basis functions, it is required to specify a knot vector, $U$, and the degree, $p$ (in a few lines we will see the really necessity of the knot vector). Another comment to do is about terminology: the half-open interval $[u_i, u_{i+1})$ is called the $i$th knot span. To finish with the presentation of the B-splines, we realize that in order to compute the $p$th-degree functions it is necessary to generate some B-splines of less order due to the recursivity of the expression before seen and it generates the next triangular table:

$$
\begin{array}{cccccccc}
N_{0,0} & N_{0,1} & & & & & & \\
N_{1,0} & N_{1,1} & N_{0,2} & & & & & \\
N_{2,0} & N_{2,1} & N_{1,2} & N_{0,3} & & & & \\
N_{3,0} & N_{3,1} & N_{2,2} & & & & & \\
& & & & & & & \\
\end{array}
$$

Figure 14. Example for $p=3$
We want to explain the most remarkable properties of the B-splines. First of all, 
\( N_{i,p}(u) = 0 \) if \( u \) is outside the interval \([u_i, u_{i+p+1})\). This property is talking about local support.

The most important property is this: in any given knot span, \([u_j, u_{j+1})\), at most \( p+1 \) of the 
\( N_{i,p} \) are non zero, the functions \( N_{j-p,p}, \ldots, N_{j,p} \). This property justifies the use 
of the knot vector. The reason is that if we want to calculate a \( p \)-th degree function 
basis in a certain point \( u = u_{\text{point}} \), we know by this property that only will be nonzero the 
p+1 functions before called (at most), and the subscript "\( j \)" now, will be this 
corresponding to the knot span that contains the \( u_{\text{point}} \). Therefore, using this property 
we can avoid to do all the calculus of all the function basis and it's useful in order to do 
an efficient numerical method.

Other property is that all \( N_{i,p}(u) \geq 0 \) for all \( i, p \) and \( u \) (nonnegativity). A part of this, 
the sum of \( N_{j-p,p}, \ldots, N_{j,p} \) in an arbitrary knot span \([u_i, u_{i+1})\) must be equal to 
one. Finally, it's important to know that all derivatives of \( N_{i,p}(u) \) exist in the interior 
of a knot span; and at a knot, it's \( p-k \) times continuously differentiable, where \( k \) is the 
multiplicity of the knot (the number of times that the value of the knot is repeated in 
the knot vector).

![Figure 15. A quadratic curve (p=2) using the knot vector U={0,0,0,1/5,2/5,3/5,4/5,1,1,1})](image)

### 2.3.2) DERIVATIVES OF B-SPLINES BASIS FUNCTIONS

The next expression give us the derivative of a basis function:

\[
N'_{i,p} = \frac{p}{u_{i+p} - u_i} N_{i,p-1}(u) - \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)
\]  
(52)

Repetead differenciation of this formula produces the general formula:
$$N_{i,p}^{(k)}(u) = p \left( \frac{N_{i,p-1}^{(k-1)}}{u_{i+p} - u_i} - \frac{N_{i+1,p-1}^{(k-1)}}{u_{i+p+1} - u_{i+1}} \right)$$  \hspace{1cm} (53)

However, it exists another generalization of this expression easier to compute:

$$N_{i,p}^{(k)} = \frac{p!}{(p-k)!} \sum_{j=0}^{k} a_{k,j} N_{i+j,p-k}$$

$$a_{0,0} = 1$$

$$a_{k,0} = \frac{a_{k-1,0}}{u_{i+p-k+1} - u_i}$$

$$a_{k,j} = \frac{a_{k-1,j} - a_{k-1,j-1}}{u_{i+p+j-k+1} - u_{i+j}} \quad j = 1, \ldots, k - 1$$

$$a_{k,k} = \frac{-a_{k-1,k-1}}{u_{i+p+1} - u_{i+k}}$$  \hspace{1cm} (54)

In order to compute a numerical method that calculate these $N_{i,p}^{(k)}$, we have to take into consideration this two remarks:

- The derivative order $K$ should not exceed $p$ because the higher derivatives will be zero.
- It is possible that the denominators involving knot differences become zero. In this case, it must be defined as zero.
- As the $N_{i,p}$, the only $N_{i,p}^{(k)}$ that are non-zero when we evaluate them in a point $u_r$ contained in a knot span $[u_j,u_{j+1}]$ are the $N_{j-p,p}^{(k)}$, ..., $N_{j-p,p}^{(k)}$.

![Figure 16. (a) Cubic basis functions, (b) derivatives corresponding to the previous basis function.](image-url)
2.3.3) B-SPLINES CURVES

A) DEFINITION

It's important to explain that B-Splines are commonly used to define curves and surfaces for computer-aided geometric design.

We can define a $p$th-degree B-spline curve by the next formula:

$$ C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \quad a \leq u \leq b $$

where $N_{i,p}(u)$ are the $p$th-degree B-spline basis function and $P_i$ are the control points. The first one are defined on the knot vector:

$$ U = \left\{ a, \ldots, a, u_{p+1}, \ldots, u_{m-p-1}, b, \ldots, b \right\} $$

As we can see in the expression of the knot vector, the first and last $p+1$ knots correspond to the limit values that $u$ can adopt. A normal choose of the values of the middle is to separate them equitatively in an increasing way until the value reach $b$. We define $m$ as the number of knots minus one. The correct way to create the B-spline curve is taking in account this relation:

$$ n+1 = m - (p+1) $$

where $n+1$ is the number of control points. The importance of the knot vector as we remark before is that depending on which point, $u_r$, we consider to evaluate $C(u)$, we should choose the knot span $[u_i, u_{i+1}]$ that contains this point and the $N_{i-p}, \ldots, N_{i}$ B-splines basis function will be only non-zero.

A part of it, we have to explain how to choose the control points that define the control polygon. Each control point multiplies its corresponding $i$th B-spline basis function. They are called degree of freedom too. Depending on the problem we want to resolve, i.e., the surface we want to describe, we will choose one control polygon or another.

B) FINDING THE CONTROL POINTS [6]

i) By curve global interpolation or by least square method

One case of finding the control points is supposing we know some points of the solution $D_0, D_1, \ldots, D_n$ for a given $t_0, t_1, t_2, \ldots, t_n$ (specific values of $u$). These last can be not given and they must be considered as parameters. Continuing with the objective to
find the control points, we use a curve global interpolation, imposing this condition:

\[ D_k = C(t_k) = \sum_{i=0}^{n} N_{i,p}(t_k)P_i \quad \text{for } 0 \leq k \leq n \]  

(58)

We consider the vectors \( D_k \) and \( P_i \) have \( s \) dimension (vectorial character). So we have a system of \( n \times s \) equations. We solve the system and we obtain the control points. However, the curve global interpolation it's not the best way to find the control points because of the non-interpolating natural character of the B-splines basis. The best method to do it is using a least square method. As we know it consists in minimize the square error. Doing that we obtain the next system of equations:

\[ \sum_{a} \left( \int N_a \cdot N_b \, dx \right) \cdot f_a = \int N_b \, f \quad \text{where} \quad f(x) = \sum N_a(x) f_a \quad \text{is our B-Spline curve solution and} \quad f_a \quad \text{are the control points.} \]  

(59)

ii) By solving a system of differential equations by galerkin method

This section connects again the background of finite elements and the background of B-splines. We have to remember that the principal connection between them is that the shape functions we use to calculate the stiffness matrix \( K \) and the force vector \( F \) are the B-spline basis functions. A part of it, we only want to remark that the displacement vector \( d \) that we want to obtain as a solution of the matrix equation \( K \cdot d = F \) and that we will use to construct the solution are the control points, precisely. So the galerkin method is the way to find the control points.

In our flexoelectric problem, we don't consider a B-Spline curve if not a B-Spline surface but all this explained before will be exactly the same. We find the control points with the finite elements method.

C) THE DERIVATIVE OF A B-SPLINES CURVE

\[ C^{(k)}(u) = \sum_{i=0}^{n} N_{i,p}^{(k)}(u)P_i \]  

(60)

We call \( C^{(k)}(u) \) to the \( k \)th derivative of a B-Spline curve and it corresponds to the next expression: In order to obtain this formula we only have to apply the linearity property of the derivative on the B-Spline curve. The \( k \)th derivative is applied on the B-Spline basis functions, that have been explained before with their recursive formula.
2.3.4) B-SPLINE SURFACES

At the previous case, the B-Spline curve only depends on one variable, \( u \). However, a B-Spline surface depends on two, \( u \) and \( v \). So, we have the formula that defines the B-Spline surfaces taking a bidirectional net of control points:

\[
S(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,q}(v) P_{i,j}
\]

As we can see, now we consider one B-Spline basis function that depends on \( u \) and another that depends on \( v \). They appear in a tensor product surfaces. We consider the control points and the surface values are scalar, as the case more used. Now, we have two knot vectors and two conditions to impose:

\[
U = \{0, \ldots, 0, u_{p+1}, \ldots, u_{r-p-1}, 1, \ldots, 1\} \quad V = \{0, \ldots, 0, v_{q+1}, \ldots, v_{s-q-1}, 1, \ldots, 1\}
\]

\[
\tau = n + p + 1 \quad \text{and} \quad s = m + q + 1
\]

Figure 17. The first picture reproduce the product of two B-splines basis functions of different variables, the tensor product while the second and third picture show a B-Splines surface: the control net and the surface respectively.

In the same way as we have obtained the \( k \)-th-derivative of a B-Spline curve, we can do it for a B-Spline surface with this formula (computing derivatives of the basis functions):

\[
\frac{\partial^{k+l}}{\partial u^k \partial v^l} S(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}^{(k)} N_{j,q}^{(l)} P_{i,j}
\]

We can show this expression in a matrix form too, easier to understand how we have to compute it:
where uspan is the subscript of the first knot of the knot span which contains the point \((u_0, v_0)\) we evaluates the derivative.

In our model, the function solution we obtain is a B-Spline surface. Therefore, we have the necessity to know how to create the algorithm that allow us to compute any point of the domain on the B-Spline surface. This algorithm is general and we have to introduce the next inputs: the order \(p\) and \(q\), the two knot vectors, the control points and the points in which we evaluates the B-Spline surface. The output is the value of the surface. Therefore, we need to know which are the control points and we will know them when we solve the problem by the finite elements method. As we explain before, they will be the components of the displacement vector.

