Escuela Politécnica Superior de Jaén

Grado en Ingeniería Mecánica

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

SOLVING MECHANICAL CHARACTERIZATION PROBLEMS USING AN INVERSE METHOD: THE VIRTUAL FIELD METHOD

Alumno: David Bonillo Martínez

Tutor: Prof. D. Francisco Alberto Díaz Garrido
Dpto: Departamento de Ingeniería Mecánica y Minera

Julio de 2018

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Ingeniería Mecánica

Don Francisco A. Díaz Garrido, tutor del Trabajo Fin de Grado titulado: Solving mechanical characterization problems using an inverse method: the virtual field method, que presenta David Bonillo Martínez, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Julio de 2018

El alumno: El tutor:

David Bonillo Martínez Francisco A. Díaz Garrido

Table of contents

1. Overview

In this project it is going to be performed a first approach to the Virtual Fields Method as an inverse method for extracting constitutive parameters from full field measurements. To that end, the project will be divided into three different sections.

At first, a Virtual Fields Method solid theoretical basis will be established in order to being able to understand and internalize this method, which will be the key on the subsequent calculations. In addition, it will be exposed the theoretical basis of the main tool used for determining the real specimens strain fields: the Digital Image Correlation.

Then, it is going to be done a study about the VFM performance into different specimens and materials (isotropic, orthotropic and anisotropic), in order to identify the best option for the VFM appliance in every case. These options will be the Manually Defined, Special and Optimized Virtual Fields.

Later, all the results obtained will be contrasted by applying the method on real specimens, in which there will be some sources of noise, which will be identified and it will be intended to reduce its influence through a good VFM usage.

Finally it will be extracted some conclusions about the method performance, identifying its mains advantages, disadvantages, ways of improving and further work.

2. Motivations and objectives

2.1. Motivations

As it is widely known, at any mechanical design of any kind of machine or component the main base lays on the material behaviour. Specifically, the mechanical parameters of the material must be known to simulate using FEM (or any other method) the response of the material to a given load.

Nowadays, most of the methods employed for determining these mechanical parameters use a specimen with a given dimensions, and they cannot be applied to complex shapes. Some years ago, this was not a problem due to the fact that composite materials were not developed yet and most of the designs were made using isotropic materials, whose mechanical parameters were defined and did not excessively change from the test specimen to the final product.

This claim is not true for the case of composite materials, such as carbon fibre or fiberglass. First, it is quite difficult to obtain the same material parameters on two different specimens, even though they are made of the same components (same resin and same fibre quantity, properties and orientation). This is due to the fact that it is almost impossible to obtain the same conditions of temperature, humidity, UV light during the resin curing procedure, as well as some variations that are supposed to be in two runs of the same resin brand and model.

In addition, as they are orthotropic materials their properties cannot be measured in just one test, and sometimes a number of them will be necessary in order to obtain reliable results.

Thus, it seems clear that the classical methods for extracting constitutive mechanical parameters are not appropriate for composite materials. Therefore, a method that is able to measure the mechanical parameters on the finished part is required, which means that it must be able to be applied into complex shapes.

For achieving this, it is possible to employ an inverse method, such as the Virtual Fields Method, which will be employed for the current final year project.

2.2. Objectives

The main objective of the present project is the following one:

Analysis of the Virtual Fields Method as a tool for extracting the mechanical parameters that govern the material behaviour, identifying its main advantages, disadvantages, ways of improving its performance and future work.

This main objective leads to the three next ones:

- Set of a solid theoretical basis in order to being able to use the method on a right way, always trying to reach the optimal configuration before making any test on any of the VFM appliance methods.
- Appliance of the method into shapes whose deformation fields are obtained using FEM. In this first approach, some knowledge and main ideas about the VFM appliance will be acquired. The constitutive mechanical parameters of the material are going to be computed by using manually defined fields and optimized virtual fields. In this case, it is going to be studied isotropic, orthotropic and anisotropic materials.
- Appliance of the method into shapes whose deformation fields are obtained using DIC. In this section, it will be critical the influence of the noise on the constitutive parameters, thus some work will be done in this direction.

3. Justification of the adopted method for developing the present project

As it has been said before, the present project has been divided into three different blocks. In this section, it is going to be justified the method followed in each section.

3.1. Theoretical basis justification

First, it is going to be set a solid theoretical basis, which will be key through this project development. To that end, it will be partially extracted from [1], [2], [3], [4] and [5].

3.2. Low noise tests justification

In the section named "The VFM in low noise tests", the VFM is going to be applied into specimens simulated through the Finite Element Method.

This method is going to be applied through the software Abaqus. This is done since this software provides many possibilities, being able to compute, as well as 2D in-plane loading, other tests such as 3D specimens, bending plates… Regarding into the materials. It is possible to compute isotropic, orthotropic and anisotropic materials in both linear and nonlinear behavior. This means that once it is developed a procedure for extracting the required data, the connection between Abaqus and Matlab-VFM will be established in any other test or material behavior that could be studied in the future.

For the mesh, it is going to be used triangular elements in order to improve and make faster the calculations, since the element number can increase without increasing the nodes quantity. The regions of the specimen will be meshed using the "Free" technique defined in Abaqus.

Regarding to the VFM appliance, it is going to be performed at first a mesh size study in order to being able to claim that the mesh does not affects the results obtained.

Then, the VFM is going to be applied into two different tests: the traction test and the disc compression test. On each test, three different geometries are going to be computed in order to observe the method behavior in different circumstances.

Three ways of applying the VFM are going to be considered: the Manually Defined, Special and Optimized Virtual Fields. Its robustness will be checked by adding a white noise to the strain fields, and then observing the results standard deviation, being the best method the one that shows less dispersion on its results.

The load is not a relevant parameter here since it will be obtained always a linear behavior on the specimen. On the other hand, the noise amplitude will be a relevant parameter because regarding to the assumptions that will be made at section 4.2.9 it must be much lower than the actual strain value. Because of that, it will be added a noise whose amplitude is 2 magnitude orders lower than the actual strains mean value.

Finally, at the end of this section it is going to be developed a Matlab GUI based on the knowledge acquired in this section, which will be helpful during the tests performance since it will be a tool which will allow to apply the VFM easily into these tests.

3.3. Real tests justification

In the section named "The VFM on DIC tests", the VFM is going to be applied into specimens whose strain fields are obtained by using Digital Image Correlation.

First, it is going to be explained the tests performed, and then it will be explained the VFM appliance into these specimens.

3.3.1. Tests justification

Specimens

In this section it is going to be presented the geometries in which the VFM will be applied.

a) Materials

Three materials are going to be tested. Its properties can be seen on the following table:

	E (GPa)	ν	Q_{11} (GPa)	Q_{12} (GPa)	σ_y (MPa)	Source
Polycarbonate Makrolon	2.38	0.38	2.78	1.057	62.05	[7]
Aluminum 2024-T3	73.1	0.33	82.0	27.1	345	[8]
Titanium Grade 2	105	0.37	121.7	45.0	200	[9]

Table 1. Materials mechanical properties

b) Geometries

In this section, it is going to be shown the schemes containing the specimen geometries info. Planes are not presented since the shapes and lengths are not a critical matter in the VFM:

- Polycarbonate Makrolon

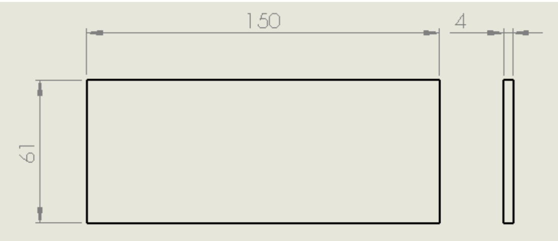


Figure 1. Homogeneous specimen - Polycarbonate

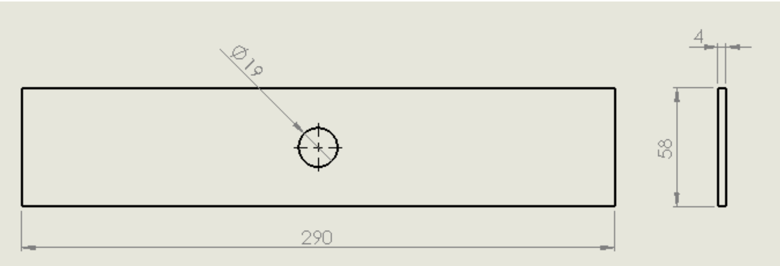


Figure 2. Hole traction specimen - Polycarbonate

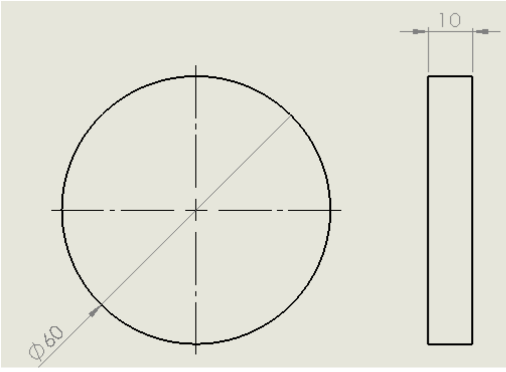


Figure 3. Disc specimen - Polycarbonate

- Aluminium 2024-T3

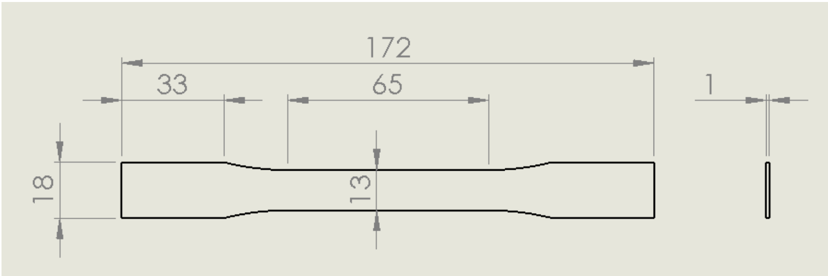


Figure 4. Homogeneous traction specimen - Aluminium

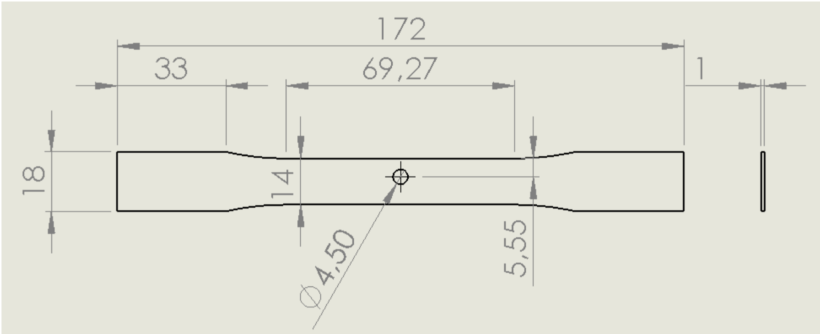


Figure 5. Hole traction specimen - Aluminium

- Titanium grade 2

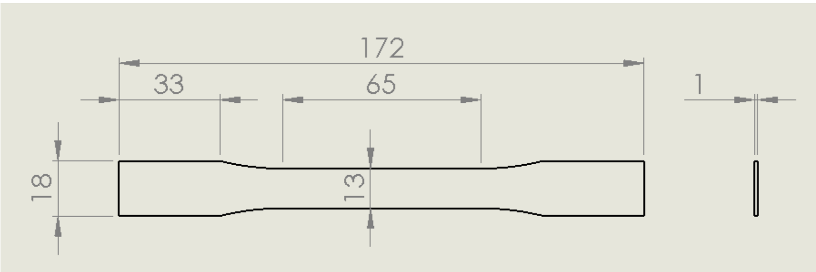


Figure 6. Homogeneous traction specimen - Titanium

c) DIC parameters

The main DIC parameters that are going to be used on the images processing are the following ones:

Material	Test	Subset radius	Subset spacing	Strain radius	µm/pixel
Polycarbonate	Homogeneous traction	40	2	18	63.9
	Turned traction	35	1	18	64.7
	Hole traction	30	1	17	65.8
	Disc under compression	23	1	9	52.5
Aluminium	Homogeneous traction	45	4	25	52.4
	Hole traction	27	3	11	22.2
Titanium	Homogeneous traction	45	4	25	53.2

Table 2. Tests DIC parameters

Tools

The tools that are going to be used during the tests can be seen down below:

- Universal testing machine

The testing machine used during the tests is the "MTS Criterion Series 40 Model 43, with a maximum rated force capacity of 10 kN either under compression or traction, allowing performing all the tests planned.

Figure 7. Universal testing machine MTS Criterion Model 43

- Mechanical grips

The grips used during the tests performance are the MTS model Advantage, being a screw action grips. They can be seen on the following picture:

Figure 8. MTS mechanical grips

- Camera

The camera used during the tests performance is the Allied Vision model Stingray F-504, whose main characteristics can be seen on the following table:

Stingray	F-504
Interface	IEEE 1394b - 800 Mb/s, 2 ports, daisy chain
Resolution	2452 (H) × 2056 (V)
Sensor	Sony ICX655
Sensor type	CCD Progressive
Sensor size	Type 2/3
Pixel size	3.45 µm × 3.45 µm
Lens mount (default)	C-Mount
Max. frame rate at full resolution	9 fps
ADC	14 bit
Image buffer (RAM)	Up to 128 MByte

Table 3. Stingray camera main characteristics

Figure 9. Allied Vision model Stingray F-504

- Camera lens

The camera lens used during the tests performance is the Schneider Kreuznach Xenoplan 1.4/23-0902, whose main characteristic can be seen on the following table:

Technical Specifications	
F-number	1.4
Focal length	22.5 mm
Image circle	11 mm
Transmission	400 - 1000 nm
Interface	C-Mount
Weight	94 gr.
Filter tread	M30.5 x 0.5
Code no.	1001917

Table 4. Schneider lens main characteristics

The lens main view can be seen on the following picture:

Figure 10. Schneider camera lens

- Tripod

The tripod used is the Manfrotto model 475B. It can be seen on the following picture:

Figure 11. Manfrotto 475B

- Camera head

The camera head is a Manfrotto model 410, which provides 3 degrees of freedom that will allow to get a correct camera orientation. This component can be seen on the following picture:

Figure 12. Camera head Manfrotto 410

- Precision lever

The precision level is an E.D.A., which provides a resolution of 5mm/m. It can be seen on the following picture:

Figure 13. Precision level E.D.A.

- Image acquisition software

For the image acquisition it has been used the AVT Smartview 1.11

- Illumination

The set illumination has been made using a regular 60W bulb. It may seem that it is not the best method, but it has proved that by placing this bulb in the correct place and angle it can avoid any reflections in the specimen and provide a constant homogeneous illumination.

Figure 14. Illumination system

- DIC Software

In order to apply DIC it has been used the software Ncorr v1.2.2, a free but powerful software based in the Matlab context. Its usage and main characteristics can be seen on the annex 10.

Figure 15. Ncorr software logo

Test procedure

The test procedure will be the following one:

At first, the specimen will be set on the grips and the camera will be aligned following the procedure described on the following section.

Once the set is prepared, it is taken a first photo on the specimen in which a small load is applied, in order to reduce the rigid body movement. This will be the reference image in the DIC appliance.

After that, some load is applied to the specimen, followed by the image acquisition. To apply that load it has been used the control placed next to the machine, but this process can be improved by programming a test on the computer that controls the machine.

Figure 16. Machine control

The loads applied into each specimen are different regarding each material and geometries, in order no not to overpass the yield tensile strength. The loads applied into each specimen can be seen on the table down below:

Test-material	Reference image load (N)	First load (N)	Last load (N)	Step (N)	Images per step
Traction (Polycarbonate)	100	1000	5000	1000	30
Traction turned (Polycarbonate)	100	5000	5000	-	30
Hole traction (Polycarbonate)	100	1000	3000	1000	30
Disc compression (Polycarbonate)	50	600	6000	600	30
Traction (Aluminium)	50	2000	4000	1000	30
Hole traction (Aluminium)	50	1500	3900	300	20
Traction (Titanium)	50	1000	4000	1000	30

Table 5. Tests loads configuration

- Set alignment:

Now, it is going to be shown and explained the set alignment in the DIC tests.

Figure 17. Set overview

The main aim in this procedure is getting orthotropy to the specimen for achieving the best possible results through the DIC analysis. This procedure will be focused on the camera and specimen alignment.

To that end, it will be useful a precision level, which will allow to orientate the camera and getting it perfectly levelled.

Figure 18. Precision level used

This level will be placed upon the camera, after which we will use the headboard to get a perfect alignment. This headboard can be seen on the following picture:

Figure 19. Headboard

Its joints will be used in this process. First of all the horizontal axis will be levelled (since it must be levelled in order to level the rotational axis as the level cannot be oriented perfectly orthogonal to its first position), after which we will adjust the roll joint on the headboard. Both procedures can be seen on the following pictures:

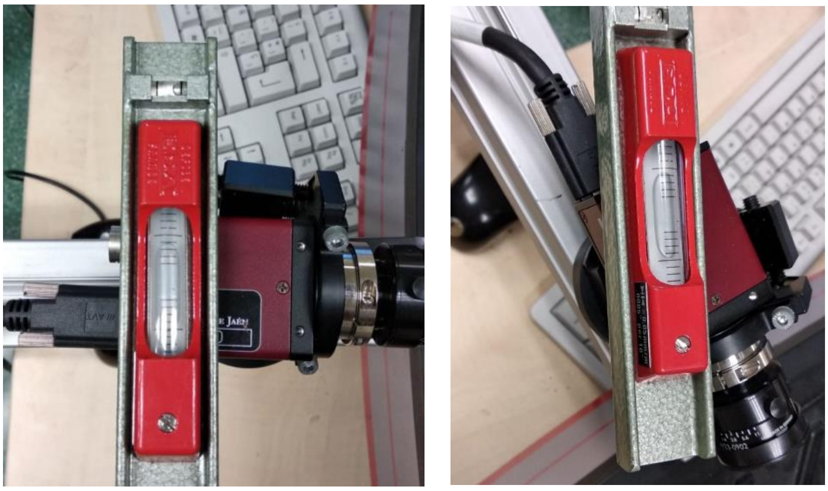


Figure 20. Camera first alignment

Then the camera must be placed looking to the specimen, in a way that allows us to see the specimen perfectly on the monitor ad centred on the image for being able to use the maximum amount of pixels possible.

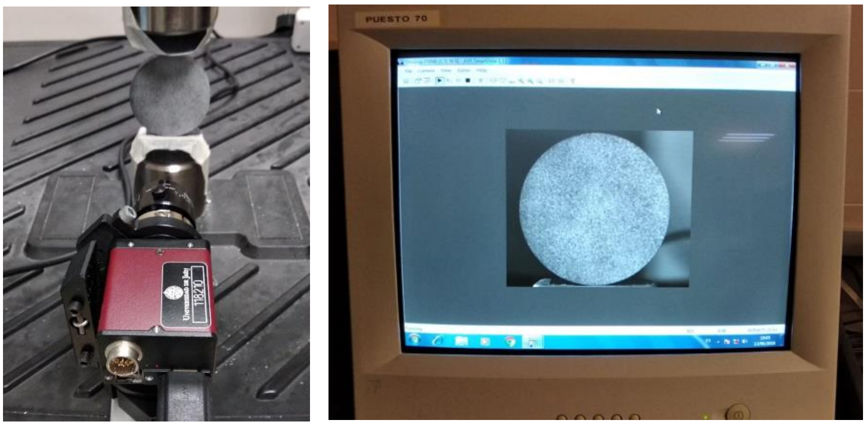
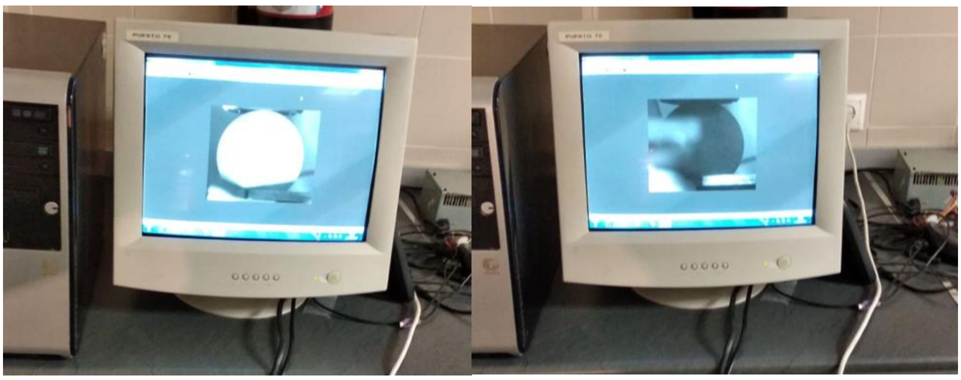


Figure 21. Camera second alignment

After that, the next step is align the disc so that it is perpendicular to the lens. In order to achieve that it is used a try square, which can be seen on the next picture.

Figure 22. Try square

The try square will be pushed against the disc, and we will turn it until the perspective shows that the disc is aligned. The picture on the left shows an unaligned picture, and the one on the right shows a good alignment.



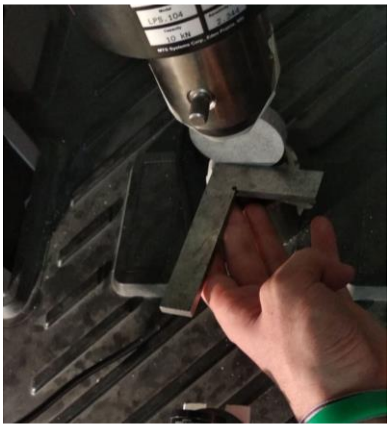


Figure 23. Disc alignment

During the disc alignment process it is mandatory to apply some load on the specimen in order to not moving it too much, causing misalignment on the axis which is aligned with the plane set by the black disc in which the disc lies.

Figure 24. Force applied during the disc alignment process

Next, the camera has to be focused on the specimen. In this process, the image is enlarged to get a better appreciation of the speckle, which will be used to ensure a good focus. Once the image is enlarged, the lens is turned until the speckle image is clear. The following photographs show an unfocused picture (left) and a focused one (right).

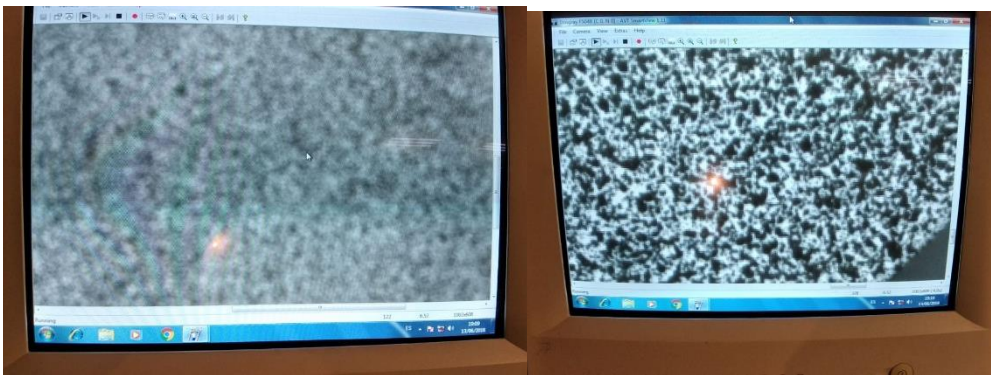


Figure 25. Focus process

Now, it is necessary to set to zero the gain and brightness in the software, since these two parameters increase the noise introduced on the picture through the camera.

Then, the file path must be set in order to save the pictures that will be taken. This will be made on the recording options located on the File section of the software top bar.

Finally, the multishot counter must be set in order to record multiple photos one right after the other. The number of photos taken on each tensional state is specified on the table included on the present section.

3.3.2. VFM appliance justification

First, it is going to be performed a VFM procedure into simulated DIC images in order to make sure that the data is being extracted in a good way from the Ncorr software.

Once that is made, the VFM will be applied into the real tests by using the methods that are considered as the less noise-sensitive ones on the section “The VFM in low noise tests”. This will allow reaching the conclusions of the present project.

4. Precedents and theoretical basis.

4.1. Historical precedents

At the present section, it is going to be presented the precedents evolving the VFM that can be found. They will set the basis in which this project will be developed.

The Virtual Fields Method is a relatively recent topic since it first paper as a method for extracting the constitutive parameters from full field measurements is dated on 2002 [2].

On these first papers it is developed the Special Virtual Fields Method [3] and the Optimized Virtual Fields Method [5], being chosen the Optimized as the best way for applying the method since they are reached through an optimization procedure, without regarding to the method computing tools and its working precision. This will be checked through the present project.

The most remarkable publication about this method is the book *The Virtual Fields Method. Extracting Constitutive Mechanical Parameters from Full-fields Deformation Measurements* [1]. In this book, it is made a compilation of all the articles written by the author since 2002 ([3], [4], [5], for example), turning it as the reference in this method.

Therefore, this project relies on this book, being the theoretical basis partially extracted from it, in order to set a solid basis in which to build the present project.

4.2. VFM theoretical basis

As it is said before, this section is partially extracted from the book *The Virtual Fields Method. Extracting Constitutive Mechanical Parameters from Full-fields Deformation Measurements* (Pierron, F. [2012]) [1]. As explained, this is made for this project to increase its value by adding a solid theoretical basis.

4.2.1. Kinematic: Displacement-Strain equation

[1]. Assuming small deformation conditions after load, the Cartesian strain components are derived from the displacement components using the following equation:

$$\varepsilon_{ij} = \frac{1}{2}(u_{i,j} + u_{j,i}) \tag{1}$$

Or

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\boldsymbol{grad\ u} + \boldsymbol{grat^t\ u}) \tag{2}$$

Where superscript “t” denotes the inverse matrix. The local rotation ω_{ij} is defined in a similar way.

$$\omega_{ij} = \frac{1}{2}(u_{i,j} - u_{j,i}) \tag{3}$$

Or

$$\boldsymbol{\omega} = \frac{1}{2}(\boldsymbol{grad\ u} - \boldsymbol{grad^t\ u}) \tag{4}$$

The displacement boundary conditions consist in writing that this quantity is prescribed over a given portion of S denoted S_u. Thus, one can write

$$\boldsymbol{u} = \overline{\boldsymbol{u}} \text{ over } S_u \tag{5}$$

4.2.2. Constitutive equations

[1]. Hereinafter just the case of linear elasticity will be considered, in which stress and strain are related through the linear constitutive equations. In this case, the stress-strain equations can be written as follows:

$$\sigma_{ij} = C_{ijkl}\varepsilon_{kl} \tag{6}$$

Where C_{ijkl} are the stiffness components, which satisfy three types of symmetry:

$$C_{ijkl} = C_{jikl} = C_{ijlk} = C_{klij} \tag{7}$$

Assuming that the material is isotropic, the C_{ijkl} coefficients only depend on two independent parameters, for instance the Lamé coefficients μ and λ. μ is equal to the shear modulus, which is usually denoted as *G.* Equation (6) becomes in this case

$$\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon} + \lambda\varepsilon_{ii}\boldsymbol{I} \tag{8}$$

Where ***I*** is the unit tensor. These parameters can be expressed in terms of more commonly used constants, Young's modulus *E* and Poisson's ratio *ν*.

$$\begin{cases} \mu = G = \dfrac{E}{2(1+\nu)} \\ \lambda = \dfrac{\nu E}{(1+\nu)(1-2\nu)} \end{cases} \tag{9}$$

Conversely,

$$\begin{cases} E = \dfrac{\mu(3\lambda+2\mu)}{\lambda+\mu} \\ \nu = \dfrac{\lambda}{2(\lambda+\mu)} \end{cases} \tag{10}$$

The particular case of a state of plane stress combined with an isotropic material is often addressed in this project. In this case, writing that $\sigma_{33} = 0$ enables to write ε_{33} as a function of ε_{11} and ε_{22}. Thus:

$$\varepsilon_{33} = -\nu(\varepsilon_{11} + \varepsilon_{22}) \tag{11}$$

The in-plane stresses components can be written as functions of the in-plane stress components as follows:

$$\begin{cases} \sigma_{11} = \dfrac{E}{(1-\nu^2)}(\varepsilon_{11} - \nu\varepsilon_{22}) \\ \sigma_{22} = \dfrac{E}{(1-\nu^2)}(\varepsilon_{22} - \nu\varepsilon_{11}) \\ \sigma_{12} = \dfrac{E}{(1+\nu)}\varepsilon_{12} \end{cases} \tag{12}$$

4.2.3. Strain energy

[1]. The strain energy density $\frac{dW}{dV}$ is defined at any point of the solid by the following quantity:

$$\frac{dW}{dV} = \int_0^{\varepsilon_{ij}} \sigma_{ij}\, d\varepsilon_{ij} \tag{13}$$

In the case of linear elasticity, $\frac{dW}{dV}$ can be written as follows:

$$\frac{dW}{dV} = \frac{1}{2}\sigma_{ij}\varepsilon_{ij} = \frac{1}{2}C_{ij}\varepsilon_{ij}\varepsilon_{kl} \tag{14}$$

Where *W* is the energy stored in the solid after applying the load. This quantity is obtained by integrating the above expression over the volume. In the case of linear elasticity, W is equal to:

$$W = \frac{1}{2}\int_V \sigma_{ij}\varepsilon_{ij}dV = \frac{1}{2}\int_V C_{ijkl}\varepsilon_{kl}\varepsilon_{ij}dV \tag{15}$$

4.2.4. Matrix notation

[1]. The matrix notation will be often used to write equations in this project. The components of ***x*** and ***u*** are denoted as follows:

$$\boldsymbol{x} : \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} \quad \boldsymbol{u} : \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} \tag{16}$$

The third component of the above quantities vanishes in case of 2D problems. Stress and strain components can also be written in a one-column vector, classically introducing the engineering shear strain equal to twice the tensorial shear strain component

$$\boldsymbol{\sigma} : \begin{Bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{23} \\ \sigma_{13} \\ \sigma_{12} \end{Bmatrix} \quad \boldsymbol{\varepsilon} : \begin{Bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{33} \\ 2\varepsilon_{23} \\ 2\varepsilon_{13} \\ 2\varepsilon_{12} \end{Bmatrix} \tag{17}$$

Contracted notation for indices is also employed in this project. Single indices for stress and strain and double indices for stiffness will be followed. In this case, the notation for $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ given above becomes:

$$\boldsymbol{\sigma} : \begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{Bmatrix} \quad \boldsymbol{\varepsilon} : \begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \end{Bmatrix} \tag{18}$$

Index 6 is used to denote the shear stress in case of 2D problems in the $x_1 - x_2$ plane even though indices 3, 4 and 5 vanish in this case. It would be a source of confusion to employ index 3 in this particular case because it denotes the third normal stress in general 3D problems. Hence, the in-plane strain and stress in the 1-2 plane can be written as follows:

$$\boldsymbol{\sigma} : \begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_6 \end{Bmatrix} \quad \boldsymbol{\varepsilon} : \begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_6 \end{Bmatrix} \tag{19}$$

The particular case of a stress state applied on an isotropic material is often addressed in this project. In this case and using the contracted notation, the stress-strain equations reduce to:

$$\begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_6 \end{Bmatrix} = \begin{bmatrix} Q_{11} & Q_{12} & Q_{16} \\ Q_{12} & Q_{22} & Q_{26} \\ Q_{16} & Q_{26} & Q_{66} \end{bmatrix} \begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_6 \end{Bmatrix} \tag{20}$$

Or

$$\boldsymbol{\sigma} = \boldsymbol{Q}\boldsymbol{\varepsilon} \tag{21}$$

Where **Q** is the in-plane stiffness matrix. If the material is orthotropic, Q_{16} and Q_{26} are null in the orthotropy basis of the material, and if the material is isotropic, the following relations exist between the components of ***Q***:

$$\begin{cases} Q_{11} = Q_{22} = \dfrac{E}{1-\nu^2} \\ Q_{12} = \dfrac{\nu E}{1-\nu^2} \\ Q_{66} = \dfrac{Q_{11} - Q_{12}}{2} \end{cases} \tag{22}$$

The equations are merely obtained by comparing (20) and (12) written with contracted indices.

4.2.5. Introduction to the VFM

[1]. Consider the body shown below.

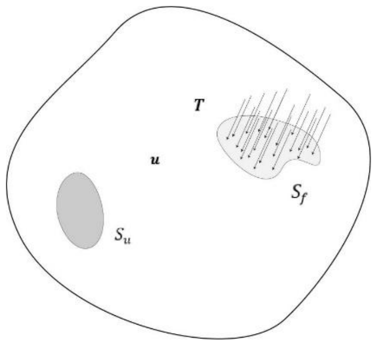


Figure 26. Solid of any shape subjected to mechanical load

It is subjected to applied forces acting on tis boundary ($\boldsymbol{T}$ over S_f) and volume forces acting in its bulk ($\boldsymbol{b}$ over V). If the body moves, a certain distribution of acceleration $\boldsymbol{a}$ takes place. Using the D'Alembert principle, such a distribution induces an additional volume force equal to $-\rho\boldsymbol{a}$.

From purely geometrical considerations and regardless of equilibrium, this body can potentially take an infinity of different configurations that satisfy the displacement boundary conditions. Such configurations induce a certain displacement distribution with respect to the actual location of the points of the body. This displacement is referred to as a virtual displacement. It is denoted $\boldsymbol{u}^*$ in the following. It will be denoted as kinematically admissible (*KA*) if it is null over S_u. It will be shown below that this condition enables to eliminate the traction over S_u, which generally remains unknown in practice.

It must be pointed out that this virtual displacement $\boldsymbol{u}^*$ is different from the actual displacement $\boldsymbol{u}$, which occurs when the load defined above is applied. Within the framework of the small perturbation theory, this actual displacement remains small compared to the dimension of the body. It is measured with respect to the initial configuration which is defined when the body is at rest, whereas the virtual displacement is considered to be an additional one measured with respect to the actual location of the points of the body once the load has been applied. This is the reason why it is often considered as a variation of the actual displacement.

The virtual work W^* done by this load when the body is subjected to this virtual displacement is now considered. Assuming this virtual displacement is C^0, this virtual work is defined as the work done by the load in the virtual displacement field $\boldsymbol{u}^*$.

$$W^* = \int_S \overline{\boldsymbol{T}} \cdot \boldsymbol{u}^* dS + \int_V \boldsymbol{b} \cdot \boldsymbol{u}^* \, dV - \int_V \rho \, \boldsymbol{a} \cdot \boldsymbol{u}^* \, dV \quad \forall \, \boldsymbol{u}^* \, KA \tag{23}$$

The virtual displacement being *KA*, $\boldsymbol{u}^* = 0$ over S_u. Hence, the contribution of the unknown traction over S_u, therefore, vanishes. $\overline{\boldsymbol{T}}$ is related to the stress tensor $\boldsymbol{\sigma}$ over the boundary through Cauchy's formula

$$\overline{\boldsymbol{T}} = \boldsymbol{\sigma n} \tag{24}$$

Where n is the unit vector perpendicular to S_u. Hence, (23) can be rewritten as follows:

$$W^* = \int_S (\boldsymbol{\sigma n}) \cdot \boldsymbol{u}^* dS + \int_V \boldsymbol{b} \cdot \boldsymbol{u}^* \, dV - \int_V \rho \, \boldsymbol{a} \cdot \boldsymbol{u}^* \, dV \quad \forall \, \boldsymbol{u}^* \, KA \tag{25}$$

The first integral on the right hand side of the above equation can be integrated by parts using the divergence theorem:

$$\begin{aligned} \int_S (\boldsymbol{\sigma n}) \cdot \boldsymbol{u}^* dS &= \int_S \sigma_{ij} n_j u_i^* \, dS = \int_V \left(\sigma_{ij} u_i^*\right)_{,j} dV \\ &= \int_V \sigma_{ij} u_{i,j}^* \, dV + \int_V \sigma_{ij,j} u_i^* \, dV \end{aligned} \tag{26}$$

u_{ij}^* can be classically considered as the sum of symmetric and antisymetric tensors.

$$u_{i,j}^* = \frac{1}{2}\left(u_{i,j}^* + u_{j,i}^*\right) + \frac{1}{2}(u_{i,j}^* - u_{j,i}^*) \tag{27}$$

The virtual strain ε_{ij} and the virtual local rotation ω_{ij}^* are derived from the virtual displacement field $\boldsymbol{u}^*$ using the same equations as those used for the actual displacements.

$$\begin{cases} \varepsilon_{ij}^* = \dfrac{1}{2}(u_{i,j}^* + u_{j,i}^*) \\ \omega_{ij}^* = \dfrac{1}{2}(u_{i,j}^* - u_{j,i}^*) \end{cases} \tag{28}$$

Thus,

$$u_{i,j}^* = \varepsilon_{ij}^* + \omega_{ij}^* \tag{29}$$

Equation (26) becomes in this case,

$$\int_S (\boldsymbol{\sigma n}) \cdot \boldsymbol{u}^* dS = \int_V \sigma_{ij}\left(\varepsilon_{ij}^* + \omega_{ij}^*\right) dV + \int_V \sigma_{ij,j} u_i^* \, dV \tag{30}$$

Because of the skew-symmetry of $\boldsymbol{\omega}$ and the symmetry of $\boldsymbol{\varepsilon}$, (20) reduces to:

$$\begin{aligned} \int_S (\boldsymbol{\sigma n}) \cdot \boldsymbol{u}^* dS &= \int_V \sigma_{ij}\varepsilon_{ij}^* \, dV + \int_V \sigma_{ij,j} u_i^* \, dV \\ &= \int_V \boldsymbol{\sigma} : \boldsymbol{\varepsilon}^* \, dV + \int_V (\boldsymbol{div}\,\boldsymbol{\sigma}) \cdot \boldsymbol{u}^* \, dV \end{aligned} \tag{31}$$

Substituting the above expression into (25) and regrouping the last three integrals obtained leads to:

$$W^* = \int_V \boldsymbol{\sigma} : \boldsymbol{\varepsilon}^* \, dV + \int_V (\boldsymbol{div}\,\boldsymbol{\sigma} + \boldsymbol{b} - \rho \boldsymbol{a}) \cdot \boldsymbol{u}^* \, dV \quad \forall\, \boldsymbol{u}^* \, KA \tag{32}$$

If equilibrium is satisfied, the integrand of the second integral in the equation above is null, so the corresponding integral vanishes. Thus,

$$W^* = \int_V \boldsymbol{\sigma} : \boldsymbol{\varepsilon}^* \, dV \qquad \forall\, \boldsymbol{u}^* \, KA \tag{33}$$

Finally, equations (23) and (33) lead to the principle of the virtual work

$$\begin{aligned} -\int_V \boldsymbol{\sigma} : \boldsymbol{\varepsilon}^* \, dV + \int_S \overline{\boldsymbol{T}} \cdot \boldsymbol{u}^* dS + \int_V \boldsymbol{b} \cdot \boldsymbol{u}^* \, dV \\ = \int_V \rho\, \boldsymbol{a} \cdot \boldsymbol{u}^* \, dV \quad \forall\, \boldsymbol{u}^* \, KA \end{aligned} \tag{34}$$

The following quantities are generally introduced:

$$\begin{cases} W_{int}^* = -\int_V \boldsymbol{\sigma} : \boldsymbol{\varepsilon}^* \, dV \ : \text{virtual work done by internal forces, or internal virtual work} \\ W_{ext}^* = \int_S \overline{\boldsymbol{T}} \cdot \boldsymbol{u}^* dS + \int_V \boldsymbol{b} \cdot \boldsymbol{u}^* \, dV \ : \text{virtual work done by the external forces, or} \\ \qquad \text{external virtual work} \\ W_{acc}^* = \int_V \rho\, \boldsymbol{a} \cdot \boldsymbol{u}^* \, dV \ : \text{virtual work done by the acceleration quantities.} \end{cases}$$

Thus,

$$W_{int}^* + W_{ext}^* = W_{acc}^* \quad \forall \boldsymbol{u}^* \ KA \tag{35}$$

4.2.6. In-plane loading

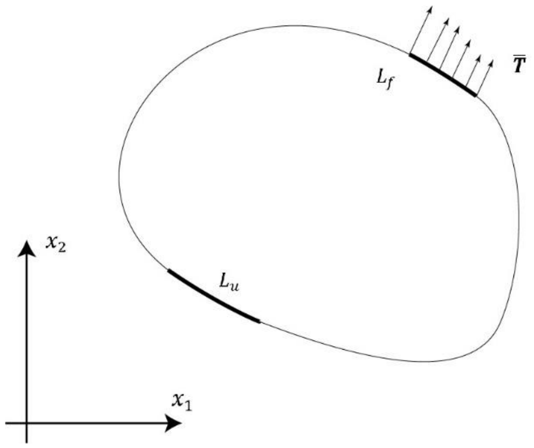


Figure 27. Thin specimen in plane stress

[1]. In this part, the load is assumed to be parallel to the specimen mid-plane denoted (Fig. 2). In this case, and in the absence of in-plane/bending coupling that occurs for certain types of layered structures, $\sigma_1, \sigma_2, \sigma_6$ as well as the acceleration $\boldsymbol{a}$ and the volume force $\boldsymbol{b}$ are supposed constant throughout the thickness of the specimen. If the virtual fields are defined so that u_1^* and u_2^* are independent from x_3, then the direct integration through the thickness of (34) is possible. Therefore, the principle of virtual work becomes as:

$$\begin{gathered} -t\int_S (\sigma_1\varepsilon_1^* + \sigma_2\varepsilon_2^* + \sigma_6\varepsilon_6^*)\, dS + t\int_{L_f} \overline{T}_i u_i^*\, dl \\ + t\int_S b_i u_i^*\, dS = t\int_S \rho a_i u_i^*\, dS \quad \forall \boldsymbol{u}^* \ KA \end{gathered} \tag{36}$$

Where t is the thickness of the specimen. This equation can be simplified by t, thus,

$$\begin{gathered} -\int_S (\sigma_1\varepsilon_1^* + \sigma_2\varepsilon_2^* + \sigma_6\varepsilon_6^*)\, dS + \int_{L_f} \overline{T}_i u_i^*\, dl + \int_S b_i u_i^*\, dS \\ = \int_S \rho a_i u_i^*\, dS \quad \forall \boldsymbol{u}^* \ KA \end{gathered} \tag{37}$$

If the load is quasi-static, $\boldsymbol{a}$ can be neglected. The body force $\boldsymbol{b}$ is also often negligible. In this case, (37) can be written as follows:

$$\int_S (\sigma_1\varepsilon_1^* + \sigma_2\varepsilon_2^* + \sigma_6\varepsilon_6^*)\, dS = \int_{L_f} \bar{T}_i\, u_i^*\, dl \quad \forall\, \boldsymbol{u}^*\ KA \tag{38}$$

This equation represents the principle of virtual work for what is usually called a plane stress problem. It is an important formulation in practice since measurements are usually available on the surface of the specimens under load only with most of the full-field measurement techniques presently available. Hence, strain or displacement measurements performed on the external surface can be assumed representative of through-thickness distributions.

4.2.7. The linear virtual fields method

The flat specimen of any shape shown in fig. 2 is considered again. It is subjected to an in-plane load, so (39) (which is directly deduced from the principle of virtual work) can be directly applied.

$$\int_S (\sigma_1\varepsilon_1^* + \sigma_2\varepsilon_2^* + \sigma_6\varepsilon_6^*) dS = \int_{L_f} \bar{T}_i u_i^*\, dl \qquad \forall \boldsymbol{u}^* KA \tag{39}$$

The first step is to introduce the constitutive equations, through which stress are substituted by strains and stiffness components. For the sake of simplicity and for didactical purposes, the simplest case of isotropic linear elasticity is used here. The stress-strain equation is adopted, and the equalities between stiffness components given in (22) are taken into account. Thus,

$$\begin{aligned}\int_S Q_{11}\varepsilon_1\varepsilon_1^*\, dS + \int_S Q_{11}\varepsilon_2\varepsilon_2^*\, dS + \int_S Q_{12}(\varepsilon_1\varepsilon_2^* + \varepsilon_2\varepsilon_1^*) dS \\ + \int_S \frac{Q_{11} - Q_{12}}{2}\varepsilon_6\varepsilon_6^* dS = \int_{L_f} \bar{T}_i u_i^*\, dl \qquad \forall \boldsymbol{u}^* KA\end{aligned} \tag{40}$$

If the material is assumed homogeneous, thus, the equation above becomes

$$\begin{aligned}Q_{11}\int_S \left(\varepsilon_1\varepsilon_1^* + \varepsilon_2\varepsilon_2^* + \frac{1}{2}\varepsilon_6\varepsilon_6^*\right) dS + Q_{12}\int_S (\varepsilon_1\varepsilon_2^* + \varepsilon_2\varepsilon_1^* - \frac{1}{2}\varepsilon_6\varepsilon_6^*) dS \\ = \int_{l_f} \bar{T}_i u_i^* dl \quad \forall \boldsymbol{u}^* KA\end{aligned} \tag{41}$$

This equation is linear. It is satisfied for *any KA* virtual field. The Virtual Fields Method is based on this peculiar property. Indeed, it is proposed to write this equation with two different *KA* virtual fields, denoted as $\boldsymbol{u}^{*(\boldsymbol{1})}$ and $\boldsymbol{u}^{*(\boldsymbol{2})}$. The corresponding virtual fields are denoted $\boldsymbol{\varepsilon}^{*(1)}$ and $\boldsymbol{\varepsilon}^{*(2)}$, respectively. This leads to the following system of two linear equations.

$$\boldsymbol{A}\,\boldsymbol{Q} = \boldsymbol{B} \tag{42}$$

With

$$\boldsymbol{A}: \begin{bmatrix} \int_S \left(\varepsilon_1 \varepsilon_1^{*(1)} + \varepsilon_2 \varepsilon_2^{*(1)} + \frac{1}{2}\varepsilon_6 \varepsilon_6^{*(1)}\right) dS & \int_S \left(\varepsilon_1 \varepsilon_2^{*(1)} + \varepsilon_2 \varepsilon_1^{*(1)} - \frac{1}{2}\varepsilon_6 \varepsilon_6^{*(1)}\right) dS \\ \int_S \left(\varepsilon_1 \varepsilon_1^{*(2)} + \varepsilon_2 \varepsilon_2^{*(1)} + \frac{1}{2}\varepsilon_6 \varepsilon_6^{*(2)}\right) dS & \int_S \left(\varepsilon_1 \varepsilon_2^{*(1)} + \varepsilon_2 \varepsilon_1^{*(2)} - \frac{1}{2}\varepsilon_6 \varepsilon_6^{*(2)}\right) dS \end{bmatrix} \tag{43}$$

And,

$$\boldsymbol{Q}: \begin{Bmatrix} Q_{11} \\ Q_{12} \end{Bmatrix} \quad \boldsymbol{B}: \begin{Bmatrix} \int_{L_f} \bar{T}_i u_i^{*(1)} dl \\ \int_{L_f} \bar{T}_i u_i^{*(2)} dl \end{Bmatrix} \tag{44}$$

Choosing two independent virtual fields leads the determinant *A* to be different from zero, so the linear system is invertible. Q_{11} and Q_{12} are then directly determined by solving this linear system.

The same approach can be applied if the material is orthotropic. In this case, four unknowns instead of two must be determined. Hence, (22) is not valid anymore, and (20) must be used instead, with Q_{16} and Q_{26} equal to zero here. Equation (39) becomes the following linear equation in which Q_{11}, Q_{22}, Q_{12} and Q_{66} are unknown.

$$\begin{aligned} Q_{11} \int_S \varepsilon_1 \varepsilon_1^* \, dS + Q_{22} \int_S \varepsilon_2 \varepsilon_2^* \, dS + Q_{12} \int_S (\varepsilon_1 \varepsilon_2^* + \varepsilon_2 \varepsilon_1^*) dS \\ + Q_{66} \int_S \varepsilon_6 \varepsilon_6^* dS = \int_{L_f} \bar{T}_i u_i^* dl \quad \forall \boldsymbol{u}^* KA \end{aligned} \tag{45}$$

The same procedure as above is applied, but four independent virtual fields $\boldsymbol{u}^{*(\mathbf{1})}$, $\boldsymbol{u}^{*(\mathbf{2})}$, $\boldsymbol{u}^{*(\mathbf{3})}$ and $\boldsymbol{u}^{*(\mathbf{4})}$ instead of two must be chosen. This leads to a system similar to that obtained in (42), but with $\boldsymbol{A}, \boldsymbol{Q}$ and $\boldsymbol{B}$ now defined as follows:

$$\boldsymbol{A}:\begin{bmatrix} \int_S \varepsilon_1\varepsilon_1^{*(1)}dS & \int_S \varepsilon_2\varepsilon_2^{*(1)}dS & \int_S (\varepsilon_1\varepsilon_2^{*(1)}+\varepsilon_2\varepsilon_1^{*(1)})dS & \int_S \varepsilon_6\varepsilon_6^{*(1)}dS \\ \int_S \varepsilon_1\varepsilon_1^{*(2)}dS & \int_S \varepsilon_2\varepsilon_2^{*(2)}dS & \int_S (\varepsilon_1\varepsilon_2^{*(2)}+\varepsilon_2\varepsilon_1^{*(2)})dS & \int_S \varepsilon_6\varepsilon_6^{*(2)}dS \\ \int_S \varepsilon_1\varepsilon_1^{*(3)}dS & \int_S \varepsilon_2\varepsilon_2^{*(3)}dS & \int_S (\varepsilon_1\varepsilon_2^{*(3)}+\varepsilon_2\varepsilon_1^{*(3)})dS & \int_S \varepsilon_6\varepsilon_6^{*(3)}dS \\ \int_S \varepsilon_1\varepsilon_1^{*(4)}dS & \int_S \varepsilon_2\varepsilon_2^{*(4)}dS & \int_S (\varepsilon_1\varepsilon_2^{*(4)}+\varepsilon_2\varepsilon_1^{*(4)})dS & \int_S \varepsilon_6\varepsilon_6^{*(4)}dS \end{bmatrix} \tag{46}$$

$$\boldsymbol{Q}:\begin{Bmatrix} Q_{11} \\ Q_{22} \\ Q_{12} \\ Q_{66} \end{Bmatrix} \quad \boldsymbol{B}:\begin{Bmatrix} \int_{L_f} \bar{T}_i u_i^{*(1)}dl \\ \int_{L_f} \bar{T}_i u_i^{*(2)}dl \\ \int_{L_f} \bar{T}_i u_i^{*(3)}dl \\ \int_{L_f} \bar{T}_i u_i^{*(4)}dl \end{Bmatrix} \tag{47}$$

In the same way, the six unknown parameters of a homogeneous and fully anisotropic material can be determined in a similar manner, by writing the principle of virtual work six times, each time with a virtual work independent from the others.

