Société de Calcul Mathématique, S. A.

Algorithmes et Optimisation

Méthodes mathématiques

pour la vérification de bases de données

Rapport adressé à

l'Agence de l'OCDE pour l'Energie Nucléaire

(à l'attention de M. Emmeric Dupont)

par la

Société de Calcul Mathématique SA

en application du contrat 2010/00064251, notifié le 28/09/2010

janvier 2011

Rédaction : Carmen Rodriguez, Hélène Bickert

Résumé

L'Agence pour l'Energie Nucléaire (AEN) est une agence spécialisée de l'Organisation de Coopération et de Développement Economique (OCDE) ayant pour mission d'aider ses pays membres à maintenir et à approfondir les bases scientifiques, technologiques et juridiques en rapport avec l'énergie nucléaire.

L'AEN dispose ainsi de plusieurs bases de données, qu'elle enrichit constamment et met à la disposition des scientifiques intéressés. Le présent travail porte particulièrement sur la base EXFOR, constituée de données expérimentales de mesures de réactions nucléaires. Cette base représente plus de 19 000 expériences et contient plus de 130 000 séries des données.

La base présente des anomalies, d'origine humaine (erreurs lors de la mesure ou du traitement des données) ou liées aux instruments de mesure. L'AEN souhaite que les bases soient de la meilleure qualité possible. L'objectif du contrat est donc de mettre en place des méthodes permettant la vérification des bases et la détection d'enregistrements erronés. Ces outils doivent être "flexibles", en ce sens que l'on doit pouvoir les utiliser sur les diverses bases. Les méthodes mises en place doivent maîtriser le taux de faux positifs, c'est-à-dire le nombre de mesures définies comme anomalies à tort.

Définition des anomalies

Les données sont regroupées sous la forme d'Ensembles Homogènes de Données (EHD) : il s'agit de données comparables entre elles (même réaction nucléaire, même quantité mesurée). La méthode s'applique à des bases de données comportant un nombre de points minimum : nous ne traitons que les EHD constitués d'au moins 5 séries de mesures (5 expériences différentes) ou au moins 10 points de mesure (pouvant provenir d'une ou plusieurs expériences). Ceci représente au total 28 000 EHD de la base EXFOR.

L'AEN a fourni à la SCM 125 cas tests représentatifs des cas d'anomalies existant dans la base de données, dont 113 en 2D : un premier jeu de données (55 réactions) est un échantillon représentatif des différents types de réactions et anomalies rencontrés dans la base EXFOR ; un second jeu de 64 réactions contient des anomalies particulières, parfois peu présentes dans la base, mais pour lesquelles la méthode de détection doit être adaptée.

Ces exemples permettent de définir ce qu'est une anomalie pour l'AEN, et d'appréhender la grande diversité des types de réactions et des types d'anomalies.

Les EHD représentent des fonctions continues, pouvant comporter des zones de résonance ; elles peuvent être complètes (continuité du nuage de points), ou par morceaux, et présenter des incertitudes. La quantité de données constituant un EHD est très variable ; certains contiennent un nombre de mesures très faible. Cette diversité importante dans les types de fonctions et d'anomalies doit être prise en compte dans la mise en place de la méthode de détection.

Nous donnons ici quelques exemples de fonctions présentant des anomalies :

