Project Report

Analysis and characterization of a photonic crystal made with coaxial cables

Presented by

Profile	 Nina MOELLO [NM]	 Enguerran VIDAL [EV]	 Maxime THUMIN [MT]
Function	Project manager	Specification Engineer	Test Engineer
Organization	Université Paul Sabatier – Toulouse, FR		

Client

Name	M. Hassan SABBAH

Last modification date: 10/01/2020

Table of contents

Introduction

Photonic crystals are periodic structures of dielectric or semiconductor materials modifying the propagation of electromagnetic waves in the same way as a periodic potential in a semi-crystal conductor affects the movement of electrons by creating permissible and prohibited energy bands.

The coaxial cable is the modeling of a photonic crystal for a lower price. It is used to connect instrumentation devices. It then serves as a current conductor but with the particularity of not producing or capturing external flow. It is characterized by its impedance and length.

This cable could be, for example, used to illustrate the study of band structures in solids which is not a topic that yields easily to experimentation. It also introduces the concept of dispersion relations.

This project focused on the behavior of an electric wave throughout such a cable. In recent years, A. Haché and A. Slimani published an article in which they studied the dispersion relation, the transfer function and the group velocity in a 120 m coaxial cable, described in this paper. The current draft repeats these results by pushing the experiments to higher frequencies. This paper is supplemented by a modelling of the system studied in python format.

Keywords: coaxial cable, periodic, propagation, dispersion.

Contract Document

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Contract_Coaxial_Cable.v1
Drafting date: 16/11/2019

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

Version	Date	Modification performed
1.0	16/11/2019	Document creation
1.1	24/11/2019	Modification after discussion with the client

Statement of work

A coaxial photonic crystal is a quasi-periodic structure obtained by alternating some coaxial cable sections of 5m length and of different characteristic impedance (50Ω or 75Ω). The two ends of this structure can be considered as the input and the output of a linear time-invariant system (LTI system). The supplier undertakes to use a generator which has a bandwidth of dozens of MHz to test the Heaviside model for a homogeneous coaxial cable. He will then raise the transfer function H(jω) of a photonic crystal. This system is a complex propagation environment where the wave cross different homogeneous environments and undergoes reflections at the interfaces represented by the borders between these homogenous environments. Based on this finding, the coaxial cable theory and using a matrix formalism, the supplier will propose a transfer function simulated that will be compared to the one measured experimentally. The supplier will measure the dispersion curve ω(k) and then, will study the propagation of a sinusoidal signal amplitude modulated by a Gaussian. He will measure the group velocity depending on the supporting function frequency. All the results will be commented and compared to the results presented in the article of reference.

The supplier will manage the project, following the methodology associated to the V-model. Particularly, he will provide to the client the requested documents on the scheduled dates.

Development Plan

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Development_Coaxial_Cable.v1
Drafting date: 24/11/2019

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

Version	Date	Modification performed
1.0	24/11/2019	Document creation
1.1	26/11/2019	Modification after discussion with the group

Table of contents

1. Requirements

After a first meeting with the client, a list of requirements and specific demands about the product expected was done.

A. Objectives

- To characterise a composite cable through its transfer function and the way it links temporal and spatial frequency;
- To study how a Gaussian wave packet behaves in this composite cable;
- To model the wave propagation in this periodic structure;
- To deliver the test and validation reports, and all the documents the client will need from us to be satisfied.

B. Article of reference

« *A model coaxial photonic crystal for studying band structures, dispersion, field localization, and superluminal effects* » A. Haché & A. Slimani, 2004

C. System studied

Assembly of coaxial cables of about 5 m each, for a total of 120 m. The cables may have different properties (they will be distinguished by their impedances of 50Ω or 75Ω). This system, which will be called the composite cable, then represents a quasi-periodic structure whose ends will be considered as the input and output of a linear time-invariant system (LTI system).

Fig. 1: Picture of the composite cable.

D. Work Packages

To achieve the purposes, there are three principal means: modelling, simulation and experiment. The project is divided in Work Packages (WP), described below.

WP1: Study of a homogeneous coaxial cable

⇒ Discover the properties of the coaxial cable;
⇒ Illustrate the phenomena of propagation, reflection and dispersion of waves easily and cheaply than with a photonic crystal;
⇒ Test Heaviside modelling of a homogeneous coaxial cable;

WP2: Restore the composite cable transfer function

⇒ Reproduce the transfer function showed in the article of reference (by Haché and Slimani);
⇒ Extend it to a wider range of frequencies;
⇒ Compare our results to the article's ones.

WP3: Find the link between spatial and temporal frequency in this system

⇒ Reproduce the evolution of the spatial frequency as a function of the temporal frequency showed in the article of reference (by Haché and Slimani);
⇒ Study and understand the phenomena behind this figure;
⇒ Compare our results with the article's ones.

WP4: Study the propagation of a Gaussian wave packet in a quasi-periodic structure

⇒ By injecting a packet of Gaussian waves (amplitude-modulated sinusoidal signal) into the composite cable, study and characterize the system output response;
⇒ Measure the group speed according to the carrier frequency.

WP5: Model the system, with the first study

⇒ Retrieve the transfer function digitally by a matrix calculation;
⇒ Theoretically model all the curves obtained experimentally.

E. Scope and limitations:

The situation described in this specification must meet the objectives and needs mentioned above. For reasons of delay, the study of the amplitude of the electric field inside the cable for a sinusoidal signal, the study of the distribution of the electric field in a symmetrical closed loop and the in-depth study of supraluminic phenomena have been excluded for the moment. However, these features are part of the long-term vision of the services that will be offered by the company.

F. Terms definitions

Term	Definition
Homogeneous cable	A coaxial cable is a central conductor (copper-based), surrounded by insulation and then a conductive sheath (braided or coiled) that acts as a shield. The whole is covered with a PVC envelope and has no impedance change.
Composite cable	The composite cable will be our main design system and will consist of a series of homogeneous coaxial cables of 50Ω (RG-58U) and 75Ω (RG-59U) of 5m each alternated to form a total system of 120m.

2. Resources

A. Human resources

Name	Nina MOËLLO	Enguerran VIDAL	Maxime THUMIN
Function	Project Manager	Specification Engineer	Test Engineer
Organization	L3 Spé Physique Université Paul Sabatier		

Note: We decided the Conception Engineer tasks were going to be achieved by the three of us, together, as we thought it was the harder work part.

B. Main material resources

- A meeting room;
- A laboratory for carrying out the experiments;
- A Frequency Generator (GBF) with its technical documentation;
- An oscilloscope;
- Software: Python, OpenOffice and Google Drive.

The budget of this study is limited: there is no authorization to make any purchase on behalf of the University Paul Sabatier Toulouse in order to answer our problem. In this respect, the use of equipment supplied by the university, or the instrumentation equipment is the only material made available.

3. Deliverables

The following documents will be delivered at the end of the project:

- A summary of the theoretical part
- The test and validation reports
- The program lists or algorithms created

All the documentation will be delivered in English, during the final review, in January 2020, in paper and PDF format.

4. Quality: methods and process

A. Development method

To complete the tasks, a development of the V-Model is established as shown on the Figure 1 below. Here are the different steps:

- Specification
- Design
- Tests (programming and conducting)
- Integration
- Validation

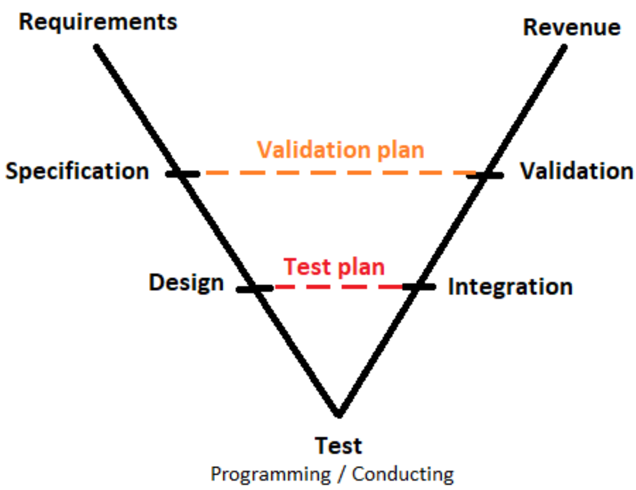


Fig. 2 : V-Model.

A weekly update is sent to the client reviewing the different Work Package tasks, their duration and their status. It also shows the risks management table and the GANTT diagram.

Weekly meetings are planned with the client in order to follow the project progress.

B. Document management

To share the document easily: creation of a Drive reachable by the team. Every published document is referenced this way:

Documentname_Coaxial_Cable.vVersionnumber

C. Risks management

The assessment of the risks leads to the filling in of the table reachable in the Risks Document. Based on the nature and the criticality of the risks, they could be discussed with the client. This scale estimates the criticality:

Major (critical impact), Moderate (important impact but not critical, Minor (negligible impact).

5. Planning

Here are the key dates:

Date	Event
25/10/2019	First meeting with the client
05/11/2019	First practical part
17/12/2019	Mid-term revue
06/01/2020	Final revue

The planning is achieved and followed thanks to the Gantt diagram.

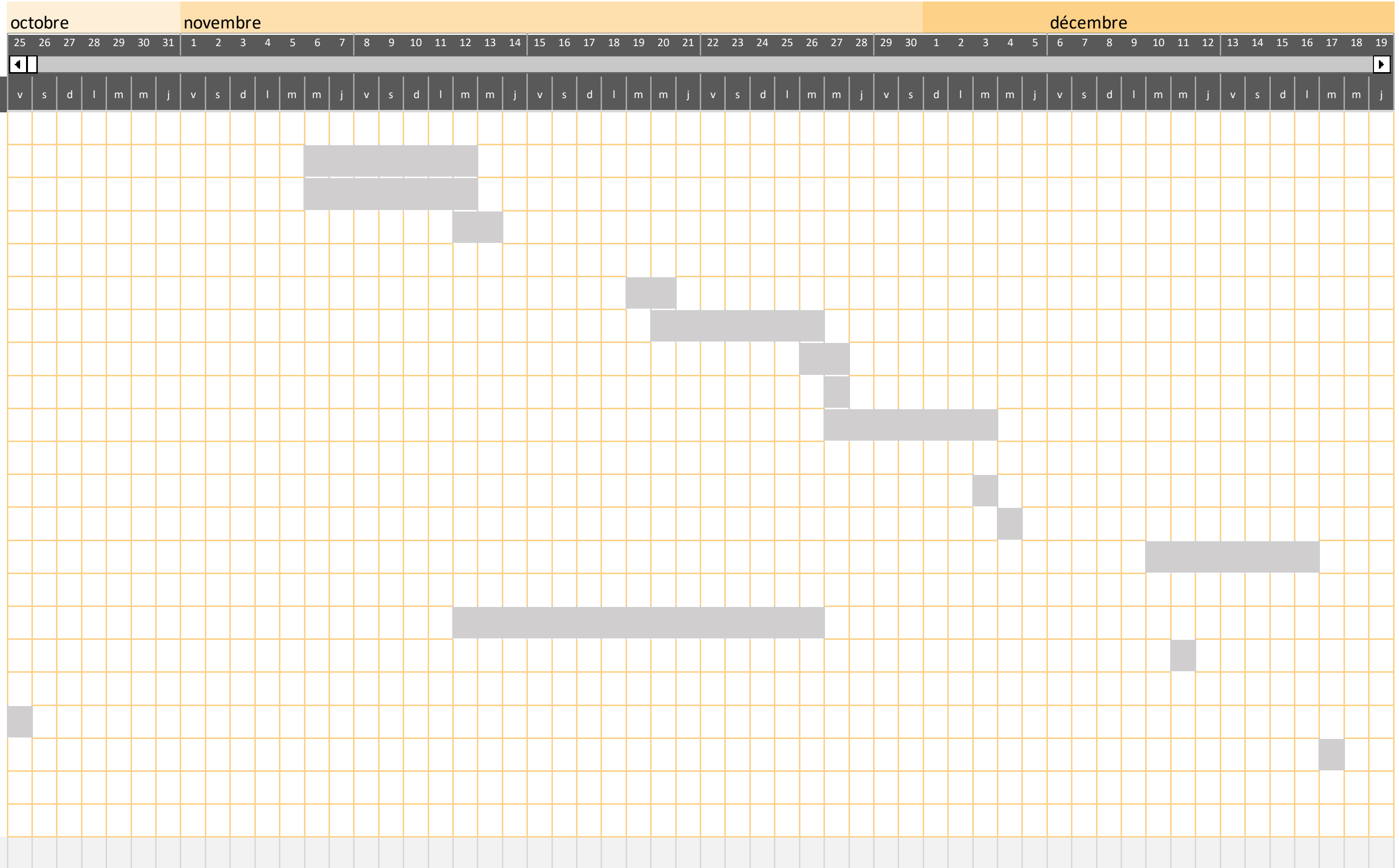
Analysis and characterization of a photonic crystal made with coaxial cables

Université Paul Sabatier - TOULOUSE
N. MOELLO, M. THUMIN, E.

Date de début du projet : 25/10/2019

Incrément de défilement : 0

Task Description	Assigned to	Progress	Start	Number of days
Study of a homogeneous coaxial cable				
Main characteristics of the different cables	Group	100%	06/11/2019	7
Pulse attack	Group	100%	06/11/2019	7
Attenuation measurement	Group	100%	12/11/2019	2
Tranfer function and dispersion relation				
Input and output measurement	NM, MT	100%	19/11/2019	2
Comparison with the theory - coding	Group	100%	20/11/2019	7
Phase shift measurement	NM, MT	80%	26/11/2019	2
Theorical calculation	EV	80%	27/11/2019	1
Comparison with the theory - coding	Group	80%	27/11/2019	7
Propagation of a gaussian wave packet				
Creation of the gaussian wave packets	EV	100%	03/12/2019	1
Measurement of the mass centre, group velocity	Group	100%	04/12/2019	1
Comparison with the theory - coding	Group	100%	10/12/2019	7
Model the system				
Matrix assembling	EV	100%	12/11/2019	15
Comparison with the precedent results	Group	100%	11/12/2019	1
Meeting dates				
First meeting with the client	Group	100%	25/10/2019	1
Stage point	Group	100%	17/12/2019	1
Holidays	Group	100%	21/12/2019	15
Final Presentation	Group	0%	14/01/2020	1



Dashboard

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Dashboard_Coaxial_Cable.v1
Drafting date: 25/10/2019

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

This document allows to know the progress of the project. It is completed and modified throughout the life of the project. It also contains the scheduled and completed meeting dates with the client and those completed internally.

Date	Done during the session	To do for the next session	Remarks
25/10/2019	Meeting with the client; explanation of the project and the different tasks.	Prepare an oral presentation which summarizes the main lines of the project.	/
04/11/2019	Group meeting and preparation of the oral presentation.	Oral presentation.	/
05/11/2019	Oral presentation of the subject.	Prepare the first task: Characterization of a homogenous coaxial cable.	/
06/11/2019	Characterization of a homogeneous coaxial cable: preliminary experiments; pulse attack for the RG58; measure of the attenuation for the RG58 and Heaviside model determination.	Continue to characterise homogenous cable by doing the same with the RG59.	/
12/11/2019	Characterization of a homogeneous coaxial cable: pulse attack for the RG59; measure of the attenuation for the RG59.	Determine the attenuation coefficient for each homogeneous cable and start the sinusoidal signal attack to determine the transfer function.	E.V. started to model the curves and results of the article on python.

13/11/2019	Modelling the attenuation coefficient curve and sinusoidal attack for each cable.	Determine the transfer function.	/
19/11/2019	Determination of the configuration used in the article; determination of the transfer function of this composite cable.	Measure more values to have a better curve around 10Mhz, 30MHz and 50MHz.	/
20/11/2019	Measurement of more values for the transfer function; Measurement of the phase shift depending on the frequency.	Continue to measure the phase shift and determine the relation dispersion.	The phase seems to be very random; no relation is apparently outstanding.
26/11/2019	Measurement of more phase shifts.	Determine the relation dispersion.	The phase shift is very difficult to measure because it is very variable.
27/11/2019	Determination of a sort of relation dispersion, the result is not very convincing.	Prepare the Gaussian wave packets to generate.	The relation dispersion is not convincing but the project must move forward.
03/12/2019	Preparation of the Gaussian wave packets.	Continue to create Gaussian wave packets with different frequencies and try them on the generator.	/
04/12/2019	Gaussian wave packets creation and test; extraction of the obtained curve in a csv format to calculate the mass centre.	Generate a python program to calculate the mass centre of our scopes; prepare a presentation of the results.	/

10/12/2019	Meeting with the client to present the results. Finalisation of the mass centre calculation; processing of the group velocity results in a graph form.	Measure more values for the transmission coefficient; finalise every result.	/
11/12/2019	Measurement of more transmission coefficients and reviewing of every result.	/	/
17/12/2019	Group meeting; redaction of the plan test document.	Finalise the redaction of every document.	/
10/01/2020	Group meeting; preparation and finalisation of the final oral presentation.	Review the final oral presentation.	/
14/01/2020	Final oral presentation with the client.	/	/

Risks Document

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Risks_Coaxial_Cable.v1
Drafting date: 27/10/2019
Last update: 17/12/2019

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

Risks description	Identification date	WP impacted	Critically	Risk origin	Risk effect	Preventive/ corrective actions	Forecasted finish date	Remarks
Loss of documents and data	27/10/2019	All	Major	Material	Productivity and organisation	Creation of a Google Drive and backup by the group leader	–	–
To run out of time	27/10/2019	All	Moderate	Lack of organization	All the project	Division and assignation of the tasks, keep the Gantt in mind and determine some important deadlines	–	–
Sharing every document in the same format, for Apple or Windows users	12/11/2019	All	Minor	Some group members don't have python and some others don't have Microsoft software.	The results are not comparable and not homogeneou s	Choose to use python for everyone or create a python code that convert every document in the right format	–	E.V. created a python code for the document conversion

Phase shift measurement	13/11/2019	WP3	Moderate	The value displayed by the measuring device is too variable	The dispersion curve is inoperable	Adjustment of scales to decrease variations	–	The problem has not been totally solved
Precision on sampling of Gaussians	03/12/2019	WP4	Moderate	The oscilloscope's sample rate is not high enough	The values of velocity are not precise enough, or even wrong	Use an oscilloscope with the highest sample rate	–	Sample rate used: 200MSa/s
Find a curve fit for the attenuation curve	11/12/2019	WP5	Minor	The attenuation curve doesn't fit with an analytic function	The attenuation curve doesn't have a model	Find the function that better fit with the curve, with as few differences as possible	–	–
Comparison of digital results and the article's results	17/12/2019	All	Major	The results of this project are digitized while the article's results are on the paper	The results are not comparable	Contact the article's writers to know if they still have a digital record of their work, or raise the curves point by hand	–	A. Haché responded to the request

Specification Document

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Specification_Coaxial_Cable.v1
Drafting date: 24/11/2019

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

Group repartition

Name	Nina MOËLLO	Enguerran VIDAL	Maxime THUMIN
Function	Project Manager	Specification Engineer	Test Engineer
Organization	L3 Spé Physique Université Paul Sabatier		

Document historic

Version	Date	Modification performed
1.0	24/11/2019	Document creation
1.1	26/11/2019	Modification after a group meeting

This document allows the client to see that his demands are understood and how it will be processed. It's the translation of the requirements after deep analyse bringing the experience of all the group members together to a better comprehension of the expectations. Questions may be asked to the client during all the project in order to a better understanding.

As the project name reveals, the aim here is to study how can a coaxial cable model a photonic crystal. To achieve the purposes, we decided to divide the work in five packages, all of them having an important characteristic of our cable to show and specified below.

Table of contents

WP1: Study of a homogeneous coaxial cable

A. Determination of the main characteristics of the two different cables

Identification of the main parts of a homogeneous coaxial cable. Discovery of the basic properties, and how it is different from a usual electrical cord.

B. Pulse attack

Determine the propagation speed of a sinusoidal signal sent in the cable. Determine the characteristic impedance of each homogeneous cable and the attenuation.

C. Determination of Heaviside modelling

Compare the behaviour of the cable with a capacity or an RLC system.

WP2: Restore the composite cable transfer function

A. Input and output amplitude measurement

For different frequencies of sinusoidal signals sent in the composite cable, calculate the ratio of the output amplitude over the input amplitude.

B. Determination of the configuration used in the article of reference

There are two ways of connecting the system, starting with the RG58 coaxial cable or the RG59 one. The precedent point may guide us on which configuration is the one Haché and Slimani studied, if there is a difference.

C. Comparison of our measurements with the theory of the article

Trace the evolution of the cable transfer function depending on the frequency and compare it to the results showed in the article.

WP3: Find the link between spatial and temporal frequency in this system

A. Phase shift measurement

Measure the phase shift for different frequencies of a sinusoidal signal sent in the composite cable.

B. Application of the article's equations and relations

Thanks to the equations determined by Haché and Slimani, calculate the evolution of the spatial frequency depending on the temporal frequency.

C. Comparison of our measurements with the theory of the article

Compare the curve with the one showed in the article.

WP4: Study the propagation of a Gaussian wave packet in a quasi-periodic structure

A. Creation of different Gaussian wave packets

The GBF doesn't produce Gaussian wave packets on his own so they have to be created from another file that can be read by the GBF. Moreover, the Gaussian's frequency and the sinusoidal frequency has to be determined before, because they can't be modified with the GBF.

B. Measurement of the packet mass centre and extraction of the group velocity

For each Gaussian wave packet, measure the mass centre of the input signal and the one of the output signal, so the group velocity can be calculated.

C. Study the evolution of the group mass velocity depending on the frequency

For each frequency, study the evolution of the group velocity calculated and compare it to the ones calculated by Haché and Slimani.

WP5: Model the system, with the first study

A. Model the system by assembling matrices

The composite cable is a system made by different homogeneous cables. Those cables can be model by a matrix that gives the transmission and reflexion coefficients. So, the entire system can be model as a matrix. The aim is to determine the coefficients, based on the precedent experiments.

