Société de Calcul Mathématique, S. A.
Outils d'aide à la décision
depuis 1995

Recovering the processing performed on a measured signal

Gottfried Berton
Société de Calcul Mathématique SA
111 Faubourg Saint Honoré
75008 Paris France

November 2017

I. Abstract

Let us assume that we have theoretical and measured data. Due to various physical effects, each measure is an average of the real data: the experimental curve is a moving average of the theoretical curve. We present an approach to obtain the coefficients of this average, that exploits Archimedes methods [AMW]. This problem is equivalent to solving a linear system of equations with uncertainties upon the coefficients. Our method allows to obtain a probability law on the solution, and also works when the system is not linear. The application of this work can be for instance to check the resolution of a sensor (does it integrate over 1 second as it should?), or to reconstruct the processing that has been done on a raw signal (image, sound, etc.).

The present work originates in a contract with the Nuclear Energy Agency (contract 400100201/500057515, 04/04/2017), which dealt with the comparison between experimental data and theoretical models to detect outliers. We thank Oscar Cabellos and Luiz Léal for bringing such question to our attention.

II. Presentation of the problem

For each experimental point b_i at abscissa E_i, let us discretize the range $[E_i-\Delta E, E_i+\Delta E]$ in n intervals $\left[E_1^i,E_2^i\right],\left[E_2^i,E_3^i\right],\ldots,\left[E_n^i,E_{n+1}^i\right]$ having the same size, where ΔE is fixed. The goal is to find the weights $x_j,\, j=1,\ldots,n$ of the following linear combinations:

$$b_i=\sum_{j=1}^{n} x_j a_{i,j}\ ,\ i=1,\ldots,N\ ,$$

with the constraints:

$$x_j \geq 0 \text{ and } \sum_{j=1}^{n} x_j = 1,\ j = 1,\ldots,n,$$

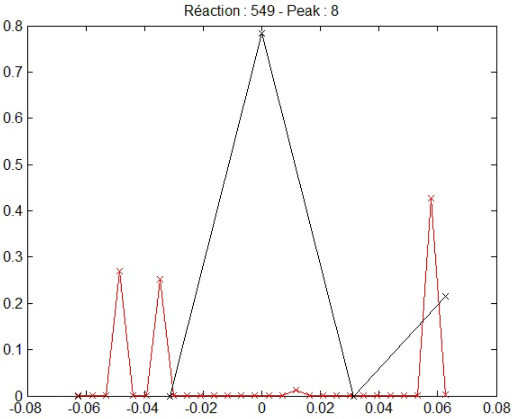
where b_i is the experimental measure, $a_{i,j}$ are the theoretical value at abscissa E_j^i, and they are both known. We are going to solve the linear system $AX = B$, where A is the matrix of the $a_{i,j}$, $B = (b_1,\ldots,b_N)$, and X the vector of the weights, also called resolution function.

In summary, the experimental curve is a moving average of the real curve. The real and measured curves have two very different shape: the measured curve is more spread than theoretical one. The comparison point by point (distance) is not correct in this case, especially when the data are very variable. Hence, this work is valuable for situations of high variability (the resonance zones for nuclear data).

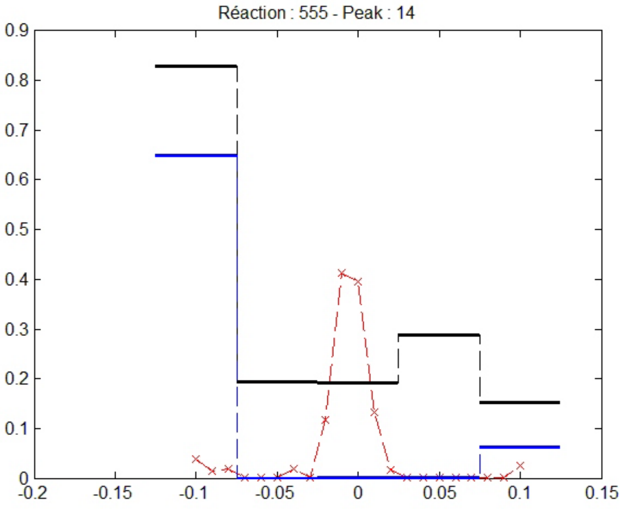
We are going to solve this system by taking into account the uncertainties on the experimental data. The method is due to Archimedes, and is presented in the book [AMW].

Remark on the number of coefficients

Before presenting the method let us make a preliminary remark on the number of coefficients that we choose to assess the averaging. One usually think that starting with a rough discretization will give us an approximate idea of the solution, and that, by increasing the discretization, the solution will converge gradually toward the solution, but this is erroneous. The shape obtained with a rough discretization can be very different from the shape for a precise discretization. The figure below shows the result for a precise discretization in red, and rough discretization in black.



