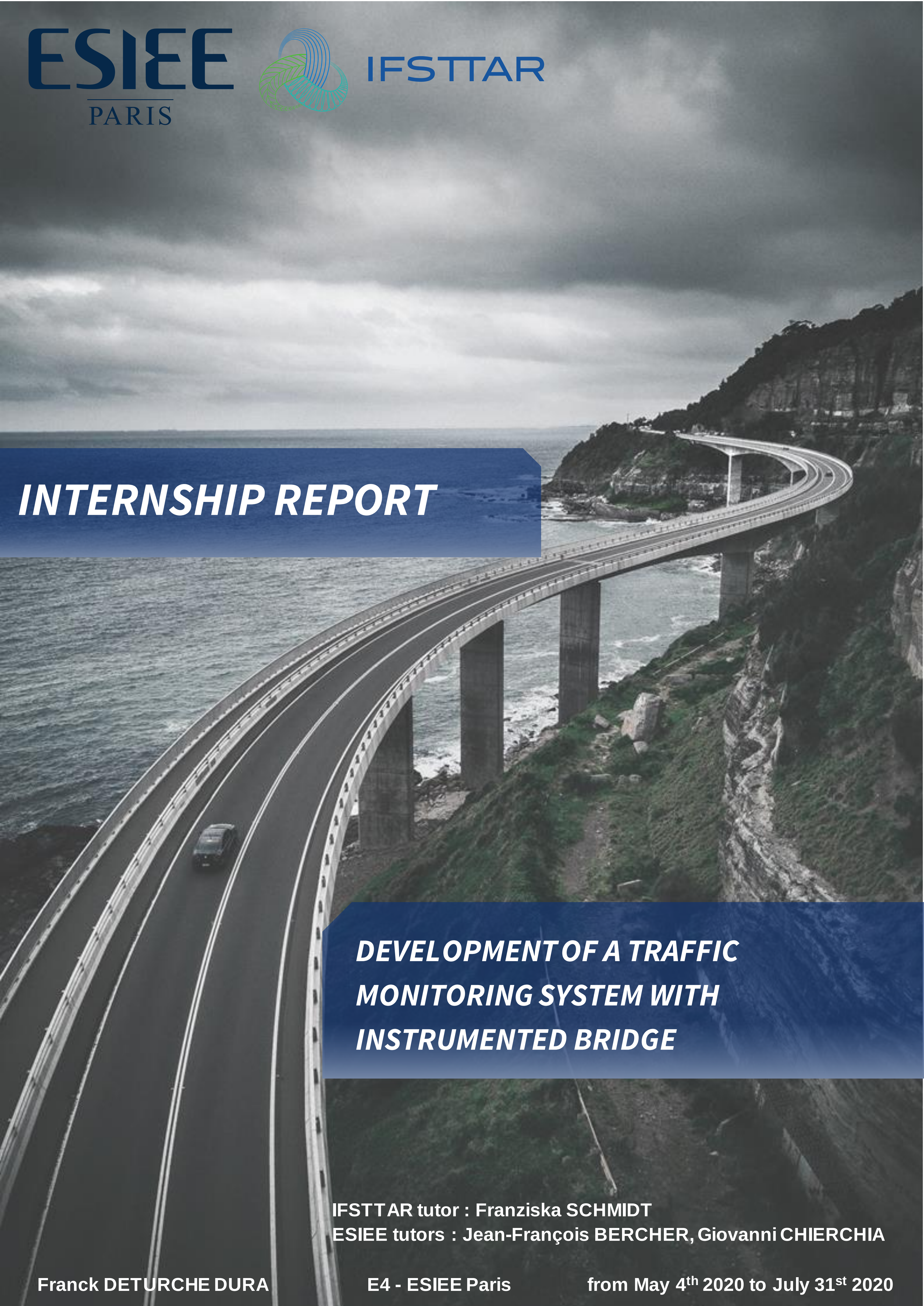


INTERNSHIP REPORT

DEVELOPMENT OF A TRAFFIC MONITORING SYSTEM WITH INSTRUMENTED BRIDGE

IFSTTAR tutor : Franziska SCHMIDT
ESIEE tutors : Jean-François BERCHER, Giovanni CHIERCHIA

Franck DETURCHE DURA **E4 - ESIEE Paris** **from May 4th 2020 to July 31st 2020**

Table of contents

Internship summary

French

Le pesage en marche par ponts instrumentés consiste à enregistrer les signaux de déformation ou de contrainte lors de passage de camions par des capteurs. Il est possible d'utiliser le modèle physique B-WIM (Bridge Weighing In Motion) pour estimer la ligne d'influence du pont puis leurs poids. Tel était le premier objectif du stage.
Plus précisément, il fallait estimer au mieux la ligne d'influence du pont à l'aide des camions de calibration puis l'utiliser pour estimer les poids des camions de trafic.
Ces calculs sont cependant sujet à incertitudes à cause de divers paramètres, liés à l'ouvrage ou aux mesures.
Une fois les résultats obtenus satisfaisants, le but était de retrouver les solutions de l'équation dynamique du pont, décrivant la ligne d'influence lorsqu'elles sont évaluées en son milieu, par physic based deep learning.
Diverses campagnes de mesure de pesage par ponts instrumentés ont été menées, avec enregistrement des signaux temporels et des données concernant les camions.
Nous avons exclusivement travaillé sur les données du pont de Senlis.

English

Weighing in motion by instrumented bridges consists of recording strain or stress signals when trucks pass through load cells. It is possible to use the physical model BWIM (Bridge- Weighing In Motion) to estimate the line of influence of the bridge and trucks weights. This was the first objective of the course.
More precisely, it was necessary to best estimate the line of influence of the bridge using the calibration trucks and then use it to estimate the weights of the traffic trucks.
However, these calculations are subject to uncertainties due to various parameters related to the structure or the measurements.
Once the results obtained were satisfactory, the goal was to find the solutions to the dynamic equation of the bridge, describing the line of influence when evaluated in its middle (of the bridge), by physics-based deep learning.
Various instrumented bridge weighing measurement campaigns were carried out, with recording of time signals and truck data.
We worked exclusively on the data from the Senlis bridge.

Company description

IFSTTAR (Institut Français des Sciences et Technologies des Transports, de l'Aménagement et des Réseaux) was created in 2011 after the merger of LCPC (Laboratoire Central des Ponts et Chaussées) and INRETS (Institut National de Recherche sur les Transports et leur Sécurité). This public establishment is jointly managed by the (French) Ministry of the Environment, Energy and the Sea and the Ministry of Higher Education and Research.

The institute is made up of six branches spread throughout France (Lille, Marne-la-vallée, Versailles, Nantes, Lyon, Marseille). Its head office is located in Marne-la-Vallée.
The institute is divided into five departments:
- MAST: Materials and Structures
- GERS: Geotechnics, Environment, Natural Hazards and Earth Sciences
- COSYS: Components and systems
- TS2: Transport, Health and Safety
- AME: Planning, mobility and the environment

As a major player in European research, IFSTTAR conducts research and expertise for ministries, administrations and companies such as EDF and EIFFAGE.
Researchers also respond to calls for projects from the ANR (French National Research Agency), ADEME, Horizon 2020, the European programme for research and innovation, etc. At the same time, IFSTTAR is committed to the transmission of scientific knowledge. IFSTTAR is also actively involved in initial and continuing research training by providing CIFRE grants to doctoral students and teaching activities with engineering schools and partner universities (ENTPE, ESIEE, ESTP, UPEM, etc.).
Certain missions are also a means of alleviating the difficulties of research funding. Indeed, only the salaries of employees and the maintenance of premises are now subsidized by the State.
The institute directs its research towards major societal challenges such as climate change and the improvement of public health. Thus, it uses its expertise in civil engineering, mobility and road safety to design and develop sustainable territories and promote eco-responsible mobility.

The objective is thus to promote the development of science and technology in various fields such as urban engineering, civil engineering, infrastructure, but also the mobility of people and goods.

The EMGCU (Experimentation and Modelling for Civil and Urban Engineering) was created in 2018 by merging the EMMS (Experimentation and Modelling of Materials and Structures) and SDOA (Safety and Sustainability of Engineering Structures) laboratories. It is part of the MAST department and specializes in the experimentation and modeling of any construction outside of traditional residential buildings. The laboratory conducts research and expertises on themes centred on the life cycle and safety of infrastructures.

Mission aims and context

In a context of constant improvement of technologies and an agreement with the DGITM (General Directorate of Infrastructures, Territories and the Sea) on the control of automated sanctioning of overloads, IFSTTAR has launched a project aimed at controlling the axle loads of heavy goods vehicles (HGVs). Various existing technologies are then implemented, tested, compared, including the one that consists in using bridges as scales to weigh traffic: B-WIM (Bridge-Weighing In Motion), i.e., weighing is carried out while moving by instrumented bridges.
This model was used in the first part of the course to estimate the influence lines and then the weights. Systems already exist today for monitoring and recording axle loads while driving on pavements, by installing sensors in the pavement. Nevertheless, B-WIM is a good alternative to other systems because it is transportable, easy to install, requires few human resources and its installation is safer and non-intrusive.
The principle is simple: in order to determine the axle loads of moving heavy goods vehicles (HGVs), the deformation of the structure is measured as a function of time using deformation sensors. The bridge then acts as a balance. (See Fig. 1)

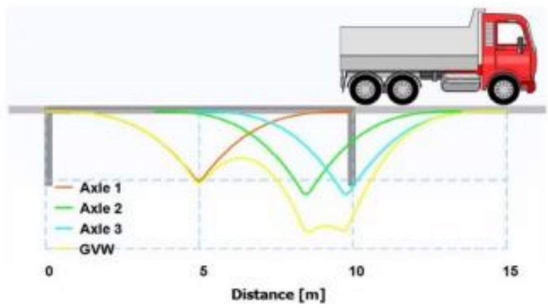


Fig.1 – Representation of deformations in B-WIM

This requires a preliminary calibration of the system with HGVs which weight and length characteristics are known, making it possible to first determine the transfer function between applied loads and measured deformations (known as the line of influence in material strength). The loads on the running axles are finally calculated by reverse calculation using the previously calculated transfer function.

The company that markets the system also provides software that allows the calculation of the loads on each axle to be visualized and indicates the trucks as being overloaded or not. However, according to IFSTTAR studies, the results of this reverse calculation do not comply with DGITM requirements (less than 5% error between actual and calculated weight, for axles, axle groups and the entire truck).
The first objective is therefore to go below this error threshold. When the results are satisfactory, deep learning will be used to calculate the solutions of the dynamic equation of the bridge (differential equation) in order to evaluate the theoretical line of influence or, conversely, to find characteristics of the bridge.

Work done and results obtained

Introduction and working environment

To achieve the objectives stated in the previous part, the work was done in ***python*** on a ***Linux Ubuntu 20.04*** distribution via ***jupyter notebooks (.ipynb)***. Many libraries were used, the most important being :

- ***NumPy,*** for the use of multidimensional arrays
- ***SciPy,*** for the implementation of complex mathematical functions (e.g. interpolation, minimization)
- ***matplotlib*** and ***seaborn,*** for curve/graph displays
- ***Scikit-learn,*** for machine learning algorithms
- ***DeepXDE*** and ***SCIANN*** (based on ***Tensorflow***), for PINN
- ***GitLab*** **and** ***Git*** : to version the code and share resources
- ***Functions*** and ***classes*** provided at the beginning of the internship

Files and data available are :

- ***1_intro.ipynb*** : Notebook briefly describing the B-WIM model.
- ***2_senlis.ipynb*** : Notebook with python functions for reading data, calibration and weight estimation.
- ***4_calibration.ipynb*** : Notebook containing different ways of estimating the line of influence.
- ***data*** : Root folder containing the truck signals recorded by the sensors.
- ***data/senlis/Calibration - Description.txt*** : File containing the information of the calibration trucks.

- ***script*** : Folder containing the script to convert the .events format to .CSV.

- ***bwim.py*** : script containing functions useful for calibration and weighing on the go

- ***plot.py*** : script containing the signal and data display functions.

- ***data.py*** : script containing the function to read the data.

- ***utils.py*** : script containing useful functions.

The files made available to us are the:

- ***.nswd :*** (proprietary format, results corresponding to the results rendered by the weight calculation system)
- ***.event :*** (proprietary format, results corresponding to the deformation signals of the individual PLs)

The course will take place as follows (Fig.2):

Fig.2 – Course of the internship

Data loading

In a first step, it was necessary to be able to use all the truck data available to us:

- Those in the calibration file

- Those in the traffic file

For this purpose, the existing function for loading the calibration trucks has been modified so that it also loads the traffic trucks.

It should be noted that the raw data are in .event format (proprietary format, results corresponding to the deformation signals of the various HGVs), requiring the axle weights to be specified in dictionary form. (See Fig.3). It was therefore necessary to implement this dictionary using a text file specifying the axle weights of the traffic trucks.

```
senlis_weights_modified = {
    ####Trafic####

    'PL1_2015-10-01-09-45-52-576':  np.array([7.35, 11.25, 8.20, 8.10, 8.25]),
    'PL2_2015-10-01-09-55-36-371':  np.array([7.65, 14.10, 6.05, 6.20, 6.30]),
    'PL3_2015-10-01-10-22-34-654':  np.array([7.40, 11.60, 8.35, 8.50, 8.50]),
    'PL4_2015-10-01-10-35-43-279':  np.array([7.60, 12.35, 7.80, 7.70, 7.80]),
    'PL5_2015-10-01-10-59-39-060':  np.array([7.35, 10.50, 9.30, 9.70, 9.45]),
    'PL6_2015-10-01-11-05-36-548':  np.array([7.30, 14.30, 6.80, 6.60, 6.70]),
    'PL7_2015-10-01-11-14-06-712':  np.array([7.05, 13.95, 7.25, 7.55, 7.90]),
    'PL8_2015-10-01-11-42-22-222':  np.array([6.40, 14.05, 6.85, 6.75, 6.65]),
    'PL9_2015-10-01-14-42-22-908':  np.array([8.20, 12.00, 8.35, 8.40, 8.35]),
    'PL10_2015-10-01-14-58-50-212': np.array([6.75, 12.45, 8.00, 8.10, 8.00]),
    'PL11_2015-10-01-15-00-57-826': np.array([7.15, 11.40, 8.05, 8.00, 8.45]),
    'PL12_2015-10-01-15-10-30-906': np.array([7.40, 10.80, 8.80, 9.05, 8.75]),
    'PL13_2015-10-01-15-19-48-718': np.array([8.15, 12.20, 7.30, 8.30, 8.80]),
    'PL14_2015-10-01-15-34-03-982': np.array([6.20,  5.05, 7.95, 7.65, 7.40, 7.65]),
    'PL15_2015-10-01-15-36-13-519': np.array([5.35,  8.25, 8.10, 7.75, 7.80, 7.60]),

    ####CALIBRATION####

    '2015-07-02-10-54-53-222': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-07-02-16-36-41-533': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-09-29-10-28-52-687': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-09-56-08-457': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-09-28-46-125': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-15-52-01-990': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-07-02-17-05-11-125': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-07-02-16-08-40-480': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-07-02-15-40-32-876': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-07-02-11-54-10-861': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-07-02-11-24-46-054': np.array([6.03315,  10.8891, 9.17235, 8.58375, 8.77995]), #semi_trailor
    '2015-09-29-15-23-14-921': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-14-55-32-562': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-14-28-15-093': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-14-00-38-369': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-12-17-33-320': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-13-33-05-960': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-11-50-31-328': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015
    '2015-09-29-11-08-45-998': np.array([6.16068,  10.8499, 9.08406, 6.98472, 7.0632]), #cal_sep_2015



}
```

Fig.3 - Definition of axle weights

The calibration HGV's each have the same number of axles and the same total weight.

The weights of the traffic HGV's are not supposed to be known. In this case they are known, so that they can be used for calibration if we lack calibration data.

The HGV's are stored as ***namedTuple*** "truck", containing the attributes of each HGV (see Annex 2 and 3) in NumPy arrays.

Calibration

Calculation of the line of influence from several trucks

Once the data was usable in a python script, calibration was the first task to be performed. Calibration is defined as the calculation of the influence line of the bridge from the data measured when the trucks pass. More precisely, we use the physical model B-WIM (Bridge-weighing-in-motion).