B. Provide the matrix operations to retrieve the experimental results

Thanks to the precedent point and the experiments, the transfer function and the evolution of the spatial frequency depending on the temporal frequency can be model by a matrix calculation.

Conception Document

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Conception_Coaxial_Cable.v1
Drafting date: 24/11/2019

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

Name	Nina MOËLLO	Enguerran VIDAL	Maxime THUMIN
Function	Project Manager	Specification Engineer	Test Engineer
Organization	L3 Spé Physique Université Paul Sabatier		

Version	Date	Modification performed
1.0	24/11/2019	Document creation
1.1	27/11/2019	Modification after a group meeting

This is a document detailing the methods, tools and processes used to achieve the purposes established in the Specification Document (Specification_Coaxial_Cable.v1). It is therefore a document entirely based on the later, thus we took the same shape document to help the lector go through.

Table of contents

WP1: Study of a homogeneous coaxial cable

A. Determination of the main characteristics of the two different cables

Material required:

- A homogeneous coaxial cable with BNC termination;
- A GBF;
- An oscilloscope;
- A homogenous coaxial cable with a banana connector;
- Two “T” connectors.

Objectives:

Identification of the main parts of a homogeneous coaxial cable. Discovery of the basic properties, and how it is different from a usual electrical cord.

Method:

Observe the different parts of the cables and their functions. Put a banana connector on the GBF output and one on the oscilloscope input and point out how a square signal (100kHz) is transmitted. Do the same experiment with BNC terminators and compare. According to the theory of the coaxial cable, the signal should be better transmitted with a BNC terminator, which prevents disturbance from the outside.

B. Pulse attack

Material required:

- A GBF;
- An oscilloscope;
- A RG58 homogeneous cable;
- A RG59 homogeneous cable;
- A variable resistor;
- A multimetre;
- “T” BNC connectors.

Objectives:

Determine the propagation speed of a sinusoidal signal sent in the cable. Determine the characteristic impedance of each homogeneous cable and the attenuation.

Method:

Change the GBF settings to send a fine pulse and a 100kHz frequency of recurrence. Observe the results with a homogeneous cable connected or not to the oscilloscope (do the same with the other homogeneous cable, after that). Thanks to the oscilloscope screen, determine the propagation speed in the cable.

Then, with the variable resistor, determine the exact cable impedance and the linear constants L and C.

Finally, without the variable resistor and by connecting the output cable to the oscilloscope, measure the attenuation coefficient.

C. Determination of Heaviside modelling

Material needed:

- A GBF;
- An oscilloscope;
- A RG58 homogeneous cable;
- A RG59 homogeneous cable;
- “T” BNC connectors.

Objectives:

Compare the behaviour of the cable with a capacity or an RLC system.

Method:

By studying the structure of the elementary quadrupole of the coaxial cable, show that the cable behaves just as a capacity, in a chosen frequency interval. Then, send a square signal (with the frequency adapted) and so determine the value of C and compare with the precedent results.

WP2: Restore the composite cable transfer function

A. Input and output amplitude measurement

Material needed:

- A GBF (50MHz);
- An oscilloscope;
- The composite cable;
- “T” BNC connectors;
- Computer with Excell.

Objectives:

For different frequencies of sinusoidal signals sent in the composite cable, calculate the ratio of the output amplitude over the input amplitude.

Method:

From 0hz to 50MHz, each 0,25MHz, note the value of the input amplitude and the one of the output amplitude. Then, calculate the ratio and plot the graph of this ratio depending on the frequency. Do it for the two configurations (RG58 connected to the GBF and then RG59).

B. Determination of the configuration used in the article of reference

Material needed:

-The results of the precedent study.

Objectives:

There are two ways of connecting the system, starting with the RG58 coaxial cable or the RG59 one. The precedent point may guide us on which configuration is the one Haché and Slimani studied, if there is a difference.

Method:

The reference article doesn’t mention if the searchers connected a RG58 to the GBF and a RG59 to the oscilloscope or inversely. Thanks to the precedent study, determine if there are differences and if so, determine which configurations is the one used in the article.

C. Comparison of our measurements with the theory of the article

Material needed:

-The results of the precedent study.

Objectives:

Trace the evolution of the cable transfer function depending on the frequency and compare it to the results showed in the article.

Method:

Thanks to the precedent points and the article, determine if all results are according. Point the main parts of the curve and note the differences of similarities.

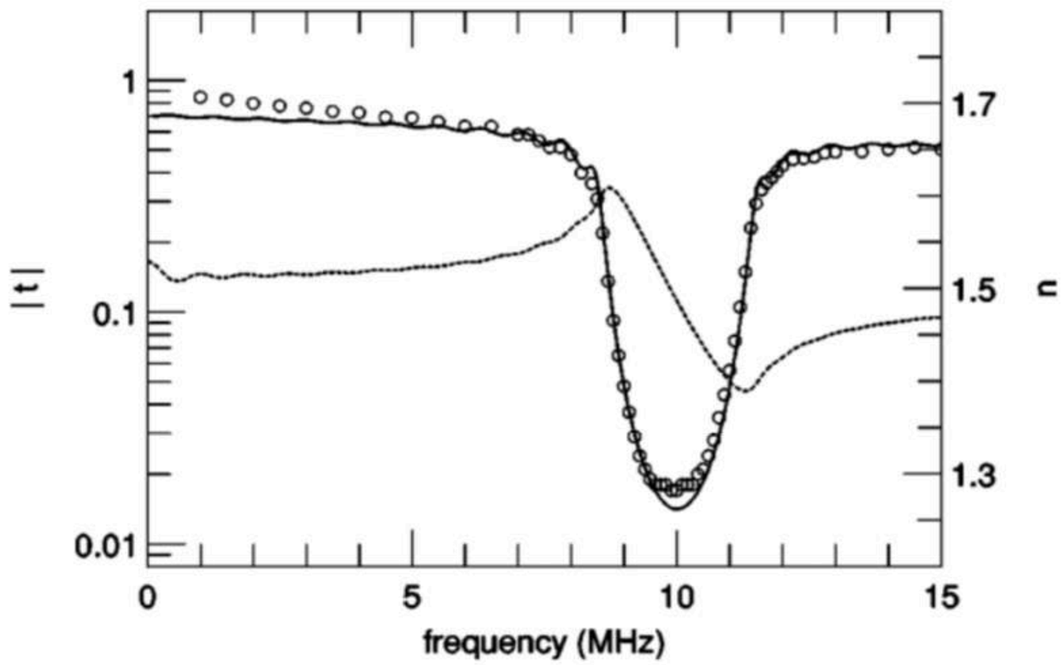


Fig 3: Measured (circles) and calculated (solid curve) transmission through 120m of periodic coaxial cable. The dashed curve is the effective index of refraction extracted from the calculated value of t. [Haché & Slimani, 2004]

WP3: Find the link between spatial and temporal frequency in this system

A. Phase shift measurement

Material needed:

- A GBF (50MHz);
- An oscilloscope;
- The composite cable;
- “T” BNC connectors;
- Computer with Excell.

Objectives:

Measure the phase shift for different frequencies of a sinusoidal signal sent in the composite cable.

Method:

From 0 to 50MHz, note the phase shift in an Excel file.

B. Application of the article’s equations and relations

Material needed:

- The results of the last study;
- A computer with Excel.

Objectives:

Determine the dispersion relation.

Method:

Thanks to the equations determined by Haché and Slimani, calculate the evolution of the spatial frequency depending on the temporal frequency.

C. Comparison of our measurements with the theory of the article

Material needed:

-The results of the last study.

Objectives:

Compare the curve with the one showed in the article.

Method:

Thanks to the precedent points and the article, determine if all results are according. Point the main parts of the curve and note the differences of similarities.

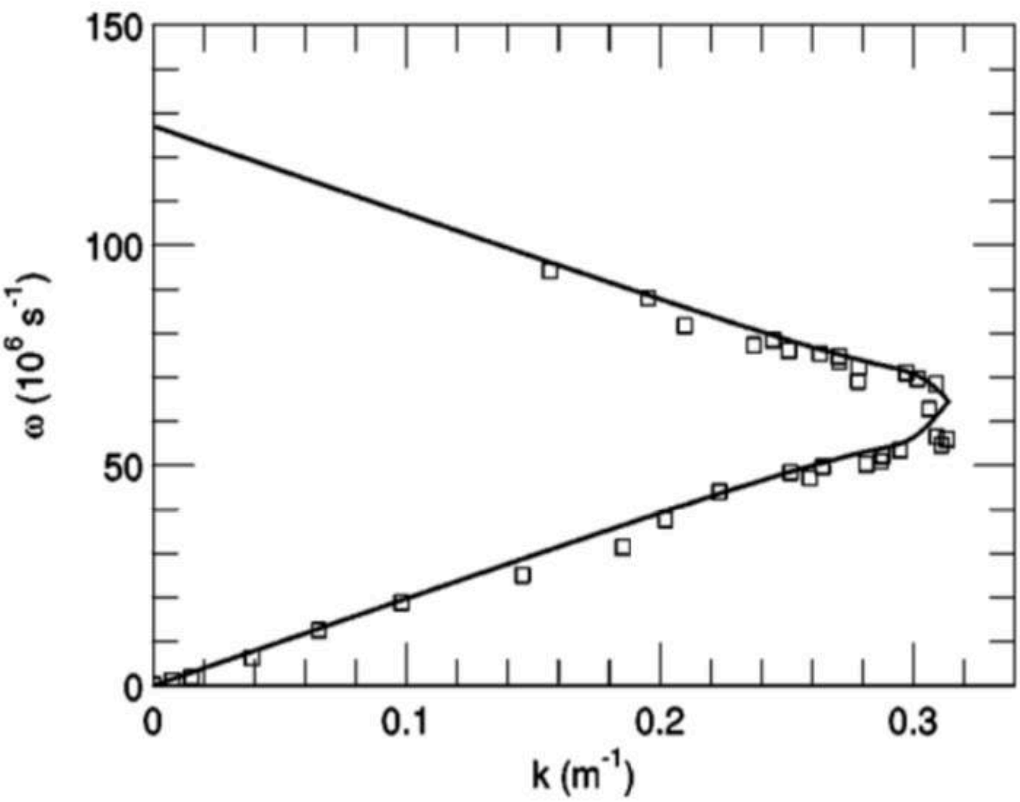


Fig 4: The theorical curve and data points for the dispersion relation and band structure of the 120m coaxial photonic crystal. At the Brillouin zone boundary, the linear relationship between k and ω breaks down. [Haché & Slimani, 2004]

WP4: Study the propagation of a Gaussian wave packet in a quasi-periodic structure

A. Creation of different Gaussian wave packets

Material needed:

- A computer with python;
- A USB device;
- A GBF with USB port.

Objectives:

The GBF doesn't produce Gaussian wave packets on his own so they have to be created from another file that can be read by the GBF. Moreover, the Gaussian's frequency and the sinusoidal frequency has to be determined before, because they can't be modified with the GBF.

Method:

Note the oscilloscope sampling frequency (the higher possible), and the file accepted by the GBF. Then, create lists of points that draw a Gaussian wave packet with a frequency from 5 to 50MHz. Don't forget to determine the Gaussian frequency.

B. Measurement of the packet mass centre and extraction of the group velocity

Material needed:

- A computer with python;
- A USB device;
- A GBF with USB port;
- An oscilloscope;
- The composite cable.

Objectives:

For each Gaussian wave packet, measure the mass centre of the input signal and the one of the output signal, so the group velocity can be calculated.

Method:

Read each file with a Gaussian wave packet on the GBF. It will send the signal on the composite cable (connected) and it will be possible to observe the results on the oscilloscope. Calculate then the mass centre of the output Gaussian wave packet, for each file, and note the masse centre of the input one. Then determine the group velocity.

C. Study the evolution of the group mass velocity depending on the frequency

Material needed:

- The results of the last study.

Objectives:

For each frequency, study the evolution of the group velocity calculated and compare it to the ones calculated by Haché and Slimani.

Method:

Thanks to the precedent points and the article, determine if all results are according. Point the main parts of the curve and note the differences of similarities.

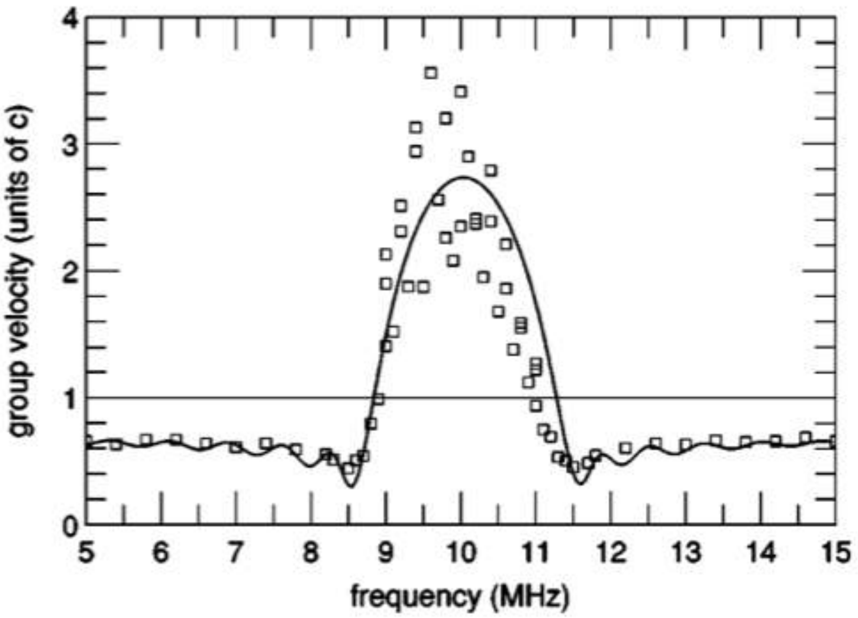


Fig 5: Measured (squares) and calculated (solid curve) group velocity of a pulse in 120 m of coaxial crystal. [Haché & Slimani, 2004]

WP5: Model the system, with the first study

A. Model the system by assembling matrices

Material needed:

- A computer with python;
- The last results.

Objectives:

The composite cable is a system made by different homogeneous cables. Those cables can be model by a matrix that gives the transmission and reflexion coefficients. So, the entire system can be model as a matrix. The aim is to determine the coefficients, based on the precedent experiments.

Method:

Thanks to the precedent experiments, determine the transmission and reflexion coefficients that characterise the composite cable.

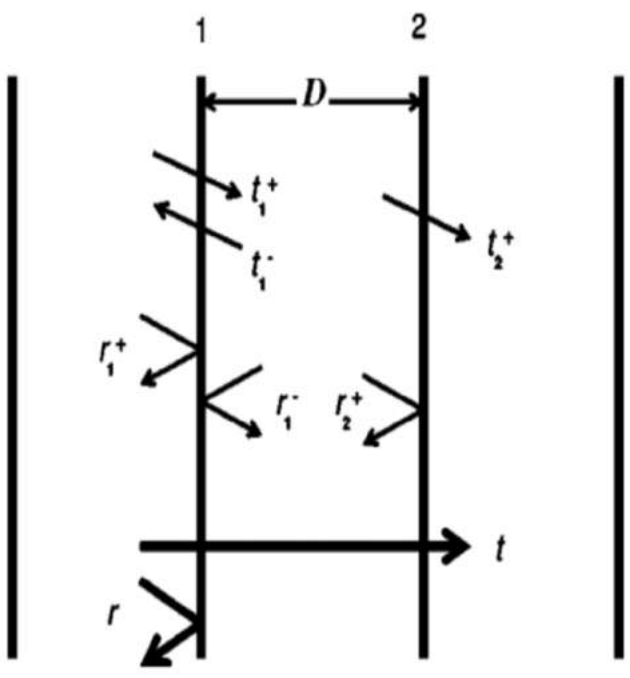


Fig 6: The transmission and reflection through a cable segment located inside a periodic system are calculated from the multilayer theory. The superscripts + and – refer to the wave traveling to the right or to the left, respectively whereas the subscripts 1 and 2 refer to the first or second interfaces. [Haché & Slimani, 2004]

B. Provide the matrix operations to retrieve the experimental results

Material needed:

- A computer with python;
- The last results.

Objectives:

Thanks to the precedent point and the experiments, the transfer function and the evolution of the spatial frequency depending on the temporal frequency can be model by a matrix calculation.

Method:

Use the experiments results to generate expressions that can link all the results theoretically with matrix calculations.

Test Document

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Test_Coaxial_Cable.v1
Drafting date: 04/12/2019

Editor

Name	Maxime THUMIN	Enguerran VIDAL
Function	Test Engineer	Specification Engineer
Organization	L3 Spé Physique Université Paul Sabatier	

Client

Name	M. Hassan SABBAH

Version	Date	Modification performed
1.0	04/12/2019	Document creation
1.1	04/01/2020	Gathering data and layout

Table des matières

WP1: Study of a homogeneous coaxial cable

A. Determination of the main characteristics of the two different cables

- First of all, this part consists in studying the coaxial cable constitution. The theory tells that it leads both the signal and the earth. These two signals must be isolated.

- To compare the coaxial cable capacities to a conventional cable (with two banana plugs), the generator sends a 100kHz frequency square signal, amplitude 2Vpp. The result should be that the coaxial cable carries a signal less disturbed.

B. Pulse attack

- The generator sends very thin pulses, characterized by a 100 kHz frequency, a 150 ns peak width and a 3 Vpp amplitude. Remember that the generator impedance is equal to 50 Ω, the one of the oscilloscope is equal to 1 MΩ and the one of the air is infinite.
  A Bayonet Neill-Concelman (BNC) connector “T” is used to connect a short coaxial cable to the oscilloscope with a 100±1 m coaxial cable that has the output “in the air” (not connected to anything).
  Because of the impedance difference between the environments, some disturbances might be observables at the interfaces.

- The theory tells that the impulsive response observed in a homogeneous cable between two points away from the distance d is$h(t) = e^{-\alpha d}\delta(t - d\sqrt{LC})$. With this formula and the oscilloscope, the propagation speed of a sinusoidal signal sent in the cable might be measurable.

- Using a variable impedance, the aim is here to cancel the reflection peak. The moment that the reflection peak is zero is when the variable impedance reached the cable impedance (there is no more impedance variation). So, Z_c is now determined and knowing that $Z_c \approx \sqrt{L/C}$, L and C are determinable.

- To measure the attenuation, the cable output has to be connected to the oscilloscope (without the variable impedance). By observing the input signal, the attenuation could be measured.

C. Determination of Heaviside modelling

- Heaviside modelled the cable this way:

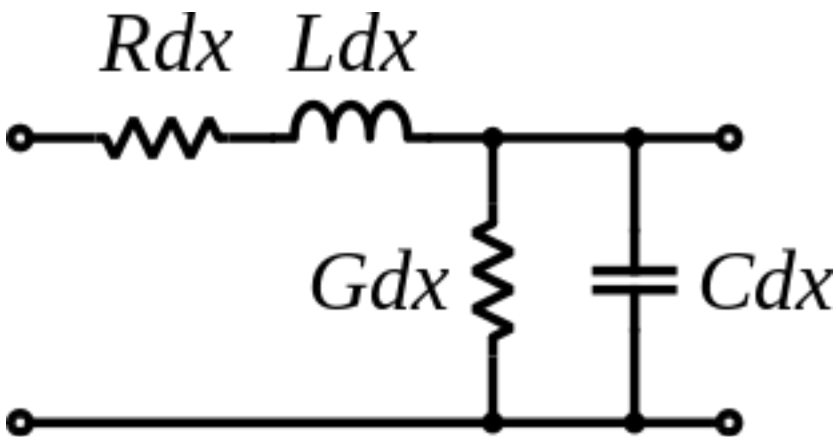


Fig 7: Heaviside model of a dx length coaxial cable. [Wikipédia]

- This circuit attenuation is given by $\alpha = \frac{1}{2}\left(R\sqrt{\frac{C}{L}} + G\sqrt{\frac{L}{C}}\right)$ where C is the capacity, L the inductance, R the resistor and G the conductance. Supposing that G is negligible, the attenuation becomes: $\alpha = \frac{1}{2}\left(R\sqrt{\frac{C}{L}}\right)$. With the precedent results, every constant is calculable.

- For some frequencies, the coaxial cable is supposed to behave as a capacity, when the generator sends a square signal. R must not be infinite so the frequency has to be around 10MHz so L is negligible. The value of R is 1MΩ.

WP2: Restore the composite cable transfer function

A. Input and output amplitude measurement

- From 0 MHz to 15 MHz, and each 0.5 MHz, we sent some sine signal in the composite cable.
- Using the “measure” function of the oscilloscope, we measure the IN and OUT amplitude of the signal.
- Put the data on a python program to calculate the ration OUT over IN.
- Plot the curve Gain in function of the frequency

B. Determination of the configuration used in the article of reference

There are actually two possible configurations for the composite cable:

- The first one starts with a RG-58/U coaxial cable (50Ω)

- The second one starts with a RG-59/U coaxial cable (75Ω)

Fig 8: Schema of the second configuration

To find out which configuration is the good one, we:
- Measure the ration OUT/IN for both configuration
- Plot the transfer function for the two configurations
- Compare to the curve of the reference article.

To gain some time, we only do these measurements for a 0-15 MHz frequency range as to gain time and quickly compare to the article.

C. Comparison of our measurements with the theory of the article

- Find out which configuration is the good one
- Compare our result with the article
- Analyse the transfer function

WP3: Find the link between spatial and temporal frequency in this system

A. Phase shift measurement

- In the same conditions as for the transfer function determination, generating a signal characterized by: 4Vpp amplitude, sinus shaped. And measure the phase shift for frequencies from 0 to 50MHz, with a distance of 5MHz. The phase shift is taken in the unit of seconds.