It clearly appears that a linear system, where the sough parameters are the unknowns, is obtained if the principle of virtual work is written with as many virtual fields as unknowns. This system can be inverted if all the parameters influence the strain field that takes place in the body (this depends on the geometry of the body and on the loading conditions) and if the virtual fields are independent. The main advantage of this approach is that it is applicable to bodies of any shape, and that the sough parameters are identified directly, without any iteration.

4.2.8. Special virtual fields

[3]. As explained previously, the VFM makes use of different virtual fields in order to determine the Q_{ij}s coefficients. The only condition they must fulfil is that they must be *KA*. Taking this all into account, there are an infinite number of virtual fields for each specimen and test. Anyway, equation (42) must always be solved.

Regarding the equation, it seems interesting to find a set of virtual fields that fulfils $\boldsymbol{A} = \boldsymbol{I}$. This leads to a particular property: the fact that the condition number of the matrix $\boldsymbol{A}$ is equal to one. This condition number is related to the degree of independence of the equations and to the sensitivity of the solution of the linear system to errors in the components of $\boldsymbol{B}$. A condition number equal to one means that the solution of (42) is less sensitive to noise in measured loading force. Moreover, the solution of this system is straightforward as can it be seen below:

$$\begin{gathered}\boldsymbol{Q} = \boldsymbol{A}^{-\mathbf{1}}\boldsymbol{B} \\ \boldsymbol{Q} = \boldsymbol{I}^{-\mathbf{1}}\boldsymbol{B} \\ \boldsymbol{Q} = \boldsymbol{B}\end{gathered} \tag{48}$$

To obtain $\boldsymbol{A} = \boldsymbol{I}$ it is needed to have diagonal components of $\boldsymbol{A}$ equal to one and the out-of-diagonal components equal to zero.

If an anisotropic material is considered, regarding equation (43), and taking into account what is exposed above, it seems clear that two virtual fields are required ($\boldsymbol{u}^{*(\mathbf{1})}, \boldsymbol{u}^{*(\mathbf{2})}$) which fulfil:

-For the case of $\boldsymbol{u}^{*(\mathbf{1})}$:

$$\begin{gathered}\int_S \left(\varepsilon_1\varepsilon_1^{*(1)} + \varepsilon_2\varepsilon_2^{*(1)} + \frac{1}{2}\varepsilon_6\varepsilon_6^{*(1)}\right) dS = 1 \\ \int_S \left(\varepsilon_1\varepsilon_2^{*(1)} + \varepsilon_2\varepsilon_1^{*(1)} - \frac{1}{2}\varepsilon_6\varepsilon_6^{*(1)}\right) dS = 0\end{gathered} \tag{49}$$

-For the case of $\boldsymbol{u}^{*(\mathbf{2})}$:

$$\int_S \left(\varepsilon_1\varepsilon_1^{*(2)} + \varepsilon_2\varepsilon_2^{*(1)} + \frac{1}{2}\varepsilon_6\varepsilon_6^{*(2)}\right) dS = 0 \tag{50}$$

$$\int_S \left(\varepsilon_1 \varepsilon_2^{*(1)} + \varepsilon_2 \varepsilon_1^{*(2)} - \frac{1}{2}\varepsilon_6 \varepsilon_6^{*(2)}\right) dS = 1$$

In this case, $\boldsymbol{A} = \boldsymbol{I}$, and therefore $\boldsymbol{Q}$ is directly identified due to the fact that $\boldsymbol{Q} = \boldsymbol{B}$. It seems clear that this approach can be extended either to isotropic materials (defining four virtual displacement fields) or to anisotropic materials (defining six virtual displacement fields).

4.2.9. Noise influence

[5]. The solid shown in figure 27 is considered. The constitutive material is assumed orthotropic. The measured strain components can be decomposed as the sum of their exact values and a white noise. Equation (45) can therefore be rewritten as follows:

$$\begin{aligned} &Q_{11} \int_S (\varepsilon_1 - \gamma\varphi_1)\varepsilon_1^* \, dS + Q_{22} \int_S (\varepsilon_2 - \gamma\varphi_2)\varepsilon_2^* \, dS \\ &\quad + Q_{12} \int_S ((\varepsilon_1 - \gamma\varphi_1)\varepsilon_2^* + (\varepsilon_2 - \gamma\varphi_2)\varepsilon_1^*) dS \\ &\quad + Q_{66} \int_S (\varepsilon_6 - \gamma\varphi_6)\varepsilon_6^* dS = \int_{L_f} \bar{T}_i u_i^* dl \quad \forall \boldsymbol{u}^* KA \end{aligned} \tag{51}$$

Where φ_1, φ_2 and φ_6 represent scalar zero-mean stationary Gaussian generalized processes on R^2 and $\varepsilon_1, \varepsilon_2$ and $\varepsilon_6$the measured strain components, γ is the amplitude of the random variable strain measurements, which can also be considered as the uncertainty of the strain measurements. The noise is assumed to be uncorrelated from one point to another. This is not strictly true because the three strain components are derived from the two in-plane displacement components bearing the noise in practice, but this assumption is needed to keep the process analytical. Equation (48) can be rewritten as follows:

$$Q_{11}\int_S \varepsilon_1\varepsilon_1^*\,dS + Q_{22}\int_S \varepsilon_2\varepsilon_2^*\,dS + Q_{12}\int_S (\varepsilon_1\varepsilon_2^* + \varepsilon_2\varepsilon_1^*)dS + Q_{66}\int_S \varepsilon_6\varepsilon_6^* dS$$
$$-\gamma\left[Q_{11}\int_S \varphi_1\varepsilon_1^*\,dS + Q_{22}\int_S \varphi_2\varepsilon_2^*\,dS + Q_{12}\int_S (\varphi_1\varepsilon_2^* + \varphi_2\varepsilon_1^*)dS\right.$$
$$\left.+ Q_{66}\int_S \varphi_6\varepsilon_6^* dS\right] = \int_{L_f}\bar{T}_i u_i^* dl \quad \forall \boldsymbol{u}^* KA$$

(52)

The principle of virtual work is now applied using four special virtual displacements fields defined by four special virtual fields denoted as $\boldsymbol{u}^{*(\mathbf{1})}, \boldsymbol{u}^{*(\mathbf{2})}, \boldsymbol{u}^{*(\mathbf{3})}$, and $\boldsymbol{u}^{*(\mathbf{4})}$. These fields fulfil equations (49) and (50), rewritten and expanded for an orthotropic specimen. Writing the principle of virtual works with these four special virtual displacement fields leads to the direct identification of Q_{11}, Q_{22}, Q_{12} and Q_{66}, respectively, as explained in the previous section. For instance, $\boldsymbol{u}^{*(1)}$ being a special virtual field which leads to the identification of Q_{11}, one can write:

$$Q_{11}\int_S \varepsilon_1\varepsilon_1^*\,dS = \int_{L_f}\bar{T}_i u_i^* dl + \gamma[Q_{11}\int_S \varphi_1\varepsilon_1^*\,dS + \cdots$$

$$\ldots + Q_{22}\int_S \varphi_2\varepsilon_2^*\,dS + Q_{12}\int_S (\varphi_1\varepsilon_2^* + \varphi_2\varepsilon_1^*)dS + Q_{66}\int_S \varphi_6\varepsilon_6^* dS] \quad \forall \boldsymbol{u}^* KA$$

(53)

Thus

$$Q11 = \gamma\left[Q_{11}\int_S \varphi_1\varepsilon_1^*\,dS + Q_{22}\int_S \varphi_2\varepsilon_2^*\,dS + Q_{12}\int_S (\varphi_1\varepsilon_2^* + \varphi_2\varepsilon_1^*)dS + Q_{66}\int_S \varphi_6\varepsilon_6^* dS\right]$$
$$+ \int_{L_f}\bar{T}_i u_i^* dl$$

(54)

Similar results are obtained with $\boldsymbol{u}^{*(\mathbf{2})}, \boldsymbol{u}^{*(\mathbf{3})}$ and $\boldsymbol{u}^{*(\mathbf{4})}$ to determine Q_{22}, Q_{12} and Q_{66}, respectively.

If noise is present but no taken into account, approximate parameters denoted $Q_{11}^{app}, Q_{22}^{app}, Q_{12}^{app}$ and Q_{66}^{app} are identified. They can be considered as the components of a vector denoted $\boldsymbol{Q}^{app}$. These components are defined by

$$\begin{cases} Q_{11}^{app} = \int_{L_f} \bar{T}_i u_i^{*(1)} dl \\ Q_{22}^{app} = \int_{L_f} \bar{T}_i u_i^{*(2)} dl \\ Q_{12}^{app} = \int_{L_f} \bar{T}_i u_i^{*(3)} dl \\ Q_{66}^{app} = \int_{L_f} \bar{T}_i u_i^{*(4)} dl \end{cases} \tag{55}$$

Coming back to Q_{11}, (51) can be rewritten as follows:

$$Q_{11} = \gamma \left[Q_{11} \int_S \varphi_1 \varepsilon_1^* \, dS + Q_{22} \int_S \varphi_2 \varepsilon_2^* \, dS + Q_{12} \int_S (\varphi_1 \varepsilon_2^* + \varphi_2 \varepsilon_1^*) dS + Q_{66} \int_S \varphi_6 \varepsilon_6^* dS \right] + Q_{11}^{app} \tag{56}$$

Where γ is the amplitude of the noise, which is assumed to be negligible compared to the L^2 norm of the strain components denoted $||\varepsilon_1||, ||\varepsilon_2||$, and $||\varepsilon_6||$, respectively. Thus,

$$\gamma \ll \min(||\varepsilon_1||, ||\varepsilon_2||, ||\varepsilon_6||) \tag{57}$$

If this assumption is satisfied, the above system of equations can be rewritten by substituting the actual values of the Q_{ij}s in the right hand side of (53) by their approximate counterparts. Thus,

$$Q_{11} \approx \gamma \left[Q_{11}^{app} \int_S \varphi_1 \varepsilon_1^* \, dS + Q_{22}^{app} \int_S \varphi_2 \varepsilon_2^* \, dS + Q_{12}^{app} \int_S (\varphi_1 \varepsilon_2^* + \varphi_2 \varepsilon_1^*) dS + Q_{66}^{app} \int_S \varphi_6 \varepsilon_6^* dS \right] + Q_{11}^{app} \tag{58}$$

The objective is now to estimate the variance of each Q_{ij}. This variance can be written as follows:

$$V(Q_{ij}) = E([Q_{ij} - E(Q_{ij})]^2) \tag{59}$$

For instance, the following expression is obtained for Q_{11}:

$$V(Q_{11}) = \gamma^2 E\left(\left[Q_{11}^{app}\int_S \varphi_1\varepsilon_1^*\, dS + Q_{22}^{app}\int_S \varphi_2\varepsilon_2^*\, dS + Q_{12}^{app}\int_S (\varphi_1\varepsilon_2^* + \varphi_2\varepsilon_1^*) dS + Q_{66}^{app}\int_S \varphi_6\varepsilon_6^* dS\right]\right) \tag{60}$$

Because of the discrete nature of the measurement, the integrals above must be discretized. The rectangle method will be used. Taking into account the properties of autocorrelation of functions $\gamma_i, i = 1,2,6$; $V(Q_{11})$ can be rewritten as follows:

$$V(Q_{11}) = \gamma^2\left(\frac{S}{n}\right)^2\left[\left(\left(Q_{11}^{app}\right)^2 + \left(Q_{12}^{app}\right)^2\right)\sum_{i=1}^{n}\left(\varepsilon_1^{*(1)}(M_i)\right)^2 + \left(\left(Q_{22}^{app}\right)^2 + \left(Q_{12}^{app}\right)^2\right)\sum_{i=1}^{n}\left(\varepsilon_2^{*(1)}(M_i)\right)^2 + 2\left(Q_{11}^{app} + Q_{22}^{app}\right)Q_{12}^{app}\sum_{i=1}^{n}\varepsilon_1^{*(1)}\varepsilon_2^{*(1)}(M_i)^2 + \left(Q_{66}^{app}\right)^2\sum_{i=1}^{n}\left(\varepsilon_6^{*(1)}(M_i)\right)^2\right] \tag{61}$$

Therefore, one can write,

$$V(Q_{11}) = \left(\eta^{(1)}\right)^2\gamma^2 \tag{62}$$

Taking into consideration,

$$\left(\eta^{(1)}\right)^2 = \left(\frac{S}{n}\right)^2\left[\left(\left(Q_{11}^{app}\right)^2 + \left(Q_{12}^{app}\right)^2\right)\sum_{i=1}^{n}\left(\varepsilon_1^{*(1)}(M_i)\right)^2 + \left(\left(Q_{22}^{app}\right)^2 + \left(Q_{12}^{app}\right)^2\right)\sum_{i=1}^{n}\left(\varepsilon_2^{*(1)}(M_i)\right)^2 + 2\left(Q_{11}^{app} + Q_{22}^{app}\right)Q_{12}^{app}\sum_{i=1}^{n}\varepsilon_1^{*(1)}\varepsilon_2^{*(1)}(M_i)^2 + \left(Q_{66}^{app}\right)^2\sum_{i=1}^{n}\left(\varepsilon_6^{*(1)}(M_i)\right)^2\right] \tag{63}$$

Finally, one can write:

$$\left(\eta^{(i)}\right)^2 = \left(\frac{S}{n}\right)^2 \boldsymbol{Q^{app}} \cdot \boldsymbol{G^{(i)}} \cdot \boldsymbol{Q^{app}} \qquad (64)$$

The standard deviation being defined as the square root of the variance, the standard deviations of Q_{11}, Q_{22}, Q_{12} and Q_{66} are equal to $\eta^{(1)}\gamma, \eta^{(2)}\gamma, \eta^{(3)}\gamma$, and $\eta^{(4)}\gamma$, respectively. Since the orders of magnitude of the different $Q_{ij}s$ are not equal, especially for highly anisotropic materials, comparing the $\eta^{(i)}s, i = 1 \dots 4$, can be somewhat misleading. In practice, it is therefore more relevant to deal with the coefficients of variation defined by the ratios of the different $\eta^{(k)}, k = 1 \dots 4$, by their corresponding $Q_{ij}s$. Thus, the following ratios $\frac{\eta^{(1)}}{Q_{11}}, \frac{\eta^{(2)}}{Q_{22}}, \frac{\eta^{(3)}}{Q_{12}}$ and $\frac{\eta^{(4)}}{Q_{66}}$ will be used hereinafter.

These coefficients will represent the sensivity to the noise of an identified parameter. This will allow to compare different virtual fields, trying to find the less noise-dependent one. They also give some information about the tests that will fit best with the VFM. At first, it can be seen that the higher the strain-to-noise ratio is, the better the results will be. On the other hand, it can be seen that the results will highly depend on *S/n,* so this value must be as low as possible for reaching better results.

4.2.10. Optimized virtual fields

[1]. It is known that the $\left(\eta^{(i)}\right)^2$ parameters depend on the chosen virtual fields, so they must be chosen carefully. This can be obtained in a systematic manner, by solving the following minimization problem.

The first step is to rewrite $\left(\eta^{(i)}\right)^2$ as follows:

$$\left(\eta^{(1)}\right)^2 = \frac{1}{2} \boldsymbol{Y_1^{*(i)}} \cdot \boldsymbol{H_1} \cdot \boldsymbol{Y_1^{*(i)}} \qquad (65)$$

Where $\boldsymbol{Y_1}$ is the vector containing the $(\mathrm{n}+1)(\mathrm{m}+1)+(\mathrm{p}+1)(\mathrm{q}+1)$ A_{ij} and B_{ij} coefficients to be determined and $\boldsymbol{H_1}$ a square matrix which elements are obtained by rewriting (64) after having introduced the definition of the virtual strain components given by the basis polynomial virtual field used for solving these optimized virtual fields:

$$u_1^* = \sum_{i=0}^{m}\sum_{j=0}^{n} A_{ij}\left(\frac{x_1}{D}\right)^i\left(\frac{x_2}{D}\right)^j$$
$$u_2^* = \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\left(\frac{x_1}{D}\right)^i\left(\frac{x_2}{D}\right)^j \quad (66)$$

Therefore, the virtual strain fields will be:

$$\begin{cases} \varepsilon_1^* = \sum_{i=1}^{m}\sum_{j=0}^{n} A_{ij}\frac{i}{D}\left(\frac{x_1}{D}\right)^{i-1}\left(\frac{x_2}{D}\right)^j \\ \varepsilon_2^* = \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\left(\frac{x_1}{D}\right)^i\frac{j}{D}\left(\frac{x_2}{D}\right)^{j-1} \\ \varepsilon_6^* = \sum_{i=0}^{m}\sum_{j=0}^{n} A_{ij}\frac{j}{D}\left(\frac{x_1}{D}\right)^i\left(\frac{x_2}{D}\right)^{j-1} + \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\left(\frac{x_1}{D}\right)^{i-1}\frac{i}{D}\left(\frac{x_2}{D}\right)^j \end{cases} \quad (67)$$

It can be proved that $\left(\eta^{(i)}\right)^2$ is strictly convex and exhibits a unique minimum. Therefore, the virtual field that minimize the noise influence is unique and can be found by solving a minimization problem. This minimization must however take into account the fact that the field must be *KA*, as well as the speciality condition on each one.

The first condition leads to a certain number of equations denoted n_{ka}, which will depend on the length of $\boldsymbol{Y_1}$ in case it is wanted a body supported on a line (in case the specimen is supported on a point the field will be *KA* with just an equation). The second one will depend on the specimen material since it will be obtained $n_{special} = 2$ in case of an isotropic material, $n_{special} = 4$ in case of an orthotropic material and $n_{special} = 6$ in case of an anisotropic material.

These two types of constrains can be merged, leading to the form shown below:

$$\boldsymbol{A}\boldsymbol{Y}^{*(\boldsymbol{i})} = \boldsymbol{Z}^{(\boldsymbol{i})} \quad (68)$$

Where $\boldsymbol{Z}^{(\boldsymbol{i})}$ is a vector that contains zero on all its positions except the one in which the special condition corresponding to the stiffness component is to be computed. This position must be set to one. The Lagrangian denoted $\mathcal{L}^{(i)}$ associated with this constrained minimization problem can eventually be written as follows:

$$\mathcal{L}^{(i)} = \frac{1}{2} \boldsymbol{Y}^{(i)*} \cdot \boldsymbol{H} \cdot \boldsymbol{Y}^{*(i)} + \boldsymbol{\vartheta}^{(i)} \cdot \left(\boldsymbol{A}\boldsymbol{Y}^{*(i)} - \boldsymbol{Z}^{(i)} \right) \quad (69)$$

Where $\boldsymbol{\vartheta}^{(i)}$ is a vector containing the Lagrange multipliers associated with the two types of constraints defined above. Minimizing $\mathcal{L}^{(i)}$ leads to the following linear system:

$$\begin{bmatrix} \boldsymbol{H} & \boldsymbol{A}^t \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{Y}^{(i)} \\ \boldsymbol{\vartheta}^{(i)} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{0} \\ \boldsymbol{Z}^{(i)} \end{Bmatrix} \quad (70)$$

As these $\left(\eta^{(i)}\right)^2$ parameters depend on the Q_{ij} parameters, facing an iterative problem is present in which the first Q_{ij} parameters must be guessed to find a first set of virtual fields, after which the $Q_{ij}s$ will be computed again, starting by this way the same process. Anyway, this iterative process converges in a couple of loops, so it is solved quickly.

4.3. DIC theoretical basis

[6]. The Digital Image Correlation is a technique used for determining the displacements fields in any specimen, within a Region of Interest (ROI), through the image processing of photos taken while the specimen deforms. The main idea is to obtain a correspondence between material points in the reference (initial undeformed picture) and current (subsequent deformed pictures) configurations.

DIC does this by taking small subsections of the reference image, called subsets, and determining their respective locations in the current configuration. For each subset, it is obtained the displacement through the transformation used to match the location of the subset in the current configuration. Many subsets are picked in the reference configuration, often with a spacing parameter that reduces the computational cost by decreasing the number of subsets that will be computed.

The result is a grid containing the displacements with respect to the initial configuration. Then, as for the VFM it is needed the strain fields, they are obtained by deriving the displacements fields, by using the equations shown in the 2.2.1 section. Therefore, the noise introduced in the displacements fields (by the camera sensor and the DIC analysis) will be magnified in the strain fields through the derivation. This fact will be analysed throughout the present project.

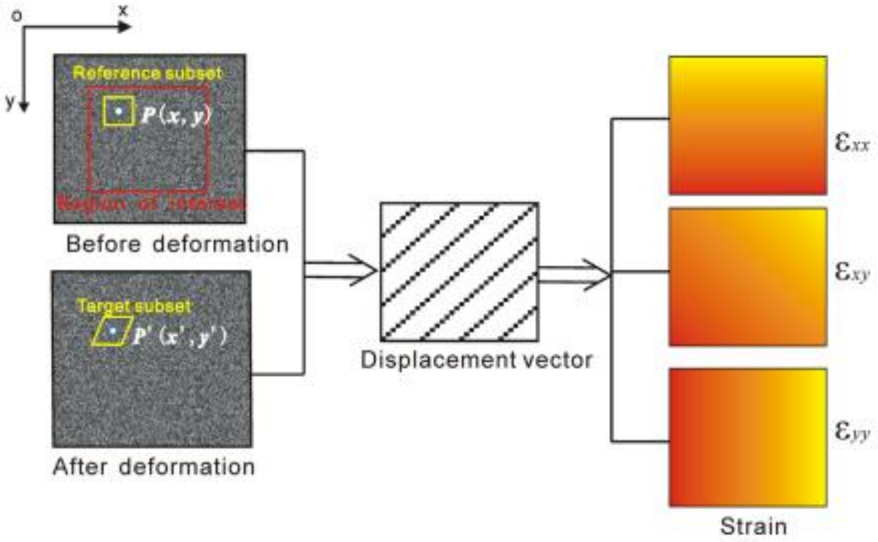


Figure 28. Principle of DIC applied on in-plane strain measurement

The most relevant DIC parameters on the image processing concerning the present project are the following ones. They will be explained from the point of view of the software used in the present project for applying DIC. It is called Ncorr. An overview of this software is done on the annex 10.

Subset radius: It will determine the size of the subset. The biggest the subset is, the less noisy the results will be, even though a big subset will lead to bad results since it will be obtained as the displacement of the subset centre point the average displacement of all the pixels included on the subset.

Subset spacing: it will determine the distance (measured in pixels) between each subset centre. It can be defined in some different ways. In this project it is used Ncorr as DIC software. In it, the subset spacing is defined as the distance in pixels between two subsets centre pixel. It means that if this distance is set to zero, the DIC will be applied in all the pixels included in the ROI. Nevertheless, if this factor is set to one, the software will neglect a pixel between subsets. This leads to a lower computational time.

In this project, this factor will be as low as possible. This is due to the fact that by setting this factor to one it seems obvious that just will be computed one from each four pixels. This means that in order to compute the strains, considering two equal strain radius (parameter that will be explained down below), the noise obtained in the displacements will be much more relevant since 4 times less points will be considered.

Strain radius: it will determine the number of points that will be used to derive the displacements fields in order to achieve the strain fields. The higher this parameter is, the less noisy the results will be since a higher number of points will reduce the noise influence. Nevertheless, a big strain radius will lead to bad results since it will be obtained an average strain on the region as a point strain.

5. The VFM in low noise tests

5.1. First approach to the VFM

The best way of understanding any method is starting by the simplest case. Therefore, it is going to be considered a simple traction test, defined by figure 3. In it, the specimen will be discretized in just one element, since homogenous displacement and strain fields are obtained.

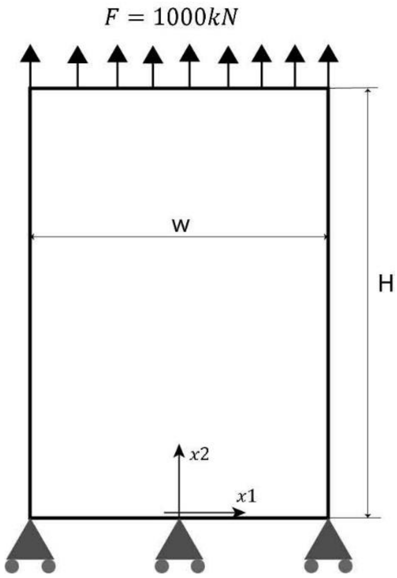


Figure 29. Simple traction test

For this specimen, the dimensions are $w = 1\text{m}, H = 2\text{m}$, and $t = 0.1\text{m}$. This provides a traction pressure of $P = 10\text{MPa}$. For the material, it will be considered $E = 1\text{GPa}$ and $\nu = 0.4$, which gest translated into $\varepsilon_2 = 0.01$, $\varepsilon_1 = -0.004$ and $\varepsilon_6 = 0$, although this value will not be used due to the virtual fields which will be used in this first case.

Once the displacement fields are known on the specimen, it must be determined the virtual fields that will be employed. As said before, the fields must be kinematically admissible. For them being *KA*, they must fulfil a zero vertical displacement in $x2 = 0$. Bearing this in mind, the selected fields are the next ones:

- $\boldsymbol{u}^{*(1)}$:

$$\begin{cases} u_1^{*(1)} = 0 \\ u_2^{*(1)} = x_2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(1)} = 0 \\ \varepsilon_2^{*(1)} = 1 \\ \varepsilon_6^{*(1)} = 0 \end{cases} \tag{71}$$

- $\boldsymbol{u}^{*(2)}$:

$$\begin{cases} u_1^{*(2)} = x_1 \\ u_2^{*(2)} = 0 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(2)} = 1 \\ \varepsilon_2^{*(2)} = 0 \\ \varepsilon_6^{*(2)} = 0 \end{cases} \tag{72}$$

Considering equation (42), (43) and (44) it is obtained:

$$\boldsymbol{A}: \begin{bmatrix} \varepsilon_2\, S & \varepsilon_1\, S \\ \varepsilon_1\, S & \varepsilon_2\, S \end{bmatrix} ; \quad \boldsymbol{B}: \begin{bmatrix} {FH}/{t} \\ 0 \end{bmatrix} \tag{73}$$

By solving the system, it is obtained:

$$\boldsymbol{Q}: \begin{bmatrix} 1.191 \\ 0.476 \end{bmatrix} \text{GPa} \tag{74}$$

Finally, clearing the Young's modulus and the Poisson's ratio, these values are reached:

$$\nu = \frac{Q_{11}}{Q_{12}} = 0.4$$
$$E = Q_{11}(1 - \nu^2) = 1\text{GPa} \tag{75}$$

Which are the ones set previously for the material. By this example, it is intended to provide a first approach to the Virtual Fields Method. It is also intended to consider the importance of the axes definition, since unless they are defined properly they will not be *KA*, and all the calculations will be erroneous. This will be clearer on virtual fields which displacements are defined by a second (or higher) order equation either in x_1 or in x_2 or in both of them.

In addition, it is important to be consistent with the units employed. SI units (N and m) will be adopted in this project, providing the stiffness components in Pa. it is also possible to compute the method using N and mm, which provides $\boldsymbol{Q}$ in MPa.

5.2. The VFM on a traction test

Hereinafter, computer-based simulations will be employed for testing the method. For these simulations, it is used the commercial software Abaqus, in which different geometries, materials, forces… will be defined. This software applies the Finite Element Method (FEM hereinafter), which means dividing the specimen in number of small elements and obtaining after the simulation the stress, strain and displacements in each one.

For applying the VFM, it is needed to know the strain components in each element, and the surface of the element for being able to compute the discretized integrals in equation (43). The process followed for obtaining this information is explained in the annex 1.

Considering this, the first tests can be made.

5.2.1. Dependence of the result on the element size

First, it is going to be observed the dependence between the element size and the results obtained. To that end a traction test on three specimens is defined, which are shown below:

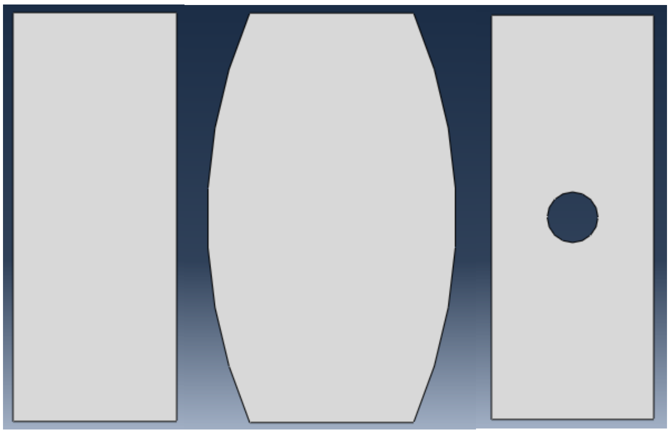


Figure 30. Specimens under test

All the specimens have the same height (0.1 m), thickness (0.001 m), width (0.04m) and are subjected to the same force (8000 N). They will be called "Homogeneous", "Non-homogenous 1" and "Non-homogenous 2" because of its strain fields, which are these ones:

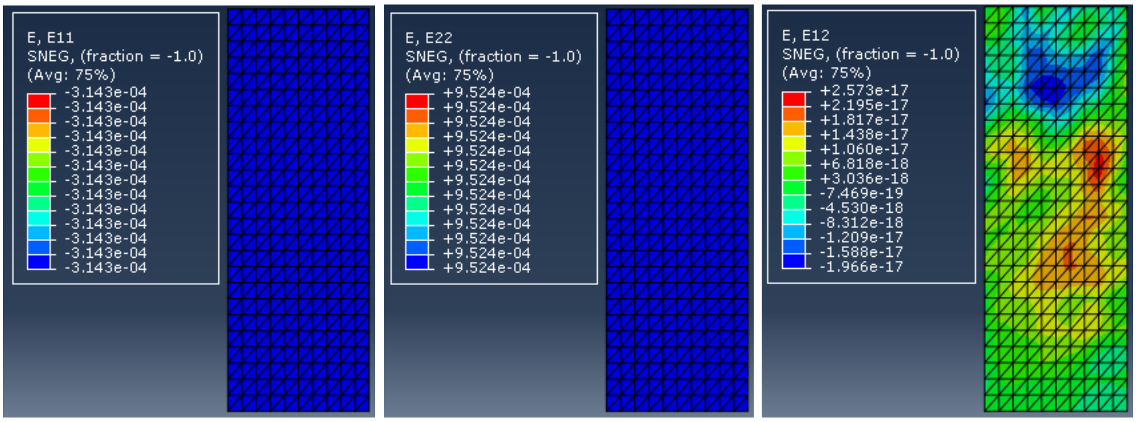


Figure 31. Strain fields on the homogenous specimen

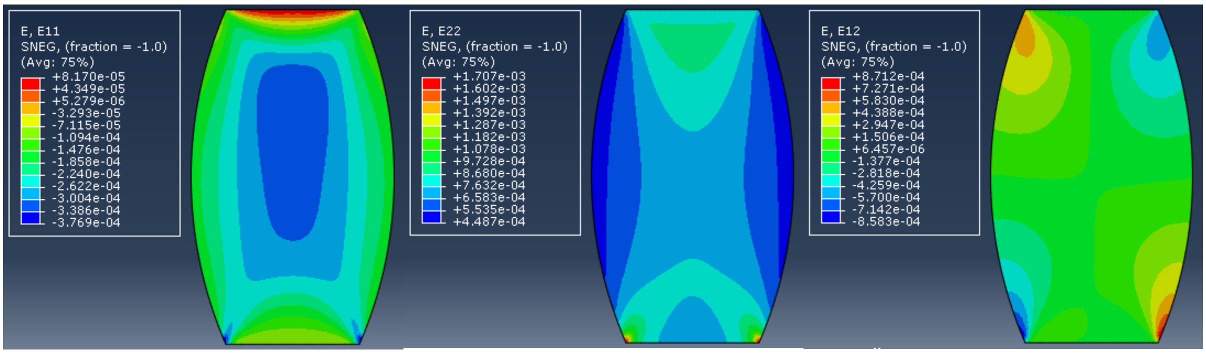


Figure 32. Strain fields on the non-homogeneous 1 specimen

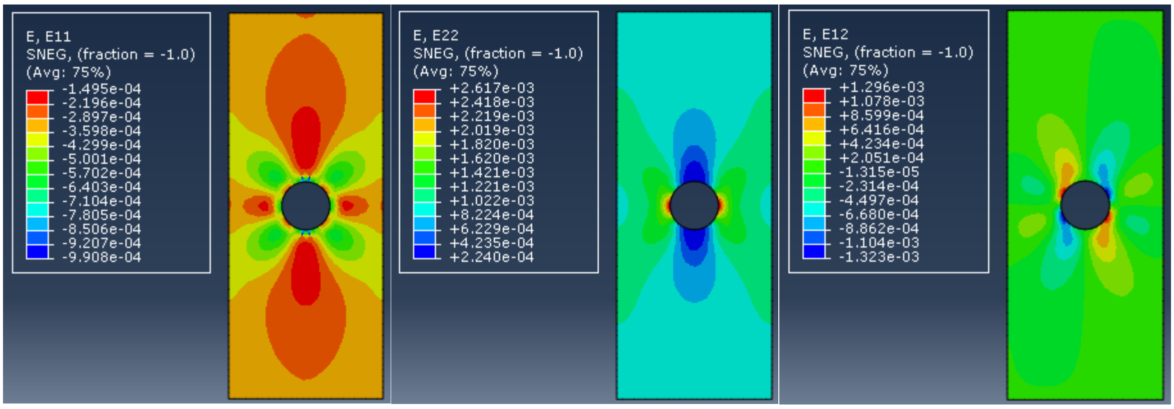


Figure 33. Strain fields on the non-homogeneous 2 specimen

For the first test it will be meshed each specimen using a different element size, getting a report of each result so that later it can be applied the VFM using Matlab. Down below are shown the meshes used on the non-homogeneous 1 specimen.

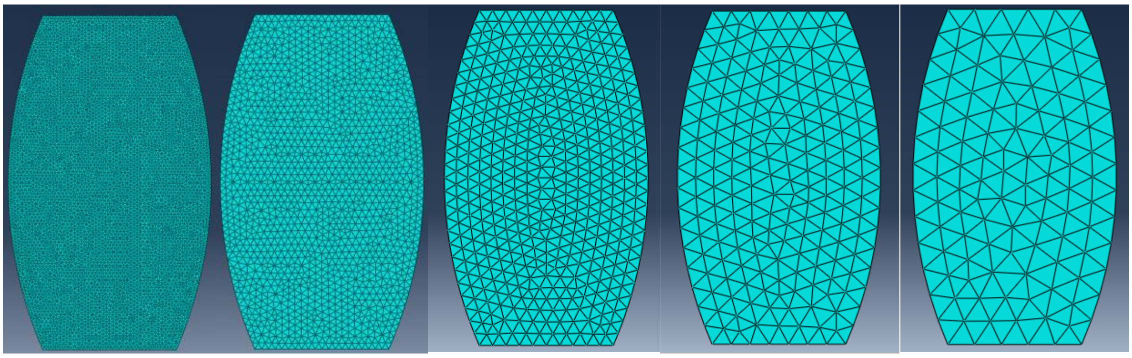
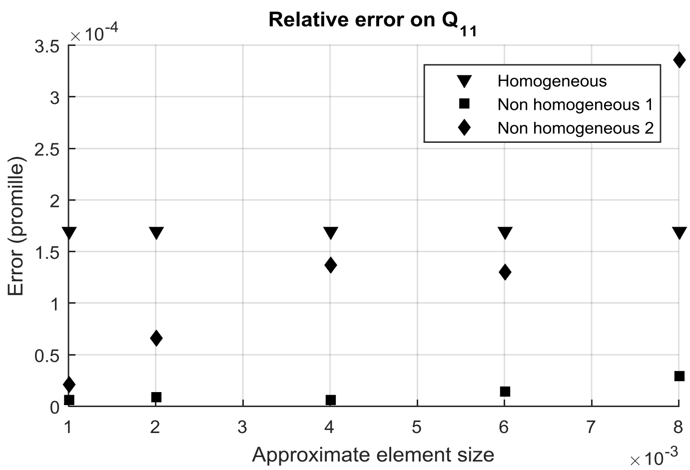


Figure 34. Meshes used in the non-homogeneous 1 specimen for studying the results dependence on the element size

As it is clear, the smaller is the element size, the better it will capture the gradients on the strain fields. Therefore, a lower the strain-to noise ratio is obtained.

Finally, the VFM will be applied on all these tests. For a simpler implementation of the VFM, it will be used the same virtual fields as in the previous section. The code implemented is shown and explained on the annex 2. Down below it can be seen some graphics showing the absolute error on the $\boldsymbol{Q}$ elements (which are known since have been defined previously into Abaqus). As it is an isotropic material just Q_{11} and Q_{12} will be calculated.



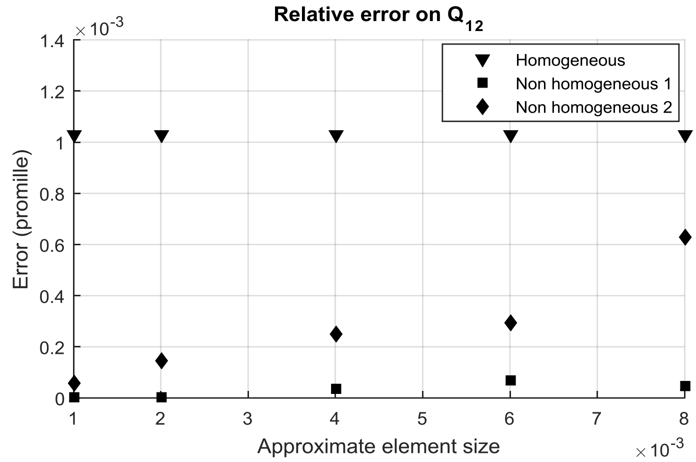
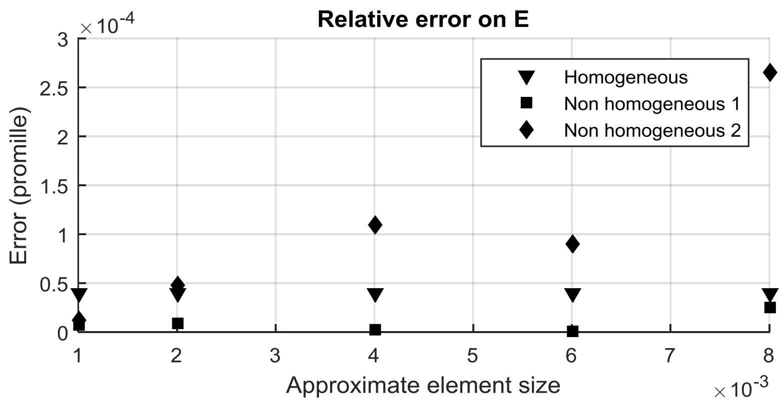


Figure 35. Relative errors on Q calculations depending on the element size

As the relative errors are really small it may appears that it cannot be obtained much information from these two graphics, but some conclusions are reached from them:

- The magnitude order of the relative error on Q_{12} is one unit bigger than the one on Q_{11}, so hereinafter it will be paid attention to this fact.
- When studying a specimen with some gradients, the smaller is the element used in the study, the more accurate the results will be.
- On the specimens with a homogeneous strain fields, the error on the calculations will be constant regardless of the size of the element used on the FEM application. It may be a further discussion what this error relies on.

Now, the same graphics will be made using the most commonly used stiffness components, the Young's Modulus and the Poisson's ratio.



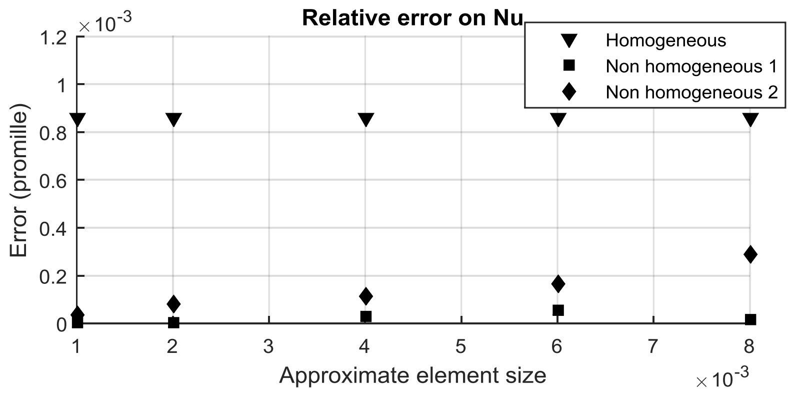


Figure 36. Relative errors on E and v depending on the element size

It is seen the same dependency on these material stiffness components. Therefore, as a conclusion, it seems clear that the smaller the element size is, the more accurate the results will be if the specimen shows non-homogeneous strain fields, especially if the strain fields show strain gradients. This rule is not fulfilled on a homogeneous field, in which the size of the element does not influence the result

5.2.2. Manually defined Virtual Fields on a traction test

On the previous section it has been studied the dependence of the results on the element size, but by making use of just one set of virtual fields. In this section, it will be studied the influence of the element size depending on the set of virtual fields which are chosen. To that end, in addition to the previous set it will be defined the two following ones:

- Second set of virtual fields:
  - $\boldsymbol{u}^{*(3)}$:

$$\begin{cases} u_1^{*(3)} = 0 \\ u_2^{*(3)} = x_2^2 x_1^2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(3)} = 0 \\ \varepsilon_2^{*(3)} = 2x_1^2 x_2 \\ \varepsilon_6^{*(3)} = 2x_1 x_2^2 \end{cases} \tag{76}$$

  - $\boldsymbol{u}^{*(4)}$:

$$\begin{cases} u_1^{*(3)} = x_1^3 \\ u_2^{*(3)} = 0 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(4)} = 3x_1^2 \\ \varepsilon_2^{*(4)} = 0 \\ \varepsilon_6^{*(4)} = 0 \end{cases} \tag{77}$$

$$\boldsymbol{A}^{(2)} = \begin{bmatrix} \int_S (2\varepsilon_2 x_1^2 x_2 + \varepsilon_6 x_1 x_2^2) dS & \int_S (2\varepsilon_1 x_1^2 x_2 - \varepsilon_6 x_1 x_2^2) dS \\ \int_S (3\varepsilon_1 x_1^2) dS & \int_S (3\varepsilon_2 x_1^2) dS \end{bmatrix} \quad (78)$$

$$\boldsymbol{B}^{(2)} = \begin{bmatrix} \frac{FL^2 w^2}{12t} \\ 0 \end{bmatrix} \quad (79)$$

- Third set of virtual fields:
  - $\boldsymbol{u}^{*(5)}$:

$$\begin{cases} u_1^{*(5)} = 0 \\ u_2^{*(5)} = x_2 \cos\left(\frac{\pi x_1}{w}\right) \end{cases}; \quad \begin{cases} \varepsilon_1^{*(5)} = 0 \\ \varepsilon_2^{*(5)} = \cos\left(\frac{\pi x_1}{w}\right) \\ \varepsilon_6^{*(5)} = -x_2 \sin\left(\frac{\pi x_1}{w}\right) \frac{\pi}{w} \end{cases} \quad (80)$$

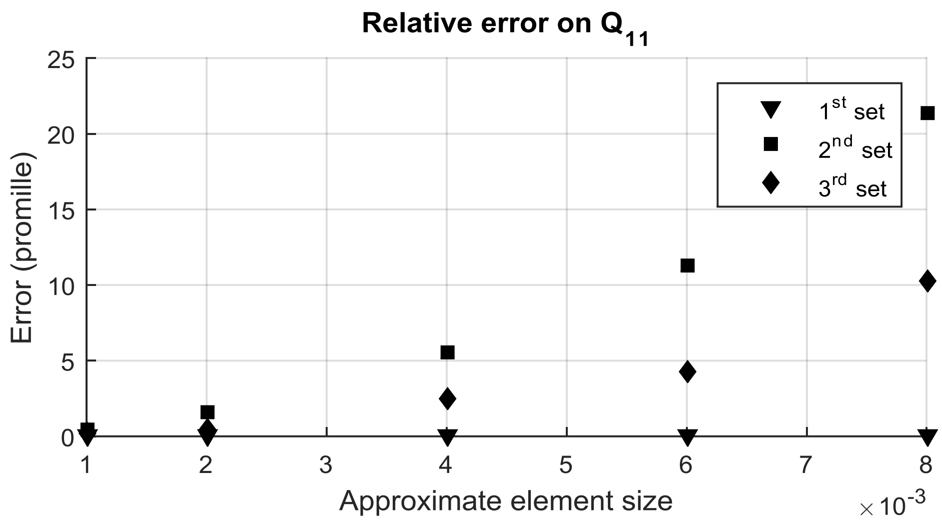
  - $\boldsymbol{u}^{*(6)}$:

$$\begin{cases} u_1^{*(6)} = \sin\left(\frac{2\pi\left(x_1 + \frac{w}{2}\right)}{w}\right) \\ u_2^{*(6)} = 0 \end{cases}; \quad \begin{cases} \varepsilon_1^{*(6)} = \cos\left(\frac{2\pi\left(x_1 + \frac{w}{2}\right)}{w}\right) \frac{2\pi}{w} \\ \varepsilon_2^{*(6)} = 0 \\ \varepsilon_6^{*(6)} = 0 \end{cases} \quad (81)$$

$$\boldsymbol{A}^{(4)} = \begin{bmatrix} \int_S \left(\varepsilon_2 \cos\left(\frac{\pi x_1}{w}\right) - \frac{1}{2}\varepsilon_6 x_2 \sin\left(\frac{\pi x_1}{w}\right)\frac{\pi}{w}\right) dS & \int_S \left(\varepsilon_1 \cos\left(\frac{\pi x_1}{w}\right) + \frac{1}{2}\varepsilon_6 x_2 \sin\left(\frac{\pi x_1}{w}\right)\frac{\pi}{w}\right) dS \\ \int_S \left(\varepsilon_1 \cos\left(\frac{2\pi\left(x_1 + \frac{w}{2}\right)}{w}\right)\frac{2\pi}{w}\right) dS & \int_S \left(\varepsilon_2 \cos\left(\frac{2\pi\left(x_1 + \frac{w}{2}\right)}{w}\right)\frac{2\pi}{w}\right) dS \end{bmatrix} \quad (82)$$

$$\boldsymbol{B}^{(4)} = \begin{bmatrix} \frac{2FL}{\pi t} \\ 0 \end{bmatrix} \quad (83)$$

The code employed on the calculations is shown and explained on the annex 3. The results obtained are the following ones:



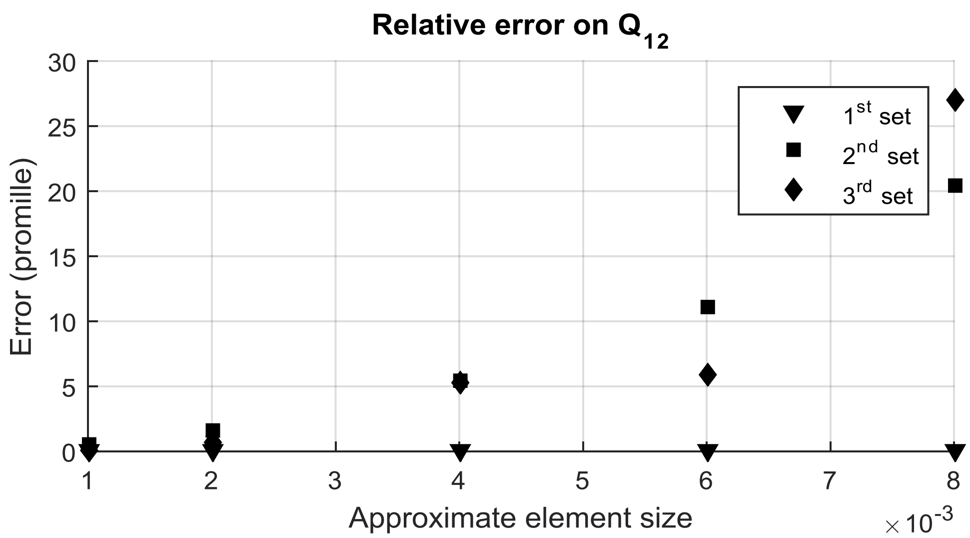


Figure 37. Relative error on Q

As it can be easily seen, the choice of the virtual fields has had an impressive effect on the errors made on the calculations. The second and third sets of virtual sets have increased the errors on 4-6 units. This result will be analysed through another experiment, which will be focused on the effect of the noise depending of the set of virtual fields. To that end, it will be picked the test with the lower approximate element size, and it will be added a random noise on it, varying its amplitude. The code used is shown and explained on the annex 4. In this test it is used a gamma vector composed by 20 elements and 10^5 tests for each element.