In the functions provided there was a set of functions allowing calibration from a single HGV. (See Fig.4)

```
def calibration(truck, length, l2_reg=None, tv_reg=None):# pour le moment aucune régularization
    A, b = prepare_least_squares(truck, length)#retourne T et y
    A, b = prepare_regularization(A, b, l2_reg, tv_reg)#Aucune régularization pour le moment
    influence ,_, _, _ = np.linalg.lstsq(A, b, rcond=None)# Retourne la solution des moindres carrés de ||y - Th|| avec h notre
ligne d'influence (donc donne h)
    influence_bundle = time_to_meter_interpolation(truck, influence)#
    return influence_bundle
```

Fig.4 – Calibration function for a single truck

Although theoretically the line of influence is unique at each point of the bridge, different factors (lateral distance of the HGV from the sensor, imperfect bridge structure, bridge wear, accuracy of measuring tools etc.) alter the results obtained. Calibration from a single HGV is therefore not well generalized. (See Fig.4 and 5)

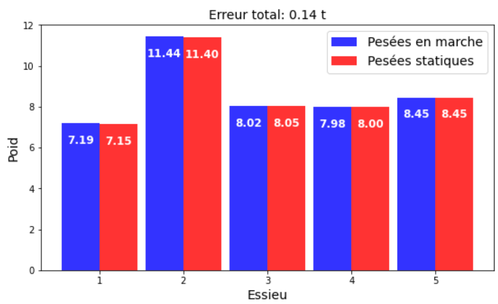


Fig.4 - Result of weighing in motion (WIM) on the calibration truck

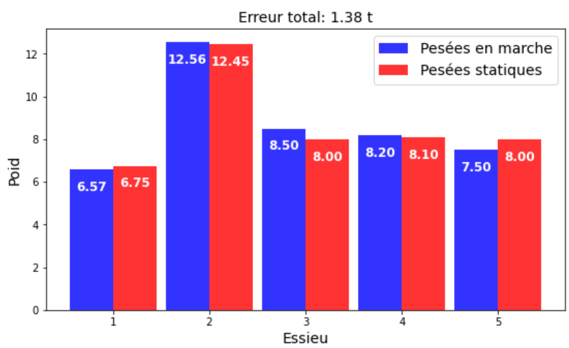


Fig.5 - WIM result on a truck other than the calibration truck

The results may vary further as traffic HGV's are likely to have more axles than calibration HGV's or a different total weight.
The major problem lies in the generalization of our results. This generalization is essential for the actual implementation of these methods and the achievement of the final objective (less than 5% accuracy error).

This observation led us to estimate the line of influence from several HGV's.

Mathematically, this amounts to moving from minimization for one truck (see Fig.6) to minimization for several trucks (see Fig.7).

$$minimize_{\mathbf{h} \in \mathbb{R}^M} \ \|\mathbf{T}\mathbf{h} - \mathbf{y}\|^2,$$

Fig.6 - Calculation of the line of influence for a truck

With :

- **T** = $\sum_{a=1}^{A} w_a \mathbf{D_a}$.

- **h** : line of influence
- **wa** : axle weight
- **Da** : Toeplitz matrix

$$\underset{\mathbf{h} \in \mathbb{R}^M}{\text{minimize}} \ \|\tilde{\mathbf{T}}\mathbf{h} - \tilde{\mathbf{y}}\|^2$$

Fig.7 - Calculation of the line of influence for several trucks

With :

$$\tilde{\mathbf{T}} = \begin{bmatrix} \mathbf{T}_1 \\ \cdots \\ \mathbf{T}_S \end{bmatrix} \qquad \tilde{\mathbf{y}} = \begin{bmatrix} \mathbf{y}_1 \\ \cdots \\ \mathbf{y}_S \end{bmatrix}$$

The implementation of a function that enables this consists of two parts:

- Modification of the function returning the signal (y) and the matrix T (as previously described) into a function returning a list of signals (y tilde) and the list of corresponding T (T tilde).

- Modification of the function performing minimization

Initially, the calibration used the method of least squares thanks to a function of the ***NumPy*** library: **numpy.linal.lstsqr()**

However, it cannot be used with matrices or multidimensional arrays whose dimension is strictly greater than 1.

With ***SciPy*** the setting of the minimization is much more interesting for this case. With the right minimization algorithm, it will then be possible to add constraints or limits for example. The main interest is to be able to minimize a function (which will return our sum) from its parameters (fixed and variable).

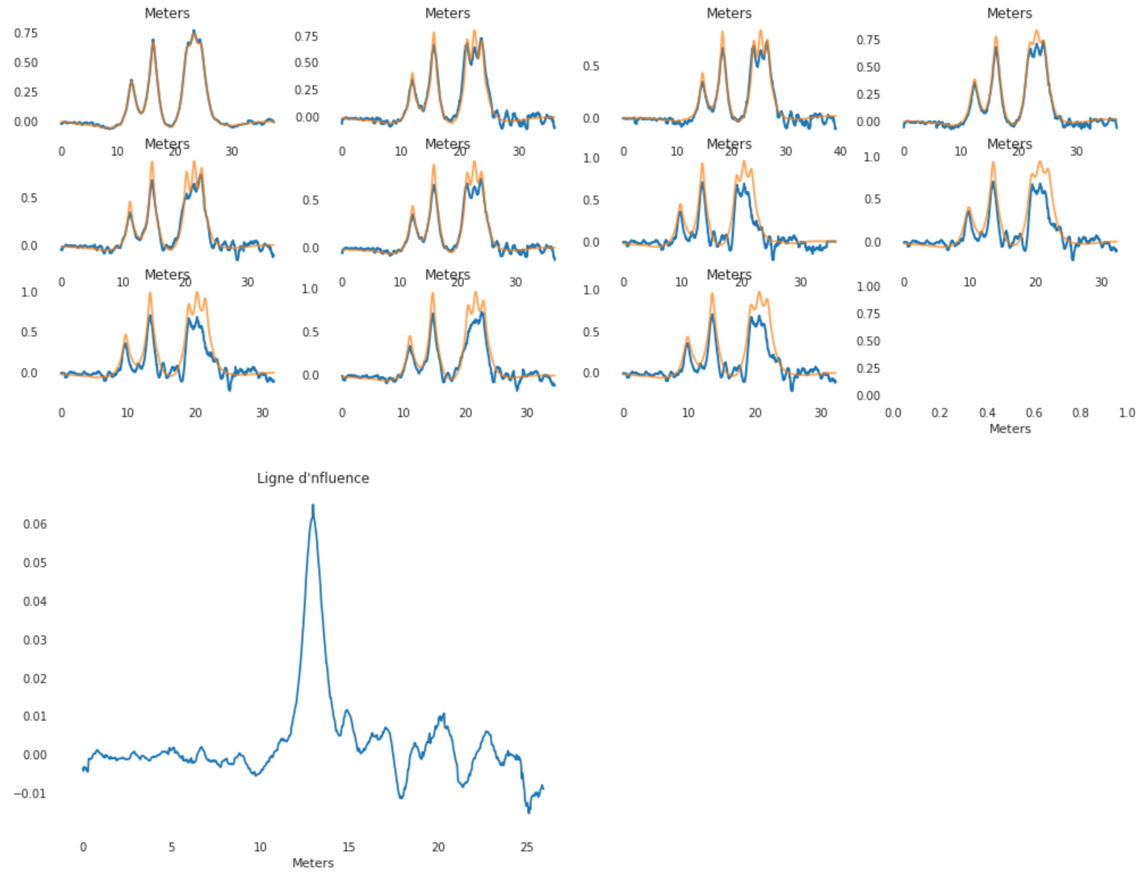


Fig.8 - Reconstruction of signals from calibration trucks via calibration (sensor 6) on the same assembly

The results for the other sensors are available on the Notebook N1 (in addition to the code set).
The results are slightly improved but are not satisfactory for actual practical use. There are therefore parameters that have not been taken into account and which it will be important to determine and then modify to obtain the desired results.
It was then possible to suspect the lack of precision of the sensors, a sampling frequency that was too low, or the noise level of the signals that could affect the quality of the results.

Weighting of signals from trucks by noise level

To ensure that the noise level of the signal was not the main cause of the previously observed problems, it was then decided to weight the signals by noise level in the minimization.

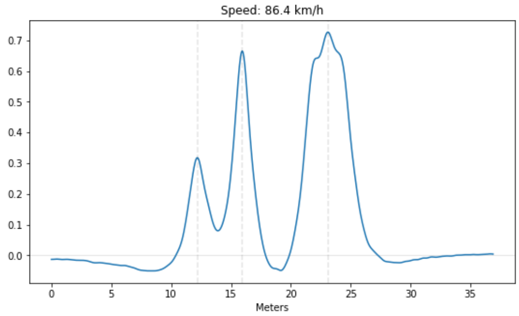


Fig.9 - Example of a truck signal

It was necessary to isolate the first five meters (see Fig.10), corresponding to the part of the signal with the most noise, calculate the noise level and deduce a coefficient for weighting.

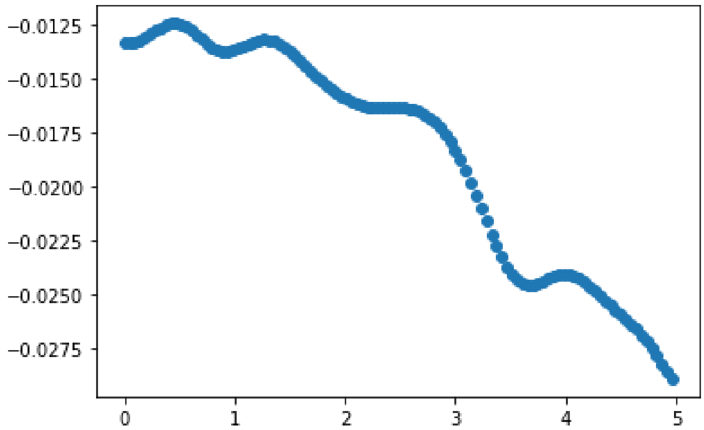


Fig.10 - Isolation of the first five meters of a signal

The observed slope supported the idea of using linear regression (see Fig.11) and then calculating the noise level from the RSS (residual sum of squares).

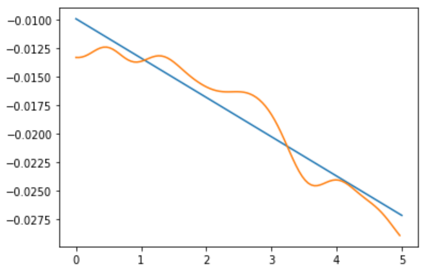


Fig.11 - Linear regression on this signal extract

A function then calculated the noise level as :

$$\text{Noise level (NL)} = \sqrt{RSS}.$$

With RSS the residual sum of squares of our regression

The signals were then weighted by 0.001/NB (0.001 to obtain coefficients centered around 1). (See Fig.12)

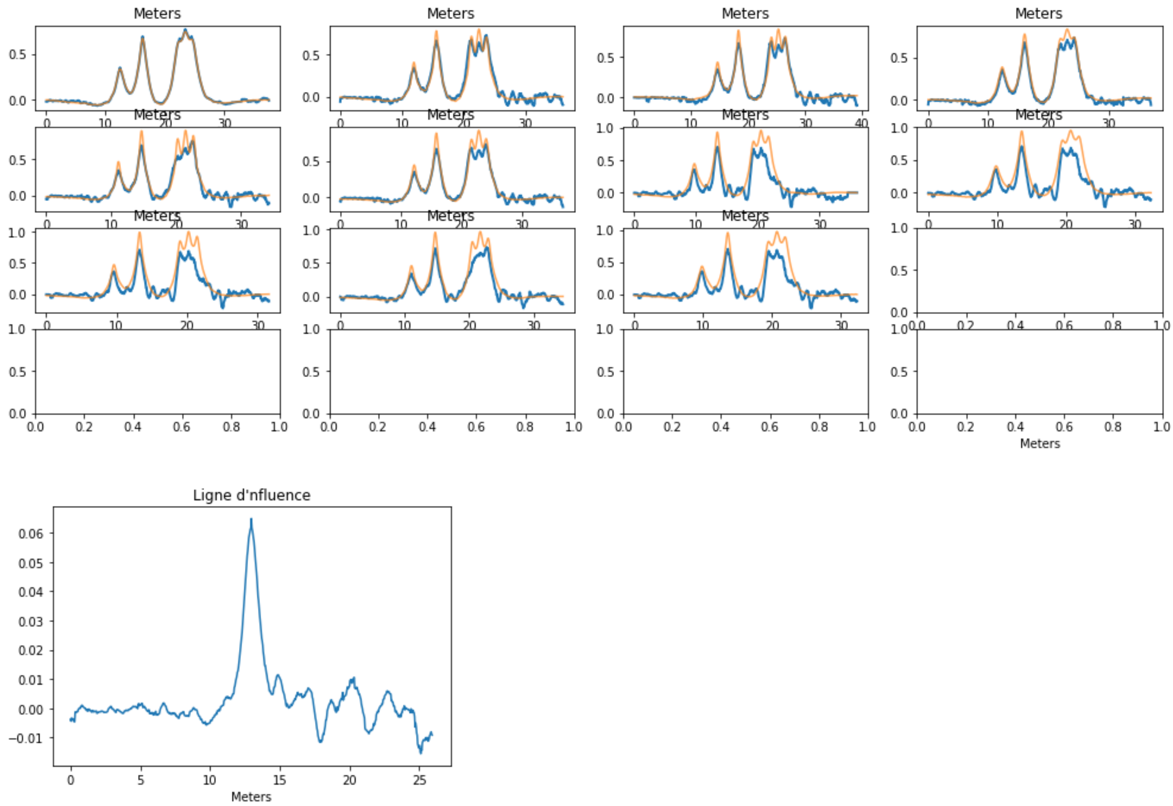


Fig.12 - Reconstruction of calibration truck signals via calibration on this assembly with noise level weighting of the signals

The noise level of the signals was therefore not the major problem. No difference was found.

Adjustment applied to the calibration for several trucks

After the implementation of a calibration for several trucks, the possible addition of a regularization to this method was also implemented, more precisely the Total Variation and L2 methods.

As a reminder :

- **Total Variation (TV)** : Assumes that noisy signals have a high total variation (i.e. the integral of the absolute gradient of the signal is high). According to this principle, the reduction of the total variation of the signal, provided it is close to the original signal, makes it possible to remove undesirable details while preserving important details.

- **L2** : Adjustment that adjusts the weights within a model downward in proportion to the sum of the squares of their values to counter over adjustment. This reduces the variance of our parameters by adding a bias.

Details of the code are available in Notebook N3.