B. Application of the article's equations and relations

- With the relations given by the article: $\phi_t = \arctan\frac{Im\ t}{Re\ t}$ is the phase sift, $\Delta\phi = \phi_t + m\pi$ where m is a natural number, and finally $\Delta\phi = kd = \frac{n(\omega)\omega}{c}d$ so $k(\omega) = \frac{n(\omega)\omega}{c}$, express the evolution of the temporal frequency depending on the spatial frequency. Moreover, the article tells that, if t is the time delay between two successive peaks or two similarly related points on the wave, then finally $\Delta\phi = t\omega$ (with ω the pulsation).

- Apply all these equations and relations to a python code in order to plot the curve with the point taken before.

C. Comparison of our measurements with the theory of the article

- Plot the experimental points on the same graph as the article's points to compare them.

- If one or two points are not correct (they diverge from the article's result), then the experimental points can be considered as rights. But if there are too many divergences, conclusions must be made to determine what is right or wrong in the article and in the experiment.

WP4: Study the propagation of a Gaussian wave packet in a quasi-periodic structure

A. Creation of different Gaussian wave packets

- The wave generator doesn't produce Gaussian packets by itself. To create it, we use a Python program. This program creates a Gaussian curve, then multiplies it by a sine wave to then apply a rectangular windowing. The packet is centred in the middle of an array which size determines the pattern frequency so that it is repeated 10 000 times per second.

The characteristics of the final signal are:

- Gaussian Amplitude: 10 Vpp (maximum)
- Gaussian frequency pattern: 10 kHz
- Gaussian width: 4 µs
- sine frequency: 0.5 MHz up to 50 MHz

The Gaussian packet that we then have created has its values put in a csv file that can be accepted by the wave generator and lets us then input it into the system.

B. Measurement of the packet mass centre and extraction of the group velocity

In order to calculate the speed of a Gaussian packet as accurate as possible, we do the following process:

- We will first take a save of the oscilloscope screen with both the input packet and the output packet on it
- Calculate the signal centroid (centre of mass) of each of the Gaussian packets
- Determine the difference between the input and output centres of mass, getting a delay between both packets
- Calculate the speed using the following formula

$$v = \frac{system\ lengtht}{\Delta t}$$

To calculate, theoretically, a signal centroid, we usually use the following formula using integrals:

$$T_{CM} = \frac{\int t \ |E(t)| \, dt}{\int |E(t)| \, dt}$$

As our work is experimental, we need to use a discrete and numerical way of calculating integrals in a simple way, for that, we decided to use the trapezes method on the data.

All the steps described above will be done through a Python algorithm which will process the scopes captures from the oscilloscope.
Each resulting speed calculated will then be stored in a separate .txt file.

C. Study the evolution of the group mass velocity depending on the frequency

From 0 MHz to 50 MHz and for each 0.5 MHz:

- Calculate the propagation time of the Gaussian packet and deduce the speed as discussed above
- Plot the following curve: group velocity as a function of frequency.

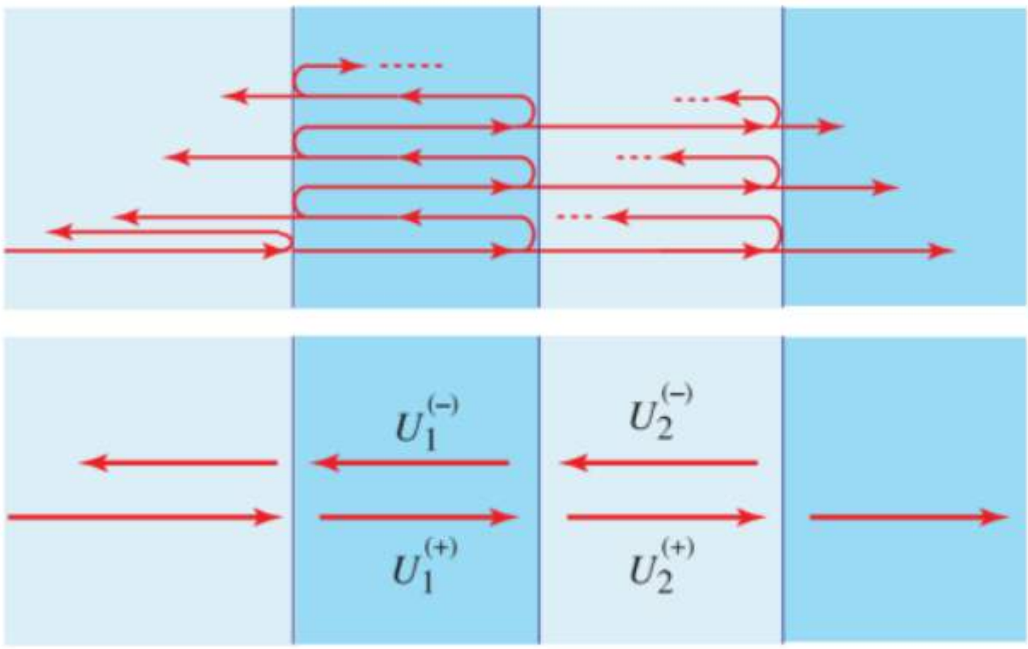
WP5: Model the system, with the first study

A. Model the system by assembling matrices

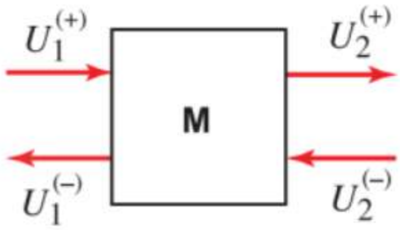
Our coaxial photonic crystal can be best described as a series of dielectric layered media, so we can absolutely apply matrix theory of layered optics so that we could in the long run, understand the theory a bit more and also come up with a theoretical transfer function for our system.

A plane wave normally incident on a layered medium as in our case undergoes reflections and transmissions at the layer boundaries, which also create more reflections and transmissions with a real infinite mess made of complex amplitudes of the reflected and transmitted waves. In theory, the overall transmittance and reflectance can be calculated by summing all those individual waves, but it is quite easy to see the difficulty we will soon encounter when we will be adding multiple layers in our periodic photonic crystal.

In a nutshell, when the number of layers becomes large, tracking the infinite number of micro-reflections and micro-transmissions becomes a nightmare. An alternative approach to this problem is based on the understanding of only two types of waves in our medium: forward waves and backward waves, travelling respectively to the right and left.



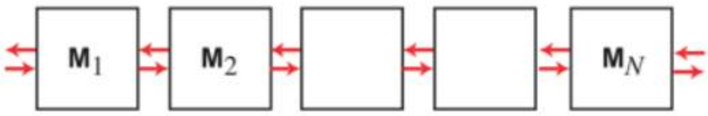
Between two arbitrary planes within a multilayered medium, the 4 resulting waves are noted as such:



$$\begin{bmatrix} U_2^{(+)} \\ U_2^{(-)} \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} U_1^{(+)} \\ U_1^{(-)} \end{bmatrix}.$$

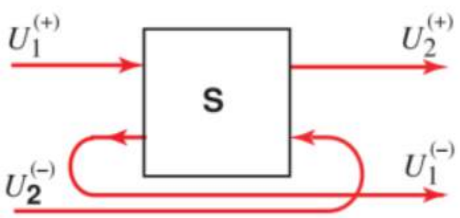
Tracking these 4 resulting waves is facilitated by the use of matrix methods. The two column matrices containing the amplitudes of the 4 resulting waves are related by the matrix equation as seen above. The matrix M, whose elements are marked as A, B, C and D is called the wave-transfer matrix or transmission matrix. Its elements actually depend on the properties of the layered medium between the two planes.

If in the case of a concatenation of basic elements described each by their own transfer matrices, the amplitudes of the forward and backward collected waves are related by a new transfer matrix which is the product of the elements transfer matrices as such:



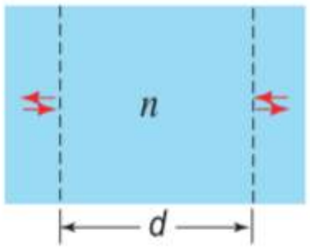
$$\mathbf{M} = \mathbf{M}_N \ldots \mathbf{M}_2 \mathbf{M}_1,$$

We can also define another relation between the different amplitudes of the resulting waves:



$$\begin{bmatrix} U_2^{(+)} \\ U_1^{(-)} \end{bmatrix} = \begin{bmatrix} \mathrm{t}_{12} & \mathrm{r}_{21} \\ \mathrm{r}_{12} & \mathrm{t}_{21} \end{bmatrix} \begin{bmatrix} U_1^{(+)} \\ U_2^{(-)} \end{bmatrix},$$

This helps us define a new two by two-dimensional matrix marked as S, named the scattering matrix. In the case of a propagation through a homogeneous medium, we can find the different matrices to be:



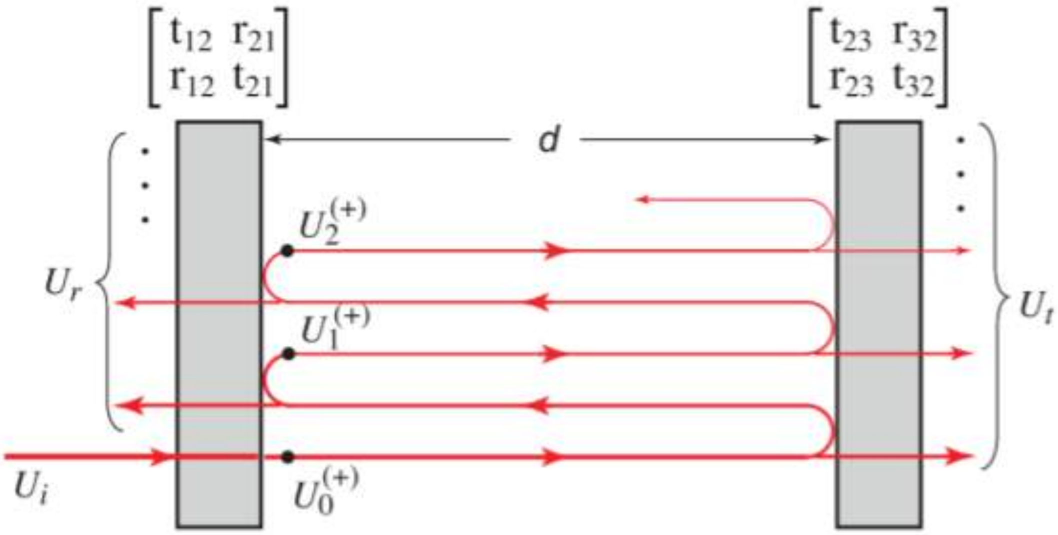
$$\mathbf{M} = \begin{bmatrix} \exp(-j\varphi) & 0 \\ 0 & \exp(j\varphi) \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} \exp(-j\varphi) & 0 \\ 0 & \exp(-j\varphi) \end{bmatrix},$$

Here, the phase is (phi)=n*ko*d where d is the propagation distance and n is the refractive index

The wave-transfer matrix and the scattering matrix are related by manipulating the defining equations of the two. By doing this, the following equations emerge that can help us create a relation between them:

$$\boxed{\begin{aligned} \mathbf{M} &= \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \frac{1}{t_{21}} \begin{bmatrix} t_{12}t_{21} - r_{12}r_{21} & r_{21} \\ -r_{12} & 1 \end{bmatrix}, \\ \mathbf{S} &= \begin{bmatrix} t_{12} & r_{21} \\ r_{12} & t_{21} \end{bmatrix} = \frac{1}{D} \begin{bmatrix} AD - BC & B \\ -C & 1 \end{bmatrix}. \end{aligned}}$$

Consider a wave transmitted through a system described by two S scattering matrices and two M wave-transfer matrices of two cascaded systems. These two systems are mediated by propagation through a homogeneous medium.



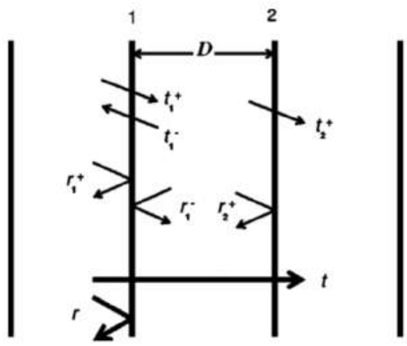
The derived formulas of this system are known as the Airy formulas:

$$\boxed{\mathrm{t}_{13} = \frac{\mathrm{t}_{12}\mathrm{t}_{23}\exp(-j\varphi)}{1-\mathrm{r}_{21}\mathrm{r}_{23}\exp\left(-j2\varphi\right)}, \quad \mathrm{r}_{13} = \mathrm{r}_{12} + \frac{\mathrm{t}_{12}\mathrm{t}_{21}\mathrm{r}_{23}\exp\left(-j2\varphi\right)}{1-\mathrm{r}_{21}\mathrm{r}_{23}\exp\left(-j2\varphi\right)}.}$$

We clearly see a similarity between the Airy formulas and the formulas mentioned in the Alain Haché and Abderrahim Slimani article. These are applications of the preceding formulas to the propagation of an electric plane wave with a linear loss.

$$t = \frac{t_1^+ t_2^+ e^{i\delta-\kappa D}}{1-r_1^- r_2^+ e^{2i\delta-2\kappa D}},$$

$$r = r_1^+ + \frac{t_1^+ t_1^- r_2^+ e^{2i\delta-2\kappa D}}{1-r_1^- r_2^+ e^{2i\delta-2\kappa D}},$$



These equations give us the transmission and reflection for a whole layer. To extract the overall transfer function, we created a Python program which calculates the overall transmission and reflection for the last layer of our composite coaxial cable. Then, this transmission replaces (t2+) and the reflection replaces (r2+) for the previous layer in the structure. With this method, we go from the end of the cable to its beginning, finally getting the overall transmission of the entire composite coaxial cable.

B. Provide the matrix operations to retrieve the experimental results

- About the attenuation:

Before calculating the transmission spectrum by using the previous method, we need to get an accurate expression of the attenuation coefficients (κ) used in the expressions above. These attenuations coefficients were measured for plain cables in a 0-15 MHz frequency range.
The expressions are given for the RG-58/U cable:

$$\kappa(\omega) = -\ln(-1.7\times 10^{-9}\omega + 0.9928)/18.6$$

and for the RG-59/U cable:

$$\kappa(\omega) = -\ln(-1.3\times 10^{-9}\omega + 0.9328)/31.5$$

And so, we cannot simply reuse the expressions of attenuation coefficients as we want to study a 0-50 MHz frequency range. We must then measure them ourselves for both plain cables and try to create an expression that models the data acquired. If the modelling fails, we can simply do a first-degree interpolation of the data (but that could give us a rough estimate and not accurate expressions).

To measure the attenuation at a certain frequency, we connect a 100 meters plain cable (RG-58/U or RG-59/U cable) at a wave generator on one end and an oscilloscope on the other. We then attack the cable with a sinusoidal signal at the frequency selected and then we use the same method that has been described in the researcher's article: we measure Ain and Aout (input and output amplitudes) and then calculate the attenuation via the usual Beer's law of attenuation:

$$A_{\text{out}}/A_{\text{in}} = \exp(-\kappa D)$$

- About the dispersion relation and group velocity:

We need now to extract the wave numbers and group velocities for our frequency range from the complex transmissions we generated by the process described further above.

To accomplish that, we calculate the argument of the transmissions, those become our (φt) as described in the article. Then we just follow diligently the rest of the instructions of the article, although we need some specifications on some steps:

- Concerning the dispersion relations, after folding the curve at the first Brillouin zone like explained, we could need to fold it many times more and we also need to keep the values positive.
- Concerning the group velocities, in the formula described in the article is a derivative term. To model a derivative in the Python program we will set up, we will use the gradient function from the Numpy library.

Results Document

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Results_Coaxial_Cable.v1
Drafting date: 04/12/2019

Editor

Name	Maxime THUMIN	Enguerran VIDAL
Function	Test Engineer	Specification Engineer
Organization	L3 Spé Physique Université Paul Sabatier	

Client

Name	M. Hassan SABBAH

Version	Date	Modification performed
1.0	04/12/2019	Document creation
1.1	04/01/2020	Gathering data and layout

Table des matières

WP1: Study of a homogeneous coaxial cable

A. Determination of the main characteristics of the two different cables

- The advantage of the coaxial cable is that it leads both signal and the earth. But these two signals are isolated from each other by a plastic sheath. The assembly then constitutes a Faraday shield.

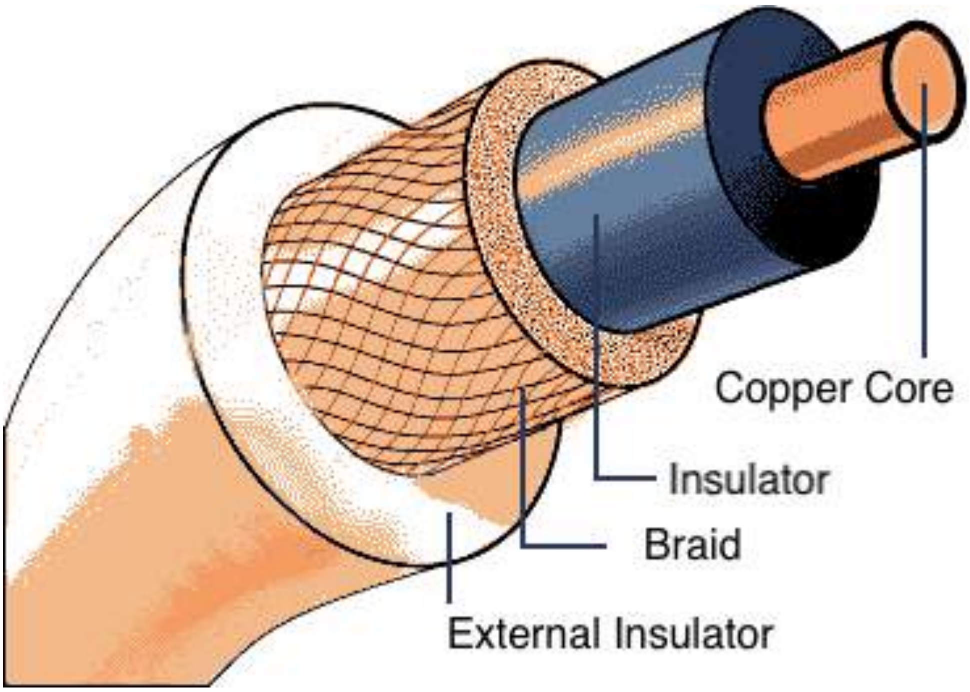


Fig. 9: Pattern of the coaxial cable constitution.

- As shown on the next oscillogram, the signal carried by the conventional cable (2) is noisy at each sudden change while the signal carried by the coaxial cable (1) remains less noisy.

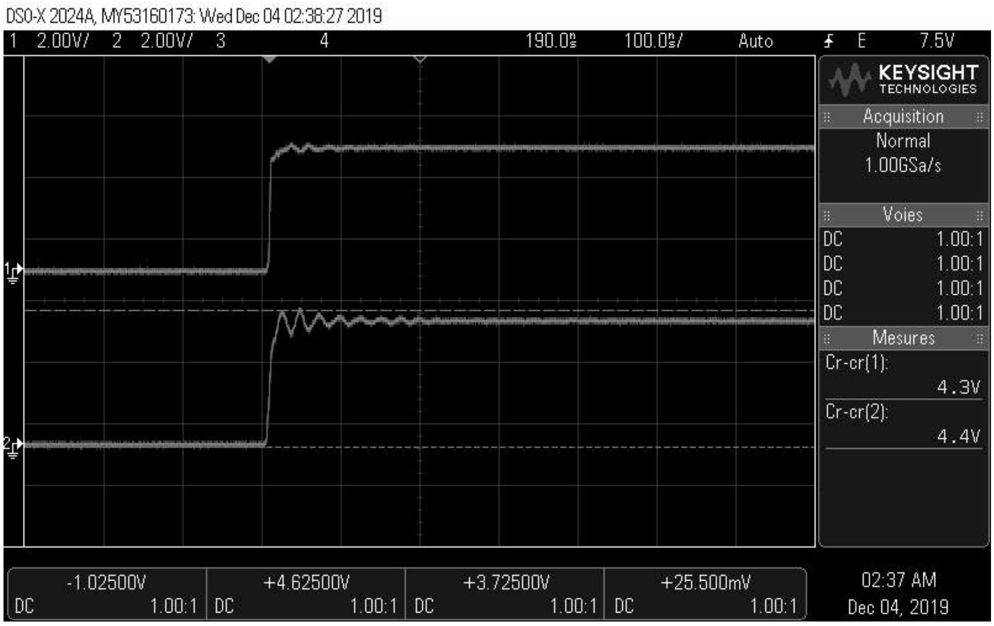


Fig 10: Oscilloscope of a square signal carried by a coaxial cable (1) and by a conventional cable (2).

There is no radiation and the signal transmitted with coaxial cable is unaffected from the outside throughout the cable, while signal 2 suffers slight disturbances at the banana plugs.

B. Pulse attack

- For the 50Ω (RG58) cable, the oscilloscope shows:

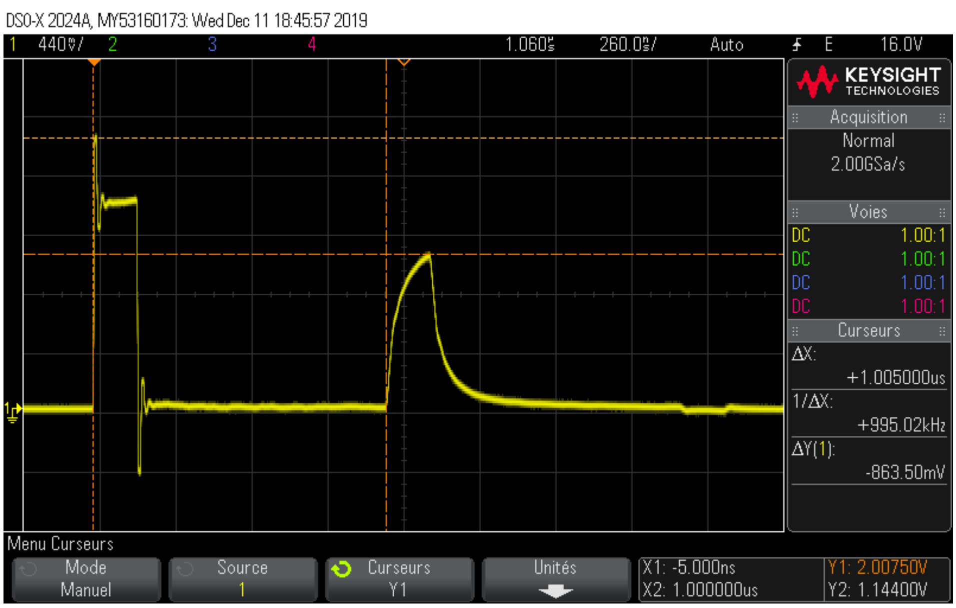


Fig 11: Oscilloscope of a pulse signal carried by a 50 Ω cable, output in the air.