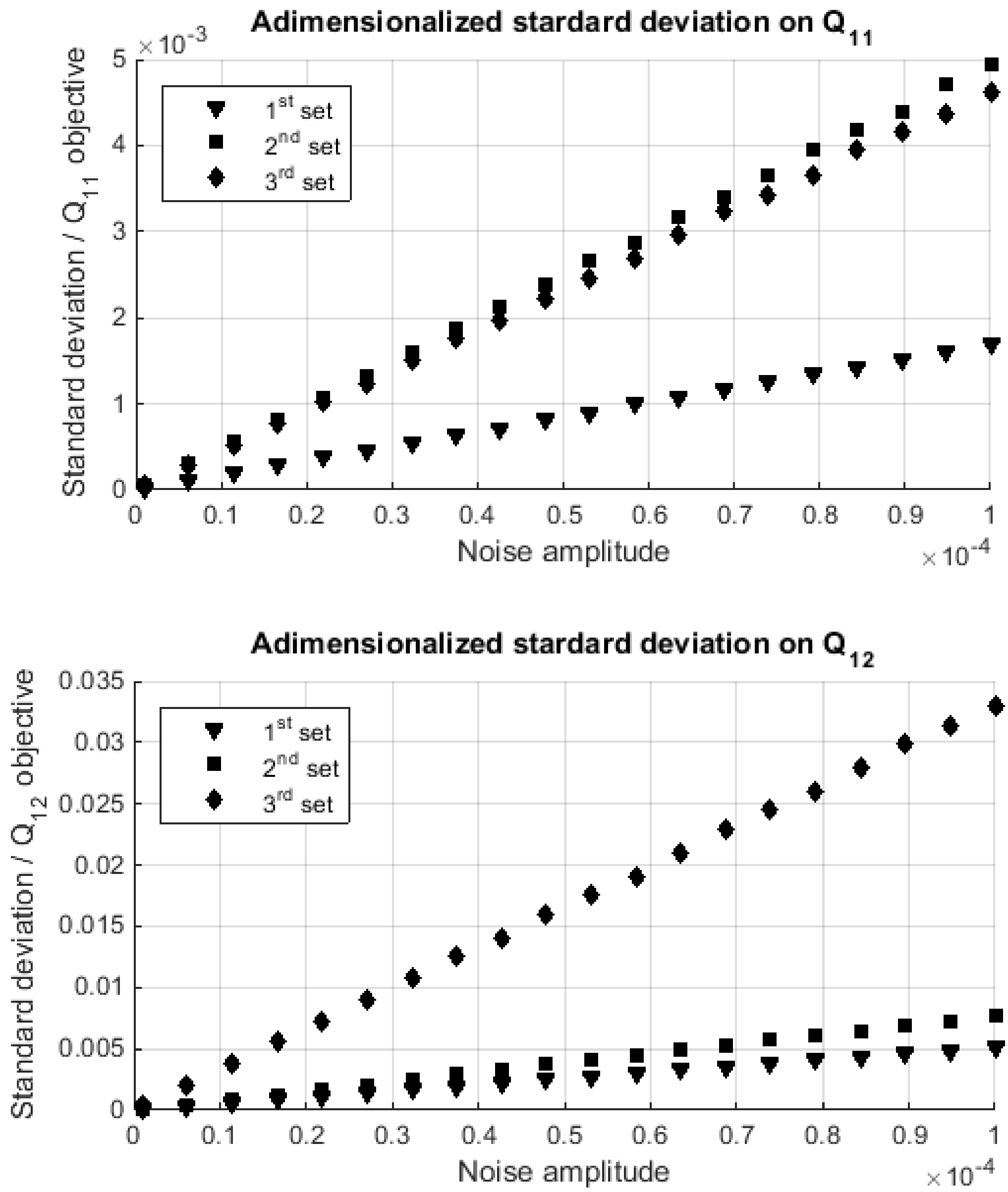
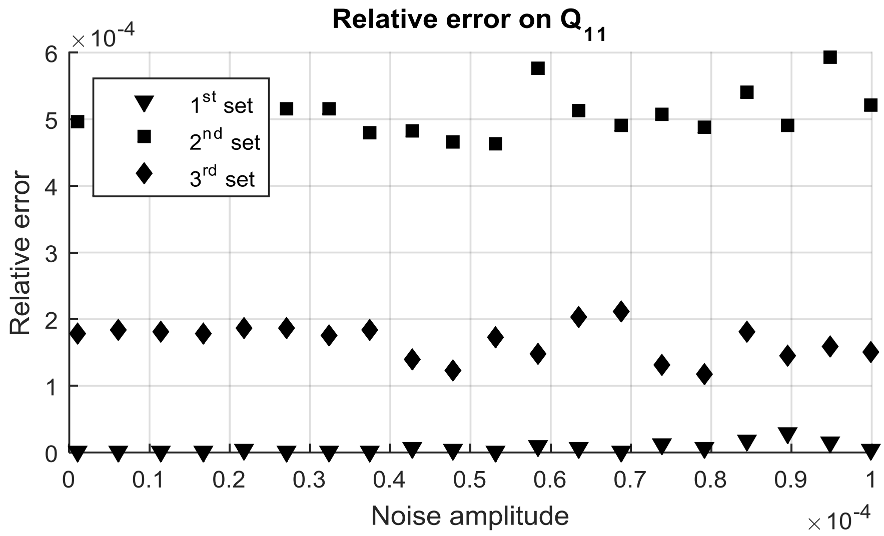


Figure 38. Adimensionalized standard deviation on Q

It seems clear that the standard deviation depends on the selected virtual field, so its correct choice may be important for the correct implementation of the VFM on real test with measured data and computed strain fields, in which the noise may be much broader.

In addition, the graphics down below show the relative error on each component:



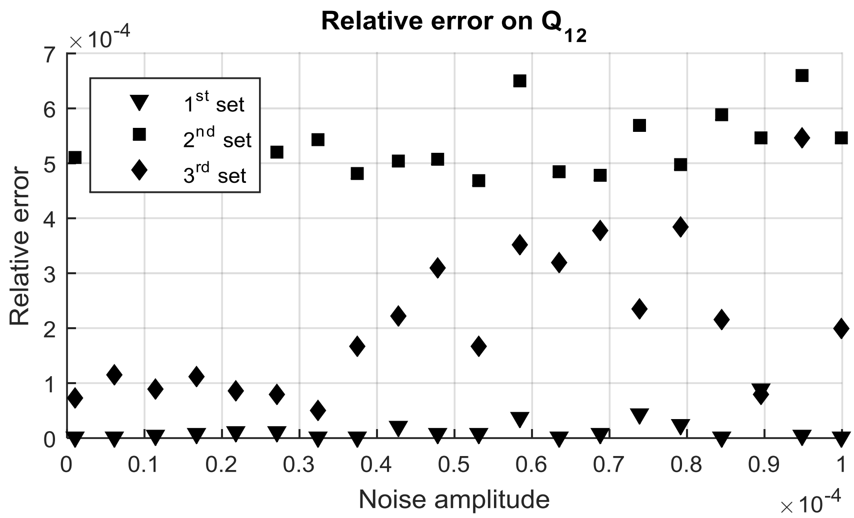


Figure 39. Relative error on Q

It may seem that the results are consistent, but it must be taken into account that for each noise amplitude 10.000 tests have been developed, so the mean value remains constant for each $\boldsymbol{Q}$ component. As on the practice fewer tests will be made, hereinafter this project will focus on decreasing the standard deviation on the computed $\boldsymbol{Q}$ components.

As a conclusion, it must be said that the virtual field definition seems to influence the error obtained on the results, so on the next chapter these virtual fields will be defined automatically, choosing the best one depending on the eta parameters explained on the theoretical basis of this project.

5.2.3. Special Virtual Fields on a traction test

On the previous section, it has been proved the fact that the virtual field choice has a huge impact on the dependence of the results on the noise amplitude. Now, some virtual fields will be generated for a selected specimen, and finally it will be chosen the one that shows less dependence of the results on the noise, by selecting the one with the lower η value. At this point, it seems clear that before reading this section the reader should have read and understood section 4.2.8.

First, the figure 28 must be considered. Taking into account the axes definition, some boundary conditions must be fulfilled for reaching a *KA* virtual field:

1. $u_2 = 0$ in $x_2 = 0$, for having a virtual work equal to zero on that L_u.
2. $u_2 = C$ in $x_2 = L$, for having $\int_{L_f} \overline{T_\iota}\, u_i^* dl = \frac{FC}{t}$
3. $u_1 = 0$ in $x_2 = 0$ and $x_2 = L$, for avoiding the influence of some possible lateral forces on these regions.

At this project, polynomial virtual fields will be used. In this case, the following ones will be used:

$$\begin{cases} u_1^* = \sum_{i=0}^{m}\sum_{j=0}^{n} A_{ij} \frac{x_2(x_2 - L)}{L^2} \left(\frac{x_1}{w}\right)^i \left(\frac{x_2}{L}\right)^j \\ u_2^* = \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij} \frac{x_2}{L} \left(\frac{x_1}{w}\right)^i \left(\frac{x_2}{L}\right)^j \end{cases} \quad (84)$$

It can be easily seen that these virtual displacement fields fulfil conditions 1 and 3. The next step is applying equation (1) for obtaining the virtual strain fields. The results obtained are presented down below:

$$\left\{\begin{array}{l} \varepsilon_1^* = \sum_{i=1}^{m}\sum_{j=0}^{n} A_{ij}\frac{x_2(x_2-L)}{L^2}\frac{i}{w}\left(\frac{x_1}{w}\right)^{i-1}\left(\frac{x_2}{L}\right)^{j} \\ \varepsilon_2^* = \sum_{i=0}^{p}\sum_{j=0}^{q}\left(B_{ij}\frac{1}{L}\left(\frac{x_1}{w}\right)^{i}\left(\frac{x_2}{L}\right)^{j} + B_{ij}\frac{x_2}{L}\left(\frac{x_1}{w}\right)^{i}\frac{j}{L}\left(\frac{x_2}{L}\right)^{j-1}\right) \\ \varepsilon_6^* = \sum_{i=0}^{m}\sum_{j=0}^{n}\left(A_{ij}\frac{2x_2-L}{L^2}\left(\frac{x_1}{w}\right)^{i}\left(\frac{x_2}{L}\right)^{j} + A_{ij}\frac{x_2(x_2-L)}{L^2}\left(\frac{x_1}{w}\right)^{i}\frac{j}{L}\left(\frac{x_2}{L}\right)^{j-1}\right)+\cdots \\ \qquad \ldots + \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\frac{x_2}{L}\frac{i}{w}\left(\frac{x_1}{w}\right)^{i-1}\left(\frac{x_2}{L}\right)^{j} \end{array}\right. \quad (85)$$

Now, the third condition defined previously must be fulfilled. To that end, the virtual field must fulfil:

$$\sum_{i=0}^{m} B_{ij} = 0 \ ; \quad j = 1 \ldots n \quad (86)$$

It may be a little confusing to reach that end, but the reader is encouraged to expand u_2^* using $p = q = 2$ and will easily reach to that conclusion.

The last step is to put it all together in a Matlab code. This code is shown and explained on the annex 5. The code is defined as a function which uses as inputs $x_1, x_2, dS, \varepsilon_1, \varepsilon_2, \varepsilon_6, L, t, w, F, m, n$, and gives as an output $Q_{11}, Q_{12}, \eta_{11}/Q_{11}$ and η_{12}/Q_{12}.

Not much to say about the first part of the function, that just applies what is said above and gives the required results. The second part may be more interesting. In it, the η components are tested. To that end, and following equation (59), it is added a zero-mean noise to $\boldsymbol{\varepsilon}$ in 400 tests and finally it is computed experimentally the eta value by dividing the standard deviation by the mean value for each noise amplitude. The experimental value through is computed through a polyfit (which is plotted). This test has been performed on the non-homogeneous specimen 1. Results obtained can be seen on the following graphic.

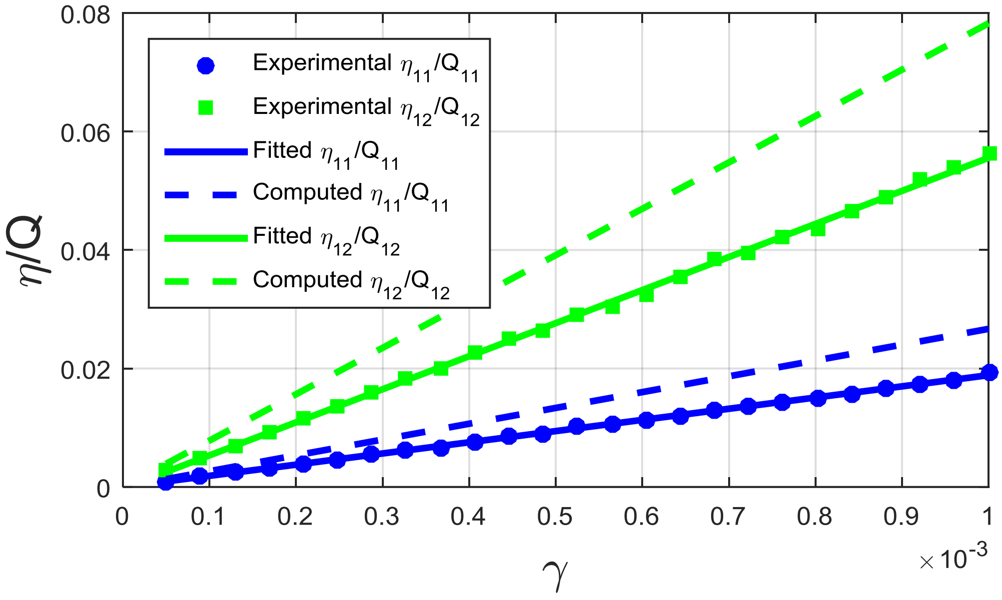


Figure 40. η comparison

Values obtained can be seen on the table down below.

Fitted eta11/Q11 = 18.88	Theoretical eta11/Q11 = 26.7045
Fitted eta12/Q12 = 55.81	Theoretical eta12/Q12 = 78.2919

Table 6. η parameters comparison

Even though some differences between the fitted and theoretical values can be found, both parameters follow similar trends, it must be taken into account the fact that these values are just a reference on how the noise affects to that value, so in practice it will just tell which parameter could be the most noise-depending.

Arrived to this point, some questions seem obvious. The first one is which polynomial degree gets the lower noise dependence, and the second one is if really these automatically generated virtual fields show less noise dependence than the manually defined ones. Both questions will be answered on the two following sections.

Degree dependence

At this section, it is going to be checked which degree is best on each case, for checking if there is some dependence of the η parameters with the chosen degree for the polynomial. While performing this test, it has been noticed that even though there are (m+2) equations (therefore on the A_{ij}s the other parameters will also be equal to zero) just two parameters are the one that are not equal to zero, which allows to make some changes on the code for improving the computational time.

The final results can be seen on the following graphs:

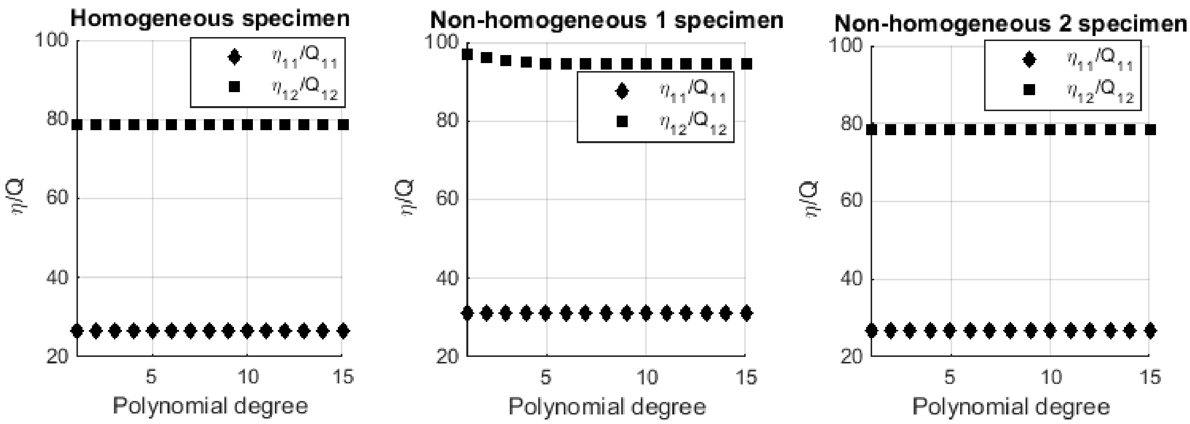


Figure 41. Degree dependence for the special virtual fields on a traction test

As can be seen, there is no dependence on the homogeneous and the non-homogeneous 2 specimen (it may appear that I have used the same graphic for both specimens, but they just are a lot alike), while on the non-homogeneous 1 it seems to be a small dependence which gets stable about degree 6-7. Therefore, on the following section it will be used degree 1 for specimens homogeneous and non-homogeneous 1 for the sake of the computational time, and degree 6 on the non-homogeneous 1 specimen for getting better results without increasing too much the computational time.

Comparison to the manually defined virtual fields

It is going to be compared this special virtual fields with the ones manually defined previously. To that end, will be added to the data, performing tests on 15 different noise amplitudes and 1000 tests on each noise amplitude, trying to find the method that shows less standard deviations on the stiffness components. The results can be seen on the graphics down below.

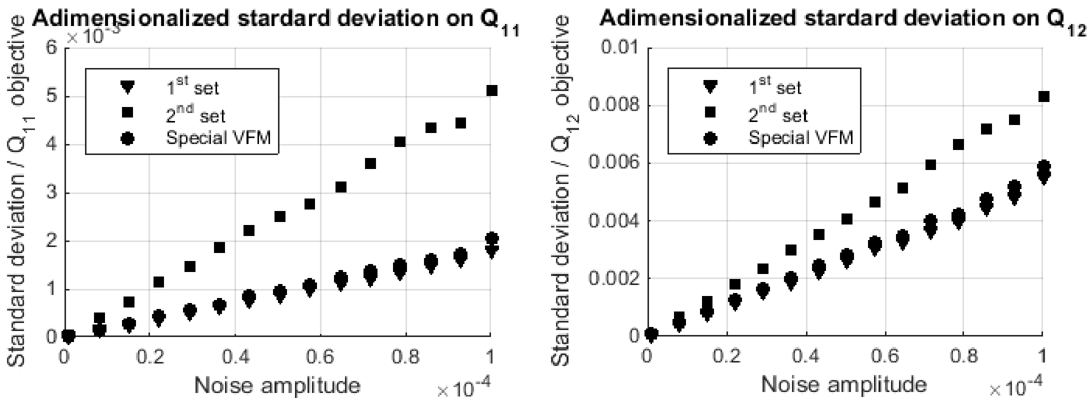


Figure 42. Results obtained in the homogeneous specimen

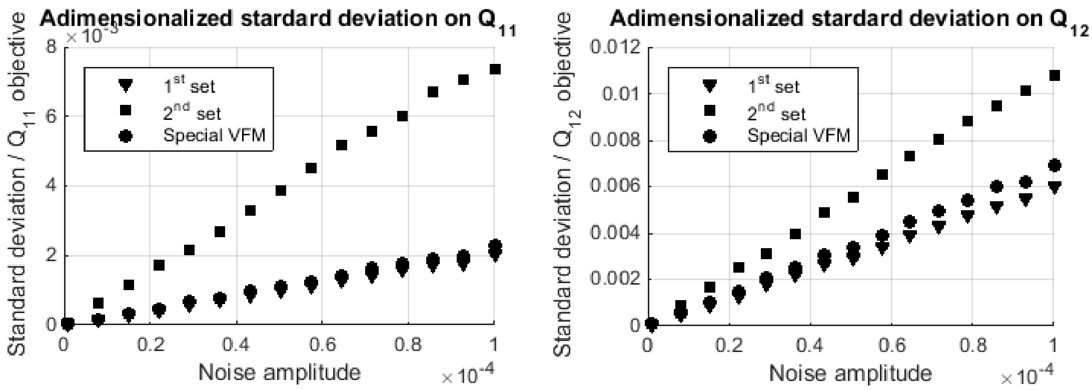


Figure 43. Results obtained in the non-homogeneous 1 specimen

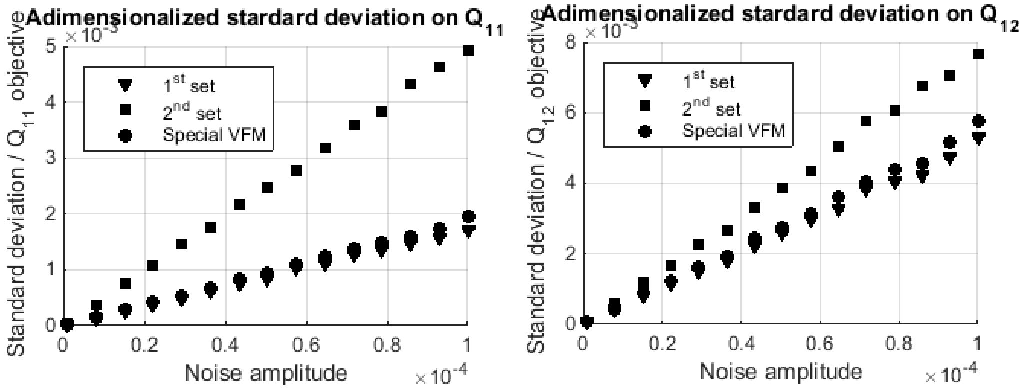


Figure 44. Results obtained on the non-homogeneous 2 specimen

As can be seen, using these specimens and these virtual fields, the set 1 and the special VFM seem to be really close, and must be said that field 1 may appear to be less noise dependent. It will be checked latter if this is also fulfilled if the optimized virtual fields are considered.

Anyway, it can be seen that the adimensionalized standard deviation is about three times bigger on Q_{12} than the one on Q_{11}, which means that when both parameters are combined for getting E and ν some deviations will be obtained, especially on ν.

5.2.4. *Optimized Virtual Fields on a traction test*

What is said at section 4.2.10 will be considered now. The code used for applying this concept is attached to the VFM book. At first, this code was developed for being applied on an Iosipescu test, which can be seen on the following picture:

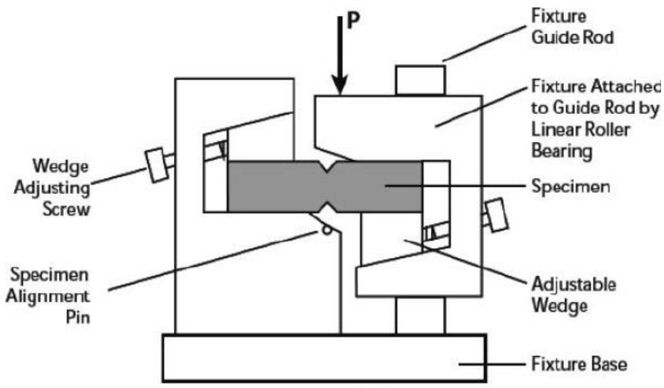


Figure 45. Iosipescu test scheme

This test is widely used for the characterization of the shear modulus. At his project, the boundary conditions have been modified for a traction test, as well as other characteristics such as the fact that originally it could not be used in non-constant area elements. The final code is shown and explained at the annex 6**.**

The main problem with that optimization codes is that they require orthotropic materials, as well as non-homogeneous strain fields. Following the equation (22), the required stiffness components can be obtained, always taking into account to choose the ones from the less noise-dependent virtual fields.

Now, the same questions as on the previous chapter are considered. Which degree shall be used, and is it less noise-dependent that the other alternatives?

<u>Degree dependence</u>

As is it said on the annex 5, the calculations will change depending on the polynomial degree, which will be an input in the optimized virtual field function. The calculations in order to compare the degree dependence for degrees 1-20 will be made by using m equal to n in all of them for the sake of simplicity. Results are presented down below.

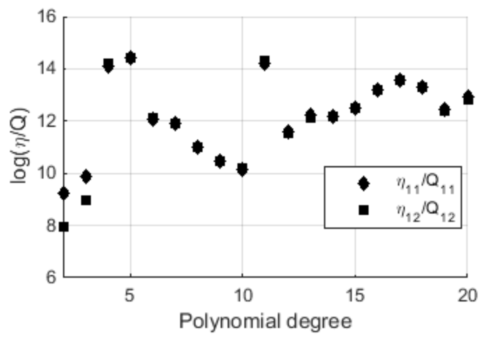


Figure 46. Degree dependence for the optimized virtual fields on the homogeneous specimen

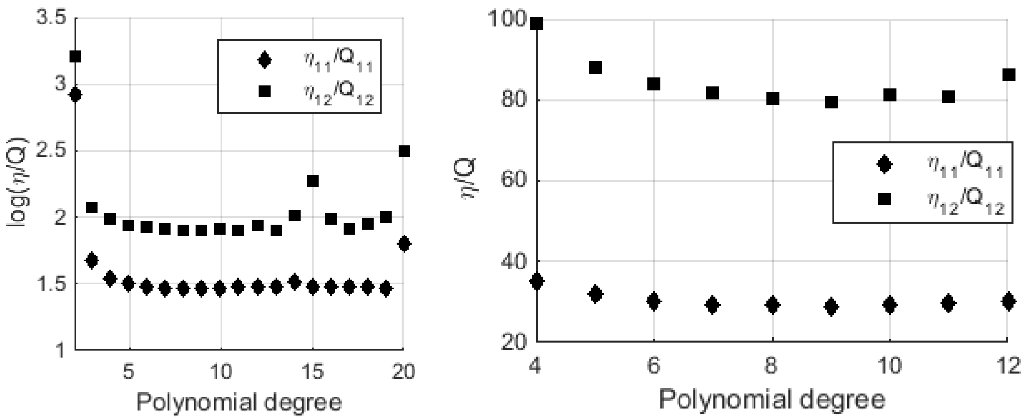


Figure 47. Degree dependence for the optimized virtual fields on the non-homogeneous 1 specimen

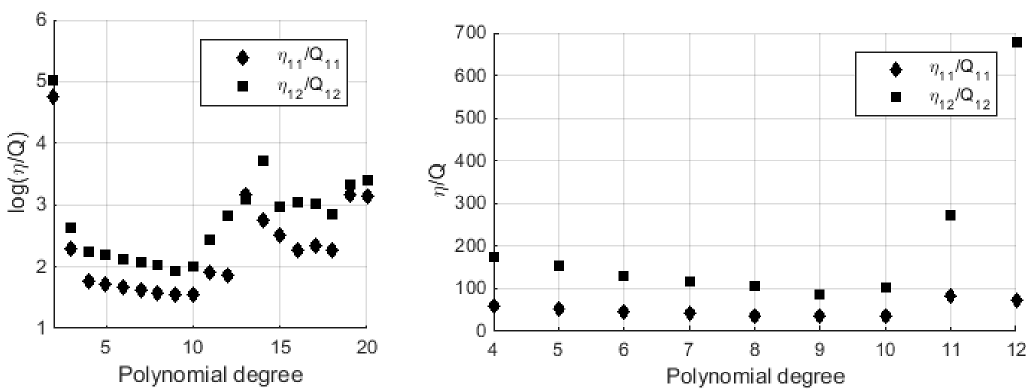


Figure 48. Degree dependence for the optimized virtual fields on the non-homogeneous 2 specimen

From these results, some conclusions can be extracted. The first one relies on the fact that this code does not supports a homogeneous strain fields, because the η parameters in this case are huge (it must be taken into account the fact that the ordinates axis is on logarithm values). This fact relies on the matrix called OptM, which will be close to zero in this case (mainly due to the influence of the Hessian matrix). This fact can be confirmed since the following message is obtained on the Matlab Command Window.

```
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 3.592036e-152.
> In OptimizedVirtualFieldsFunction (line 209)
  In Procesado (line 12)
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 3.592036e-152.
> In OptimizedVirtualFieldsFunction (line 210)
  In Procesado (line 12)
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 1.505913e-277.
> In OptimizedVirtualFieldsFunction (line 207)
  In Procesado (line 12)
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 1.505913e-277.
> In OptimizedVirtualFieldsFunction (line 208)
  In Procesado (line 12)
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 1.505913e-277.
> In OptimizedVirtualFieldsFunction (line 209)
  In Procesado (line 12)
```

Figure 49. Working precision issue in Matlab

Anyway, regarding the other two specimens it seem obvious that depending on the degree the noise influence will change, and the best degree will be nine in both cases, due to the fact that on higher degree numbers the method becomes unstable. This instability also relies in the computer working precision, since the matrix OptM is closer to singular as the polynomial degree increases. This degree will be taken for the calculations on the following section.

Comparison with the manually defined virtual fields

At this section it will be compared both the special and optimized virtual fields with the manually defined. The code used to generate the results is shown and explained on the annex 7. Standard deviations will be taken into account for comparing these fields.

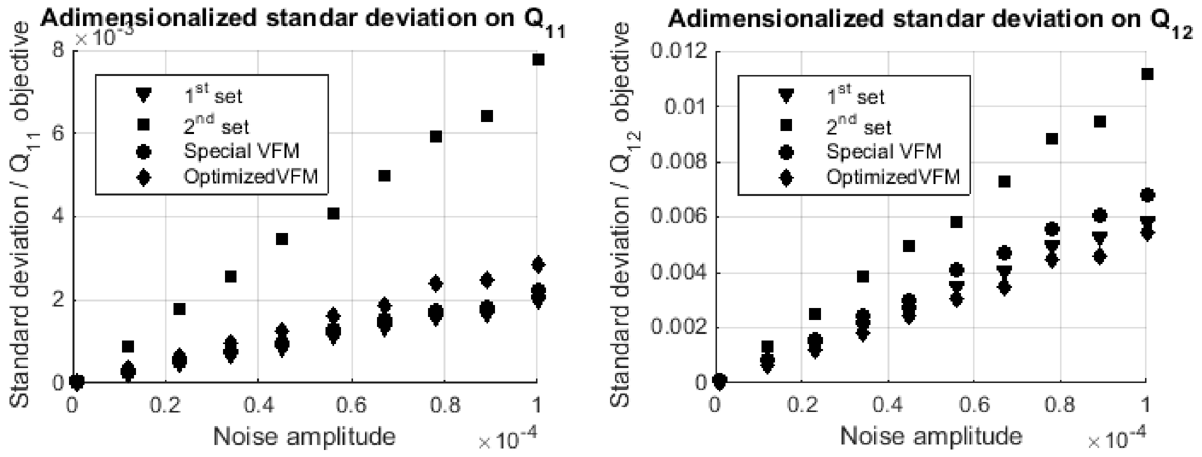


Figure 50. Noise influence on the non-homogeneous 1 traction specimen

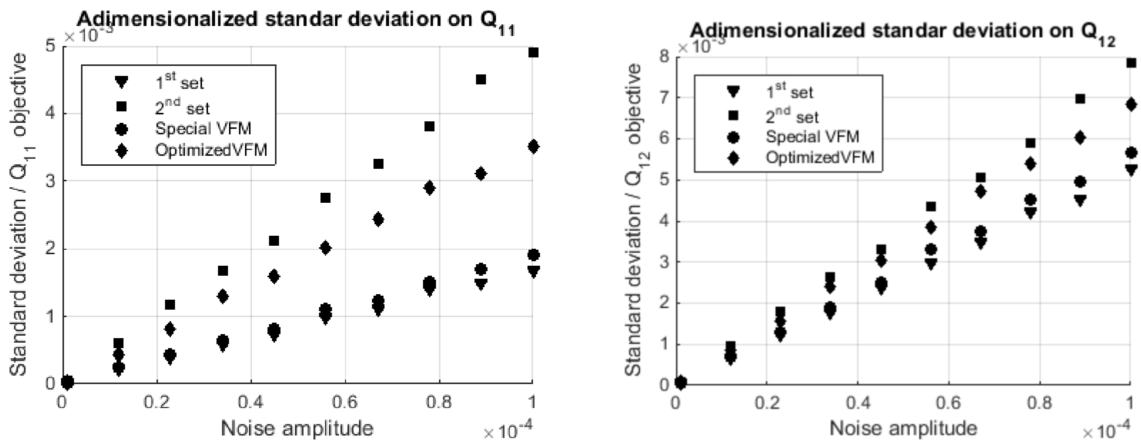


Figure 51. Noise influence on the non-homogeneous 2 traction specimen

It seems that the Optimized Virtual Fields is not the best option in this case, so maybe it can be choose the Special VF as the best option because it has almost the same noise dependency as the manually defined virtual fields, but providing some info about the noise dependency.

5.3. The VFM on a disc compression test

This whole section is going to study different ways of applying the VFM on a disc compression test, in order to find the best one. To that end, it will follow the same structure as on the previous section.

First, three specimens are going to be defined on Abaqus, which are shown below:

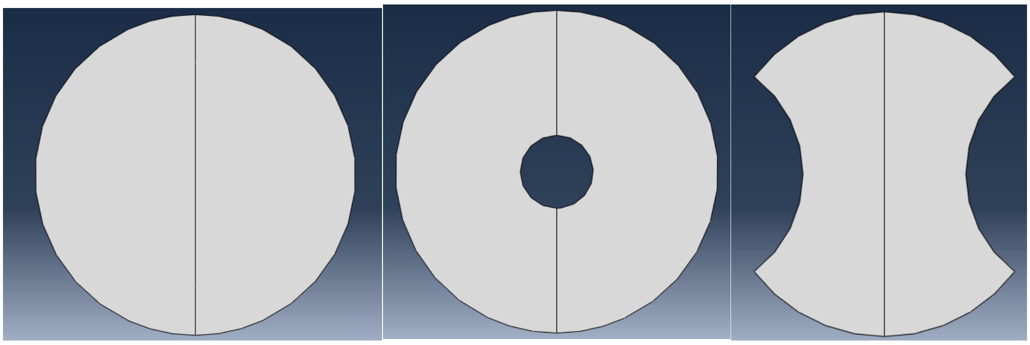


Figure 52. Specimens used in the disc compression test

The strain fields obtained on each specimen are the following ones:

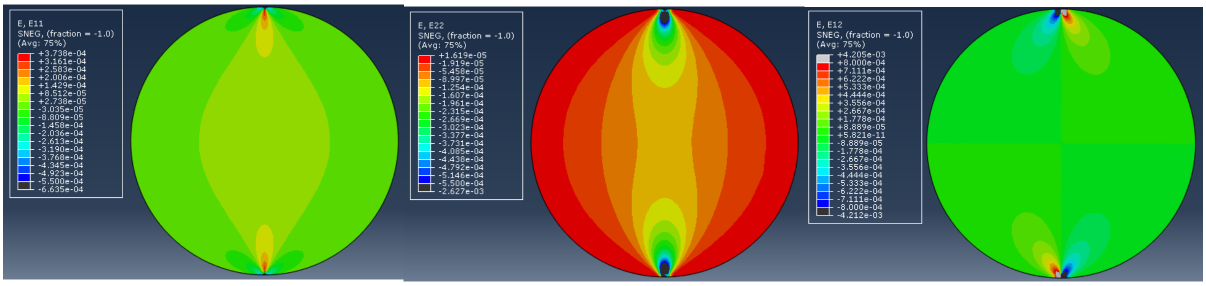


Figure 53. Strains obtained on the disc 1 specimen

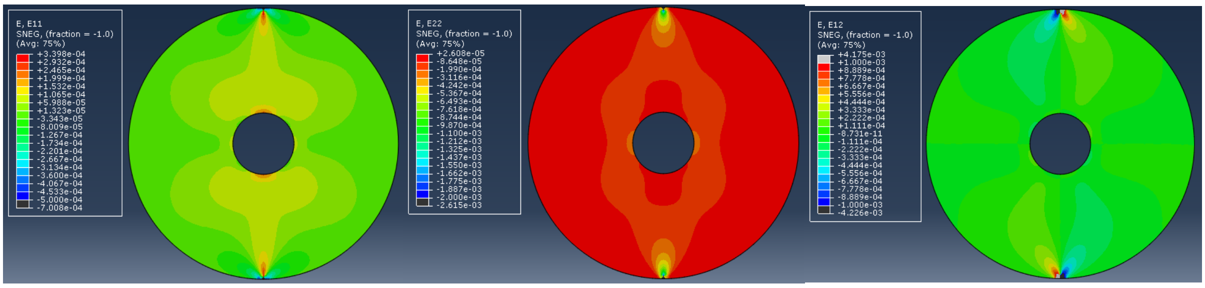


Figure 54. Strains obtained on the disc 2 specimen

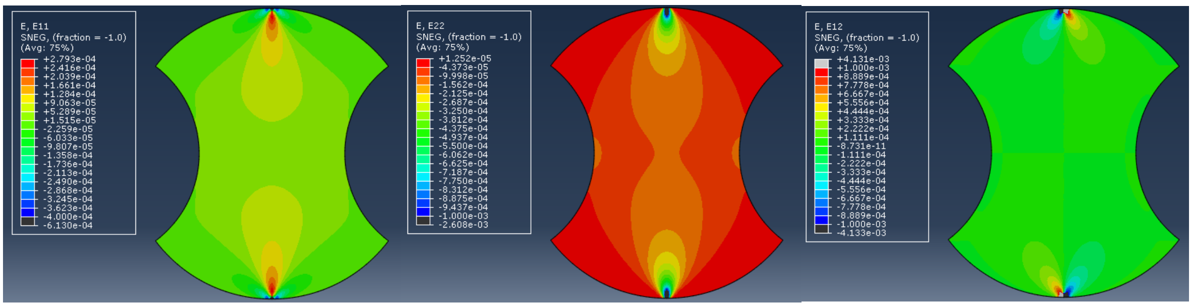


Figure 55. Strains obtained on the disc 3 specimen

Finally, instead of making an analysis on the element size as the one made on the previous section, this time it will just be computed the relative error on each specimen using the following manually defined virtual fields:

- $\boldsymbol{u}^{*(1)}$:

$$\begin{cases} u_1^{*(1)} = 0 \\ u_2^{*(1)} = x_2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(1)} = 0 \\ \varepsilon_2^{*(1)} = 1 \\ \varepsilon_6^{*(1)} = 0 \end{cases} \tag{87}$$

- $\boldsymbol{u}^{*(2)}$:

$$\begin{cases} u_1^{*(2)} = x_1 \\ u_2^{*(2)} = 0 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(2)} = 1 \\ \varepsilon_2^{*(2)} = 0 \\ \varepsilon_6^{*(2)} = 0 \end{cases} \tag{88}$$

$$\boldsymbol{A}: \begin{bmatrix} \int_S \varepsilon_2 \, dS & \int_S \varepsilon_1 \, dS \\ \int_S \varepsilon_1 \, dS & \int_S \varepsilon_2 \, dS \end{bmatrix} ; \quad \boldsymbol{B}: \begin{bmatrix} FD/t \\ 0 \end{bmatrix} \tag{89}$$

The reader could have notice that this is the same field as the manually defined 1 on the traction test. This time B vector has the same appearance as on the previous section since $\boldsymbol{u}^{*(1)}$ shows a constant displacement in $x_2 = D$. Following the path of using the same fields as on the previous section, it must be defined the axes zero on the same point, so it will be set it in the x_1 middle and x_2 bottom of the specimen.

The results for the relative errors obtained on each specimen are shown on the following table:

	Relative error on Q_{11}	Relative error on Q_{12}
Disc 1	2.426e-8	1.1682e-8
Disc 2	2.6145e-8	2.0451e-8
Disc 3	2.8230e-8	7.2924e-8

Table 7. Relative errors on the FEM disc under compression test

As it can be seen, the noise on the data concerning the three specimens is almost none. Taking this into account, the calculations can continue with no worry on any errors on our data base.

5.3.1. Manually defined Virtual Fields on a disc compression test

The same procedure as followed on the section 5.2.2 will be followed on the present section. In it, three sets of virtual fields are defined, and then a white noise is added on a progressive way to the data. The sets of virtual fields are the one presented on the previous section and the two following ones:

- Second set of virtual fields:
  - $\boldsymbol{u}^{*(3)}$:

$$\begin{cases} u_1^{*(3)} = 0 \\ u_2^{*(3)} = x_2^2 x_1^2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(3)} = 0 \\ \varepsilon_2^{*(3)} = 2x_1^2 x_2 \\ \varepsilon_6^{*(3)} = 2x_1 x_2^2 \end{cases} \tag{90}$$

  - $\boldsymbol{u}^{*(4)}$:

$$\begin{cases} u_1^{*(3)} = x_1^3 \\ u_2^{*(3)} = x_2^2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(4)} = 3x_1^2 \\ \varepsilon_2^{*(4)} = 2x_2 \\ \varepsilon_6^{*(4)} = 0 \end{cases} \tag{91}$$

$$\boldsymbol{A}^{(2)} = \begin{bmatrix} \int_S (2\varepsilon_2 x_1^2 x_2 + \varepsilon_6 x_1 x_2^2) dS & \int_S (2\varepsilon_1 x_1^2 x_2 - \varepsilon_6 x_1 x_2^2) dS \\ \int_S (3\varepsilon_1 x_1^2) dS & \int_S (3\varepsilon_2 x_1^2) dS \end{bmatrix} \tag{92}$$

$$\boldsymbol{B}^{(2)} = \begin{bmatrix} 0 \\ \frac{FD^2}{t} \end{bmatrix} \tag{93}$$

- Third set of virtual fields:
  - $\boldsymbol{u}^{*(5)}$:

$$\begin{cases} u_1^{*(5)} = 0 \\ u_2^{*(5)} = x_2 \cos\left(\frac{\pi x_1}{D}\right) \end{cases}; \quad \begin{cases} \varepsilon_1^{*(5)} = 0 \\ \varepsilon_2^{*(5)} = \cos\left(\frac{\pi x_1}{D}\right) \\ \varepsilon_6^{*(5)} = -x_2 \sin\left(\frac{\pi x_1}{D}\right)\frac{\pi}{D} \end{cases} \tag{94}$$

- $\boldsymbol{u}^{*(6)}$:

$$\begin{cases} u_1^{*(6)} = \sin\left(\frac{2\pi\left(x_1 + \frac{D}{2}\right)}{D}\right) \\ u_2^{*(6)} = 0 \end{cases}; \quad \begin{cases} \varepsilon_1^{*(6)} = \cos\left(\frac{2\pi\left(x_1 + \frac{D}{2}\right)}{D}\right)\frac{2\pi}{D} \\ \varepsilon_2^{*(6)} = 0 \\ \varepsilon_6^{*(6)} = 0 \end{cases} \tag{95}$$

$$\boldsymbol{A}^{(4)} = \begin{bmatrix} \int_S \left(\varepsilon_2 \cos\left(\frac{\pi x_1}{D}\right) - \frac{1}{2}\varepsilon_6 x_2 \sin\left(\frac{\pi x_1}{D}\right)\frac{\pi}{D}\right) dS & \int_S \left(\varepsilon_1 \cos\left(\frac{\pi x_1}{D}\right) + \frac{1}{2}\varepsilon_6 x_2 \sin\left(\frac{\pi x_1}{D}\right)\frac{\pi}{D}\right) dS \\ \int_S \left(\varepsilon_1 \cos\left(\frac{2\pi\left(x_1 + \frac{D}{2}\right)}{D}\right)\frac{2\pi}{D}\right) dS & \int_S \left(\varepsilon_2 \cos\left(\frac{2\pi\left(x_1 + \frac{D}{2}\right)}{D}\right)\frac{2\pi}{D}\right) dS \end{bmatrix} \tag{96}$$

$$\boldsymbol{B}^{(4)} = \begin{bmatrix} \frac{FD}{t} \\ 0 \end{bmatrix} \tag{97}$$

After adding a white noise using different noise amplitudes (1500 tests for each one), the following results are obtained:

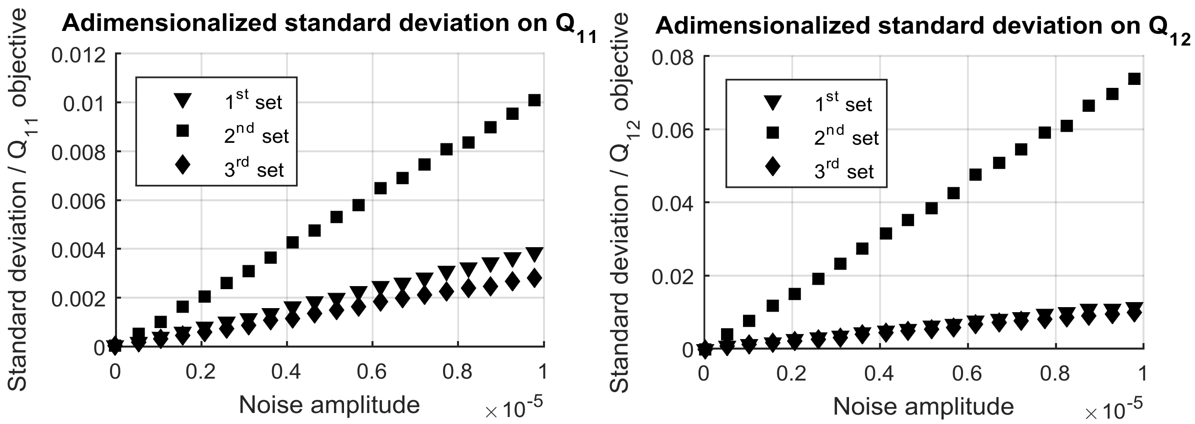


Figure 56. Noise influence on the manually defined virtual fields for the disc 1

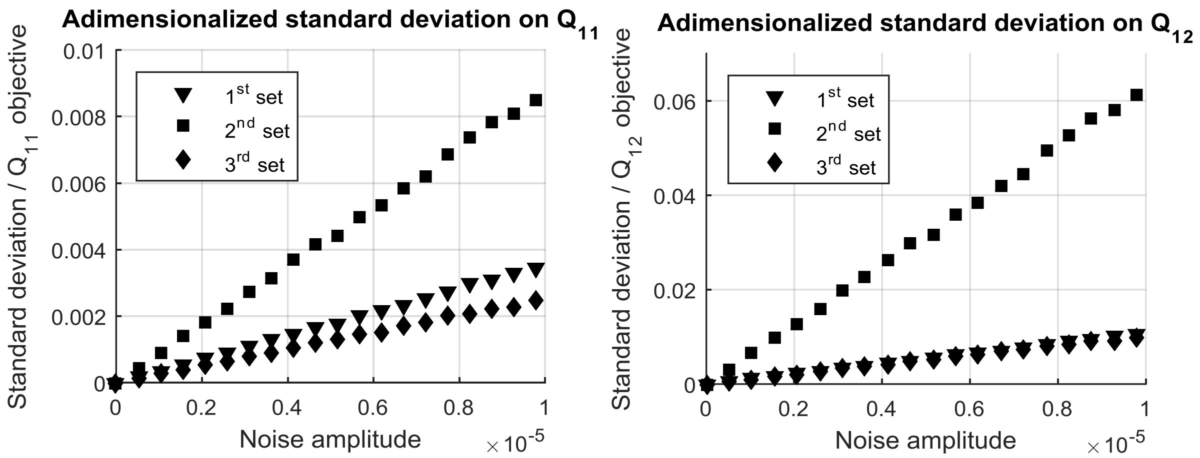


Figure 57. Noise influence on the manually defined virtual fields for the disc 2

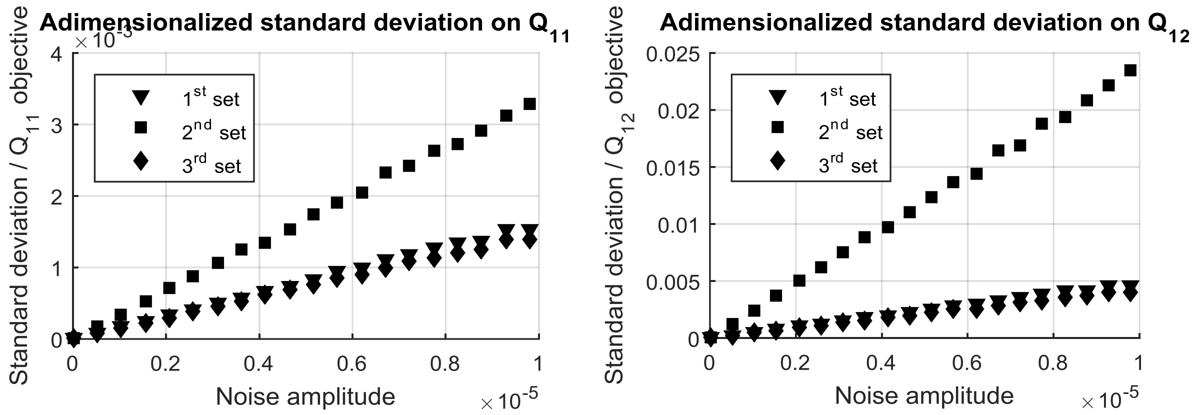


Figure 58. Noise influence on the manually defined virtual fields for the disc 3

As can be seen, this time the third set seems to be the less noise-dependent one. It may be due to the fact that it gives more importance to the middle zone, where there is a lot of information about the biggest strains on the specimens.

5.3.2. Special Virtual Fields on a disc compression test

In this section, the special virtual fields on a disc under compression will be computed. The boundary conditions that must be fulfilled this time are the following ones:

1. $u_2 = 0$ in $x_2 = 0$, for having a virtual work equal to zero on L_u.
2. $u_1 = 0$ in $x_2 = 0$, for avoiding the influence of the possible horizontal forces on the real tests.

3. $u_1 = 0$ in $x_2 = D$, for avoiding the influence of the possible horizontal forces on the real tests.

In this case, it is going to be purposed a polynomial virtual field that follows this expression:

$$u_1^* = \sum_{i=0}^{m}\sum_{j=0}^{n} A_{ij}\left(\frac{x_1}{D}\right)^i\left(\frac{x_2}{D}\right)^j$$
$$u_2^* = \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\left(\frac{x_1}{D}\right)^i\left(\frac{x_2}{D}\right)^j \qquad (98)$$

Therefore, the strain fields will be:

$$\begin{cases} \varepsilon_1^* = \sum_{i=1}^{m}\sum_{j=0}^{n} A_{ij}\frac{i}{D}\left(\frac{x_1}{D}\right)^{i-1}\left(\frac{x_2}{D}\right)^j \\ \varepsilon_2^* = \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\left(\frac{x_1}{D}\right)^i\frac{j}{D}\left(\frac{x_2}{D}\right)^{j-1} \\ \varepsilon_6^* = \sum_{i=0}^{m}\sum_{j=0}^{n} A_{ij}\frac{j}{D}\left(\frac{x_1}{D}\right)^i\left(\frac{x_2}{D}\right)^{j-1} + \sum_{i=0}^{p}\sum_{j=0}^{q} B_{ij}\left(\frac{x_1}{D}\right)^{i-1}\frac{i}{D}\left(\frac{x_2}{D}\right)^j \end{cases} \qquad (99)$$

As it can be seen, in order to fulfil the first and second expression it must be set $A_{00} = 0$ and $B_{00} = 0$. The third expression is more difficult to fulfil, as it must follow the following expression:

$$\sum_{j=1}^{n} A_{0j} = 0 \qquad (100)$$

The code that applies all these calculations as well as the special VFM is shown and explained on the annex 8.