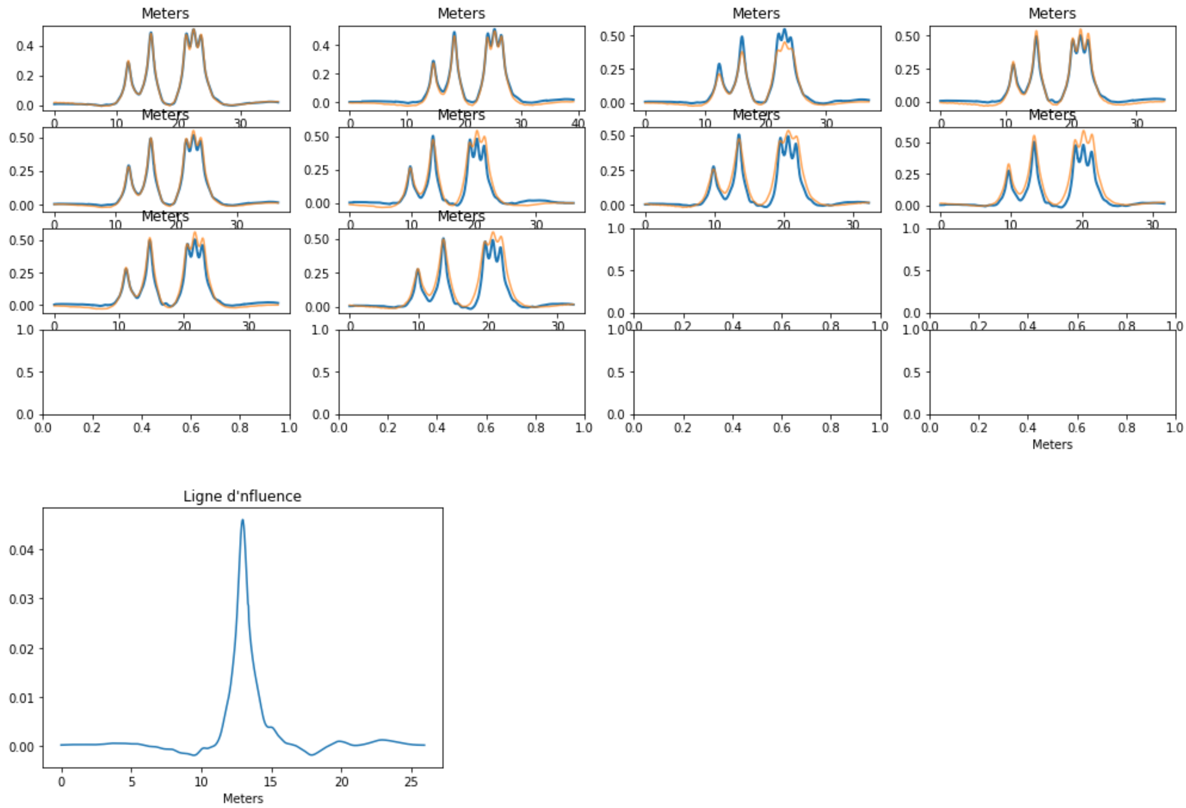


Fig. 13 - Reconstruction of signals from calibration trucks with calibration on this set and TV regularization

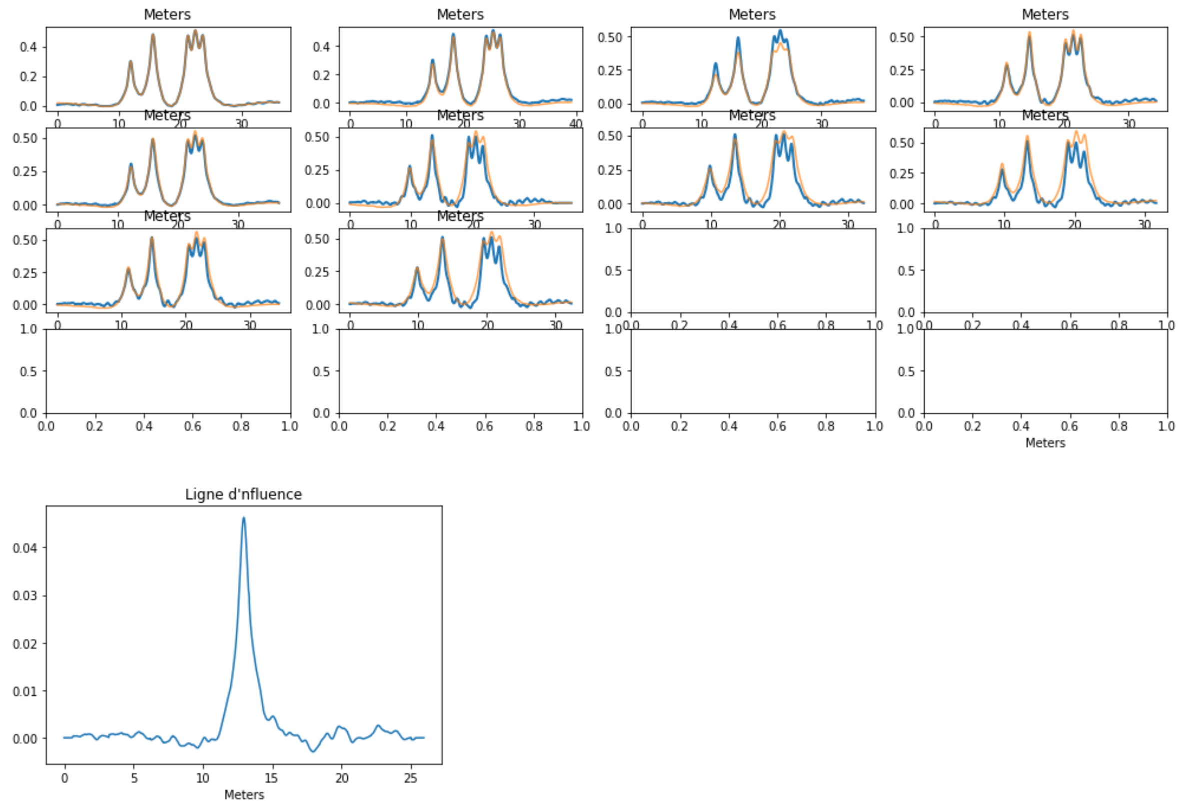


Fig.14 - Reconstruction of signals from calibration trucks with calibration on this set and L2 regularization

TV regularization brings a real added value on signal reconstruction. The L2 regularization does not significantly change the results.

Spline regularization has not been implemented although it can give interesting results.

Standardization of the signal scale

A hypothesis formulated concerning the problems observed on the results was the need to normalize the signal scale to overcome the different speeds of the calibration HGV's. Indeed, the variation of speed in the set of HGV's can cause scaling problems due to the sampling of the sensors.

Initially, the HGV signals were interpolated with the associated distance list (HGV speed * HGV time list). Then the signal had to be reconstructed from this function at the desired (interval) points. This step was performed in the function for preparing the signal lists and Toeplitz matrices useful for calibration.

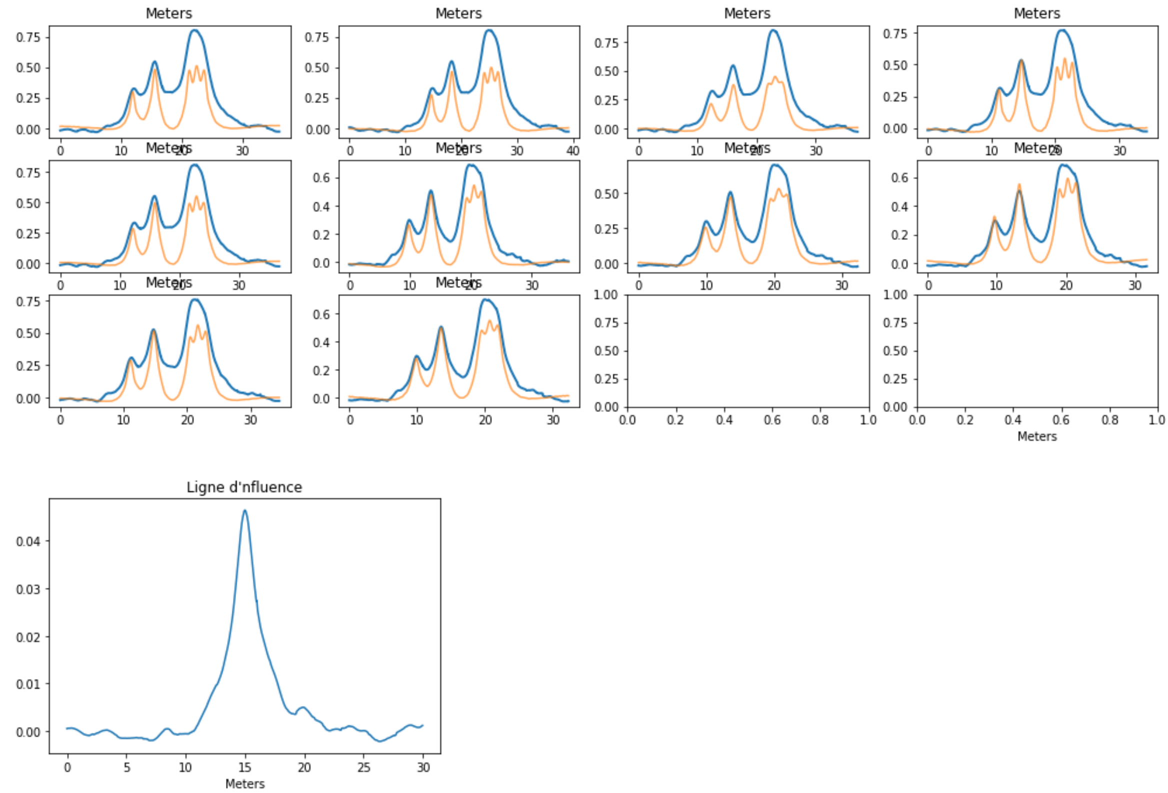


Fig.15 - Reconstruction of signals after calibration with signal normalization following the first method

The results obtained *(see Fig.15)* with this method of signal scaling normalization are also unsatisfactory. There seems to be a problem with the width of the main peak of the influence line. To make sure that the error is not due to a bad implementation, a second attempt at normalization has been undertaken, this time integrating the signal normalization in the creation of the ***namedTuple*** Truck.

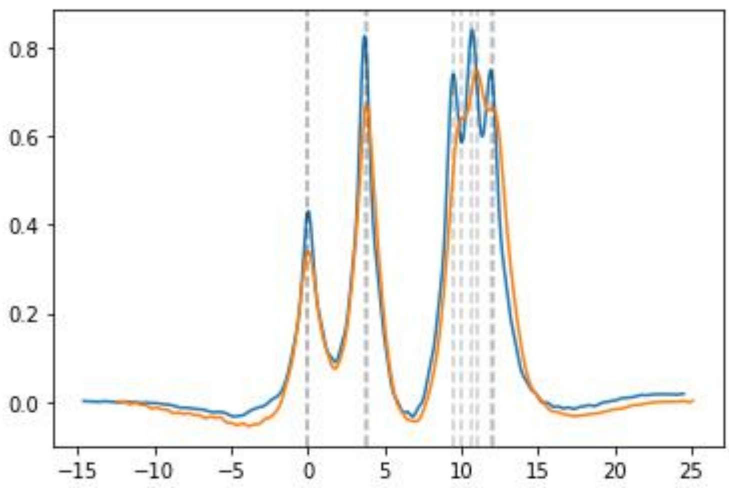


Fig.16 - Comparison of two signals of the new namedTuple Truck (integrated normalization)

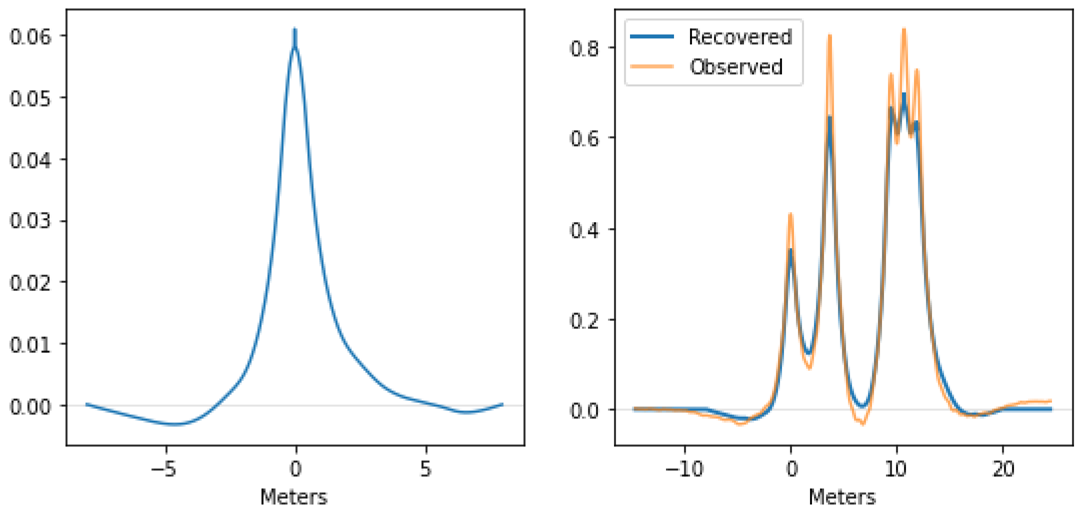


Fig.17 - Signal reconstruction after calibration with integrated signal normalization

In order to evaluate this method, the on-the-spot weighing of these results was carried out:

- On the calibration HGV's :

Mean absolute total error measured (in tons): **8,057 t**

- On traffic HGV's :

Average measured absolute total error (in tons): **5.23 t**

Scale is therefore not the solution for generalization

Calibration with sorting by truck speed

The finding on the previous results concerns the speed of the trucks. The results seem to deteriorate for high speeds.
This can be explained by sampling. However, it is complicated to explain why the line of influence calculated on trucks travelling at 70 km/h is not generalized.
It is calculated from the most reliable signals (low speed) and gives the best results on trucks running at this speed (successful generalization by speed range). Starting from the principle that there is only one line of influence per point (thus per sensor), there is a problem between theory and observations.

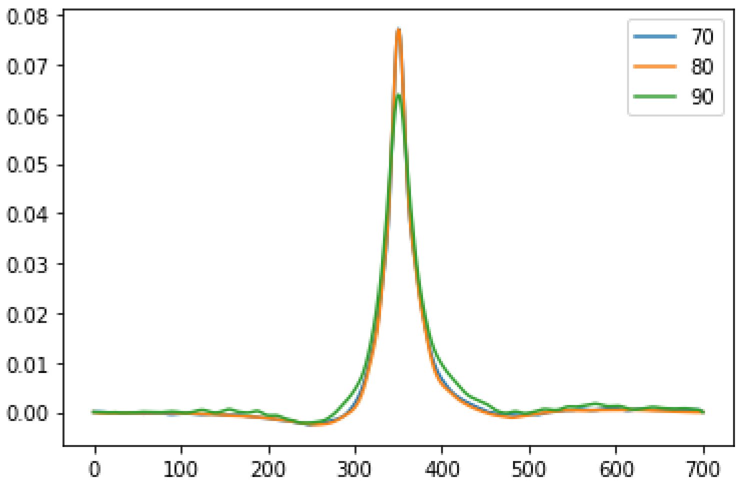


Fig.18 - Influence lines calculated for trucks travelling at different speeds (70 km/h, 80 km/h and 90 km/h)

There is an amplitude factor (see Fig.18) between the influence lines calculated on trucks travelling at different speeds.
Moreover, these factors do not seem to evolve linearly but rather seem to follow a parabolic curve (maximum amplitude for the influence line of trucks travelling at 80 km/h). This speed-dependent amplitude factor may be the key to further research.

Amplitude factors on the line of influence according to truck speed

In a first step, the amplitude problem was attempted to be solved directly in the minimization (of the calibration). This time, instead of minimizing a function dependent only on the line of influence, it was necessary to minimize a function dependent on the line of influence and a set of coefficients weighting the signals (and in an induced way their amplitude). This was implemented in a single table to simplify the code. At the beginning of the function, it was then sufficient to separate this table by two. One for the influence line and the other for the coefficients (as many as HGV's).

```
Alphas finaux :  [0.99999277 1.00023322 1.00039534 0.9997654  0.99971593 0.99924522
 0.99906682 0.99841124 0.99910643 0.99887528]
Alphas finaux :  [1.00000383 0.99985646 0.99971732 0.99898155 0.99959821 0.99952971
 0.99940864 0.99905438 0.9991801  0.99874882 0.99894142 0.99902309]
Alphas finaux :  [0.99995068 0.99973542 0.99840037 0.99910812 0.99913218 0.99871379
 0.9972225  0.99759873 0.99712686 0.99705102 0.99723104]
Alphas finaux :  [1.00000045 1.00004677 0.99991587 0.99953333 0.99963691 0.99951472
 0.9995326 ]
```

Fig.19 - Coefficients calculated by minimization on all calibration trucks for the four sensors

The calculated coefficients (see Fig.19) are all centered around 1, but do not correct the problem we observed. They do not correspond to the observed amplitude differences (up to 0.04).

It was then interesting to observe the influence of speed on the sampling of our signals to make sure that the problem was not due to the sensors. (See Fig.19)

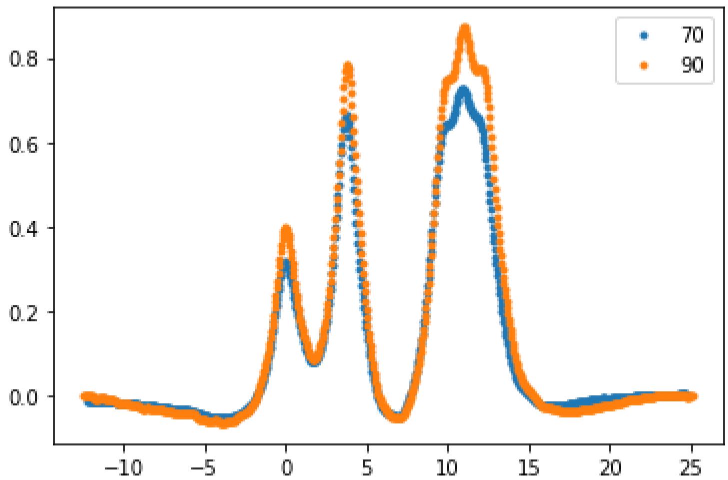


Fig.20 - Comparison of a signal from a truck travelling at 70 km/h with that from a truck travelling at 90 km/h

The speed seems to induce only a slight shift in amplitude and width (See Fig.20).