In another tipus of work it is possible to dispose some data of the solution. With the purpose of finding the control points we can use the least square method as before in the B-Splines curves. The difference with the former case is that now we have a B-Splines basis that is a tensor product of two B-Splines basis.
3. Computational model of flexoelectricity in 2D

3.1 Theoretical development of our model

We are going to develop theoretically an interesting model that can allow us to study the electromechanical response of linear dielectric solids in two dimensions. The idea is to deal with the high-order coupled partial differential equations of flexoelectricity using a galerkin method that uses as their shape functions, the B-Spline basis functions. After all, we will solve a system of equations that will give us the degrees of freedom necessary to approximate our solution: the mechanical displacement field $u$ and the electrical potential $\phi$.

We begin with the density of enthalpy energy:

$$ H(\varepsilon_{i,j}, E_i, \varepsilon_{jk,l}) = \frac{1}{2} C_{ijkl} \varepsilon_{ij} \varepsilon_{kl} - e_{ijkl} E_i \varepsilon_{kl} + f_{ijkl} E_i \varepsilon_{jk,l} + d_{ijkl} E_i \varepsilon_{kl} - \frac{1}{2} k_{ij} E_i E_j $$

(64)

This enthalpy depends on the variables: strain tensor $\varepsilon_{ij}$ (called $u_{ij}$ in a previous section of the work), electric field $E_i$, strain gradient and electric field gradient, as is showed in the expresion (respectively). The first term of the enthalpy is the elastic potential energy density where $C_{ijkl}$ is the fourth-order stiffness tensor (this term is like $E_p = \frac{1}{2} K x^2$). The second term is the piezoelectric term, or the strain-electric field coupling term and $e$ is the third order tensor of piezoelectricity. The third and four term explain the direct and converse flexoelectric effect respectively. The first one couples the strain gradient with the electric field and it’s multiplying by the fourth-order tensor $f$ while the second one couples the electric field gradient with the strain through a fourth-order tensor $d$. The last term is the electrostatic energy density where $K$ is the second-order dielectric tensor. Integrating by parts we can express the equation of before as:

$$ H(\varepsilon_{i,j}, E_i, \varepsilon_{jk,l}) = \frac{1}{2} C_{ijkl} \varepsilon_{ij} \varepsilon_{kl} - e_{ijkl} E_i \varepsilon_{kl} - \mu_{ijkl} E_i \varepsilon_{jk} - \frac{1}{2} k_{ij} E_i E_j $$

(65)

Now the two flexoelectric terms are expressed by only one term $\mu_{ijkl} = d_{ijkl} - f_{ijkl}$. The two expressions of the enthalpy only differ in the natural boundary conditions. Once we have have obtained the enthalpy, we define the usual stress, the electric displacement, the high order stress and electrical displacement respectively:
\[ \sigma_{ij} = -\frac{\partial H}{\partial \varepsilon_{ij}} \quad D_i = -\frac{\partial H}{\partial E_i} \quad \sigma_{\tilde{ij},k} = -\frac{\partial H}{\partial E_{ij,k}} \quad \tilde{D}_{ij} = -\frac{\partial H}{\partial E_{i,j}} \]  

(66)

With these expressions we can define the physical stress and the physical electric displacement as:

\[ \begin{align*}
\sigma_{ij} &= \sigma_{\tilde{ij}} - \sigma_{\tilde{ijk},k} = C_{ijkl} \varepsilon_{kl} - e_{kij} E_k + \mu_{ijkl} E_{l,k} \\
D_i &= D_{\tilde{i}} - D_{\tilde{ij},j} = e_{ijkl} \varepsilon_{kl} + k_{ij} E_j + \mu_{ijkl} \varepsilon_{j,k,l}
\end{align*} \]

(67)

These two equations are the constitutive equations of electromechanics and they will allow us to obtain the strong form of our problem. Depending on the work or study you do, the constitutive equations can be more or less complex including more or less physical effects. Some of the possible approaches we can distinguish are:

<table>
<thead>
<tr>
<th>Constitutive equations</th>
<th>Physical effects included</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma = C \varepsilon ) ( D = k E )</td>
<td>Elastic, Dielectric</td>
</tr>
<tr>
<td>( \sigma = C \varepsilon - e E ) ( D = e \varepsilon + k E )</td>
<td>Elastic+Piezo, Dielectric+Piezo</td>
</tr>
<tr>
<td>( \sigma = C \varepsilon - e E + \mu \nabla (E) ) ( D = e \varepsilon + k E + \mu \nabla (\varepsilon) )</td>
<td>Elastic+Piezo+Flexo, Dielectric+Piezo+Flexo</td>
</tr>
</tbody>
</table>

Therefore, we are studying the third case that include: the elastic, piezoelectric and flexoelectric physical effects in a dielectric material.

The strong form of our problem is obtained replacing the constitutive electromechanical equations on the mechanical and electrostatic equilibrium equations. Therefore, we have the statement of the strong form of our problem [7]:

\[ \nabla \sigma = 0 \quad \nabla \cdot D = 0 \quad \text{where} \]

\[ \begin{align*}
\sigma_{ij} &= \sigma_{\tilde{ij}} - \sigma_{\tilde{ijk},k} = C_{ijkl} \varepsilon_{kl} - e_{kij} E_k + \mu_{ijkl} E_{l,k} \\
D_i &= D_{\tilde{i}} - D_{\tilde{ij},j} = e_{ijkl} \varepsilon_{kl} + k_{ij} E_j + \mu_{ijkl} \varepsilon_{j,k,l}
\end{align*} \]

(68)

The statement of the strong form include the boundary conditions and we need them
in order to apply the galerkin method. These are the natural and electrical boundary conditions:

\[
\phi = \bar{\phi} \quad \text{on} \quad \Gamma_{\phi}
\]

\[
D_j n_i = - \bar{w} \quad \text{on} \quad \Gamma_{\nu}
\]

where \(\bar{\phi}\) and \(\bar{w}\) are the prescribed electrical potential and surface charge density.

We have to take into account the mechanical boundary conditions too:

\[
u_i = \bar{u}_i \quad \text{on} \quad \Gamma_u
\]

\[
t_k = n_j (\partial_{jk} - \sigma_{ijk}) - D_j (n_i \sigma_{ijk}) - (D_p n_p) n_j \sigma_{ijk} = \bar{t}_k \quad \text{on} \quad \Gamma_t
\]

where \(\bar{u}_i\) and \(\bar{t}_k\) are the prescribed mechanical displacement and tractions.

Once we have defined the strong form of the problem, we have to find the weak form following the galerkin formulation. So, we multiply the strong form equations by the weighting functions \(w, w'\) and we integrate them through integration by parts. Then, we apply the boundary conditions.

\[
\int_{\Omega} w \nabla \cdot \sigma = 0 \quad \rightarrow \quad - \int_{\Omega} \nabla (w) \cdot \sigma + \int_{\Gamma_s} w \sigma n_i - \int_{\Omega} \nabla (w) \cdot \sigma + \int_{\Gamma_s} w \bar{t} = 0
\]

\[
\int_{\Omega} w' \nabla \cdot D = 0 \quad \rightarrow \quad - \int_{\Omega} \nabla (w') \cdot D + \int_{\Gamma_t} w' D n_i - \int_{\Omega} \nabla (w') \cdot D - \int_{\Gamma_t} w' \bar{w} = 0
\]

where \(\Omega\) is refered to all the domain and \(\Gamma_{\phi}, \Gamma_{\nu}\) are referred to the boundary.

After that, we apply the Galerkin method (a little variant of which has been explained before). It consists in replacing the functions \((w, w')\) by the virtual displacement and the virtual electrical potential, respectively. These formulation is often called virtual work or virtual displacement principles. So, now we have:

\[
- \int_{\Omega} \nabla (\delta u) \cdot \sigma + \int_{\Gamma_s} \delta u \bar{t} = 0
\]

\[
- \int_{\Omega} \nabla (\delta \phi) \cdot D - \int_{\Gamma_t} \delta \phi \bar{w} = 0
\]

Now, we forget about the last term of these equations because they will stay equal in the weak form and we focus on the first term of them. We have to remember the next expresions of the strain and electric field (applied to virtual displacements):
So we substitute these in the first terms of the integral equations. Therefore, these terms will be:

\[-\int_{\Omega} \nabla (\delta u) \sigma = -\int_{\Omega} \delta \varepsilon \sigma ; \quad -\int_{\Omega} \nabla (\delta \phi) D = \int_{\Omega} \delta E D\]

(74)

After that, we develop the first of the two expressions substituting \( \sigma = (\hat{\sigma} - \hat{\sigma}^\prime) \):

\[-\int_{\Omega} \delta \varepsilon \sigma = -\int_{\Omega} \delta \varepsilon (\hat{\sigma} - \hat{\sigma}^\prime) = -\int_{\Omega} \delta \varepsilon \hat{\sigma} + \int_{\Omega} \delta \varepsilon \hat{\sigma}^\prime = -\int_{\Omega} \delta \varepsilon \hat{\sigma} - \int_{\Omega} \nabla (\delta \varepsilon) \hat{\sigma}\]

(75)

In the last step, we have applied integration by parts too. Finally, we sum the Eq. (71) and Eq. (72), using the last expression (74) and (75) as the first terms. Then, we obtain the weak form of our problem, that's the same of the article in which this part is based[7]:

\[\int_{\Omega} (\delta \varepsilon \hat{\sigma} + \nabla (\delta \varepsilon) \hat{\sigma} - \delta E D) d\Omega - \int_{\Gamma} (\delta u \bar{t}) dS + \int_{\Gamma} (\delta \phi \bar{w}) dS = 0\]

(76)

Now, following the method, we have to keep in mind the following expression for \( \hat{\sigma}_{ij} \), \( \hat{\sigma}^\prime_{ijk} \) and \( D \) that can be extracted of the constitutive equations before announced:

\[\hat{\sigma} = C \varepsilon - eE \quad ; \quad \hat{\sigma} = \int -\mu \nabla (E) = -\mu E \quad ; \quad D = e \varepsilon + k E + \mu \nabla (\varepsilon)\]