Polynomial degree choice

First of all, the polynomial degree is compared:

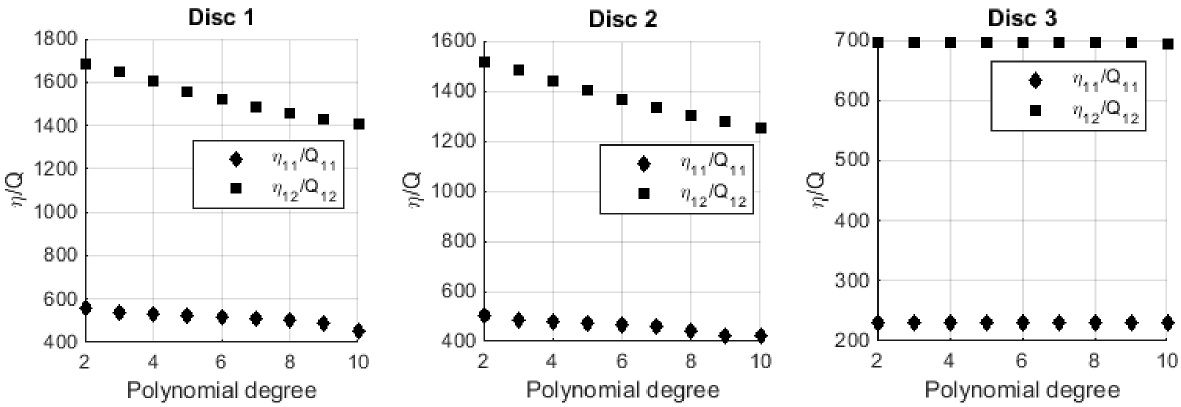


Figure 59. Noise dependence on the special virtual fields for a disc under compression

As it can be seen, the higher the degree, the less dependent the result is on noise. Nevertheless, the processing times spent in all the cases can be seen down below:

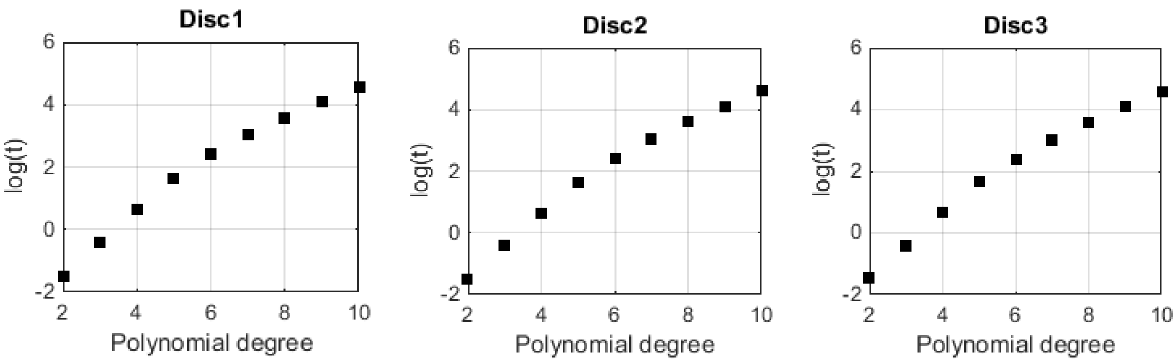


Figure 60. Computational time for the special virtual fields on a disc under compression

It may seem a good idea the fact of saving the terms that must be computed, but nothing further. When trying it, it is discovered the fact that a special virtual field (I mean, the $A_{ij}s$ coefficients which are not set to zero) which is valid for a geometry with a certain noise amplitude may not be valid if the noise amplitude gets bigger.

5.3.3. Optimized Virtual Fields on a disc compression test

Finally, this chapter will be finished by computing the optimized virtual fields for the disc under compression. In this case, the basis is the optimized virtual fields on the traction test, and the boundary conditions are changed in order to fulfil the conditions shown on the previous section. This code is shown and explained on the annex 9.

Finally, just like it has been seen previously, the best polynomial degree is chosen for this method and it is compared with the manually defined and the special virtual fields.

Degree dependence:

Computing the degree dependence just the way it was explained before, the following results are obtained:

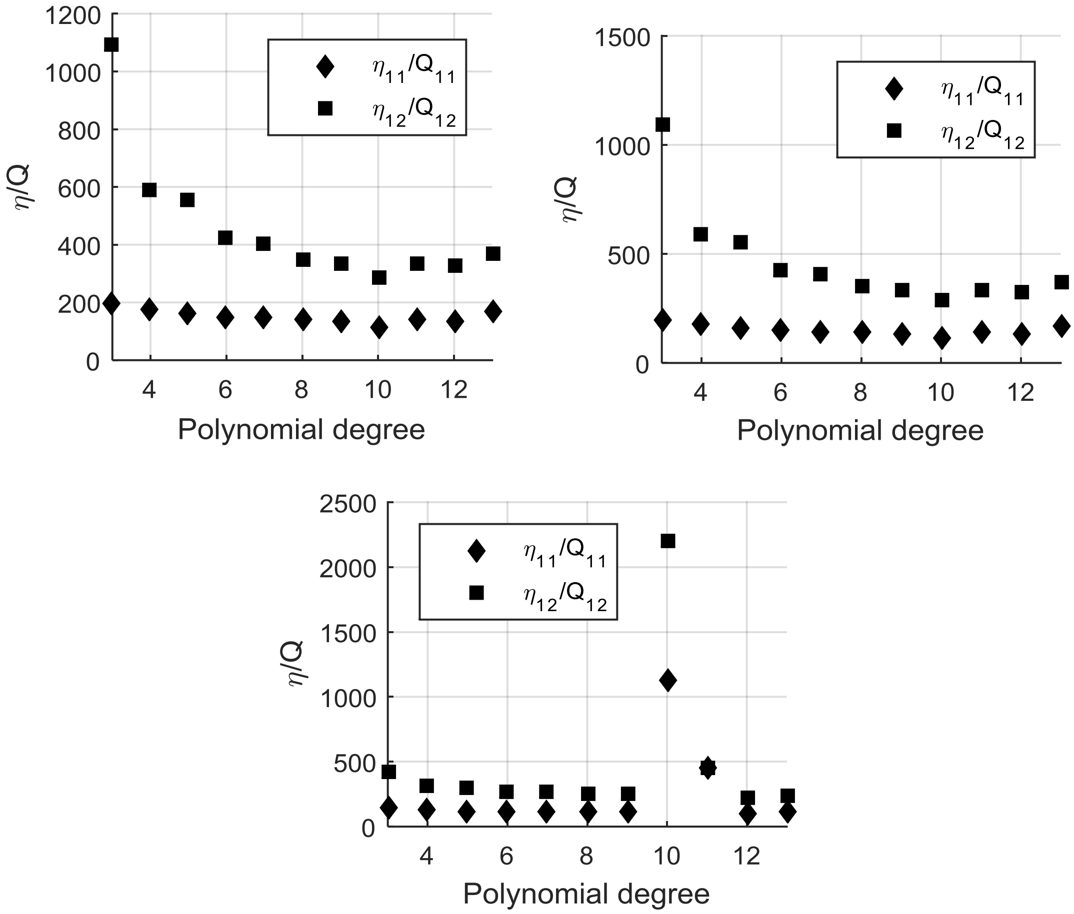


Figure 61. Degree dependence for the optimized virtual fields on disc 1, 2 and 3

In this case, a good choice will be a degree equal to 9, due to on a higher degree it is noticeable some instability, due to the machine working precision as it is explained on section 5.2.4. This degree is selected for computing the optimized virtual fields on the following section.

Comparison to the manually defined and optimized virtual fields

Finally, the Optimized VF will be compared to the Manually defined and the Special ones, in order to being able to choose the best method for computing this geometry.

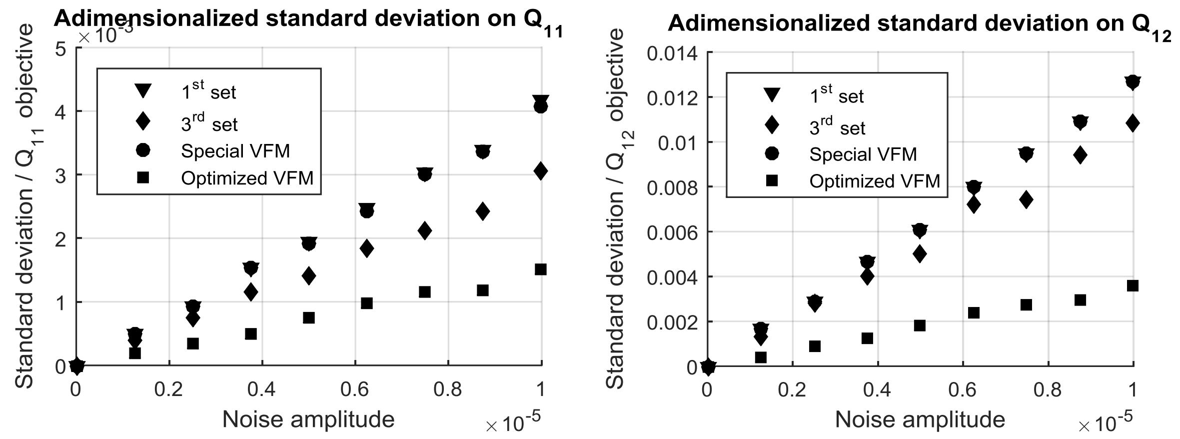


Figure 62. Noise influence on the VFM considering disc 1

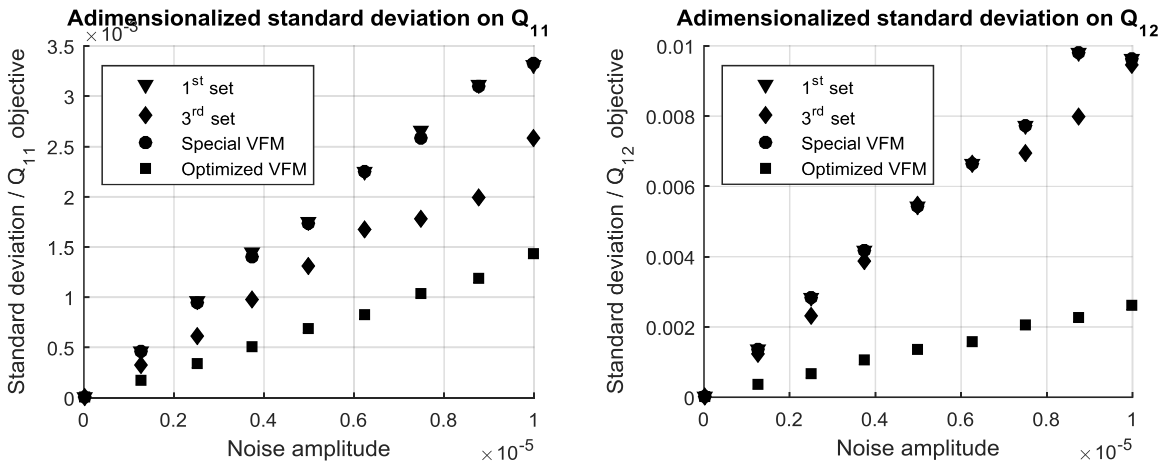


Figure 63. Noise influence on the VFM considering disc 2

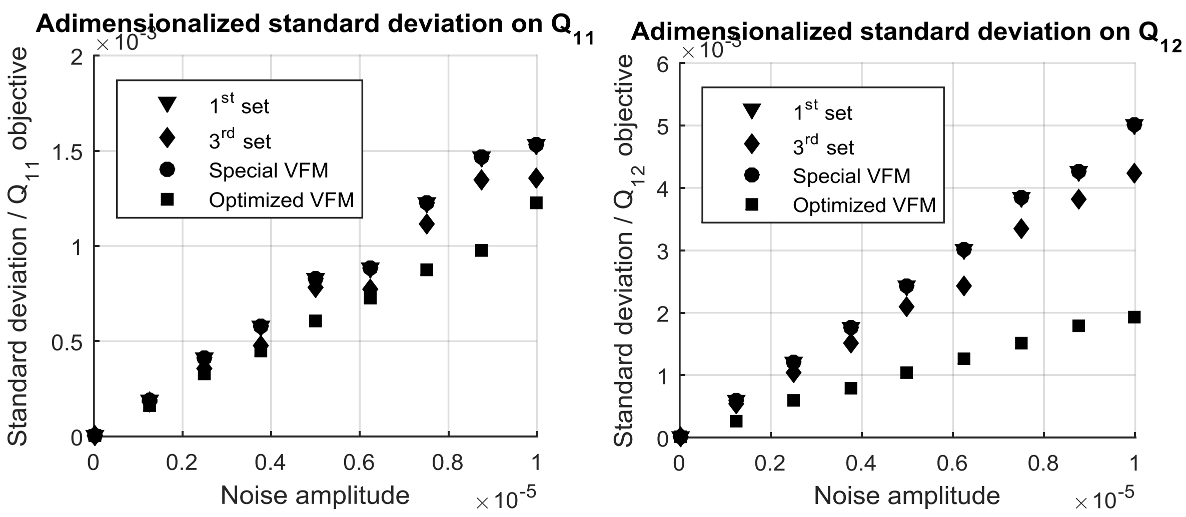


Figure 64. Noise influence on the VFM considering disc 3

In this case, it is quite clear that the Optimized VF are the best option for the VFM appliance, as it is noise influence is much lower than the influence on the other methods. As in this case this claim is true on the three different geometries, for the case of an isotropic material being performed a disc compression test, it just will be added the Optimized VF option on the GUI that will be developed later.

5.4. The VFM on orthotropic materials

For this section, the Virtual Fields Method will be computed on isotropic materials. To that end, an isotropic material will be defined on two different geometries, which are the same as on the specimen non-homogeneous 2 and the disc 2.

These are the strain fields obtained on each specimen:

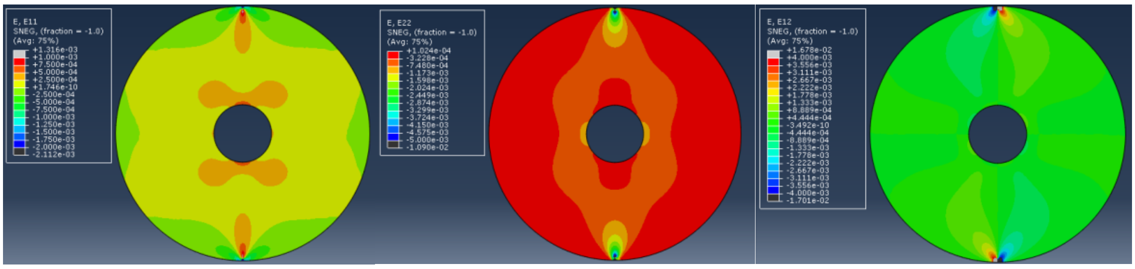


Figure 65. Stains obtained on the anisotropic disc

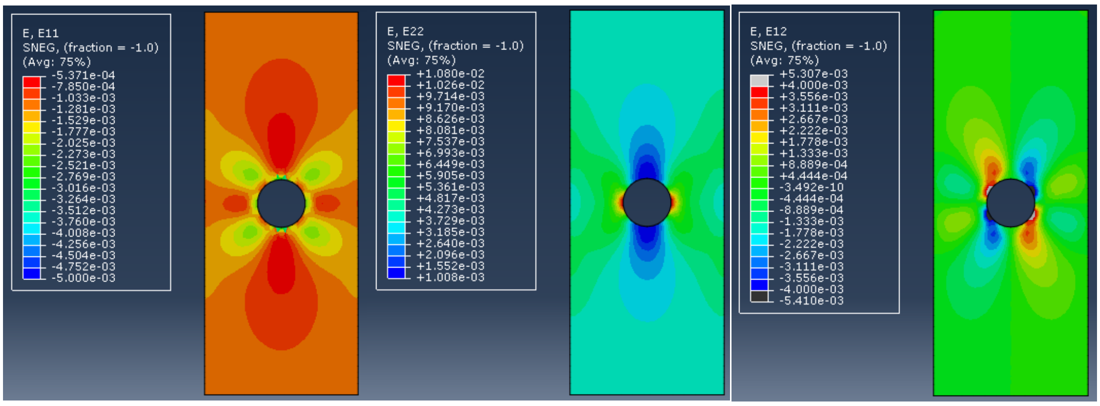


Figure 66. Strains obtained on the anisotropic traction test

5.4.1. *Manually defined Virtual Fields*

As a begin it will be applied the VFM through manually defined virtual fields. As an orthotropic material is considered, 4 virtual fields must be defined for each set (corresponding to the four stiffness components that will be computed.

- Disc under compression

The four virtual fields that are going to be used are the following ones:

- $\boldsymbol{u}^{*(\boldsymbol{1})}$:

$$\begin{cases} u_1^{*(1)} = 0 \\ u_2^{*(1)} = x_2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(1)} = 0 \\ \varepsilon_2^{*(1)} = 1 \\ \varepsilon_6^{*(1)} = 0 \end{cases} \tag{101}$$

- $\boldsymbol{u}^{*(\boldsymbol{2})}$:

$$\begin{cases} u_1^{*(2)} = x_1 \\ u_2^{*(2)} = 0 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(2)} = 1 \\ \varepsilon_2^{*(2)} = 0 \\ \varepsilon_6^{*(2)} = 0 \end{cases} \tag{102}$$

- $\boldsymbol{u}^{*(\boldsymbol{3})}$:

$$\begin{cases} u_1^{*(3)} = 0 \\ u_2^{*(3)} = x_2^2 x_1^2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(3)} = 0 \\ \varepsilon_2^{*(3)} = 2x_1^2 x_2 \\ \varepsilon_6^{*(3)} = 2x_1 x_2^2 \end{cases} \tag{103}$$

- $\boldsymbol{u}^{*(\boldsymbol{4})}$:

$$\begin{cases} u_1^{*(3)} = x_1^3 \\ u_2^{*(3)} = x_2^2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(4)} = 3x_1^2 \\ \varepsilon_2^{*(4)} = 2x_2 \\ \varepsilon_6^{*(4)} = 0 \end{cases} \tag{104}$$

$$\boldsymbol{A}: \begin{bmatrix} 0 & \int_S \varepsilon_2 dS & \int_S \varepsilon_1 dS & 0 \\ \int_S \varepsilon_1 dS & 0 & \int_S \varepsilon_2 dS & 0 \\ 0 & \int_S 2x_1^2 x_2 \varepsilon_2 dS & \int_S 2x_1^2 x_2 \varepsilon_1 dS & \int_S 2x_1 x_2^2 \varepsilon_6 dS \\ \int_S 3x_1^2 \varepsilon_1 dS & \int_S 2x_2 \varepsilon_2 dS & \int_S (2x_2\varepsilon_1 + 3x_1^2 \varepsilon_2) dS & 0 \end{bmatrix} \tag{105}$$

$$\boldsymbol{B}: \begin{bmatrix} FD/t \\ 0 \\ 0 \\ FD^2/t \end{bmatrix} \tag{106}$$

- Traction test

For the traction test the same virtual fields are used, with some variations:

- $\boldsymbol{u}^{*(1)}$:

$$\begin{cases} u_1^{*(1)} = 0 \\ u_2^{*(1)} = x_2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(1)} = 0 \\ \varepsilon_2^{*(1)} = 1 \\ \varepsilon_6^{*(1)} = 0 \end{cases} \tag{107}$$

- $\boldsymbol{u}^{*(2)}$:

$$\begin{cases} u_1^{*(2)} = x_1 \\ u_2^{*(2)} = 0 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(2)} = 1 \\ \varepsilon_2^{*(2)} = 0 \\ \varepsilon_6^{*(2)} = 0 \end{cases} \tag{108}$$

- $\boldsymbol{u}^{*(3)}$:

$$\begin{cases} u_1^{*(3)} = 0 \\ u_2^{*(3)} = x_2^2 x_1^2 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(3)} = 0 \\ \varepsilon_2^{*(3)} = 2x_1^2 x_2 \\ \varepsilon_6^{*(3)} = 2x_1 x_2^2 \end{cases} \tag{109}$$

- $\boldsymbol{u}^{*(4)}$:

$$\begin{cases} u_1^{*(3)} = x_1^3 \\ u_2^{*(3)} = 0 \end{cases} ; \quad \begin{cases} \varepsilon_1^{*(4)} = 3x_1^2 \\ \varepsilon_2^{*(4)} = 0 \\ \varepsilon_6^{*(4)} = 0 \end{cases} \tag{110}$$

$$\boldsymbol{A}: \begin{bmatrix} 0 & \int_S \varepsilon_2 dS & \int_S \varepsilon_1 dS & 0 \\ \int_S \varepsilon_1 dS & 0 & \int_S \varepsilon_2 dS & 0 \\ 0 & \int_S 2x_1^2 x_2 \varepsilon_2 dS & \int_S 2x_1^2 x_2 \varepsilon_1 dS & \int_S 2x_1 x_2^2 \varepsilon_6 dS \\ \int_S 3x_1^2 \varepsilon_1 dS & 0 & \int_S 3x_1^2 \varepsilon_2 dS & 0 \end{bmatrix} \tag{111}$$

$$\boldsymbol{B}: \begin{bmatrix} FL/_t \\ 0 \\ \dfrac{FL^2w^2}{12t} \\ 0 \end{bmatrix} \quad (112)$$

In both cases, the fields are derived of sets 1 and 2 of the manually defined on each respective chapter. Results are presented down below:

$\boldsymbol{Q_{11}}$	$\boldsymbol{Q_{22}}$	$\boldsymbol{Q_{12}}$	$\boldsymbol{Q_{66}}$
79.03 GPa	56.45 GPa	22.58 GPa	20 GPa

Table 8. Orthotropic material parameters

	Relative error on Q_{11} (%)	Relative error on Q_{12} (%)	Relative error on Q_{22} (%)	Relative error on Q_{66} (%)
Disc under compression	3.10	0.35	3.10	1.14
Traction test	0.45	0.05	0.45	0.34

Table 9. Relative errors on the orthotropic material using Manually Defined Virtual Fields

On both cases, it can be seen that errors are much bigger than the ones obtained by using isotropic formulation. This may be due to the fact that in this case four virtual fields must be defined. Therefore, each stiffness component depends on four different fields, adding the noise-dependence on each one. It is more; even though I defined some virtual fields on a different way (and they were well defined by comparing the B defined and obtained by multiplying A matrix by a vector containing the objective Q vector), it has been necessary to make some changes in order to reduce the noise dependence as the relative errors obtained were about 60%.

5.4.2. Special Virtual Fields

In this section, the special virtual fields are going to be computed on both cases. To that end, it just have to be added the two integrals corresponding to the two additional stiffness components. The reader who has read and understood the isotropic sections will need no further information, as at this point it is a trivial matter.

As this time when two more integrals have to be computed the computational cost of these special virtual fields increase as the combination possibilities increase. Because of that, this time the calculation will be made between degree 1 and 3.

Degree dependence

In this case the results will be presented in a table instead of on a graphic due to there are few points for showing them on a graph. The results obtained are the following ones:

	Degree	η_{11}/Q_{11}	η_{22}/Q_{22}	η_{12}/Q_{12}	η_{66}/Q_{66}
Traction	2	1311.3	154.55	1326	852.41
	3	849.7	101.17	860.3	513.63
Disc compression	2	652.2	100.26	661.5	477.76
	3	1345.5	183.10	1267.7	$3.79 \cdot 10^6$

Table 10. Degree dependence on the orthotropic Special Virtual Fields

In the traction test, the higher the degree is, the less noise-dependent the result is. On the other hand, this rule is not fulfilled on the compression disc. The main reason to those results is the fact that the computer working precision is not enough for performing these tests, which makes a direct influence on the results.

5.4.3. *Optimized Virtual Fields*

In the previous sections the special virtual fields have been applied by computing the optimized virtual fields as if it was an orthotropic material, and the combining the stiffness components for getting the ones used on an isotropic material. Therefore, in this section it must be deleted the part of the code which makes this combination for obtaining the four desired stiffness components.

Degree dependence

In this section, the η values will be computed for these optimized virtual fields. To that end, the calculations will be made between degree 3 and 15 since on a higher or lower value the results obtained will be unstable.

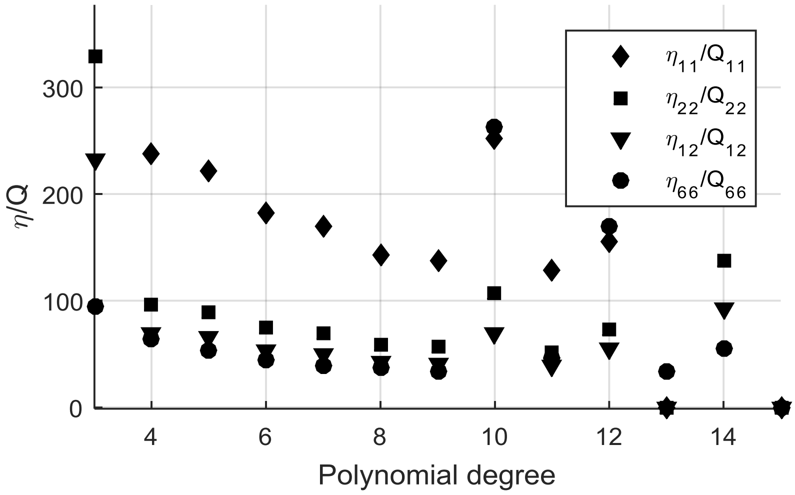


Figure 67. Degree dependence on an Optimized Virtual field for a traction orthotropic test

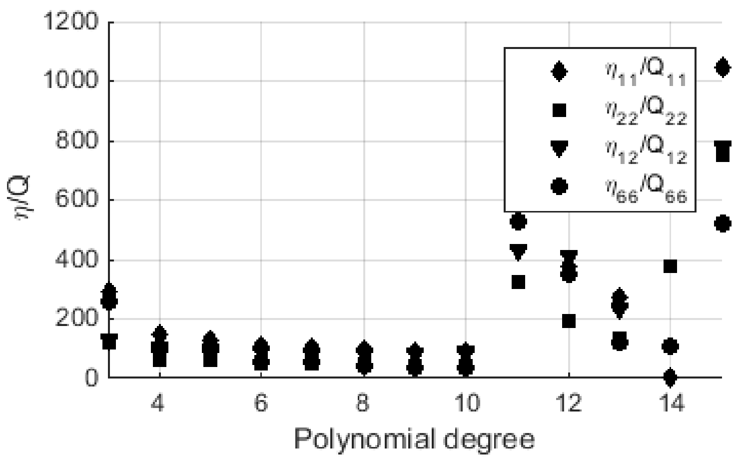


Figure 68. Degree dependence on an Optimized Virtual field for a disc compression orthotropic test

It seems that 9 is a stable degree with low noise dependence. Therefore, it will be used to make the comparison between the three alternatives of virtual fields.

Comparison with the manually defined and the special virtual fields

Finally, all the alternatives for applying the VFM on an orthotropic material specimen will be compared. As said before, the manually defined virtual fields, the special virtual fields on a degree 2 (so that all methods have the same computational cost) and the optimized virtual fields on a degree 9 will be computed.

The results can be seen on the graphics down below:

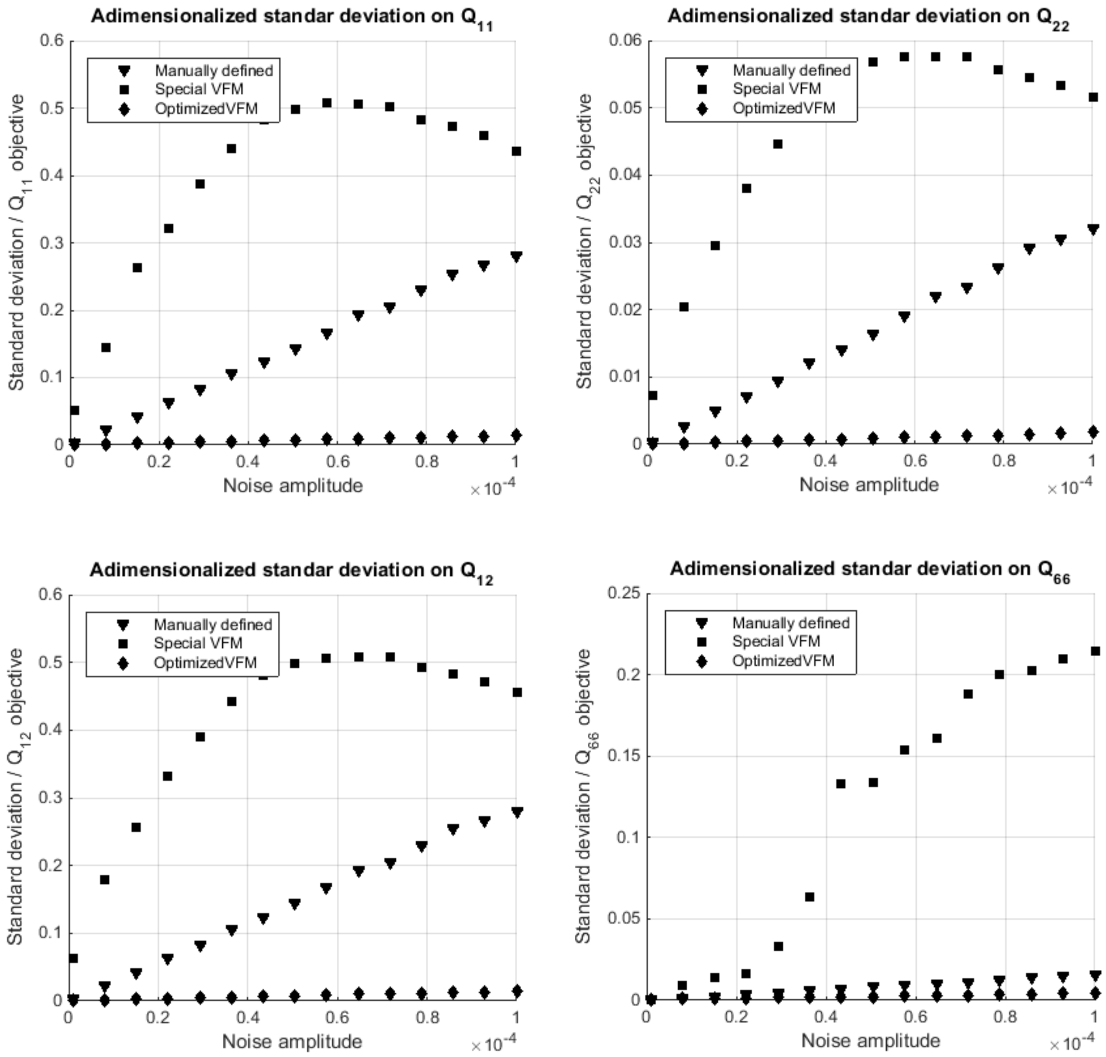
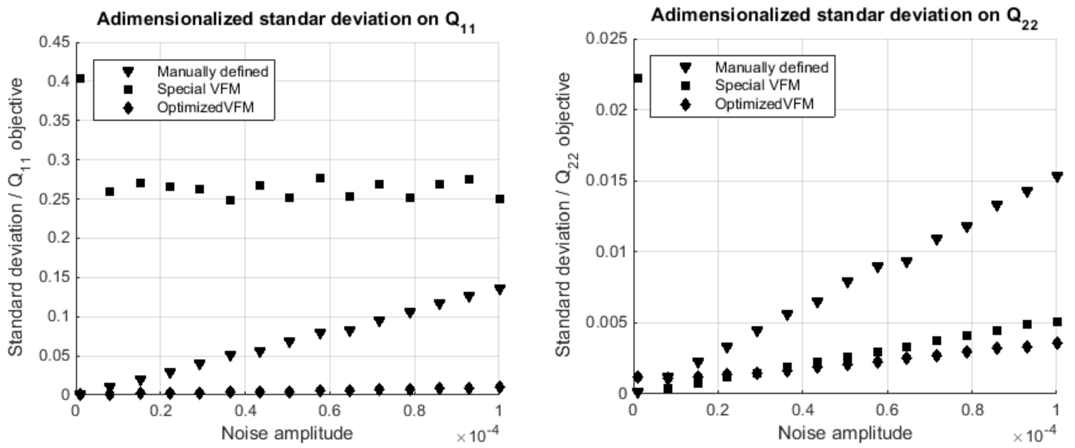


Figure 69. Noise dependence on a traction test, orthotropic material



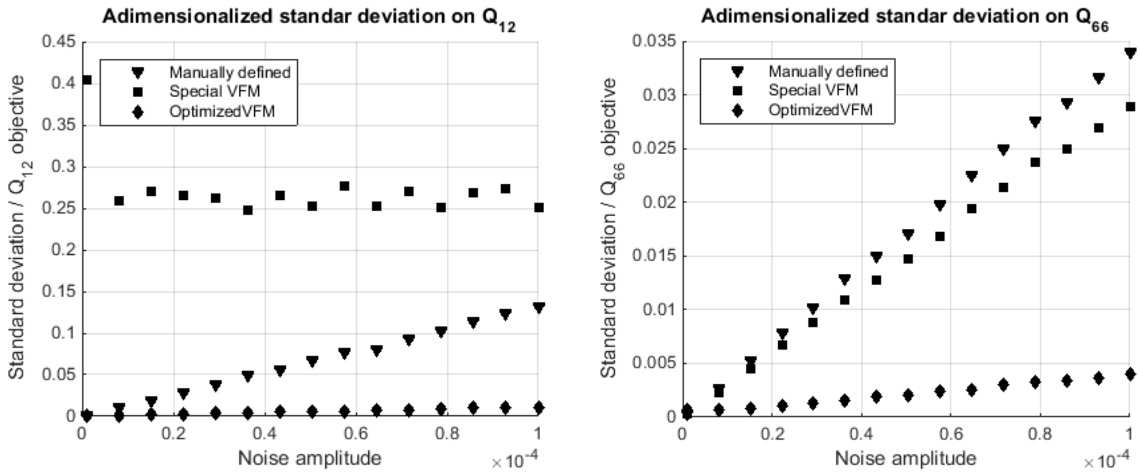


Figure 70. Noise dependence on a disc compression, orthotropic material

As a conclusion, it can be seen that the special virtual fields do not seem to make a good performance on these tests. This fits in with the fact that adjusting the code was a hard job, which makes no sense since the code should work with no problem if the boundary conditions and the integrals are defined properly.

Anyway, the best option in all the cases seems to be the optimized virtual fields, so they will be the ones that will be used hereinafter for computing the orthotropic specimens.

Finally, it is going to be plotted the adimensionalized standard deviations obtained on all the parameter by using the optimized virtual fields:

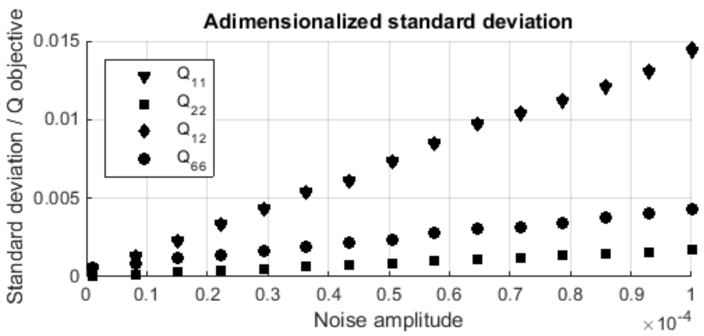


Figure 71. Adimensionalized standard deviations on the traction test

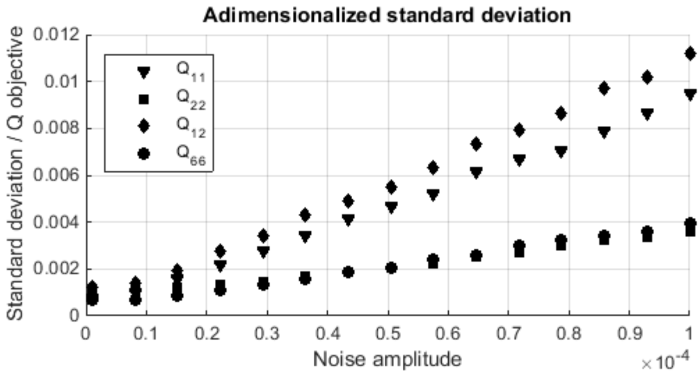


Figure 72. Adimensionalized standard deviations on the disc under compression

On both specimens are critical the Q_{11} and Q_{12} parameters, due to the Adimensionalized standard deviation is bigger on them than on the other two parameters.

5.5. The VFM on anisotropic materials

At the case of an anisotropic material, the VFM can be solved by solving a six equations linear equation system. Nevertheless, it must be taken into account the fact that the most used anisotropic materials are the fibre composite materials, which are usually used (and very often in its most consolidated application, the aerospace industry) in the shape of laminates. Within the Love-Kirchoff thin plate theory, the mechanical behaviour of such laminates can be calculated from the orthotropic stiffness of each individual plies. Therefore, the identification of the in-plane behaviour of such orthotropic plies is of great importance.

This assumption is taken from the book “The Virtual Fields Method” [1]. In its case, the author relies all his calculations on this assumption. Taking into account the success that Pierron, F. obtains on his results, it seems clear to me that it is possible take this assumption.

Therefore, as it is just needed to compute the VFM in case of an orthotropic material, the same calculations as on the previous section will have to be made, so it seems senseless the fact of repeating them again.

5.6. Matlab GUI for applying the VFM

In this section, a Matlab GUI is going to be developed for making the VFM appliance more user-friendly. The reader can take a first look to that GUI on the following figure:

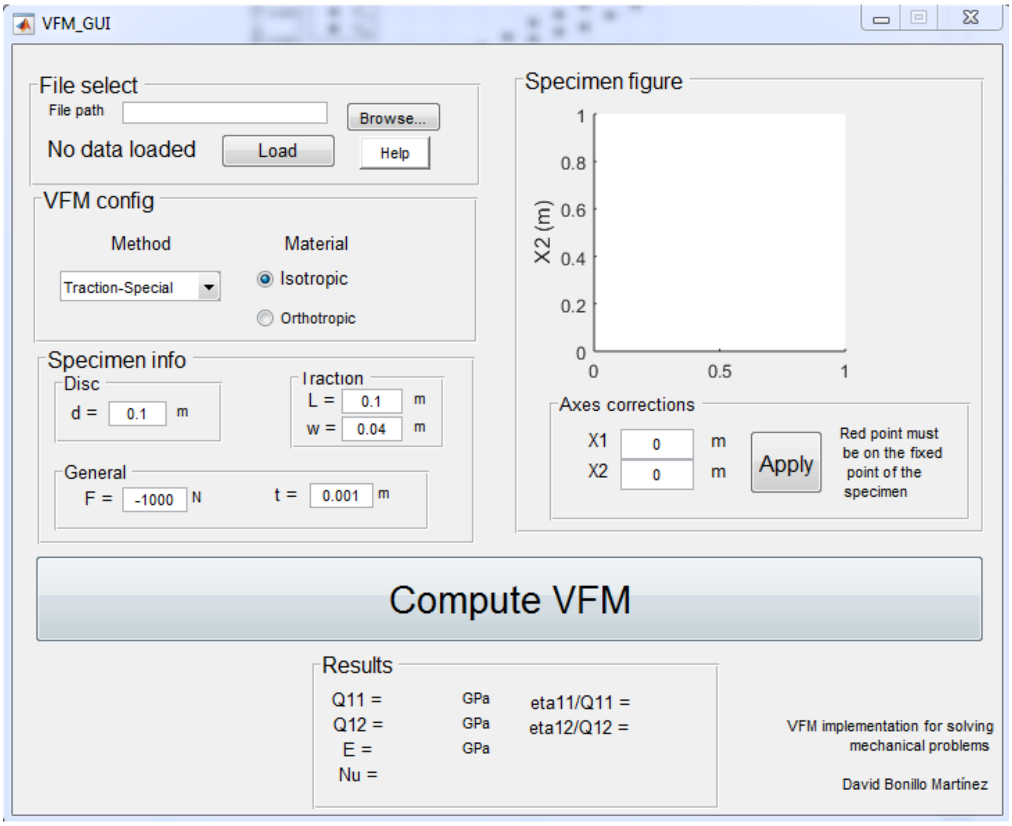


Figure 73. First look to the VFM GUI

Now, the main using paths for this GUI will be explained. First of all, the data to process must be loaded. To that end, it is needed a file including a vector containing the strains in the axis 1 (Eps1), 2 (Eps2) and 12 (Eps6). They will contain the info about the strains on each surface element. In addition, the file must contain the vectors X1 (containing the x coordinate of the surface element), X2 (containing the y coordinate of the surface element) and dS (containing the surface value of the element). In case of using DIC data, the vector dS will contain as much elements as points are on the specimen, and all the elements will be equal to the total surface divided by the number of points.

The file containing this info must be selected in the load section of the GUI. After selecting the file and clicking the load button, the window must look like this:

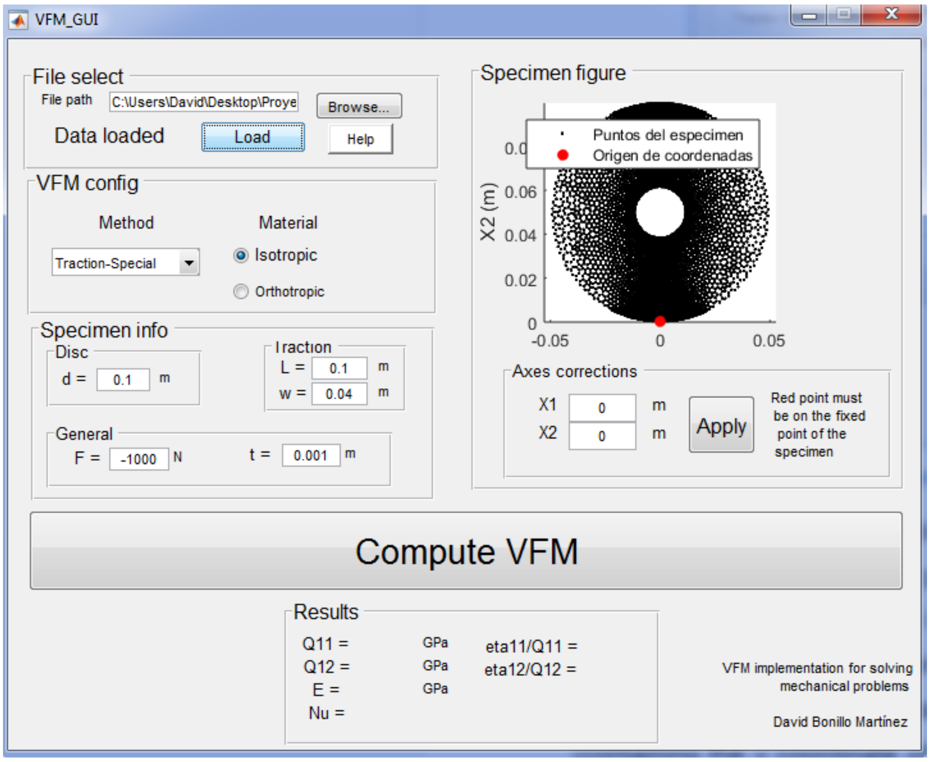


Figure 74. Data loaded in the GUI

The next step is checking that the coordinate origin is well defined, since an error in it will result into a wrong result for the stiffness coefficients. To that end, the user must check the fact that the red point on the graphics must be on the fixed point of the specimen in case of a compression disc, and must be on any point of the fixed line in a traction test.

If the axes are not well aligned the user must use the axes corrections section placed underneath the graphics. An example will be performed even though in this case the axis origins is well placed. The axis origin will be displaced 5 cm to the right in the x_1 direction.

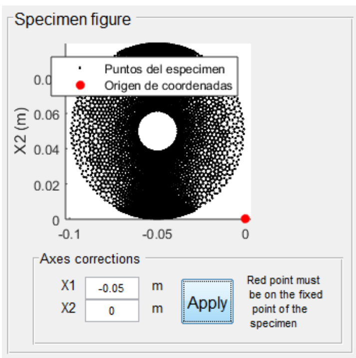


Figure 75. Axis origin displacement

Once defined the axis zero, the user must select the method he wants to use in order to compute the VFM. To that end, he must make his choice on the VFM Config window. In it, some options can be chosen. For a traction specimen it can be used either special virtual fields (just on isotropic specimens since it has been proved previously its bad work on orthotropic specimens) or optimized virtual fields (which can be used both in orthotropic or isotropic specimens, but in some cases they seem to not being the best option). For a compression disc test, it just is possible to choose the optimized virtual fields (either in isotropic or orthotropic specimens) since they have proved to be the best option in this type of set.

Therefore, in the example the user must choose the following options:

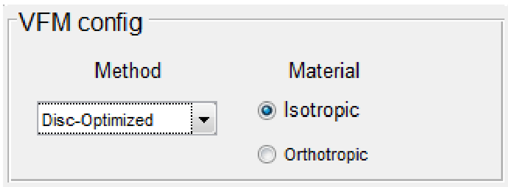


Figure 76. VFM configuration in the example

Finally, the specimen info must be introduced for the calculations. If a disc compression test is being computed the interface will neglect the information contained in the traction section, and vice versa. In this case, the default data is correct for this specimen, so no further changes are necessary.

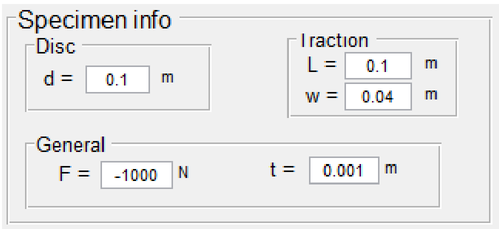


Figure 77. Specimen info in the example

Finally, after having introduced all this info the user must click on the Calculate button, which will apply the VFM following the path that has been previously defined. After that, the results will be presented on the results window:

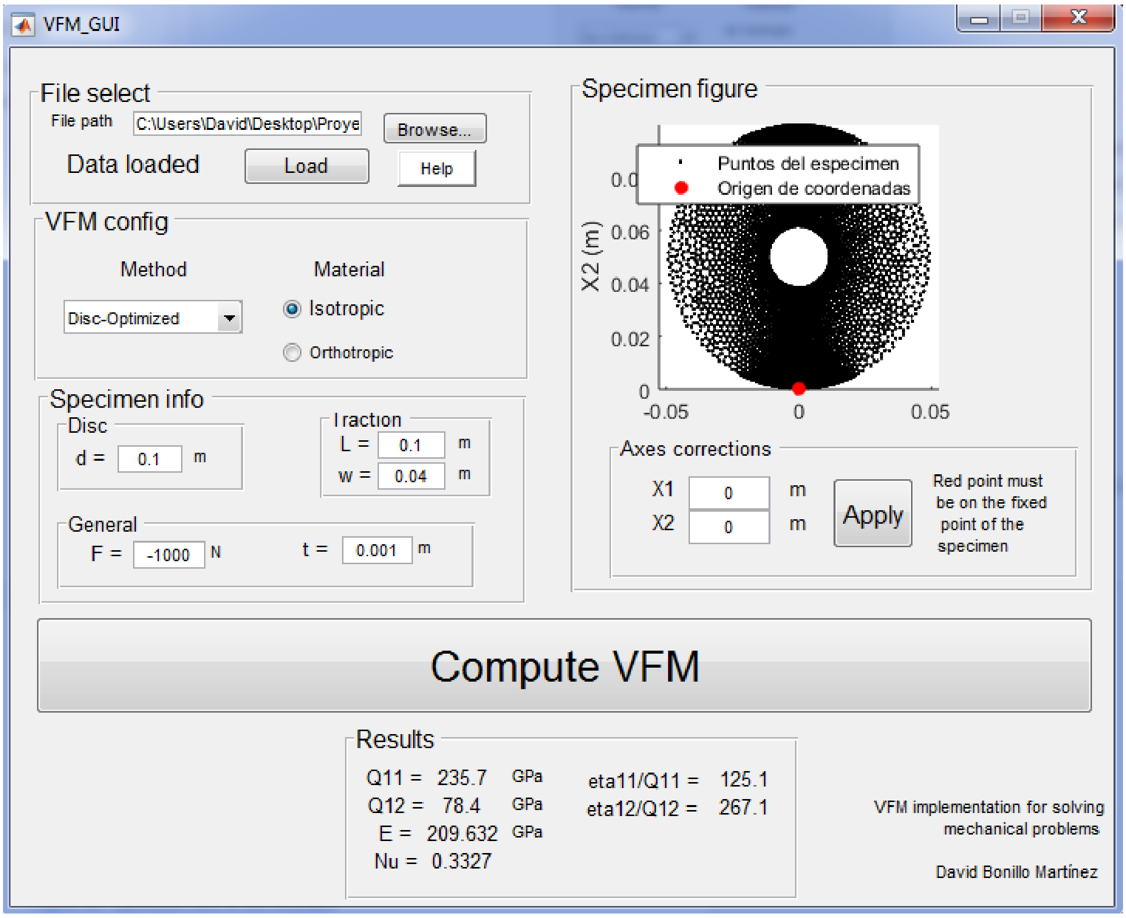


Figure 78. GUI appearance after the calculations

The user must be careful about the axis zero, the method used (since the optimized virtual fields may be not accurate in some configurations due to the matrix is almost singular), the specimen orientation (the main force must always be applied on the y-axis) and the specimen info (most of all the sense of the force).

For finishing this section, it is presented an example to compute the VFM through this GUI in a traction specimen, in order to show the reader the way it should be done.