Exemple un seul point anomalie, avec zone de résonance

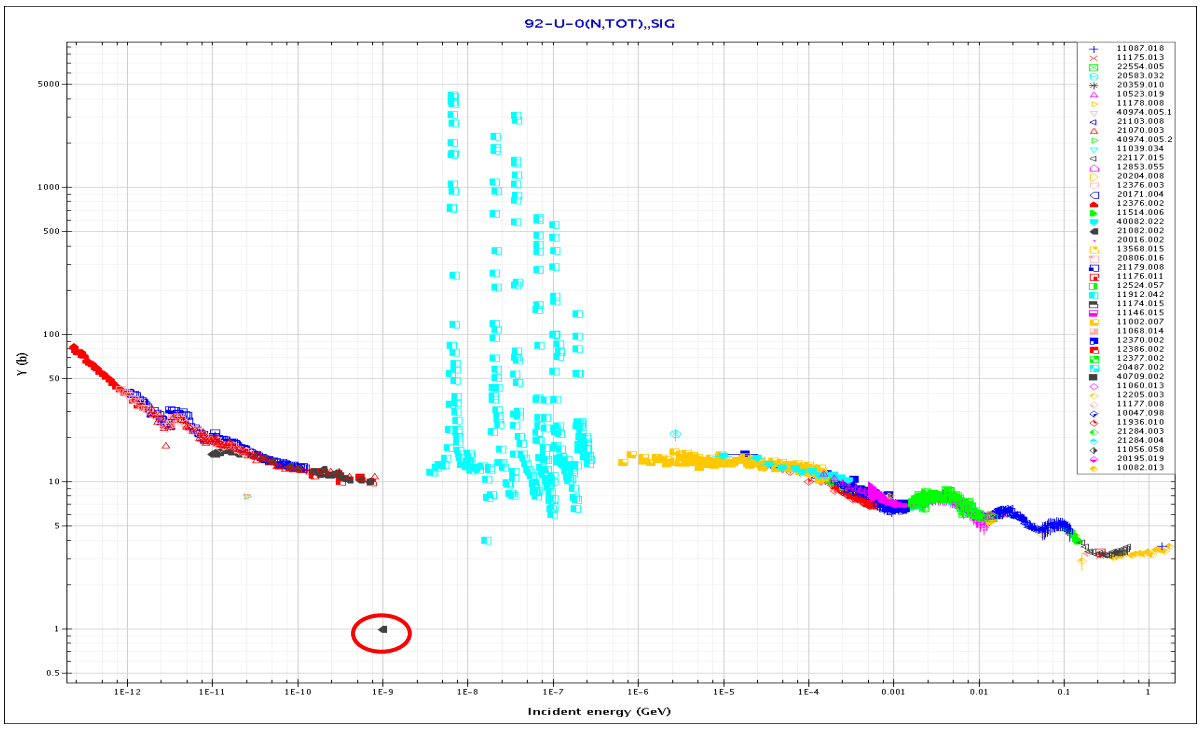


Figure 1 : Exemple d'un point anormal, dans une fonction comportant une zone de résonance