(77)

So, we substitute these expressions in the weak form and we obtain:

\[\int_{\Omega} (\delta \varepsilon C \varepsilon - \delta \varepsilon eE - \nabla (\delta \varepsilon) \mu E - \delta E e \varepsilon - \delta E k E - \delta E \mu \nabla (\varepsilon)) d\Omega - \int_{\Gamma} (\delta u \bar{t}) dS + \int_{\Gamma} (\delta \phi \bar{w}) dS = 0\]

(78)

In order to make easier the development, the variables \( \delta \varepsilon \), \( \delta E \) will become the original variables \( \nabla (\delta u) \), \( -\nabla (\delta \phi) \) (it seems that the opposite conversion made in a few steps ago has no sense because we finally use the original variables, but the reason why we use it is to demonstrate we obtain the same expression of the weak form than the article of the reference). Also, the real electric field and strain variables are replaced by the potential gradient and the displacement gradient. Therefore, we have:

\[\int_{\Omega} (\nabla (\delta u) C \nabla (u) + \nabla (\delta u) e \nabla (\phi) + \nabla^2 (\delta u) \mu \nabla (\phi) + \nabla (\delta \phi) e \nabla (u) - \nabla (\delta \phi) k \nabla (\phi) + \nabla (\delta \phi) \mu \nabla^2 (u)) d\Omega - \int_{\Gamma} (\delta u \bar{t}) dS + \int_{\Gamma} (\delta \phi) \bar{w} dS = 0\]

(79)
Continuing with the method we define the discrete approximations of the mechanical displacement "u" and the electrical potential "ϕ", i.e., of the solutions of our problem:

\[ u_h = \sum_{b=1}^{N} p_b(\bar{x}) u_b \quad ; \quad \phi_h = \sum_{b=1}^{N} p_b(\bar{x}) \phi_b \]  

(80)

where \( p_b(\bar{x}) \) are a set of smooth basis functions. In our case they will be the B-Spline basis function.

Due to the condition of \( \delta u, \delta \phi \) of being virtual variables:

\[ \delta u = \sum_{a=1}^{N} p_a(\bar{x}) \quad ; \quad \delta \phi = \sum_{a=1}^{N} p_a(\bar{x}) \]  

(81)

Now, we substitute the discrete approximations of our solutions and the new expressions of the virtual variables into the Eq. (79). Applying the linearity of integration, we obtain the next system of equations:

\[
\sum_{a,b} \int_{\Omega} \nabla_u(p_a) C \nabla_u(p_b) u_b + \nabla_u(p_a) \epsilon \nabla_\phi(p_b) \phi_b + \nabla^2_u(p_a) \mu \nabla_\phi(p_b) \phi_b + \nabla_\phi(p_a) \eta \nabla_u(p_b) u_b \\
- \nabla_\phi(p_a) k \nabla_\phi(p_b) \phi_b + \nabla(p_a) \mu \nabla^2(p_b) u_b \ d\Omega = \int_{\Gamma_\nu} \bar{t} \ dS - \int_{\Gamma_\nu} \bar{\nu} \ dS 
\]

(82)

We apply the Gauss-Legendre quadrature for approximate the definite integrals involved. A quadrature rule "substitute" the integrals by a weighted sum of function values at specified points within the domain of integration:

\[
\int_{-1}^{1} f(x) \ dx = \sum_{i=1}^{n} w_i f(x_i) 
\]

(83)

The domain of integration for such a rule is conventionally taken as \([-1, 1]\). For this domain, the gaussian quadrature points and their corresponding weights are stipulated in the literature. They appear in the next table:
If we want to use the quadrature for another domain, we have to do the change of variable 
\[ z = \frac{b-a}{2} x + \frac{a+b}{2} \]
obtaining the next formula:

\[
\int_a^b f(x) dx = \frac{b-a}{2} \sum_{i=1}^{n} w_i f(z_i)
\]  \hspace{1cm} (84)

Once we know how to compute the integrals we turn into the (82) expression. We can organize the discrete algebraic equations in a matrix form [7]:

\[
\begin{bmatrix}
A_{UU} & A_{U\phi} \\
A_{\phi U} & A_{\phi\phi}
\end{bmatrix}
\begin{bmatrix}
U \\
\phi
\end{bmatrix} =
\begin{bmatrix}
f_U \\
f_\phi
\end{bmatrix}
\]  \hspace{1cm} (85)

where the local contribution of each quadrature point to the matrix of the system has the structure:

\[
\begin{array}{|c|c|}
\hline
\text{Number of points, } n & \text{Points, } x_i & \text{Weights, } w_i \\
\hline
1 & 0 & 2 \\
2 & \pm \sqrt{\frac{1}{3}} & 1 \\
3 & 0 & \frac{8}{9} \\
& \pm \sqrt{\frac{3}{5}} & \frac{5}{9} \\
4 & \pm \frac{3}{7} - \frac{2}{7} \sqrt{\frac{6}{5}} & \frac{18+\sqrt{30}}{36} \\
& \pm \frac{3}{7} + \frac{2}{7} \sqrt{\frac{6}{5}} & \frac{18-\sqrt{30}}{36} \\
5 & 0 & \frac{128}{225} \\
& \pm \frac{1}{3} \sqrt{5 - 2 \sqrt{\frac{10}{7}}} & \frac{322+13\sqrt{70}}{900} \\
& \pm \frac{1}{3} \sqrt{5 + 2 \sqrt{\frac{10}{7}}} & \frac{322-13\sqrt{70}}{900} \\
\hline
\end{array}
\]

Figure 18. Table for the gaussian quadrature points and for their corresponding weights for the domain [-1,1].
Now, the symbols $B_u$ and $B_\phi$, $H_u$ are substituing the symbols of the gradient and hessian ($\nabla u$, $\nabla_\phi$, $\nabla_\phi^2$). They are acting on the B-splines basis. The matrix operators involved, i.e. the matrix gradient operator (respect $u$ and $\phi$) and the matrix hessian operator (respect $u$) are defined in the next way:

$$A_{uu}^{ab} = B_u(p^a)CB_u^T(p^b),$$
$$A_{u\phi}^{ab} = B_u(p^a)eB_\phi^T(p^b) + H_u(p^a)\mu B_\phi^T(p^b),$$
$$A_{\phi u}^{ab} = B_\phi(p^b)e^TB_u^T(p^a) + B_\phi(p^b)\mu H_u^T(p^a),$$
$$A_{\phi\phi}^{ab} = -B_\phi(p^a)KB_\phi^T(p^b),$$
$$f_u = \bar{\ell}p^a, \quad f_\phi = -\omega p^a,$$

(86)

Finally, we have to define the representation as matrices of the stiffness tensor $C$, the dielectric tensor $K$, the piezoelectric tensor $e$ and the flexoelectric tensor $\mu$:

$$B_u = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & \frac{\partial}{\partial y} \\ 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix}, \quad B_\phi = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix},$$

$$H_u = \begin{bmatrix} \frac{\partial^2}{\partial x^2} & 0 & \frac{\partial^2}{\partial y \partial x} & \frac{\partial^2}{\partial x \partial y} & 0 & \frac{\partial^2}{\partial y^2} \\ 0 & \frac{\partial^2}{\partial y \partial x} & \frac{\partial^2}{\partial x \partial y} & 0 & \frac{\partial^2}{\partial y^2} & \frac{\partial^2}{\partial x \partial y} \end{bmatrix}.$$

(87)

Finally, we have to define the representation as matrices of the stiffness tensor $C$, the dielectric tensor $K$, the piezoelectric tensor $e$ and the flexoelectric tensor $\mu$:
\[ C = \frac{Y}{(1 + \nu)(1 - 2\nu)} \begin{bmatrix} 1 - \nu & \nu & 0 \\ \nu & 1 - \nu & 0 \\ 0 & 0 & \frac{1 - 2\nu}{2} \end{bmatrix}, \]

\[ K = \begin{bmatrix} \kappa_{11} & 0 \\ 0 & \kappa_{33} \end{bmatrix}, \quad e^T = \begin{bmatrix} 0 & 0 & e_{15} \\ e_{31} & e_{33} & 0 \end{bmatrix}, \]

\[ \mu = \begin{bmatrix} \mu_{11} & \mu_{12} & 0 & 0 & 0 & \mu_{44} \\ 0 & 0 & \mu_{44} & \mu_{12} & \mu_{11} & 0 \end{bmatrix}. \]
3.2 Numerical methodology of the model

In this section we explain how to solve numerically the system before commented. We will describe all the numerical steps in a detailed way. It is complemented by the numerical code that can be found in the appendix, written in the programming language MATLAB. We have to mention that the model encoded is not exactly the system commented in the previous section. In order to learn better how to programme it we only solve for the mechanical part of the problem, i.e., we focus on the first matrix of the system $A_{UU}$. We add a new term in this matrix, the strain elasticity term \([8]\), that doesn't appear in the model before described, but it's necessary in order to validate the correct running of the code and will be explained later. The others terms $A_{UU}, A_{U\Phi}, A_{\Phi U}, A_{\Phi \Phi}$ can be added to the system in a generalized way. We don't do that because after incorporating all the terms we have to think about the electromechanical boundary conditions necessary to simulate our flexoelectric material and it requires more detailed studies. A part of it, it's more difficult to validate the code, we leave this as a future research and we concentrate in our objective, that is to understand how to carry out an interesting 2D flexoelectrical model.