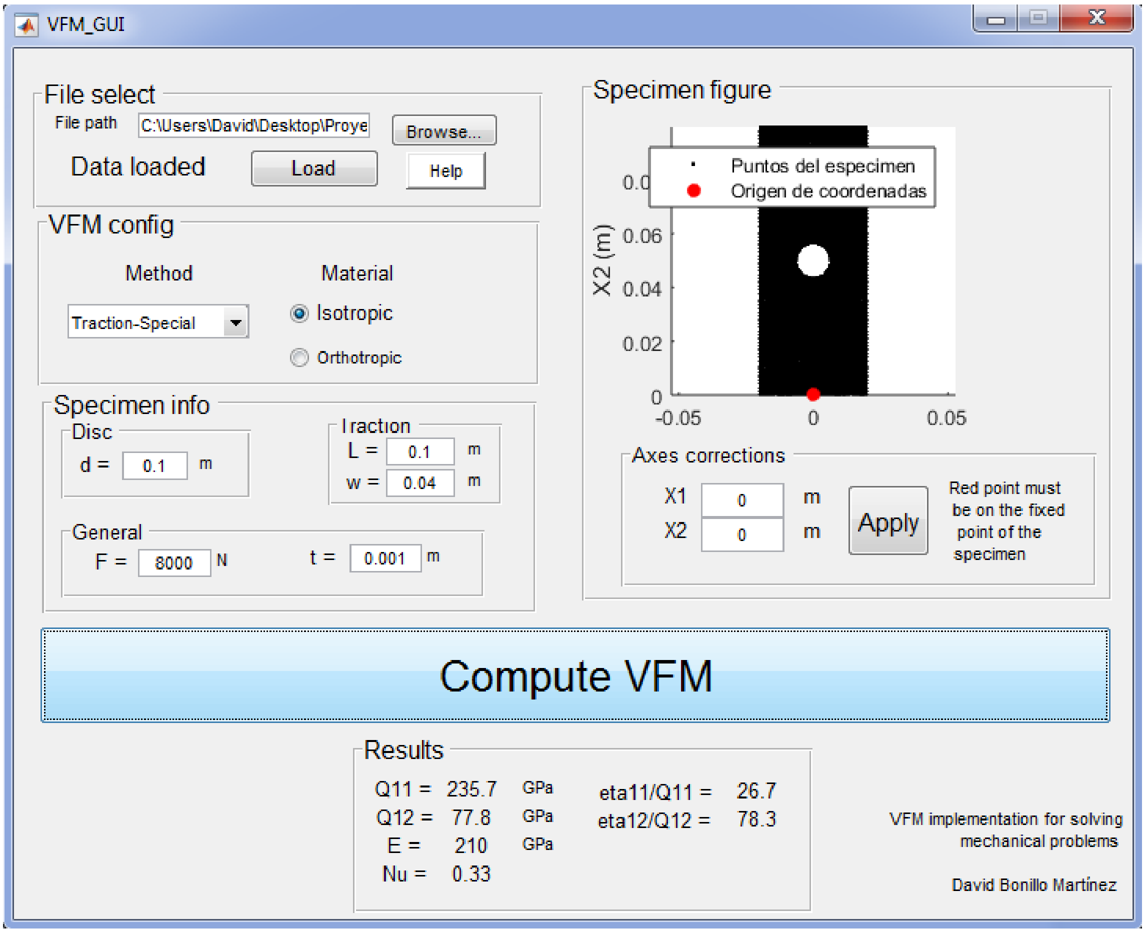


Figure 79. GUI configuration for a traction test

6. The VFM on DIC tests

6.1. VFM on simulated DIC tests

In this section it is going to be applied the VFM on a specimen which has been submitted to a uniaxial tension. To that end two Matlab-based software will be needed. The first one is the DIC simulator developed by “The Franck Lab”. It can be found on the website [10]. The main window can be seen down below:

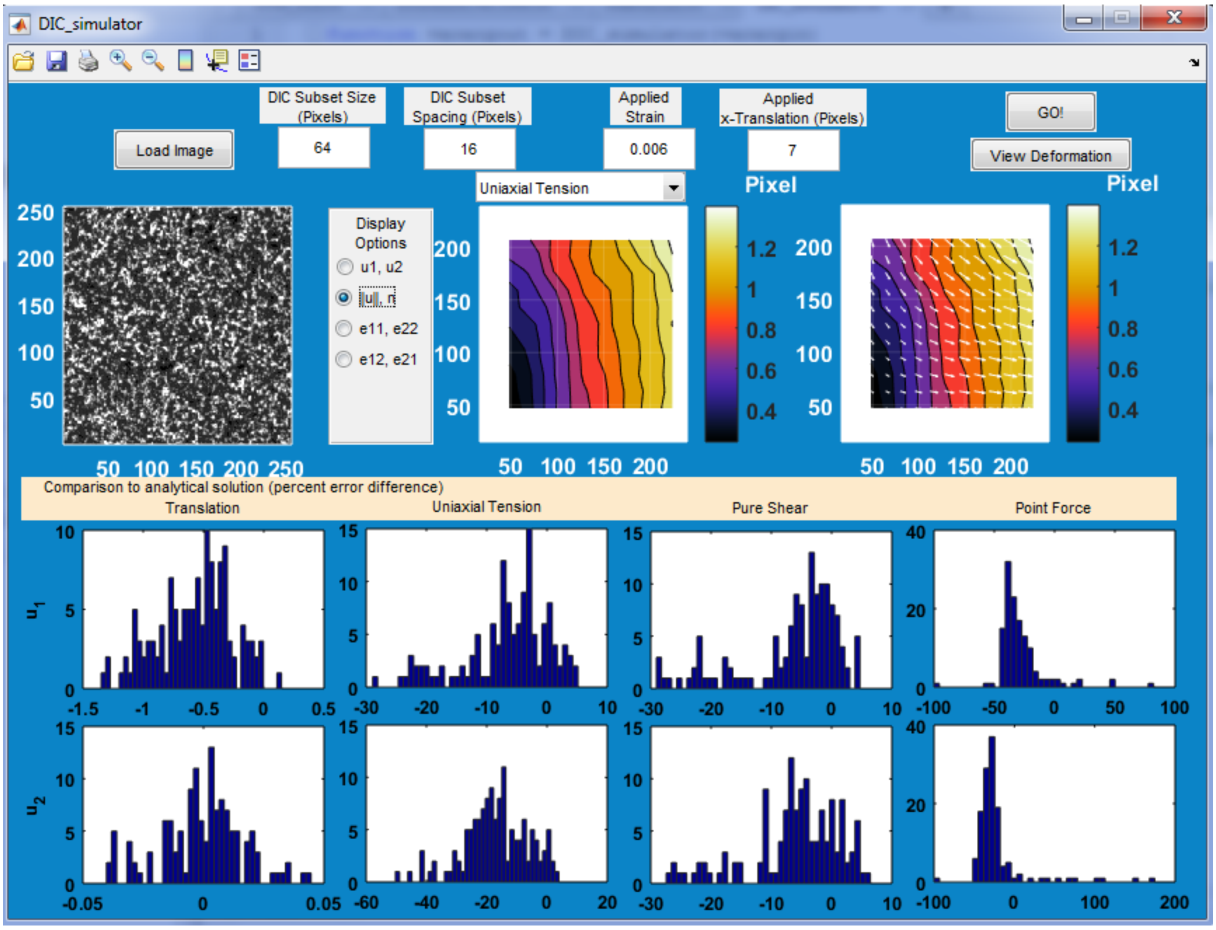


Figure 80. DIC simulator interface

Even though this software allows to perform a speckle optimization, this job has been substituted by taking into consideration [11]. Therefore, it will be just considered as the software outputs the images of deformed and undeformed specimen. The pictures can be seen down below:

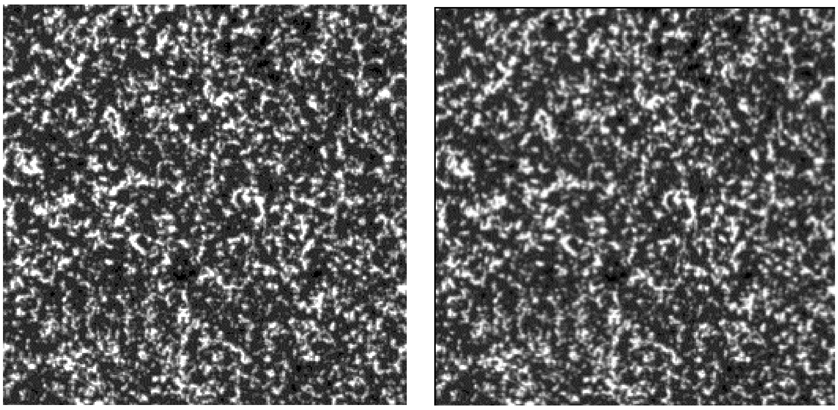


Figure 81. Undeformed and deformed specimen

These images are rotated in order to align the main force with the y-axis, and then they are processed by using the software Ncorr. Here it can be seen the main window of this software. The way of using the software, as well as how to export the data to the GUI is explained on the annex 10. This software is based on the DIC techniques, so a brief introduction to this method is done on the section 4.3**.**

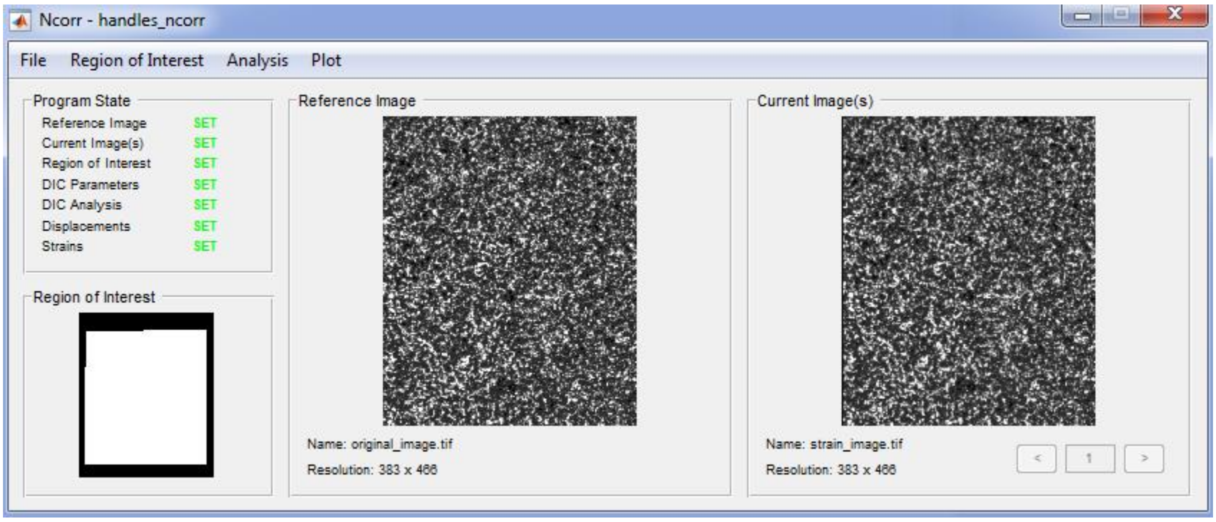


Figure 82. Ncorr interface

After having processed the images on the Ncorr software, the following strain fields are obtained:

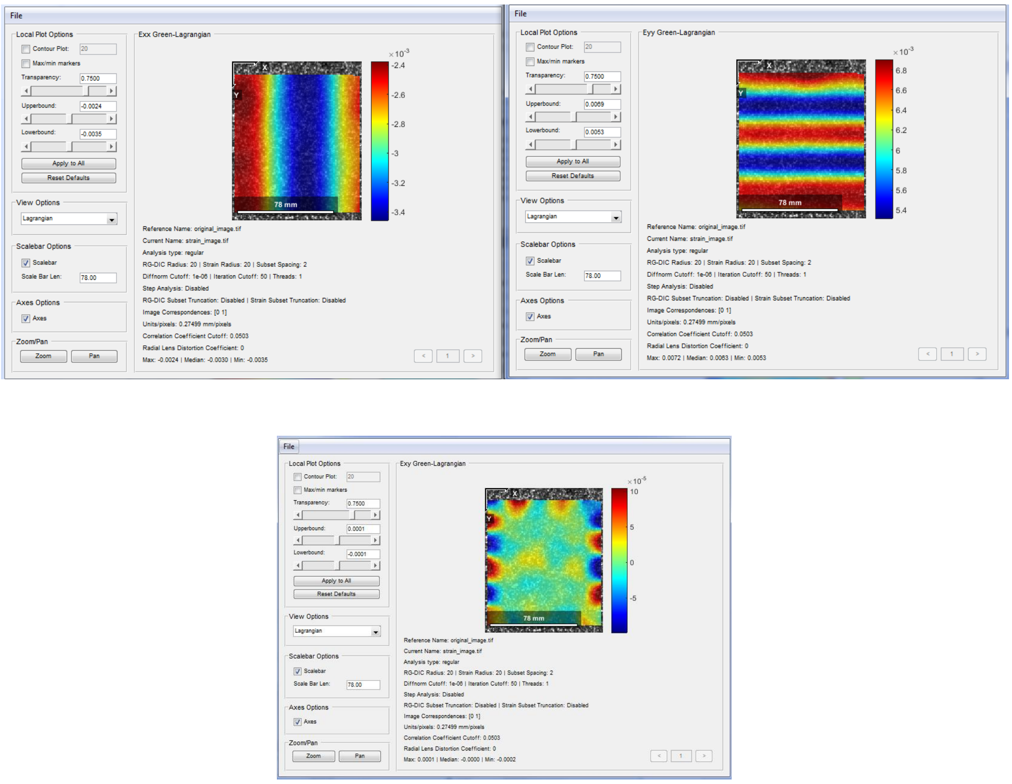


Figure 83. Eps1, Eps2 and Eps6 strain fields plots

Once the data has been exported, it must be introduced into the GUI generated. To that end, some parameters must be supposed. In this case, it is set a thickness of 0.001m and an applied force of 10000N. This gives an $E = 16.67MPa$, and it is known that $\nu = 0.5$ since it is defined on the DIC simulator software.

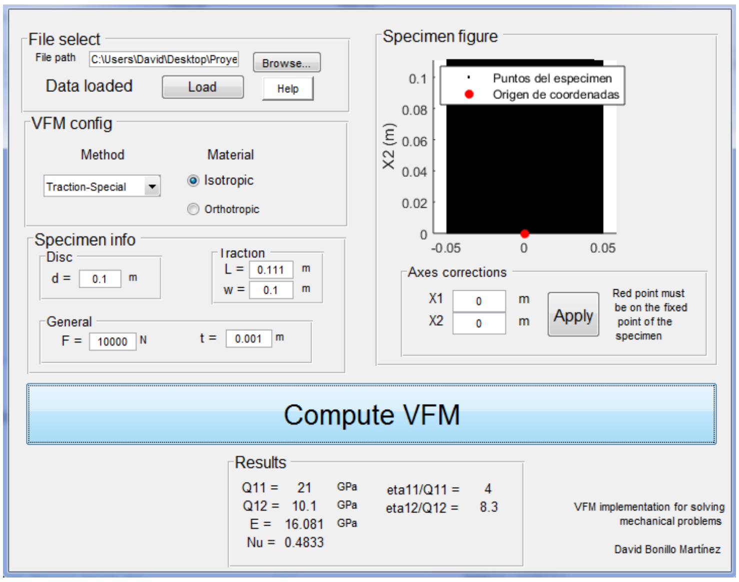


Figure 84. VFM applied to simulated DIC data

As a conclusion, on this section it has been proved the fact that it has been found a way of putting together the DIC software and the GUI that computes the VFM. As seems clear, the next step will be making this same example by using real DIC images.

6.2. VFM on real DIC tests

In this section, the VFM will be finally applied into real specimens whose strain fields are obtained through Digital Image Correlation. To that end, it is used a set whose mains characteristics have been explained at section 3.3.1**.** Three materials are going to be studied: Polycarbonate, aluminium and titanium.

6.2.1. VFM on Polycarbonate

At first, the VFM will be applied on specimens made of polycarbonate. To that end, three different specimens will be defined, which are the three ones that can be seen down below:

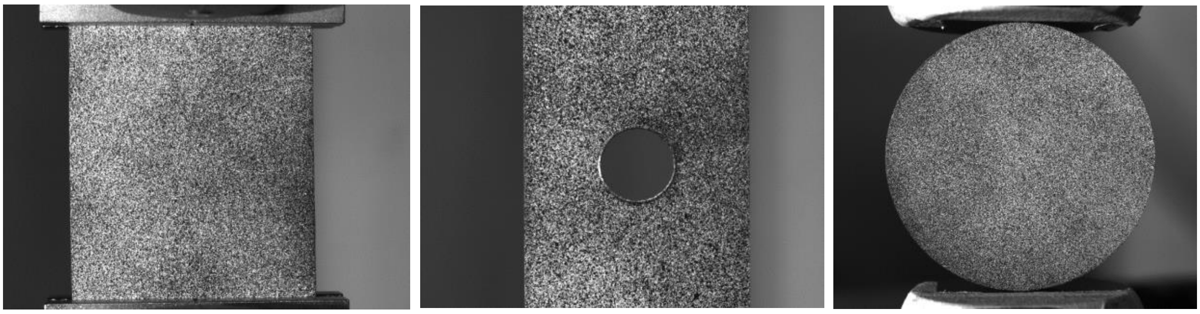


Figure 85. PC specimens used in real tests

Homogeneous specimen:

As a beginning, a homogeneous specimen was studied. 150 photos were taken in five different tensional states of the specimen. This means that 30 photos were taken on each tensional state. Each tensional state was related to a certain load. With this information, it will be possible to reach some conclusions, which will be presented at the end of this section.

First, It is going be shown the results obtained through the software Ncorr at each tensional state:

Finally, these results will be processed using the VFM. According to the VFM fundamentals illustrated at section 5.2.4, it will be demonstrated that in this case it is no possible to compute the Optimized virtual fields, so it is required to l use the Special ones.

- $F = 1000\text{N}$

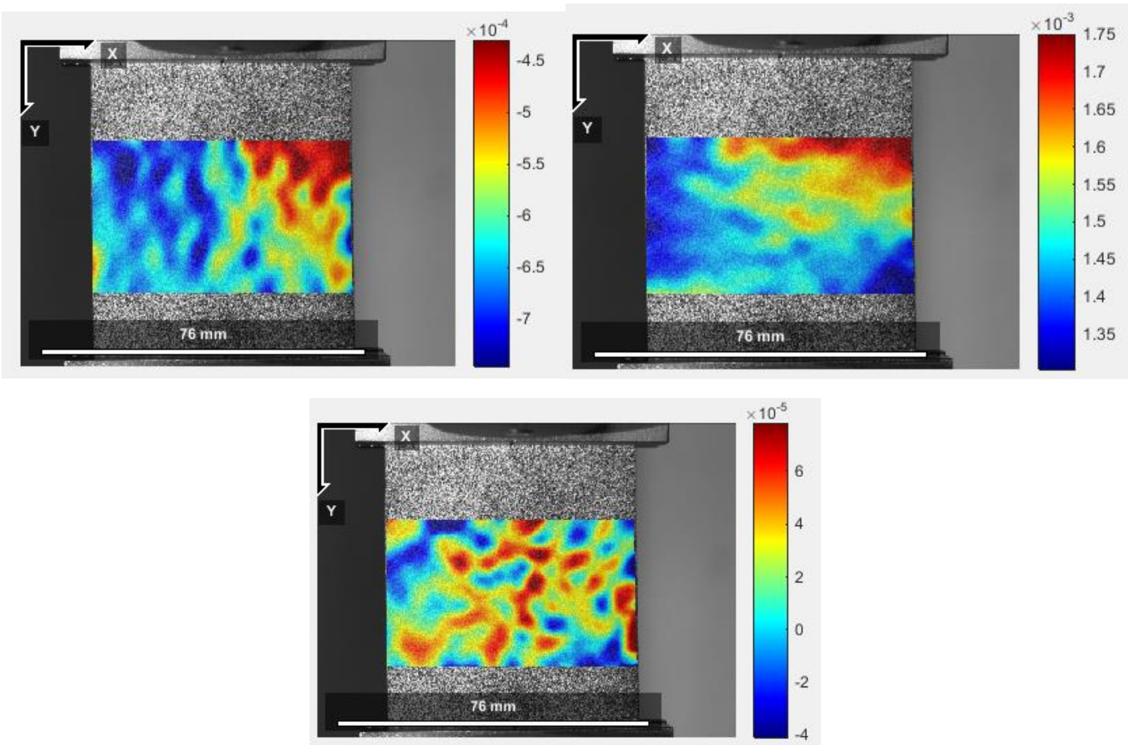
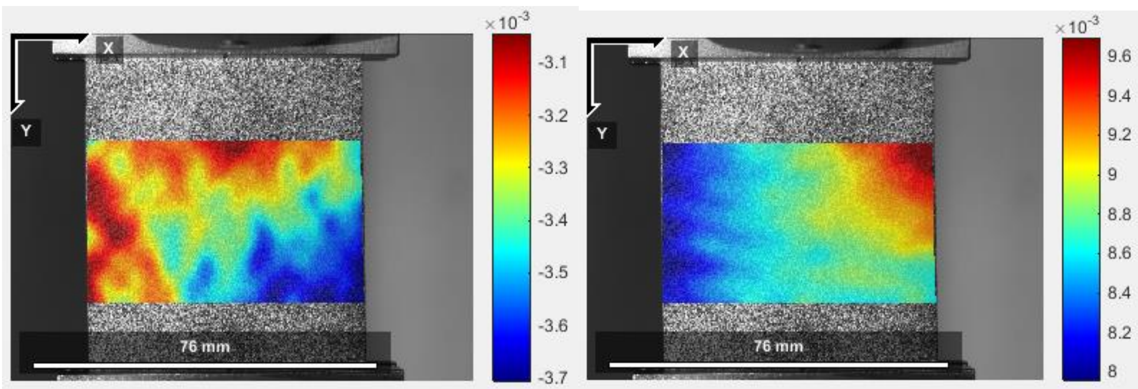


Figure 86. Eps1, Eps2 and Eps6 on the homogeneous specimen (real test, 1000N)

- $F = 5000\text{N}$



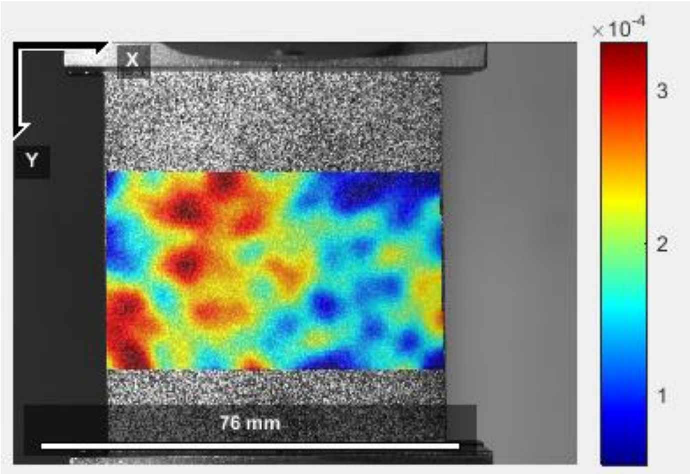


Figure 87. Eps1, Eps2 and Eps6 on the homogeneous specimen (real test, 5000N)

As it can be seen, the strain fields are noisier on the specimen in which less force is applied, since the mm/pixel ratio is much lower than on the case in which 5000N are applied. This noise influence will studied later.

It is also clear that there is some kind of trend in the strains obtained, which does not follow the noisy random pattern that is expected on a homogeneous strain specimen. This is due to the influence of the wedge on the grips, which introduced an undesired moment on the specimen.

That claim is confirmed through the FEM, as can be seen on the following pictures in which a moment has been applied on the superior edge of the specimen. The fields may not be equal in both cases, but they are close enough regarding to the fact that knowing exactly the grips influence on the specimen is not a trivial matter.

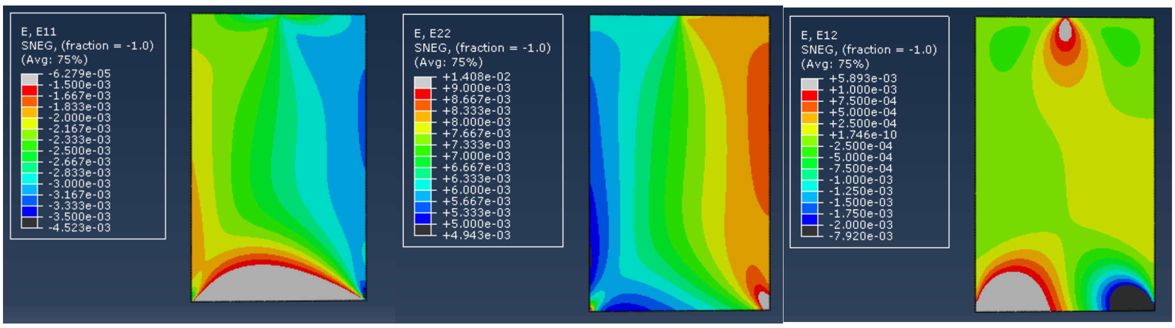


Figure 88. Traction tests including a moment on the edges

This moment is not a problem on the VFM appliance, since through the VF boundary conditions any force on the X-axis or any moment produced by these grips will have no influence on the results obtained, as it will be observed in the results obtained.

The results obtained through the VFM are the following:

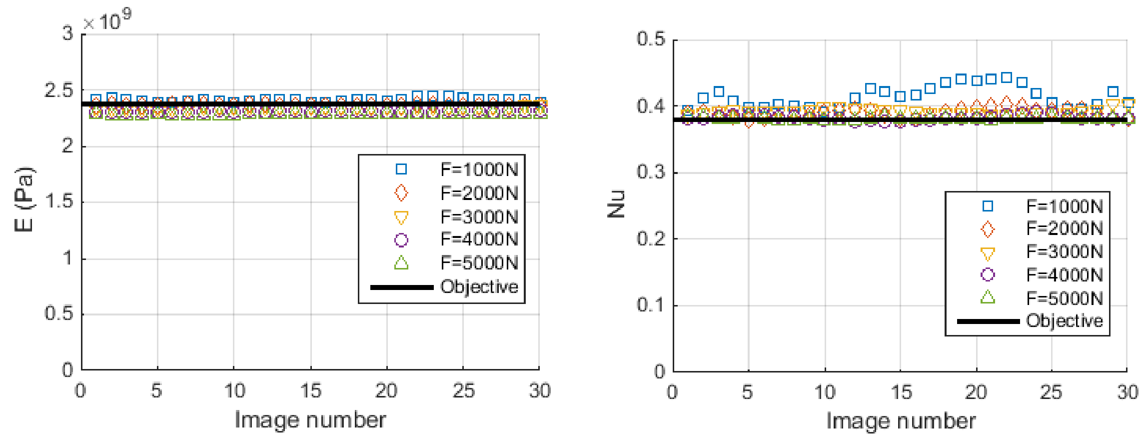
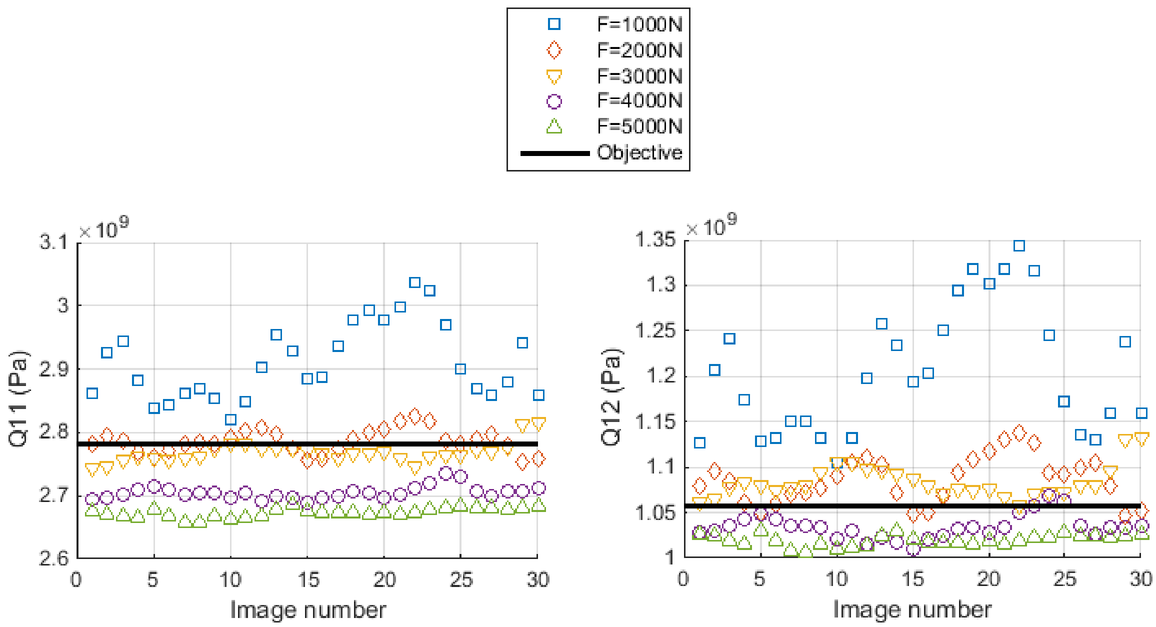


Figure 89. Real traction tests first results

In these first images, it can be seen that the results are really close to the desired ones, even though by using these limits on the y axis lots of information is missed. Therefore,



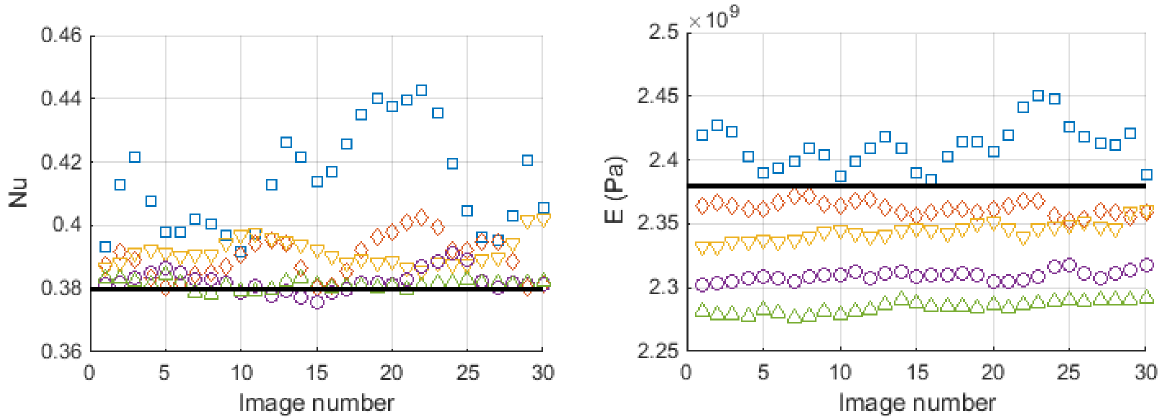


Figure 90. Results obtained on the real homogeneous specimen

Once arrived to this point, some conclusions can be obtained. It is required to identify noise sources before continuing with the analysis. These sources are in this case the noise due to the camera orientation, which will be constant since these experiments are done without any changes during the test set, the noise introduced by the camera sensor and the noise introduced by the DIC analysis.

Due to the specimen shape, it may appear that there is no influence on the DIC parameters since the strain fields obtained are quite homogeneous, so this noise source will be neglected.

Finally, it can be said that in this case the main source of noise is the noise introduced by the camera. Its influence on the results will be lower if we increase the displacement divided by units per pixel (of course, not taking into account rigid solid movements on the test). This means that the noise will be less influent if we increase the applied force, as can be seen on the following graphics:

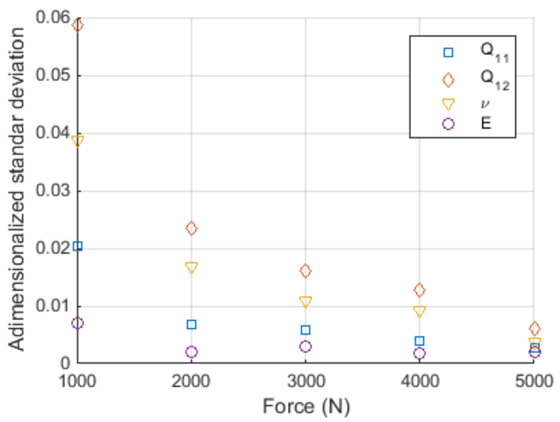


Figure 91. Adimensionalized standard deviation on the homogeneous specimen

Finally, it can be said that the results obtained are really close to the expected ones, obtaining on the better case an error around 2.7% on the Poisson's coefficient, and being E around its normal values, and with an expected progression since its mean value decreases as the force value increases.

Before going on, a second test is going to be performed on the same specimen to check whether or not is possible to obtain repeatability in the results. To that end, the whole set has been disconfigured in order to configure it again. The results will be performed between 1000N and 3000N, just to not keep so much information in this project.

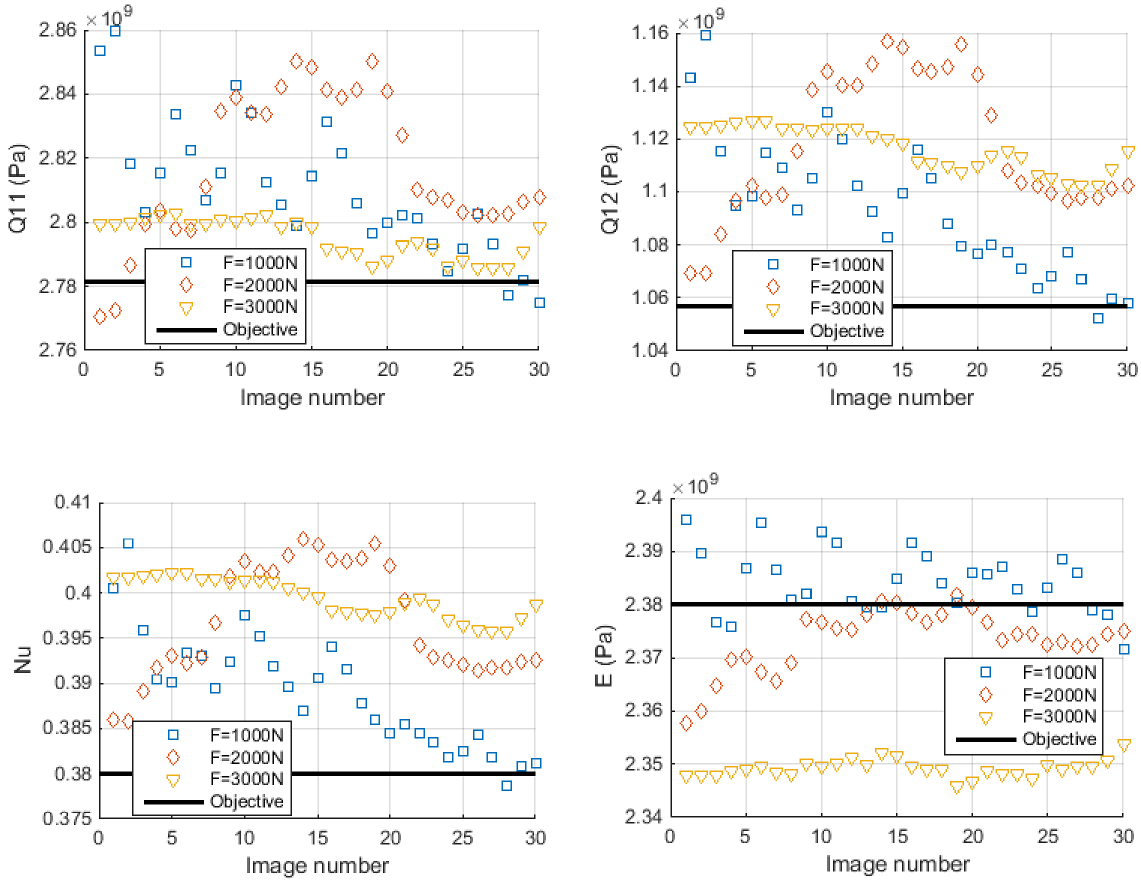


Figure 92. Results obtained on the repeatability test

It can be seen that the results are really close to the ones obtained on the first test, even though we have repeated the most critical ones. It would be interesting doing some more tests in a more accurate way, but this job has not been done in the present project because of a lack of time.

It may seem obvious the fact that it is senseless to compute a homogeneous specimen using the VFM, since to obtain its mechanical properties it can be used a couple of strain gauges. Therefore, for finishing the calculations involving this specimen it is going to be computed a test in which the specimen is turned between 15º and 20º, and 5000N are applied. The strain fields can be seen on the following pictures:

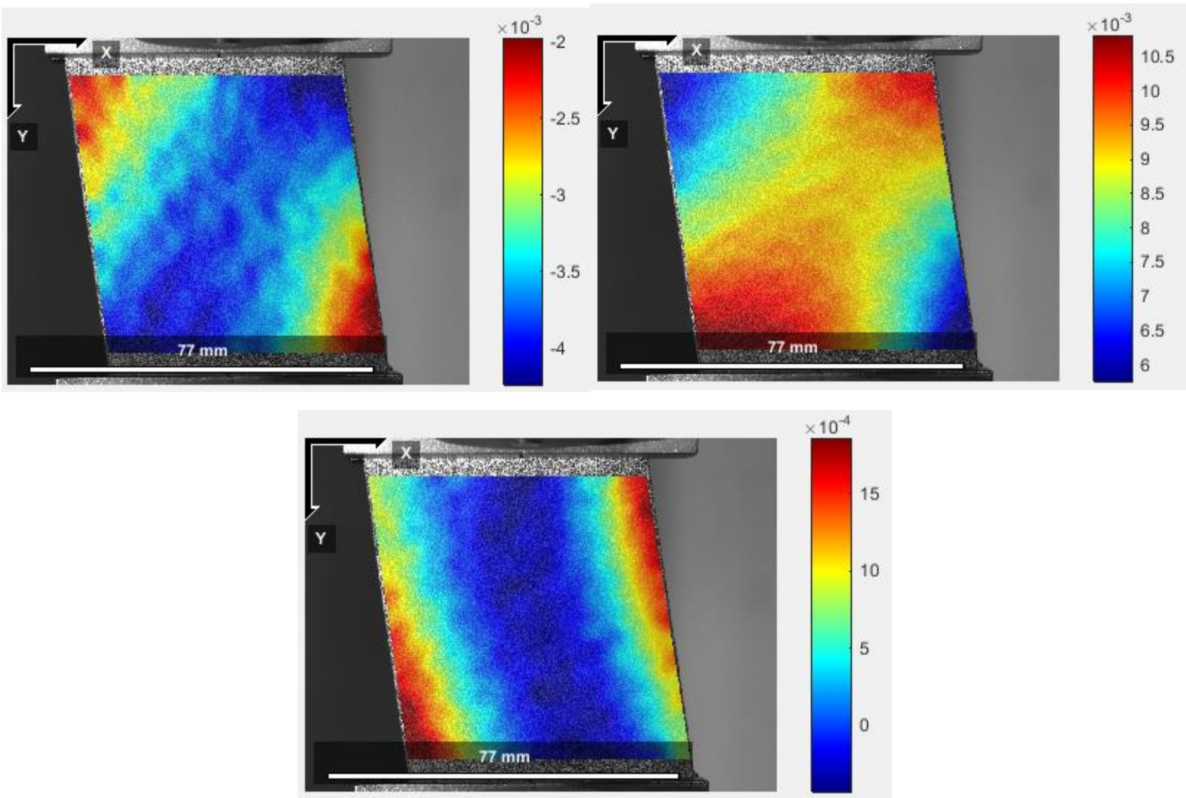


Figure 93. Eps1, Eps2 and Eps6 on the turned homogeneous specimen (real test, 5000N)

As here there is just one tensional state (the one that matches with 5000N), the results are going to be presented on the next table:

Parameter	Mean value	Error (%)	Adimensionalized standard deviation (%)
Q_{11}	2.804 GPa	0.803	0.0580
Q_{12}	1.104 GPa	4.43	0.2161
ν	0.3937	3.61	0.1749
E	2.369 GPa	0.462	0.0478

Table 11. Results obtained on the turned homogeneous specimen

For these calculations it has been employed the Special Virtual Fields, since the gradients seem not big enough as for using the Optimized ones. It can be seen that the results are close to the expected ones, but not as close as they were on the previous tests. This may be due to the loss of information that happens on the DIC technique as the gradients increase, due to both the strain radius and the subset radius.

Hole specimen:

In this section the specimen will show some more gradients, which must be taken into account while performing the DIC analysis. Because of that, the pictures will be processed without any subset spacing (which means that the DIC will be applied into every pixel) and with lower subset radius and strain radius. The strains obtained can be seen on the following pictures:

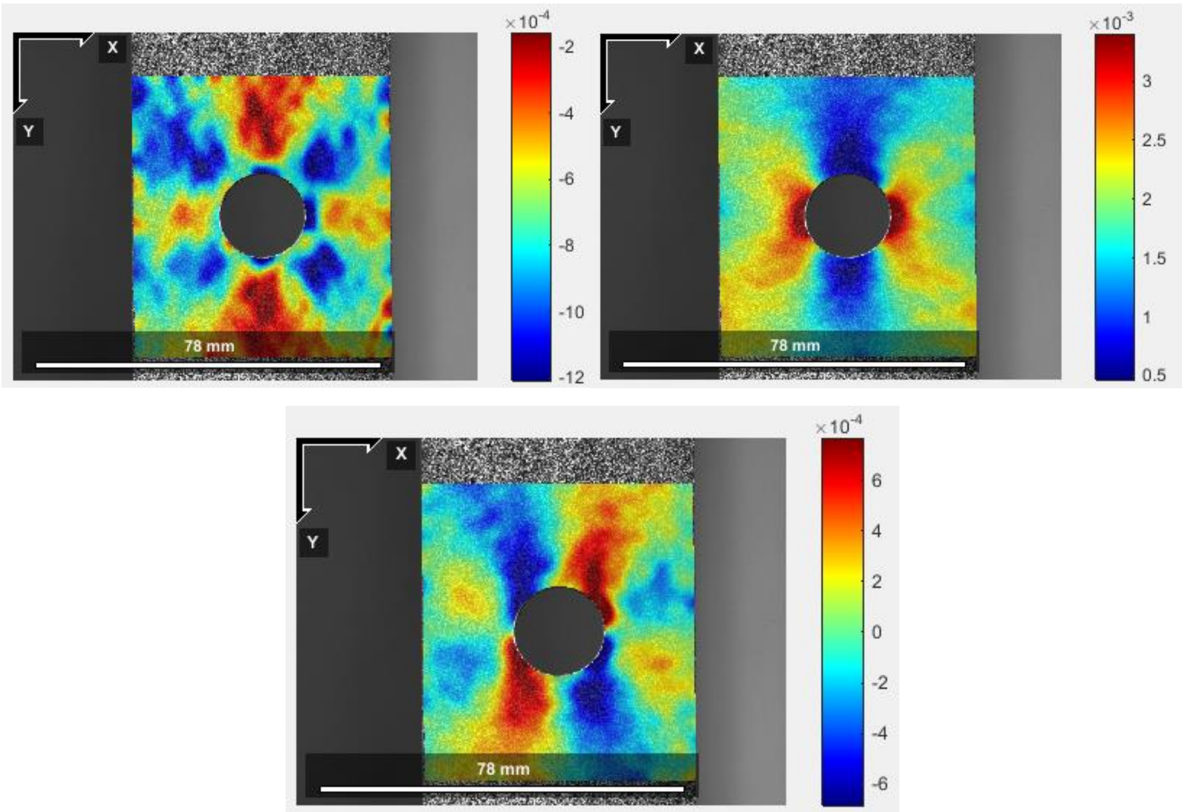


Figure 94. Figure 3. Eps1, Eps2 and Eps6 on the hole specimen (real test, 3000N)

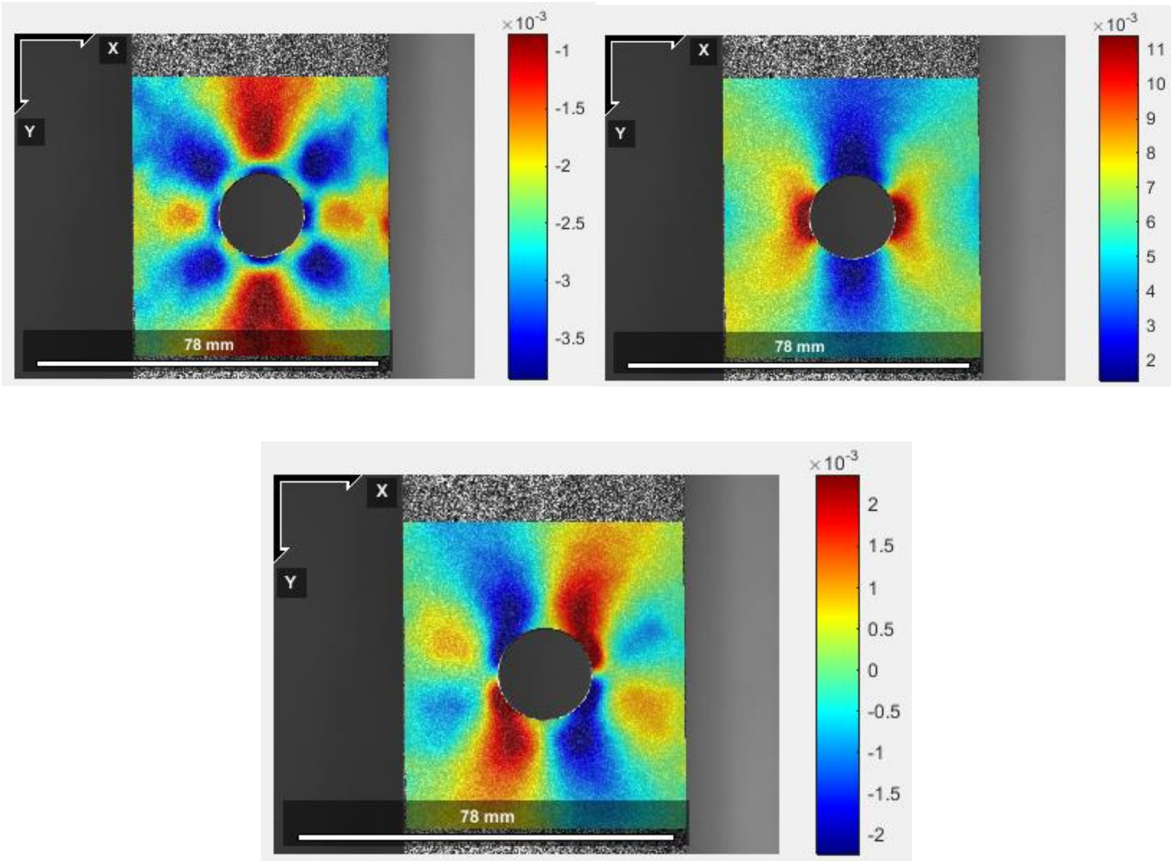
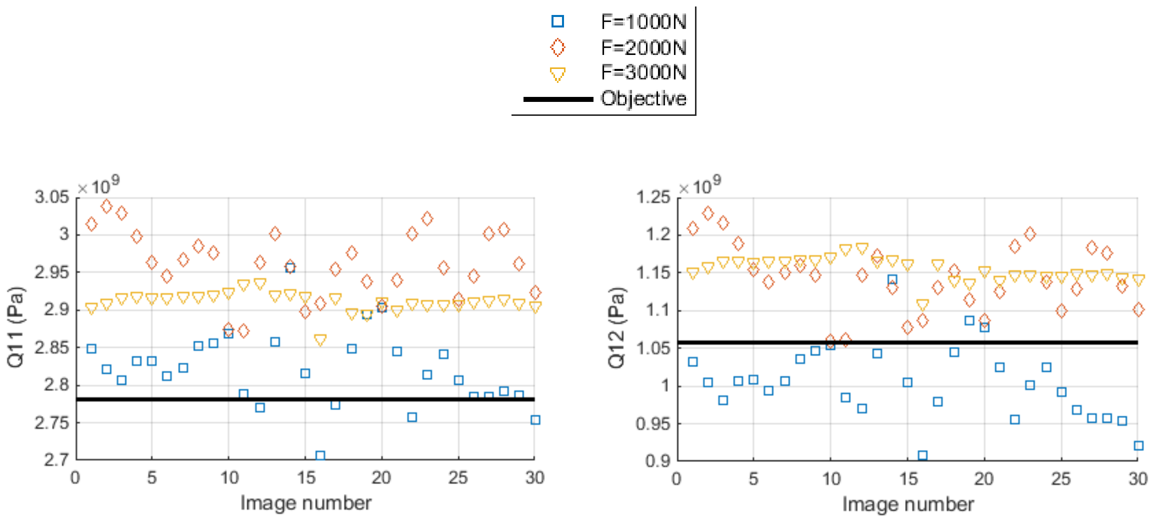


Figure 95. Eps1, Eps2 and Eps6 in the hole specimen (real test, 3000N)

First, the results concerning the Special Virtual Fields will be considered. They can be seen on the graphics down below:



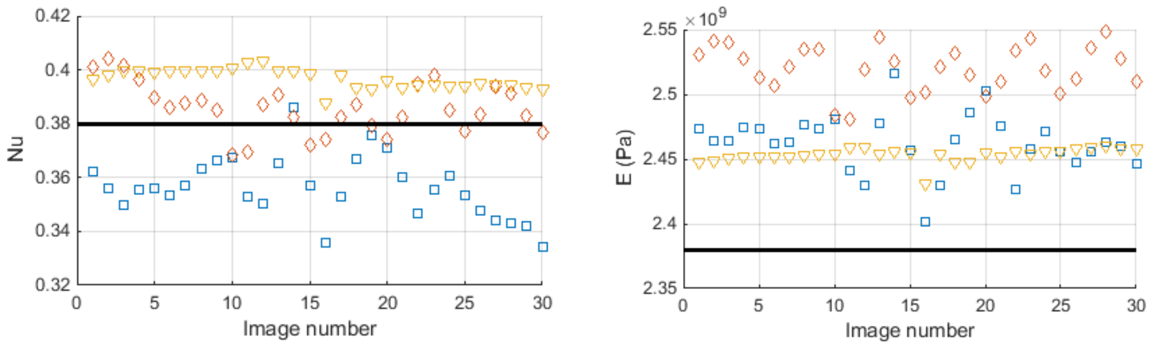


Figure 96. Results obtained on the real hole specimen (Special VF)

Here, the results obtained are further from the objective, about 3.07% on the Young modulus and a maximum of 4.40% of error from the Poisson ratio. This may be due to the DIC analysis, in which more information is supposed to be missed since there are high strain gradients.