Exemple d'une série aberrante avec zone de résonance

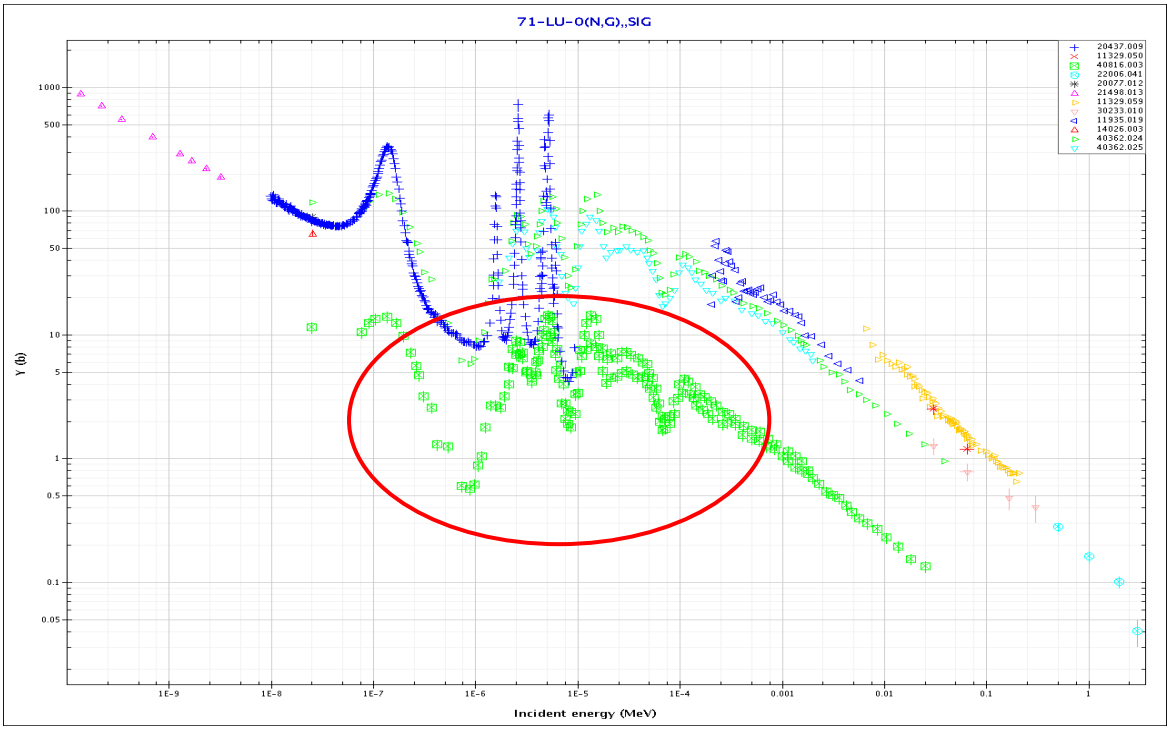


Figure 2 : Exemple d'une série aberrante avec zone de résonance

Exemple de mesures suspectes, mais non fausses compte tenu de l'incertitude

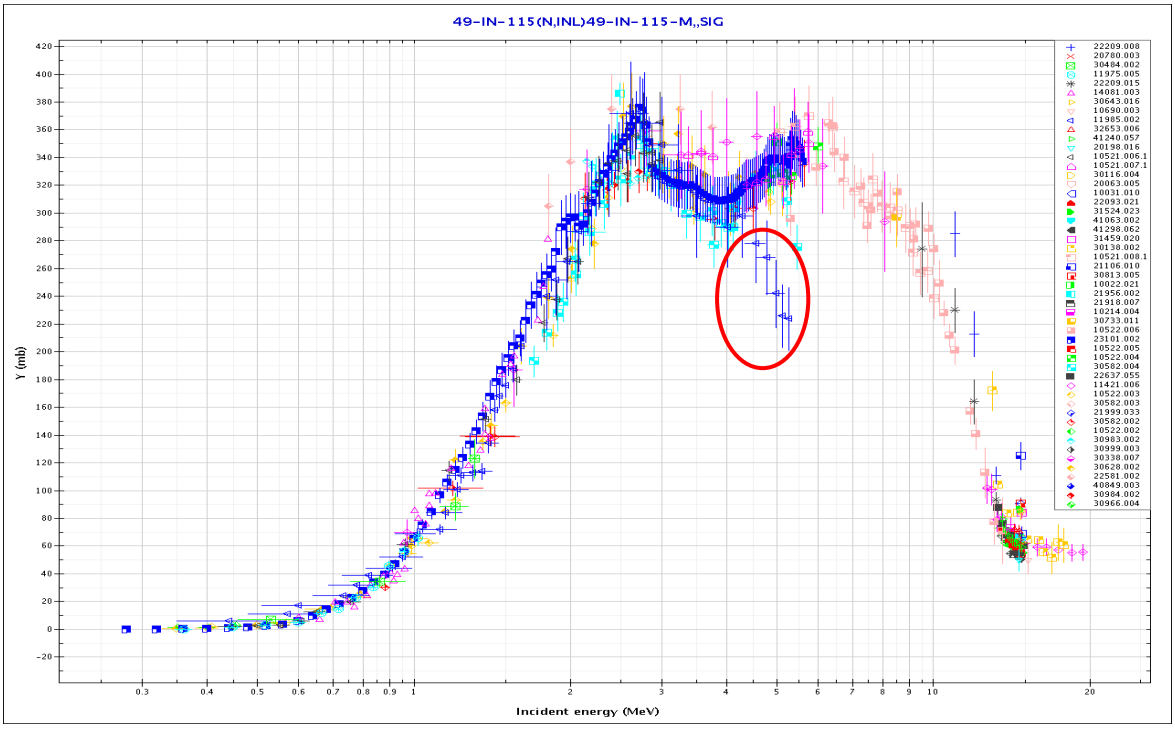


Figure 3 : exemple de mesures suspectes mais acceptables compte-tenu des incertitudes

Exemple de séries de mesures non cohérentes

Dans l'exemple ci-dessous, l'une des deux tendances est fausse, mais il n'est pas possible de savoir laquelle :