By taking the quotient of our maximum amplitudes (highest peak) between each HGV's signal, it was possible to observe a kind of linear correlation between speed and amplitude factor. (See Fig.21)
Below is an example obtained by using an HGV as a reference and calculating the quotient explained above in relation to the other trucks:

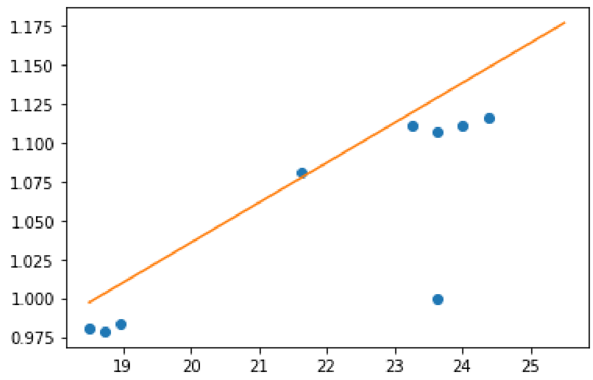


Fig.21 - Amplitude ratios of calibration trucks relative to another calibration truck according to speed

Calibration optimization

As most of the assumptions about the error on generalization proved to be inconclusive (mostly correct but not sufficient), it was then undertaken to optimize the calibration method to obtain the most accurate results possible, with performance indicators (metrics).

In a first step, a hypothesis was put forward concerning the calculation of the total weight of an HGV from the calculated line of influence and its signal. Indeed, the total weight can be described as the ratio of the area under the truck's signal and the area under the line of influence.
This method then allowed to add a constraint in the minimization function on the total weight. Unfortunately, this method, depending on the quality of the line of influence, returned calculated total weights that varied greatly, sometimes to more than 40% of the actual total weight. (See Fig.22 and 23)

```
Poids réel : 40.14256
Poids estimé :  41.255534643303626
Différence (en valeur absolue):  1.1129746433036232


Poids réel : 40.14256
Poids estimé :  40.8933289690534
Différence (en valeur absolue):  0.7507689690533965


Poids réel : 40.14256
Poids estimé :  38.505897396720684
Différence (en valeur absolue):  1.6366626032793192
```

Fig.22 - Total weights calculated by this method by calibration on the truck

```
Poids réel : 43.3
Poids estimé :  52.92951082380425
Différence (en valeur absolue):  9.629510823804253


Poids réel : 43.05
Poids estimé :  52.67940088082072
Différence (en valeur absolue):  9.629400880820725


Poids réel : 44.800000000000004
Poids estimé :  53.30839762825562
Différence (en valeur absolue):  8.508397628255615
```

Fig.23 - Total weights calculated by this method by calibration on all calibration trucks

Nevertheless, an attempt was made to incorporate this constraint into the minimization with intervals around the estimated total weight that are more or less large. The variations in the quality of the results did not allow a stable method to be established.

The differences between the lines of influence calculated on different load cells were then observed. (See Fig.24)

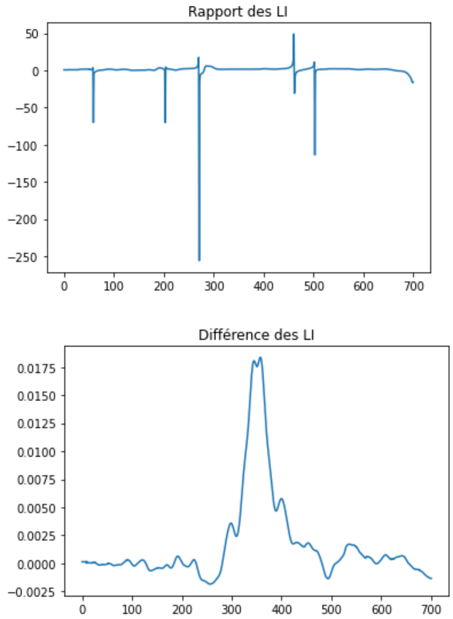


Fig.24 - Ratio and difference of the influence lines calculated on all calibration trucks on sensors 3 and 4

Only an amplitude factor between the lines of influence is observable (normal).

In order to accurately assess performance, (metric) measurement indicators had to be set up:

- A metric on signal reconstruction: Euclidean distance between reconstructed signal and real signal. (See Fig.25)

- Another on weighing on the move: total and axle errors.

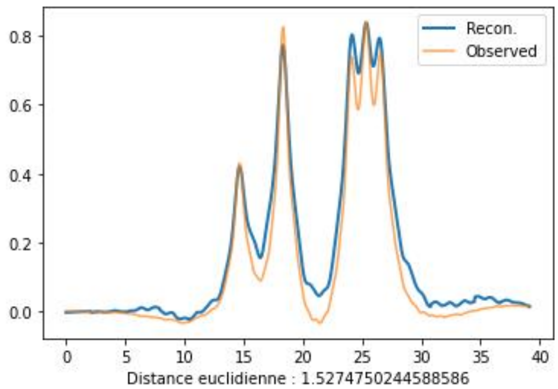


Fig.25 - Example of Euclidean distance between real and reconstructed signal

These metrics were then evaluated on all the calibration PLs for each sensor, to determine the sensors with the best signals:

- Mean Euclidean distance on sensor 3: **3.39**

- Mean Euclidean distance to sensor 4: **3.24**
- Mean Euclidean distance to sensor 6: **1.87**
- Mean Euclidean distance to sensor 7: **4.79**

It appears that sensor 6 (certainly thanks to its positioning on the roadway) is the one to be used for the rest of the calculations.
It was then attempted to resample the line of influence (rather than the signals, seen in the normalization part). The ***resample*** function of the ***scipy.signal*** library was used for this purpose. (See Fig.26)

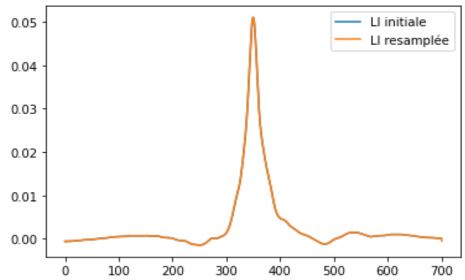


Fig.26 - Example of re-sampled LI (on 5000 points instead of 701 and then rescaled)

The results (on the criterion of the Euclidean distance between reconstructed and real signal) were not improved (or only slightly).
Finally, the calculation of a mean line of influence between two sensors in the hope of compensating for the problem of the lateral position of the trucks in relation to the sensors was also undertaken. Only the creation of sets composed of trucks from different sensors was necessary. (See Fig.27).

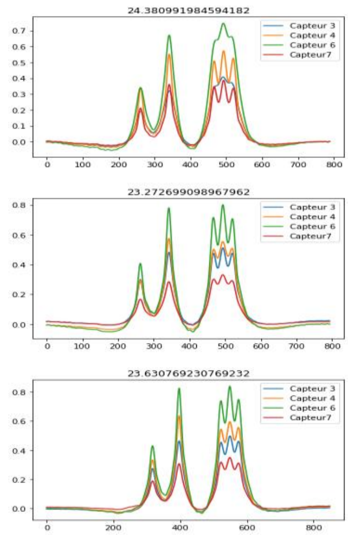


Fig.27 - Examples of calibration truck signals for our four sensors

Sensors 4 and 6 seem to be the ones to keep to get the 'strongest' signals.

Here are some results to compare:

- Average Euclidean distance (reconstructed/actual signal) sensor 4 on calibration trucks with total regularization variation: **1.74**

- Average Euclidean distance (reconstructed/actual signal) sensor 4 on traffic trucks with total regularization variation : **1.48**

- Average Euclidean distance (reconstructed/actual signal) sensor 4 and 6 on calibration trucks with total regularization variation : **2.87**

- Average Euclidean distance (reconstructed/actual signal) sensor 4 and 6 on calibration trucks with total regularization variation : **2.28**

The results thus make it possible to discard this track as well.

After having carried out these various tests, the only certainty was that it was necessary to use sensor 6 for the continuation of the calculations. The rest of the experiments did not give any major solution/research track.

Energy conservation

One idea to constrain minimization during calibration (as with the total weight) was to observe a kind of energy conservation between the PLs, describing the energy as :

$$\boldsymbol{E} = \sqrt{\boldsymbol{area}}(\boldsymbol{signal}^2)$$

The energy does not seem to be conserved and seems to evolve according to the speed of the HGV :

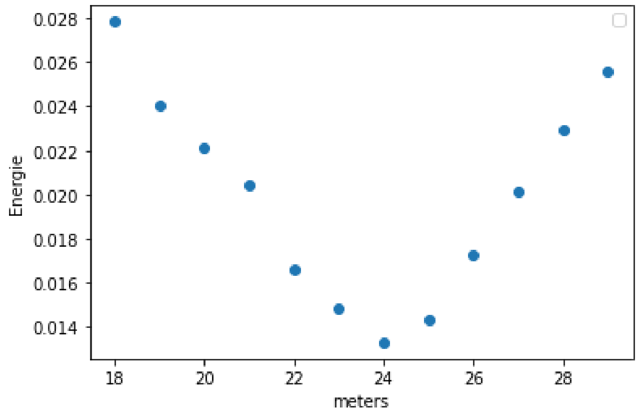


Fig.28 - Evolution de l'énergie selon la vitesse

Attempts have been made to weight the signals useful for calibration by these factors, without result.

This track has not been further investigated but may be worth pursuing in the future.

Interpolation of line of influence according to truck speeds

At this stage, the major problem represented by these amplitude factors between the lines of influence according to the speed of the HGV 's had no solution.

It was then undertaken to interpolate a set of lines of influence, calculated for each HGV, with their speeds.
Initially, the HGV 's were sorted by speed with the idea of finding a law of evolution of the lines of influence according to the speeds.
However, the little data available did not allow this to be done correctly.
The interpolation was therefore carried out from each truck individually.
By taking all the calibration HGV's, a function was obtained giving the line of influence as a function of speed (in m/s), whose amplitude varied. (See Fig.29)

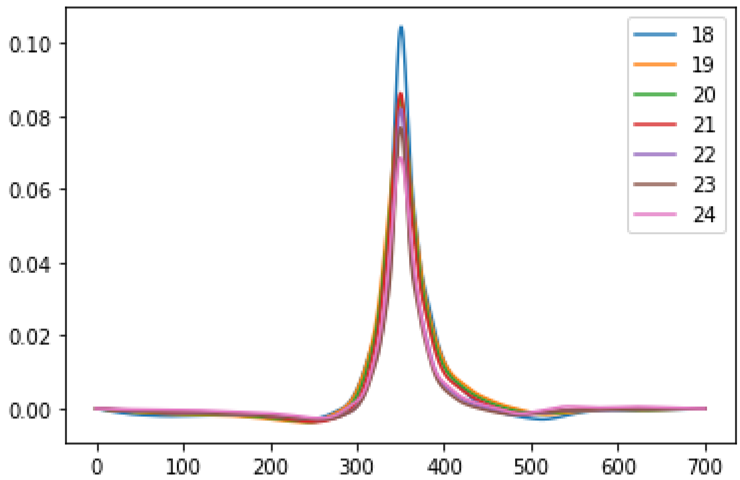


Fig.29 - Line of influence interpolated according to truck speed (from 18 to 24 m/s)

Much more satisfactory results were obtained, but the increase in speed continued to seriously degrade the results. (See Fig.30)

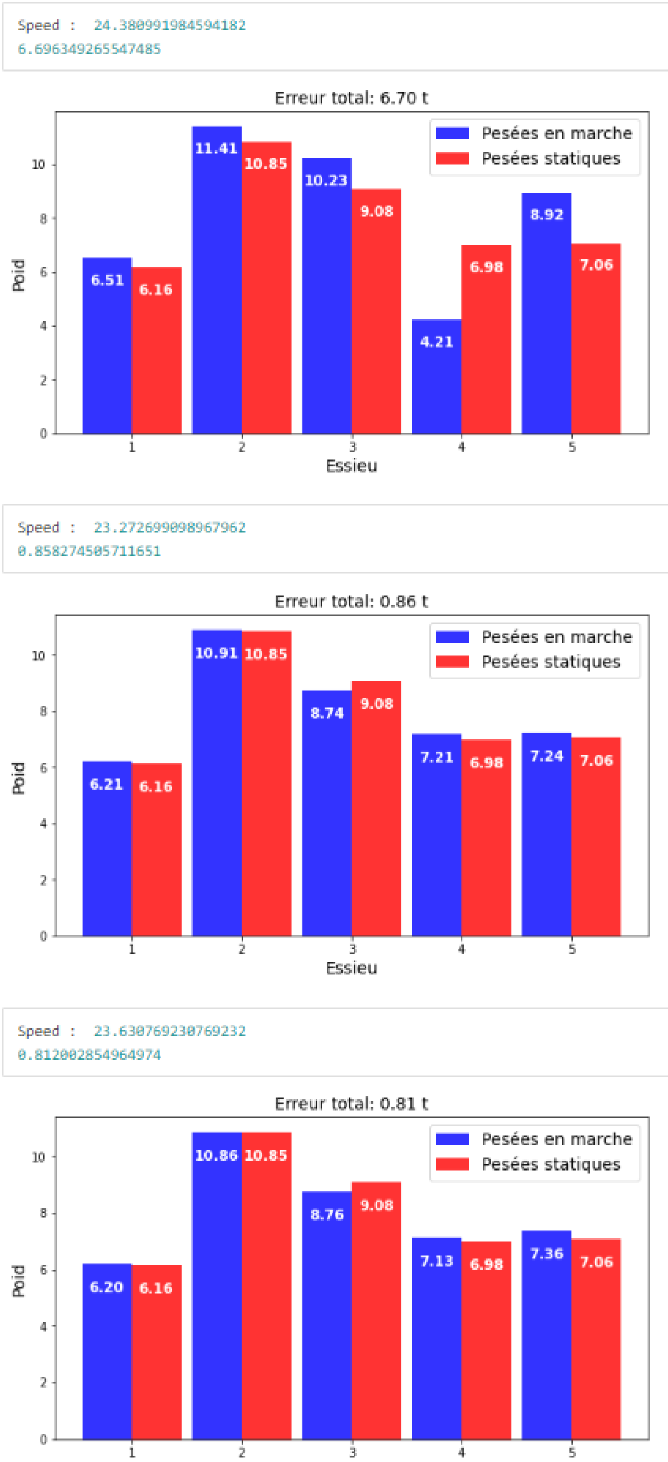


Fig.30 - Interpolation results on on-the-fly weighing for different speeds

This problem could be due to the lack of high-speed (more than 24m/s) trucks in the calibration set.
The traffic HGV's had much higher speeds on average than the calibration HGV's.
As the weights of the traffic HGV's in the traffic set were known, some of them were used for interpolation, in addition to the calibration HGV's. (See Fig.28)
The best results (without further optimization) were obtained this way.

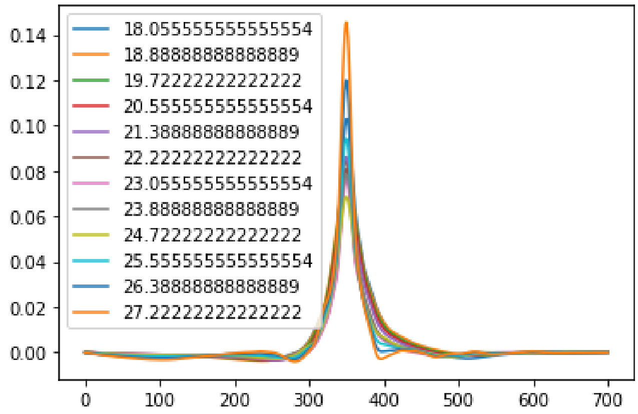


Fig.31 - Interpolation of the LI according to speeds with calibration trucks and some traffic trucks travelling at more than 24 m/s

- Average total error (in tons) on calibration trucks: **1.78 t**

- Average total error (in tons) on traffic trucks: **2.78 t**

Given that the total weight of the HGVs was mostly close to 41 tons, the results corresponded to an error of plus or minus 4% for calibration HGVs and 7% for traffic HGVs.

To improve the results, various experiments were carried out in addition.
The interpolation functions were reimplemented manually to compare them and keep the best one.