Before reaching any conclusion, the same data will be processed by using the Optimized Virtual Fields. The results can be seen down below:

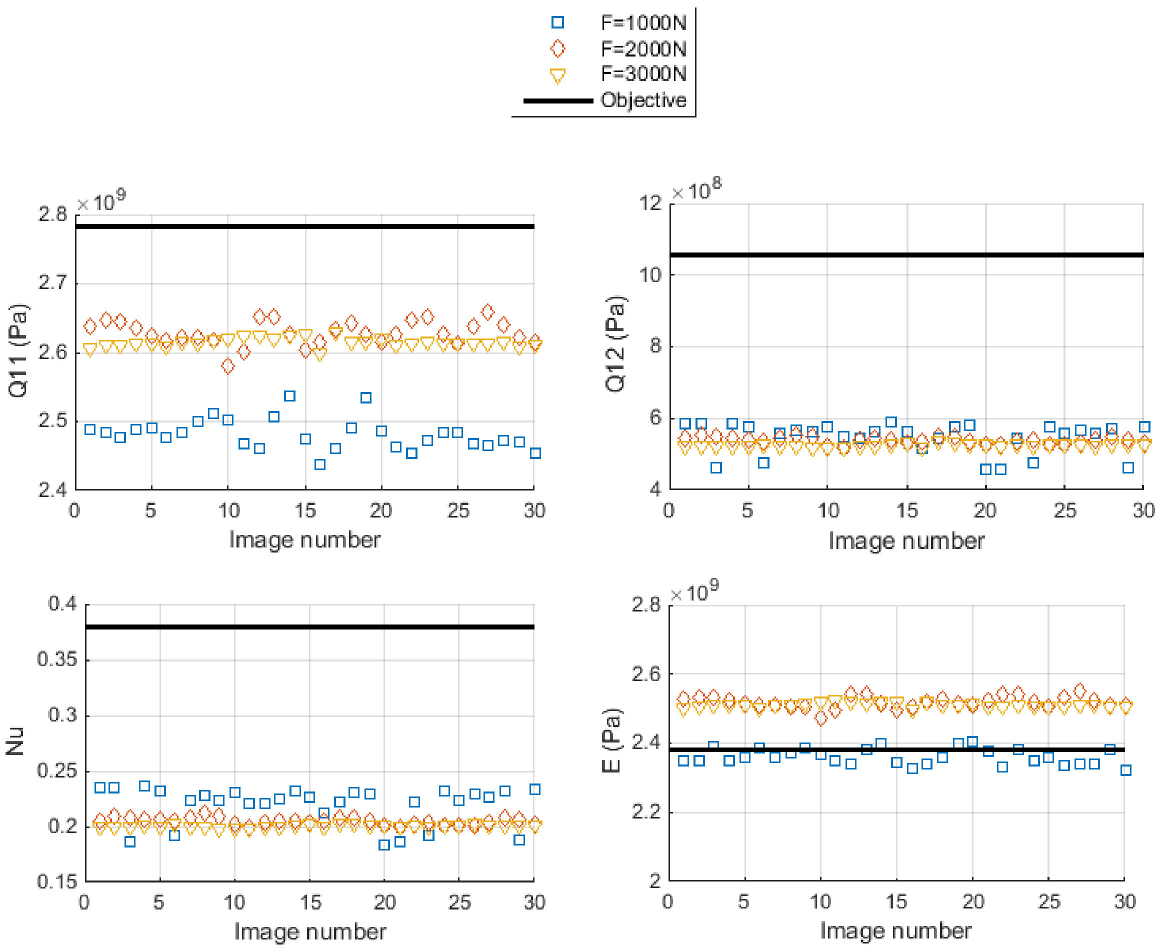


Figure 97. Results obtained on the real hole specimen (Optimized VF)

At first view, as the results are far from the desired results it may seem that the calculations are wrong, but it must be taken into account the fact that there are lot of noise sources in the extracted data. Therefore, the η parameters are goin to be analyses in detail. If the medium η_{12} values are considered, in the case of the Optimized VF the noise dependence is 3.96 times bigger than this dependence on the Special VF, which explains these results.

The fact that the VFM has been applied in a good way has been checked through simulating the same geometry, with the same loads, and the results obtained are the expected ones. In addition, there is a bigger noise influence (computed through the η parameters) if the optimized virtual fields are employed, which matches with the results presented in section 5.2.4.

Test with a disc under compression.

The final test performed on Polycarbonate will be a compression test. To that end, it is used a disc 10mm thick and a diameter equal to 60mm.

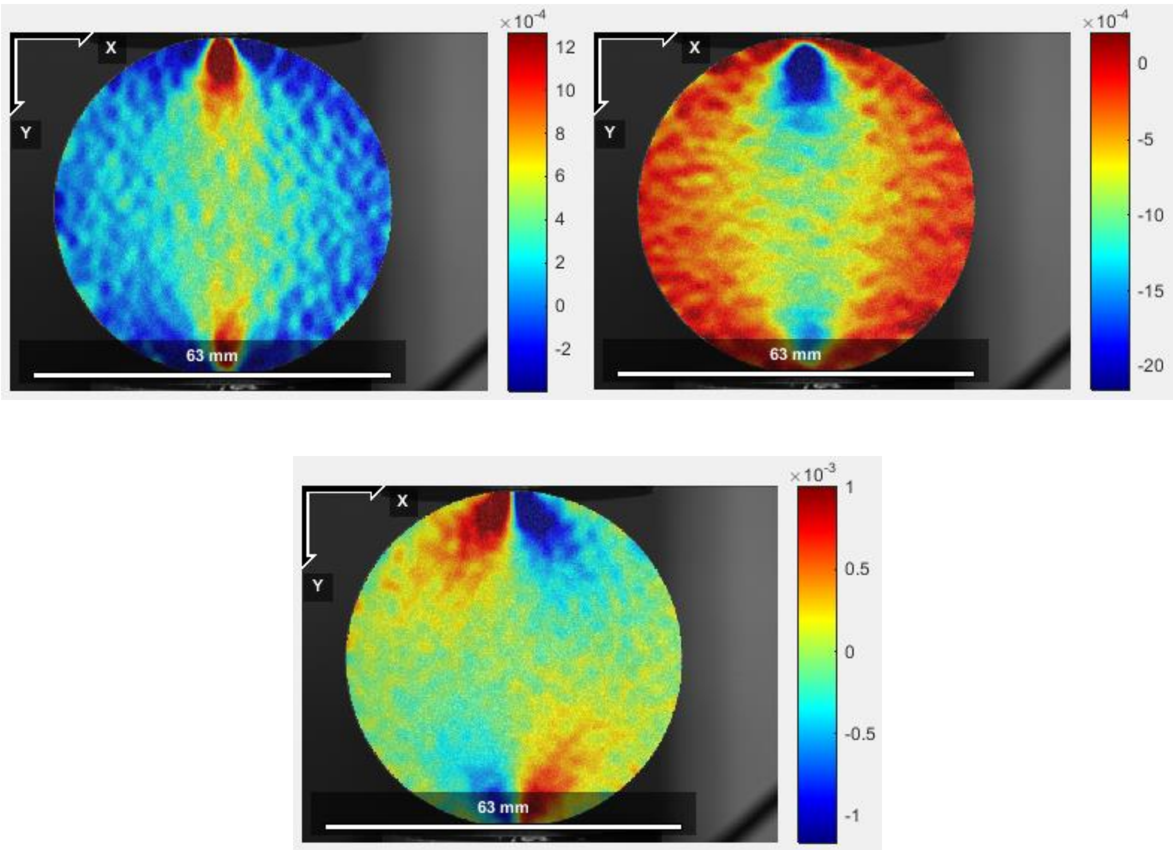


Figure 98. Eps1, Eps2 and Eps6 on the disc specimen (real test, 600N)

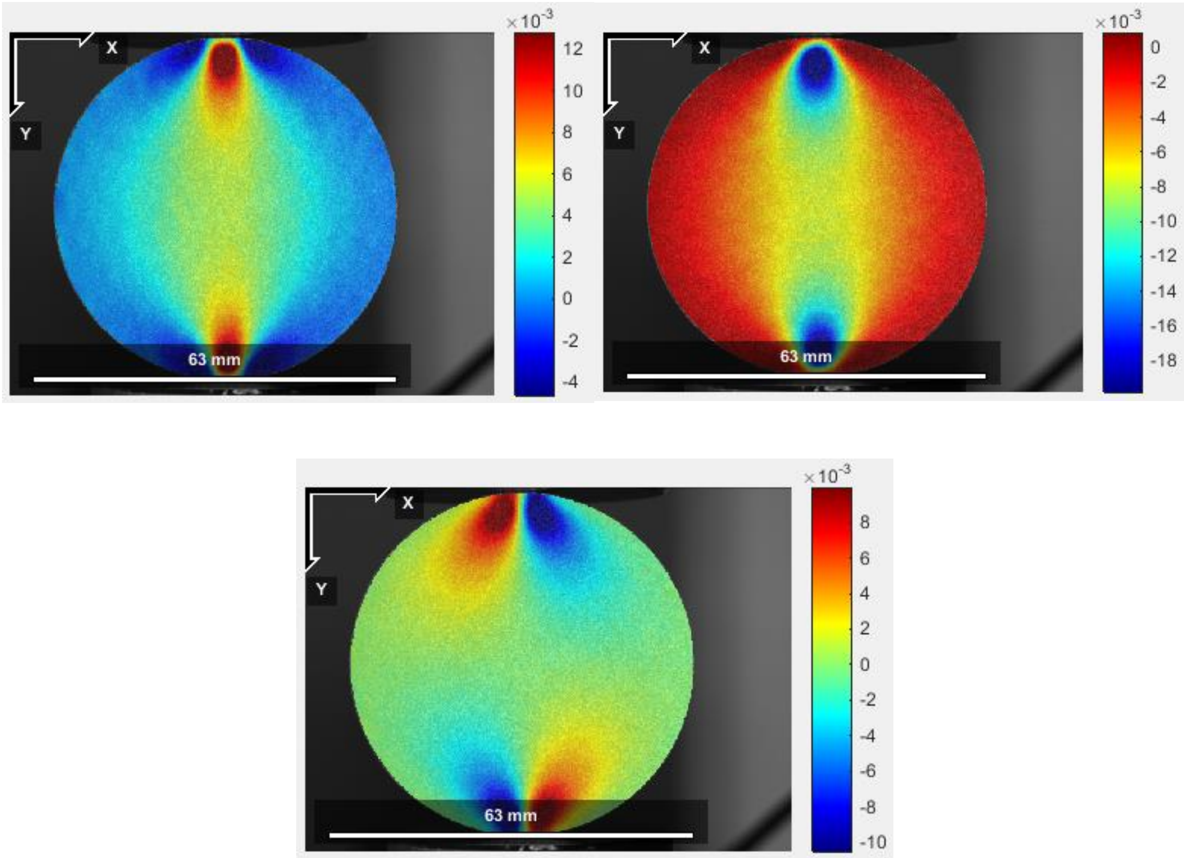
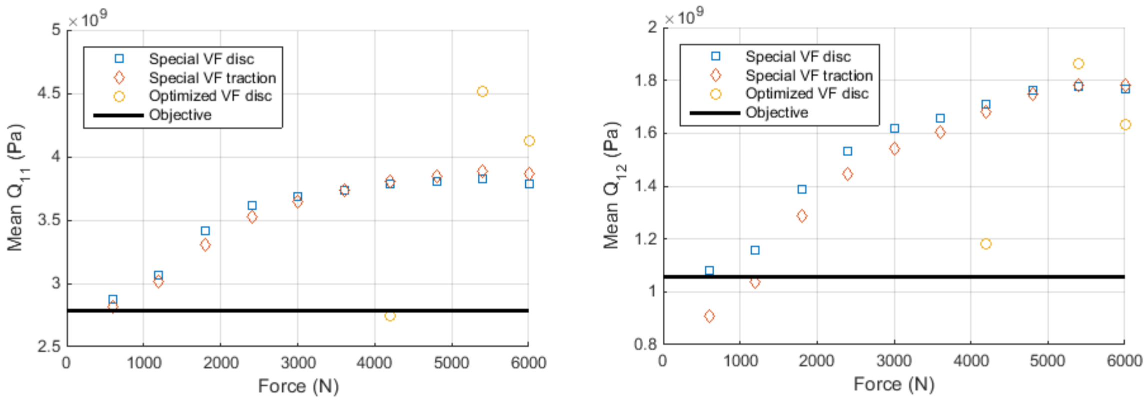


Figure 99. Eps1, Eps2 and Eps6 on the disc specimen (real test, 6000N)

As it can be seen, there is a lot of noise in the image when a small load is applied. This is due to the low value of the displacements to pixels per unit ratio. This fact will be discussed later. After the VFM appliance, the results obtained are the following ones:



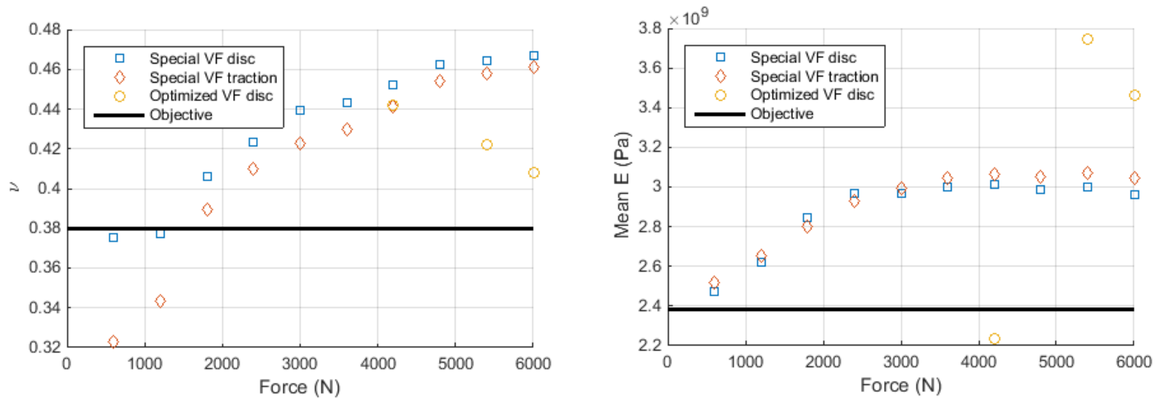


Figure 100. Results obtained on the real compression disc

It can be seen that the results tend to stabilize on a value far from the reference values, which may be due to some circumstances, such as some misalignment on the specimen, a wrong DIC analysis or regions in which there is some plasticity. This will be discussed later. Now, the results of the noise dependence will be analysed.

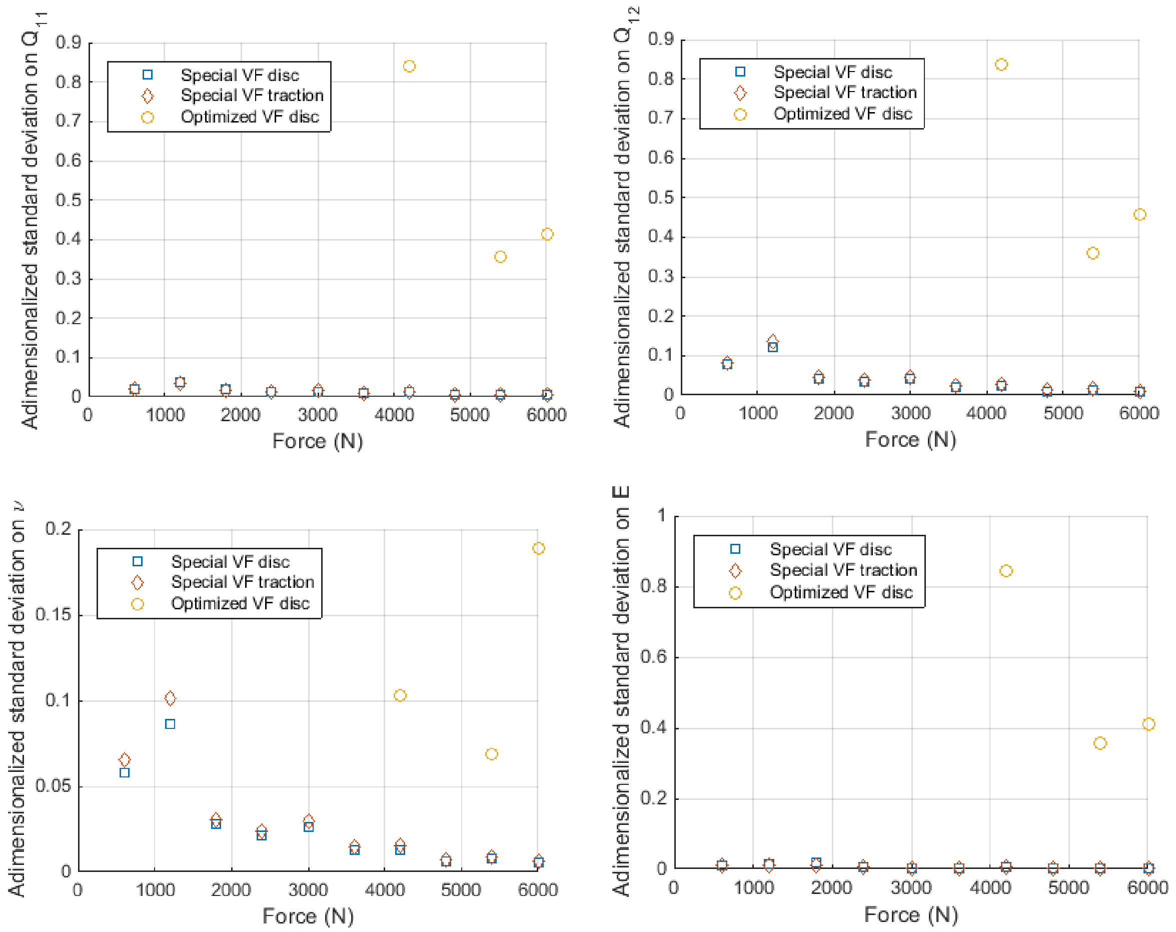


Figure 101. Results obtained on the real compression disc

It seems obvious that the best option are the Special VF computed for a disc. Results are close to the ones obtained on the Special VF defined for a traction test, Anyway, the Optimized VF seems to not being an option, just like it happens on the traction test. This will be analysed at the final conclusion section of the present project. Due to that fact, the Special Virtual Fields for a disc compression tests have been added to the Matlab VFM GUI:

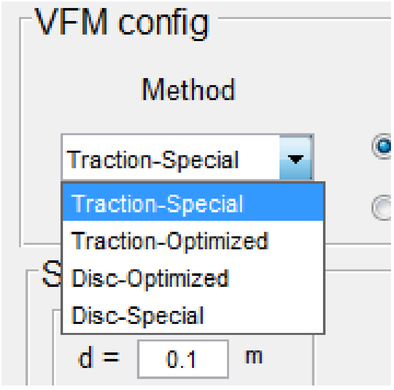


Figure 102. Methods available on the final VFM GUI

Now, the first results will be taken into account. By simulating the specimen, it seems that there is some plastification on the top and bottom of the disc. In addition, at the middle section of the disc the displacements are very small, which means that the displacement to units per pixel ratio is low, allowing the noise to have a big influence. Nevertheless, by neglecting just the top and bottom we can process a higher number of points, which will reduce the noise influence since its average value may decrease.

Taking all this into account, it will be processed a region 1 cm far from the top and 0.5 cm far from the middle line (left picture, region 1 hereinafter), and a region 1 cm far from the top and 1 cm far from the bottom (right picture, region 2).

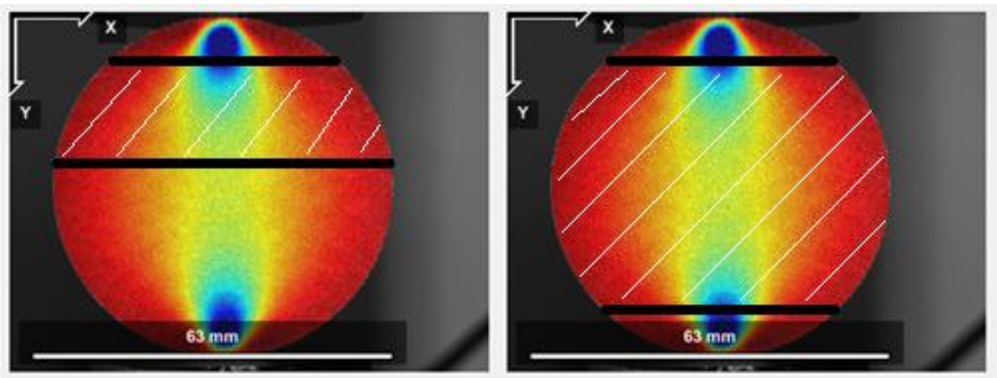


Figure 103. Regions computed on the disc test

These sections must be computed through the Special VF traction method, since now the load is not applied in a single point, but will be applied on the whole marked line. The results obtained can be seen in the following graphics:

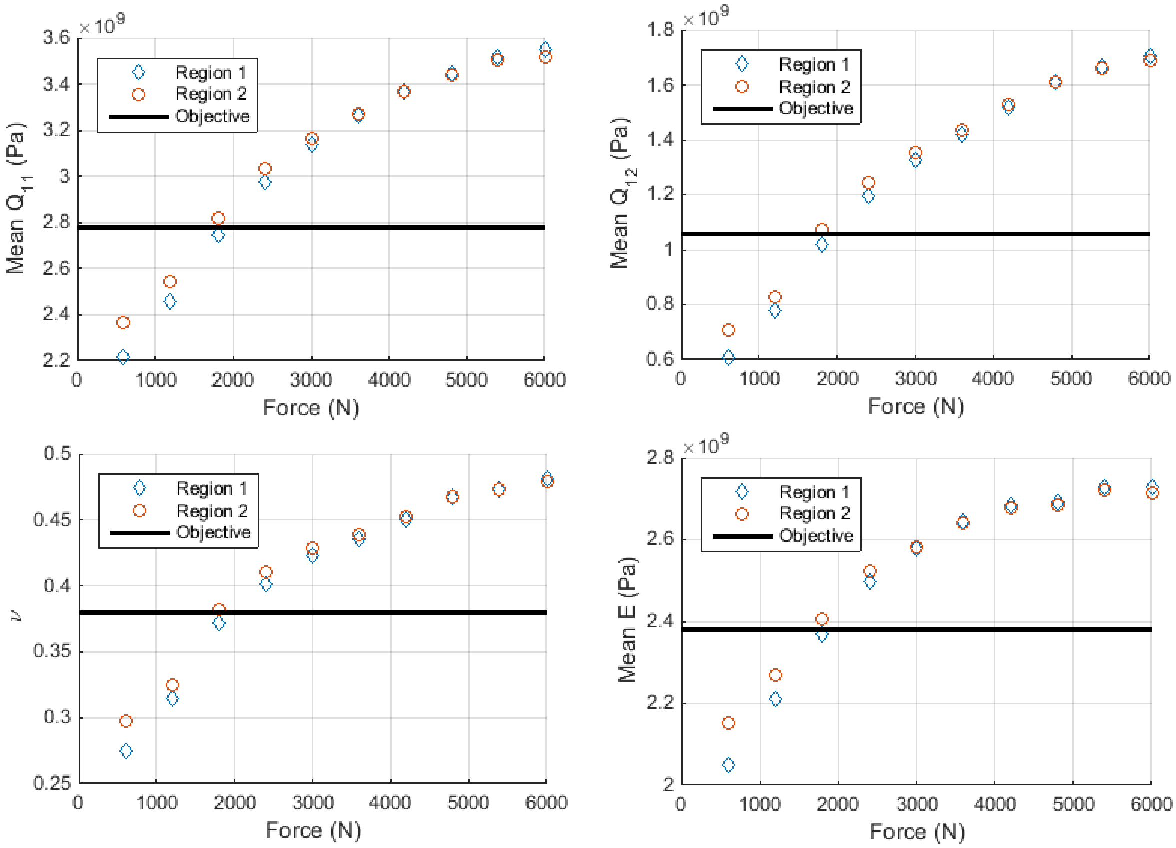


Figure 104. Results obtained on the fractioned disc compression test

It can be seen that the results are more accurate this time, since on the Young modulus the values tend to stabilize computing an error about 12.5%, which is the half that on the previous case, in which the error on this parameter was about 25%. Therefore these regions in which there are high strain values and gradients affect adversely to the obtained results, either due to the plastification zones or to a bad DIC processing on these zones (mainly due to loss of information located on the high strain field gradients regions).

If the noise influence is compared, it is possible to see which region is the least noise-dependent. This can be done by looking at the following graphs:

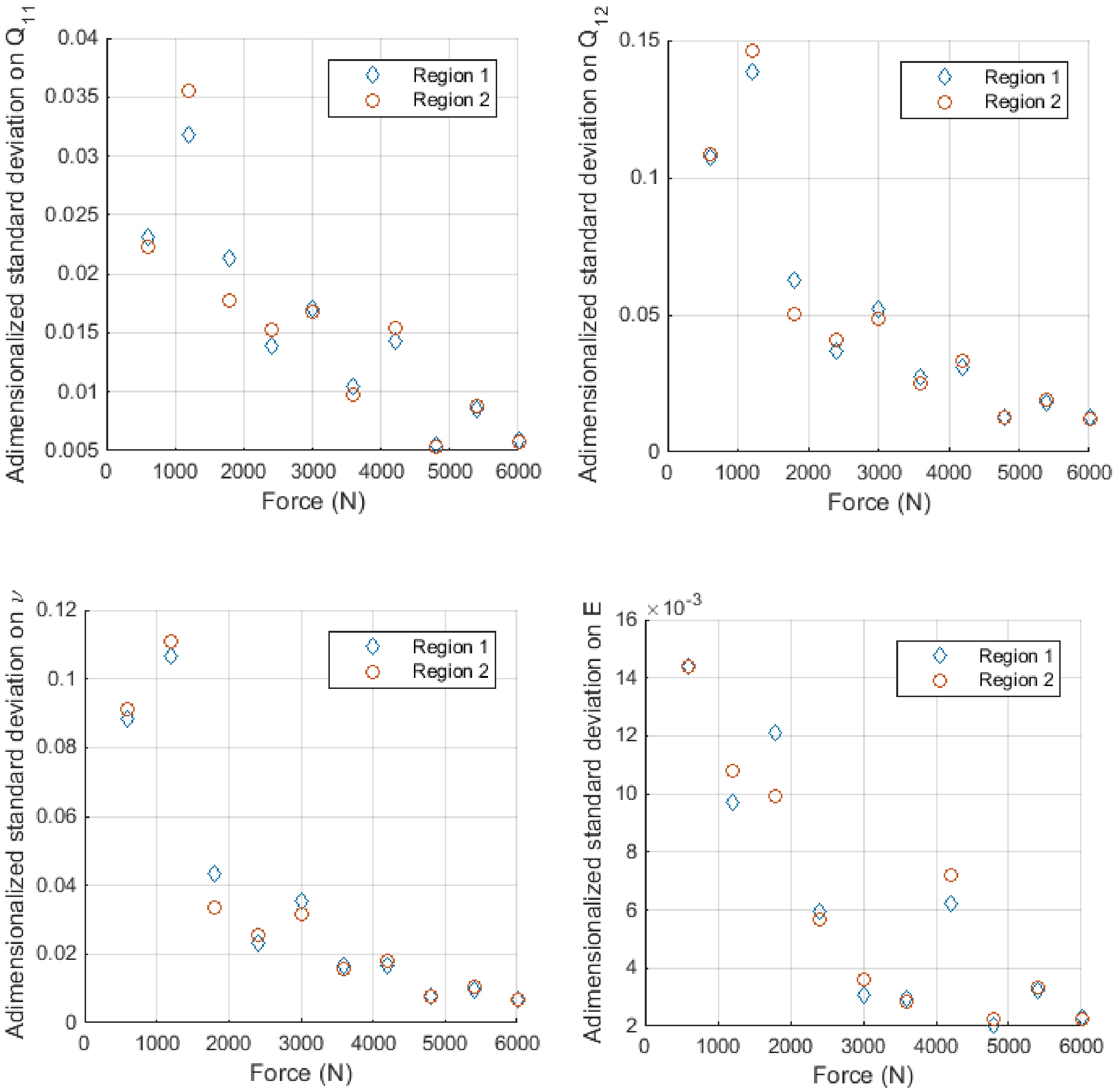


Figure 105. Noise dependence on the partial disc compression

It can be seen that there is no difference between both regions. Therefore, the small strains on the middle region of the disc are not that noisy, so they can be computed.

After that, it has been checked the fact that the misalignment is not influencing the results. To that end, it has been used a FEM simulation in which the disc has been turned 5º. The test can be seen on the following picture:

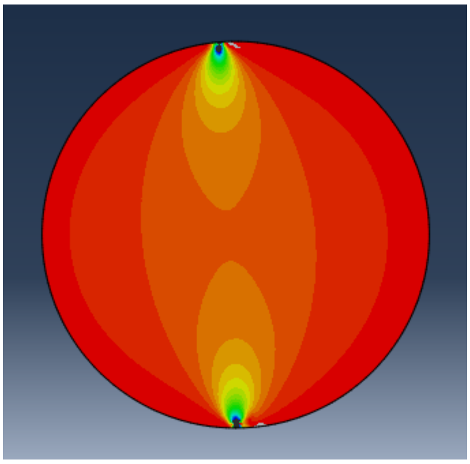


Figure 106. Turned disc

In this test, it has been proven that this effect has almost no influence since the results have varied in less than 1%, which is neglectible compared to the 35% error that is being obtained on the ν value.

Maybe, all these errors are coming from another fact, which is the specimen thickness. On the previous specimens, the thickness was equal to 4mm, but in this specimen it is equal to 10mm.

6.2.2. VFM on Aluminum 2024-T3

In this section it is going to be tested the VFM performance considering aluminum specimens, which will be processed through DIC just like it is done on the previous section.

Homogeneous specimen:

First, a homogeneous specimen will be tested. It will be applied a load of 1000, 2000, 3000 and 4000N. The strain fields obtained are the following ones:

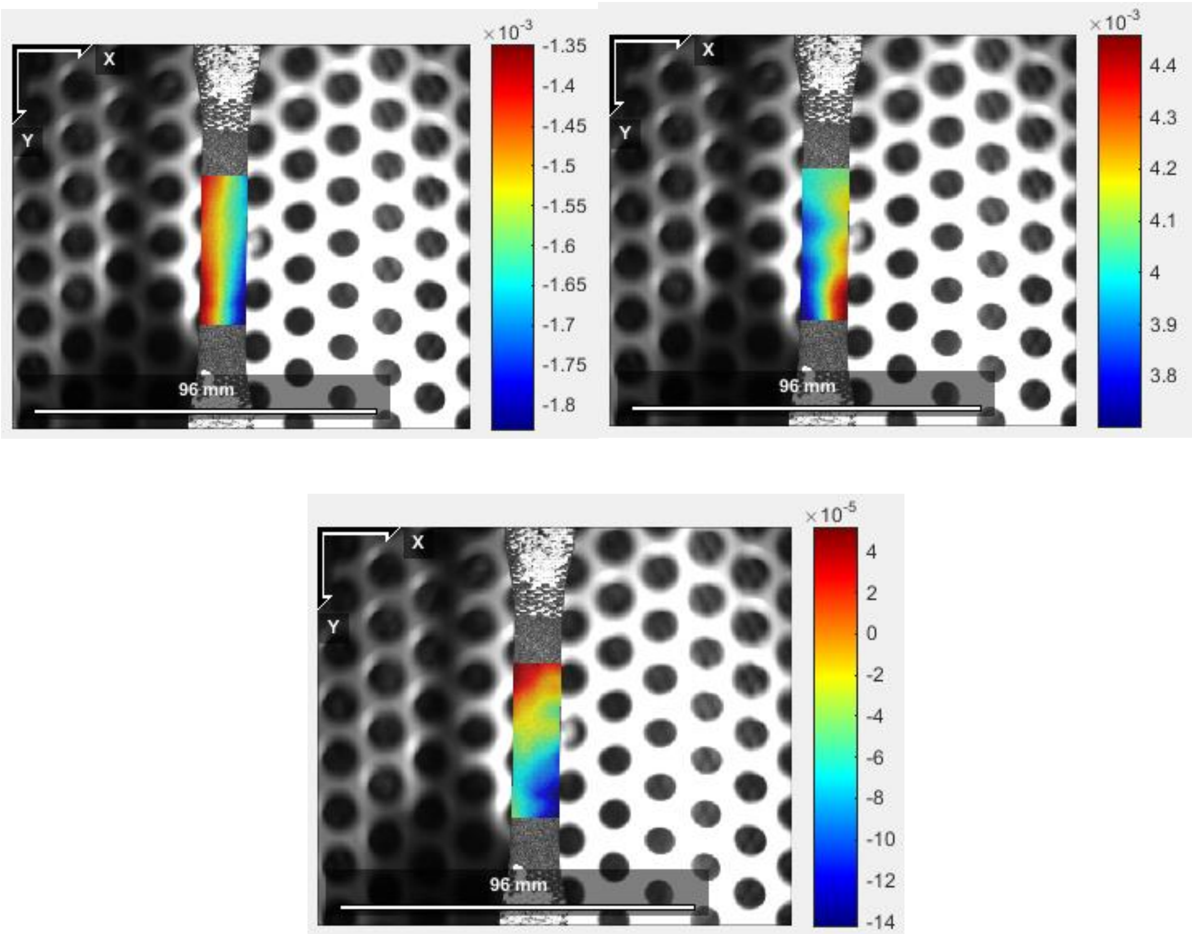
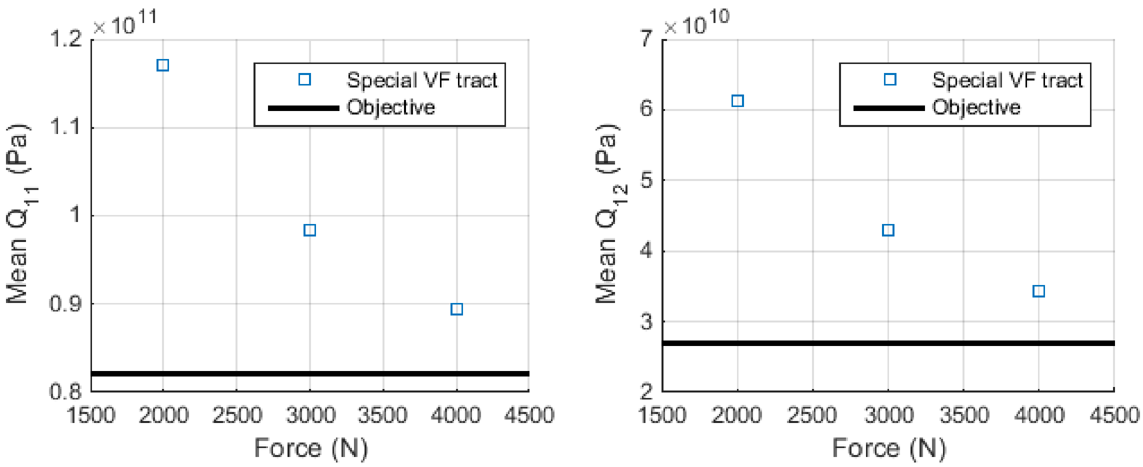


Figure 107. Eps1, Eps2 and Eps6 on the aluminium traction specimen (real test, 4000N)

As it is obvious, these data will be processed through Special VF. Through this processing, it has been clear that by using 1000N the results are too noisy as for applying the VFM, so the results will not be presented for this load. These results can be seen in the following graphs:



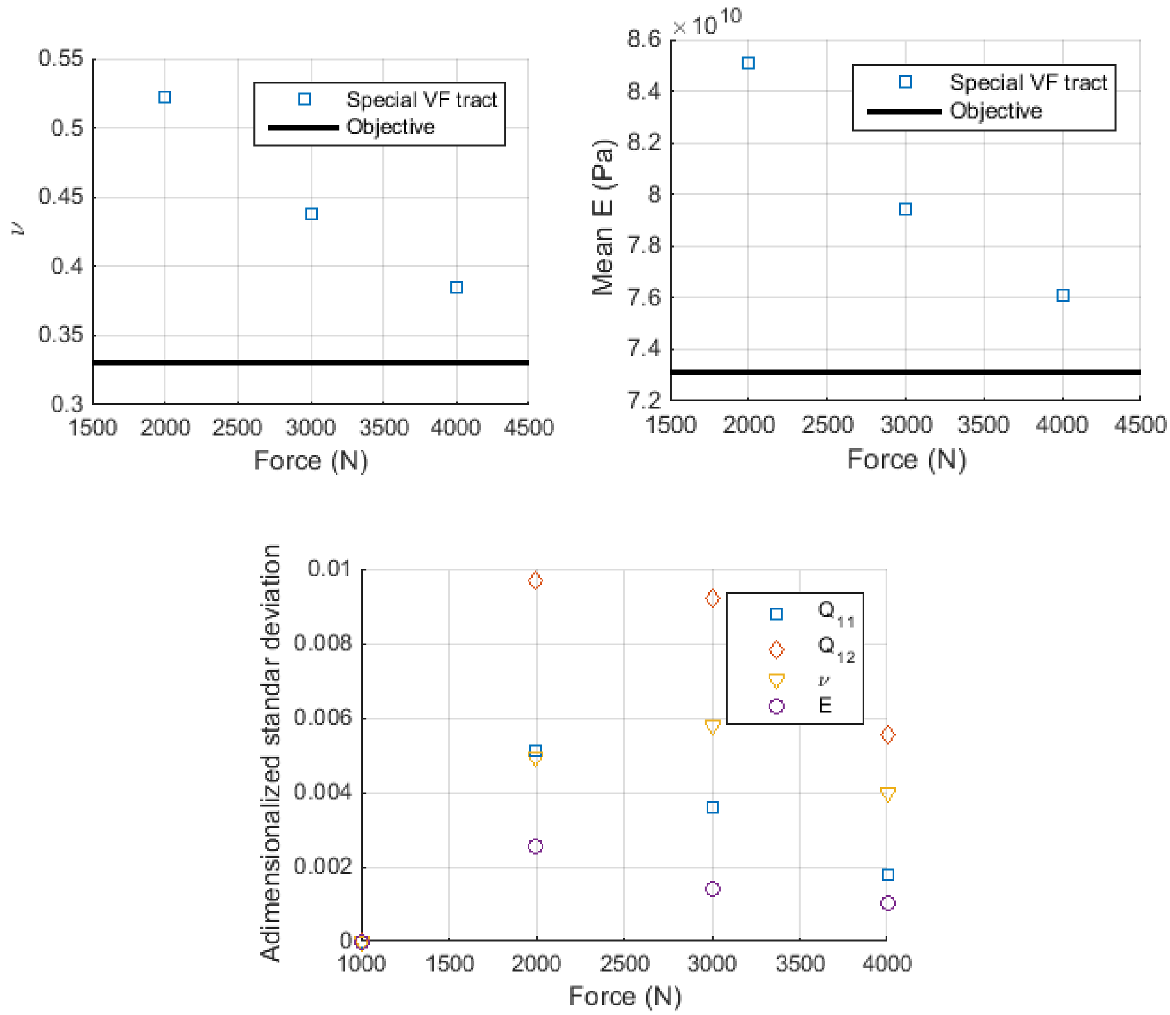


Figure 108. Results obtained on the aluminium real traction test

As it can be seen, the results tend to stabilize while the force increases, which has been justified in the previous section. The problem in this case is that the strains are not large enough as for being relevant the noise introduced throughout all the process. It can be said that the force could have been increased in order to increase the strains, but it must be taken into account the fact that it is needed to work in the elastic region of the stress-strain curve. By using a load of 5000N the specimen will be introduced into the plastic region, which will lead to inaccurate results. This fact is going to be studied in the specimen tested in the following section.

In this specimen, it has been obtained a final error of 4.11% on the Young Modulus and 16.73% on the Poisson ratio. This can be explained through the fact that the ε_{11} strains are around 4 times lower than the ε_{12} ones, which mean that the first ones will be much noisier, and this noise will have a much bigger influence in the final results.

Finally it must be said that it was intended to perform this test after an image rotation using the function “imrotate” in Matlab, but there is some loss of information in this process, which leads to some strange regions in the displacement fields, leading to inaccurate results. Therefore, this option was not considered.

<u>Specimen with a central hole:</u>

In this section it is going to be performed the same tests on an aluminum specimen with a central hole under a traction state. In this case, the Optimized VF will not be employed since it has been showed in a previous section that it is not so effective in the real tests.

The studied specimen and the strain field obtained are showed in the following pictures:

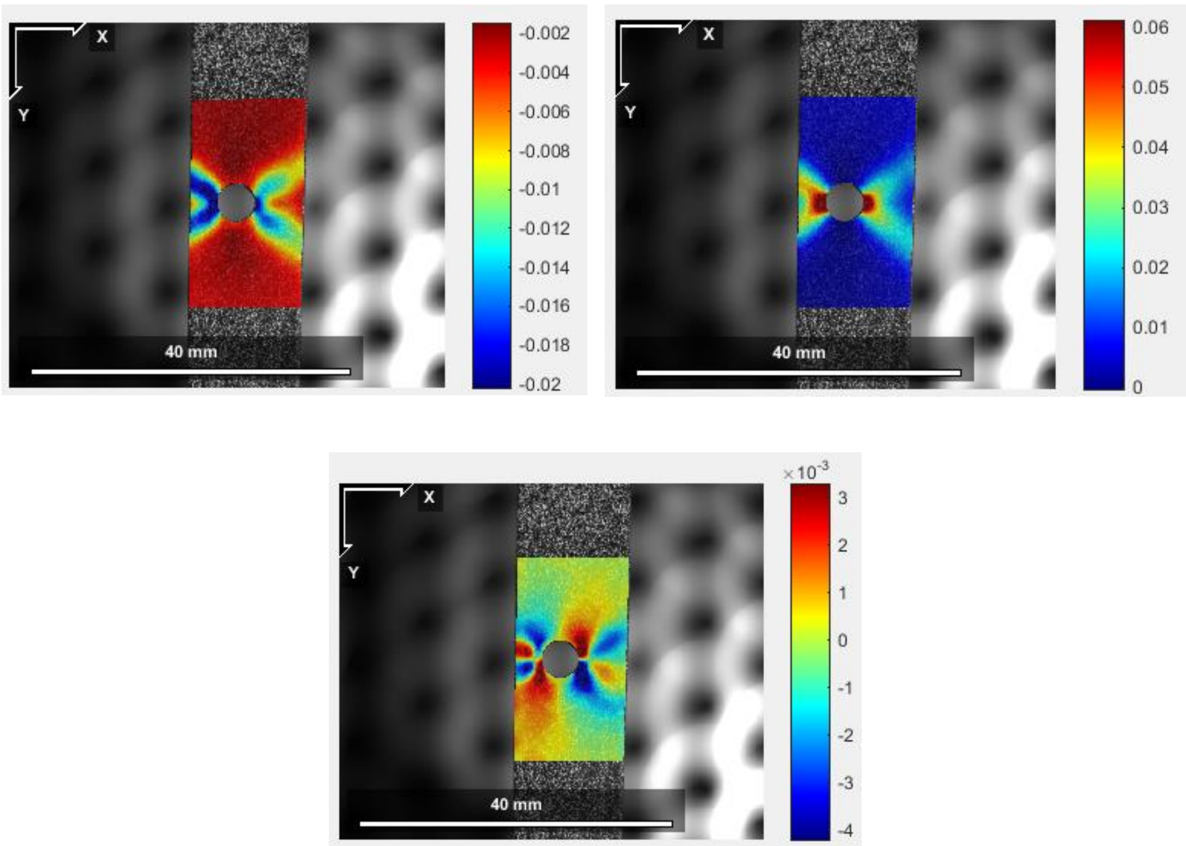


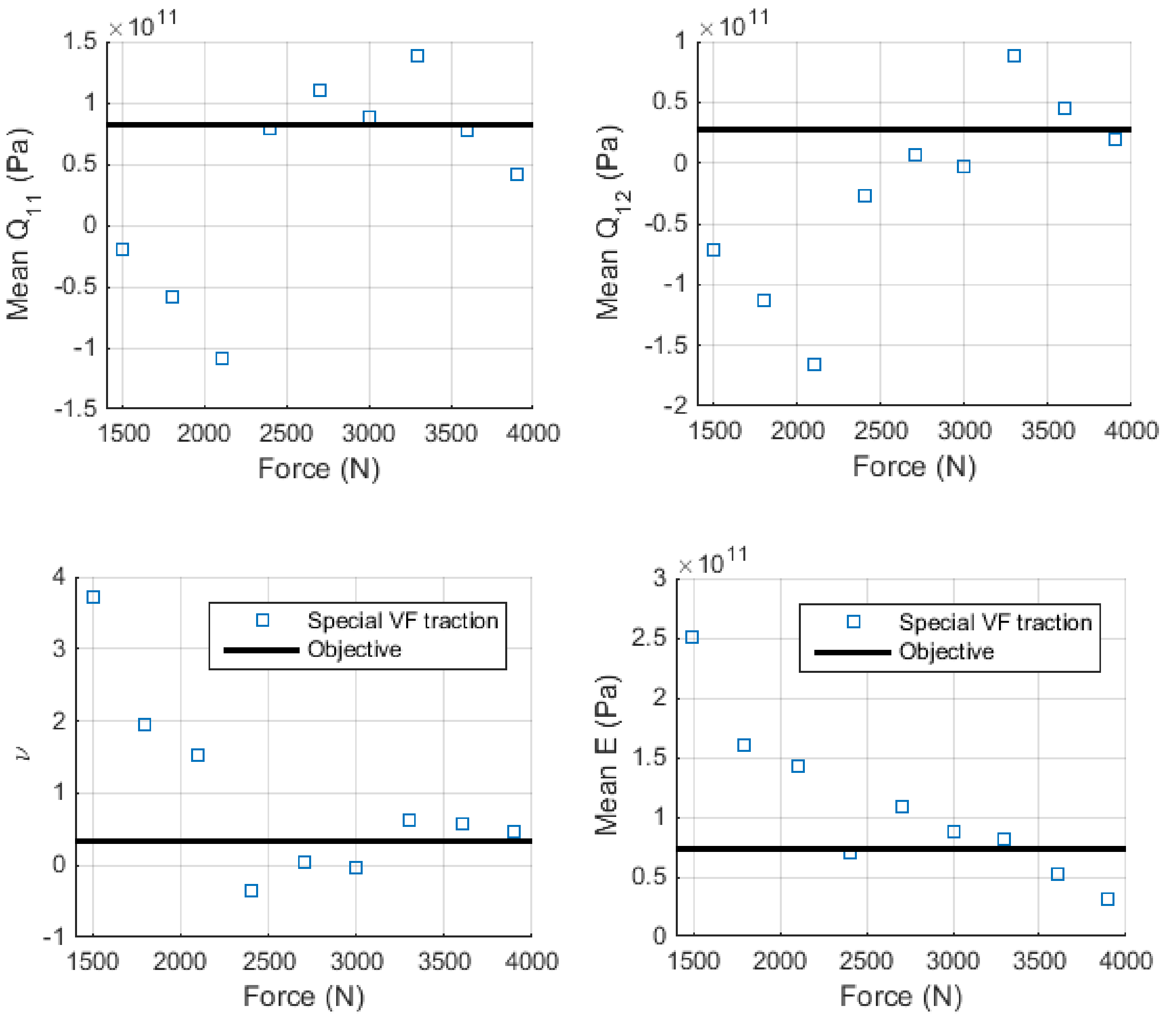
Figure 109. Eps1, Eps2 and Eps6 on the aluminium hole traction specimen (real test, 3900N)

This specimen has almost the same width as the previous ones, so it is clear that the camera has to be placed closer to the specimen in order to increase the accuracy when measuring the strain gradients at the area close to the hole.

It is widely known that the stress concentration factor for a hole in an infinite width plate is equal to 3. In this case this factor could be a little higher because of the specimen width (not infinite) and the fact that the hole is not centered. Nevertheless, the calculation of this factor will not be a relevant issue, so a value of 3 will be considered as an approximation to problem solution.

With this in mind, the specimen will experience plasticity at certain regions if the applied load is above 1500N. This will be a fact which will be taken into account while analyzing the results. The elongation at the tensile yield strength is equal to 0.472%, which means that in the picture presented above there is a big plastic region.

After applying the VFM through Special VF, the following results are obtained:



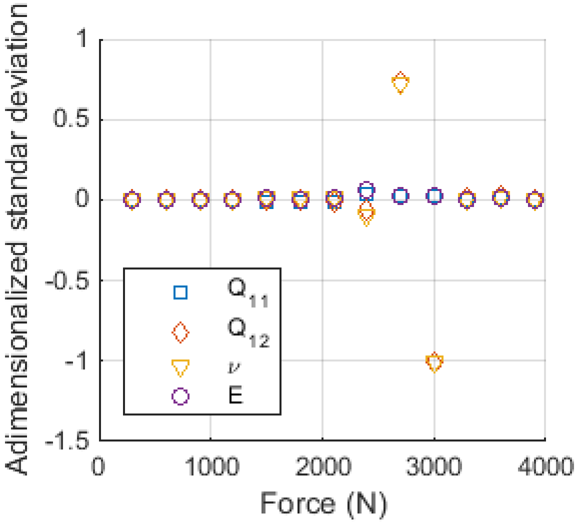


Figure 110. Results obtained on the aluminium real hole traction test

In this specimen it can be seen that there is a big influence of noise, which decreases after reaching the tensile yield strength and then leads to inaccurate results. In addition, strains measured far from the hole are very small, leading to a big region in which the noise is the leading factor.

Therefore, it just can be said that in this specimen, made of aluminum, it is not possible to obtain a reliable data using the Stingray camera. It would be needed another one which provides more spatial resolution.

6.2.3. VFM on Titanium

In this final section, the VFM will be applied to a titanium specimen. To that end, it is going to be used just a homogeneous specimen, since it seems senseless to perform, for example, a hole traction test in a specimen which has big gradients both on displacements and strains, which will not be captured due to the big material stiffness compared to the previous material.