Two peaks are observable while a pulse that generates only one peak; indeed, as the air does not have the same impedance as the cable, and similarly for the oscilloscope, reflections at these interfaces take place. Hence the appearance of several peaks (two significant ones), because the current makes “back and forth”. Actually, one part of the signal passes through the short cable and joins the oscilloscope while the other part passes through the long cable. This signal is reflected at the end of the long cable, returns to the generator and joins the oscilloscope. So, there is a part of the signal that goes twice through the entire distance of the long cable.

A peak is measurable at 2,0 V and a second peak at 1.14 V. The amplitude delivered by the generator seems equal to the sum of these observed peak amplitudes.

- By measuring the time delay between the two peaks and knowing the cable length, it is possible to determine the propagation speed in the cable. The cable is 100±1 m long, but the signal runs twice the length so $\Delta d = 200 \pm 2\ m$. The cursors on oscilloscope show a time delay of 1,0 µs, but the precision of the cursor placement is about 0,05µs.

$$v = {}^{\Delta d}/_{\Delta t} = {}^{200}/_{1.10^{-6}} = 2.10^{8}\ m/s$$

And the measure of uncertainty:

$$\frac{\Delta v}{v} = \sqrt{(\frac{\Delta d}{d})^2 + (\frac{\Delta t}{t})^2} = \sqrt{(\frac{2}{200})^2 + (\frac{0{,}05}{1})^2} = 5{,}1.10^{-2}$$

So $\Delta v = 2.10^{8}.5{,}1.10^{-2} = 1{,}02.10^{7}$

Finally, $v_{50} = 2.10^{8} \pm 1{,}02.10^{7}\ m/s$.

- Cancelling the reflexion with the variable impedance, we observe:

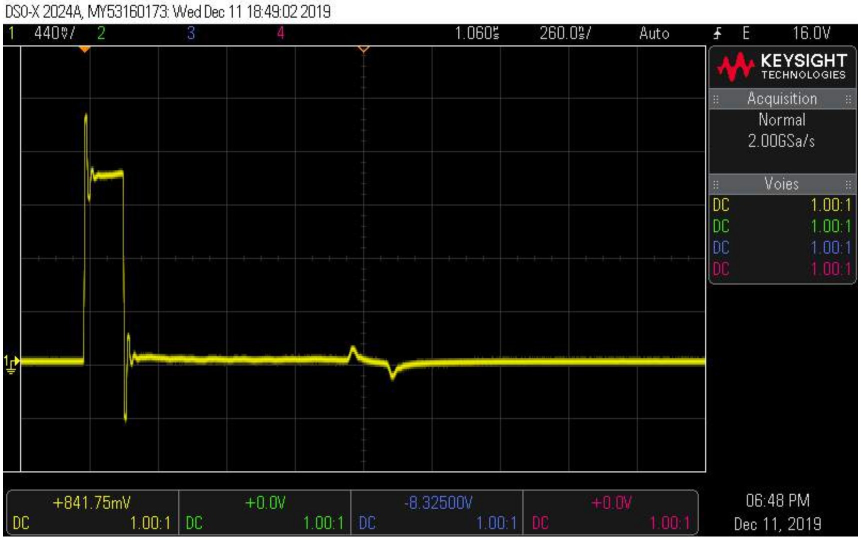


Fig 12: Oscilloscope of a pulse signal carried by a 50 Ω cable connected to a variable impedance.

We note the value of the impedance thanks to a multimetre: R=50Ω ±2Ω. But we know that $Z_c \sim \sqrt{L/C}$=50Ω. So, we measure the capacity with a multimetre C= 9,8 nF. The linear capacity is then $9{,}8.10^{-9}/_{100} = 9{,}8.10^{-11}\ F/m$. Finally. $L = Z_c^2\ C = 2500\ \times\ 9{,}8.10^{-11} = 2{,}45.10^{-7}$ H/m

- Measurement of the attenuation, for the same cable:

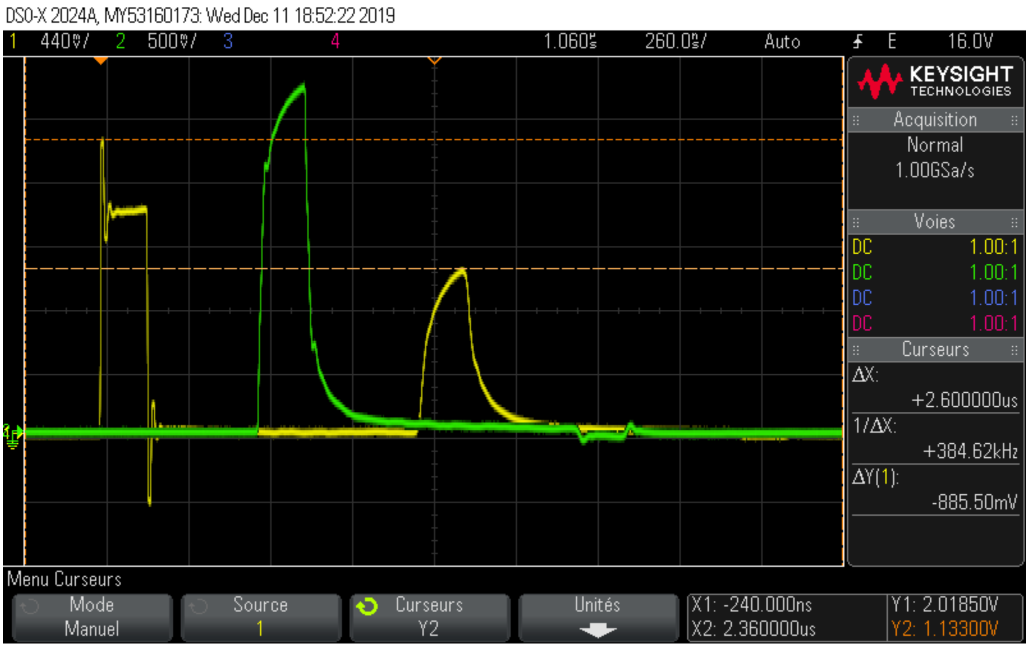


Fig 13: Oscilloscope of a pulse signal carried by a 50 Ω cable connected to the oscilloscope to measure the attenuation.

As showed by the values of Y1 and Y2, there are two peaks at 1,13V and 2,02V. The attenuation is given by:

$$X_{dB} = 20\log\left(\frac{peak\ 1}{peak\ 2}\right) = 20\log\left(\frac{2{,}02}{1{,}13}\right) = 5{,}045\ dB$$

Or

$$X_{\frac{dB}{km}} = {X_{dB}}/{cable\ length} = {5{,}045}/{100.10^{-3}} = 50{,}45\,\frac{dB}{km}$$

And $$\alpha = {ln\left(\frac{peak\ 1}{peak\ 2}\right)}/{100} = \frac{0{,}58}{100} = 5{,}8.10^{-3}\ \text{neper/m}$$

- We did the exactly same experiments for the 75 Ω cable (RG59):

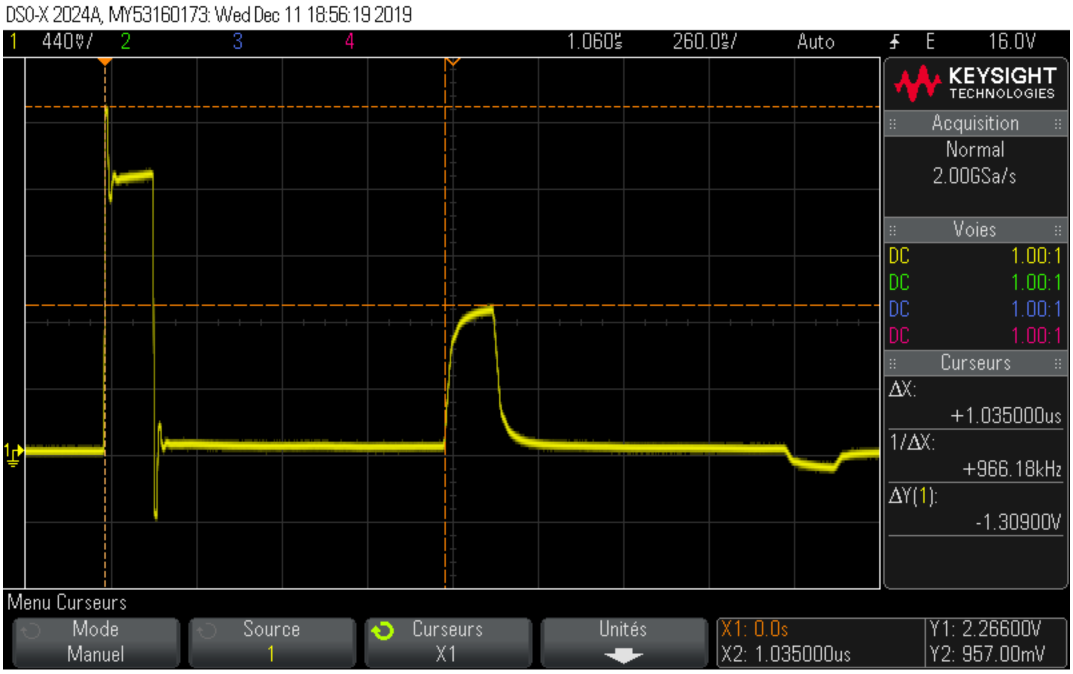


Fig 14: Oscilloscope of a pulse signal carried by a 75 Ω cable, output in the air.

The same reactions are observable. A peak is measurable at 2,27 V and a second peak at 0,96 V. The time delay between these two peaks is 1,035µs, but the precision of the cursor placement is about 0,05µs. Finally, $v_{75} = 1{,}93.10^{8} \pm 9{,}51.10^{6}\ m/s$.

- Cancelling the reflexion with the variable impedance, we observe:

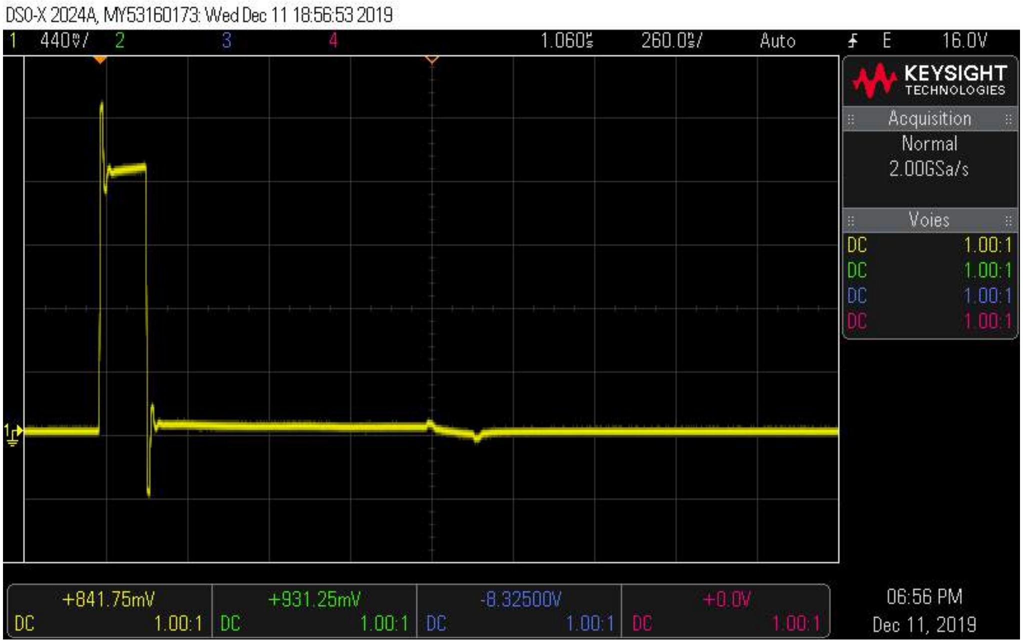


Fig 15: Oscilloscope of a pulse signal carried by a 75 Ω cable connected to a variable impedance.

We note value: R=73,6Ω ±2Ω. And with the same calculations as for the RG58: C = 6,906 e-11 F/m and L = 3,741 e-7 H/m.

- Measurement of the attenuation:

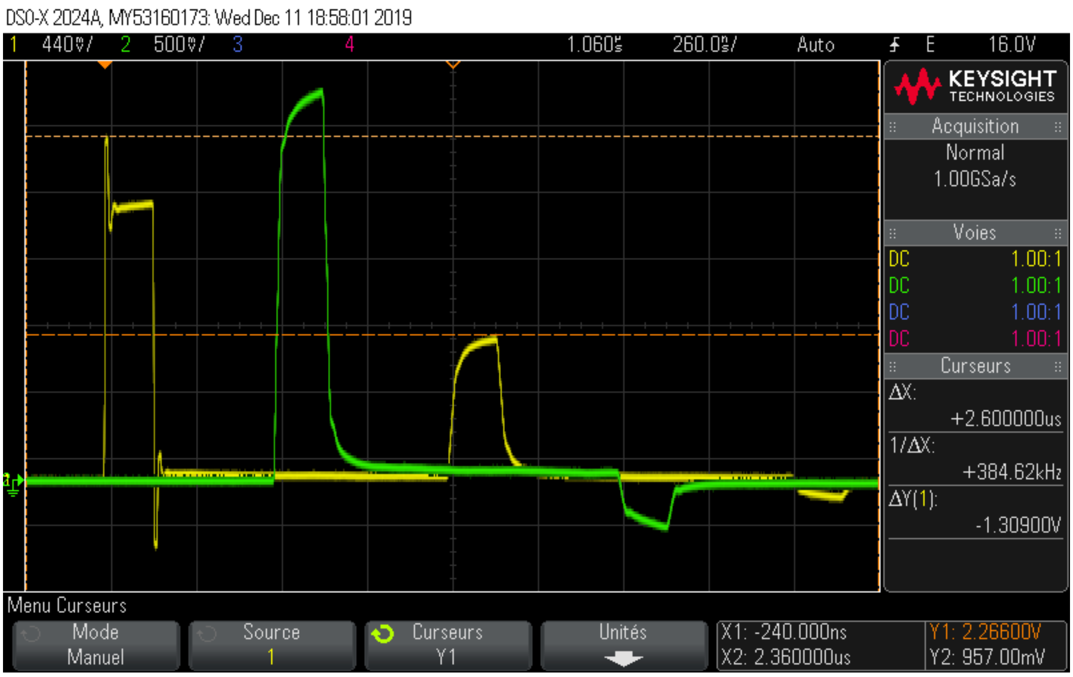


Fig 16: Oscilloscope of a pulse signal carried by a 75 Ω cable connected to the oscilloscope to measure the attenuation.

The peaks are at 2,27V and 0,96V so $X_{\frac{dB}{km}} = 74{,}75\frac{dB}{km}$ and $\alpha = 8{,}61.10^{-3}$ neper/m

- This table summarizes previous results

Quantity to be measured	50Ω cable	75Ω cable
V (m/s)	$2.10^{8}\ \pm 1{,}02.10^{7}$	$1{,}93.10^{8}\ \pm 9{,}51.10^{6}$
R (Ω)	50 ±2	73,6±2
C (F/m)	$9{,}8.10^{-11}$	6,906 e-11
L (H/m)	$2{,}45.10^{-7}$	3,741 e-7
X (dB/km)	50,45	74,75
α (neper/m)	$5{,}8.10^{-3}$	$8{,}61.10^{-3}$

C. Determination of Heaviside modelling

- When the generator sends a 9MHz frequency square signal, the oscilloscope shows a capacity behavior. The capacity charges and discharges, as we seen on the figure. But if the frequency is too high, the capacity doesn't have the time to discharge and recharge.

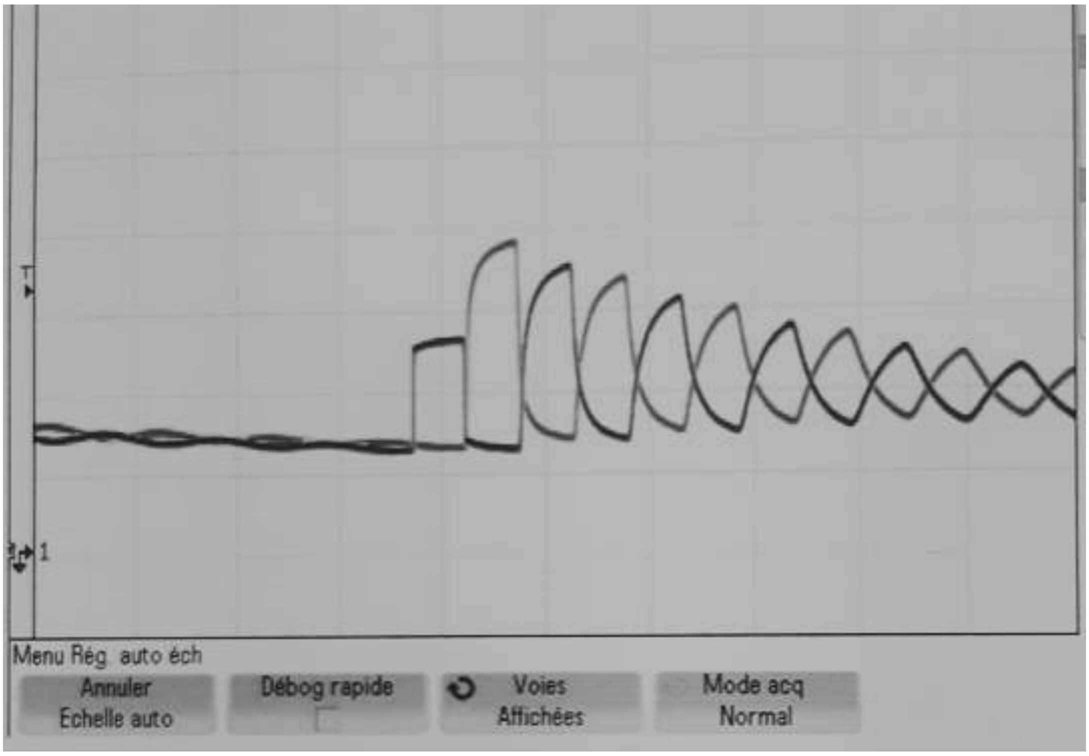


Fig 17: The cable behaves as a capacity which charges and discharges.

- With this scope, the value of τ represent the time spent to go from 0 to 67% of the value of C and 2,2τ represent the time to go from 10 to 90% of the value of C. Moreover τ=RC and R is equal to 1MΩ in this experiment. We measure $\tau = 5{,}5.\,10^{-3}$s. So $C = {}^{\tau}/_{R} = 5{,}5\ nF$. The value of C is different from the one we calculated before because we didn't consider the internal resistor of the generator and the one of the oscilloscope. We observe the same results for both cables.

WP2: Restore the composite cable transfer function

A. Input and output amplitude measurement

From 0 MHz to 15 MHz, and each 0.5 MHz, we sent some sine signal in the composite cable. The following picture is a screenshot of the oscilloscope. We can see a 2.63 Vpp sine signal in yellow (IN) and a 0.26Vpp sine signal in green (OUT).

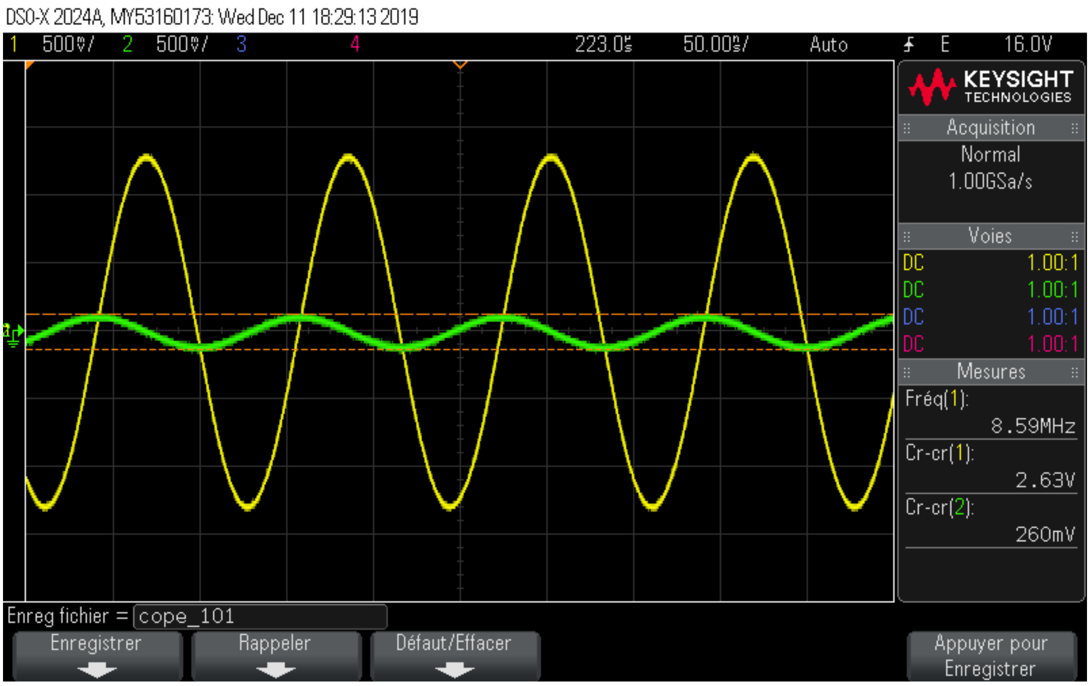


Fig 18: 9 MHz sine signal IN (yellow) & OUT (green)

Thanks to the measure function of the oscilloscope we measure the IN and OUT amplitude for each frequency and we plot the OUT/IN ratio in function of frequency curve.