- **Linear interpolation reimplemented**

Average total error (in tons) on calibration trucks: **116 t**

- **Re-implemented polynomial interpolation**

Average total error (in tons) on calibration trucks: **3.2 t**

Average total error (in tons) on traffic trucks: **21.54 t**

- **By machine learning :**

It was also attempted to use a regression ridge to predict the evolution of the line of influence according to speed. (See Fig.32)

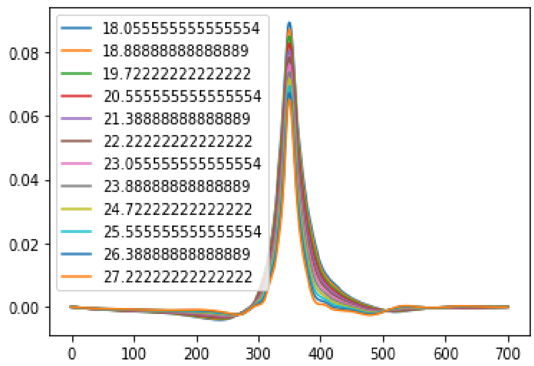


Fig.32 - Interpolation by regression ridge regression of the LI according to velocities

Average total error (in tons) on calibration trucks: **3.73 t**

The interpolation worked almost perfectly for speeds between 18m/s and 23 m/s but the results varied significantly for higher speeds.

The maximum amplitudes of the lines of influence interpolated according to the velocities were then observed. (See Fig.33)

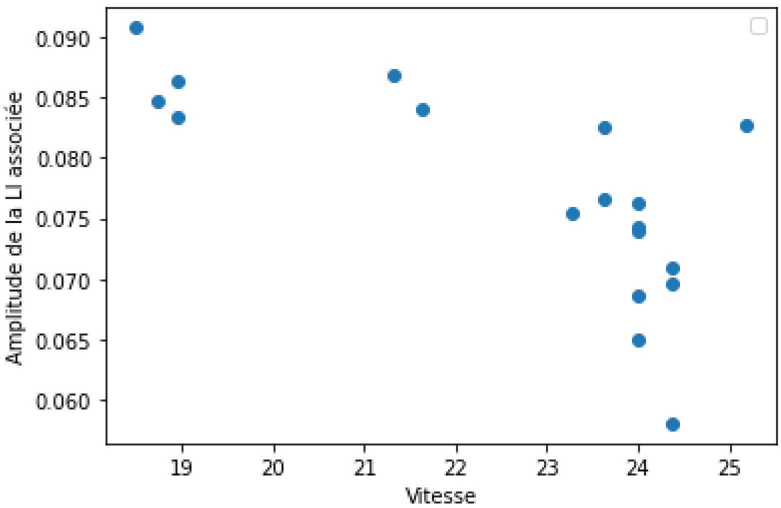


Fig.33 - Amplitude maximums of the LI interpolated according to the speeds

Around 23-24 m/s, there is a lot of disparity, so interpolation is complicated at this level. This phenomenon may be due to our small amount of data, which is therefore not necessarily representative of reality. With a much larger amount of data, this method would certainly give better results (as a reminder, data from 25 HGV were usable).

To finish with interpolation, a 2D interpolation was implemented, to determine the evolution of the line of influence as a function of distance and time. This did not work.
(All the results of the report are observable in detail in the Notebooks).

In addition, an idea was put forward about calibration according to the total weight of the HGV's. If the weights of the HGV's used in calibration varied significantly, they would have to be sorted by weight to make different calibrations. To automate this task, a clustering algorithm (unsupervised learning) was implemented with ***sklearn*** to sort the HGV's according to their list of axle weights (in three categories). (See Fig.34)

```
Poids total :  20
Peu chargé


Poids total :  22
Peu chargé


Poids total :  24
Peu chargé


Poids total :  26
Peu chargé


Poids total :  28
Normal


Poids total :  30
Normal


Poids total :  32
Très chargé


Poids total :  34
Très chargé


Poids total :  36
Très chargé


Poids total :  38
Très chargé
```

Fig.34 - Clustering results in three categories according to axle weights

Weighing in motion (WIM)

First estimation of weights

Once the calibration for several trucks was functional, it was necessary to implement the functions allowing weighing in motion in order to observe the results of interest to us (although in the previous results were already visible from the WIM results). Until then, only the criterion of the Euclidean distance between reconstructed signal and observed signal was used (although we have already observed WIM results in the previous section for questions of understanding.
Still using the B-WIM physical model, the weights are isolated starting from the line of influence (hence the importance of an efficient and accurate calibration).
By using the same notations as for the minimization corresponding to the calibration, the running weight is:

$$\mathit{minimize}_{\mathbf{w}\in\mathbb{R}^A} \ \|\mathbf{H}\mathbf{w}-\mathbf{y}\|^2$$

With:

$$\mathbf{H} = [\mathbf{D}_1\mathbf{h} \quad \mathbf{D}_2\mathbf{h} \quad \dots \quad \mathbf{D}_A\mathbf{h}]$$

(see page 9 for other ratings)

Several cases are then to be differentiated:

- When the calibration and the deconvolution is done on the same HGV, the result is almost perfect (as observed previously).

- When the calibration is performed on one HGV and the deconvolution is done on the other HGV's, the results vary greatly. The phenomenon of degradation of the results according to the speed is also visible. (See Fig.35 and 36)

Fig.35 - Result on the WIM with calibration on a calibration truck with speed close to the truck

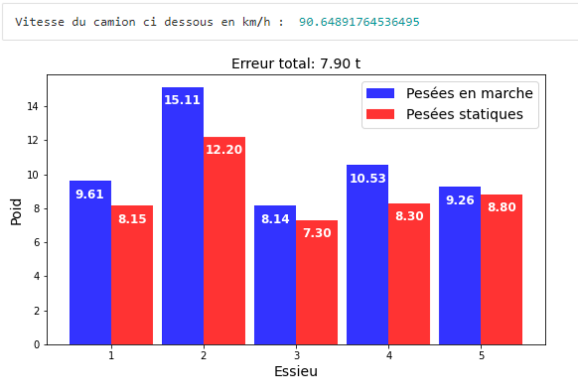


Fig.36 - Result on WIM with calibration on a calibration truck with speed away from the truck

Average total error (in tons) on calibration trucks: **12,156 t**

Average total error (in tons) on traffic trucks: **5.74 t**

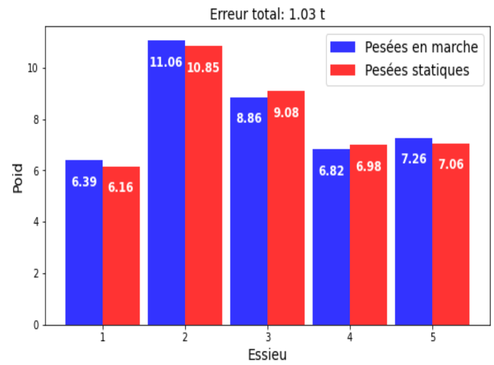
● When the calibration is performed on all calibration Hgv's, the results are slightly better but vary in the same way.

Average total error (in tons) on calibration trucks: **12,074 t**

Average total error (in tons) on traffic trucks: **5.68 t**

As the results seem to depend on speed (but standardization did not bring the hoped-for solution), the trucks were sorted into three speed groups (70 km/h, 80 km/h, 90 km/h).
The results were significantly better. Sorting by speed gives satisfactory results (total errors in tons). The lower the speed, the better they are (certainly due to the sampling of the sensor).

- **For 70 km/h :**




```
8]:
Poids total mesuré : 40.39266786101596
Poids total réel 40.14256
Vitesse du camion ci dessous en km/h :  68.2668670831572
```

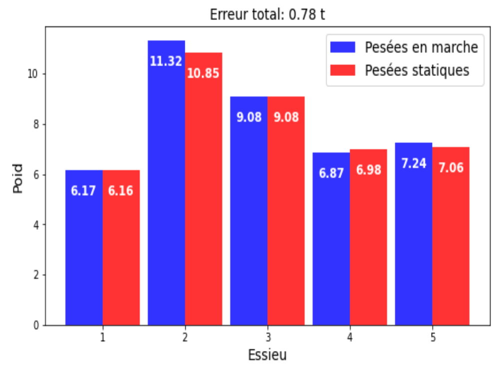
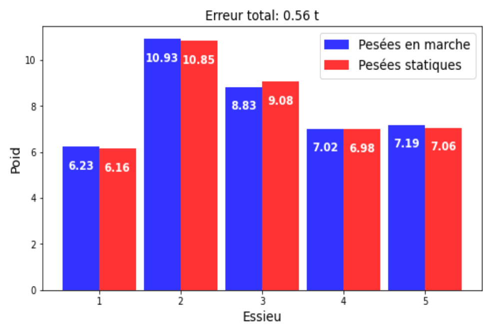


Fig.37 - WIM on trucks travelling at 70 km/h after calibration on trucks travelling at this speed

- **For 80 km/h :**




```
9]:
Poids total mesuré : 40.20734570631327
Poids total réel 40.14256
Vitesse du camion ci dessous en km/h :  76.8000296453673
```

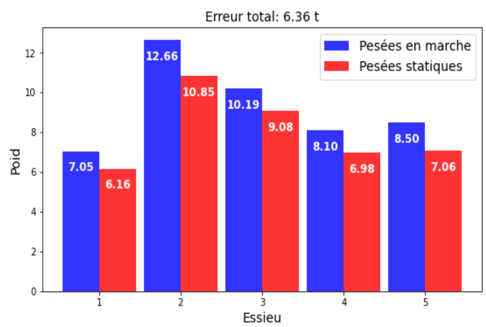
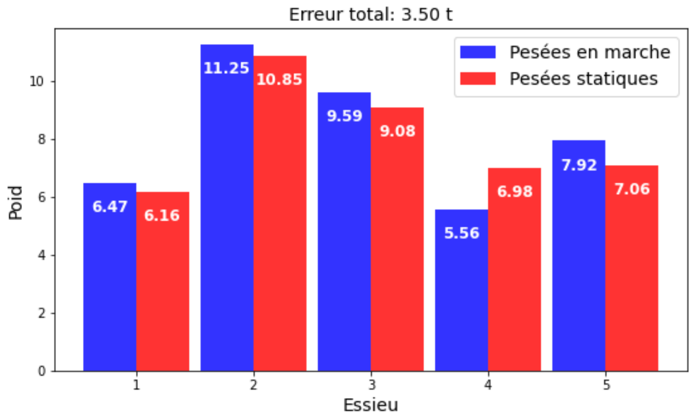


Fig.38 - WIM on trucks travelling at 80 km/h after calibration on trucks travelling at this speed

- **For 90km/h :**




```
Poids total mesuré : 40.7846005302306
Poids total réel 40.14256
Vitesse du camion ci dessous en km/h :  83.78171675628467
```

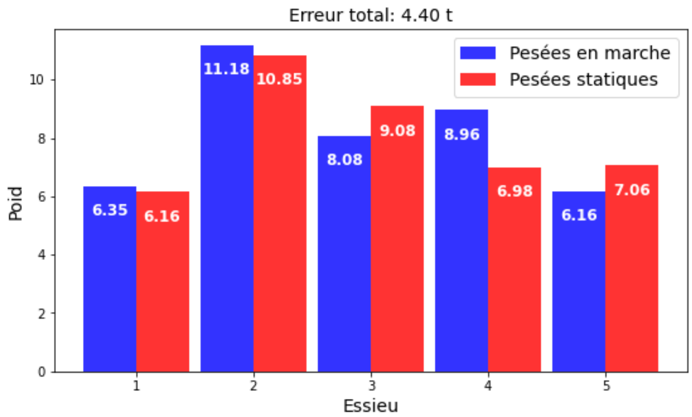


Fig.39 - WIM on trucks travelling at 90 km/h after calibration on trucks travelling at this speed

Speed plays a major role, as noted in the calibration section.

Peak shifts on signals

An additional idea to improve the results was the finding that the sampling frequency could lead to a shift in the actual position of the signal peaks (corresponding to the axle passes over the sensors). It was then conceivable to shift the peaks of the HGV's over a certain interval and then keep the best combination.

- Global offset

Initially, the peaks were all shifted by the same amount to corroborate this idea. (See Fig.40)

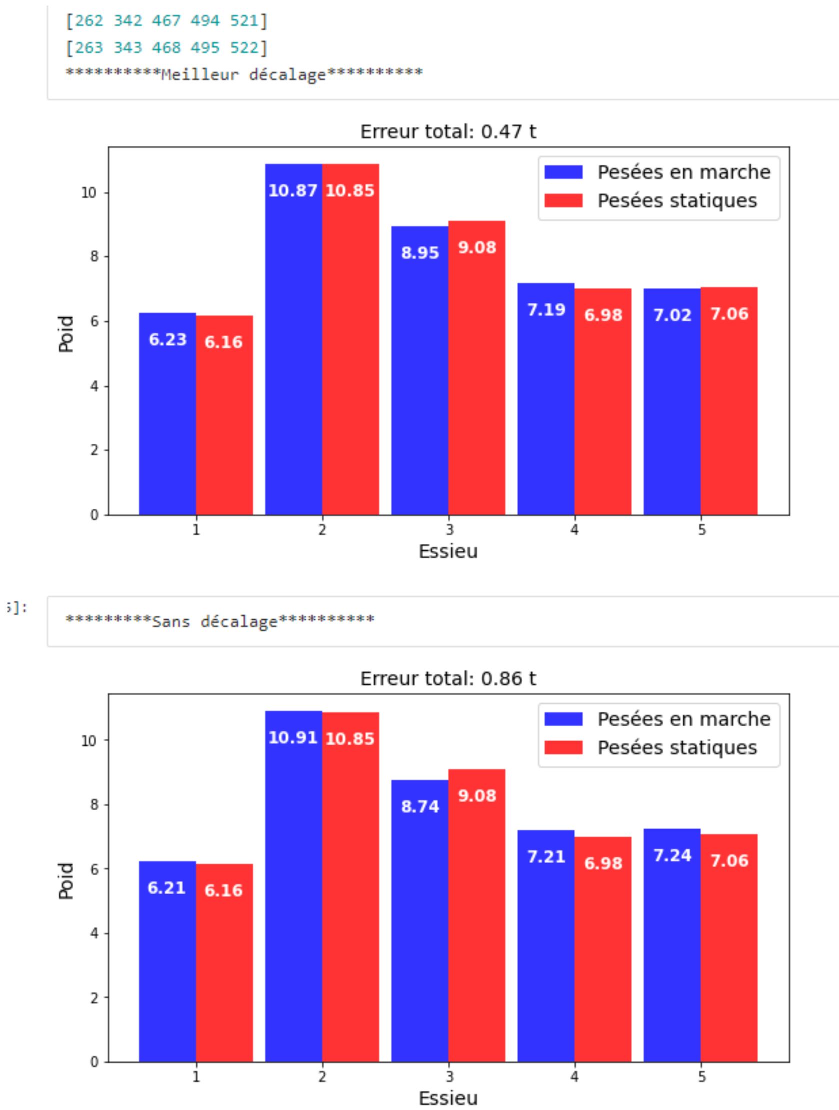


Fig.40 - Comparison of WIM with and without global shift

The overall gap is already making a significant improvement.

Total error (in tons) on calibration trucks: **1.24 t**

Average total error percentage on calibration trucks: **3%.**

Total error (in tons) on traffic trucks: **2.28 t**

Average total error percentage on traffic trucks: **5.56%.**

- Local offset

Each peak was then shifted independently of the others. In order not to overburden the calculations and to obtain results in a reasonable time, we reduced the offset over an interval centered on each peak (interval of [peak -2, peak+2]), which corresponded to an already high number of combinations.
The results, which take a long time to compute, have been further improved (problem of saving the results, we would have to restart the computations on a big machine).

A variant has been implemented to allow a shift of the limits if the found combination has one or more values at the limits of the shift interval.

To overcome the computation time problem, the ***nevergrad*** library has been used to quickly find a local minimum in the set of combinations described above. The results are therefore not reproducible (local minimum) and the algorithm returns the initial peaks in the worst case if no improving combination has been found.

The evaluation criterion was also modified, using the Euclidean distance between reconstructed and measured signal rather than the weight error. This will subsequently allow this method (peak shift) to be deployed on HGV's with unknown weights.

By including this method to the calibration function (use of shifted peaks rather than initial peaks), we obtain (calibration and deconvolution on the same HGV) :

Total error (in tons) on the calibration trucks: **0.369 t**

Average total error percentage on calibration trucks: **0.25%.**

Total error (in tons) on traffic trucks: **0.171 t**

Average total error percentage on traffic trucks: **0.018%.**

A problem encountered on some HGV's was that they could not be used in the calculations because they had fewer peaks than axles. This was due to poor signal measurement by the sensors. (See Fig.41)
We then used this peak shift principle to find the missing peak(s) in case we encountered this problem.