So the system encoded will be:

$$
[A_{UU}] [U] = [f_U]
$$

where $A_{UU}^{ab} = B_u(p^a) C B_u^T(p^b) + H_s(p^a) h H_s^T(p_b)$ \hspace{1cm} (89)

This new term, called the gradient elasticity term, which its formulation haven't been demonstrated in this work, contains the strain gradient tensor $h$ and a new type of Hessian operator $H_s$ that are described by the next matrices:

$$
h = I_1^2
$$

where $c_{ij} = (c_{11} - c_{22})/2$. \hspace{1cm} (90)

$$
H_s = \begin{bmatrix}
\frac{\partial^2}{\partial x^2} & 0 & \frac{\partial^2}{\partial x \partial y} & 0 & \frac{\partial^2}{\partial y \partial x} & \frac{\partial^2}{\partial y^2} \\
0 & \frac{\partial^2}{\partial y^2} & 0 & \frac{\partial^2}{\partial y \partial x} & \frac{\partial^2}{\partial x^2} & \frac{\partial^2}{\partial x \partial y}
\end{bmatrix}
$$

where $c_{11} = E(1-\nu)/(1+\nu)(1-2\nu), c_{12} = E \nu/(1+\nu)(1-2\nu)$ and $c_{11} = (c_{11} - c_{22})/2$. \hspace{1cm} (91)
$\nu$ is the poisson coefficient, $E$ (or $Y$) is the Young modulus of the material and $l_1$ is a conservative value for the length scale.

It's very important to consider the gradient elasticity term because doing that an interesting elastic effect, already founded by the scientific community, appeared. It's called the size effect of cantilever beam bending. It will be the results the our working and the validation that our numerical code works correctly. A part of validation, the gradient elasticity term is necessary to having in account because it assures stability to the system and it allows us to reproduce faithfully the properties and conditions of very small samples (about micro or nano scales).

Another thing to remark is that the solution of our system of equations is the mechanical displacement vector $u$ (two components: $u_x$ and $u_y$). Solving the system we obtain the degrees of freedom of our approximate solution "u" and multiplying them by their corresponding two-dimensional B-Splines basis functions we can obtain the surface solution. In order to make a correct simulation we have to impose a particular configuration of our sample specifying the boundary conditions. In our case we try this configuration: a cantilever beam (rectangular sample) fixed on one end and applying an external force $P$ in the other, as is sketched in the picture:

![Figure 19. Configuration of our cantilever beam.](image)

**Development of the code**

In order to develop the numerical code that solves the system before mentioned and involve a galerkin method using B-Splines, we have to keep in mind all the ingredients that have to be compiled. First of all, the B-splines basis functions $N_{i,p}$ and its derivatives (first and second derivatives, for one dimension), that will be stored as some numerical values when we evaluate them in a specific point and they will be of order $p$ (in our case $p=2$). Another important ingredient are the collection of points choosen for using the gaussian quadrature rule and its corresponding gaussian weights. Finally, we have to take in consideration the division of our sample into elements. They are choosen throughout the selection of the different values of the two knot vectors in $x$ and in $y$. The tensorial product of two knot spans, each one corresponding to a different knot vector, generates an element.
Our code is divided in some files (all are in the appendix) that correspond to different steps of the development. We are going to tell the different steps involved in the method but in order to understand correctly the reader should complement it with the code written in the appendix of the work because we talk about some details of it. The main steps, ordered on time, that we execute are:

- Creation of the B-Spline basis functions and their derivatives evaluated in one specific point. It corresponds to the file evaluate_Bsp. This file consist in a function that entering the next incoming variables: one knot vector, the order of the B-Splines basis functions, the maximum derivative we want to apply and the point "u" in which we want to evaluate them, it gives us the B-Splines basis function and their derivatives evaluated in this specific point. First of all, the code find the knot span in which "u" lies, meaning that this value is contained between the two values of the the knot span. After that we compute the nonzero basis function. In order to do that, we recall the recursive expression of the definition of the B-Spline basis function. We realize there is a new form to compute the basis functions with the introduction of a new notation. We call:

\[
\text{left}(k) = u - u_{i+k} \quad \text{right}(k) = u_{i+k} - u
\]

The new form to compute the basis in a knot span \([u_i, u_{i+1}]\) is:

\[
N_{i-j,p} = \frac{\text{left}(j+1)}{\text{right}(p-j) + \text{left}(j+1)} \cdot N_{i-j,p-1} + \frac{\text{right}(p+1-j)}{\text{right}(p+1-j) + \text{left}(j)} \cdot N_{i-j+1,p-1}
\]

We realize that the expression \(N_{i-j+1,p-1} \cdot \text{right}(p+1-j) + \text{left}(j)\) which appears in the second term of the equation will appear in the first term of \(N_{i-j+1,p}\). Equally, the second term of this other basis function will repeat in the next iteration. Therefore, all this allow us to get an optimized code. We have to remember that we only compute the \(N_{i-p,p}, N_{i-p+1,p},..., N_{i,p}\) basis because there are the only nonzero functions. Finally, we obtain a matrix called arbitrarily "ders", that contains in the first row the B-splines basis evaluated at the point u while the second, the third row and successives files, contain the first and second derivatives and the successives derivatives, respectively, evaluated at the same point. We obtain the index of the span in which it's contained the point evaluated, called "sp". As an example for \(p=2\), and maximum derivative 2 we obtain:

\[
ders = \begin{bmatrix}
N_{i-2,2}(u) & N_{i-1,2}(u) & N_{i,2}(u) \\
N_{i-2,2}^{[1]}(u) & N_{i-1,2}^{[1]}(u) & N_{i,2}^{[1]}(u) \\
N_{i-2,2}^{[2]}(u) & N_{i-1,2}^{[2]}(u) & N_{i,2}^{[2]}(u)
\end{bmatrix}
\]
- Evaluation of the B-splines basis functions $N_{i,p}(x)$ and their derivatives in a selection of gaussian quadrature points. It corresponds to the file basis_function.m. This file consist in a function that entering the next incoming variables: one knot vector, the order of the B-Splines basis functions, the number of quadrature points contained in any knot span and the maximum derivative we want to apply, it gives us mainly the B-Splines basis function and their derivatives evaluated in the corresponding quadrature points. In order to obtain the values of the quadrature points it’s necessary to encode a subroutine. This subroutine will be described in another file called gaquad.m. This file saves the gaussian quadrature points and the corresponding weights that are stipulated on the literature, for integrals with this span: $[-1,1]$. Turning to the first file, in order to obtain the appropriate quadrature points (and weights) for any knot span of the knot vector, we must do a change of variable. Another thing to comment is that specifying the knot vector we are specifying the number of elements we have in one dimension (the dimension for $x$). The elements are the knot spans that are defined by different values at their edges. Once we know it, we make a double loop for the different elements defined and for each quadrature point contained in each element. In any step of the double loop, we adjust the general quadrature points and weights extracted of file gaquad.m (that are compiled between -1 and 1) to the interval of the corresponding element. A part of it, in the double loop we execute the subroutine evaluate_Bsp described before considering as "u" the specific gaussian quadrature point. We store all the values of the quadrature points and weights adjusted and we store three matrices for the B-Spline basis functions evaluated at the quadrature points and for the first and second derivatives of them (considering maximum derivative equal to 2). The first matrix called arbitrarily "Nshape" contains for the first rows the non-zero B-Splines basis function evaluated at the gaussian points of the first element (each row for each point). After these rows, we store the basis evaluated at the gaussian points of the second element and successively (the same for the matrices for the derivatives). An example for three gaussian points for knot span, with three elements and $p=2$ we obtain:

$$N_{shape} = \begin{bmatrix}
N_{0,2}(x_0) & N_{1,2}(x_0) & N_{2,2}(x_0) \\
N_{0,2}(x_1) & N_{1,2}(x_1) & N_{2,2}(x_1) \\
N_{1,2}(x_2) & N_{2,2}(x_2) & N_{3,2}(x_2) \\
N_{1,2}(x_3) & N_{2,2}(x_3) & N_{3,2}(x_3) \\
N_{2,2}(x_4) & N_{3,2}(x_4) & N_{4,2}(x_4) \\
N_{2,2}(x_5) & N_{3,2}(x_5) & N_{4,2}(x_5)
\end{bmatrix}$$

Each row has the $N_{i-p,p}, N_{i-p+1,p}...N_{i,p}$ basis.

A part of it we create a matrix called "connect" that compute for each element the global index of the B-spline basis used. For the previous "Nshape" it corresponds the
next matrix "connect":

\[
\text{connect} = \begin{bmatrix}
1 & 2 & 3 \\
2 & 3 & 4 \\
3 & 4 & 5 \\
\end{bmatrix}
\] \hspace{1cm} (96)

- **Creation of the global Stiffness matrix** \( K \) of the galerkin formulation. It corresponds to the file \text{create\_matrix.m} . This file consists in a function in which we introduce the next incoming variables: the matrix "Nshape" calculated before, i.e., the B-Splines basis functions evaluated in the gaussian quadrature points corresponding to a knot vector; the matrix "dNsh" and "ddNsh", that corresponds to the first and second derivatives of the "Nshape"; the number of elements in one dimension called "n\_el", the number of gaussian points that are calculated in each element and the values of the gaussian weights stored following the order of the elements, called "gaussno" and "gweight" respectively; and finally the number of control points called "num\_cp". All these represent only one dimension, so we have to introduced twice, for the knot vector that represents the "x" dimension and the knot vector that represents the "y" dimension. We don't have to forget that our solution is a B-Spline surface, so the basis of this will be the tensorial product of B-Splines basis functions, one evaluated at "x" points and the other evaluated at "y" points. Another thing to comment is to remember the definition of the control points or degrees of freedom. They form the vector solution of our method called the displacement vector. They are the coefficients that multiply the corresponding two-dimensional B-Spline basis, \( N_{i,p}(x)N_{j,q}(y) \) in order to approximate our final solution (\( u \), the mechanical displacement). If we want to know what's the number of control points corresponding for one dimension, we only have to take the maximum value of the last row of the "connect" matrix. The reason is because this value is the index of the last one-dimensional B-splines basis. Therefore, to obtain the total number of control points in two dimensions we have to multiply the last index of the "x" B-Splines basis times the last index of the "y" B-Spline basis. Any total control point is related to one of these component of the matrix (for \( p=2 \), number of elements in one dimension= 3):

\[
N(x, y) \text{ basis} = \begin{bmatrix}
N_0(x)N_0(y) & N_0(x)N_1(y) & N_0(x)N_2(y) & N_0(x)N_3(y) & N_0(x)N_4(y) \\
N_1(x)N_0(y) & N_1(x)N_1(y) & N_1(x)N_2(y) & N_1(x)N_3(y) & N_1(x)N_4(y) \\
N_2(x)N_0(y) & N_2(x)N_1(y) & N_2(x)N_2(y) & N_2(x)N_3(y) & N_2(x)N_4(y) \\
N_3(x)N_0(y) & N_3(x)N_1(y) & N_3(x)N_2(y) & N_3(x)N_3(y) & N_3(x)N_4(y) \\
N_4(x)N_0(y) & N_4(x)N_1(y) & N_4(x)N_2(y) & N_4(x)N_3(y) & N_4(x)N_4(y)
\end{bmatrix}
\] \hspace{1cm} (97)
This association is due to the definition of a B-Spline surface. This matrix it isn’t compilate in the code but it help us to understand what we are doing. We can count off these component in the way that the five values of the first row are related with the 1,2,3,4,5 control points; the values of the second row are related with the 6,7,8,9,10 control points and successively. This is the global numeration. Another way to understand it that’s better is consider that the values of one row of the matrix match with the control points localized at their corresponding column of the element mesh. So, as an example, we can draw our element mesh with the control points counted off in this way:

![Graphical description of our element mesh](image)

You can observe that there are some points localized out of the 9 elements sketched. These control points correspond to the boundary of our sample. So the elements of the \( N(x,y) \) basis corresponding to the boundary control points are these that contains the \( N_0 \) or the \( N_4 \) one dimensional B-Spline basis. It’s important to keep in mind this configuration because we have localized the control points in our sample. As a consequence, we can impose boundary conditions in a specific control point of the sample.

Now, turning to the explanation of the code of this file we have to comment that the outcoming variable is the own global Stiffness matrix \( K \). First of all, the code execute three double loops for the element of "x" and "y" choosen, for the "x" and "y" quadrature point used for this element and for the "x" and "y" B-Spline basis. This triple loop allow us to calculate the tensorial product of the one-dimension non-zero B-Splines basis functions derivatives of each element evaluated in all the gaussian points of the element. When we talk about non-zero B-Splines we are referring to the \( N_{i-p,p}, N_{i-p+1,p} \ldots N_{i,p} \) of a corresponding element "i". Therefore, we obtain the partial derivative of the two dimensional B-Spline basis respect "x" or "y", the second derivative respect "x" or "y" and the cross derivatives (there are scalar values), that are arbitrarily called "DN_x", "DN_y", "D2N_x", etc., in the code. With them we construct the \( B_u(p_a) \) and \( H_s(p_a) \) local matrices as are defined in the expression \( (87) \), and we call them "B_mn" and "Hs_mn" in the code. Then we collect them in some bigger
matrices called "B_Matrix", "Hs_Matrix". These matrices contains the local matrices for each 2-dimensional non-zero local B-spline basis function in an ordered vertical way (a=1,2...(p+1)·(p+1)) calculated for one gaussian point of one element. As an example we sketch the "B_Matrix":

\[
B_{\text{Matrix}} = \begin{bmatrix}
DN_x & 0 & DN_y \\
0 & DN_y & DN_x \\
& & \\
& & \\
& & \\
& & \\
& & \\
& & \\
& & \\
& & \\
\end{bmatrix}
\]

As you can see the numeration of "a" doesn't correspond to the global numeration before explained which is represented in the N(x,y) basis but it's the local numeration.

After we have encode the "B_Matrix" that corresponds to the \( B_u(p_a) \) calculated for all the local basis of an element, we calculate the matrix product: \( B_u(p^a)C_b U^T(p^b) \). This product is executed for one local basis and for one gaussian quadrature point. The advantage of having constructed the "B_Matrix" is that we can obtain this matricial product for all the basis at the same step. The result is called "Mat_loc". As we are calculating this integral \( \int_\Omega e B_u(p^a)C_b U^T(p^b) \) we have to sum all the "Mat_loc"'s for all the gaussian points of the element and multiply it by the corresponding weights of the 2D gaussian point. We do all this in this code expression:

\[
\text{Mat} \_\text{loc} = \text{Mat} \_\text{loc} + B_{\text{Matrix}} \ast C \ast B_{\text{Matrix}}' \ast \text{gweight} \_\text{x}(\text{gp} \_\text{global} \_\text{x}) \ast \text{gweight} \_\text{y}(\text{gp} \_\text{global} \_\text{y})
\]

Now we already have these integrals calculated for all the local 2D basis for all the elements. We have calculated all the's (i.e., all the "Mat_loc"'s). We can illustrate the structure of this matrix:
So the $k_{ab}^e$ that are matrices 2x2 (they have this dimension because we have to consider the control points corresponding to the mechanical displacement in "x", and in "y") are organized in the $k^e$ as we show. In order to assemble this $k_{ab}^e$ to the corresponding position in the global $k$ we compile a variable called temp2 that assign a global index to any local index "a". You can see how is encoded:

$$\text{temp2} = \text{connect}_y(j,n) + (\text{connect}_x(i,m)-1)\times\text{num}_\text{cp}_y;$$

The "temp2" count the global number of control points we have until arrive to the corresponding element of the loop (we have to visualize in our mind the figure (4)).

Finally, we colocate each local $k_{ab}^e$ at the global matrix $K$. In order to compute this term $H_s(p^a)hH_s^T(p_b)$ and add it to the global matrix we repeat the same steps described in this file and finally, we sum the two contributions.

- Development of a main routine that execute the previous steps giving them the incoming variables required. It obtains the force vector, solves the system $Kd=F$ and draw the solution also. This file is called main_elasticity.m. It creates the discretization into elements of our sample, encoding the two knot vectors. We indicate all the parameters in this file, as the order of our B-splines, the maximum derivative required, the number of elements, etc. After discretize our sample it calls the basis_function.m file, for the purpose of evaluating the B-splines basis functions $N_i, p(x)$ and their derivatives in the set of gaussian quadrature points corresponding to the two knots vector. These values are stored and then, they are introduced as incoming variables calling the create_matrix.m function file. We create the global stiffness matrix $K$. After that we create the force vector in this main file. In order to do that, we only have to impose the appropriate boundary conditions. As we have comment previously, in order to verify that all the numerical method it's fine we reproduce the boundary conditions.
corresponding to the next sketch: a cantilever beam fixed in one end and applying a vertical force in the other as it’s drawn in the figure (40). We eliminate the equations of the components of F that corresponds to the global control points of the fixed surface. We impose all the other components of F are zero except the control point related with the position in which we apply the vertical force. In this component, we put the corresponding value of the force. After that, we solve the system K·d=F obtaining the degrees of freedom of the problem. Using them, we construct the solution multiplying them by their corresponding two-dimensional B-Spline basis leading to our vectorial B-Spline surface solution. First, we obtain the values of our B-Splines surface solution taking advantage of the mesh of gaussian point stored and after that we evaluated in a more extensive mesh of points.

Finally, we draw the mechanical displacement solution in a vectorial graph. Also, we calculate and plot the new position of our beam (sample), adding the displacement to the original position of the beam particles. Playing a little with the parameters, it allows us to see a known phenomenon of a very scientific interest called the size effect of a cantilever beam beanding. We will show this in the next chapter and that allows us to verify the correct implementation of our model. There will be the results.
4. Results that validates the model

In this section we are going to describe some of the interesting results we obtain in this period of work. We focus on two results. The first one is the size effect of a cantilever beam bending, that verifies the correct working of our code. The second one is an additional simulation about surface piezoelectricity in a dielectric beam with two piezoelectric material at the surfaces.

4.1 The size effect of a cantilever beam beanding

This effect apperars when we reduce the size of our beam mantaining the proporcionality of both dimensions (we don't talk about the third dimension because our model is two-dimensional). When we do that, the mechanical stiffness increase. It's the same to say that the deformation or mechanical displacement "u" of our beam is smaller. This effect is revealed when we consider the gradient elasticity term $H_s(p^a)hH_s(p_h)$ in our system equation. Without considering it, the stiffness would be the same for any size. In order to graph the results we can take the values of the next table, used on the article of the reference [7] for single crystals of barium titanate $BaTiO_3$:

<table>
<thead>
<tr>
<th>Y (or E)</th>
<th>$\nu$</th>
<th>L1</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 GPa</td>
<td>0.37</td>
<td>10nm</td>
<td>1E-5N</td>
</tr>
</tbody>
</table>

These values are put in the strain gradient tensor h and in the mechanical stiffness tensor C. Knowing that, we plot in the next page the maximum mechanical displacement "u" as a function of the normalized thickness. For BatiO3 the normalized thickness $h'=1$ corresponds to $h=0.23\mu m$.

We add some insets that plot the new position of the particles of the beam for a specific thickness. The blue colour of the graph is related with the consideration of the strain gradient elasticity term while the red colour is related with its exclusion.
As we can observe, the cantilever beam in which we have considered the strain gradient elasticity term, stays stiffer (blue colour) than the other one (red colour). As the dimensions of our cantilever beam are more bigger, we get closer the classical behaviour in an asymptotic way. The gradient elasticity term is a second order term that's necessary to included when we reach nano-dimensions. Depending the material we study, we would observe this phenom in other values for the thickness.

Figure 22. The size effect of a cantilever beam bending. Maximum mechanical displacement $u$ vs normalized thickness $\frac{L_y}{h'}$. 

$21.5\text{nm}$ $0.28\mu\text{m}$
4.2 An additional model of surface piezoelectricity

In this part, we show an additional simulation to complement with our results. As the other results are only related with the mechanical behaviour of our flexoelectric system of equations, we want to obtain some data about the electromechanical response. In this simulation, we also treat with the electromechanical partial differential equation system studied previously but now we use a different method to solve them. Instead of using the galerkin method associated with B-Splines, we use the the local maximum-entropy (LME) approximants [10]. This method is belonged to the family of mesh-free methods, so it provides an approximation to continuum field equations based on basis functions that do not rely on a mesh or its connectivity. This method instead of develop the weak form of the problem it's based in using the electromechanical enthropy in order to derive the system equation. We don't focus on this method because we would need another thesis work to understand it and it isn't the objective.