Furthermore, let us calculate the lower and higher bounds of the solution: all solution having a coefficient outside these bounds will have a null probability. The figure below show that the solution obtained with high discretization doesn't lie inside the bounds obtained with rough discretization (blue and black).



Using the rough discretization, the intuition is that the function is a peak at left, but the real solution is a peak at the middle.

III. Method

A. General principle

The principle is to generate random sets of weights x_i and to test the solutions. The test consists in calculating the right hand side of the equations above, that is the vector $C = AX$, and to compare this vector with the values actually measured $b_1, b_2, \ldots$ The weight obtained for the candidate solution is the product of the probabilities of the distance between b_1 and c_1, b_2 and c_2, etc., obtained using the uncertainty of each measure. This uncertainty can be represented by any distribution, and they don't need to be the same for all the measures. If a distance is too large with respect to the experimental uncertainty, the probability is set to zero. To obtain a fast algorithm, we generate only solutions having non-null probability, using the method presented in section B.

Eventually, after normalization by the sum of the weights when we generate all the candidate solutions, we obtain a probability law on each coefficient. Each coefficient is calculated as the expectation of this distribution, represented in blue on the figure below. The upper and lower bounds are represented in black.

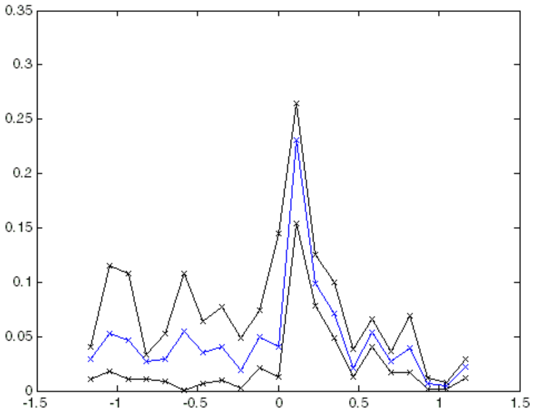


Figure 1: example of coefficient obtained function (blue) with upper and lower bounds (black)

The shape of the averaging can be compared across different situations. For example, does the variance of the coefficients change for different situations (e.g high and low energies)?

B. Generation of the candidate solutions

To generate the candidate solutions, the first approach we may think of is to compute the bounds $\left[x_i^{\min}, x_i^{\max}\right]$ by solving the minimization (or maximization) problem as below for each dimension $k = 1, \ldots, n$, and generating samples inside this parallelepiped.

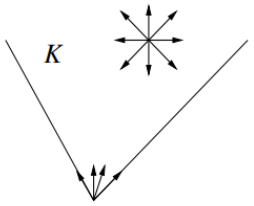
$$\min\ x_k \quad \begin{cases} b_i^{\min} \le \sum_{j=1}^{n} x_j a_{i,j} \le b_i^{\max}, & i = 1, \ldots, N \\ \sum_{j=1}^{n} x_j = 1,\ x_j \ge 0, & \mathrm{j} = 1, \ldots, n \end{cases}, \qquad (P)$$

where $\left[b_i^{\min}, b_i^{\max}\right]$ is an interval that includes the measure b_i with high probability (e.g 95%).

However, in practice, there are more than 50 coefficients, and the volume of the parallelepiped $\left[x_i^{\min}, x_i^{\max}\right]$ is very large compared to (P)'s. There is no chance to generate a solution with non-null probability in a reasonable computational time.

We present a method to generate only solutions satisfying all the inequalities, by performing a random walk inside the convex space (P). The algorithm has 4 steps:

- Step 1: Choose a first solution that satisfies all the inequalities in (P) using the simplex algorithm. This allows also to check for the existence of a solution;
- Step 2: Generate a random direction $d \in \mathbb{R}^n$ such that $\exists \varepsilon > 0,\ x + \varepsilon d \in P$.



Find the active constraints: the constraints for which the bounds are reached, if any. The space of all the possible directions is defined as:

$$T_P = \left\{ d \in \mathbb{R}^n \mid Z_k d \leq 0, \sum_{i=1}^{n} d_i = 0 \right\}$$

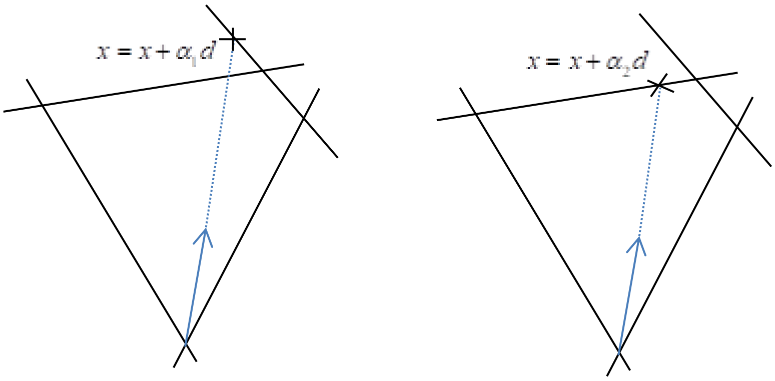
where Z_k the lines $i \in I \subset \{1,...,n\}$ of the matrix A corresponding to the coefficients of the active constraints. We are going to generate a random direction in this space. We consider two situations in function of which constraints are active:

- If there are only simple bounds constraints on x_k of the typr $x_k \geq C$ (that is, each line of Z_k has only zeros and -1 or 1 at one column): for the coefficients d_k, $k \in I^{\sup}$ corresponding to superiority constraints on x_k, we are going to generate negative random values, and for d_k, $k \in I^{\inf}$ positive values. For the remaining $d_k, k \in \overline{I}$, we generate positive random values and normalize the $d_k, k \in \overline{I}$ by the term $-\sum_{k \in I \setminus \overline{I}} d_k / \sum_{k \in \overline{I}} d_k$ so the sum of the d_k is equal to zero.

- If there are more complex active constraints ($b_i^{\min} \leq \sum_{j=1}^{n} x_j a_{i,j} \leq b_i^{\max}$) we use the simplex algorithm on problem T_P. This operation is seldom needed, because these complex constraints are rarely reached, and once they are reached, this operation moves x_k away from the border.

- If no constraint is active (other than sum equal to one which is always active), then any direction having sum equal to zero is suitable.

– Step 3: For each constraint i, calculate the step α for which $x = x + \alpha d$ hits this constraint. The α_i are computed such as $A_i(x + \alpha_i d) = B$, that is $\alpha_i = \frac{B_i - (Ax)_i}{(Ad)_i}$. The maximum step to stay in the bound in this direction is $\alpha = \min_{\alpha_i \geq 0} \alpha_i$. In the example below we choose α_2 because this is the smallest:



- Step 4: We generate a random $\theta \in \left]0,\alpha\right]$, update $x = x + \theta d$, and use this solution as initial point for Step 2.

The first step is important because it detects whether it is possible to find a set of coefficient given the uncertainties on the measures. There must exists a set of weights such that when we use it to broaden the theoretical data, the results lays inside a confidence interval of each measured value (red), otherwise it can mean that there is a potential outlier in the measure.

When it is easy to compute the vertices of the convex defined by (P), one can generate a sample X uniformly on this subspace using the following formula (see theorem 2.1 in [DEVROYE]):

$$X = \sum_{i=1}^{d} S_i v_i \ ,$$

where $v_1,\dots,v_d$ are the vertices of the convex in $\mathbb{R}^n$ and $S_1,\dots,S_d$ the weights generated by a uniform sample of size d on $[0,1]$.

IV. Deterministic methods

The problem can also be solved in the least square meaning, but in this case we will return no uncertainties on the coefficients. There are two standard methods to solve the constrained least square problem: projected-gradient and interior-point method. We show that projected-gradient method is not suitable for the type of data from the NEA, because the matrices are bad conditioned. It converges very slowly compared to interior-points.

Since there are more equations than unknown, there is no exact solution to the problem because of uncertainties upon the coefficients. An approximated solution can be found by minimising the quantity:

$$\varphi(w_1,\dots,w_n) = \sum_{i=1}^{N} \left(b_i - \sum_{j=1}^{n} a_{i,j} w_j \right)^2$$

The solution of the minimization is the solution of this system of n equation and n unknowns:

$$\frac{\partial}{\partial w_1}\varphi\left(w_1,\ldots,w_n\right)=-a_{1,1}\left(b_1-\sum_{j=1}^{n}a_{1,j}w_j\right)-a_{2,1}\left(b_2-\sum_{j=1}^{n}a_{1,j}w_j\right)-\ldots-a_{N,1}\left(b_N-\sum_{j=1}^{n}a_{1,j}w_j\right)$$

$$\frac{\partial}{\partial w_2}\varphi\left(w_1,\ldots,w_n\right)=-a_{1,2}\left(b_1-\sum_{j=1}^{n}a_{1,j}w_j\right)-a_{2,2}\left(b_2-\sum_{j=1}^{n}a_{1,j}w_j\right)-\ldots-a_{N,2}\left(b_N-\sum_{j=1}^{n}a_{1,j}w_j\right)$$

etc.

$$\frac{\partial}{\partial w_n}\varphi\left(w_1,\ldots,w_n\right)=-a_{1,n}\left(b_1-\sum_{j=1}^{n}a_{1,j}w_j\right)-a_{2,n}\left(b_2-\sum_{j=1}^{n}a_{1,j}w_j\right)-\ldots-a_{N,n}\left(b_N-\sum_{j=1}^{n}a_{1,j}w_j\right)$$

When there are constraints, it is not sufficient to simply solve this system, one have to minimize φ while maintaining the constrained satisfied. We present below two methods standard methods to do so.