B. Determination of the configuration used in the article of reference

In order to find the configuration used in the article, we plot the transfer function of the two configurations. The following curve are:

- Transfer function of the article:

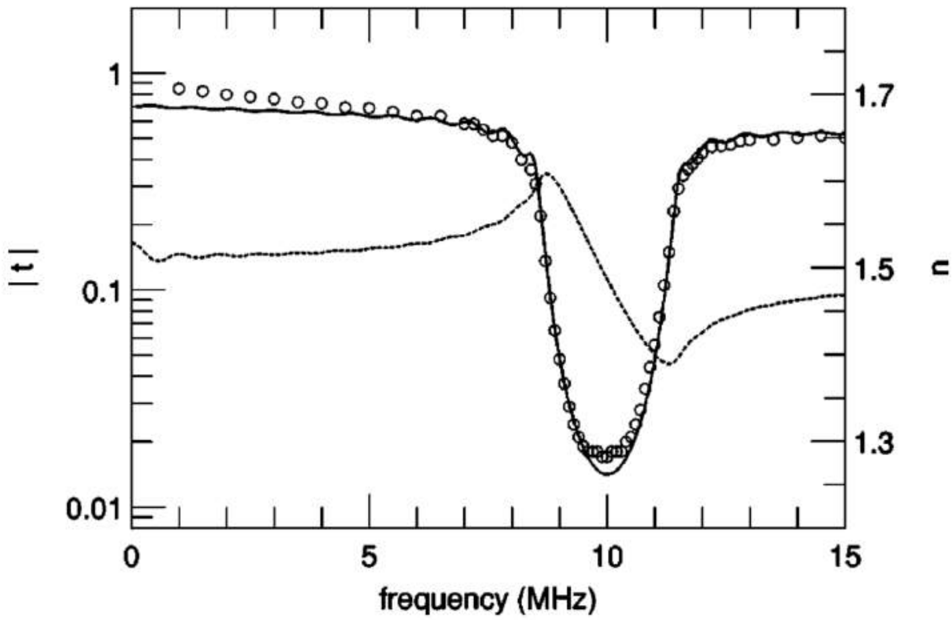


Fig 19: Transfer function from the article

As we can see, the function is constant with a well around 10 MHz. We will call this well a band gap.

- Transfer function of the first configuration:

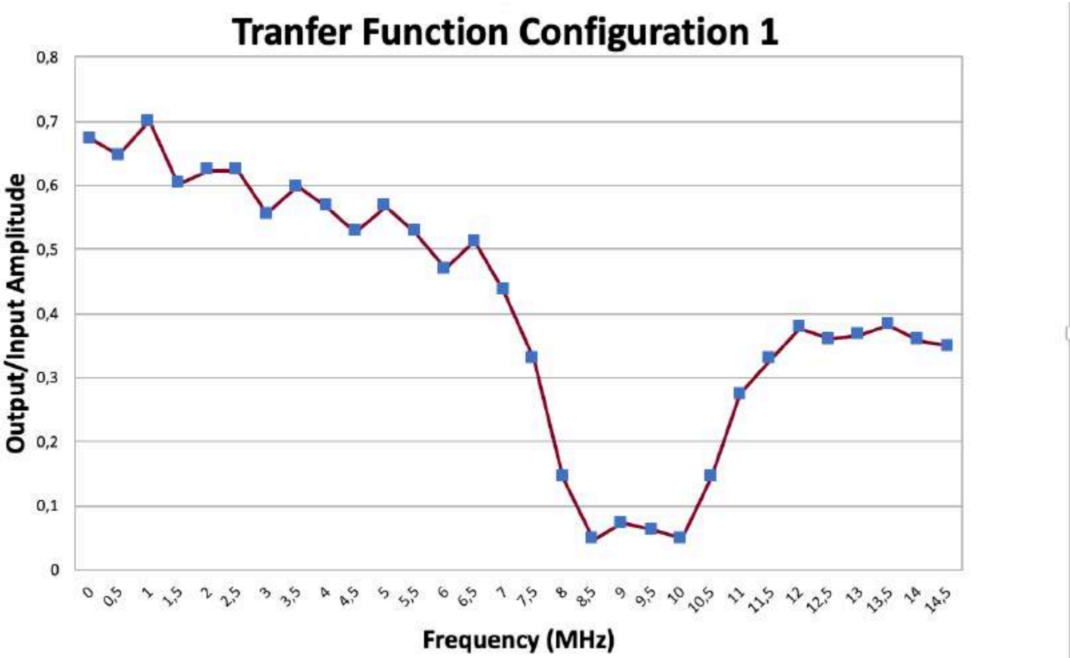


Fig 20: Transfer function of the 1st configuration (from 0 MHz up to 15 Mhz)

As we can see, the band gap is around 9 MHz and the curve has the same behaviour than the article one.

- Transfer function of the second configuration:

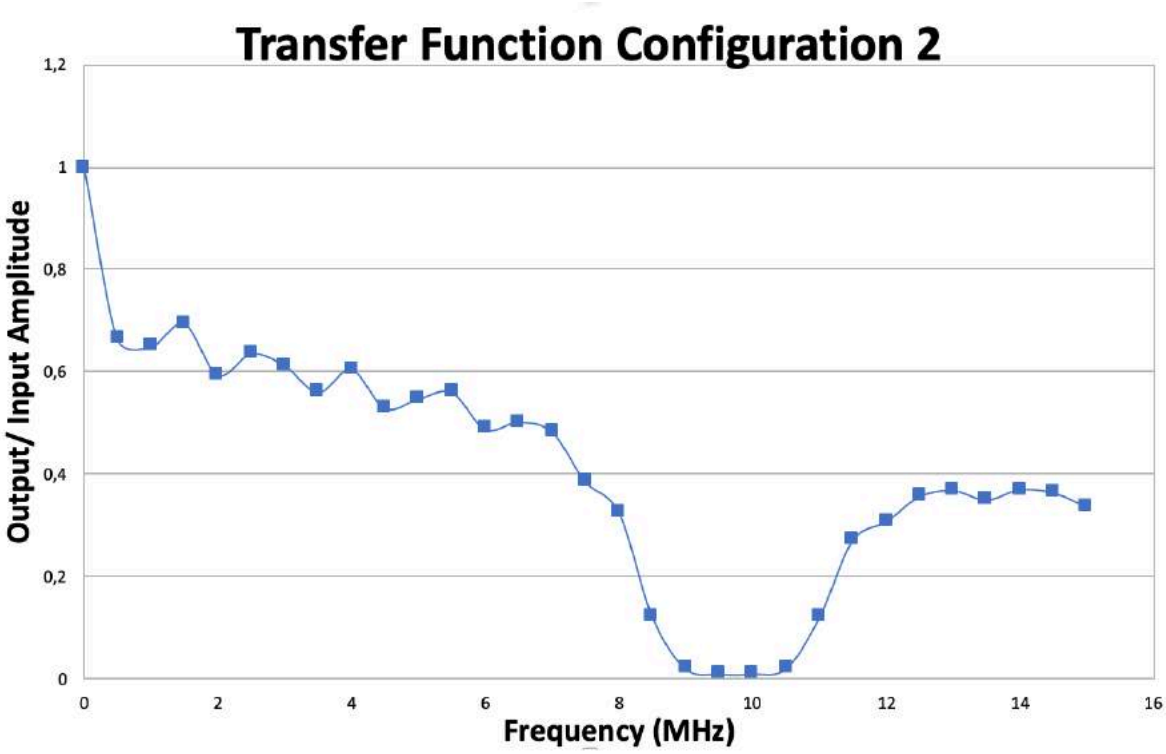


Fig 21: Transfer function of the second configuration (from 0 MHz up to 15 Mhz)

As we can see, the band gap is around 10 MHz and the curve has the same behaviour than the one in the article.

Conclusion: the second configuration seems to be the one used in the article.

C. Comparison of our measurements with the theory of the article

We found the configuration use in the article, so, we can start the physic analysis of the curve.

- The curve looks like the transfer function of a notch filter. Which means:
Every sine signal with a frequency between 0 MHz and 8 MHz or 12 MHz and 15 MHz will get through the filter without any attenuation.

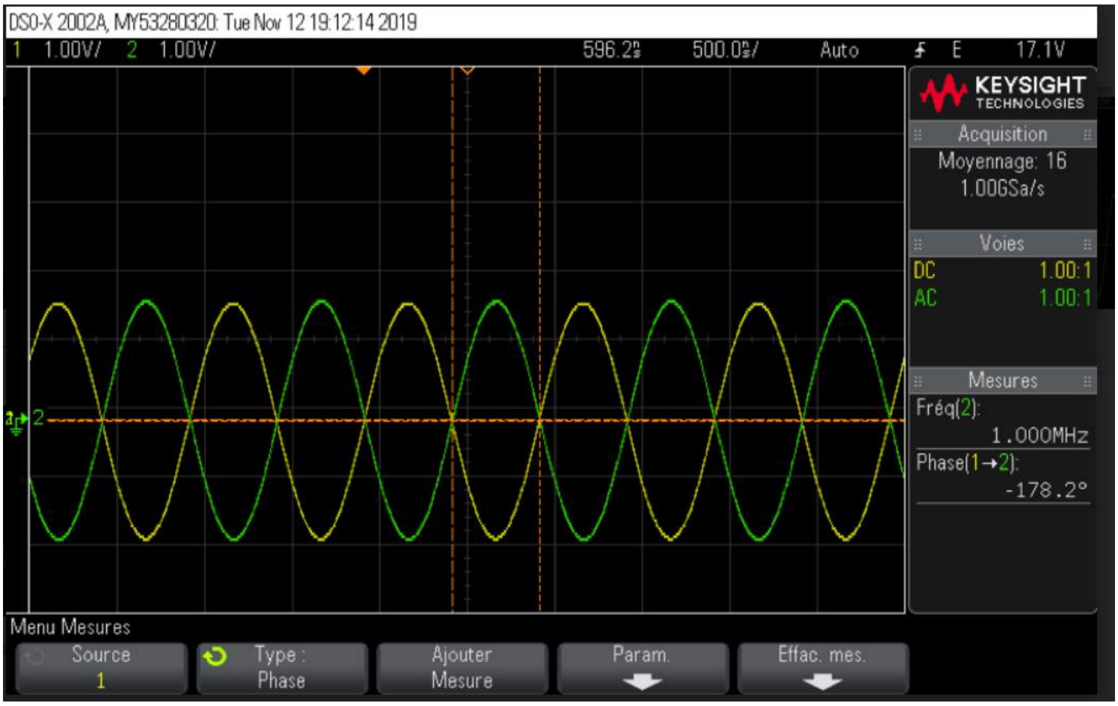


Fig 22: 1 MHz sine signal IN (yellow) & OUT (green)

For example, this is a 1 MHz sine signal: IN (yellow) and OUT (green). The amplitude is approximately the same.

In the other and, every sine signal with a frequency between 8 MHz and 12 MHz will be really attenuate. This also means that for this frequency range, the electrical interferences in the cable are such that the electrical wave doesn't reach the oscilloscope.

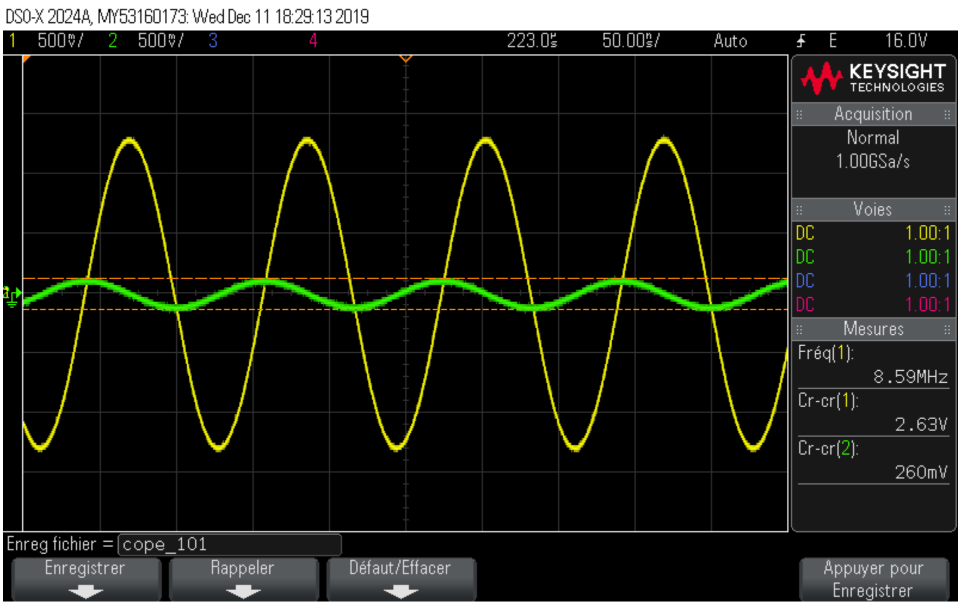


Fig 23: 9 MHz sine signal IN (yellow) & OUT (green)

The IN signal (yellow) is a 8.6 MHz sine with 2.63 Vpp of amplitude. The OUT signal is a 8.6 MHz sine signal but the amplitude is only 0.26 Vpp.

In order to understand the behaviour of the composite cable for a bigger frequency range, we continue our measure up to 50 MHz.

The following curve is the transfer function of the composite cable. We also plot the article data.

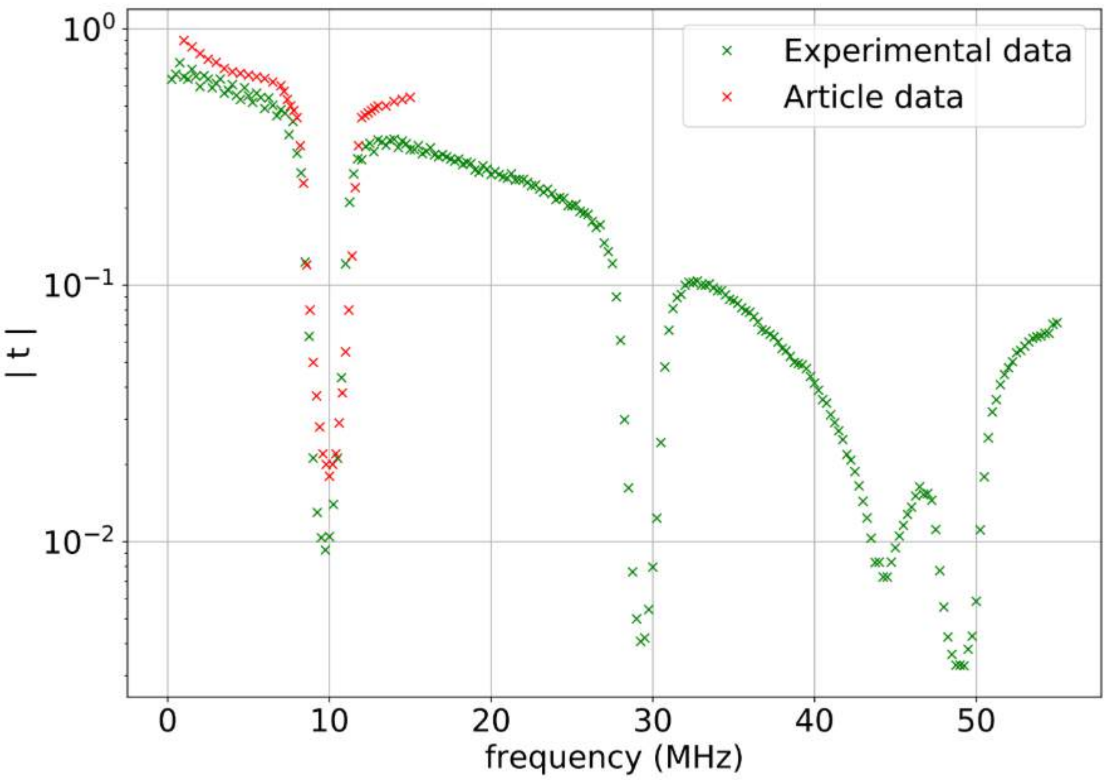


Fig 24: Transfer function of the composite cable (from 0 MHz up to 50 Mhz)

As we can see, the composite cable is a kind of a multi notch filter which is related to the periodic structure of the system. We can also note that our data fits quite good with the article data.

WP3: Find the link between spatial and temporal frequency in this system

A. Phase shift measurement

- The phase shift measurement wasn't like we expected it. Actually, the oscilloscope wasn't stable so some values oscillated a lot. We decided to take the mean of these values.
- Moreover, the points were not going as we expected. There was no relation, like if all these values were random.

B. Application of the article's equations and relations

- Even if we apply our equations to these values, any relation is remarkable.
- The points are still confusing.

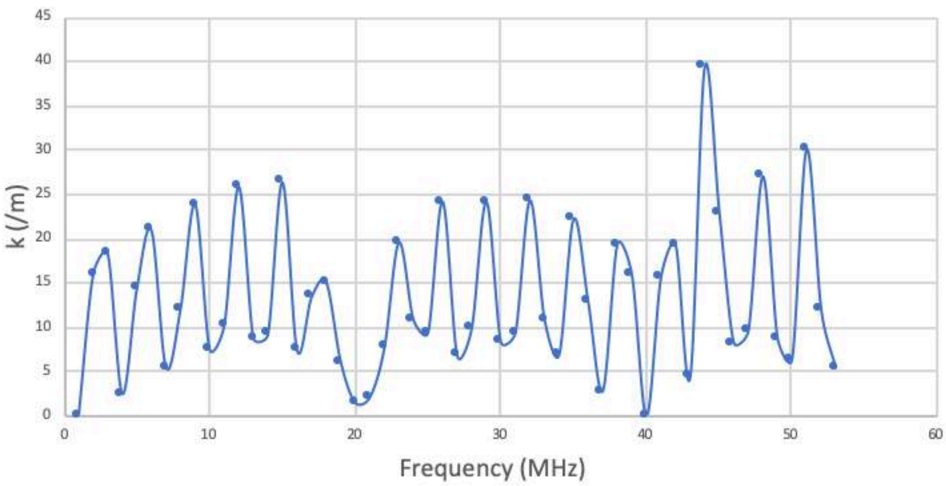


Fig 25: Our experimental dispersion relation

C. Comparison of our measurements with the theory of the article

- Our points were not comparable to the article's ones. We decided to make again the step A (take the phase shift values), but the result was every time the same.
- Maybe our method is not the right one, or the measurement material is not adapted. But we didn't have the time to test something else.

WP4: Study the propagation of a Gaussian wave packet in a quasi-periodic structure

A. Creation of different Gaussian wave packets

The Gaussian packet that we use is a periodic signal where the pattern is a sine function with the shape of a Gaussian curve. Another important feature is the frequency spectrum of the gaussian packet. It is a gaussian curve centred on the sine frequency.

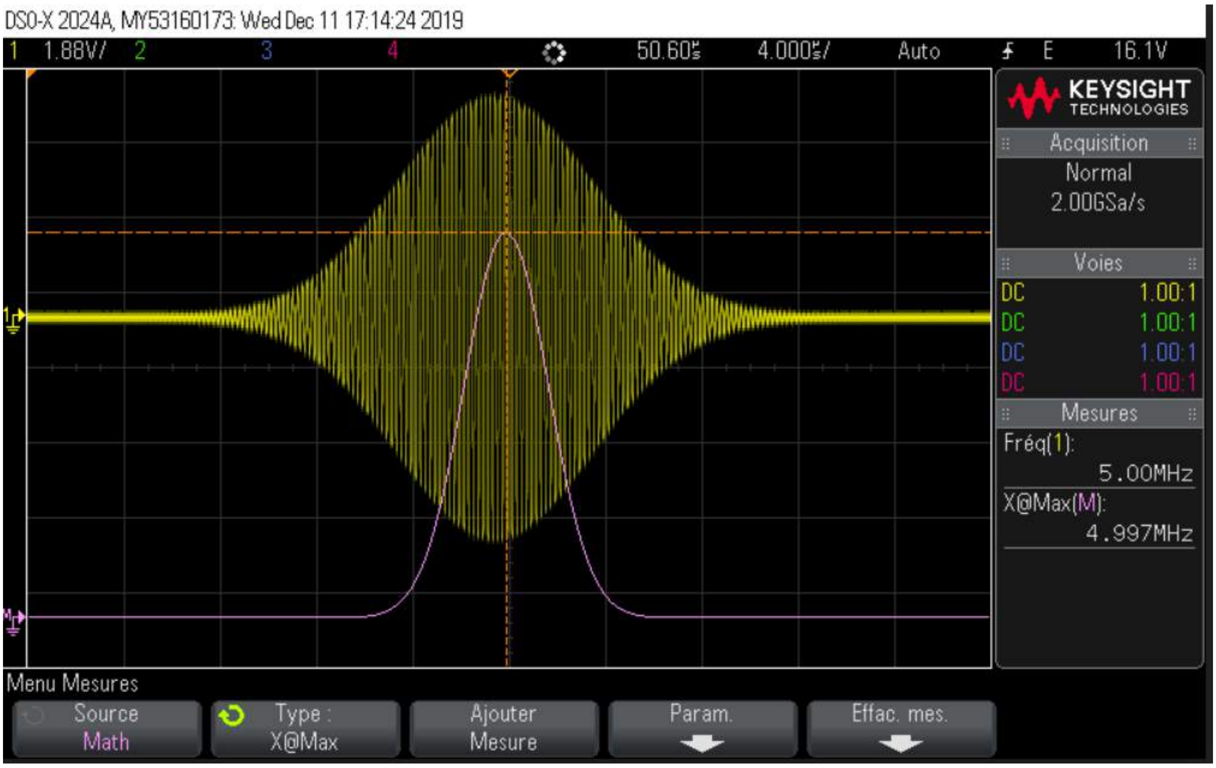


Fig 26: 5 MHz gaussian packet & frequency spectrum (purple curve)

B. Measurement of the packet mass centre and extraction of the group velocity

For the measure of the speed, we sent gaussian packet and extract the data.