Forgetting about the method used, the model proposed consists in a dielectric and elastic sample with a piezoelectric material at their surfaces connected to a voltage source. This would be the configuration to analyze if we extend our code to deal with the electromechanical problem. Only adding the others matrices that involves the electromechanical effects in our discretized system and imposing the right boundary condition allow us to simulate this. We sketch a picture of the configuration to understand it better:

As is shown in the picture, the polarization induced due to the voltage source has different signs in the two piezoelectrics (the polarization vector comes out). It produces an accumulation of positive charge in the upper layer and negative in the other. The idea of this sketched construction is to use the piezoelectric effect to mimic the response produced in a flexoelectric sample under an electric field gradient (converse flexoelectric effect). The response we want to obtain is an strain gradient. Due the fact that the polarization has different signs in the two piezoelectric, it generates different strains in each one. As a consequence, the difference of strains between the two layers
generate a strain gradient, that is what we want.

\[ u/h' \]

![Figure 24. Bending of the cantilever beam simulated with the local maximum-entropy (LME) approximants (values normalized).](image)

We can appreciate the strain gradient observing the bending of the cantilever beam simulated. The colours indicate the values of the potential, being the colour blue related with 0 V and the color red related with 1V.

We have obtain another result of this simulation, that we plot:

![Figure 25.](image)

We calculate the vertical displacement as a function of the thickness of the piezoelectrics layers, all this normalized by the thickness of the sample. We observe that when the sample thickness increase for a fixed $\lambda$, the vertical displacement decrease. This phenomenon is logical if we understand surface piezoelectricity. As we have comment in the theory section, the response of surface piezoelectricity can mimic the bulk flexoelectric effect only for thin-films. Therefore, if we don't accomplish this condition, we will have a weak electromechanical response.
5. Conclusions & future works

In this project, we have focused on the development of a two-dimensional continuum model of continuum flexoelectricity based on a Galerkin method that uses B-Splines basis functions. The main objective of this work has consisted in understanding the model proposed and encoding the numerical method that simulates it in a correct way and it has been accomplished.

One of the firsts motivations of doing the flexoelectric model has been the recent controversy of the flexoelectric field, in which are raising doubts about the contribution of the bulk flexoelectric effect. It’s very difficult to distinguish the contribution of the surface piezoelectricity of the sample with the contribution of the flexoelectric bulk effect. Therefore, we wanted to construct a model capable to resolve the electromechanical equations on a sample which allow us to impose different boundary conditions to represent different configurations, as for example, to consider a flexoelectric sample with two piezoelectric materials at the surface, to connect a source voltage in one of the surfaces, etc., in order to investigate the controversies.

The model suggested has consisted in dividing our sample into various elements, step characteristic of the galerkin method and solving the continuum electromechanical equations (the strong form of the problem) on them:

\[
\begin{align*}

\nabla \cdot \sigma &= 0 \\
\nabla \cdot D &= 0
\end{align*}
\]

where

\[
\begin{align*}
\sigma_{ij} &= \hat{\sigma}_{ij} - \sigma_{ij,k,k} = C_{ijkl} \varepsilon_{kl} - e_{kij} E_k + \mu_{ijk} E_{l,k} \\
D_{ij} &= \hat{D}_{ij} - D_{ij,j} = \kappa_{ij} \varepsilon_{kl} + \epsilon_{ij} E_{j} + \lambda_{ijkl} \varepsilon_{jk,l}
\end{align*}
\]

The solution of these equations are the mechanical displacement "\(u\)" and the electrical potential "\(\phi\)" of the sample.

We have simplify the resultant discretized system of equations solving only for the mechanical part and as a consequence obtaining only the mechanical displacement "\(u\)". To do that, we consider the terms of the discretized system that involve the elastical behaviour of the material and the gradient elasticity term. Doing that, we can’t go as far as to obtain flexoelectric results that allow us to investigate the controversies, but we obtain some interesting mechanical results that verifies the correct development of the numerical method and that allow us to trust on the complete model, the first step to understand the flexoelectric issue.

A part of the mechanical results we have analysed the flexoelectric results obtained
by solving the general electromechanical equations with another numerical method, the local maximum-entropy (LME) approximants (we don't focus on it), and it complements our work.

The main result of our mechanical simulation obtained is the called "size effect of the cantilever beam bending" in which we show that reducing the size of a cantilever beam in a proportional way, the sample becomes stiffer, and as a consequence the mechanical displacement is lower. You can see again the results in the next picture:

![Figure 26](image.png)

Figure 26. The size effect of a cantilever beam bending. Maximum mechanical displacement $u$ vs normalized thickness

In order to get the objective of developing a two-dimensional model of continuum flexoelectricity, it has been necessary to understand three backgrounds of theory that are coupled in this work: the theory of flexoelectricity, the finite element method and the B-Spline basis function theory. We began without having any knowledge about these three theories and we have been required to understand a lot of concepts in a short period of time. Therefore, reaching an acceptable knowledge level in these fields has been one of the achievements of the work.

Finally, we have to resume the conclusions of our work. The first conclusion is that we have achieved the two main goals of the work. First of all, we develop a 2D model of flexoelectricity with a galerkin method associated to B-Splines basis functions. Despite we solve a little variation of it, focusing on the mechanical behaviour of the problem, we get some interesting results: the size effect of the cantilever beam bending. Furthermore, we have analysed some flexoelectric results using another numerical
method to solve the electromechanical equations. Secondly, we have reached a good knowledge level of flexoelectricity theory, the finite element method background and the B-Splines background.

We leave as future work solving the system considering the electric part too obtaining the mechanical displacement and the potential solutions. It could be another thesis to study different configurations of a flexoelectric sample solving the continuum flexoelectric system of equations as we have done with the mechanical part and imposing the necessary boundary conditions.

There is another future work that promise so great results [11]. We talk about to connect the microscopic theory of flexoelectricity, (mentioned at the introduction) with the continuum model of flexoelectricity, worked in this thesis. Until now, this theories give us different results. The simulated results of the second one exceeds greatly the theoretical values of the first one. Therefore, solving this controversy and unifying these two points of view of flexoelectricity can help us to do an important step in the flexoelectric field. Progressing in this area is very important to develop more efficient transducer and develop new technologies as the strain gradient engineering commented in its corresponding section of the work.
APPENDIX

OUR NUMERICAL CODE (FILES)

1. evaluate.Bsp.m

```matlab
function [ders,sp]=evaluate_BSp(U,p,de,x);
%
%This function evaluate the values of shape functions and derivatives
%at a given point
%
m = length(U)-1;
n = m-p-1;
mm = m+1;
pm = p+1;
%
%Computing the span
if x>=U(n+1)-(U(n+1)*1e-9);
    sp = n+1;
else
    b=x>=U;
    b(b==0)=[];
    sp=(length(b));
end
%
%Computing Shape functions
Nshape = zeros(pm,pm);
left = zeros(1,p);
right = zeros(1,p);
Nshape(1,1) = 1;
%
for j = 1:p
    left(j) = (x-U(sp+1-j));
    right(j) = (U(sp+j)-x);
    saved =0;
    r=0;
    while r<j;
        rm = r+1;
        Nshape(j+1,rm) = right(1,r+1)+left(1,j-r);
        temp = Nshape(rm,j)/Nshape(j+1,rm);
        Nshape(rm,j+1) = saved+right(1,r+1)*temp;
        saved = left(j-r)*temp;
        r=rm;
    end
    Nshape(j+1,j+1) = saved;
end
%
%Computation of derivatives
for j = 0:1:p
    jm = j+1;
    ders(1,jm) = Nshape(jm,pm);
end
for r = 0:1:p
```

63
rm = r+1;
s1 = 1;
s2 = 2;
a(1,1) = 1;
for k = 1:de
    km = k+1;
    d = 0;
    rk = r-k;
    rkm = rk+1;
    pk = p-k;
    pkm = pk+1;
    if r >= k
        a(s2,1) = a(s1,1)/Nshape(pkm+1,rkm);
        d = a(s2,1)*Nshape(rkm,pkm);
    end
    if rk >= -1
        j1 = 1;
    else
        j1 = -rk;
    end
    if (r-1) <= pk
        j2 = k-1;
    else
        j2 = p-r;
    end
    for l = j1:j2
        lm = l+1;
        a(s2,lm) = (a(s1,lm)-a(s1,lm-1))/Nshape(pkm+1,rkm+l);
        d = d+a(s2,lm)*Nshape(rkm+l,pkm);
    end
    if r<= pk
        a(s2,km) = -a(s1,km-1)/Nshape(pkm+1,rm);
        d = d+a(s2,km)*Nshape(rm,pkm);
    end
ders(km,rm) = d;
j = s1-1;
s1 = s2;
s2 = j+1;
end
end

% Multiplying the derivatives by correction factor
ra = p;
for ka = 1:de
    kam = ka+1;
    for ja = 0:p
        jam = ja+1;
        ders(kam,jam) = ders(kam,jam)*ra;
    end
    ra = ra*(p-ka);
end

2. basis_function.m

function [Nshape,dNsh,ddNsh,connect,spv,gweight,gaussp] = basis_functions(U,p,gaussno,de)
\[ a = U'; \]
\[ a(1:p) = []; \]
\[ a(end-p+1:end) = []; \]
\[ b = a; \]
\[ a(length(a)) = []; \]
\[ b(1) = []; \]
\[ [gpoint, gweight_{local}] = gaquad(gaussno); \]
\[ h = (b-a)/2; \]
\[ Nshape = zeros(length(a)*gaussno, (p+1)); \]
\[ dNsh = zeros(length(a)*gaussno, (p+1)); \]
\[ ddNsh = zeros(length(a)*gaussno, (p+1)); \]
\[ gweight = zeros(length(a)*gaussno, 1); \]
\[ connect = zeros(length(a), p+1); \]
\[ iglobal = 0; \]
\[ for i = 1:length(a) \]
\[ \quad for k = 1:gaussno \]
\[ \quad \quad iglobal = iglobal + 1; \]
\[ \quad \quad gweight(iglobal) = gweight_{local}(k) \cdot h(i); \]
\[ \quad \quad gaussp(iglobal) = ((b(i)-a(i))/2) \cdot gpoint_{(k)} + ((a(i)+b(i))/2); \]
\[ \quad \quad [Nsh, sp] = evaluate_{BSp}(U, p, de, gaussp(iglobal)); \]
\[ \quad \quad Nshape(iglobal,:) = Nsh(1,:); \]
\[ \quad \quad dNsh(iglobal,:) = Nsh(2,:); \]
\[ \quad \quad ddNsh(iglobal,:) = Nsh(3,:); \]
\[ \quad \quad spv(iglobal) = sp - p; \]
\[ \quad end \]
\[ \quad connect(i,:) = [i:i+p]; \]
\[ end \]