A. Interior points method

The basic principle of the interior-points method is to write the optimality conditions for the problem written under quadratic form:

$$\begin{cases}\min\limits_{x\in\mathbb{R}^n}\frac{1}{2}x^T Hx+C^T x\\ \qquad (1,\ldots,1)x=1,\\ \qquad\qquad x\geq 0\\ \end{cases}$$

where $H=-A^T W^{-1}A,\ C=A^T W^{-1}B$. The following conditions are necessary and sufficient for optimality, solving them gives the best solution that satisfies the constraints:

$$\begin{cases}L(x,\lambda,\mu)=Hx+C+\lambda-\mu=0\\ (1,\ldots,1)x=1\\ \mu_i x_i=0,\ \forall i\\ \mu\geq 0\end{cases}$$

where $L(x,\lambda,\mu)$ is the Lagrangian and μ,λ are Lagrangian multipliers associated to bound constraints and equality constraints. We solve this non-linear system of equations iteratively using Newton method, and maintaining the positivity of μ at each iteration.

B. Projected gradient

This type of constrained quadratic minimization can also be solved using projected gradient method, but the interior-point method performs better for the data from the NEA. The principle of this method is to choose at each step the direction toward the descending slope (that is negative gradient) and to project the solution on the constraints. However, this projection is complicated, and a complement of the projected method allows to use projected gradient on a problem that is equivalent to the initial problem, and for which the projection is now trivial (Uzawa method). This problem is to find a solution $\left(x^*,\lambda^*,\mu^*\right)$ such that:

$$L(x^*,\cdot,\cdot)\le L(x^*,\lambda^*,\mu^*)\le L(\cdot,\lambda^*,\mu^*)\,,$$

that is,

$$\begin{array}{l}\max\limits_{\lambda,\mu}\left(\min\limits_{x} L\left(x,\lambda,\mu\right)\right)\\ \mu\ge 0\end{array},$$

where $x*$ minimizes the lagrangian. We use projected gradient on the function $G(\lambda,\mu)=\min\limits_{x} L(x,\lambda,\mu)$ for which we can calculate the gradient and value easily, and the projection on constraint is only to ensure positivity of μ.

The standard version of this algorithm converges for a step $0<\rho<eig_1(H)$ where $eig_1(H)$ represents the smallest eigenvalue of A [ENS]. In our real case with real data, the matrices A are very bad-conditioned and the eigenvalues are close to 0. The steps are very small and the algorithm converges slowly.

The general objection one can make to this kind of method is that, it is not robust to a change in the objectives and constraints: one can decide that the objective is not a quadratic form anymore but something else, the constraints can be non-linear; one would need to change the whole modelling. Moreover, the principle of minimizing a sum of squares is arbitrary and is justified only when the uncertainty are normal laws, which is seldom the case in real situations. Eventually, this is necessary to obtain an uncertainty upon the solution. In the case of the NEA for example, it was important to obtain an uncertainty on the resolution, and not just a precise result.

V. Conclusion

This work can be applied to various situations where there is a raw signal, and a processed signal, and we wish to assess the nature of the processing that has been applied. This can be used for example to test a sensor: does it have the resolution that we think it has? The sensors should measure the temperature every second, so the temperature should be integrated over this second (over a uniform law for example). We will verify that this is the case by comparing it's measure with the real values measured by another sensor. If it integrates over two seconds instead of one, this could lead to erroneous decisions for the user of the device. This verification cannot be done easily visually by simply looking at the data. This work is also useful when we want to check what filter has been applied to an image.

Our approach allows to find the probability law of the solution to a system of equations, using the different sources of information (each measure), and the uncertainties on it. This approach can work for non-linear equations.

This methodology that allows also to detect a very specific situation in a large dimensional space when linear constraints on this situation are available, without performing a blind random exploration that would be impossible.

One can show the variability of the solution for different experimental parameters, for example, in the case of nuclear data, the averaging is not the same when the energy is in 10-100 eV than on interval 100-1000 eV.

Eventually we compared two deterministic methods to solve the systems with constraints: Uzawa was not suitable for the data that we used because of bad conditioned matrix, but interior point was doing fine.

VI. References

[AMW] Bernard Beauzamy : Archimedes' Modern Works, SCM SA, ISBN 978-2- 9521458-7-9, ISSN 1767-1175, relié, 224 pages. Août 2012.

[DEVROYE] Luc Devroye, Non-uniform random variate generation, Springer-Verlag, New York Berlin Heidelberg Tokyo, 1986.

[ENS] Introduction to optimization, J.-F. Aujol, 2008
https://www.math.u-bordeaux.fr/~jaujol/PAPERS/optim_agreg.pdf