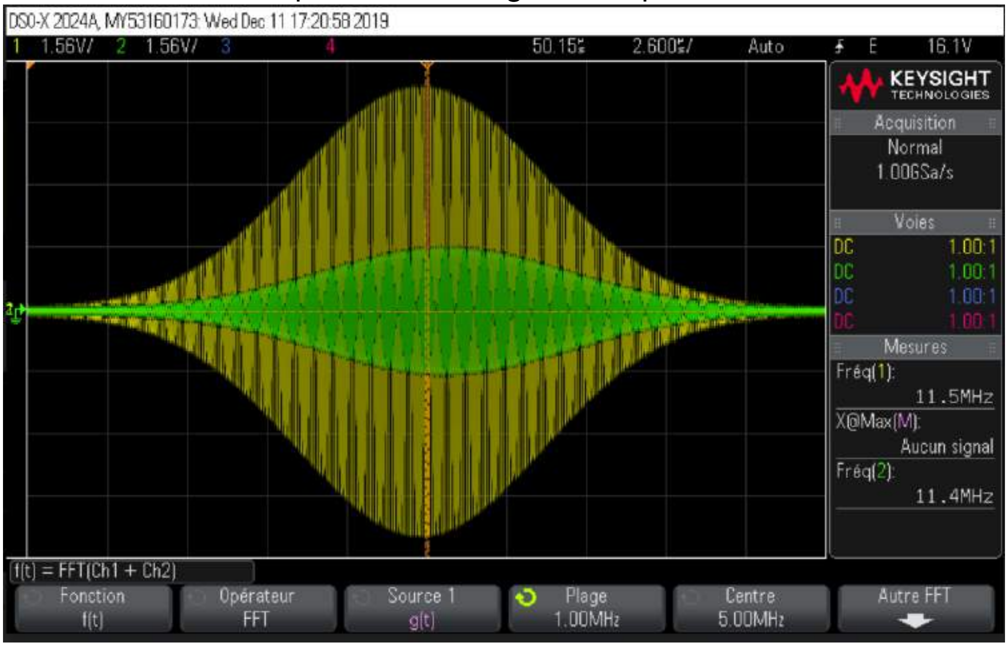


Fig 27: 11 MHz Gaussian Packet IN (yellow) & OUT (green)

To extract the information of speed, we calculate the centre of mass of each gaussian packet.

This screenshot represents a 15 MHz gaussian packet. For each signal, we calculate the centre of mass. They are represented on the curve by point.

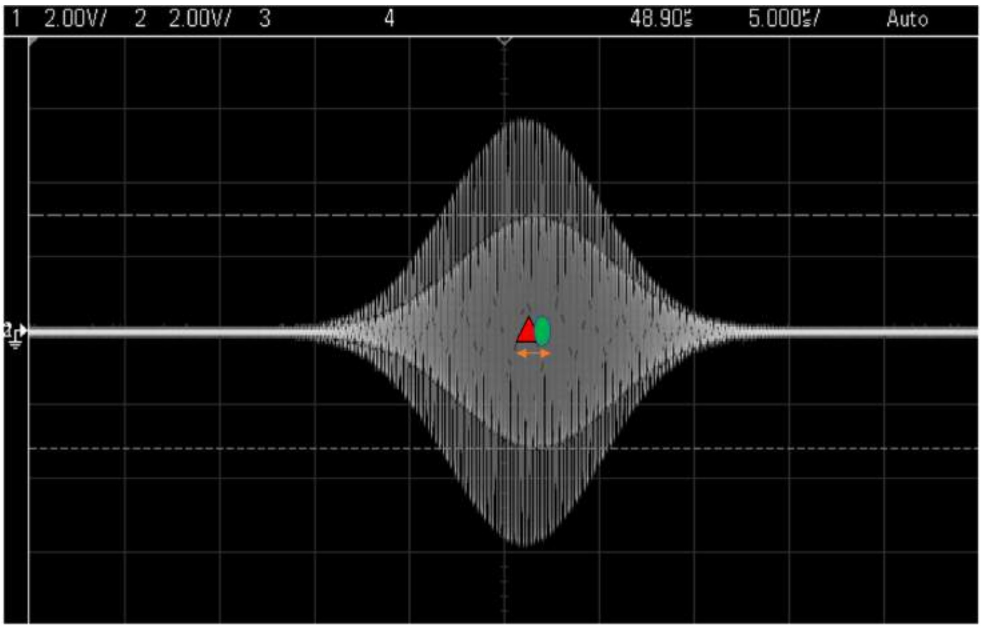


Fig 28: Gaussian Packets with their centre of mass

As we can see, there is un time delay between both centre of mass. We calculate the speed for each frequency and plot the curve.

C. Study the evolution of the group mass velocity depending on the frequency

In this part, we will analyse the group velocity curve.

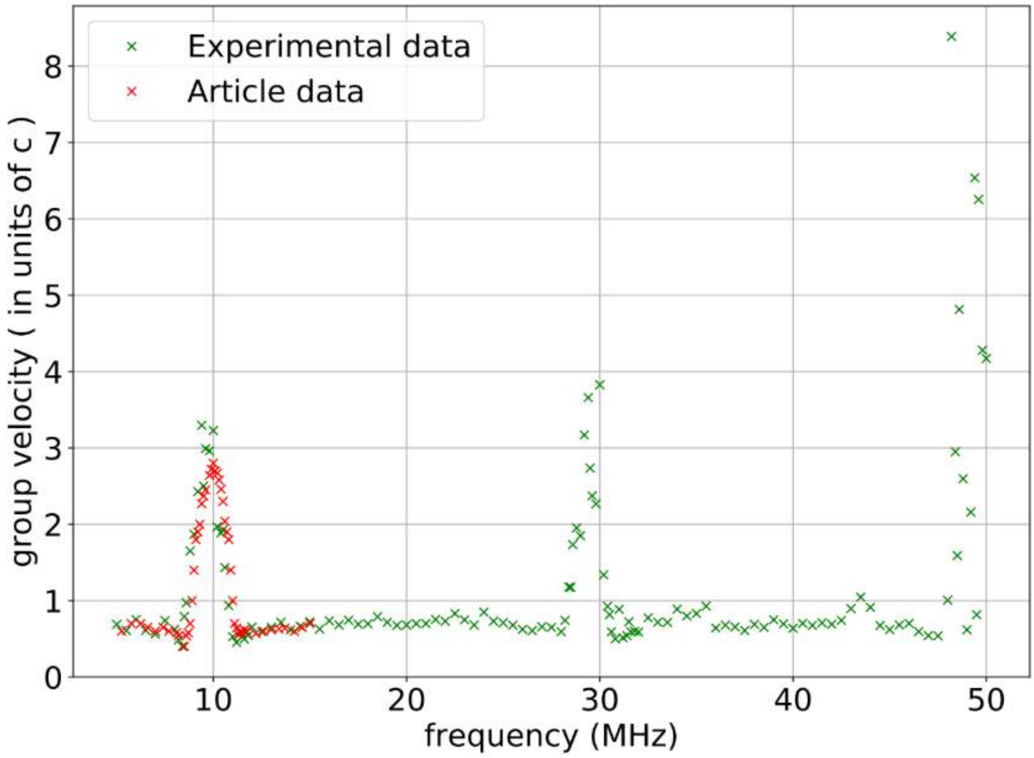


Fig 29: Group Velocity in function of frequency curve

As we can see;

- Our data fits really good with the article data
- For each band gap, the group velocity is greater than the speed of lights

It seems to be a violation of Einstein Special Relativity but it is not. Actually, the speed of the information is less than the speed of light. To explain this phenomenon, we will use an analogy with quantum mechanics.

For each band gap, the transmission is a minimum. Which means the electric light is very attenuated.

This is really similar to a quantum matter mater wave going through a potential region higher than the energy of the wave by quantum tunnelling effect. When you calculate the wave function beyond the potential barrier it seems that the particle is going faster than the light. This is almost the same here with the following analogy:

IN signal	⇔	Wave function
OUT signal	⇔	Wave function beyond the potential barrier
Band gap	⇔	Potential greater than the energy of the particle

The same “periodical” behaviour is also here. Each 20MHz another band gap and another supraluminical effect.

WP5: Model the system, with the first study

A. Model the system by assembling matrices

Like specified in the test document, the researchers of the source article measured the attenuation coefficients in a 0-15 MHz frequency range as to get the most accurate coefficients possible. We decided to do the same on our 0-50 MHz frequency range to then create the theoretical curves using the methods described in the test document.

For the RG-58/U cable, the attenuation coefficients were measured and the monotony of the data meant we could easily apply a polynomial to do a curve fit with the data, the data and the fitted curve are observable below:

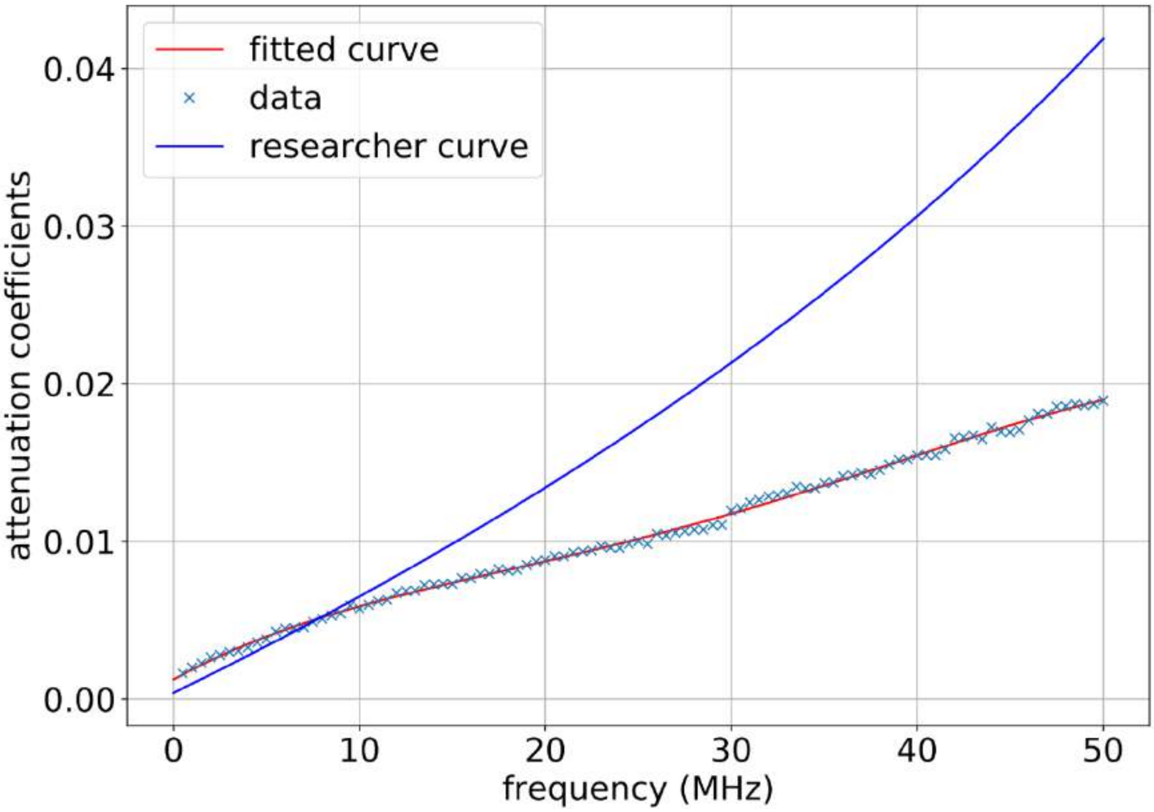


Fig 30: Attenuation depending on the frequency for the RG-58U

For the RG-59/U cable, things were not as simple. The measurements showed a big peak of attenuation around the frequencies of 42-44 MHz that was first thought to be an anomaly resulting from an error of protocol, but as the measures of transmissions through our composite coaxial cable showed, we also observe a gap in transmission around the same frequency range.

However, this non-monotony of the coefficients perturbed our curve fit, rendering it unable to complete, so, as mentioned in the test documents, we figured a first-degree interpolation of the data could make a good enough substitute. The data and resulting curves can be seen below:

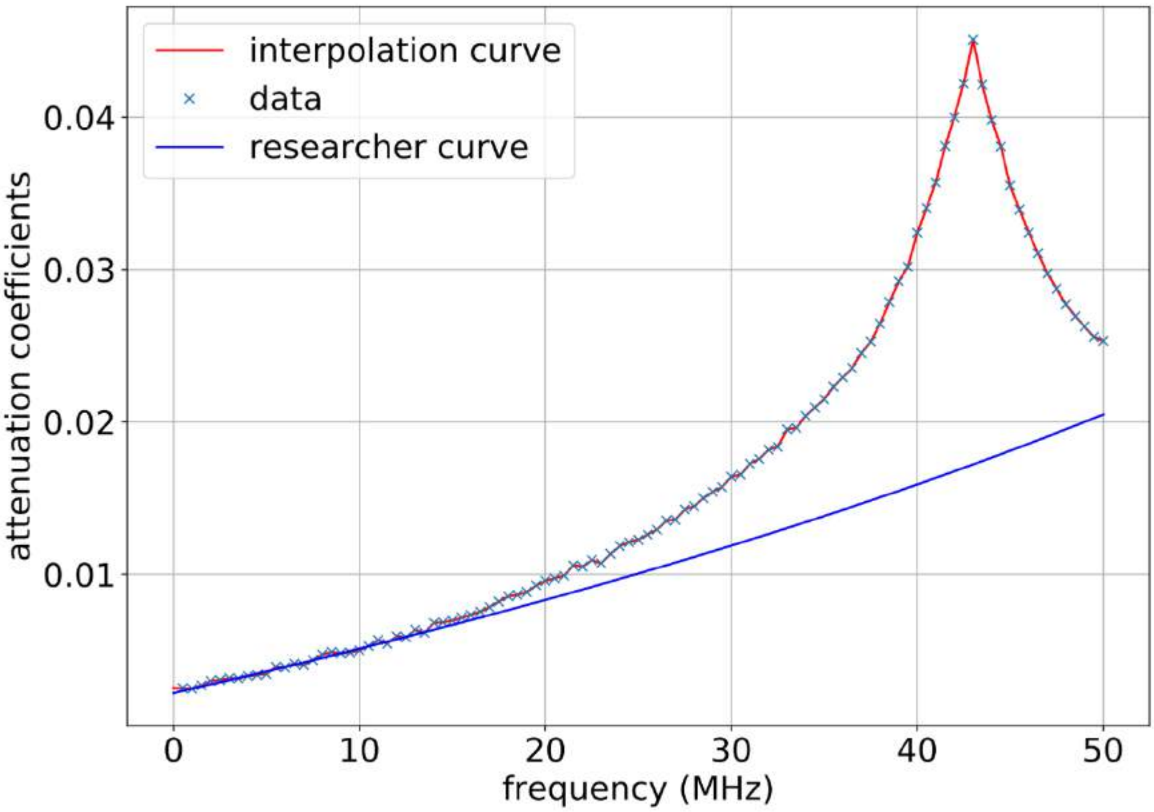


Fig 31: Attenuation depending on the frequency for the RG-59U

We then have been able to use the expression or direct values of the created curves to model our attenuation coefficients through our composite coaxial cable model.

B. Provide the matrix operations to retrieve the experimental results

We created a Python program using the different formulas mentioned in the article. This program is done with an object-oriented approach to the problem, using two Python « classes », creating two objects : one representing an homogeneous medium with its properties, the other representing the overall composite cable model, letting us specify the order of the different cables types and the properties of the cable in general.

These two classes have been the centre of the theoretical modelling of the problem, and the resulting curves can be seen just below, accompanied by our own data and the researcher's data taken directly from the article.

Transfer function (transmission through our model):

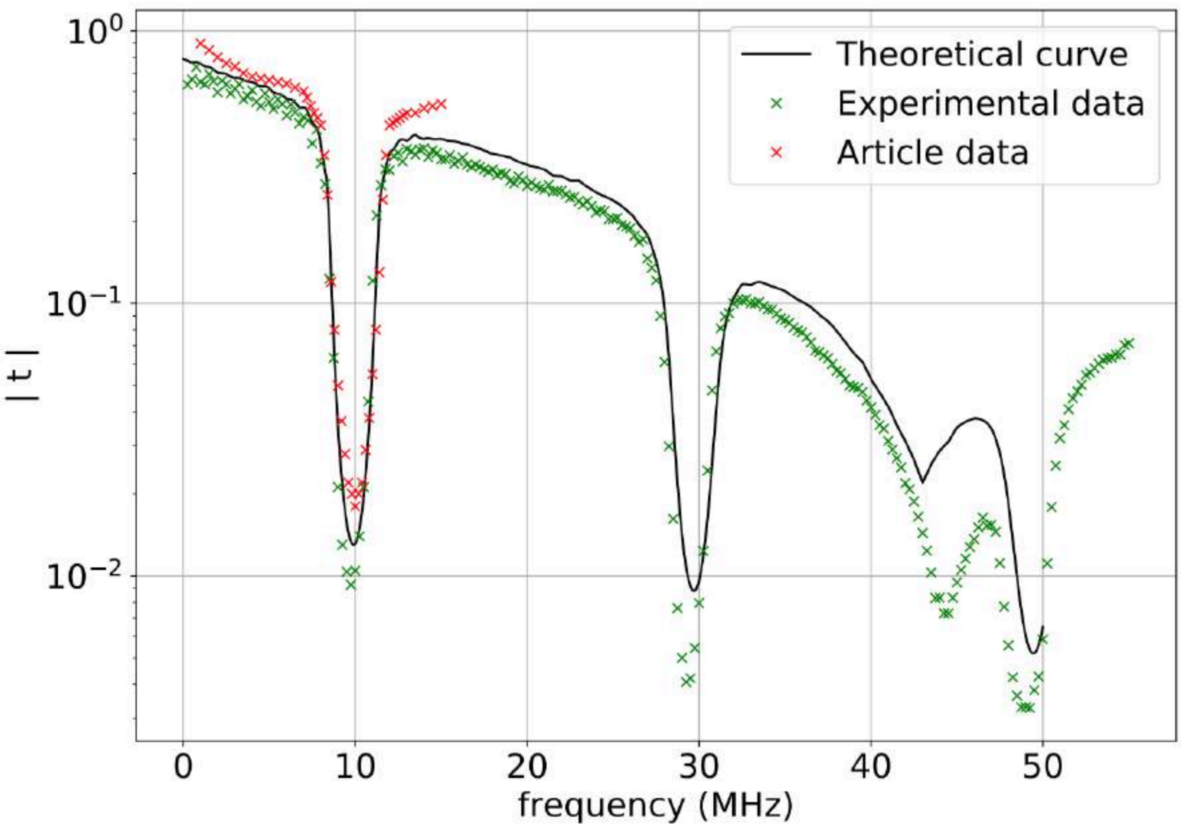


Fig 32: Article's, experimental and theorical results for the transfer function

We can observe that our experimental data but also the article data in its restricted frequency range clearly follow the resulting theoretical curve quite well. Although we can also notice that it seems to lose track around the 42-44 MHz frequency range, right where the attenuation peak could have been observed through our measurements.

It seems as the attenuation peak measured for the RG-59/U cable resulted in a transmission band gap around the same frequencies, where there was not supposed to be anything of the sort as the only band gaps discussed previously were supposed to be at 10,30 and 50 MHz

Dispersion Relation:

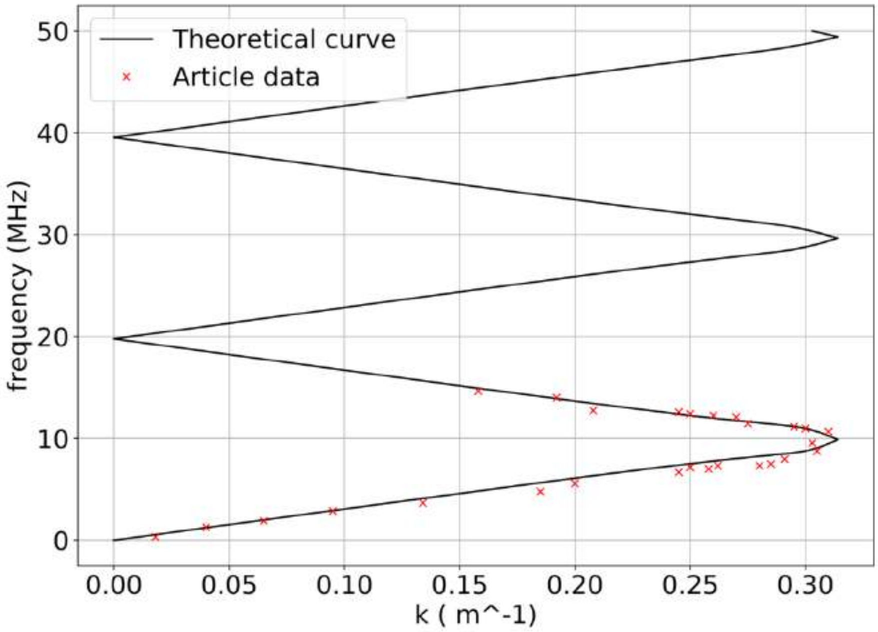


Fig 33: Article's, experimental and theorical results for dispersion relation

We can clearly see we succeeded in folding the theoretical curve as wished and that the article data fits well around our result.
We can also see that the relation of dispersion reaches the value of the first zone of Brillouin three times, each time at a frequency corresponding with the three band gapes of 10,30 and 50 MHz

<u>Group velocities:</u>

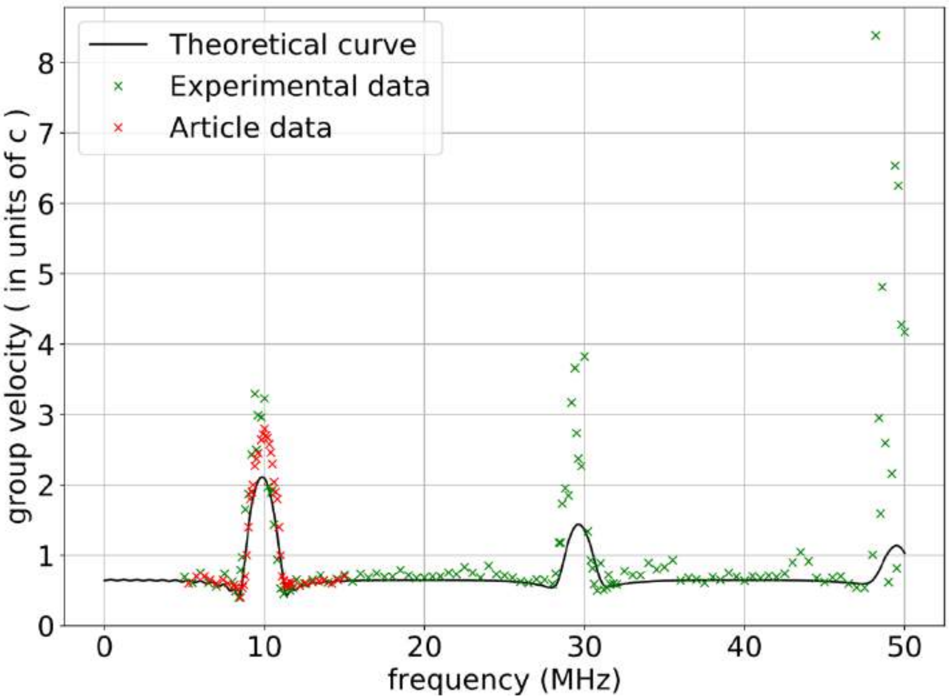


Fig 34: Article's, experimental and theorical results for the group velocities

We can observe three peaks of group velocities becoming supraluminal corresponding to the frequencies of the three band gaps. The data we measured is pretty consistent with our theoretical as is the data from the article, going overboard only in the supraluminal zones where the data assets become both inconsistent.