So we just had to create a list with the right number of axles and test different values and then keep the best one (always on the signal reconstruction criterion). (See Fig.42)

```
Peaks : [260 340 493]
Weights :  [ 6.16068 10.8499   9.08406  6.98472  7.0632 ]
```

```
&lt;IPython.core.display.Javascript object&gt;
```

```
Peaks : [275 358 488 508 531]
Weights :  [6.2  5.05 7.95 7.65 7.4  7.65]
```

Fig.41 - Examples of unusable trucks for our calculations

```
Peaks : [260. 340. 416. 493. 503.]
Weights :  [ 6.16068 10.8499   9.08406  6.98472  7.0632 ]
[ 5.24729619 10.0851535   9.37095913  6.8146071   8.02487522]
```

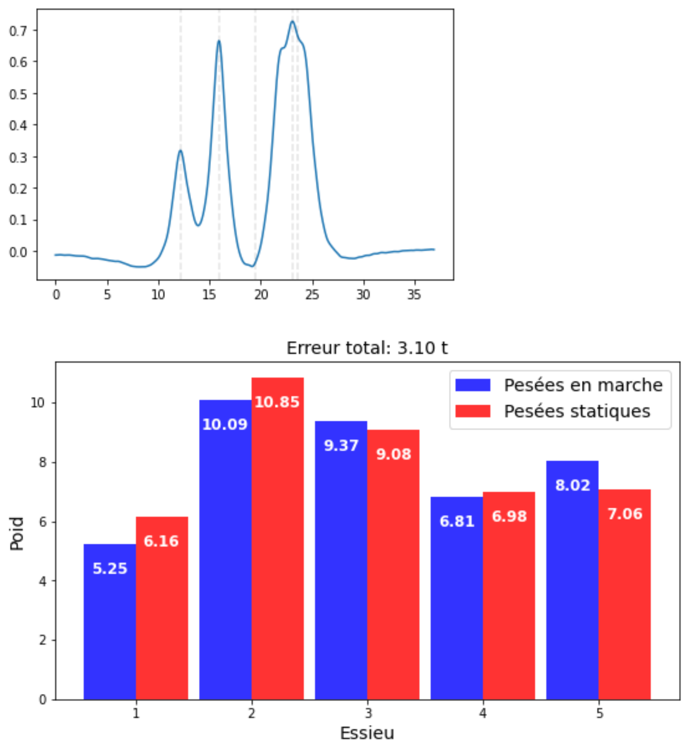


Fig.42 - Example of WIM result on an initially unusable truck

To simplify the calculation and avoid looking for the best results on all possible peak positions (i.e. to know in advance the relative position of the missing peak), the use of the peak width criterion was used to see if the widest could correspond to two (or more) superimposed peaks.
This criterion was not sufficient. (See Fig.43)

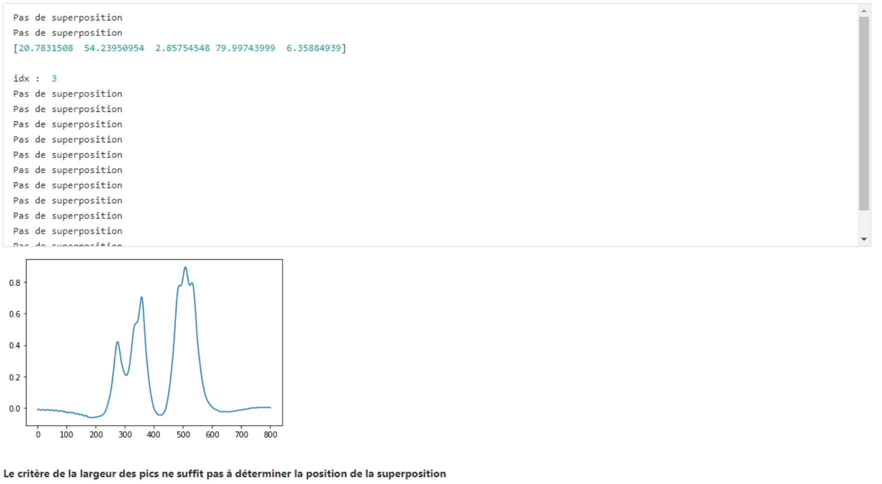


Fig.43 - Testing the use of the Broadest Peak Criterion

The principle of exhaustive testing of peak positions was used on signals with no anomalies (a kind of autonomous method).

Always on the criterion of the reconstructed signal and with a calibration on each HGV, almost perfect results were obtained. (See Fig.44)

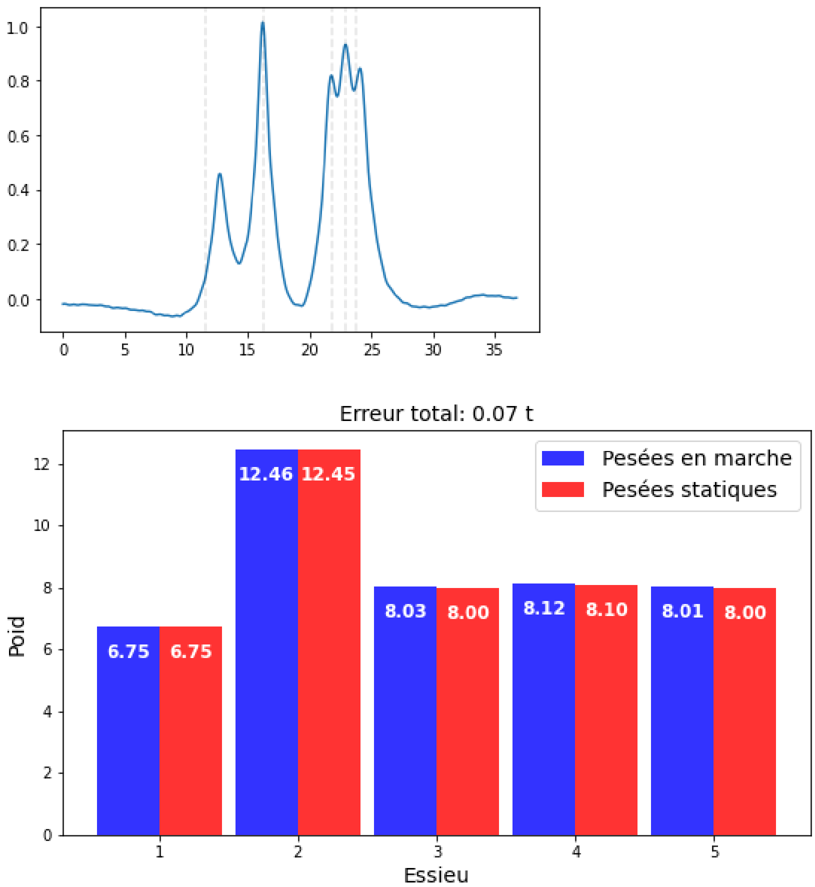


Fig.44 - WIM with peak-to-peak offset after calibration on the same truck

Since the objective was to use this method in deployment, the use of a 0 line of influence (calculated from a randomly drawn truck) was attempted, in the hope that the shift would compensate for the difference with the best line of influence for each HGV.

The same problem was initially encountered: depending on the speeds, the results (although improved), varied greatly.

Finally, the overestimation of the number of axles was carried out to see if the previous method was 'sound'. So by asking the algorithm to find a combination of peaks with one peak more than axles, the calculated overestimated peak was always in line with one of the existing peaks. (See Fig.45)

```
[266. 367. 367. 527. 559. 591.]
```

Fig.45 - Results of overestimation of the number of peaks

Procedure for the evaluation of statistics on our methods

At the end of all this research, it was time to put in place a procedure to accurately evaluate these different methods.
To this end, a statistical procedure was put in place, operating as follows:

- **Creation of two subsets**

From the set of HGV's at our disposal, two subsets of HGV's are created in a random way (using the ***random.shuffle()*** function of the ***NumPy*** library to mix the indices).

- **Calibration**

The first subassembly is used for calibration (calculation of the interpolation function giving the line of influence as a function of speed). The individual influence lines used for interpolation are calculated with a total variation regularization.

- **Weighing in motion**

The interpolated function (line of influence/speed) is used to carry out WIM of the HGV's of the second subassembly. The function takes as a parameter the desired peak offset (if we want one). Specifically, the weighing can be performed without peak shift, with an offset on the combinations of the interval [peak-2, peak+2] for each peak or with a peak-by peak offset (method resulting from the separation of two superimposed peaks).

- **Recovery of metrics**

Once these three steps are completed, we retrieve the metrics we are interested in, store them in ***NumPy*** arrays and save them in ***.npy*** format.
The data is then studied in another notebook simply by loading the data in ***.npy*** format.

The metrics retrieved are :

- The total error (in tons) on each truck
- The total error in absolute value (in tons) on each truck
- Error (in tons) on each axle

It is easily possible to retrieve other metrics by adding them to the function.

Several things were then observed:

- **Procedure without peak offset**

The first results observed are those of this procedure (at least iterated 30 times to consider our results as statistics) without performing peak shifting.
The results for several distributions of calibration HGV's /WIM were calculated and then the evolution of the confidence intervals (at 95%) according to these distributions was observed.
Here are the results (on total errors) :

- **With 10% of calibration trucks**

```
MU : -0.12007728116926424
SIGMA :  3.9410748251919148
dims :  (262,)
(262,)
```

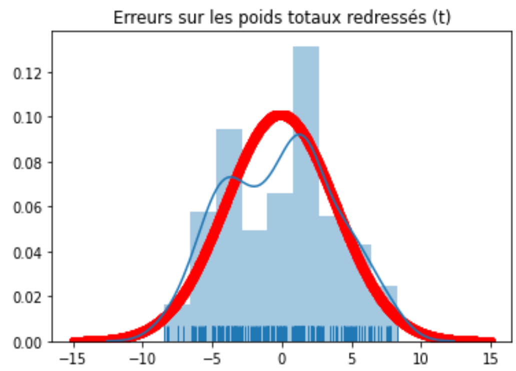


Fig.46 - Procedure with 10% calibration trucks

- **With 20% of calibration trucks**

```
MU : -0.19034197592263774
SIGMA :  3.828975065161104
dims :  (466,)
(466,)
```

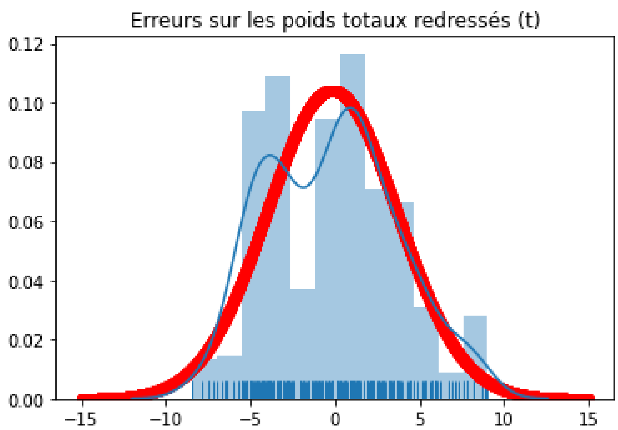


Fig.47 - Procedure with 20% calibration trucks

- **With 30% of calibration trucks**

```
MU : 0.16032629368421647
SIGMA :  3.744239464638809
dims :  (434,)
(434,)
```

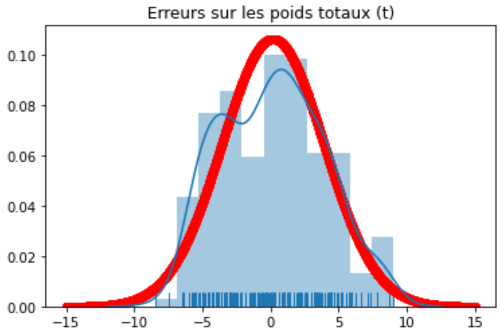


Fig.48 - Procedure with 30% calibration trucks

- **With 40% of calibration trucks**

```
MU : -0.07515429479057788
SIGMA :  3.719427856656481
dims :  (394,)
(394,)
```

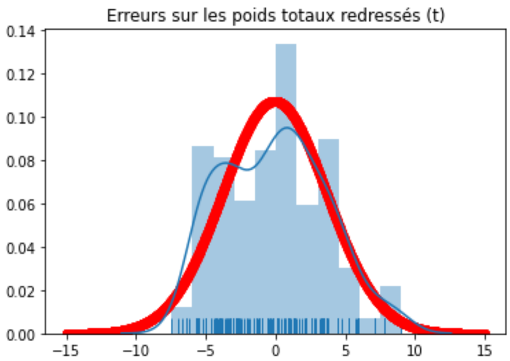


Fig.49 - Procedure with 40% calibration trucks

- **With 50% of calibration trucks**

```
MU :  -0.05791139054548957
SIGMA :   3.7977474805471045
dims :   (342,)
(342,)
```

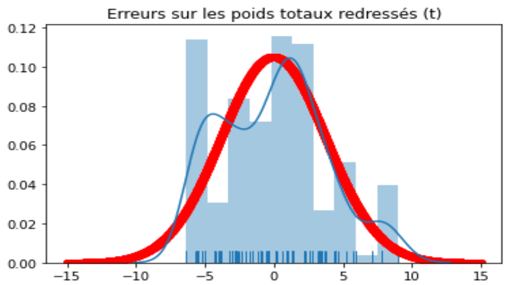


Fig.50 - Procedure with 50% calibration trucks

- **With 60% of calibration trucks**

```
MU :  0.6678147518731834
SIGMA :   3.957563357106753
dims :   (300,)
(300,)
```

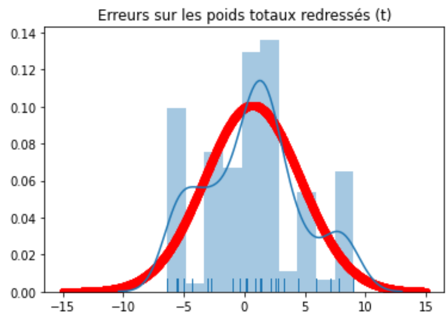


Fig.51 - Procedure with 60% calibration trucks

- **With 70% of calibration trucks**

```
MU :  0.3152516874220737
SIGMA :   4.228123772818377
dims :   (210,)
(210,)
```

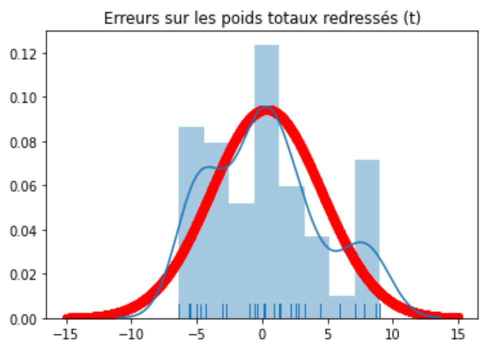


Fig.52 - Procedure with 70% calibration trucks

- **With 80% of calibration trucks**

```
MU : 0.040023931600414876
SIGMA :  4.4708912119361575
dims :  (150,)
(150,)
```

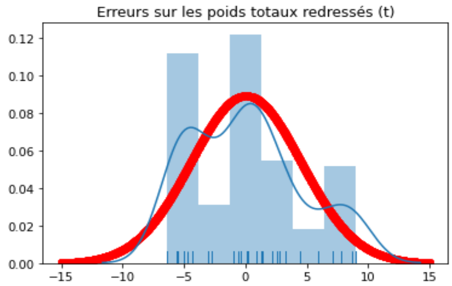


Fig.53 - Procedure with 80% calibration trucks

- **With 90% of calibration trucks**

```
MU : -0.11820339866902453
SIGMA :  4.578770480051701
dims :  (60,)
(60,)
```

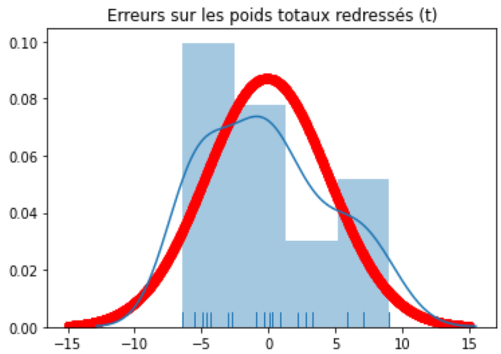


Fig.54 - Procedure with 90% calibration trucks

- **And the laws of evolution of certain parameters as a function of the distribution :**

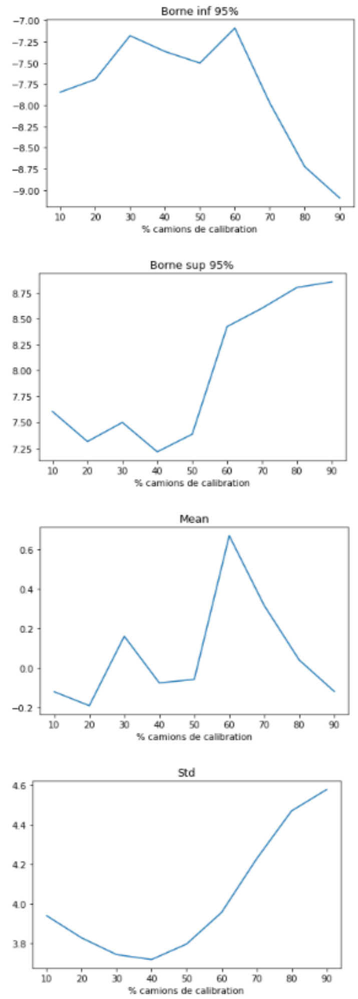


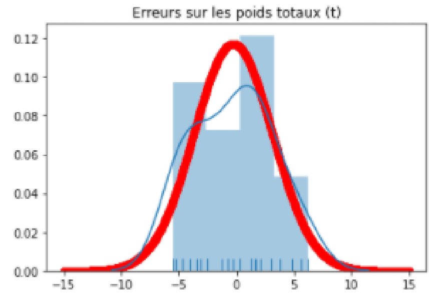
Fig.55 - Evolution of the lower/upper bound of the 95% confidence intervals, mean and standard deviation as a function of the calibration truck/traffic split

Due to the small amount of data, it is difficult to extract information from these developments. When the distribution becomes large, the variance increases (e.g., 90% = 1 weighing truck).