2.1 Gaquad.m

\begin{verbatim}
function [gpoint, gweight] = gaquad(gaussno);
% Function to assign the gauss points and weights
if gaussno == 2;
    gweight = [1 1];
    gpoint = [(-1/sqrt(3)) (1/sqrt(3))];
end
if gaussno == 3;
    gweight = [(5/9) (8/9) (5/9)];
    gpoint = [(-sqrt(15)/5) 0 (sqrt(15)/5)];
end
if gaussno == 4;
    gweight = [(18-sqrt(30))/36 (18+sqrt(30))/36 (18-sqrt(30))/36 (18+sqrt(30))/36];
    gpoint = [(-sqrt(525+sqrt(30)*70))/35 -(sqrt(525-sqrt(30)*70))/35 (sqrt(525-sqrt(30)*70))/35 (sqrt(525+sqrt(30)*70))/35];
end
if gaussno > 10;
    gweight = ones(gaussno);
    gpoint = [-1:(2/(gaussno-1)) : 1];
end
end

3. create_matrix.m

function Mat_glob = create_matrix(connect_x, connect_y, Y, nu, num_cp, num_cp_x, num_cp_y, num_el_x, num_el_y, gaussno, ...
\[
N_{\text{shape}}, N_{\text{shape}}, dN_{\text{sh}}, dN_{\text{sh}}, ddN_{\text{sh}}, ddN_{\text{sh}}, gweight_x, gweight_y
\]

\[l_1=10^{-9};\]
\[E=100E9;\]
\[c_{11}=E(1-nu)/(1+nu)(1-2*nu);\]
\[c_{12}=E*nu/(1+nu)(1-2*nu);\]
\[c_{44}=(c_{11}-c_{12})/2;\]
\[h=(l_1^2)*[c_{11} 0 0 c_{12} 0 0; 0 c_{11} c_{12} 0 0 0; 0 c_{12} c_{11} 0 0 0; c_{12} 0 0 c_{11} 0 0; 0 0 0 c_{44} 0; 0 0 0 0 0 c_{44}];\]
\[\text{size}_\text{loc} = \text{size}(\text{connect}_x, 2) * \text{size}(\text{connect}_y, 2);\]
\[CC = Y/(1+nu)/(1-2*nu) * [1-nu, nu, 0; nu, 1-nu, 0; 0, 0, 0.5-nu];\]
\%Build stiffness matrix

\[
\text{Mat}_{\text{glob}} = \text{zeros}(\text{num}_\text{cp}*2);
\text{Mat}_{\text{glob strgradient}}=\text{zeros}(\text{num}_\text{cp}*2);
\text{for } i = 1: \text{num}_\text{el}_x
  \text{for } j = 1: \text{num}_\text{el}_y
    \text{Mat}_{\text{loc}} = \text{zeros}(\text{size}_\text{loc}*2);
    \text{Mat}_{\text{loc strgradient}}=\text{zeros}(\text{size}_\text{loc}*2);
    \text{for } k = 1: \text{gauss}_\text{no}
      \text{for } l = 1: \text{gauss}_\text{no}
        \text{gp}_\text{global}_x = (i-1)* \text{gauss}_\text{no} + k;
        \text{gp}_\text{global}_y = (j-1)* \text{gauss}_\text{no} + l;
        \text{B}_{\text{Matrix}} = [];
        \text{Hs}_{\text{Matrix}}=[];
        \text{for } m = 1: \text{size}(\text{connect}_x, 2)
          \text{for } n = 1: \text{size}(\text{connect}_y, 2)
            \text{DN}_x =
            \text{N}_{\text{shape}}(\text{gp}_\text{global}_x, m)\text{N}_{\text{shape}}(\text{gp}_\text{global}_y, n);
            \text{DN}_y =
            \text{N}_{\text{shape}}(\text{gp}_\text{global}_x, m)\text{N}_{\text{shape}}(\text{gp}_\text{global}_y, n);
            \text{D2N}_x =
            \text{dn}_{\text{shape}}(\text{gp}_\text{global}_x, m)\text{dn}_{\text{shape}}(\text{gp}_\text{global}_y, n);
            \text{D2N}_y =
            \text{dn}_{\text{shape}}(\text{gp}_\text{global}_x, m)\text{dn}_{\text{shape}}(\text{gp}_\text{global}_y, n);
            \text{D2N}_{xy} =
            \text{dn}_{\text{shape}}(\text{gp}_\text{global}_x, m)\text{dn}_{\text{shape}}(\text{gp}_\text{global}_y, n);
            \text{D2N}_{yx} =
            \text{dn}_{\text{shape}}(\text{gp}_\text{global}_x, m)\text{dn}_{\text{shape}}(\text{gp}_\text{global}_y, n);
            \text{D2N}_{xy} =
            \text{B}_{\text{Matrix}} = [\text{B}_{\text{Matrix}}; \text{B}_{\text{mn}}];
            \text{Hs}_{\text{Matrix}}= [\text{Hs}_{\text{Matrix}};\text{Hs}_{\text{mn}}];
          \end{for}
        \end{for}
      \end{for}
    \end{for}
  \end{for}
\end{for}
for m = 1:size(connect_x,2)
    for n = 1:size(connect_y,2)
        temp2 = connect_y(j,n) + (connect_x(i,m)-1)*num_cp_y;
        connect_mn = [2*temp2-1, 2*temp2];
        connect_loc = [connect_loc, connect_mn];
    end
end

Mat_glob(connect_loc,connect_loc) = Mat_glob(connect_loc,connect_loc) + Mat_loc;
Mat_glob_strgradient(connect_loc,connect_loc) = Mat_glob_strgradient(connect_loc,connect_loc) + Mat_loc_strgradient;
end
end

Mat_glob=Mat_glob+Mat_glob_strgradient;

4.main_elasticity.m

clear all*

%% Problem definition
Lx = 10E-1;
Ly = 1E-1;
Y = 1;
nu = 0.3;
Force = 10;

%% set up discretization
p = 2;
de = 2;
num_el_x = 3;
U_x = [0:Lx/num_el_x:Lx];
list_el_x = 1:num_el_x;
aux = ones(1,p);
U_x = [aux*U_x(1) U_x aux*U_x(end)];
um_el_y = 3;
U_y = [0:Ly/num_el_y:Ly];
list_el_y = 1:num_el_y;
U_y = [aux*U_y(1) U_y aux*U_y(end)];

gaussno = 2;
[Nshape_x,dNsh_x,ddNsh_x,connect_x,spv_x,gweight_x,gaussp_x] = basis_functions(U_x,p,gaussno,de);
[Nshape_y,dNsh_y,ddNsh_y,connect_y,spv_y,gweight_y,gaussp_y] = basis_functions(U_y,p,gaussno,de);
num_cp_x = max(max(connect_x));
num_cp_y = max(max(connect_y));
num_cp = num_cp_x*num_cp_y;

%% Create stiffness matrix
K = create_matrix(connect_x, connect_y, Y, nu, num_cp, num_cp_x, num_cp_y, num_el_x, num_el_y, gaussno, 
    Nshape_x, Nshape_y, dNsh_x, dNsh_y, ddNsh_x, ddNsh_y, gweight_x, gweight_y);

%% Apply boundary conditions
dof_force = ((num_cp_x-1)*(num_cp_y)+1)*2;
F = zeros(size(K,1),1);
F(dof_force) = -Force;
dof_constrained = 1:2*(num_cp_y);
K(dof_constrained,:) = [];
K(:,dof_constrained) = [];
F(dof_constrained) = [];
u = K\F;
u = [zeros(2*(num_cp_y),1);u];

u_top = u(2:2*num_cp_y:end);

u_x = u(1:2:end);
u_y = u(2:2:end);
u_x_field = zeros(length(gaussp_x),length(gaussp_y));
u_y_field = zeros(length(gaussp_x),length(gaussp_y));

figure(99)
hold on
for i = 1:num_el_x
    for j = 1:num_el_y
        connect_loc1 = [];
        for m = 1:size(connect_x,2)
            for n = 1:size(connect_y,2)
                temp2 = connect_y(j,n) + (connect_x(i,m)-1)*num_cp_y;
                connect_loc1 = [connect_loc1; temp2];
            end
        end
        for k = 1:gaussno
            for l = 1:gaussno
                gp_global_x = (i-1)*gaussno + k;
                gp_global_y = (j-1)*gaussno + l;
                N_a = [];
                for m = 1:size(connect_x,2)
                    for n = 1:size(connect_y,2)
                        N_a = [N_a Nshape_x(gp_global_x,m)*Nshape_y(gp_global_y,n)]; %Por aqui;
                    end
                end
                u_x_field(gp_global_x,gp_global_y) = N_a*u_x(connect_loc1);
                u_y_field(gp_global_x,gp_global_y) = N_a*u_y(connect_loc1);
% subplot(2,2,1)
quiver(gaussp_x',gaussp_y',u_x_field', u_y_field')
axis equal

deflection = Force*(Lx/Ly)^3*4/Y;
defl_num = u(dof_force);
disp(deflection)
disp(defl_num)

%x=0 y=1:10; y=0 x=0:1; x=1 y=1:10; y=10 x=0:1
PA=50; %Total number of point where I evaluate
PB=50;
x=(0:Lx/PA:Lx); %There are PA+1 points in x
y=(0:Ly/PB:Ly); %There are PB+1 points in y

u_x_fieldmesh = zeros(length(gaussp_x),length(gaussp_y));
u_y_fieldmesh = zeros(length(gaussp_x),length(gaussp_y));
N_p=[];