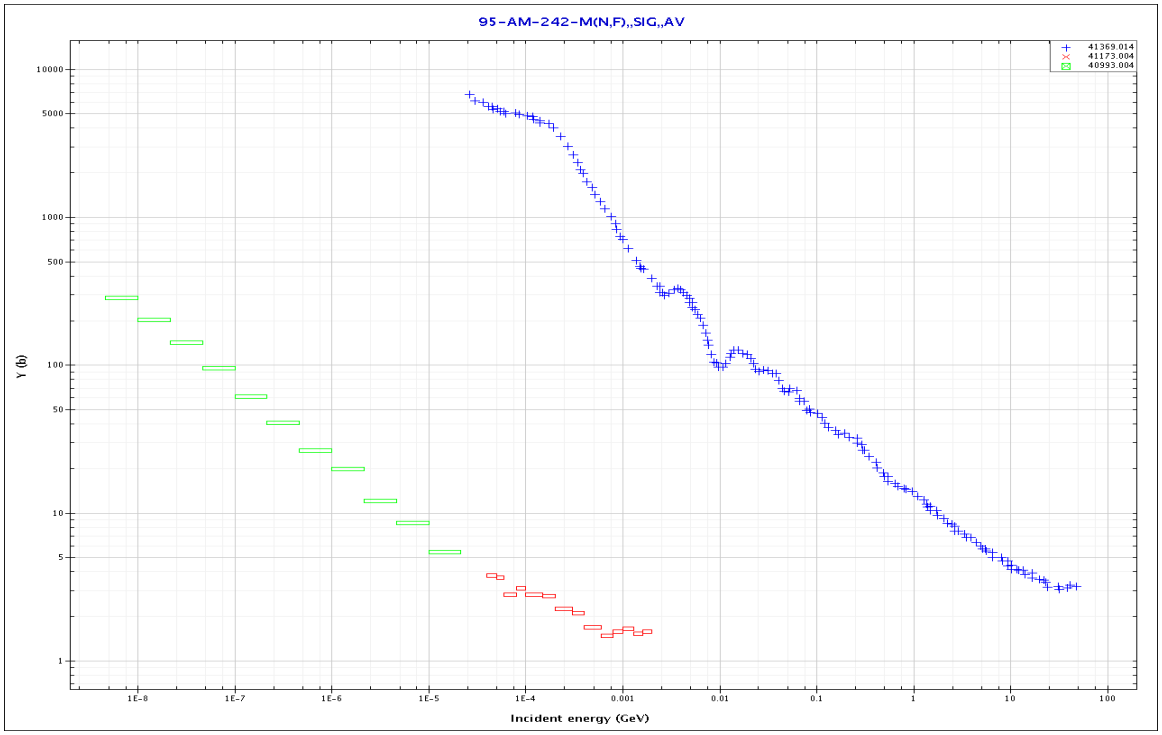


Figure 4 : Exemple de séries de mesures incohérentes

Après l'analyse des différents cas tests, nous avons défini une anomalie comme suit : une anomalie est une mesure ou un ensemble de mesures (série) qui se démarque de la tendance de l'ensemble des points de l'EHD.

Méthode de détection des anomalies

L'approche mise en place pour l'automatisation de la détection de données aberrantes s'inspire de la définition d'une anomalie et de la méthode actuellement utilisée par l'AEN : un expert visualise les graphes des résultats, et détermine à l'œil nu l'existence de données ou séries suspectes pour une réaction nucléaire. La méthode développée par la SCM permet d'automatiser cette détection, et de classer les anomalies en fonction de leur degré de « suspicion ».

Recherche de l'échelle optimale

On s'aperçoit alors que le choix de l'échelle de représentation des mesures est très important : suivant le choix de l'échelle, on peut passer à coté d'anomalies ou au contraire définir comme aberrantes des mesures qui ne le sont pas. Nous avons donc mis en place une méthode permettant d'identifier de manière automatique l'échelle optimale pour la représentation des résultats.

Il existe différents types d'échelles pour la représentation de données, notamment les échelles linéaires ($g_k = k$), logarithmiques ($g_k = \alpha^k$), polynomiales ($g_k = k^\alpha$), exponentielles ($g_k = log_\alpha(k)$), etc. Parmi les différentes échelles existantes, nous cherchons celle permettant de visualiser l'ensemble des valeurs avec la même précision : c'est cette échelle qui permettra une visualisation optimale des données.

Nous avons décidé de nous limiter à l'étude des échelles linéaires et polynomiales :

- L'échelle linéaire attribue la même importance à l'ensemble des valeurs. Cette échelle est donc adaptée lorsque les valeurs sont distribuées à des intervalles équidistants ;

- L'échelle polynomiale contracte les grandes valeurs et dilate les valeurs les plus faibles. Le choix de cette échelle est approprié lorsqu'il existe une concentration importante des valeurs faibles, qu'il y a peu de valeurs élevées et que celles-ci sont très dispersées, ce qui est le cas des données provenant des réactions nucléaires de la base EXFOR. De plus, cette échelle est similaire à l'échelle logarithmique utilisée par l'AEN, mais plus simple à mettre en œuvre.

Nous donnons ci-dessous deux exemples de réactions avant et après l'utilisation de l'échelle optimale permettant de visualiser les données. L'utilisation de ce procédé entraîne une modification importante de la configuration des graphiques qui affecte également la détection des anomalies :

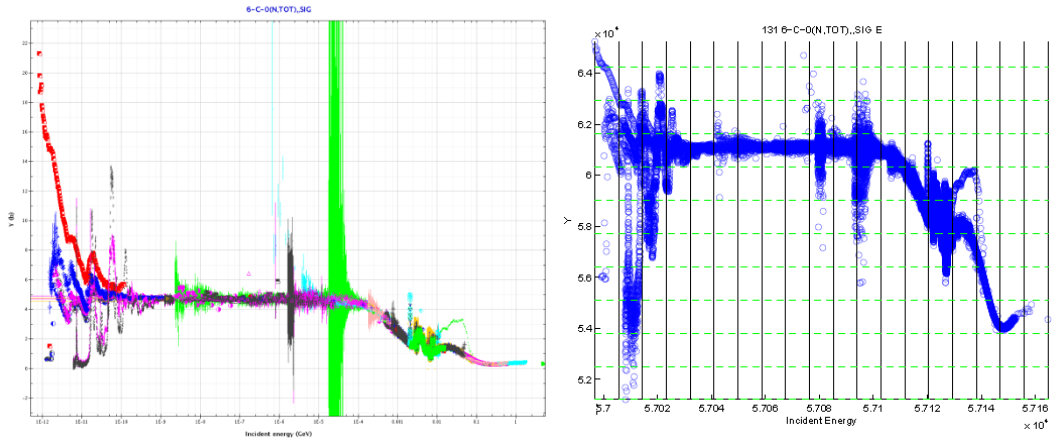


Figure 5 : Visualisation de la réaction `131 6-C-O(N,TOT),,SIG E` avant et après la mise en place de l'échelle optimale