A distribution around 40% of calibration trucks seems to give the best compromise.
This procedure was then reproduced with both types of peak shift on this distribution (40%) (as the calculations are extremely heavy).
As a reminder, the shift is based on the criterion of the Euclidean distance between the reconstructed signal and the observed signal. The weights of the trucks do not need to be known in advance and it is therefore possible to deploy this method on new trucks.

- **Procedure with offset on combinations :**

```
MU : -0.3007804330544285
SIGMA :  3.4029107609671336
dims :  (28,)
(28,)
```

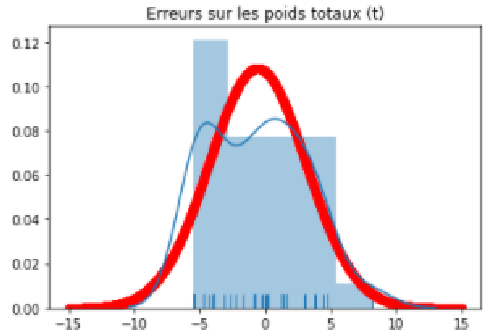



```
Intervalle à 95%  (-6.970362967153799, 6.368802101044943)
```

Fig.56 - Procedure with 40% calibration trucks and peak shift tested on a set of combinations

- **Peak-to-peak offset procedure :**

```
MU : -0.5998502199417537
SIGMA :  3.6895690871746973
dims :  (66,)
(66,)
```

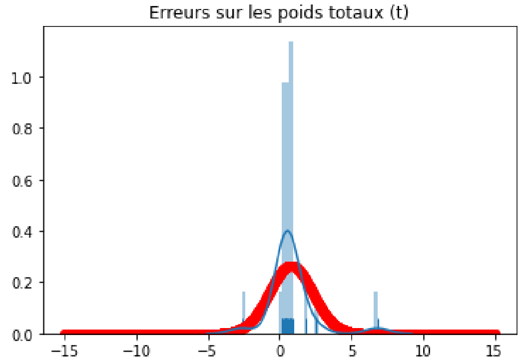



```
Intervalle à 95%  (-7.831272749276484, 6.631572309392976)
```

Fig.57 - Procedure with 40% calibration trucks and peak-to-peak offset

Finally, to observe the best results, a well-chosen interpolated function (calculated with the calibration trucks and some high-speed traffic trucks) was used. A peak-to-peak offset was applied. Here are the results (therefore considered the best):

```
MU : 0.7879672243192921
SIGMA :  1.5069335652986349
dims :  (24,)
(24,)
```




```
Intervalle à 95%  (-2.1655682907605702, 3.7415027393991545)
```

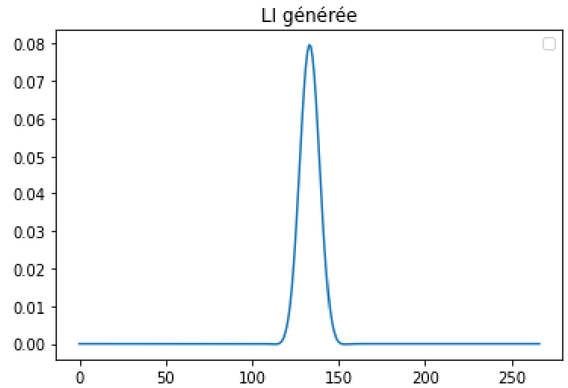
Fig.58 - Procedure with a given line of influence (calculated on a well-chosen set of PLs)

The average of the adjusted total errors (absolute values) is 1.05 tons (average error of 2.5%). (Average weight of all trucks: 41.92t).

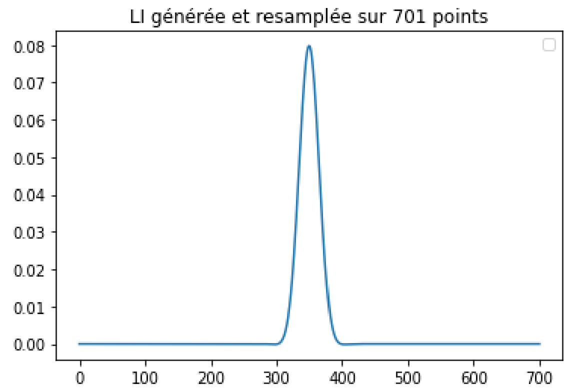
- Mean absolute total error (in tons) on axle 1: **1.05 t (15%)**
- Mean absolute total error (in tons) on axle 2: **2.02 t (17%)**
- Mean absolute total error (in tons) on axle 3: **2.26 t (26%)**
- Mean absolute total error (in tons) on axle 4: **1.02 t (13%)**
- Mean absolute total error (in tons) on axle 5: **0.91 t (12%)**

Evaluation of results on simulated data

To ensure that the procedure is robust, it was performed on simulated (generated) data. For the line of influence, the product of a cosine and a Gaussian was used. (See Fig.59) It was then weighted by a coefficient derived from the lines of influence calculated by experimentation so that the sequence of elements generated could be used.




```
No handles with labels found to put in legend.
```



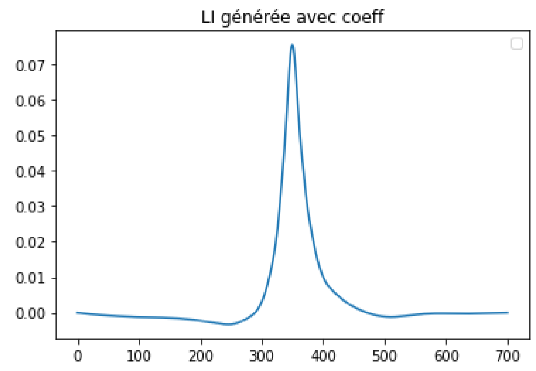


Fig.59 - Line of influence generated (without and with coefficient)

For the axle configuration (weight per axle), normal distribution was used. (See Fig.60)

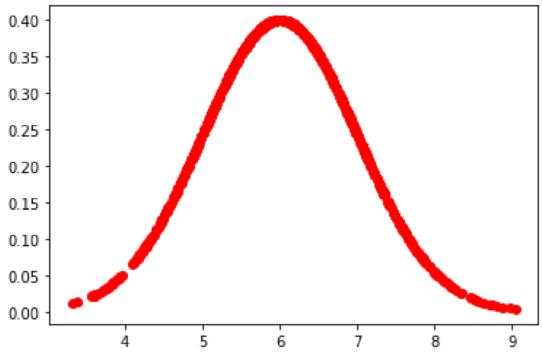


Fig.60 - Normal distribution used to generate axle configuration

Once this was done, a weighting was added according to the location of the axle so that it would be as close as possible to reality (first axles less heavy than the second, etc.).
The whole had to correspond to a total weight fixed in parameter. (See Fig.61)

```
[ 3.94682248 10.89253811  6.78084414  8.74727158 10.56943952]
```

Fig.61 - Example of generated axle weights

For the location of the axles (and thus the peaks), the means and standard deviations of those of the HGV's at our disposal were used to randomly generate (normal distribution) locations.
To generate the signal, the B-WIM model was then repeated to obtain a signal resembling those observed. (See Fig.62)

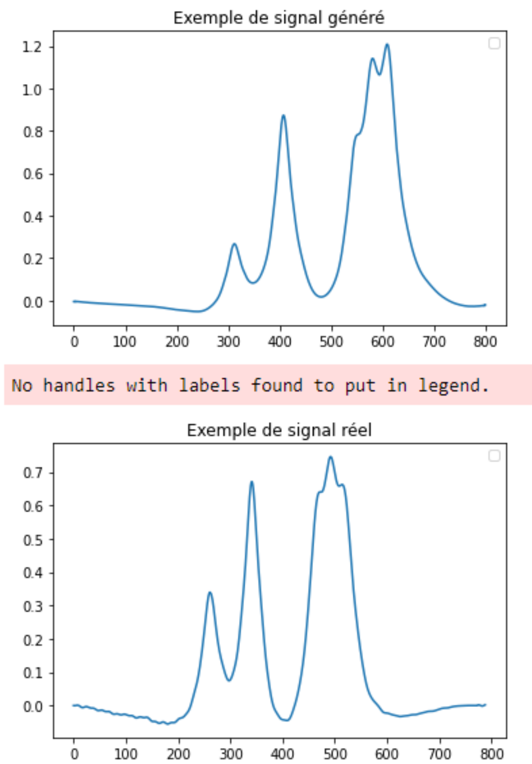


Fig.62 - Comparison of an observed and generated signal

Once the set of attributes describing a truck existed, we created a new ***namedTuple*** with generated attributes and restarted the procedure described above. (See Fig.63)

```
MU : 0.004422653566077487
SIGMA :  0.006205839404011655
dims :  (531,)
(531,)
```

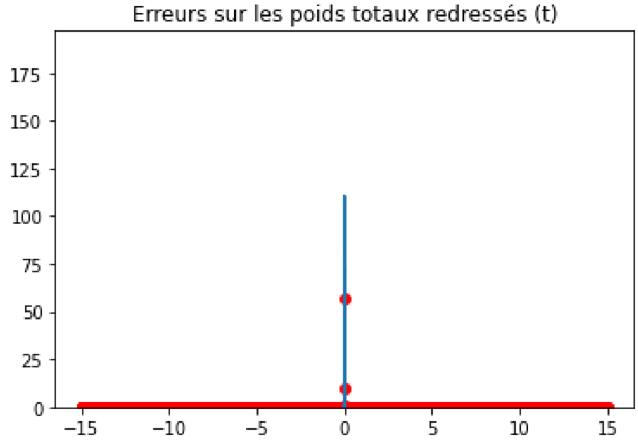


Fig.63 - Results of the simulated data procedure

To observe the effect of the noise level of a signal on the results, we then added noise to our signals (at different levels) to observe the evolution of the results. (See Fig.64 and 65)

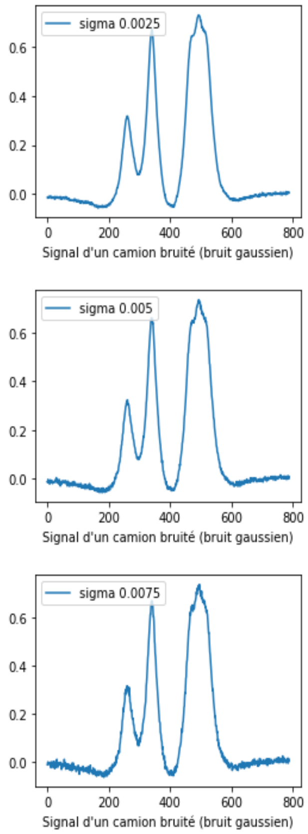


Fig.64 - Signals generated with different levels of Gaussian noise (sigma)

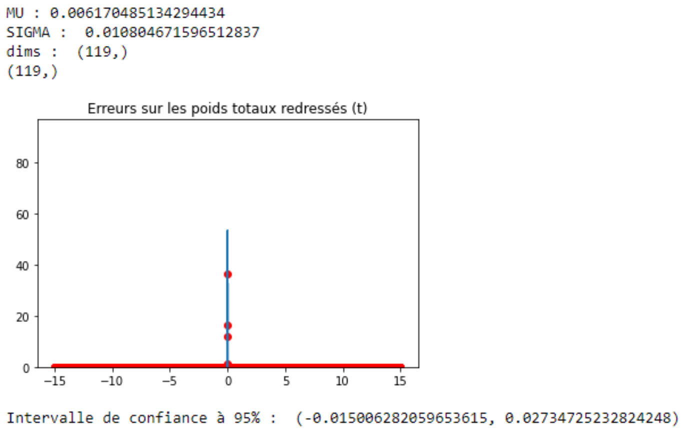


Fig.65 – Results of the procedure on simulated data with Gaussian signal noise such that the standard deviation = 0.005

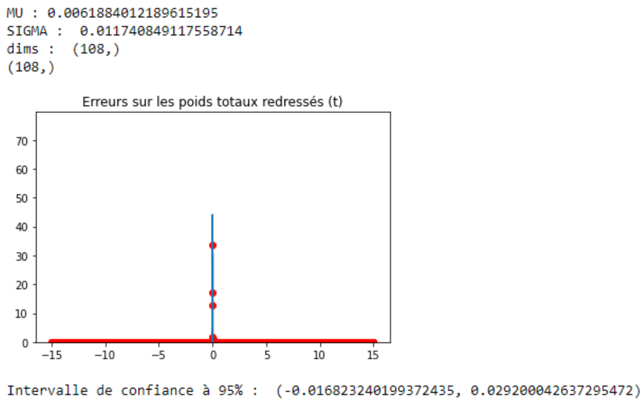


Fig.66 – Results of the procedure on simulated data with Gaussian signal noise such that the standard deviation = 0.02

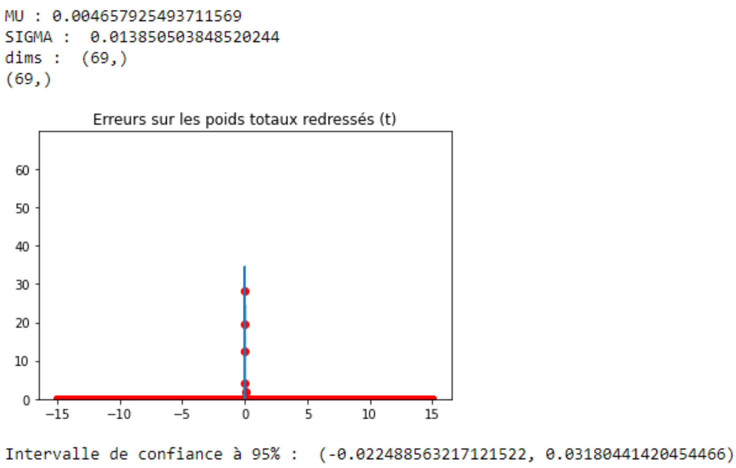


Fig.67 – Results of the procedure on simulated data with Gaussian signal noise such that the standard deviation = 0.03

Finally, the 95% confidence intervals widen well in proportion to the noise level of the signals.

Determination of the solutions of the dynamic equation of the bridge by physic informed neural network (PINN)

Principle

Finally, the last task of this course was to try to solve the differential equation describing the dynamic equation of the bridge (which corresponds to the line of influence when evaluated in its middle).

For this, a relatively recent method was used: physic informed deep learning. Its principle resides in the evaluation of the solutions of the differential equation by a neural network.
As a reminder, the differential equation to be solved is written in the form :

$$
\begin{aligned}
EI\frac{\partial^4 u}{\partial x^4}(x,t) + \rho A\frac{\partial^2 u}{\partial t^2}(x,t) &= \delta(x), \\
0 &\leq x \leq L \\
0 &\leq t \leq \frac{L}{v} \\
u(0,t) &= u(L,t) = 0 \\
\frac{\partial^2 u}{\partial t^2}(0,t) &= \frac{\partial^2 u}{\partial t^2}(L,t) = 0,
\end{aligned}
\tag{3}
$$

où:

- x est l'abscisse curviligne le long de la poutre,
- t le temps,
- $u(x,t$ est le déplacement vertical de la poutre en x et t,
- $\delta(x)$ est le dirac en x.

La ligne d'influence qui nous intéresse est donc celle du déplacement à mi-portée, soit $u(x)(\frac{L}{2},t)$.