We can conclude the matrix theory has been successful in accessing the model behaviour and predicting its results. We can although notice some mishaps in some figures, meaning the model is not perfect.

Conclusion

This draft examines a homogeneous coaxial cable in order to better understand the effects observed in the composite cable. Then the transfer function gives the way that a sinusoidal signal is transmitted throughout the system and the relation dispersion gives the link between spatial and temporal frequency. The determination of the group velocity shows the impact of this periodic structures on a gaussian wave packet, in terms of speed. Finally, the model of the system allows a comparison of every experimental result with the theorical curves. Every result was compared to the article of reference.

Haché and Slimani studied the same experiments but for 0-15 MHz. Here, the study is pushed to 50 MHz. A divergence is observable for the highest frequencies on the attenuation curve, for each homogeneous cable. And we observe a periodicity on the transfer function curve and the group velocity one.

The band gaps are significant around 10MHz, 30MHz and 50MHz (each 20MHz). Moreover, the RG59 attenuation curve plots a peak around 40MHz that we find again on the transfer function and group velocity curves.

The model coaxial photonic crystal is useful for exploring effects encountered in periodic media, it is an educational tool easily modifiable and not very expensive.

This project was an educational way to show us how to build an engineering draft. The group members were chosen randomly so we had to work with people we don't use to work with. We all agree to say that this was such an advantage for our group, in fact every member has different skills that were all very useful. Moreover, there was a very productive atmosphere in our group that allows the project to be very interesting and stimulating. We put into practice our knowledge acquired during the instrumentation courses, autonomously. We learnt how to present an engineering project document (divided in several documents, with particular information), which is very different from the research project we used to do.

Annex

Analysis and characterization of a photonic crystal made with coaxial cables

Reference of the document: Annex_Coaxial_Cable.v1
Drafting date: 07/01/2020

Editor

Name	Nina MOËLLO
Function	Project Manager
Organization	L3 Spé Physique Université Paul Sabatier

Client

Name	M. Hassan SABBAH

Table des matières

1. RG-58 U Attenuation CODE

```
################################################################################
#
#   The goal of this program is to approach the curve made by the
#   attenuations according to the frequencies with a curve fit method.
#
#   This program is coded in Python language and needs special
#   libraries to work. Especially Numpy, Matplotlib and Scipy
#   libraries are necessary and the program can't work without them.
#
################################################################################
# IMPORTS ------------------------------------------------------------

################################################################################

  We first import the necessary libraries like mentioned.

################################################################################


from scipy.optimize import curve_fit
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages

# FUNCTIONS ----------------------------------------------------------

################################################################################

  We then need a few functions to ease up the rest of the program.
```

```
def file_reading(nom):
    '''
    This function just reads the contents of a txt file and put them into a list of the lines.
    '''
    file=open(nom,'r')
    t=file.readlines()
    file.close()
    return t

def file_translator(nom):
    '''

 This function reads a file setup a certain way :
    frequency
    attenuation
    frequency
    attenuation
    ....
    The goal of this function is to order them into two separate lists and then return them
    '''
    t=file_reading(nom)
    n=len(t)
    images=[]
    labels=[]
    for i in range(n):
        if i%2==0:
            image=float(t[i])
            images.append(image)
        else:
            label=float(t[i])
            labels.append(label)
    return images,labels

def equation_fit(x,a,b,c,d,e):
    '''
    This function serves as a container of the model function that we want to make fit with the
data
    '''
    return a*x**4+b*x**3+c*x**2+d*x+e

def researcher_fit(x):
    '''
    This function is the associated researcher curve that can be found within the study.
    '''
    return -np.log(-1.7*10**(-9)*x+0.9928)/18.6

# MAIN PROGRAM ---------------------------------------------------------

##########################################################################

  We then code the main program and use all that we defined above.

##########################################################################
```

```
# We first use the two first functions above to obtain a list of frequencies and a list of
attenuations
list_frequencies,list_attenuations=file_translator('attenuations.txt')
# We then generate two arrays : one of pulsations and another of freqencies
pulsations=2*np.pi*np.array(list_frequencies)
attenuations=np.array(list_attenuations)

# We then use Scipy curve_fit method applied on the data, giving a set of initial values for the
coefficients mentionned
# in the equation_fit fucntion above.
popt,pcov=curve_fit(equation_fit,pulsations,attenuations,p0=(-1.0,0.0,1.0*10**(-
9),1.0,1.0),maxfev=8000)

# Since the adapted coefficients are contains inside popt, we extract them all
a=popt[0]
print('a = ',a)
b=popt[1]
print('b = ',b)
c=popt[2]
print('c = ',c)
d=popt[3]
print('d = ',d)
e=popt[4]
print('e = ',e)

# We then plot the data using the newly found curve fit function.
x_frequencies=np.linspace(0,50*10**6,num=3000)
x_pulsations=2*np.pi*x_frequencies
y_attenuations=equation_fit(x_pulsations,a,b,c,d,e)

pdf_save=True
plt.rcParams.update({'font.size': 21})

if pdf_save:
    # If pdf_save is True, then we create a pdf called 'RG58U_pdf.pdf' containing all the
values
    with PdfPages('RG58U_pdf.pdf') as pdf:
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(x_frequencies/(10**6),y_attenuations,'r',label='fitted curve')
        plt.plot(pulsations/(2*np.pi*10**6),attenuations,'x',label = 'data')
        plt.plot(x_frequencies/(10**6),researcher_fit(2*np.pi*x_frequencies),'b',label='researcher
curve')
        plt.xlabel('frequency (MHz)')
        plt.ylabel('attenuation coefficients')
        plt.legend()
        plt.grid()
        pdf.savefig(fig)
        plt.close()

else:
    # If pdf_save is False, we then just plot the values calculated just above
    plt.plot(x_frequencies/(10**6),y_attenuations,'r',label='fitted curve')
```

```
    plt.plot(pulsations/(2*np.pi*10**6),attenuations,'x',label = 'data')
    plt.plot(x_frequencies/(10**6),researcher_fit(2*np.pi*x_frequencies),'b',label='researcher curve')
    plt.xlabel('frequency (MHz)')
    plt.ylabel('attenuation coefficients')
    plt.legend()
    plt.grid()
    plt.show()
```

2. RG-59U Attenuation CODE

```
from scipy.optimize import curve_fit
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages

def file_reading(nom):
    file=open(nom,'r')
    t=file.readlines()
    file.close()
    return t

def file_translator(nom):
    t=file_reading(nom)
    n=len(t)
    images=[]
    labels=[]
    for i in range(n):
        if i%2==0:
            image=float(t[i])
            images.append(image)
        else:
            label=float(t[i])
            labels.append(label)
    return images,labels


list_frequencies,list_attenuations=file_translator('attenuations.txt')
frequencies=np.array(list_frequencies)
pulsations=2*np.pi*np.array(list_frequencies)
attenuations=np.array(list_attenuations)


x_frequencies=np.linspace(0,50*10**6,num=3000)
x_pulsations=2*np.pi*x_frequencies
y_attenuations=np.interp(x_frequencies,frequencies,attenuations)

def researcher_fit(x):
    return -np.log(-1.3*10**(-9)*x+0.9328)/31.5
```

```
pdf_save=True
plt.rcParams.update({'font.size': 21})

if pdf_save:
   with PdfPages('RG59U_pdf.pdf') as pdf:
      fig = plt.figure(figsize=(11.69,8.27))
      plt.plot(x_frequencies/(10**6),y_attenuations,'r',label='interpolation curve')
      plt.plot(pulsations/(2*np.pi*10**6),attenuations,'x',label = 'data')
      plt.plot(x_frequencies/(10**6),researcher_fit(2*np.pi*x_frequencies),'b',label='researcher
curve')
      plt.xlabel('frequency (MHz)')
      plt.ylabel('attenuation coefficients')
      plt.legend()
      plt.grid()
      pdf.savefig(fig)
      plt.close()

else:
   plt.plot(x_frequencies/(10**6),y_attenuations,'r',label='fitted curve')
   plt.plot(pulsations/(2*np.pi*10**6),attenuations,'x',label = 'data')
   plt.plot(x_frequencies/(10**6),researcher_fit(2*np.pi*x_frequencies),'b',label='researcher
curve')
   plt.xlabel('frequency (MHz)')
   plt.ylabel('attenuation coefficients')
   plt.legend()
   plt.grid()
   plt.show()
```