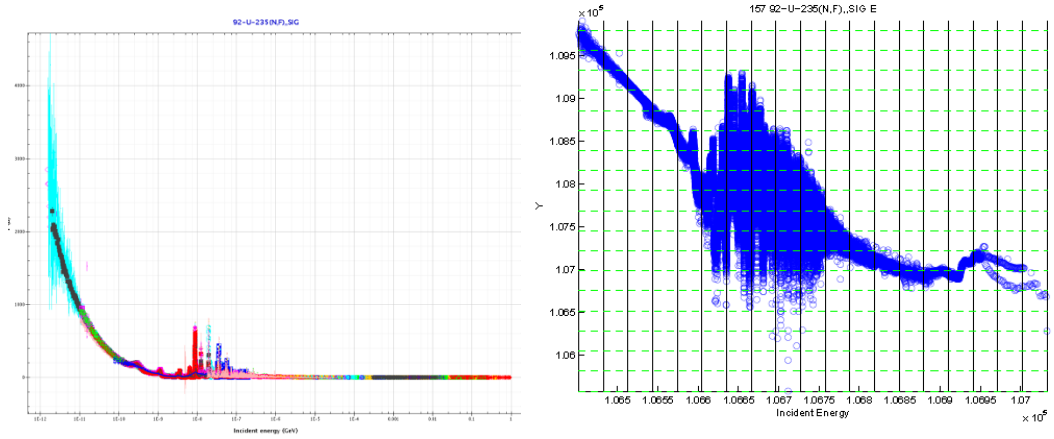


Figure 6 : Visualisation de la réaction `157 92-U-235(N,F),,SIG E` avant et après la mise en place de l'échelle optimale

Détection des anomalies et indicateurs

L'approche choisie est probabiliste ; elle permet de repérer à la fois les singularités isolées et les groupes de données cohérents entre eux mais suspects par rapport à l'ensemble des données. La méthode consiste à discrétiser le plan en tranches verticales. Sur chacune, nous construisons l'histogramme des mesures : si un point de mesure tombe dans une case de la discrétisation en y, on ajoute 1 à l'histogramme.

L'étude de ces histogrammes nous permet ensuite de déterminer la présence de singularités : nous considérons que l'EHD présente une anomalie lorsqu'au moins une loi de probabilité présente une ou plusieurs discontinuités, c'est-à-dire lorsque la loi est composée de deux ou plusieurs ensembles distincts, séparés par un certain nombre d'intervalles vides. Plus cette distance (nombre d'intervalles vides séparant les ensembles de données) est élevée, plus la probabilité de détecter une vraie anomalie est importante.

L'exemple suivant représente un histogramme sur une tranche verticale comportant une anomalie :

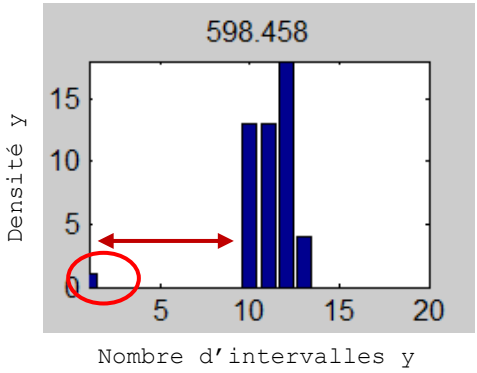


Figure 7 : Loi de probabilité des mesures présentant une anomalie

Les deux graphiques ci-dessous représentent chacun une tranche verticale pour laquelle la distribution est continue. En conséquence, on ne détecte pas d'anomalies.