Fig.68 - Dynamic equation of the bridge

With :

- **E** : Young's module
- **p** : beam density
- **A** : beam surface
- **I** : Inertia of the beam
- **L** : Beam length

Resolution with SCIANN

We first used ***SCIANN***, a library developed by MIT based on ***Tensorflow*** to solve this kind of problem.
The problem we encountered was the use of a unitary Dirac function (delta(X)) which was difficult to integrate with this library.
However, the results are available on the N18 Notebook.

Resolution with DeepXDE

DeepXDE, another open source library, was then used, allowing more modifications from the user (although more complicated to understand).
After defining the equation to be solved, we had to describe the shape of the solutions of this equation as a neural network. (See Fig.69)

```
net = dde.maps.FNN( [2] + [32]*2 +[64]+[32] +[1], "tanh", "Glorot normal")
```

Fig.69 - Neural network representing the solution of the differential equation

This is a classic feedforward network whose layers are shown above. The activation function used is tanh (notably for the amplitude of its derivative, allowing to avoid the evanescence of the gradient in the backward propagation algorithm. (See Fig.70) Especially since the manipulated values are of the order of a thousandth.

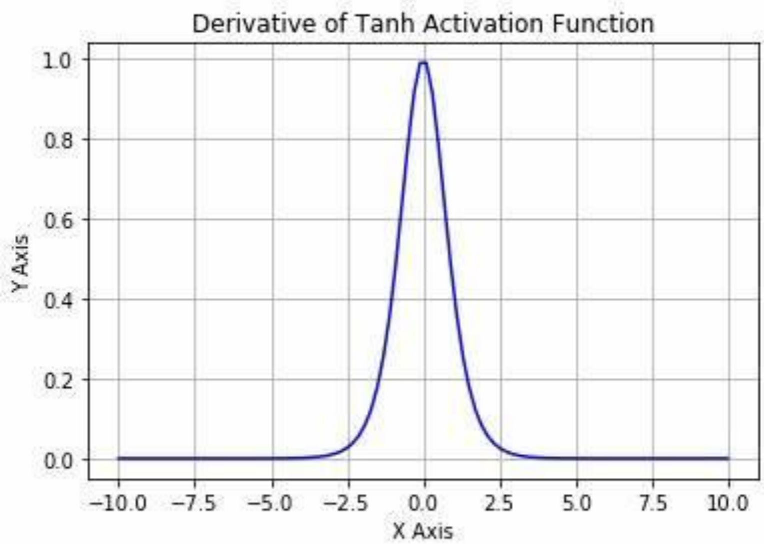


Fig.70 - Tanh derivative

In comparison, we could observe the sigmoid derivative, another known activation function (generally used for probas because of its arrival interval). (See Fig.71)

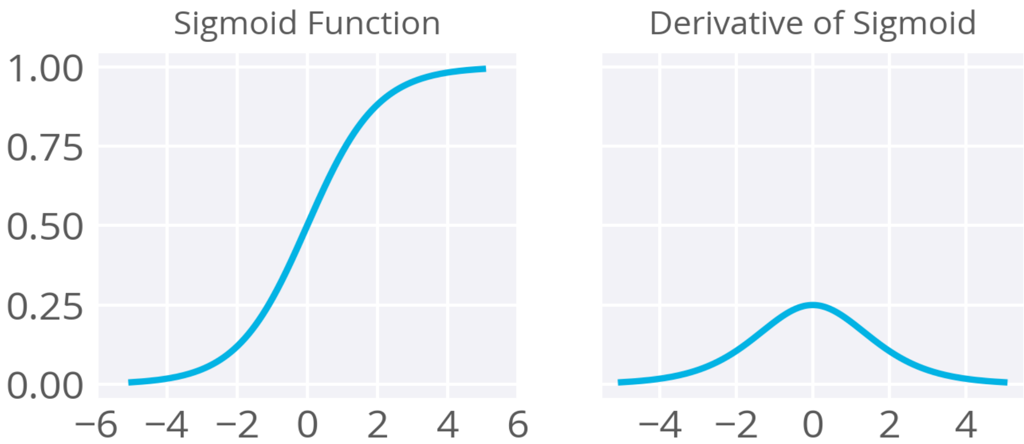


Fig.71 - Sigmoid derivative

The gradient descent algorithm used is Adam.
The details of the parameters (definition intervals etc) are available on the Notebook.
Here are different results obtained after various trainings :

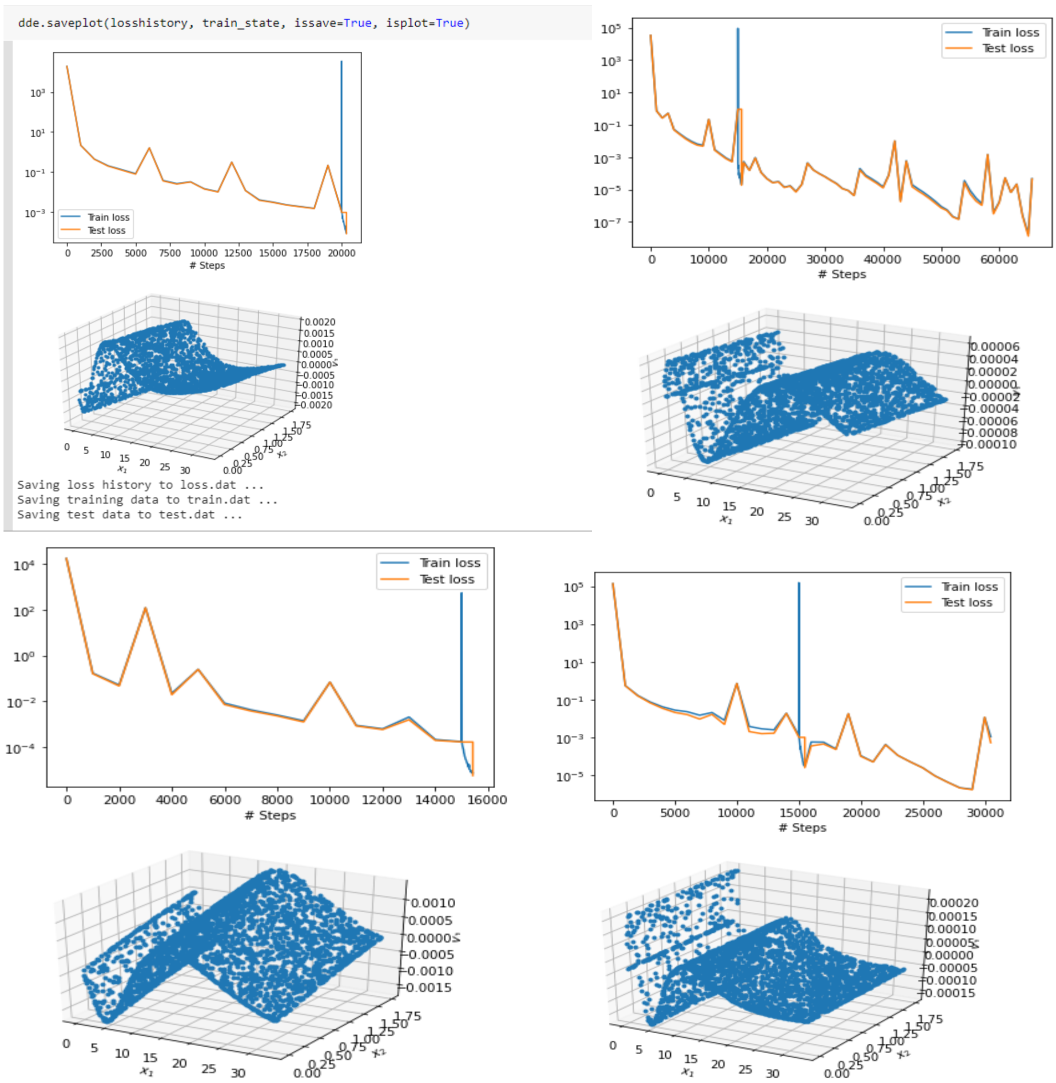


Fig.66 – Results of the resolution of the differential equation by DeepXde

Overall, the shape of a line of influence can be recognized. However, depending on the accuracy (lower and lower loss), the results vary.
They are therefore difficult to use at this stage.
These calculations were made with the characteristics of the Senlis bridge.
One idea would have been to solve this equation by classical differential equation solving methods. However, because of the intervening derivatives (4th in space and 2nd in time), it is difficult to find a suitable library.
Due to the lack of time, the subject could not be further investigated.

Assessment of technical, organizational and human experience

The whole internship required the use of various libraries essential to the practice of data science and artificial intelligence, and I broke with their use.

More precisely, from a technical point of view, I was able to perfect my knowledge of ***NumPy***, ***Scipy***, ***Seaborn***, ***Matplotlib*** or ***Sklearn*** and more globally of the python syntax (use of ***list comprehension***, ***enumerate loop*** and creation of a library).
I also discovered a new part of artificial intelligence interesting and complex: physic informed deep learning.

The technical part was mixed with a large scientific part, based on signal processing, algebra, analytics, and a set of concepts from the mathematical and physical fields.

This mix of knowledge and technique has allowed me to raise my level in these fields which are an integral part of my engineering curriculum.

My tutors Mrs Franziska SCHMIDT, Mr Jean-François BERCHER and Mr Giovanni CHIERCHIA accompanied me remarkably well throughout the training course and took the time to explain to me sometimes complex notions so that this training course goes as well as possible. I thank them for this.

At the end of this training course, I feel much more at ease in ***python*** and in all the fields mentioned above.

It is also important to note the importance of the subject, which, thanks to its captivating character, made the acquisition of this knowledge much simpler and more fluid.

In parallel with the technical and scientific aspects, I was able to put into practice a professional organization for the first time, notably with the setting up of weekly meetings (even more frequent), the writing of reports at the end of these meetings, the use of Git and GitLab for sharing resources and the adaptability necessary to carry out this training course in good conditions despite a complicated health situation (Covid-19) involving its realization in telework.

I regret not having been able to work on site at IFSTTAR to meet new people and enrich my scientific culture. These regrets are largely compensated by all the elements mentioned above.

I would also like to take this opportunity to thank, in addition to my tutors, IFSTTAR in general for having trusted me and entrusted me with this most interesting mission.

Annexes

Annex 1 - Bibliography

All the scientific documents used to train me are available in the 'papers' folder.
I have also used ***Stackoverflow***, ***Wikipedia*** and ***official library documentation*** on many occasions.

Ressource technique utilisée	**Lien vers la ressource**
python	https://www.python.org/
NumPy	https://numpy.org/
SciPy	https://www.scipy.org/
scikit learn	https://scikit-learn.org/stable/
Seaborn	https://seaborn.pydata.org/
matplotlib	https://matplotlib.org/
git	https://git-scm.com/
GitLab	https://git.esiee.fr/
TensorFlow	https://www.tensorflow.org/

Annex 2 - DataFrame of calibration trucks

	name	time	speed	signals	peaks	weights
0	2015-09-29-09-28-46-125	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.999990	[-0.0132997, -0.0133036, -0.0132623, -0.013174...	[260, 340, 493]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
1	2015-09-29-09-56-08-457	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.380992	[0.000226915, -8.02875e-05, -0.000158966, -1.4...	[261, 341, 472, 492, 514]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
2	2015-09-29-10-28-52-687	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.272699	[-0.00589252, -0.00536799, -0.00480485, -0.004...	[262, 342, 467, 494, 521]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
3	2015-09-29-11-08-45-998	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.630769	[0.00175452, 0.00153422, 0.00130749, 0.0011000...	[317, 397, 522, 549, 575]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
4	2015-09-29-11-50-31-328	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.000029	[-0.00212467, -0.00248653, -0.00277889, -0.002...	[263, 342, 471, 494, 518]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
5	2015-09-29-12-17-33-320	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	21.633769	[-0.0144981, -0.0136787, -0.0132154, -0.013189...	[263, 347, 480, 508, 536]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
6	2015-09-29-13-33-05-960	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.630779	[-0.0215423, -0.0214133, -0.0212388, -0.021041...	[265, 344, 470, 496, 522]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
7	2015-09-29-14-00-38-369	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	18.731689	[-0.0243554, -0.0245404, -0.0246975, -0.024829...	[266, 368, 528, 561, 593]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
8	2015-09-29-14-28-15-093	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	18.963019	[-0.0254103, -0.0254015, -0.0250018, -0.024329...	[266, 366, 526, 558, 590]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
9	2015-09-29-14-55-32-562	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	18.505997	[-0.020827, -0.0209844, -0.0211789, -0.0214018...	[266, 367, 527, 560, 593]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
10	2015-09-29-15-23-14-921	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	21.333342	[-0.0167387, -0.0168498, -0.0170026, -0.017174...	[265, 354, 492, 521, 550]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]
11	2015-09-29-15-52-01-990	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	18.962942	[-0.0185622, -0.0196713, -0.0207321, -0.021570...	[266, 367, 527, 559, 591]	[6.16068, 10.8499, 9.08406, 6.98472, 7.0632]

Annex 3 - DataFrame of traffic trucks

	name	time	speed	signals	peaks	weights
0	PL10_2015-10-01-14-58-50-212	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.630779	[-0.0191894, -0.0187526, -0.0184853, -0.018449...	[276, 351, 472, 497, 522]	[6.75, 12.45, 8.0, 8.1, 8.0]
1	PL11_2015-10-01-15-00-57-826	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.999981	[-0.00928965, -0.00987199, -0.0105873, -0.0113...	[276, 355, 469, 492, 517]	[7.15, 11.4, 8.05, 8.0, 8.45]
2	PL12_2015-10-01-15-10-30-906	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.630816	[-0.0229519, -0.0233275, -0.0233024, -0.022886...	[277, 354, 478, 503, 529]	[7.4, 10.8, 8.8, 9.05, 8.75]
3	PL13_2015-10-01-15-19-48-718	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	25.180255	[-0.0206496, -0.0207517, -0.0208063, -0.020829...	[277, 350, 465, 489, 512]	[8.15, 12.2, 7.3, 8.3, 8.8]
4	PL14_2015-10-01-15-34-03-982	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.381001	[-0.0116909, -0.0108816, -0.010028, -0.009303,...	[275, 358, 488, 508, 531]	[6.2, 5.05, 7.95, 7.65, 7.4, 7.65]
5	PL15_2015-10-01-15-36-13-519	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	22.260846	[-0.0136828, -0.0139356, -0.0142175, -0.014500...	[272, 399, 425, 518, 568, 622]	[5.35, 8.25, 8.1, 7.75, 7.8, 7.6]
6	PL1_2015-10-01-09-45-52-576	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	22.260904	[-0.014285, -0.0140879, -0.013944, -0.013884, ...	[275, 361, 473, 501, 529]	[7.35, 11.25, 8.2, 8.1, 8.25]
7	PL2_2015-10-01-09-55-36-371	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.000000	[-0.0100142, -0.00989681, -0.0097338, -0.00955...	[274, 353, 477, 503, 529]	[7.65, 14.1, 6.05, 6.2, 6.3]
8	PL3_2015-10-01-10-22-34-654	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.000029	[-0.014935, -0.0149689, -0.015053, -0.0152061,...	[274, 349, 470, 495, 519]	[7.4, 11.6, 8.35, 8.5, 8.5]
9	PL4_2015-10-01-10-35-43-279	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.380962	[-0.0168291, -0.0172036, -0.017665, -0.0181426...	[276, 354, 462, 487, 513]	[7.6, 12.35, 7.8, 7.7, 7.8]
10	PL5_2015-10-01-10-59-39-060	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.272746	[0.00408244, 0.00418735, 0.00422049, 0.0041573...	[274, 353, 452, 478, 505]	[7.35, 10.5, 9.3, 9.7, 9.45]
11	PL6_2015-10-01-11-05-36-548	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.000048	[-0.015224, -0.0151083, -0.0148442, -0.0144753...	[276, 354, 478, 503, 529]	[7.3, 14.3, 6.8, 6.6, 6.7]
12	PL7_2015-10-01-11-14-06-712	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	24.380893	[-0.0128941, -0.0127466, -0.012642, -0.0126112...	[275, 350, 469, 493, 518]	[7.05, 13.95, 7.25, 7.55, 7.9]
13	PL8_2015-10-01-11-42-22-222	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.630750	[-0.0150944, -0.0153917, -0.0159598, -0.016714...	[271, 344, 468, 493, 519]	[6.4, 14.05, 6.85, 6.75, 6.65]
14	PL9_2015-10-01-14-42-22-908	[0.0, 0.00195312, 0.00390625, 0.00585938, 0.00...	23.272718	[-0.0253963, -0.0255132, -0.0254464, -0.025214...	[277, 357, 474, 502, 528]	[8.2, 12.0, 8.35, 8.4, 8.35]