The strain fields obtained in this specimen are the following ones:

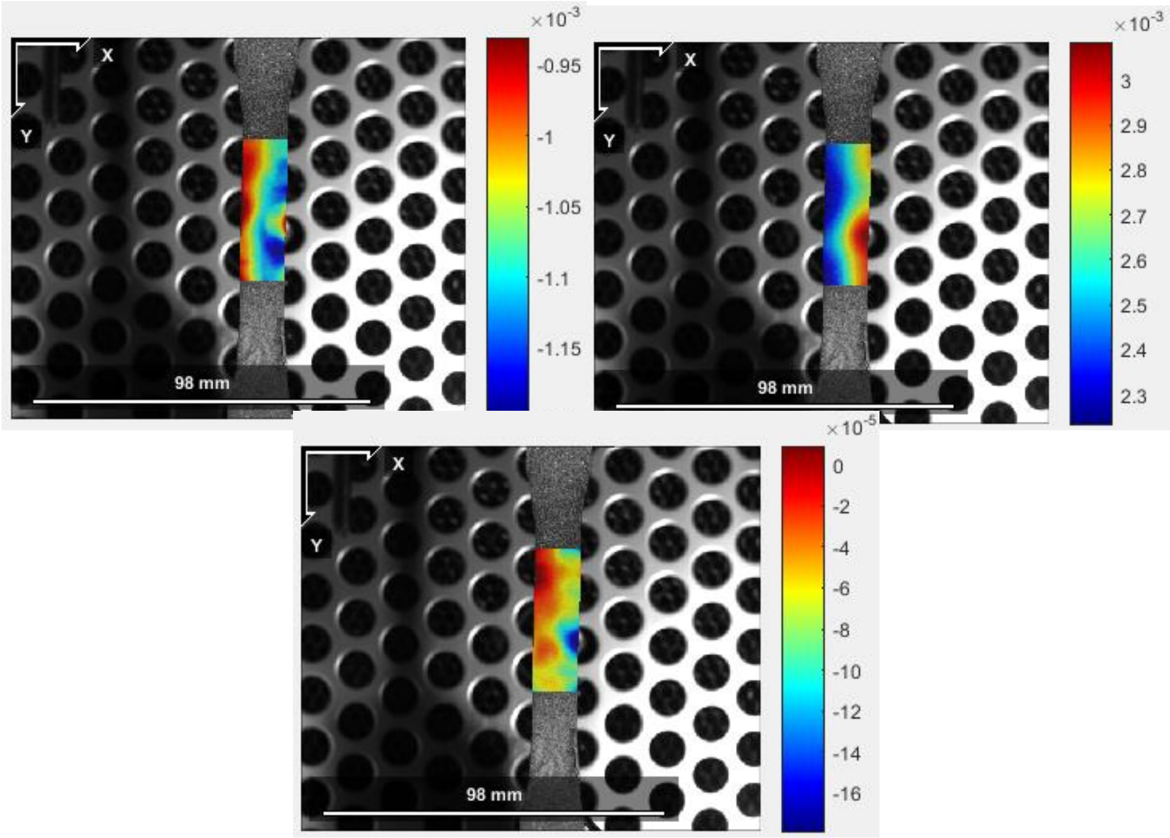
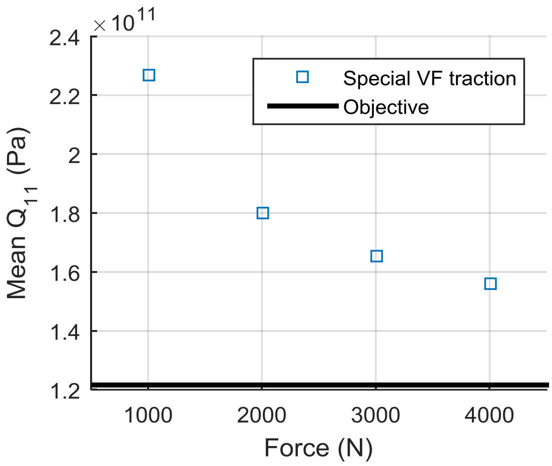
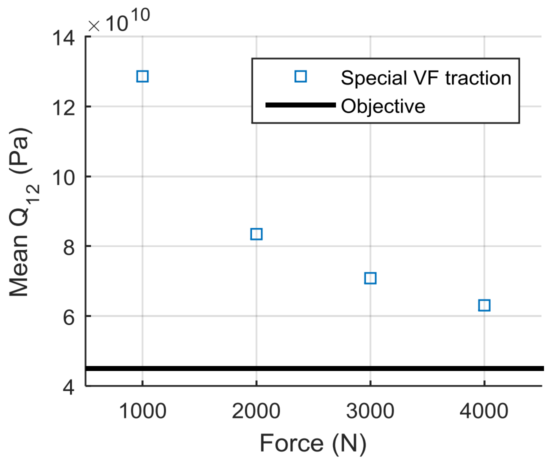


Figure 111. Eps1, Eps2 and Eps6 on the titanium traction specimen (real test, 4000N)

After processing the data by using the VFM, the following results are obtained:





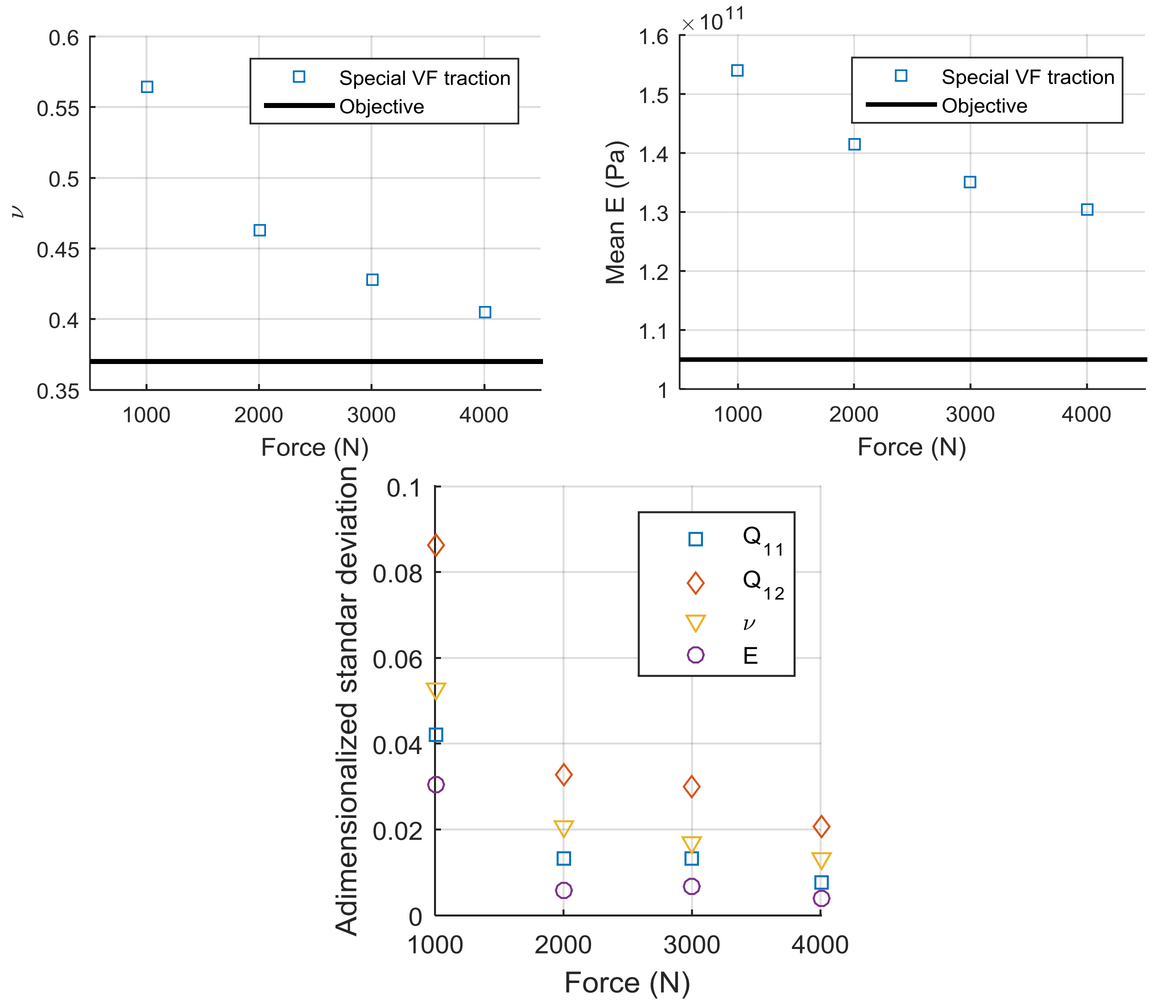


Figure 112. Results obtained on the titanium real traction test

It can be seen that the errors obtained on this specimen are equal to 24.13% in the Young Modulus and 9.44% on the Poisson ratio, which matches with what has been said on the aluminium homogeneous traction test, and has the additional error caused by a stiffer material, what is translated into a lower displacement to mm/pixel ratio.

7. Conclusions

In this section, it is going to be presented the conclusions that have been reached through this project development.

- The first and main conclusion of this project is the fact that the VFM is a powerful tool, which can provide the material stiffness components through the strain fields obtained by applying a certain load on the specimen. This will allow computing these mechanical parameters on final parts or assemblies, providing crucial information about the final behaviour on any kind of part.
- It has been proved the fact that the Optimized Virtual Fields are the less noise-sensitive VFM appliance method, with a better performance than the Special Virtual Fields and the Manually Defined Virtual Fields.
- Even though it will be obtained some dispersion on the results due to the noise influence, if enough tests are performed, it will finally be obtained a mean value close to the real one.
- On the isotropic homogeneous traction test, the lack of gradients on the strain fields leads in the VFM appliance to a matrix close to singular, which cannot be computed due to the computer working precision. In this case, the Special Virtual Fields are the best option for applying the VFM.
- On the orthotropic homogeneous traction test, the lack of gradients lead to a lack of information that gets translated into the fact that it is not possible to obtain all the mechanical parameters of the material in just one test (neither Special nor Optimized Virtual fields show a good performance).
- Regarding to the orthotropic materials, the addition of equations to solve on the equation system is translated into the fact that the Special Virtual Fields require a bigger computer working precision, leading to a bad method behaviour.
- Considering the real tests, as more points are computed the working precision required for the VFM appliance is bigger. This means that in isotropic materials (which are the ones that have been tested) the best option turns to be the Special Virtual Fields.
- On the real tests, the main sources of error are the specimen-camera orientation, the noise introduced by the camera sensor and the loss of information on the DIC analysis.

- In order to reduce the loss of information through the DIC analysis in specimens which show big strain gradients (such as the disc) it is needed to increase the camera resolution, increasing by that way the displacements to mm/pixel ratio. This need is justified since there is a loss of information on the displacements computing and on the strains computing, both of them due to the averaging methods performed on both procedures.

8. Future work

Regarding to the job that has been done, there is a lot of future job that can be done in order to improve the results obtained. Here, these works are going to be listed:

- Study of the images resolution influence on the results obtained, in order to find the minimum displacement to units per pixel ratio regarding to the specimen gradients.
- Study of the computer precision improvement for being able to compute higher degrees virtual fields, as well as the Optimized Virtual Fields in real tests.
- Study of the VFM appliance regarding to techniques that provide directly strain fields or stress fields, such as photoelasticity or thermography.
- VFM appliance on 3D specimens for extracting the constitutive mechanical parameters.
- VFM appliance for extracting the mechanical parameters that govern a bending plate behaviour.
- VFM appliance for identifying a composite material damage model.
- VFM appliance for extracting the stiffness and damping coefficients from vibrating plates.

9. Annexes

9.1. Annex 1: Process followed for obtaining the required information from the software Abaqus CAE.

First of all, inside Abaqus it must be defined the specimen. This step must be done carefully because in it the axes origin will be defined. This origin will be on the 0,0 point on the sketch. The steps followed are shown in the following images.

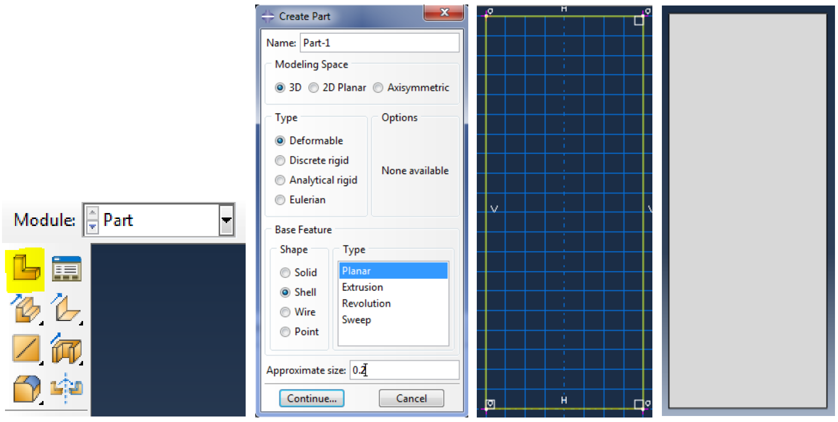


Figure 113. Part definition

Once the part has been defined, it must be set its material and section. To that end, it must be selected:

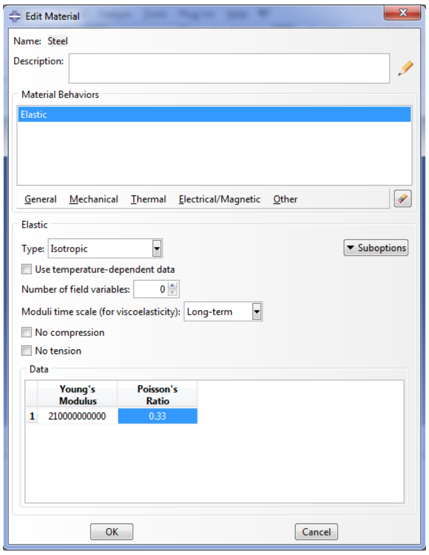


Figure 114. Material properties

Once we have defined the material properties, we must assign it to the required section:

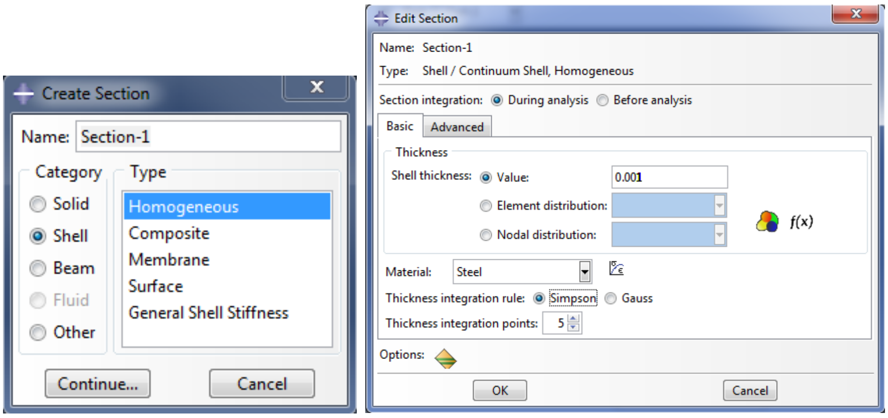


Figure 115. Section properties

Now we must assign the section we have just created to the part we defined previously.

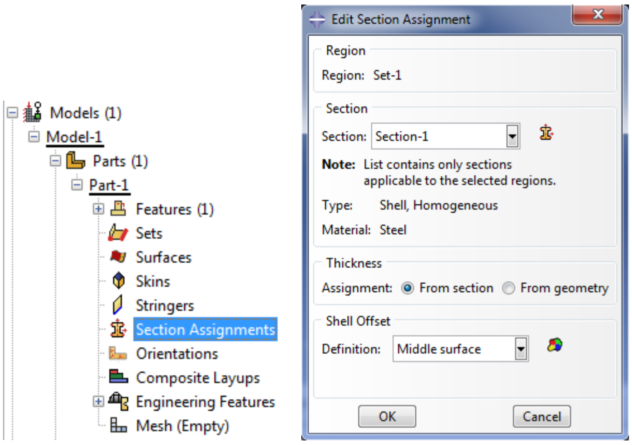


Figure 116. Section assignment

We can now mesh the part. It is important to select triangles as element shape for the proper operation of the further Matlab code.

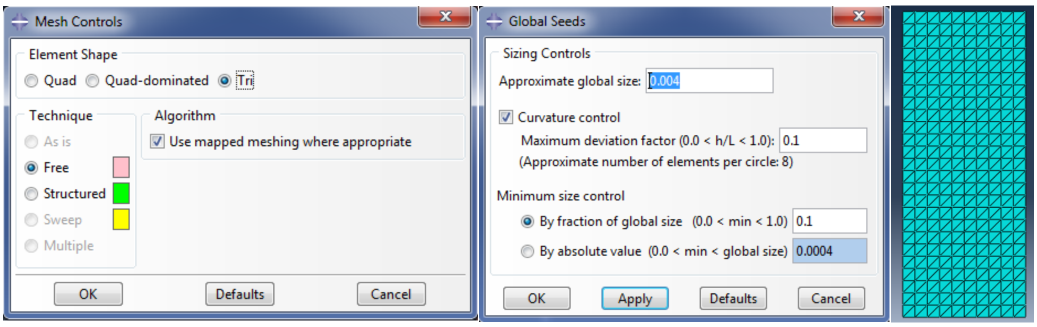


Figure 117. Mesh properties

Now, we must define the instance on the specimen. This will also be a critical step because of the possibility of modifying the axes' zero.

Figure 118. Assembly instances

We finally define the steps, forces, boundary conditions, and both define and submit the job (I do not explain the process because it is not critical on the VFM appliance). The results obtained are shown below:

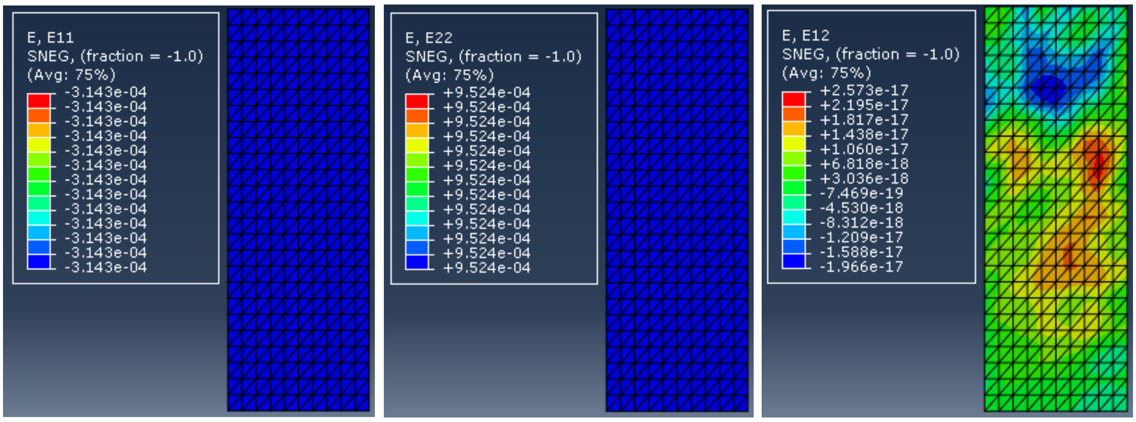


Figure 119. Strain fields on the specimen

The gradients obtained on E12 (which corresponds to ε_6) may be explained by the machine precision, due to the fact that the computer cannot give back a zero after a calculation. On the other hand, it may be due to some rotation on the mesh elements since they are triangular (even though it is less likely because in this case the strain should change depending on the triangle orientation). Anyway, its magnitude order is 14 times lower than the strains on x_1 and x_2, so E12 can be neglected.

Finally, we must obtain the strains information. To that end, we obtain three different data, which we will finally process. First of all, we obtain the stress fields by selecting Report, Field Output and:

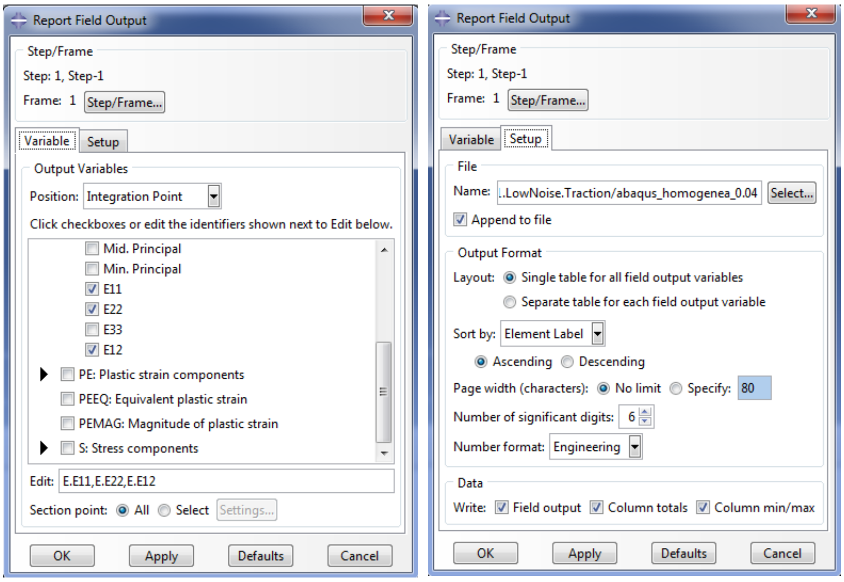


Figure 120. Abaqus report

In this file, which I usually call abaqus.txt, the strain information on each element integration point is saved. Next, we will obtain the element geometrical info by using the .INP file we will find in the temporary folder of the computer. We will divide this file into two different ones. The first one will contain the nodes Cartesian coordinates, and the second one will tell us which points correspond to an element edges.

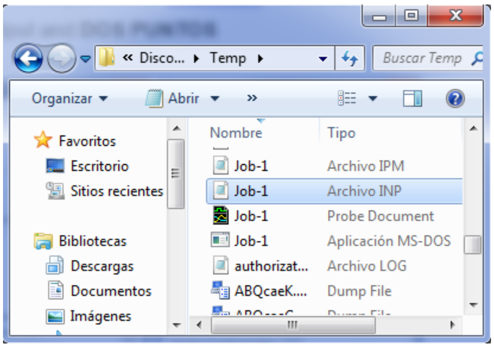


Figure 121. Location of the .inp file

We will save both files as .txt. After having generated the three files, we will run the code DataGenerator into Matlab. This code is shown down below:

```
%Scrip that reads the files obtained in abaqus and exports it into a .mat
%file for its further using into Matlab.


disp('Leyendo los archivos')
[Element,Eps1,Eps2,Eps6] = LeeDeformaciones(['abaqus.txt']);%Reads the
strain value
[Element_code2,Nodo1,Nodo2,Nodo3] = LeeElementos(['elementos.txt']);%Reads
the element code and the nodes code which it is formed by.
[CodigoNodo,X,Y,~] = LeeCoordenadasNodos(['nodos.txt']);%Reads the node
code and its coordinates
```

```
X1=zeros(length(Element),1);
X2=zeros(length(Element),1);
dS=zeros(1,length(Element));

disp('Computing the dS and GC of the elements')
for i=1:length(Element)
    if Element(i)~=Element_code2(i)
        error('Error on the abaqus file. Consider revising it.')
    end
    %Pick up the vertex coordinates for the triangle

xx=[X(Nodo1(i)==CodigoNodo),X(Nodo2(i)==CodigoNodo),X(Nodo3(i)==CodigoNodo)
];

yy=[Y(Nodo1(i)==CodigoNodo),Y(Nodo2(i)==CodigoNodo),Y(Nodo3(i)==CodigoNodo)
];

    %Compute the gravity center of the element

X1(i)=mean([X(Nodo1(i)==CodigoNodo),X(Nodo2(i)==CodigoNodo),X(Nodo3(i)==Cod
igoNodo)]);

X2(i)=mean([Y(Nodo1(i)==CodigoNodo),Y(Nodo2(i)==CodigoNodo),Y(Nodo3(i)==Cod
igoNodo)]);

    %Compute the element surface
    AB=[yy(2)-yy(1),xx(2)-xx(1)];
    nAB=[AB(2),-1*AB(1)];
    AC=[yy(3)-yy(1),xx(3)-xx(1)];
    dS(i)=0.5*nAB*AC';

end

dS=dS';
save('Data.mat','Eps1','Eps2','Eps6','dS','X1','X2')

%% Plotting section, for checking that orientation and values are correct

figure
hold on
disp('Representando...')
xlin = linspace(min(X1),max(X1),1000);
ylin = linspace(min(X2),max(X2),1000);
[X,Y] = meshgrid(xlin,ylin);
f = scatteredInterpolant(X1,X2,Eps1);
Z = f(X,Y);
contourf(X,Y,Z,40) %interpolated
colorbar,colormap('jet')
axis tight; hold on
clearvars -except Eps1 Eps2 Eps6 X1 X2 dS
```

9.2. Annex 2. Matlab code for section 5.2.1

```
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.2.1
% Dependence of the result on the element size
```

```
% DESCRIPTION OF THE INPUT FILE:
% The input file will be just like the input files on the DataGenerator.mat
% code

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% d is the diameter of the disc in mm
% t is the thickness of the disc in mm

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Material
% Eonj is the material's Young's modulus (to be identified: 210 GPa)
% Nuobj is the material's Poisson's ratio (to be identified: 0.33)

% Units are in N and m: stiffnesses will be obtained in Pa


clear all, close all
part1={'homogenea','nohomogenea1','nohomogenea2'};
part2={'001','002','004','006','008'};
F=8000;t=0.001;L=0.1;w=0.04;

Eobj=210e9;Nuobj=0.33;Q11obj=Eobj/(1-Nuobj^2);Q12obj=Nuobj*Q11obj;

for ii=1:3
    for jj=1:5
        %% Files reading
        [Element,Eps1,Eps2,Eps6] =
LeeDeformaciones(['abaqus_',part1{ii},'_',part2{jj},'.txt']);%Lee el valor
de las deformaciones
        [Element_code2,Nodo1,Nodo2,Nodo3] =
LeeElementos(['elementos_',part1{ii},'_',part2{jj},'.txt']);%Lee el codigo
del elemento y el codigo de los nodos que lo forman
        [CodigoNodo,X,Y,~] =
LeeCoordenadasNodos(['nodos_',part1{ii},'_',part2{jj},'.txt']);%Lee el
codigo del nodo y sus coordenadas

        X1=zeros(length(Element),1);
        X2=zeros(length(Element),1);
        dS=zeros(length(Element),1);

        for i=1:length(Element)
            if Element(i)~=Element_code2(i)
                error('Error on the abaqus file. Consider revising it.')
            end
            %Pick up the vertex coordinates for the triangle
```

```
xx=[X(Nodo1(i)==CodigoNodo),X(Nodo2(i)==CodigoNodo),X(Nodo3(i)==CodigoNodo)
];

yy=[Y(Nodo1(i)==CodigoNodo),Y(Nodo2(i)==CodigoNodo),Y(Nodo3(i)==CodigoNodo)
];

            %Compute the gravity center of the element

X1(i)=mean([X(Nodo1(i)==CodigoNodo),X(Nodo2(i)==CodigoNodo),X(Nodo3(i)==Cod
igoNodo)]);

X2(i)=mean([Y(Nodo1(i)==CodigoNodo),Y(Nodo2(i)==CodigoNodo),Y(Nodo3(i)==Cod
igoNodo)]);

            %Compute the element surface
            AB=[yy(2)-yy(1),xx(2)-xx(1)];
            nAB=[AB(2),-1*AB(1)];
            AC=[yy(3)-yy(1),xx(3)-xx(1)];
            dS(i)=0.5*nAB*AC';

        end

        dS=dS';

        %% VFM appliance

        A(1,1)=sum(Eps1.*dS);
        A(1,2)=sum(Eps2.*dS);
        B(1,1)=0;

        A(2,1)=sum(Eps2.*dS);
        A(2,2)=sum(Eps1.*dS);
        B(2,1)=F*L/t;

        Q=A\B;

        Q11result(ii,jj)=Q(1);
        Q12result(ii,jj)=Q(2);

        clear Eps1 Eps2 Eps6 dS A B
    end
end

%% Results processing and plotting

Nuresult=Q12result./Q11result;
Eresult=Q11result.*(1-Nuresult.^2);

figure (1), hold on, grid on, title('Relative error on
Q_1_1'),xlabel('Approximate element size'), ylabel('Error (promille)')
figure (2), hold on, grid on, title('Relative error on
Q_1_2'),xlabel('Approximate element size'), ylabel('Error (promille)')
figure (3), hold on, grid on, title('Relative error on
E'),xlabel('Approximate element size'), ylabel('Error (promille)')
figure (4), hold on, grid on, title('Relative error on
Nu'),xlabel('Approximate element size'), ylabel('Error (promille)')
el=[0.001 0.002 0.004 0.006 0.008];
markers=['vsd'];
```

```
for i=1:3
    figure(1)
    plot(el,abs(Q11result(i,:)-
Q11obj)/Q11obj*1000,'LineStyle','none','Marker',markers(i),'MarkerFaceColor
','k','MarkerEdgeColor','k')
    figure(3)
    plot(el,abs(Eresult(i,:)-
Eobj)/Eobj*1000,'LineStyle','none','Marker',markers(i),'MarkerFaceColor','k
','MarkerEdgeColor','k')
    figure(4)
    plot(el,abs(Nuresult(i,:)-
Nuobj)/Nuobj*1000,'LineStyle','none','Marker',markers(i),'MarkerFaceColor',
'k','MarkerEdgeColor','k')
    figure(2)
    plot(el,abs(Q12result(i,:)-
Q12obj)/Q12obj*1000,'LineStyle','none','Marker',markers(i),'MarkerFaceColor
','k','MarkerEdgeColor','k')
end
figure(1),legend('Homogeneous','Non homogeneous 1','Non homogeneous 2')
figure(2),legend('Homogeneous','Non homogeneous 1','Non homogeneous
2'),ylim([0,1.4e-3])
figure(3),legend('Homogeneous','Non homogeneous 1','Non homogeneous 2')
figure(4),legend('Homogeneous','Non homogeneous 1','Non homogeneous
2'),ylim([0 1.2e-3])
```

9.3. Annex 3. Matlab code for section 5.2.2 (1)

```
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.2.2
% Dependence of the chosen virtual fields

% DESCRIPTION OF THE INPUT FILE:
% The input file will be just like the input files on the DataGenerator.mat
% code

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% d is the diameter of the disc in mm
% t is the thickness of the disc in mm

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Material
% Eonj is the material's Young's modulus (to be identified: 210 GPa)
% Nuobj is the material's Poisson's ratio (to be identified: 0.33)
```

```
% Units are in N and m: stiffnesses will be obtained in Pa

clear all, close all

part2={'001','002','004','006','008'}; %Files data
F=8000;t=0.001;L=0.1;w=0.04; %Specimen data
Eobj=210e9;Nuobj=0.33;Q11obj=Eobj/(1-Nuobj^2);Q12obj=Nuobj*Q11obj;
%Material data


for jj=1:5
    %% Files reading
    disp('Leyendo los archivos')
    [Element,Eps1,Eps2,Eps6] =
LeeDeformaciones(['abaqus_nohomogenea2_',part2{jj},'.txt']);%Lee el valor
de las deformaciones
    [Element_code2,Nodo1,Nodo2,Nodo3] =
LeeElementos(['elementos_nohomogenea2_',part2{jj},'.txt']);%Lee el codigo
del elemento y el codigo de los nodos que lo forman
    [CodigoNodo,X,Y,~] =
LeeCoordenadasNodos(['nodos_nohomogenea2_',part2{jj},'.txt']);%Lee el
codigo del nodo y sus coordenadas

    X1=zeros(length(Element),1);
    X2=zeros(length(Element),1);
    dS=zeros(length(Element),1);

    disp('Calculando los dS y CG de los elementos')
    for i=1:length(Element)
        if Element(i)~=Element_code2(i)
            error('El vector Element_code2 esta desordenado')
        end
        %Cojo las coordenadas de los vertices del nodo triangular

xx=[X(Nodo1(i)==CodigoNodo),X(Nodo2(i)==CodigoNodo),X(Nodo3(i)==CodigoNodo)
];

yy=[Y(Nodo1(i)==CodigoNodo),Y(Nodo2(i)==CodigoNodo),Y(Nodo3(i)==CodigoNodo)
];

        %Calculo el centro de gravedad del elemento

X1(i)=mean([X(Nodo1(i)==CodigoNodo),X(Nodo2(i)==CodigoNodo),X(Nodo3(i)==Cod
igoNodo)]);

X2(i)=mean([Y(Nodo1(i)==CodigoNodo),Y(Nodo2(i)==CodigoNodo),Y(Nodo3(i)==Cod
igoNodo)]);

        %Calculo la superficie del elemento
        AB=[yy(2)-yy(1),xx(2)-xx(1)];
        nAB=[AB(2),-1*AB(1)];
        AC=[yy(3)-yy(1),xx(3)-xx(1)];
        dS(i)=0.5*nAB*AC';

    end

    %% VFM appliance

    %First set of virtual fields
```

```
    A(1,1)=sum(Eps1.*dS);
    A(1,2)=sum(Eps2.*dS);
    B(1,1)=0;

    A(2,1)=sum(Eps2.*dS);
    A(2,2)=sum(Eps1.*dS);
    B(2,1)=F*L/t;

    Q=A\B;

    Q111(jj)=Q(1);
    Q121(jj)=Q(2);

    %Second set of virtual fields

    A(1,1)=sum((Eps2.*(X1.^2)+Eps6.*X1.*(X2)).*dS);
    A(1,2)=sum((Eps1.*(X1.^2)-Eps6.*X1.*(X2)).*dS);
    B(1,1)=F*L*w^3/12/t/w;

    A(2,1)=sum(3*Eps1.*(X1.^2).*dS);
    A(2,2)=sum(3*Eps2.*(X1.^2).*dS);
    B(2,1)=0;

    Q=A\B;

    Q112(jj)=Q(1);
    Q122(jj)=Q(2);

    %Third set of virtual fields

    A(1,1)=sum((Eps2.*cos(pi*X1/w)-0.5*Eps6.*X2.*sin(pi*X1/w)*pi/w).*dS);
    A(1,2)=sum((Eps1.*cos(pi*X1/w)+0.5*Eps6.*X2.*sin(pi*X1/w)*pi/w).*dS);
    B(1,1)=2*F*L*w/pi/t/w;

    A(2,1)=sum((Eps1.*cos(2*pi*(X1+w/2)/w)*2*pi/w).*dS);
    A(2,2)=sum((Eps2.*cos(2*pi*(X1+w/2)/w)*2*pi/w).*dS);
    B(2,1)=0;

    Q=A\B;

    Q113(jj)=Q(1);
    Q123(jj)=Q(2);

    clear Eps1 Eps2 Eps6 dS A B
end

%% Results processing and plotting

Q11result=[Q111;Q112;Q113];
Q12result=[Q121;Q122;Q123];
Nuresult=Q12result./Q11result;
Eresult=Q11result.*(1-Nuresult.^2);
figure (1), hold on, grid on, title('Relative error on
Q_1_1'),xlabel('Approximate element size'), ylabel('Error (promille)')
figure (2), hold on, grid on, title('Relative error on
Q_1_2'),xlabel('Approximate element size'), ylabel('Error (promille)')
```

```
figure (3), hold on, grid on, title('Relative error on
E'),xlabel('Approximate element size'), ylabel('Error (promille)')
figure (4), hold on, grid on, title('Relative error on
Nu'),xlabel('Approximate element size'), ylabel('Error (promille)')
el=[0.001 0.002 0.004 0.006 0.008];
markers=['vsd'];
for i=1:3
    figure(1)
    plot(el,(abs(Q11result(i,:)-
Q11obj)/Q11obj*1000),'LineStyle','none','Marker',markers(i),'MarkerFaceColo
r','k','MarkerEdgeColor','k')
    figure(3)
    plot(el,(abs(Eresult(i,:)-
Eobj)/Eobj*1000),'LineStyle','none','Marker',markers(i),'MarkerFaceColor','
k','MarkerEdgeColor','k')
    figure(4)
    plot(el,(abs(Nuresult(i,:)-
Nuobj)/Nuobj*1000),'LineStyle','none','Marker',markers(i),'MarkerFaceColor'
,'k','MarkerEdgeColor','k')
    figure(2)
    plot(el,(abs(Q12result(i,:)-
Q12obj)/Q12obj*1000),'LineStyle','none','Marker',markers(i),'MarkerFaceColo
r','k','MarkerEdgeColor','k')
end
figure(1),legend('1^s^t set','2^n^d set','3^r^d set')
figure(2),legend('1^s^t set','2^n^d set','3^r^d set')
figure(3),legend('1^s^t set','2^n^d set','3^r^d set')
figure(4),legend('1^s^t set','2^n^d set','3^r^d set')
```

9.4. Annex 4: Matlab code for section 5.2.2 (2)

```
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.2.2
% Dependence of the result on the chosen virtual fields including noise

% DESCRIPTION OF THE INPUT FILE:
% The input file will be DataHomogenea.mat, DataNoHomogenea1.mat and
% DataNoHomogenea2.mat

% The file will contain the following info:

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% d is the diameter of the disc in mm
% t is the thickness of the disc in mm

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
```

```
% shear strain)

% Material
% Eobj is the material's Young's modulus (to be identified: 210 GPa)
% Nuobj is the material's Poisson's ratio (to be identified: 0.33)

% Units are in N and m: stiffnesses will be obtained in Pa

clear all,close all

load DataHomogenea.mat %Load the required file

n_amp=20; %Number of amplitudes studied
n_tests=10000; %Number of tests performed on each noise amplitude

gamma=linspace(0,1e-4,n_amp); %Noise amplitudes

%% Calculations

n_points=length(Eps11);
Eps11=Eps1;Eps22=Eps2;Eps66=Eps6;
for jj=1:length(gamma)
    for ii=1:n_tests
        %Noise adittion on each strain
        Eps1=Eps11+randn(n_points,1)*gamma(jj);
        Eps2=Eps22+randn(n_points,1)*gamma(jj);
        Eps6=Eps66+randn(n_points,1)*gamma(jj);

        % VFM appliance

        %First set of virtual fields

        A(1,1)=sum(Eps1.*dS);
        A(1,2)=sum(Eps2.*dS);
        B(1,1)=0;

        A(2,1)=sum(Eps2.*dS);
        A(2,2)=sum(Eps1.*dS);
        B(2,1)=F*L/t;

        Q=A\B;

        Q111(jj,ii)=Q(1);
        Q121(jj,ii)=Q(2);

        %Second set of virtual fields

        A(1,1)=sum((Eps2.*(X1.^2)+Eps6.*X1.*(X2)).*dS);
        A(1,2)=sum((Eps1.*(X1.^2)-Eps6.*X1.*(X2)).*dS);
        B(1,1)=F*L*w^3/12/t/w;

        A(2,1)=sum(3*Eps1.*(X1.^2).*dS);
        A(2,2)=sum(3*Eps2.*(X1.^2).*dS);
        B(2,1)=0;

        Q=A\B;

        Q112(jj,ii)=Q(1);
```

```
        Q122(jj,ii)=Q(2);

        %Third set of virtual fields

        A(1,1)=sum((Eps2.*cos(pi*X1/w)-
0.5*Eps6.*X2.*sin(pi*X1/w)*pi/w).*dS);

A(1,2)=sum((Eps1.*cos(pi*X1/w)+0.5*Eps6.*X2.*sin(pi*X1/w)*pi/w).*dS);
        B(1,1)=2*F*L*w/pi/t/w;

        A(2,1)=sum((Eps1.*cos(2*pi*(X1+w/2)/w)*2*pi/w).*dS);
        A(2,2)=sum((Eps2.*cos(2*pi*(X1+w/2)/w)*2*pi/w).*dS);
        B(2,1)=0;

        Q=A\B;

        Q113(jj,ii)=Q(1);
        Q123(jj,ii)=Q(2);

        clear Eps1 Eps2 Eps6 A B
    end
end

%% Results processing and plotting

figure, hold on, grid on
xlabel('Noise amplitude')
ylabel('Standard deviation / Q_1_1 objective')
title('Adimensionalized standar deviation on Q_1_1')
plot(gamma,std(Q111')/Q11obj,'LineStyle','none','Marker','v','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q112')/Q11obj,'LineStyle','none','Marker','s','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q113')/Q11obj,'LineStyle','none','Marker','d','MarkerFaceCol
or','k','MarkerEdgeColor','k')
legend('1^s^t set','2^n^d set','3^r^d set','Location','northwest')

figure, hold on, grid on
xlabel('Noise amplitude')
ylabel('Standard deviation / Q_1_2 objective')
title('Adimensionalized standar deviation on Q_1_2')
plot(gamma,std(Q121')/Q12obj,'LineStyle','none','Marker','v','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q122')/Q12obj,'LineStyle','none','Marker','s','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q123')/Q12obj,'LineStyle','none','Marker','d','MarkerFaceCol
or','k','MarkerEdgeColor','k')
legend('1^s^t set','2^n^d set','3^r^d set','Location','northwest')

figure, hold on, grid on
xlabel('Noise amplitude')
ylabel('Relative error')
title('Relative error on Q_1_1')
plot(gamma,abs(mean(Q111')-
Q11obj)/Q11obj,'LineStyle','none','Marker','v','MarkerFaceColor','k','Marke
rEdgeColor','k')
plot(gamma,abs(mean(Q112')-
Q11obj)/Q11obj,'LineStyle','none','Marker','s','MarkerFaceColor','k','Marke
rEdgeColor','k')
```

```
plot(gamma,abs(mean(Q113')-
Q11obj)/Q11obj,'LineStyle','none','Marker','d','MarkerFaceColor','k','Marke
rEdgeColor','k')
legend('1^s^t set','2^n^d set','3^r^d set','Location','northwest')

figure, hold on, grid on
xlabel('Noise amplitude')
ylabel('Relative error')
title('Relative error on Q_1_2')
plot(gamma,abs(mean(Q121')-
Q12obj)/Q12obj,'LineStyle','none','Marker','v','MarkerFaceColor','k','Marke
rEdgeColor','k')
plot(gamma,abs(mean(Q122')-
Q12obj)/Q12obj,'LineStyle','none','Marker','s','MarkerFaceColor','k','Marke
rEdgeColor','k')
plot(gamma,abs(mean(Q123')-
Q12obj)/Q12obj,'LineStyle','none','Marker','d','MarkerFaceColor','k','Marke
rEdgeColor','k')
legend('1^s^t set','2^n^d set','3^r^d set','Location','northwest')
```

9.5. Annex 5: Matlab code for section 5.2.3

```
function
[Q11,Q12,eta11,eta12]=SpecialVirtualFieldsFunction(Eps1,Eps2,Eps6,dS,X1,X2,
F,L,w,t,m,n)
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.2.3
% Special virtual fields function

% DESCRIPTION OF THE INPUTS:
% The inputs will be the following ones:

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% L is the length of the specimen (in e2)
% w is the width of the specimen (in e1)
% t is the thickness of the disc in mm
% m is the degree of the polynomial on e1
% n is the degree of the polynomial on e2

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Material
% Eobj is the material's Young's modulus (to be identified: 210 GPa)
% Nuobj is the material's Poisson's ratio (to be identified: 0.33)

% Units are in N and m: stiffnesses will be obtained in Pa
```

```
%% VFM preparing:

n_dof=(m+1)*(n+1); %Number of degrees of freedom
n_points=length(X1); %Number of points on the specimen

for dd=2:n_dof
    j=floor((dd-1)/(m+1)); %browses through all possible couples of (i,j)
values
    i=(dd-1)-j*(m+1);

    ilog(dd)=i; % For information: all incremental values of i
    jlog(dd)=j; % For information: all incremental values of j

    dx1_x=X2.*(X2-L)/(L^2)*i.*((X1/w).^(i-1))*1/w.*(X2/L).^(j); % Monomials
derivative with respect to X1
    dx1_y=(2*X2-L)/(L^2).*(X1/w).^(i).*(X2/L).^(j)+X2.*(X2-
L)/(L^2)*j.*((X1/w).^(i))*1/L.*(X2/L).^(j-1);

    dx2_y = 1/L*(X1/w).^(i).*(X2/L).^(j)+X2/L.*(X1/w).^(i).*(X2/L).^(j-
1)*j/L; % Monomials derivative with respect to X2
    dx2_x = X2/L*i.*(X1/w).^(i-1)/w.*(X2/L).^(j);

    b1(:,dd)=dx1_x; % Contains information on epsilon1*, only depends on
the Aij's
    b2(:,dd)=zeros(n_points,1); % information on epsilon2* does not depend
on the Aij's
    b6(:,dd)=dx1_y; % Contains information on epsilon6*, depends on the
Bij's
    b1(:,dd+n_dof)=zeros(n_points,1); % information on epsilon1* does not
depend on the Bij's
    b2(:,dd+n_dof)=dx2_y; % Contains information on epsilon2*, only depends
on the Bij's
    b6(:,dd+n_dof)=dx2_x; % Contains information on epsilon6*, depends on
the Aij's
end

%We calculate now the vectors corresponding to integrals pertaining to Q11
(first) and Q12 (second)

Eps11=repmat(Eps1,1,size(b1,2));
Eps22=repmat(Eps2,1,size(b1,2));
Eps66=repmat(Eps6,1,size(b1,2));
dSS=repmat(dS,1,size(b1,2));

Integral11=sum(((Eps11.*b1)+(Eps22.*b2)+(0.5*Eps66.*b6)).*dSS);
Integral12=sum(((Eps22.*b1)+(Eps11.*b2)-(0.5*Eps66.*b6)).*dSS);

E11=zeros(n_dof*2,1);E11(1)=1;
E12=zeros(n_dof*2,1);E12(2)=1;

H11=(b1.*dSS)'*(b1.*dSS);
H22=(b2.*dSS)'*(b2.*dSS);
H12=(b1.*dSS)'*(b2.*dSS);
H66=(b6.*dSS)'*(b6.*dSS);

iter=0;
tic
```

```
posiciones=2:3; %For reducing computational time
%posiciones=1:(m+2); %Original positions vector
posiciones(end)=posiciones(end)-1;

%% Special VFM calculation

continuar=1;
while continuar
    %% Coefficients which will be non equal to zero

    posiciones(end)=posiciones(end)+1;
    if posiciones(end)>n_dof*2
        salir=0;
        ident=length(posiciones);
        while salir==0
            if ident-1==0
                salir=1;
                continuar=0;
            end
            for j=ident-1:length(posiciones)
                if j==ident-1
                    posiciones(j)=posiciones(j)+1;
                else
                    posiciones(j)=posiciones(j-1)+1;
                end
            end
            if posiciones(end)>n_dof*2
                ident=ident-1;
                if ident-1==0
                    salir=1;
                    continuar=0;
                end
            else
                salir=1;
            end
        end

    end
    if posiciones(1)>2,continuar=0;end %For reducing computational time

    %% Virtual fields calculation on the desiredpositions

    D=zeros(n_dof*2);
    D(1,:)=Integral11;
    D(2,:)=Integral12;
    for j=1:m
        vector=zeros(1,n_dof*2);
        vector(n_dof+j+1:m+1:n_dof*2)=1;
        D(j+2,:)=vector;
    end
    xd=ones(1,n_dof*2);
    xd(posiciones)=0;
    xx=find(xd);

    for p=(2+m+1):n_dof*2
        D(p,xx(p-2-m))=1;
    end

    if round(det(D),16)~=0
        Y11=D\E11;
```

```
        Y12=D\E12;
        if sum(Y11(n_dof+1:m+1:n_dof*2))~=0 &&
sum(Y12(n_dof+1:m+1:n_dof*2))~=0

            Q11=sum(Y11(n_dof+1:m+1:n_dof*2))*F/t;
            Q12=sum(Y12(n_dof+1:m+1:n_dof*2))*F/t;
            Q66=(Q11-Q12)/2;
            Nu=Q12/Q11;
            E=Q11*(1-Nu^2);


eta11=sqrt(Y11'*(2*((Q11^2+Q12^2)*H11+(Q11^2+Q12^2)*H22+Q66^2*H66+4*Q11*Q12
*H12))*Y11);

eta12=sqrt(Y12'*(2*((Q11^2+Q12^2)*H11+(Q11^2+Q12^2)*H22+Q66^2*H66+4*Q11*Q12
*H12))*Y12);


            if E~=0

                iter=iter+1;
                mat(:,iter)=[E Nu Q11 Q12 eta11 eta12];
                Y111(:,iter)=Y11;
                Y122(:,iter)=Y12;
                letras(:,iter)=posiciones;
            end
        end
    end
end

%% Stiffness components choosing

[eta11,pos11]=min(mat(5,:));
Q11=mat(3,pos11);

[eta12,pos12]=min(mat(6,:));
Q12=mat(4,pos12);
```