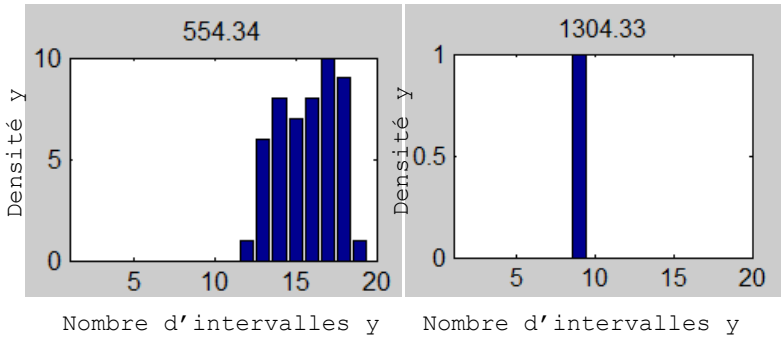


Figure 8 : Lois de probabilité des mesures sur les troisième et cinquième tranches des abscisses

Cette méthode est adaptée aux particularités propres aux réactions nucléaires ; en particulier, elle permet une identification robuste des anomalies, même si on ne dispose pas de la totalité des données ou s'il existe des zones de résonance d'amplitude très importantes en comparaison au reste de la tendance. De plus, la présence de ces zones de résonance impose l'étude de l'histogramme complet : sur les autres parties des fonctions, l'étude des indicateurs de dispersion classiques (variance, écart-type) permettrait de détecter la présence d'anomalies ; elle n'est pas pertinente dans les zones de résonance, où il est normal que la dispersion soit importante.

Indicateurs

La détection d'anomalies est réalisée à l'aide de plusieurs indicateurs. Le premier est la distance (nombre d'intervalles) séparant les points constituant l'anomalie du reste des points dans l'histogramme : plus sa valeur est élevée, plus la probabilité que le ou les points repérés soient effectivement une anomalie est élevée.

Cet indicateur de distance est donc pertinent, mais il n'est pas suffisant. En effet, dans la base de données EXFOR, il est courant d'observer des séries des données décalées de seulement quelques intervalles vides (2 ou 3), mais dont la tendance est clairement différente de celle de l'ensemble des points. Il s'agit donc bien d'anomalies, dont l'importance ne sera pas quantifiée de manière satisfaisante par l'indicateur de la distance. Afin d'identifier ces cas, nous avons

ajouté un nouvel indicateur qui indique le nombre de fois qu'une même série est identifiée comme suspecte pour la même réaction.

Par exemple, le graphique suivant représente une EHD ou réaction composée de plusieurs séries de données : le graphique de gauche (représentation de l'AEN) permet de distinguer les différentes séries de données ; celui de droite correspond aux résultats de la même réaction, représentés après recherche de l'échelle optimale et discrétisation des axes.

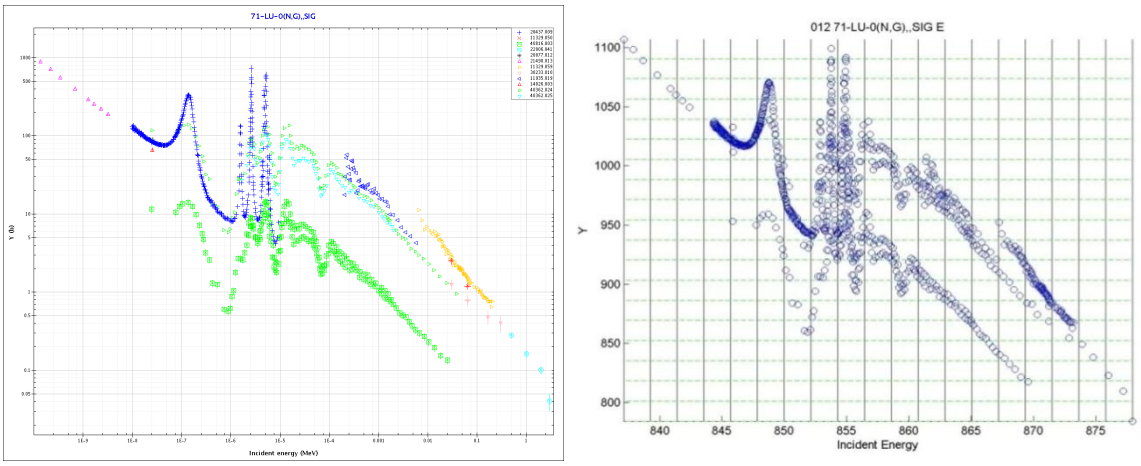


Figure 9 : Représentation d'une EHD composée de plusieurs séries de données

Nous pouvons observer sur le graphique de droite que le nombre des cases vides n'est pas supérieur à deux, quelle que soit la tranche verticale considérée. Toutefois, l'existence d'une anomalie est incontestable : la même série est identifiée comme aberrante sur 8 tranches verticales (Figure 10).