%We will have a mesh of (PA+1 x PB+1) points
for i=1:PA+1
    x11=x(i);
    [ders1,sp1]=evaluate_BSp(U_x,p,de,x11);
    elementx(1,i)=sp1-p;
    for j=1:PB+1
        x22=y(j);
        [ders2,sp2]=evaluate_BSp(U_y,p,de,x22);
        elementy(1,j)=sp2-p;
        %elementx(1,i)=sp1-p;
        %elementy(1,j)=sp2-p
        connect_loc2 = [];
        N_p=[];
        for m = 1:size(connect_x,2)
            for n = 1:size(connect_y,2)
                N_p = [N_p ders1(1,m)*ders2(1,n)];
                temp2 = connect_y(elementy(j),n) +
(connect_x(elementx(i),m)-1)*num_cp_y;
    connect_loc2 = [connect_loc2; temp2];
end
end
u_x_fieldmesh(i,j) = N_p*u_x(connect_loc2);
u_y_fieldmesh(i,j) = N_p*u_y(connect_loc2);
end
end

% subplot(2,2,2)
% quiver(x,y,u_x_fieldmesh', u_y_fieldmesh')

%E I will obtain the plot of the cantilever beam bending.
posfinalx=[];
posfinaly=[];
posxvec=[];
posyvec=[];
u_x_vec=[];
u_y_vec=[];
for i=1:PA+1
    for j=1:PB+1
        x11=x(i);
x22=y(j);
posfinalx(i,j)=x11+u_x_fieldmesh(i,j)*1E-5;
posfinaly(i,j)=x22+u_y_fieldmesh(i,j)*1E-5;
    end
    u_x_vec=[u_x_vec u_x_fieldmesh(i,:)];
    u_y_vec=[u_y_vec u_y_fieldmesh(i,:)];
posxvec=[posxvec posfinalx(i,:)];
posyvec=[posyvec posfinaly(i,:)];
end
maxdeformx=max(abs(u_x_vec));
maxdeformy=max(abs(u_y_vec));
maxdeform=max(abs(sqrt(u_x_vec.^2+u_y_vec.^2)));%

% subplot(2,2,3)
% plot(posxvec,posyvec,'o')
Files to obtain the results (the final plots)

5. main_elasticity_function
function [maxdeformx, maxdeformy, maxdeform, posxvec, posyvec] = main_elasticity_function(Lx, Ly)

%% Problem definition
% Lx = 1E-1; %Estos valores pueden modificarse a 10 m Lx y 1m Ly.
Luego se cambia u*1E-4;
% Ly = 1E-1;
% Y = 1;
Y=100E9;
nu = 0.37;
Force = 1E-5;

%% set up discretization
p = 2;
dp = 2;
num_el_x = 3;
U_x = [0:Lx/num_el_x:Lx];
list_el_x = 1:num_el_x;
aux = ones(1,p);
U_x = [aux*U_x(1) U_x aux*U_x(end)];
num_el_y = 3;
U_y = [0:Ly/num_el_y:Ly];
list_el_y = 1:num_el_y;
U_y = [aux*U_y(1) U_y aux*U_y(end)];

gaussno = 2;
[Nshape_x, dNsh_x, ddNsh_x, connect_x, spv_x, gweight_x, gaussp_x] =
basis_functions(U_x, p, gaussno, dp);
[Nshape_y, dNsh_y, ddNsh_y, connect_y, spv_y, gweight_y, gaussp_y] =
basis_functions(U_y, p, gaussno, dp);
num_cp_x = max(max(connect_x));
num_cp_y = max(max(connect_y));
num_cp = num_cp_x*num_cp_y;

%% Create stiffness matrix
K = create_matrix(connect_x, connect_y, Y, nu, num_cp, num_cp_x,
num_cp_y, num_el_x, num_el_y, gaussno, ...
Nshape_x, Nshape_y, dNsh_x, dNsh_y, ddNsh_x, ddNsh_y, gweight_x,
weight_y);

%% Apply boundary conditions
dof_force = ((num_cp_x-1)*(num_cp_y)+1)*2;
F = zeros(size(K,1),1);
F(dof_force) = -Force;
dof_constrained = 1:2*(num_cp_y);
K(dof_constrained,:) = [];
K(:,dof_constrained) = [];
F(dof_constrained) = [];
u = K\F;
u = [zeros(2*(num_cp_y),1);u];
u_top = u(2:2*num_cp_y:end);

u_x = u(1:2:end);
u_y = u(2:2:end);
u_x_field = zeros(length(gaussp_x),length(gaussp_y));
u_y_field = zeros(length(gaussp_x),length(gaussp_y));

% figure(99)
% hold on

for i = 1:num_el_x
    for j = 1:num_el_y
        connect_loc1 = [];
        for m = 1:size(connect_x,2)
            for n = 1:size(connect_y,2)
                temp2 = connect_y(j,n) + (connect_x(i,m)-1)*num_cp_y;
                connect_loc1 = [connect_loc1; temp2];
            end
        end
        for k = 1:gaussno
            for l = 1:gaussno
                gp_global_x = (i-1)*gaussno + k;
                gp_global_y = (j-1)*gaussno + l;
                N_a = [];
                for m = 1:size(connect_x,2)
                    for n = 1:size(connect_y,2)
                        N_a = [N_a
                        Nshape_x(gp_global_x,m)*Nshape_y(gp_global_y,n)]; %Por aqui;
                    end
                end
                u_x_field(gp_global_x,gp_global_y) = N_a*u_x(connect_loc1);
                u_y_field(gp_global_x,gp_global_y) = N_a*u_y(connect_loc1);
            end
        end
    end
end

% subplot(2,2,1)
% quiver(gaussp_x',gaussp_y',u_x_field', u_y_field')
% axis equal

deflection = Force*(Lx/Ly)^3*4/Y;
defl_num = u(dof_force);
%disp(deflection)
%disp(defl_num)

%MINE
%Solo tengo que identificar en que elemento estoy
%x=0 y=1:10; y=0 x=0:1; x=1 y=1:10; y=10 x=0:1

PA=50; %Numero total de puntos donde avaluaré
PB=50;
x=(0:Lx/PA:Lx); %Hay PA+1 puntos en x
y=(0:Ly/PB:Ly); %Hay PB+1 puntos en y

u_x_fieldmesh = zeros(length(gaussp_x),length(gaussp_y));
u_y_fieldmesh = zeros(length(gaussp_x),length(gaussp_y));
N_p=[];

%Tendremos una malla de (PA+1 x PB+1) puntos

for i=1:PA+1
    x11=x(i);
    [ders1,sp1]=evaluate_BSp(U_x,p,de,x11);
    elementx(1,i)=sp1-p;
    for j=1:PB+1
        x22=y(j);
        [ders2,sp2]=evaluate_BSp(U_y,p,de,x22);
        elementy(1,j)=sp2-p;
        %en que elemento i,j está cada punto?
        %elementx(1,i)=sp1-p;
        %elementy(1,j)=sp2-p
        connect_loc2 = [];
        N_p=[];
        for m = 1:size(connect_x,2)
            for n = 1:size(connect_y,2)
                N_p = [N_p ders1(1,m)*ders2(1,n)];
                temp2 = connect_y(elementy(j),n) +
                (connect_x(elementx(i),m)-1)*num_cp_y;
                connect_loc2 = [connect_loc2; temp2];
            end
        end
        u_x_fieldmesh(i,j) = N_p*u_x(connect_loc2);
        u_y_fieldmesh(i,j) = N_p*u_y(connect_loc2);
    end
end

% subplot(2,2,2)
% quiver(x,y,u_x_fieldmesh', u_y_fieldmesh')

%Ahora obtendré el dibujo de la biga doblada
posfinalx=[];
posfinaly=[];
posxvec=[];
posyvec=[];
u_x_vec=[];
u_y_vec=[];
for i=1:PA+1

    for j=1:PB+1
        x11=x(i);
x22=y(j);
posfinalx(i,j)=x11+u_x_fieldmesh(i,j);%*1E-5;
posfinaly(i,j)=x22+u_y_fieldmesh(i,j);%*1E-5;

    end

    u_x_vec=[u_x_vec u_x_fieldmesh(i,:)];
    u_y_vec=[u_y_vec u_y_fieldmesh(i,:)];
posxvec=[posxvec posfinalx(i,:)];
posyvec=[posyvec posfinaly(i,:)];
end

maxdeformx=max(abs(u_x_vec));
maxdeformy=max(abs(u_y_vec));
maxdeform=max(abs(sqrt(u_x_vec.^2+u_y_vec.^2)));

6. main_results_gradient

N=50;
lnormalized=0.23E-6;
lengthstep=lnormalized/40;

maxdeformatex=zeros(1,N);
maxdeformatey=zeros(1,N);
maxdeformate=zeros(1,N);
Lx=zeros(1,N);
Ly=zeros(1,N);
Lx0=1E-5;
Ly0=1E-8;
i=0;
for j=1:N
    i=i+1;
    % Lx(i)=Lx0*j;
    % Ly(i)=Ly0*j;
    Ly(i)=Ly0+j*lengthstep;
    Lx(i)=(Ly(i)/Ly0)*Lx0;

[maxdeformx, maxdeformy, maxdeform, posxvec, posyvec] = 
main_elasticity_function(Lx(i), Ly(i));

maxdeformatex(i) = maxdeformx;
maxdeformatey(i) = maxdeformy;
maxdeformate(i) = maxdeform;

if \( j = 2 \)
    figure(1)
    subplot(2,1,1)
    plot(posxvec, posyvec, 'o')
end

if \( j = N-2 \)
    subplot(2,1,2)
    plot(posxvec, posyvec, 'o')
end

end

figure(3)
Lynorm = Ly/(0.23E-6);

plot(Lynorm, maxdeformatex, 'o')
title('maxdeformatex')
plot(Lynorm, maxdeformatey, 'o')
title('maxdeformatey')
plot(Lynorm, maxdeformate)
hold on
plot(Lynorm, 3.245E-7, 'red')
title('Maximum mechanical displacement u vs normalized thickness')
plot(Lynorm, 1./maxdeformate, 'o')
title('bendingrigidity')