9.6. Annex 6: Matlab code for section 5.2.4 (1)

```
function
[Qiso1,Qiso2,etaiso1,etaiso2]=OptimizedVirtualFieldsFunction(Eps1,Eps2,Eps6
,dS,X1,X2,F,L,w,t,m,n)

%% Original developers:
% The Virtual Fields Methods: Extracting constitutive mechanical parameters
% from full-field deformation measurements, based on experimental data.
% F. Pierron, M. Grédiac
% Springer
% April 2010
% Based on the paper "Sensitivity to the virtual fields method to noisy
% data", Avril, Grediac, Pierron, - Comp. Mech. (2004)

%% Modified by:
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.2.4
```

```
% Optimized virtual fields function

% DESCRIPTION OF THE INPUTS:
% The inputs will be the following ones:

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% L is the length of the specimen (in e2)
% w is the width of the specimen (in e1)
% t is the thickness of the disc in mm
% m is the degree of the polynomial on e1
% n is the degree of the polynomial on e2

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Material
% Eobj is the material's Young's modulus (to be identified: 210 GPa)
% Nuobj is the material's Poisson's ratio (to be identified: 0.33)

% Units are in N and m: stiffnesses will be obtained in Pa

% define the size of the arrays
n_dof=(m+1)*(n+1); %number of degrees of freedom for each component of the
virtual displacements

n_points=length(Eps1); % Total number of measurement points

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Building up the virtual fields
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
b1=zeros(n_points,n_dof*2); % Initialization to zero
b2=zeros(n_points,n_dof*2);
b6=zeros(n_points,n_dof*2);

for dd=2:n_dof
    j=floor((dd-1)/(m+1)); %browses through all possible couples of (i,j)
values
    i=(dd-1)-j*(m+1);

    ilog(dd)=i; % For information: all incremental values of i
    jlog(dd)=j; % For information: all incremental values of j

    if i==0 % if i=0, then dx2 is zero for all points
        dx1=zeros(n_points,1);
    else
        dx1=i*1/w*(X1/w).^(i-1).*(X2/L).^(j); % Monomials derivative with
respect to X1
    end
    if j==0 % if j=0, then dx2 is zero for all points
```

```
        dx2=zeros(n_points,1);
    else
        dx2=j*1/L*(X1/w).^(i).*(X2/L).^(j-1); % Monomials derivative with
respect to X2
    end

    b1(:,dd)=dx1; % Contains information on epsilon1*, only depends on the
Aij's
    b2(:,dd)=zeros(n_points,1); % information on epsilon2* does not depend
on the Aij's
    b6(:,dd)=dx2; % Contains information on epsilon6*, depends on the Bij's
    b1(:,dd+n_dof)=zeros(n_points,1); % information on epsilon1* does not
depend on the Bij's
    b2(:,dd+n_dof)=dx2; % Contains information on epsilon2*, only depends
on the Bij's
    b6(:,dd+n_dof)=dx1; % Contains information on epsilon6*, depends on the
Aij's

    u1(:,dd)=(X1/w).^(i).*(X2/L).^(j); % u1 only depends on the Aij's
    u1(:,dd+n_dof)=zeros(n_points,1);
    u2(:,dd)=zeros(n_points,1); % u2 only depends on the Aij's
    u2(:,dd+n_dof)=(X1/w).^(i).*(X2/L).^(j);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Construction of the quantities based on the
% virtual fields in the final optimization matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The Bij's will be used to build up the conditions of speciality
% of the virtual fields (in submatrix A of the final system)

B11=zeros(1,n_dof*2); % Initialization
B22=zeros(1,n_dof*2);
B12=zeros(1,n_dof*2);
B66=zeros(1,n_dof*2);

for kk=1:n_dof*2
    B11(kk)=sum(b1(:,kk).*Eps1.*dS); % Quantities for the speciality
conditions
    B22(kk)=sum(b2(:,kk).*Eps2.*dS);
    B12(kk)=sum((b2(:,kk).*Eps1+b1(:,kk).*Eps2).*dS);
    B66(kk)=sum(b6(:,kk).*Eps6.*dS);
end

% The Hij's will be used to build up the Hessian matrix
% H of the final system

dSS=repmat(dS,1,size(b1,2));

H11=(b1.*dSS)'*(b1.*dSS);
H22=(b2.*dSS)'*(b2.*dSS);
H12=(b1.*dSS)'*(b2.*dSS);
H66=(b6.*dSS)'*(b6.*dSS);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Virtual boundary conditions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Aconst=zeros(3*(n+1)+n,n_dof*2); % There are 3*(n+1)+n boundary conditions
here
```

```
rowA=1; % Conditions only involve n since all conditions have fixed x1
for i=0:n % U1*(x2=0)=0. (n+1) conditions
    Aconst(rowA,i+1)=1;
    rowA=rowA+1;
end
for i=0:n % U2*(x2=0)=0. (n+1) conditions
    Aconst(rowA,(m+1)*(n+1)+i+1)=1;
    rowA=rowA+1;
end
for i=0:n % U1*(x2=L)=0. (n+1) conditions
    Aconst(rowA,i+1:(m+1):n_dof)=1;
    rowA=rowA+1;
end
for i=1:n % U2*(x2=L) does not depend on x2. n conditions
    Aconst(rowA,(m+1)*(n+1)+i+1:(m+1):n_dof*2)=1;
    rowA=rowA+1;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Constitution of the Z vector containing zeros
% except for the conditions of speciality
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Za is the Z vector used for finding the virtual field to identify Q11
% Zb is the Z vector used for finding the virtual field to identify Q22
% Zc is the Z vector used for finding the virtual field to identify Q12
% Zd is the Z vector used for finding the virtual field to identify Q66
Za=zeros(1,n_dof*2+size(Aconst,1)+1); % Za is the Z vector used for finding
the virtual field to identify Q11
Zb=Za;Zc=Za;Zd=Za;

Za(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[1 0 0 0]; % special
virtual field condition for Q11
Zb(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[0 1 0 0]; % special
virtual field condition for Q22
Zc(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[0 0 1 0]; % special
virtual field condition for Q12
Zd(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[0 0 0 1]; % special
virtual field condition for Q66

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Building up the A matrix containing all constraints:
% boundary conditions and speciality and the
% B matrix containing zeros
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A=[Aconst;B11;B22;B12;B66]; % Constraints for the boundary conditions
(Aconst) and the speciality conditions (Bij)
B=zeros(size(A,1));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Solving the system (optimization)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Q=[1 1 1 1]; % initializing value for Q because H depends on the Qij's

n_iter=20; % maximum number of iteration
delta_lim=0.0001; % tolerance

delta=10; % Starting with a tolerance larger than delta_lim
i=1;
Qold=Q; % Storing the current Q into Qold for checking convergence
```

```
while i<n_iter && delta>delta_lim

    % Hessian matrix
    H =2*((Q(1)^2+Q(3)^2)*H11+(Q(2)^2+Q(3)^2)*H22+...
        Q(4)^2*H66+2*(Q(1)+Q(2))*Q(3)*H12);

    %NOTE: to avoid the numerical "Warning: Matrix is close to singular or
    %       badly scaled" matrix Opt can be scaled with the parameter corr.
    %       It does not change the results of the optimization.
    %       To avoid using, put corr=1;

    corr=max(max(A))/max(max(H)); % Normalization coefficient
    OptM=[H*corr,A'*corr;A,B]; % Matrix for the optimization of the virtual
fields

    Ya=OptM\Za'; %Vector containing the polynomial coefficients for Q11 and
the Lagrange multipliers
    Yb=OptM\Zb'; %Vector containing the polynomial coefficients for Q22 and
the Lagrange multipliers
    Yc=OptM\Zc'; %Vector containing the polynomial coefficients for Q12 and
the Lagrange multipliers
    Yd=OptM\Zd'; %Vector containing the polynomial coefficients for Q66 and
the Lagrange multipliers

    Ya(n_dof*2+1:size(Ya))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest
    Yb(n_dof*2+1:size(Yb))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest
    Yc(n_dof*2+1:size(Yc))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest
    Yd(n_dof*2+1:size(Yd))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest

    Q(1)=(sum(Ya(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q11 from the
first optimized virtual field
    Q(2)=(sum(Yb(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q22 from the
second optimized virtual field
    Q(3)=(sum(Yc(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q12 from the
third optimized virtual field
    Q(4)=(sum(Yd(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q66 from the
fourth optimized virtual field

    delta=sum((Qold-Q).^2./Q.^2); % Difference between the current and
previous values of the identified parameters
    Qold=Q; % The current parameters are stored as the previous one before
the next iteration
    i=i+1; % Next step
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Final results
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Final Hessian matrix for computing the eta parameters
H =2*((Q(1)^2+Q(3)^2)*H11+(Q(2)^2+Q(3)^2)*H22+...
    Q(4)^2*H66+2*(Q(1)+Q(2))*Q(3)*H12);

eta(1) = sqrt(0.5*Ya'*H*Ya); % Sensitivity to noise parameter for Q11
eta(2) = sqrt(0.5*Yb'*H*Yb); % Sensitivity to noise parameter for Q22
```

```
eta(3) = sqrt(0.5*Yc'*H*Yc); % Sensitivity to noise parameter for Q12
eta(4) = sqrt(0.5*Yd'*H*Yd); % Sensitivity to noise parameter for Q66

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Elements choosing
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if eta(1)>eta(2)
    Qiso1=Q(2);
    etaiso1=eta(2);
else
    Qiso1=Q(1);
    etaiso1=eta(1);
end

if (etaiso1+eta(4))/2>eta(3)
    Qiso2=Q(3);
    etaiso2=eta(3);
else
    Qiso2=(Qiso1-Q(4))/2;
    etaiso2=(etaiso1+eta(4))/2;
end
```

9.7. Annex 7: Matlab code for section 5.2.4 (2)

```
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.2.4
% Noise dependence of the different methods for a traction test

% DESCRIPTION OF THE INPUT FILE:
% The input file will be DataHomogenea.mat, DataNoHomogenea1.mat and
% DataNoHomogenea2.mat

% The file will contain the following info:

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% d is the diameter of the disc in mm
% t is the thickness of the disc in mm

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Material
% Eobj is the material's Young's modulus (to be identified: 210 GPa)
% Nuobj is the material's Poisson's ratio (to be identified: 0.33)
```

```
% Units are in N and m: stiffnesses will be obtained in Pa

clear all,close all

load DataHomogenea.mat %Load the data of the specimen we want to study
Eps11=Eps1;Eps22=Eps2;Eps66=Eps6; %Save the original data

Eobj=210e9;Nuobj=0.33; %Material data
Q11obj=2.1e11/(1-0.33^2);Q12obj=Q11obj*0.33;
disp('Estudiando la influencia del ruido...')
n_points=length(Eps11);

%% Calculations

n_amp=15; %length of the noise amplitudes vector
n_tests=1000;
gamma=linspace(0,1e-4,n_amp);

for jj=1:length(gamma)
    for ii=1:n_tests
        Eps1=Eps11+randn(n_points,1)*gamma(jj);
        Eps2=Eps22+randn(n_points,1)*gamma(jj);
        Eps6=Eps66+randn(n_points,1)*gamma(jj);

        %% VFM appliance

        %First set of virtual fields

        A(1,1)=sum(Eps1.*dS);
        A(1,2)=sum(Eps2.*dS);
        B(1,1)=0;

        A(2,1)=sum(Eps2.*dS);
        A(2,2)=sum(Eps1.*dS);
        B(2,1)=F*L/t;

        Q=A\B;

        Q111(jj,ii)=Q(1);
        Q121(jj,ii)=Q(2);

        %Second set of virtual fields

        A(1,1)=sum((Eps2.*(X1.^2)+Eps6.*X1.*(X2)).*dS);
        A(1,2)=sum((Eps1.*(X1.^2)-Eps6.*X1.*(X2)).*dS);
        B(1,1)=F*L*w^3/12/t/w;

        A(2,1)=sum(3*Eps1.*(X1.^2).*dS);
        A(2,2)=sum(3*Eps2.*(X1.^2).*dS);
        B(2,1)=0;

        Q=A\B;

        Q112(jj,ii)=Q(1);
        Q122(jj,ii)=Q(2);

        %Third set of virtual fields
```

```
        A(1,1)=sum((Eps2.*cos(pi*X1/w)-
0.5*Eps6.*X2.*sin(pi*X1/w)*pi/w).*dS);

A(1,2)=sum((Eps1.*cos(pi*X1/w)+0.5*Eps6.*X2.*sin(pi*X1/w)*pi/w).*dS);
        B(1,1)=2*F*L*w/pi/t/w;

        A(2,1)=sum((Eps1.*cos(2*pi*(X1+w/2)/w)*2*pi/w).*dS);
        A(2,2)=sum((Eps2.*cos(2*pi*(X1+w/2)/w)*2*pi/w).*dS);
        B(2,1)=0;

        Q=A\B;

        Q113(jj,ii)=Q(1);
        Q123(jj,ii)=Q(2);

        %Special virtual fields

[Q11,Q12,~,~]=SpecialVirtualFieldsFunction(Eps1,Eps2,Eps6,dS,X1,X2,F,L,w,t,
1,1);
        Q114(jj,ii)=Q11;
        Q124(jj,ii)=Q12;

        %Special virtual fields

[Q11,Q12,~,~]=OptimizedVirtualFieldsFunction(Eps1,Eps2,Eps6,dS,X1,X2,F,L,w,
t,9,9);
        Q115(jj,ii)=Q11;
        Q125(jj,ii)=Q12;

        clear Eps1 Eps2 Eps6 A B
    end
end

figure, hold on, grid on
xlabel('Noise amplitude')
ylabel('Standard deviation / Q_1_1 objective')
title('Adimensionalized standar deviation on Q_1_1')
plot(gamma,std(Q111')/Q11obj,'LineStyle','none','Marker','v','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q112')/Q11obj,'LineStyle','none','Marker','s','MarkerFaceCol
or','k','MarkerEdgeColor','k')
%plot(gamma,std(Q113')/Q11obj,'LineStyle','none','Marker','d','MarkerFaceCo
lor','k','MarkerEdgeColor','k')
plot(gamma,std(Q114')/Q11obj,'LineStyle','none','Marker','o','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q115')/Q11obj,'LineStyle','none','Marker','d','MarkerFaceCol
or','k','MarkerEdgeColor','k')
legend('1^s^t set','2^n^d set','Special
VFM','OptimizedVFM','Location','northwest')

figure, hold on, grid on
xlabel('Noise amplitude')
ylabel('Standard deviation / Q_1_2 objective')
title('Adimensionalized standar deviation on Q_1_2')
plot(gamma,std(Q121')/Q12obj,'LineStyle','none','Marker','v','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q122')/Q12obj,'LineStyle','none','Marker','s','MarkerFaceCol
or','k','MarkerEdgeColor','k')
%plot(gamma,std(Q123')/Q12obj,'LineStyle','none','Marker','d','MarkerFaceCo
lor','k','MarkerEdgeColor','k')
```

```
plot(gamma,std(Q124')/Q12obj,'LineStyle','none','Marker','o','MarkerFaceCol
or','k','MarkerEdgeColor','k')
plot(gamma,std(Q125')/Q12obj,'LineStyle','none','Marker','d','MarkerFaceCol
or','k','MarkerEdgeColor','k')
legend('1^s^t set','2^n^d set','Special
VFM','OptimizedVFM','Location','northwest')

save ResultadosHomogenea.mat
```

9.8. Annex 8: Matlab code for section 5.3.2

```
function
[Q11,Q12,eta11,eta12,letras11,letras12,determinantes]=SpecialVirtualFieldsD
isc(Eps1,Eps2,Eps6,X1,X2,dS,d,t,F,m,n)
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.3.3
% Special virtual fields function for a compressed disc

% DESCRIPTION OF THE INPUTS:
% The inputs will be the following ones:

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% L is the length of the specimen (in e2)
% w is the width of the specimen (in e1)
% t is the thickness of the disc in mm
% m is the degree of the polynomial on e1
% n is the degree of the polynomial on e2

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Units are in N and m: stiffnesses will be obtained in Pa
%Datos para el VFM:

n_dof=(m+1)*(n+1);
n_points=length(X1);

for dd=2:n_dof
    j=floor((dd-1)/(m+1)); %browses through all possible couples of (i,j)
values
    i=(dd-1)-j*(m+1);

    ilog(dd)=i; % For information: all incremental values of i
    jlog(dd)=j; % For information: all incremental values of j

    if i==0 % if i=0, then dx2 is zero for all points
```

```
        dx1=zeros(n_points,1);
    else
        dx1=i*(X1/d).^(i-1).*(X2/d).^(j)*1/d; % Monomials derivative with
respect to X1
    end
    if j==0 % if j=0, then dx2 is zero for all points
        dx2=zeros(n_points,1);
    else
        dx2=j*(X1/d).^(i).*(X2/d).^(j-1)/d; % Monomials derivative with
respect to X2
    end

    b1(:,dd)=dx1; % Contains information on epsilon1*, only depends on the
Aij's
    b2(:,dd)=zeros(n_points,1); % information on epsilon2* does not depend
on the Aij's
    b6(:,dd)=dx2; % Contains information on epsilon6*, depends on the Bij's
    b1(:,dd+n_dof)=zeros(n_points,1); % information on epsilon1* does not
depend on the Bij's
    b2(:,dd+n_dof)=dx2; % Contains information on epsilon2*, only depends
on the Bij's
    b6(:,dd+n_dof)=dx1; % Contains information on epsilon6*, depends on the
Aij's
end

%We calculate now the vectors corresponding to integrals pertaining to Q11
(first) and Q12 (second)

Eps11=repmat(Eps1,1,size(b1,2));
Eps22=repmat(Eps2,1,size(b1,2));
Eps66=repmat(Eps6,1,size(b1,2));
dSS=repmat(dS,1,size(b1,2));

Integral11=sum(((Eps11.*b1)+(Eps22.*b2)+(0.5*Eps66.*b6)).*dSS);
Integral12=sum(((Eps22.*b1)+(Eps11.*b2)-(0.5*Eps66.*b6)).*dSS);

E11=zeros(n_dof*2,1);E11(4)=1;
E12=zeros(n_dof*2,1);E12(5)=1;

H11=(b1.*dSS)'*(b1.*dSS);
H22=(b2.*dSS)'*(b2.*dSS);
H12=(b1.*dSS)'*(b2.*dSS);
H66=(b6.*dSS)'*(b6.*dSS);

iter=0;
cont=0;
tic
for i=1
    for j=2:n_dof*2-3
        for k=j+1:n_dof*2-2
            for l=k+1:n_dof*2-1
                for o=l+1:n_dof*2
                    D=zeros(n_dof*2);
                    D(1,1)=1;
                    D(2,n_dof+1)=1;
                    D(3,1:m+1:n_dof)=1;
                    D(4,:)=Integral11;
                    D(5,:)=Integral12;

                    xd=ones(1,n_dof*2);
```

```
                    xd(i)=0;xd(j)=0;xd(k)=0;xd(l)=0;xd(o)=0;
                    xx=find(xd);

                    for p=6:n_dof*2
                        D(p,xx(p-5))=1;
                    end
                    cont=cont+1;
                    determinantes(cont)=det(D);
                    if round(det(D),21)~=0

                        Y11=D\E11;
                        Y12=D\E12;
                        if sum(Y11(n_dof+1:m+1:n_dof*2))~=0 ||
sum(Y12(n_dof+1:m+1:n_dof*2))~=0

                            Q11=sum(Y11(n_dof+1:m+1:n_dof*2))*F/t;
                            Q12=sum(Y12(n_dof+1:m+1:n_dof*2))*F/t;
                            Q66=(Q11-Q12)/2;
                            Nu=Q12/Q11;
                            E=Q11*(1-Nu^2);


eta11=sqrt(Y11'*(2*((Q11^2+Q12^2)*H11+(Q11^2+Q12^2)*H22+Q66^2*H66+4*Q11*Q12
*H12))*Y11);

eta12=sqrt(Y12'*(2*((Q11^2+Q12^2)*H11+(Q11^2+Q12^2)*H22+Q66^2*H66+4*Q11*Q12
*H12))*Y12);


                            if E~=0
                                iter=iter+1;
                                mat(:,iter)=[E Nu Q11 Q12 eta11 eta12];
                                Y111(:,iter)=Y11;
                                Y122(:,iter)=Y12;
                                letras(:,iter)=[i j k l o];
                            end
                        end
                    end
                end
            end
        end
    end
end
[eta11,pos11]=min(mat(5,:));
[eta12,pos12]=min(mat(6,:));
Q11=mat(3,pos11);
Q12=mat(4,pos12);
letras11=letras(:,pos11);Y111(letras11,pos11)==0
letras12=letras(:,pos12);Y122(letras12,pos12)==0
```

9.9. Annex 9: Matlab code for section 5.3.3

```
function
[Qiso1,Qiso2,etaiso1,etaiso2]=OptimizedVirtualFieldsFunctionDisc(Eps1,Eps2,
Eps6,dS,X1,X2,F,L,w,t,m,n)

%% Original developers:
% The Virtual Fields Methods: Extracting constitutive mechanical parameters
% from full-field deformation measurements, based on experimental data.
% F. Pierron, M. Grédiac
```

```
% Springer
% April 2010
% Based on the paper "Sensitivity to the virtual fields method to noisy
% data", Avril, Grediac, Pierron, - Comp. Mech. (2004)

%% Modified by:
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Section 4.3.4
% Optimized virtual fields function for a compressed disc

% DESCRIPTION OF THE INPUTS:
% The inputs will be the following ones:

% Geometrical parameters
% e1 is the horizontal axis, e2 the vertical one (compression is along e2)
% X1 is the vector containing the abscissa of the elements centroids
% X2 is the vector containing the ordinates of the elements centroids
% dS is the vector containing the surface of each element
% L is the length of the specimen (in e2)
% w is the width of the specimen (in e1)
% t is the thickness of the disc in mm
% m is the degree of the polynomial on e1
% n is the degree of the polynomial on e2

% Load
% F is the compressive load in N

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)

% Units are in N and m: stiffnesses will be obtained in Pa

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Parameter definition and data formatting
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% define the size of the arrays
n_dof=(m+1)*(n+1); %number of degrees of freedom for each component of the
virtual displacements
% Total number of degrees of freedom: 2*(m+1)(n+1).
% This number should be larger than the number of constraints, ie, 4 for
% the speciality conditions (or number of stiffnesses to identify) and
% 3(n+1)+n for the virtual boundary conditions (specific to this test
% configuration).

n_points=length(Eps1); % Total number of measurement points

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Building up the virtual fields
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
b1=zeros(n_points,n_dof*2); % Initialization to zero
b2=zeros(n_points,n_dof*2);
b6=zeros(n_points,n_dof*2);
```

```
for dd=2:n_dof
    j=floor((dd-1)/(m+1)); %browses through all possible couples of (i,j)
values
    i=(dd-1)-j*(m+1);

    ilog(dd)=i; % For information: all incremental values of i
    jlog(dd)=j; % For information: all incremental values of j

    if i==0 % if i=0, then dx2 is zero for all points
        dx1=zeros(n_points,1);
    else
        dx1=i*1/w*(X1/w).^(i-1).*(X2/L).^(j); % Monomials derivative with
respect to X1
    end
    if j==0 % if j=0, then dx2 is zero for all points
        dx2=zeros(n_points,1);
    else
        dx2=j*1/L*(X1/w).^(i).*(X2/L).^(j-1); % Monomials derivative with
respect to X2
    end

    b1(:,dd)=dx1; % Contains information on epsilon1*, only depends on the
Aij's
    b2(:,dd)=zeros(n_points,1); % information on epsilon2* does not depend
on the Aij's
    b6(:,dd)=dx2; % Contains information on epsilon6*, depends on the Bij's
    b1(:,dd+n_dof)=zeros(n_points,1); % information on epsilon1* does not
depend on the Bij's
    b2(:,dd+n_dof)=dx2; % Contains information on epsilon2*, only depends
on the Bij's
    b6(:,dd+n_dof)=dx1; % Contains information on epsilon6*, depends on the
Aij's

    u1(:,dd)=(X1/w).^(i).*(X2/L).^(j); % u1 only depends on the Aij's
    u1(:,dd+n_dof)=zeros(n_points,1);
    u2(:,dd)=zeros(n_points,1); % u2 only depends on the Aij's
    u2(:,dd+n_dof)=(X1/w).^(i).*(X2/L).^(j);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Construction of the quantities based on the
% virtual fields in the final optimization matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The Bij's will be used to build up the conditions of speciality
% of the virtual fields (in submatrix A of the final system)

B11=zeros(1,n_dof*2); % Initialization
B22=zeros(1,n_dof*2);
B12=zeros(1,n_dof*2);
B66=zeros(1,n_dof*2);

for kk=1:n_dof*2
    B11(kk)=sum(b1(:,kk).*Eps1.*dS); % Quantities for the speciality
conditions
    B22(kk)=sum(b2(:,kk).*Eps2.*dS);
    B12(kk)=sum((b2(:,kk).*Eps1+b1(:,kk).*Eps2).*dS);
    B66(kk)=sum(b6(:,kk).*Eps6.*dS);
end
```

```
% The Hij's will be used to build up the Hessian matrix
% H of the final system

dSS=repmat(dS,1,size(b1,2));

H11=(b1.*dSS)'*(b1.*dSS);
H22=(b2.*dSS)'*(b2.*dSS);
H12=(b1.*dSS)'*(b2.*dSS);
H66=(b6.*dSS)'*(b6.*dSS);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Virtual boundary conditions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Aconst=zeros(3,n_dof*2); % There are 3*(n+1)+n boundary conditions here

Aconst(1,1)=1;
Aconst(2,n_dof+1)=1;
Aconst(3,1:m+1:n_dof)=1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Constitution of the Z vector containing zeros
% except for the conditions of speciality
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Za is the Z vector used for finding the virtual field to identify Q11
% Zb is the Z vector used for finding the virtual field to identify Q22
% Zc is the Z vector used for finding the virtual field to identify Q12
% Zd is the Z vector used for finding the virtual field to identify Q66
Za=zeros(1,n_dof*2+size(Aconst,1)+1); % Za is the Z vector used for finding
the virtual field to identify Q11
Zb=Za;Zc=Za;Zd=Za;

Za(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[1 0 0 0]; % special
virtual field condition for Q11
Zb(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[0 1 0 0]; % special
virtual field condition for Q22
Zc(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[0 0 1 0]; % special
virtual field condition for Q12
Zd(n_dof*2+size(Aconst,1)+1:n_dof*2+size(Aconst,1)+4)=[0 0 0 1]; % special
virtual field condition for Q66

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Building up the A matrix containing all constraints:
% boundary conditions and speciality and the
% B matrix containing zeros
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A=[Aconst;B11;B22;B12;B66]; % Constraints for the boundary conditions
(Aconst) and the speciality conditions (Bij)
B=zeros(size(A,1));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Solving the system (optimization)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Q=[1 1 1 1]; % initializing value for Q because H depends on the Qij's

n_iter=20; % maximum number of iteration
delta_lim=0.0001; % tolerance

delta=10; % Starting with a tolerance larger than delta_lim
i=1;
```

```
Qold=Q; % Storing the current Q into Qold for checking convergence

while i<n_iter && delta>delta_lim

    % Hessian matrix
    H =2*((Q(1)^2+Q(3)^2)*H11+(Q(2)^2+Q(3)^2)*H22+...
        Q(4)^2*H66+2*(Q(1)+Q(2))*Q(3)*H12);

    %NOTE: to avoid the numerical "Warning: Matrix is close to singular or
    %       badly scaled" matrix Opt can be scaled with the parameter corr.
    %       It does not change the results of the optimization.
    %       To avoid using, put corr=1;

    corr=max(max(A))/max(max(H)); % Normalization coefficient
    OptM=[H*corr,A'*corr;A,B]; % Matrix for the optimization of the virtual
fields

    Ya=OptM\Za'; %Vector containing the polynomial coefficients for Q11 and
the Lagrange multipliers
    Yb=OptM\Zb'; %Vector containing the polynomial coefficients for Q22 and
the Lagrange multipliers
    Yc=OptM\Zc'; %Vector containing the polynomial coefficients for Q12 and
the Lagrange multipliers
    Yd=OptM\Zd'; %Vector containing the polynomial coefficients for Q66 and
the Lagrange multipliers

    Ya(n_dof*2+1:size(Ya))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest
    Yb(n_dof*2+1:size(Yb))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest
    Yc(n_dof*2+1:size(Yc))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest
    Yd(n_dof*2+1:size(Yd))=[]; % Removing the Lagrange multipliers from the
Y vectors because they are of no interest

    Q(1)=(sum(Ya(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q11 from the
first optimized virtual field
    Q(2)=(sum(Yb(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q22 from the
second optimized virtual field
    Q(3)=(sum(Yc(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q12 from the
third optimized virtual field
    Q(4)=(sum(Yd(n_dof+1:m+1:n_dof*2))*F/t); % Calculating Q66 from the
fourth optimized virtual field

    delta=sum((Qold-Q).^2./Q.^2); % Difference between the current and
previous values of the identified parameters
    Qold=Q; % The current parameters are stored as the previous one before
the next iteration
    i=i+1; % Next step
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Final results
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Final Hessian matrix for computing the eta parameters
H =2*((Q(1)^2+Q(3)^2)*H11+(Q(2)^2+Q(3)^2)*H22+...
    Q(4)^2*H66+2*(Q(1)+Q(2))*Q(3)*H12);

eta(1) = sqrt(0.5*Ya'*H*Ya); % Sensitivity to noise parameter for Q11
```

```
eta(2) = sqrt(0.5*Yb'*H*Yb); % Sensitivity to noise parameter for Q22
eta(3) = sqrt(0.5*Yc'*H*Yc); % Sensitivity to noise parameter for Q12
eta(4) = sqrt(0.5*Yd'*H*Yd); % Sensitivity to noise parameter for Q66

if eta(1)>eta(2)
    Qiso1=Q(2);
    etaiso1=eta(2);
else
    Qiso1=Q(1);
    etaiso1=eta(1);
end

if (etaiso1+eta(4))/2>eta(3)
    Qiso2=Q(3);
    etaiso2=eta(3);
else
    Qiso2=(Qiso1-Q(4))/2;
    etaiso2=(etaiso1+eta(4))/2;
end
```

9.10. Annex 10: Ncorr usage and data export

In this annex we are going to show the way of using the Ncorr software in order to obtain the strain fields, and the code used to export this cata into a suitable mat file for the developed GUI.

A view of the software just as it is once opened can be seen here:

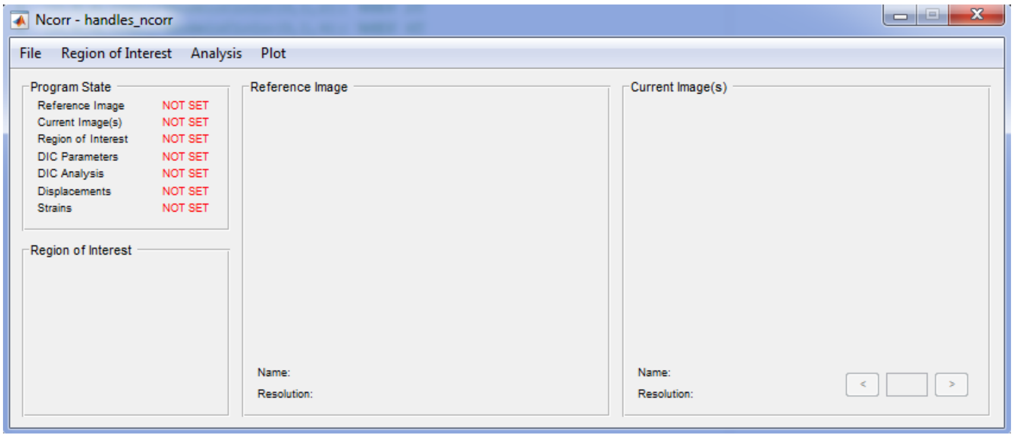


Figure 122. First view of the Ncorr software

The first step now will be loading the reference images as well as the deformed images, or current images as they are called in the software.

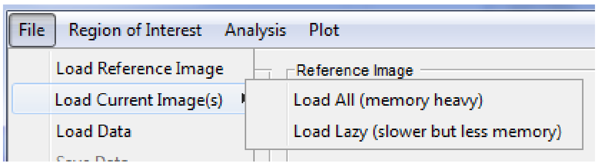


Figure 123. Images loading menu

Once the images have been loaded, the software should look like this:

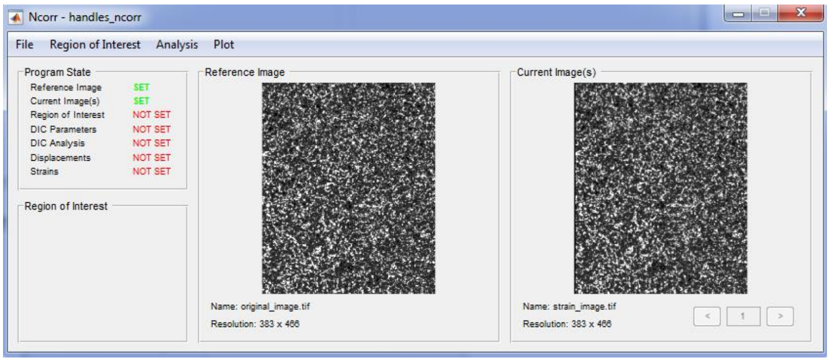


Figure 124. Software appearance after image loading

The next step is defining the region of interest of the image. To that end, we must select the following options:

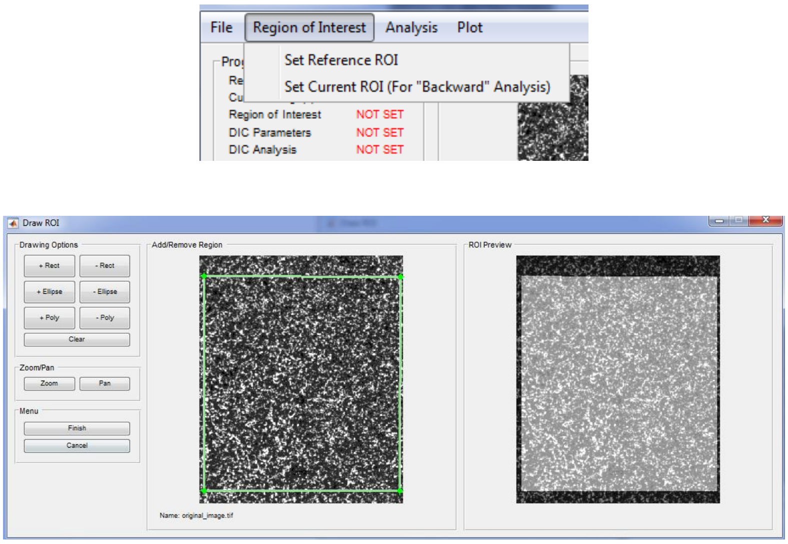


Figure 125. ROI drawing

Once the ROI is defined we must set the DIC parameters which will be used. In this example we will choose the following ones:

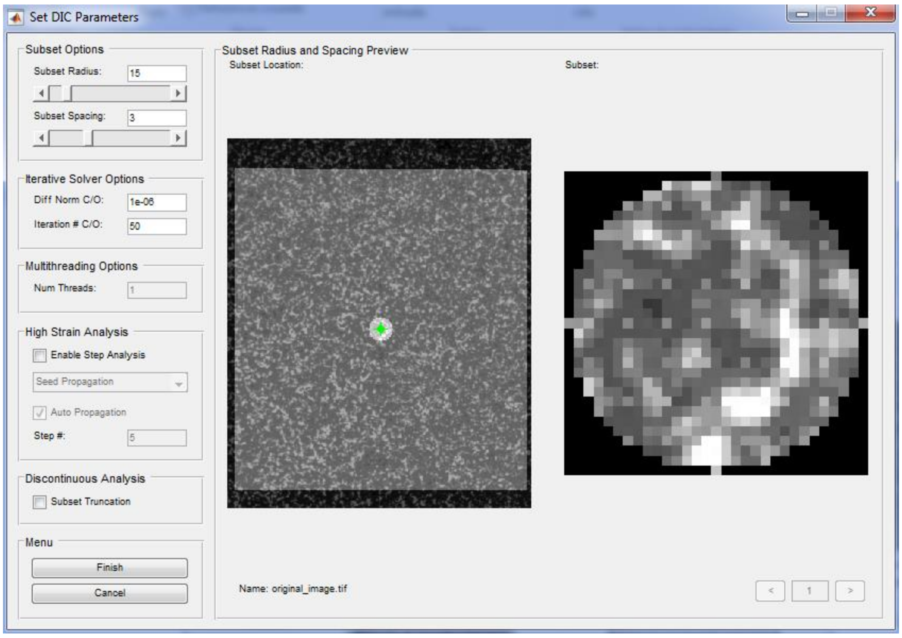


Figure 126. DIC parameters

Once the subset radius and subset spacing is set (we let the other terms as they are, and do not enable the high strain analysis since it seems to not being required in this test). The next step will be the seed checking for being sure that the defined parameters will provide good results.

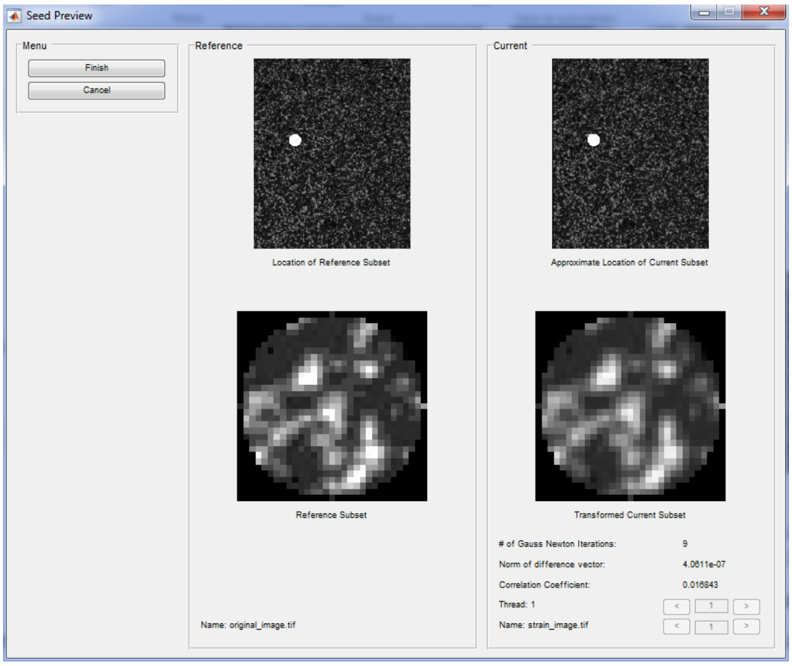


Figure 127. Seed preview

If we like the seeds displacement, we are ready then for applying DIC techniques to the images. This will be done after accepting the seed preview. Now we go into the format displacements menu, in which we will assign the pixel-millimetres ratio.

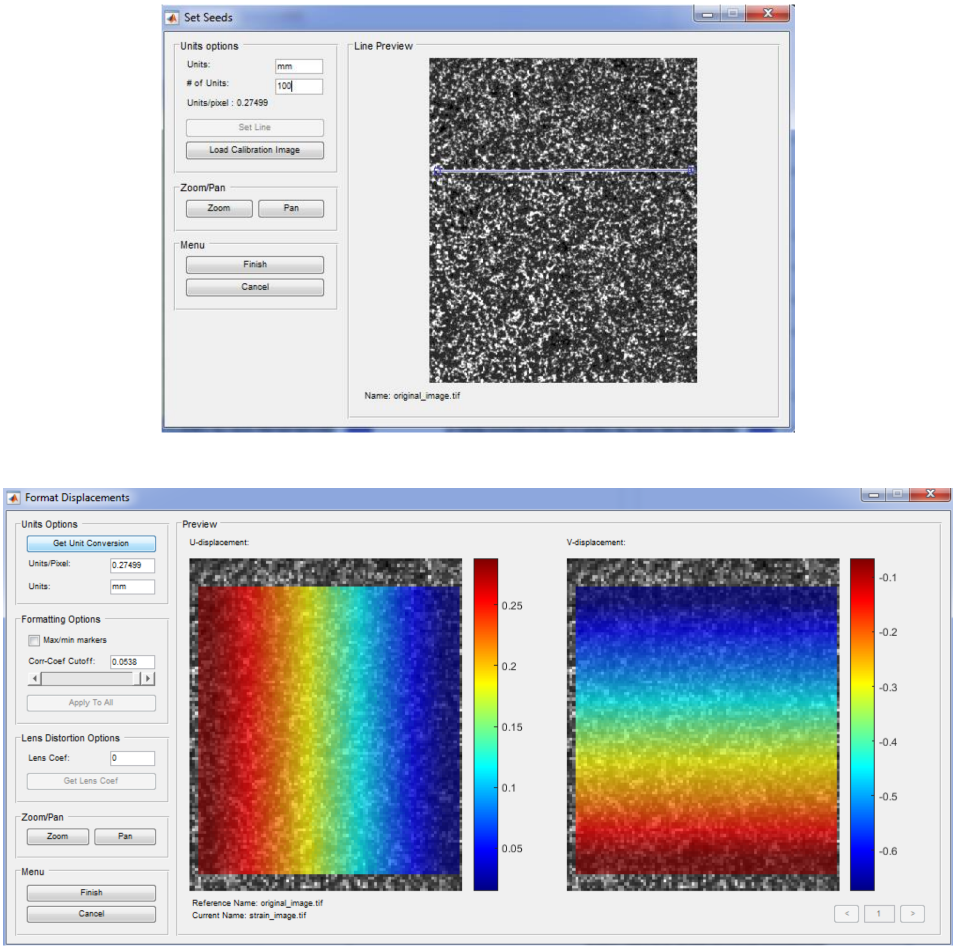


Figure 128. Format displacements menu

The last step on this software will be computing the strains obtained. To that end we get into the strains menu, which should look like this:

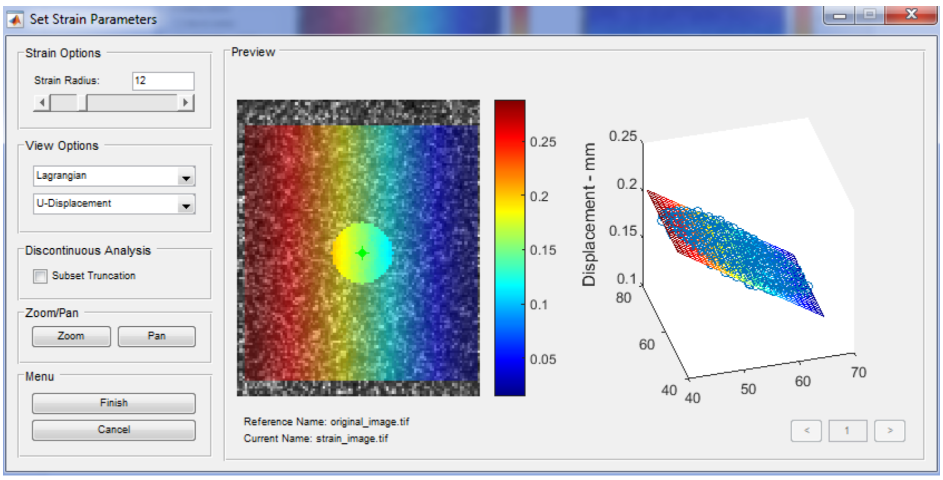


Figure 129. Strain parameters menu

Here we can set a large strain radius since we have an homogeneous displacement field. Once we finish in this window the DIC analysis will be completed. Now we can consult the outputs in the plot menu:

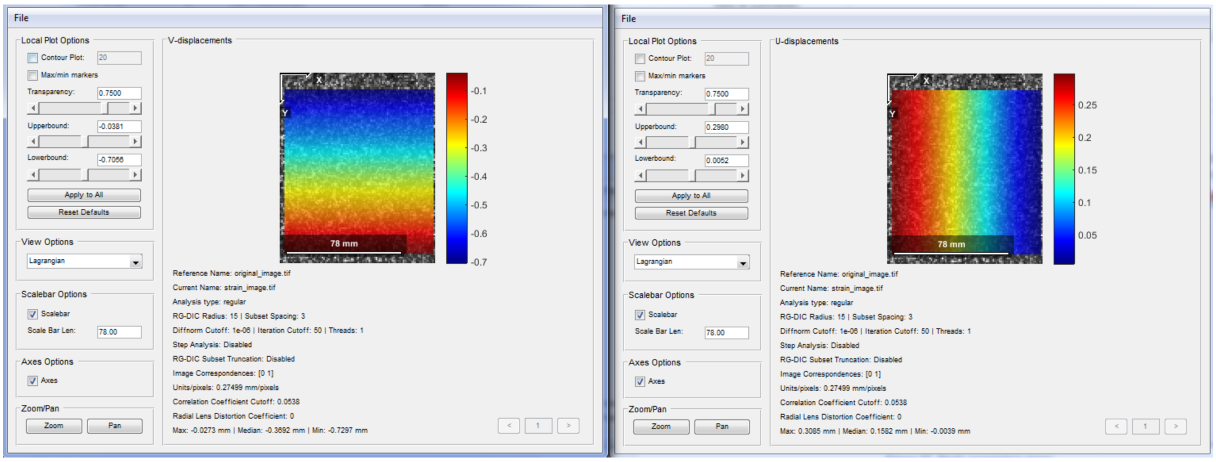


Figure 130. Displacements fields plots

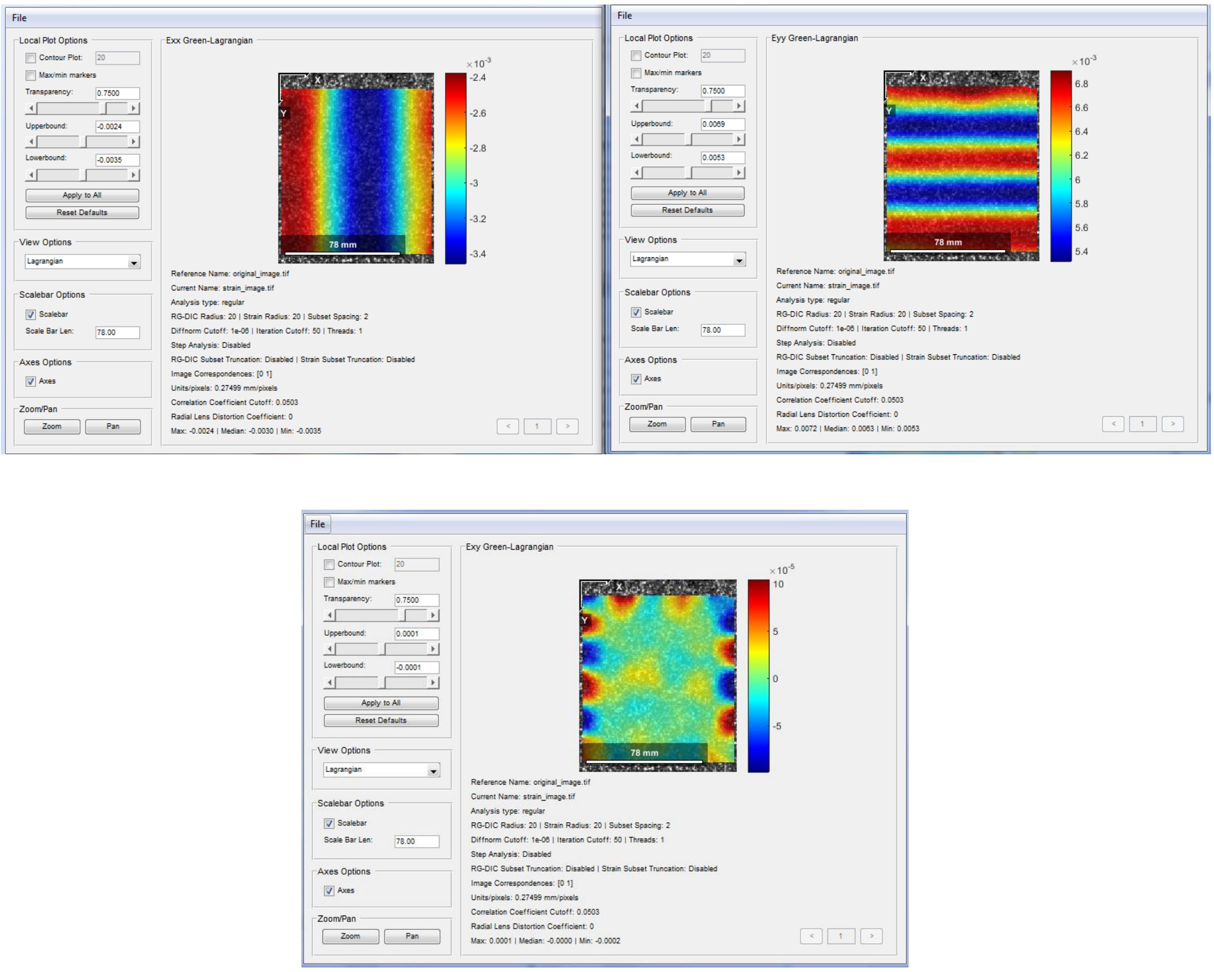


Figure 131. Strain fields plots

Now we must export this data into a file for being able to introduce it into our GUI. To that end, we use the following Matlab code:

```
% VFM implementation for solving mechanical problems
% David Bonillo Martínez
% 2018

% Ncorr exporting script

% For the correct using of the present script the software Ncorr must be
% open and must have loaded the required data.

%Inputs:
% im is the number of the image which we want to export the data from
% w is the width of the specimen in metres, or its diameter in case we are
looking at
% a compression disc test

% Deformations
% Eps1 is the vector containing the epsilon 1 strain values
% Eps2 is the vector containing the epsilon 2 strain values
% Eps6 is the vector containing the epsilon 6 strain values (engineering
% shear strain)
% dS is the vector containing the elemt surface on each point
% X1 is te vector containing the X1 coordinate of each point
% X2 is te vector containing the X1 coordinate of each point


clearvars -except handles_ncorr
close all

im=1; %Image from which you want to extract the required data
w=0.1;
clearvars -except handles_ncorr im Es Nus Q11s Q12s eta11s eta12s w

exx=handles_ncorr.data_dic.strains(im).plot_exx_ref_formatted;
eyy=handles_ncorr.data_dic.strains(im).plot_eyy_ref_formatted;
exy=handles_ncorr.data_dic.strains(im).plot_exy_ref_formatted;
exx=flipud(exx);
eyy=flipud(eyy);
exy=flipud(exy);
upp=handles_ncorr.data_dic.dispinfo.pixtounits/1e3; %m/pixel
pdl=(handles_ncorr.data_dic.dispinfo.spacing+1); %pixeles/diferencial de
longitud

%%Recortamos las matrices
[f,c]=size(exx);
i=0;
while i<f
    i=i+1;
    if find(exx(i,:),1)

    else
        exx(i,:)=[];
        eyy(i,:)=[];
        exy(i,:)=[];

        i=i-1;
```

```
        f=f-1;
    end
end
[f,c]=size(exx);
i=0;
while i<c
    i=i+1;
    if find(exx(:,i),1)

    else
        exx(:,i)=[];
        eyy(:,i)=[];
        exy(:,i)=[];

        i=i-1;
        c=c-1;
    end
end

[f,c]=size(exx);
xlim=linspace(-w/2+upp*pdl*0.501,w/2-upp*pdl/2,c);
ylim=linspace(0+upp*pdl/2,f*upp*pdl-upp*pdl/2,f);
[X,Y] = meshgrid(xlim,ylim);

num=0;
for i=1:f
    for j=1:c
        if exx(i,j)~=0

            num=num+1;
            Eps1(num,1)=exx(i,j);
            Eps2(num,1)=eyy(i,j);
            Eps6(num,1)=exy(i,j);
            X1(num,1)=X(i,j);
            X2(num,1)=Y(i,j);
        end
    end
end

upp=handles_ncorr.data_dic.dispinfo.pixtounits/1e3; %m/pixel
pdl=(handles_ncorr.data_dic.dispinfo.spacing+1); %pixeles/diferencial de
longitud
pds=pdl^2; %Pixeles/diferencial de superficie
dS=ones(num,1)*pds*upp^2;
clearvars -except Eps1 Eps2 Eps6 X1 X2 dS handles_ncorr

 Eps6=-Eps6;
save Data_Test.mat Eps1 Eps2 Eps6 X1 X2 dS
```

10. List of illustrations

11. Tables index

12. Literature

[1]. Pierron, F. (2012), *The Virtual Fields Method. Extracting Constitutive Mechanical Parameters from Full-fields Deformation Measurements.* Springer.

[2]. Grédiac, M.; Pierron, F. (2002). *Special virtual fields for the direct determination of material parameters with the virtual fields method. 1–Principle and definition.*

[3]. Grédiac, M.; Pierron, F. (2002). *Special virtual fields for the direct determination of material parameters with the virtual fields method. 2–Application to in-plane properties*

[4]. Grédiac, M.; Pierron, F. (2002). *Special virtual fields for the direct determination of material parameters with the virtual fields method. 3–Application to the bending rigidities of anisotropic plates.*

[5]. Grédiac, M.; Pierron, F. (2004). *Sensitivity of the virtual fields method to noisy data.*

[6]. Dai, X.; Yang, F.; Chen, Z.; Shao, X.; He, X.; (2014). *Strain field estimation based on digital image correlation and radial basis function.*

[7]. Hawinkels, R.J.H (2011). *Fatigue crack propagation in polycarbonate.*

[8]. *http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=ma2024t3*

[9]. *http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=MTU020*

[10]. *https://github.com/FranckLab/DIC-Simulator*

[11]. Carter, J.; Uchic, M. (2014). *Impact of speckle pattern parameters on DIC strain resolution calculated from in-situ experiments.*

[12]. Kramer, S.; Scherzinger, W. (2014). *Implementation and Evaluation of the Virtual Fields Method: Determining Constitutive Model Parameters From Full-Field Deformation Data.* Sandia National Laboratories.