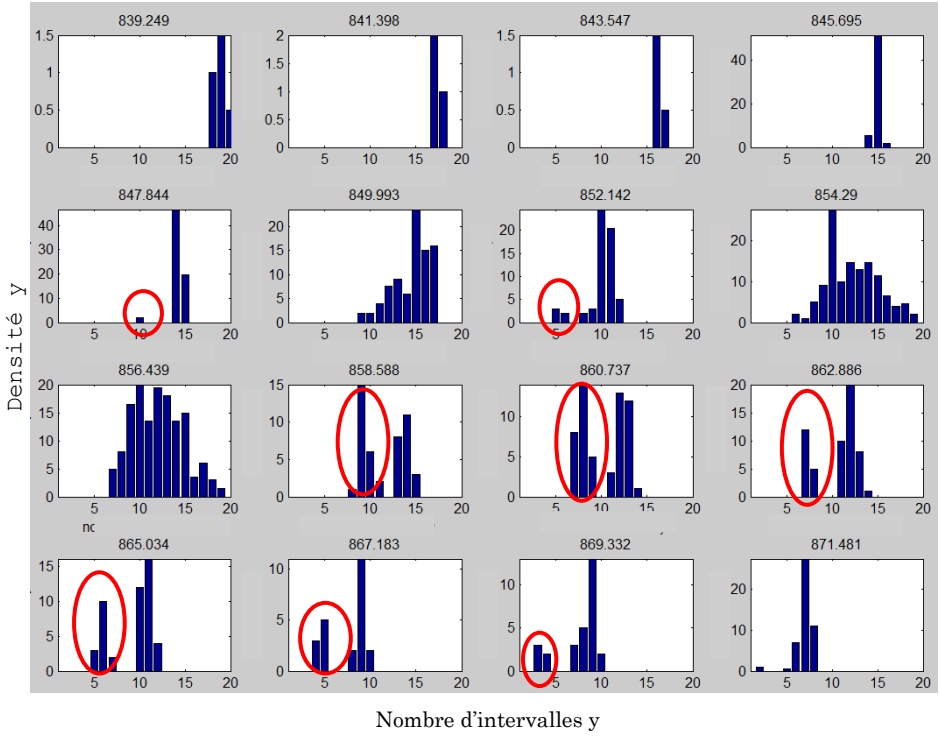


Figure 10 : Lois de probabilité des mesures sur les seize premières tranches verticales

Les résultats finaux doivent donc être triés en fonction de la distance *nombre_carrés* mais aussi en fonction du nombre de fois où la série est définie comme aberrante *nb_serie_aberrante*. Par exemple, si $nombre\_carrés = 2$ mais $nb\_serie\_aberrante = 8$, il y a sûrement une anomalie. L'utilisateur pourra alors traiter les cas les plus urgents (c'est-à-dire ceux dont la probabilité d'être aberrants est la plus élevée) en priorité, laissant les possibles faux positifs à la fin de la liste (distance et nombre de fois où la série est identifiée comme aberrante faibles). Le nombre total d'anomalies repérées dans une EHD doit également être pris en compte dans le traitement des fichiers.

Prise en compte des incertitudes et de la résolution des mesures

La majorité des mesures collectées par l'AEN sont entachées d'incertitudes, de sources diverses :

- incertitude sur la masse de l'échantillon ;
- variation du nombre et/ou de l'énergie des particules incidentes sur l'échantillon au cours de la mesure ;
- incertitude statistique liées aux taux de comptage ;
- incertitude des corrections appliquées à la mesure brute, e.g. temps mort de l'électronique, absorption des particules dans l'échantillon, efficacité des détecteurs…
- normalisation de la mesure.

De plus, la résolution (précision des instruments) n'est pas toujours optimale. La méthode a donc été approfondie afin de prendre en compte ces aspects.

Nous représentons l'incertitude et la résolution verticales par des lois uniformes : cela signifie que la mesure peut être égale à n'importe quelle valeur entre les bornes de l'intervalle, avec une probabilité identique pour toutes les valeurs. Il s'agit d'un choix de la SCM : en effet, l'incertitude est supposée être représentée par une loi normale, mais cette information n'est pas nécessaire pour la détection des anomalies ; seul le support de la loi importe. La modélisation est plus grossière, mais les résultats de la détection d'anomalies sont identiques, en réduisant de manière significative les temps de calcul et la complexité des algorithmes.

Le principe de détection reste le même lorsque l'on prend en compte les incertitudes : si la loi uniforme représentant l'incertitude de la valeur croise une case de discrétisation en y, alors on ajoute $\frac{1}{N}$ à l'histogramme de cette case, N étant le nombre de cases sur lesquelles s'étale l'intervalle d'incertitude.