3. Matrix modelling CODE

```
######################################################################
The goal of this program is to model the coaxial photonic
cristal using a matrix solving method to retrieve the theoretical
curves contained in the original study of Alain Haché and
Abderrahim Slimani that was the base of our Study Bureau. We also wish
to get all data taken from our own experiments and
the article to compare them by plotting them.
This program is coded in Python language and needs special
libraries to work. Especially Numpy and Matplotlib libraries are necess
ary and the programm can't work without them
######################################################################

# IMPORTS ------------------------------------------------------------

######################################################################
We first import the necessary libraries like mentioned
######################################################################
```

```
import numpy as np
import csv
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages
# CLASSES -------------------------------------------------------------
#######################################################################
We create a Python Object modelizing a certain medium.
#######################################################################
class Medium():
    '''
    The Medium class creates a certain medium and saves up its specific
s useful for the rest
    of the program.

    Parameters :

    - length : length of the medium ( default value = None )
    - impedance : electrical impedance of the medium ( default value =
None )
    - phase_velocity : the phase velocity of electical waves in the med
ium ( default value = None )
    - attenuation_function : a function of lineic attenuation depending
 on the pulsation of the electrical wave inside the medium ( default va
lue = None )
    - name : the designated name of the medium ( default value = None )


    Attributes :

    - length : length of the medium
    - impedance : electrical impedance of the medium
    - phase_velocity : the phase velocity of electical waves in the med
ium
    - attenuation_function : a function of lineic attenuation depending
 on the pulsation of the electrical wave inside the medium
    - name : the designated name of the medium

    Fonctions :

    - none

    '''
    def __init__(self,length=None,impedance=None,phase_velocity=None,at
tenuation_function=None,name=None):
        self.length=length
        self.impedance=impedance
        self.phase_velocity=phase_velocity
        self.attenuation_function=attenuation_function
```

```
        self.name=name

######################################################################
We create a Python Object modelizing the all photonic cristal
coaxial model and easing the calculations of the theoretical
values and curves.
######################################################################

class Multi_Layered_Media_Model():
    '''
    The Multi_Layered_Media_Model class creates a coaxial photonic cris
tal model and lets you decide the layout of layers and their specifics.

    Parameters :

    - layers : defines the medium that will be used to create the layer
s layout, input a list of Medium Objects ( default value = [] )
    - build_mode : defines how the layers defined just before will be u
sed to create the layout.
                  Two values can be used :
                  . 'direct_define' : layout is created by directly put
ting the layers as given
                  . 'alternation' : layout is created by putting the la
yers given on a periodic system
    - n_alternation : number of periods of layouts using the layers giv
en ( default value = None )
    - input_medium : defines an input medium for the model ( default va
lue = None )
    - output_medium : defines an output medium for the model ( default
value = None )

    Attributes :

    - layers : the layout of the layers, contains Medium objects
    - base_layers : base layers used in the creation of the layout
    - build_mode : defines how the base layers were used in the creatio
n of the model
    - n_alternation : number of periods of layouts using the layers giv
en ( default value = None )
    - input_medium : input medium for the model
    - output_medium : output medium for the model

    - pulsations : contains the pulsations range on which further calcu
lations are done
    - c_transmissions : contains the complex values of transmissions do
ne on the pulsations range
    - c_reflections : contains the reflections ratio done on the pulsat
ions range
```

```
    - phases : contains the arguments of the c_transmissions ( phases )
    - phase_shifts : contains the phase shifts done on the pulsations r
ange
    - refraction_indexes : contains the rafraction indexes calculated o
n the pulsations range
    - wave_numbers : contains the wave numbers calculated on the pulsat
ions range
    - group_velocities : contains the group velocities calculated on th
e pulsations range

    Fonctions :

    - __str__ : lets the user use the print() function directly to get
the layout of the model
    - total_length : calculates the total length of the coaxial model
    - overall_transmission : calculates and returns the c_transmissions
 and c_reflections ( calculates the transfer function )
    - overall_phases : gets the arguments of the complex transmissions
and return them ( calculates the phases )
    - overall_phase_shifts : calculates the phase shifts using the phas
es
    - overall_refraction_indexes : calculates the refraction indexes us
ing previous data
    - overall_wave_numbers : calculates the wave numbers using previous
 data
    - overall_group_velocities : calculates the group velocities using
previous data

    '''
    def __init__(self,layers=[],build_mode='direct_define',n_alternatio
n=None,input_medium=None,output_medium=None):
        # Function that initialize the model and constructs the layout
based on the base layers given and the building mode given
        if build_mode=='alternation':
            # If build_mode is ' alteration', we take the base layers a
nd repeat them as many as n_alternation
            assert type(layers)==list and len(layers)!=0
            self.base_layers=layers
            n=n_alternation
            k=len(layers)
            self.layers=[]
            for i in range(n):
                for j in range(k):
                    self.layers.append(self.base_layers[j])
            assert len(self.layers)==len(self.base_layers)*n

        if build_mode=='direct_define':
```

```
            # If build_mode is ' direct_define', the layers layout is a
 direct copy of the base layers
            self.base_layers=layers
            self.layers=layers
        # We then define other variables using the inputs of the initia
lization
        self.n_layers=len(self.layers)
        self.n_alternation=n_alternation
        self.input_medium=input_medium
        self.output_medium=output_medium

        # Saved values for later calculations, made in None variables a
t first
        self.pulsations=None
        self.c_trannsmissions=None
        self.c_reflections=None
        self.phases=None
        self.phase_shifts=None
        self.refraction_indexes=None
        self.wave_numbers=None
        self.group_velocities=None

    def __str__(self):
        # As mentionned above, prints out the layout of the model
        string='Coaxial Model = | '
        for i in range(self.n_layers):
            string=string+str(self.layers[i].name)+' | '
        return string

    def total_length(self):
        # As mentionne above, gives the total length of the model
        d=0
        for i in range(len(self.layers)):
            d=d+self.layers[i].length
        return d

    def overall_transmission(self,w):
        # As mentionned above, calculates the transfer function values
for a given pulsation range ( given with w )
        # We first verify that we have a input and output medium as the
 methode used needs one
        assert self.input_medium!=None ;" No Input Medium defined "
        assert self.output_medium!=None ;" No Output Medium defined "
        t=1
        r=1
        # We calculate the transmission and reflection ratio based on t
he formulas derived from the matrix modelization of the model
        for k in range(self.n_layers): # k : calculation index
```

```
            i=(self.n_layers-1)-k      # i : medium index in the model
            # Calculation for the last medium of the model layout ( fir
st one to be studied --> k=0 )
            if k==0:
                LayerH=self.layers[i-1]# Layer H : previous layer
                LayerI=self.layers[i] # Layer I : studied layer
                LayerJ=self.output_medium # Layer J : layer just after
                # Each time we calculate the different
                t1plus=transmission(LayerH,LayerI)
                t2plus=transmission(LayerI,LayerJ)
                t1minus=transmission(LayerI,LayerH)
                r1plus=reflection(LayerH,LayerI)
                r2plus=reflection(LayerI,LayerJ)
                r1minus=reflection(LayerI,LayerH)
                D=LayerI.length
                k=LayerI.attenuation_function(w)
                phase_shift=w*D/LayerI.phase_velocity
                t=(t1plus*t2plus*np.exp(1j*phase_shift-k*D))/(1-
r1minus*r2plus*np.exp(2j*phase_shift-2*k*D))
                r=r1plus+(t1plus*t1minus*r2plus*np.exp(2j*phase_shift-
2*k*D))/(1-r1minus*r2plus*np.exp(2j*phase_shift-2*k*D))
            # Calculation for the first medium of the model layout ( la
st one to be studied --> k=n_layers-1 )
            elif k==self.n_layers-1:
                LayerH=self.input_medium
                LayerI=self.layers[i]
                LayerJ=self.layers[i+1]
                t1plus=transmission(LayerH,LayerI)
                t2plus=t
                t1minus=transmission(LayerI,LayerH)
                r1plus=reflection(LayerH,LayerI)
                r2plus=r
                r1minus=reflection(LayerI,LayerH)
                D=LayerI.length
                k=LayerI.attenuation_function(w)
                phase_shift=w*D/LayerI.phase_velocity
                t=(t1plus*t2plus*np.exp(1j*phase_shift-k*D))/(1-
r1minus*r2plus*np.exp(2j*phase_shift-2*k*D))
                r=r1plus+(t1plus*t1minus*r2plus*np.exp(2j*phase_shift-
2*k*D))/(1-r1minus*r2plus*np.exp(2j*phase_shift-2*k*D))
            # Calculation for the rest of the layers
            else:
                LayerH=self.layers[i-1]
                LayerI=self.layers[i]
                LayerJ=self.layers[i+1]
                t1plus=transmission(LayerH,LayerI)
                t2plus=t
                t1minus=transmission(LayerI,LayerH)
```

```
                r1plus=reflection(LayerH,LayerI)
                r2plus=r
                r1minus=reflection(LayerI,LayerH)
                D=LayerI.length
                k=LayerI.attenuation_function(w)
                phase_shift=w*D/LayerI.phase_velocity
                t=(t1plus*t2plus*np.exp(1j*phase_shift-k*D))/(1-
r1minus*r2plus*np.exp(2j*phase_shift-2*k*D))
                r=r1plus+(t1plus*t1minus*r2plus*np.exp(2j*phase_shift-
2*k*D))/(1-r1minus*r2plus*np.exp(2j*phase_shift-2*k*D))
        # We store the values of the results and return them to the use
r as well
        self.c_transmissions=t
        self.r_transmissions=r
        self.pulsations=w
        return t,r

    def overall_phases(self):
        # We extract the phases by calculating the arguments of the tra
nsmissions that are imaginary values
        self.phases=np.arctan(self.c_transmissions.imag/self.c_transmis
sions.real)
        return self.phases

    def overall_phase_shifts(self):
        # We calculate the phase shifts by adding 180° each time the ph
ases go from +180° to -180°,if we reason in degrees.
        # Here we reason in radians
        phase_shifts=np.zeros_like(self.phases)
        m=0
        for i in range(self.phases.shape[0]):
            if i!=0 and self.phases[i-1]==abs(self.phases[i-
1]) and self.phases[i]==-abs(self.phases[i]):
                m=m+1
            phase_shifts[i]=self.phases[i]+m*np.pi
        self.phase_shifts=phase_shifts
        return phase_shifts

    def overall_refraction_indexes(self):
        # We calculate the refraction indexes using the phase shifts ca
lculated just above, we then store the velues
        indexes=(self.phase_shifts*constant('c'))/(self.total_length()*
self.pulsations)
        self.refraction_indexes=indexes
        return indexes

    def overall_wave_numbers(self):
```

```
        # We calculate the wave numbers using the dispersion relation a
nd storing those values
        numbers=(self.refraction_indexes*self.pulsations)/constant('c')
        N=numbers.shape[0]
        k0=0.314
        for i in range(N):
            k=numbers[i]
            while k>=k0:
                k=2*k0-abs(k)
                k=abs(k)
            numbers[i]=k
        self.wave_numbers=numbers
        return numbers

    def overall_group_velocities(self):
        # We calculate the group velocities on the pulsation range and
store the values
        dw=self.pulsations[1]-self.pulsations[0]
        dndw=np.gradient(self.refraction_indexes,dw) #np.gradient() is
a finite derivative method so that we can get dn/dw
        self.group_velocities=constant('c')/(self.refraction_indexes+se
lf.pulsations*dndw)
        return self.group_velocities


# FUNCTIONS -------------------------------------------------------------
#########################################################################
We then need a few other needed functions used above.
#########################################################################
# The new attenuations that we mesured and modelized or interpolized to
 get rough estimates

# For the RG58U cable, we did a curve fit on the data taken from our me
asurements and got the following expression
attenuation_RG58U=lambda w : -3.788179118764273 5*10**(-
36)*(w**4)+2.735326073454789*10**(-27)*(w**3)-6.367690393696193*10**(-
19)*(w**2)+1.0397790765150726*10**(-10)*w+0.0012431624352975269
# It essentially is a fourth degree polynom with coefficients best adap
ted to the data fed to the program that came up with them

# For the RG59U cable, we sadly couldn't make a curve fit, the fucntio
being to difficult to modelize simply
# We are then forced to do a simple but rough first degree interpolatio
n ( meaning we draw lines between the data points and use that as a mak
eshift function )

def attenuation_RG59U(w):
```

```
    list_frequencies,list_attenuations=txt_file_translator('RG59U frequ
encies-attenuations.txt')
    frequencies=np.array(list_frequencies)
    pulsations=2*np.pi*np.array(list_frequencies)
    attenuations=np.array(list_attenuations)
    return np.interp(w,pulsations,attenuations)

def file_reading(nom):
    file=open(nom,'r')
    t=file.readlines()

    file.close()
    return t

def txt_file_translator(nom):
    t=file_reading(nom)
    n=len(t)
    images=[]
    labels=[]
    for i in range(n):
        if i%2==0:
            image=float(t[i])
            images.append(image)
        else:
            label=float(t[i])
            labels.append(label)
    return images,labels

def csv2txt(csv_file_name,txt_file_name):
    ''' Transforms a csv file into a txt file ( easier to read in Pytho
n ) '''
    with open(txt_file_name, "w") as my_output_file:
        with open(csv_file_name, "r") as my_input_file:
            [ my_output_file.write(" ".join(row)+'\n') for row in csv.r
eader(my_input_file)]
    my_output_file.close()


def standarization(txt_file_name):
    ''' Focuses on getting floats into an anglophone syntax ( basically
 , --> . )'''
    with open(txt_file_name,"r")as file:
        LINES=file.readlines()
        file.close()
    N=len(LINES)
    for i in range(N):
        line=str(LINES[i])
        line=line.replace(' ','.')
```

```
        LINES[i]=line
    with open(txt_file_name,"w")as file:
        for j in range(N):
            file.write(LINES[j])

def read_float_file(file_name):
    ''' This function takes a file_name as input and ouputs the float c
ontents of the file
        in a ordered list
    '''
    content=[]

    with open(file_name,'r') as file:
        LINES=file.readlines()
        N=len(LINES)
        for i in range(N):
            content.append(float(LINES[i]))
        file.close()
    return content

def transmission(medium1,medium2):
    ''' calculates the transmission ratio from a medium1 to a medium2''
'
    z1=medium1.impedance
    z2=medium2.impedance
    return (2*z1)/(z1+z2)

def reflection(medium1,medium2):
    ''' calculates the reflection ratio from a medium1 to a medium2'''
    z1=medium1.impedance
    z2=medium2.impedance
    return (z1-z2)/(z1+z2)
def constant(name):
    ''' returns constants from a given name in SI units'''
    if name=='c':
        return 299792458
# MAIN PROGRAM -------------------------------------------------------
########################################################################
   We then code the main programm and use all that we defined above
########################################################################
# We first create the frequency/ pulsation range up to 50 MHz
frequencies=np.linspace(1,50*10**6,num=100000)
impulsions=2*np.pi*frequencies

# We the, define the 4 media used in the model ( input, output, RG58U a
nd RG59U )
Input=Medium(impedance=50,name='GBF')
Output=Medium(impedance=50,name='Termination')
```

```
RG58U=Medium(length=5,impedance=50,
             phase_velocity=0.66*constant('c'),
             attenuation_function=attenuation_RG58U,
             name='RG58U')
RG59U=Medium(length=5,impedance=75,
             phase_velocity=0.66*constant('c'),
             attenuation_function=attenuation_RG59U,
             name='RG59U')

# We the use the Multi_Layered_Media_Model class to create the coaxial
model
Coaxial=Multi_Layered_Media_Model(layers=[RG59U,RG58U],
                                  build_mode='alternation',
                                  n_alternation=12,
                                  input_medium=Input,
                                  output_medium=Output)

# We finally calculate all the theroetical values by calling all the fu
nctions attached to the model
c_transmissions,c_reflections=Coaxial.overall_transmission(impulsions)
transmissions,reflections=abs(c_transmissions),abs(c_reflections)

phases=Coaxial.overall_phases()
phases_shifts=Coaxial.overall_phase_shifts()

indexes=Coaxial.overall_refraction_indexes()
numbers=Coaxial.overall_wave_numbers()
velocities=Coaxial.overall_group_velocities()

# We print some useful informations like the model layout and the total
 length
print(Coaxial)
print('Model total length : ',Coaxial.total_length(),' m')

# We then need the useful data to plot them with the theoreatical curve
s
#-------------- Article data ----------------#
# TRANSMISSIONS
csv2txt('Transmissions article.csv','Transmissions article.txt')
standarization('Transmissions article.txt')
LINES=file_reading('Transmissions article.txt')
AT_Frequencies=[]
AT_Transmissions=[]
LINES.pop(0)
N=len(LINES)
for i in range(N):
    if len(LINES[i])>0:
        line=LINES[i].split(';')
```

```
        AT_Frequencies.append(float(line[0])*10**(6))
        AT_Transmissions.append(float(line[1]))
AT_Frequencies=np.array(AT_Frequencies)
AT_Transmissions=np.array(AT_Transmissions)
#DISPERSION RELATION
csv2txt('Dispersion Relation article.csv','Dispersion Relation article.
txt')
standarization('Dispersion Relation article.txt')
LINES=file_reading('Dispersion Relation article.txt')

AK_Frequencies=[]
AK_WaveNumbers=[]
LINES.pop(0)
N=len(LINES)
for i in range(N):
    if len(LINES[i])>0:
        line=LINES[i].split(';')
        AK_Frequencies.append(float(line[0])*10**(6)/(2*np.pi))
        AK_WaveNumbers.append(float(line[1]))
AK_Frequencies=np.array(AK_Frequencies)
AK_WaveNumbers=np.array(AK_WaveNumbers)
#GROUP VELOCITIES
csv2txt('Group Velocities article.csv','Group Velocities article.txt')
standarization('Group Velocities article.txt')
LINES=file_reading('Group Velocities article.txt')
AS_Frequencies=[]
AS_Speeds=[]
LINES.pop(0)
N=len(LINES)
for i in range(N):
    if len(LINES[i])>0:
        line=LINES[i].split(';')
        AS_Frequencies.append(float(line[0])*10**(6))
        AS_Speeds.append(float(line[1]))
AS_Frequencies=np.array(AS_Frequencies)
AS_Speeds=np.array(AS_Speeds)
#-------------- Experimental data ----------------#
# TRANSMISSIONS
csv2txt('Composite Cable Experimental Transmissions.csv','Composite Cab
le Experimental Transmissions.txt')
standarization('Composite Cable Experimental Transmissions.txt')
LINES=file_reading('Composite Cable Experimental Transmissions.txt')
ET_Frequencies=[]
ET_Transmissions=[]
LINES.pop(0)
LINES.pop(0)
N=len(LINES)
for i in range(N):
```

```
    if len(LINES[i])>0:
        line=LINES[i].split(';')
        ET_Frequencies.append(float(line[0])*10**(6))
        ET_Transmissions.append(float(line[2])/float(line[1]))
ET_Frequencies=np.array(ET_Frequencies)
ET_Transmissions=np.array(ET_Transmissions)
# GROUP VELOCITIES
ES_Speeds=read_float_file('Composite Cable Experimental Group Velocities.txt')
ES_Frequencies=read_float_file('Composite Cable Experimental Group Velocities Frequencies.txt')
ES_Speeds=np.array(ES_Speeds)
ES_Frequencies=np.array(ES_Frequencies)
pdf_save=True
plt.rcParams.update({'font.size': 21})

if pdf_save:
    # If pdf_save is True, then we create a pdf called multipage_pdf.pdf containing all the values
    with PdfPages('multipage_pdf.pdf') as pdf:
        #-------------------TRANSMISSIONS-----------------#
        # Transmission theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),transmissions,'k',label='Theoretical curve')
        plt.yscale("log")
        plt.xlabel('frequency (MHz)')
        plt.ylabel('| t |')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Transmission experimental
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(ET_Frequencies/(10**6),ET_Transmissions,'gx',label='Experimental data')
        plt.yscale("log")
        plt.xlabel('frequency (MHz)')
        plt.ylabel('| t |')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Transmission experimental+theoretical
        fig = plt.figure(figsize=(11.69,8.27))
```

```
        plt.plot(frequencies/(10**6),transmissions,'k',label='Theoretic
al curve')
        plt.plot(ET_Frequencies/(10**6),ET_Transmissions,'gx',label='Ex
perimental data')
        plt.yscale("log")
        plt.xlabel('frequency (MHz)')
        plt.ylabel('| t |')
        plt.grid()
        plt.legend()

        pdf.savefig(fig)
        plt.close()

        # Transmission article
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(AT_Frequencies/(10**6),AT_Transmissions,'rx',label='Ar
ticle data')
        plt.yscale("log")
        plt.xlabel('frequency (MHz)')
        plt.ylabel('| t |')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Transmission article+experimental+theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),transmissions,'k',label='Theoretic
al curve')
        plt.plot(ET_Frequencies/(10**6),ET_Transmissions,'gx',label='Ex
perimental data')
        plt.plot(AT_Frequencies/(10**6),AT_Transmissions,'rx',label='Ar
ticle data')
        plt.yscale("log")
        plt.xlabel('frequency (MHz)')
        plt.ylabel('| t |')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Transmission article+experimental
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(ET_Frequencies/(10**6),ET_Transmissions,'gx',label='Ex
perimental data')
        plt.plot(AT_Frequencies/(10**6),AT_Transmissions,'rx',label='Ar
ticle data')
        plt.yscale("log")
```

```
        plt.xlabel('frequency (MHz)')
        plt.ylabel('| t |')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),(phases*360)/(2*np.pi),'k',label='
Theoretical curve')

        plt.xlabel('frequency (MHz)')
        plt.ylabel('phases (°)')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),(phases_shifts*360)/(2*np.pi),'k',
label='Theoretical curve')
        plt.xlabel('frequency (MHz)')
        plt.ylabel('phase shifts (°)')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),indexes,'k',label='Theoretical cur
ve')
        plt.xlabel('frequency (MHz)')
        plt.ylabel('refraction index')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        #-------------------DISPERSION RELATION-----------------#
        # Dispersion relation theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(numbers,frequencies/(10**6),'k',label='Theoretical cur
ve')
        plt.ylabel('frequency (MHz)')
        plt.xlabel('k ( m^-1)')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
```

```
        plt.close()

        # Dispersion relation article+theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(numbers,frequencies/(10**6),'k',label='Theoretical cur
ve')
        plt.plot(AK_WaveNumbers,AK_Frequencies/(10**6),'rx',label='Arti
cle data')
        plt.ylabel('frequency (MHz)')
        plt.xlabel('k ( m^-1)')
        plt.grid()

        plt.legend()
        pdf.savefig(fig)
        plt.close()

        #------------------GROUP VELOCITIES-----------------#
        # Group velocities theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),velocities/constant('c'),'k',label
='Theoretical curve')
        plt.ylabel('group velocity ( in units of c )')
        plt.xlabel('frequency (MHz)')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Group velocities experimental
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(ES_Frequencies/(10**6),ES_Speeds,'gx',label='Experimen
tal data')
        plt.ylabel('group velocity ( in units of c )')
        plt.xlabel('frequency (MHz)')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Group velocities experimental+theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),velocities/constant('c'),'k',label
='Theoretical curve')
        plt.plot(ES_Frequencies/(10**6),ES_Speeds,'gx',label='Experimen
tal data')
        plt.ylabel('group velocity ( in units of c )')
        plt.xlabel('frequency (MHz)')
        plt.grid()
```

```
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Group velocities article+experimental
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(ES_Frequencies/(10**6),ES_Speeds,'gx',label='Experimen
tal data')
        plt.plot(AS_Frequencies/(10**6),AS_Speeds,'rx',label='Article d
ata')
        plt.ylabel('group velocity ( in units of c )')
        plt.xlabel('frequency (MHz)')

        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

        # Group velocities article+experimental+theoretical
        fig = plt.figure(figsize=(11.69,8.27))
        plt.plot(frequencies/(10**6),velocities/constant('c'),'k',label
='Theoretical curve')
        plt.plot(ES_Frequencies/(10**6),ES_Speeds,'gx',label='Experimen
tal data')
        plt.plot(AS_Frequencies/(10**6),AS_Speeds,'rx',label='Article d
ata')
        plt.ylabel('group velocity ( in units of c )')
        plt.xlabel('frequency (MHz)')
        plt.grid()
        plt.legend()
        pdf.savefig(fig)
        plt.close()

else:
    # If pdf_save is False, we then just plot the values calculated ear
lier
    fig=plt.figure() # We define the figure

    ax1=fig.add_subplot(131)
    ax1.plot(frequencies/(10**6),transmissions)
    plt.yscale("log")
    plt.xlabel('frequency (MHz)')
    plt.ylabel('| t |')

    ax2=fig.add_subplot(132)
    ax2.plot(numbers,frequencies/(10**6))
    plt.ylabel('frequency (MHz)')
    plt.xlabel('k ( m^-1 )')
```

```
ax3=fig.add_subplot(133)
ax3.plot(frequencies/(10**6),velocities/constant('c'))
plt.ylabel('group velocity ( units of c )')
plt.xlabel('frequency (MHz)')

plt.show()# We show the figure
```

4. Gaussian wave packets creation CODE

```
#################################################################
The goal of this program is to generate a Gaussian packet of certa
in frequency in a csv format which is readable by the
used in our experiments with the coaxial photonic cristal model. T
he frequencies are taken from a seperate file containing the
ones we carefully chose.
This programm is coded in Python language and needs special librar
ies to work. Especially Numpy, Matplotlib and Csv libraries are ne
cessary and the programm can't work without them
##########################################################

# IMPORTS -----------------------------------------------

    ##########################################################
    #   We first import the necessary libraries like mentionned
    ###########################################################

    import numpy as np
    import matplotlib.pyplot as plt
    import csv

    # FUNCTIONS ------------------------------------------------------------

    # We need a few fucntions to generate the packets

    def Gauss_function(x,mu,sigma):
        '''
        This function generates a Gaussian function ( bell function ) on a x interval or x value
        '''
        return (1/(sigma*np.sqrt(2*np.pi)))*np.exp(-(x-mu)**2/(2*sigma**2))

    def create_file_Gaussian(fw,f0,width,fGBF,Amplitude,name):
```

```
    '''
    This function creates a csv file containing a Gaussian packet. The packet is put in the middle of an interval
    of a length of 1/fw by calculationg the product between a Gaussian function and a sin fucntion with a
    period of 1/f0. The Gaussian function is put at a mean of 0 and with a certain width.
    The name of the csv file is given by the user.

    The Gaussian packet is created with a number of points generated specifically for the sampling
    frequency of the GBF expected to read the file.
    '''
    N=int(fGBF/fw)
    Tw=1/fw
    X=np.linspace(-Tw/2,Tw/2,num=N)
    Sinusoid=np.sin(2*np.pi*f0*X)
    Gaussian=Gauss_function(X,0,width)
    G0=Gauss_function(0,0,width)
    Y=Gaussian*Amplitude*Sinusoid/G0
    with open(name, 'w', newline='') as csvfile:
        float_writer = csv.writer(csvfile, delimiter=' ',quotechar='|', quoting=csv.QUOTE_MINIMAL)
        for i in range(N):
            float_writer.writerow([str(Y[i])])

def name(f,i,n):
    '''
    This function's goal is to generate a name for the Gaussian packets file.
    The name contains the frequency but also an index for more accurate management of the files.
    '''
    i_str=str(i)
    n_str=str(n)
    len_i=len(i_str)
    len_n=len(n_str)
    if len_i<len_n:
        num='0'*(len_n-len_i)+i_str
    else:
        num=i_str
    return "Gaussian_Packet_"+num+"_"+str(f)+".csv"

# We first mention the Gaussian packet outer frequency ( the frequency at which the packest will appear )
fw=10000

```

```
    # We then need to mention the sampling frequency of the GBF and the width of the Gaussian packets
    width=4*10**(-6)
    fGBF=200*10**6

    # We then extract the frequencies we want from the 'frequencies.txt' file containing them all
    frequencies=[]
    with open('frequencies.txt','r') as file:

        LINES=file.readlines()
        N=len(LINES)
        for k in range(N):
            frequencies.append(float(LINES[k]))
        file.close()

    # Then we use all the functions defined above to generate a csv file for every frequency extracted just above
    N=len(frequencies)
    for i in range(int(N)):
        create_file_Gaussian(fw,frequencies[i],width,fGBF,10,name(frequencies[i],i,1000))
```

5. Group velocities calculation CODE

```
########################################################################
The goal of this program is to calculate the locations of center
of masses of gaussian packets on oscilloscope scopes, obtaining
the group velocity of the packet through our coaxial model by
calculating the differences between the center of mass of the
input packet and the center of mass of the output packet. This position
delay will be expressed as a velocity, giving us the group velocity of
the Gaussian packet through the model.
This program is coded in Python language and needs special
libraries to work. Especially Numpy, Matplotlib, Os and Csv
libraries are necessary and the program can't work without
them.
########################################################################
# IMPORTS --------------------------------------------------------------
########################################################################
We first import the necessary libraries like mentioned
########################################################################

import csv
import numpy as np
import matplotlib.pyplot as plt
import os
```

```
# FUNCTIONS -----------------------------------------------------------
######################################################################
We then need a few functions to easy up the rest of the program
######################################################################
def file_extension(file_name):
    ''' This function gives the extension of the file.
        Example : file01.txt ---> txt
    '''
    file=file_name.split('.')

    return file[1]

def name_file(file_name):
    ''' This function gives the name of the file without the extension.
        Example : file01.txt ---> file01
    '''
    file=file_name.split('.')
    return file[0]

def trapeze_method(X,Y):
    ''' From two lists X and Y with Y = f(X), calculate an approximatio
n of the area below the curve of f
        on the interval [ X[0] , X[-
1] ] by using the trapezes method of integral calculation.
    '''
    assert len(X)==len(Y)
    dX=X[1]-X[0]
    N=len(X)
    S=0
    for i in range(N-1):
        S=S+(Y[i+1]+Y[i])*dX/2 # We each time add a little trapeze betw
een X[i+1] and X[i]
    return S

def is_file(file_name):
    ''' This function returns True if the input is a file. It does that
 by checking for extensions which
        directories don't have'''
    file=file_name.split('.')
    if len(file)==2:
        return True
    else:
        return False

def signal_centroid(X,Y):
    ''' This function returns the x position of the centroid of a signa
l inputes through the Y list of points
        associated with X. '''
```

```
    assert len(X)==len(Y)
    XY=X*abs(Y)
    ES=trapeze_method(X,abs(Y))
    tES=trapeze_method(X,XY)
    center=tES/ES
    return center

def constant(name):
    ''' This function returns a wanted known constant value by giving i
ts name '''

    if name=='c':
        return 299792458
# MAIN PROGRAM -----------------------------------------------------------

########################################################################
We then code the main programm and use all that we defined above
#######################################################################
# We first need to specify the directory where all the scopes ( that ar
e in a csv format ) are located
input_directory="C://Users//engue/Desktop/scopes"

# We then open a new file where all the group velocities will be stored
 once calculated
time_delay_file=open('speeds_in_c.txt','w')
# We then scan the directory mentionned for any object, be it files or
other directories
files_list = os.listdir(input_directory)

Speeds=[]
files_number=len(files_list)
for i in range(files_number):
    csv_file_name=files_list[i]
    if is_file(csv_file_name) and file_extension(csv_file_name)=='csv':
 # We check if it is a csv format file
        print(csv_file_name)
        # We then create the name of a file that will contain all the v
alues in the csv by using the name of the csv file
        # And we also load all the values and infos of the csv into thi
s new txt file
        txt_file_name=name_file(csv_file_name)+'.txt'
        with open(txt_file_name, "w") as my_output_file:
            with open(csv_file_name, "r") as my_input_file:
                [ my_output_file.write(" ".join(row)+'\n') for row in c
sv.reader(my_input_file)]
        my_output_file.close()
        # Once the file txt done, we read all its lines
        txt_file=open(txt_file_name,'r')
```

```
        LINES=txt_file.readlines()
        txt_file.close()
        # We get rid of useless lines ( the first and second one )
        LINES.pop(0)
        LINES.pop(0)
        # Then we retrieve the values of the X coordinates and the CH1
and CH2 values, getting them
        # into the Y1 and Y2 lists
        n_lines=len(LINES)
        X=[]
        Y1=[]

        Y2=[]
        for j in range(n_lines):
            line=LINES[j].split()
            if len(line)==3:
                X.append(float(line[0]))
                Y1.append(float(line[1]))
                Y2.append(float(line[2]))
        # We transform the obtained lists into arrays
        X=np.array(X)
        Y1=np.array(Y1)
        Y2=np.array(Y2)
        # We obtain the center of masses of each Gaussian packet by usi
ng the signal_centroid() function defined above
        t1=signal_centroid(X,Y1)
        t2=signal_centroid(X,Y2)
        # We then get the difference between the two
        time_delay=abs(t1-t2)
        # We then convert it into the group velocities by calculating t
he speed associated with this time delay when
        # the signal go through our 120 m long coaxial photonic cristal
        speed=(120/time_delay)/constant('c')
        Speeds.append(speed)
        # We then write this speed in the file 'speeds_in_c.txt' opened
 at the very beginning
        time_delay_file.write(str(speed)+'\n')
# Once all the scopes read and the speeds obtained we close the file, s
aving all the speeds in the process
time_delay_file.close()
```