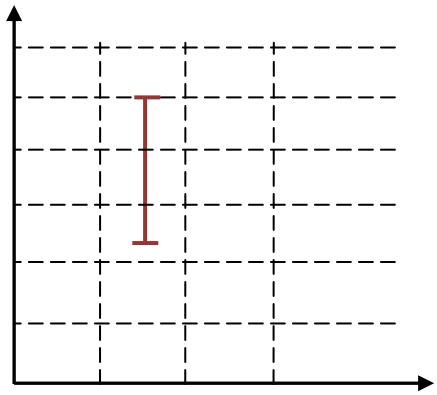


Figure 11 : Exemple de mesure avec incertitude

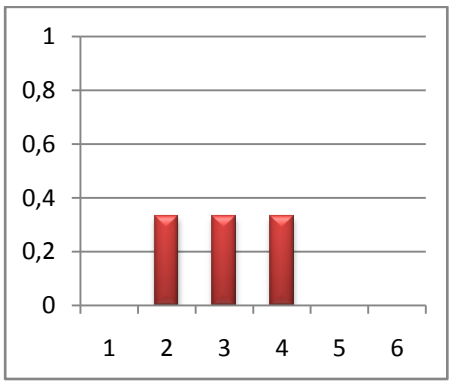


Figure 12 : Loi de probabilité des mesures sur la deuxième tranche verticale

En conséquence, le nombre de cases occupées par une mesure est d'autant plus grand que la résolution faible et l'incertitude élevée.

Si l'incertitude porte sur l'axe horizontal, le principe est identique : la mesure sera prise en compte dans la loi de probabilité des tranches verticales que croise la mesure incertaine.

Sorties de l'outil

La méthode a été implémentée sous Matlab, afin de réaliser la validation sur les cas tests. La méthode a ensuite été intégrée aux outils de l'AEN, dans les codes adaptés aux outils existants.

L'outil identifie la présence de données suspectes parmi les mesures d'une réaction donnée, et fournit, pour chaque anomalie détectée (ou pour chaque série dont les points font partie d'une anomalie), les indicateurs suivants :

- Le nombre d'intervalles vides délimitant la discontinuité dans la loi de probabilité. Plus ce nombre (supérieur ou égal à 1) est élevé, plus la mesure est suspecte ;

- Le nombre de points de mesure formant l'anomalie ;

- Le poids des données identifiées comme suspectes, par rapport à l'ensemble des données contenues dans la tranche verticale. Cette information, combinée avec le nombre de points, est intéressante car elle peut permettre d'identifier la nature de l'anomalie : si le poids des données suspectes et le nombre de points sont très faibles, il s'agira probablement d'un problème ponctuel. Par contre, un poids et un nombre de points élevés suggèrent un problème récurrent touchant plusieurs mesures;

- Le nom de la série et le nombre de fois que la série a été identifiée comme suspecte. Plus ce dernier indicateur est élevé, plus la série a de chances d'être suspecte.

Ces indicateurs renseignent sur le degré de suspicion des points repérés.

Nous avons également développé un module de recherche qui permet de repérer les données suspectes dans la base de données originale (sous format Excel). Le logiciel identifie les inter-

valles sur l'axe x et sur l'axe y qui comprennent des données suspectes. Ensuite, il recherche sur la base de données originale les données comprises dans les intervalles suspects. En sortie, le logiciel indique le numéro de ligne de la base de données originale qui contient des données suspectes.

Le format des résultats est le suivant :

nom fichier	nom serie	nombre de fois série suspecte	inter-valle x minimum	inter-valle x maximum	inter-valle y minimum	inter-valle y maximum	nombre carrés	poids	nombre données aberrantes	numéro ligne

Tableau 1 : Format des résultats

Validation de la méthode : résultats, difficultés et limites

Résultats

La méthode a été validée sur l'échantillon de 113 réactions fourni par l'AEN. Le tableau ci-dessous présente les résultats obtenus sur les deux types de cas tests (échantillon représentatif et échantillon de particularités). Nous avons considéré que les données suspectes sont aberrantes lorsque la distance ou le nombre de fois que la série apparait comme suspecte dépassait au moins le seuil de 2 (la valeur d'un de ces deux indicateurs doit être strictement supérieure à deux).

Type	**taille**	**Nb fichiers**	**Nbre cas aberrants**	**cas de nuage de points**	**Nbre total d'anomalies**	**Nbre de faux positifs**
CS cas représentatifs	**<500 ko**	**50**	**15**	**5**	**118**	**6**
CS cas représentatifs	**>500 ko**	**5**	**0**	**1**	**0**	**2**
CS cas difficiles	**<500 ko**	**49**	**23**	**0**	**86**	**0**
CS cas difficiles	**>500 ko**	**8**	**3**	**0**	**3**	**0**
Total	***nombre***	***113***	***41***	***6***	***207***	***8***
	%		***36.3%***	***5.3%***		***3.9%***

Tableau 2 : Résultats de la validation

Le nombre de faux positifs doit être le plus faible possible, notamment sur l'échantillon représentatif de la base totale : il permet de prévoir si la méthode sera efficace sur toute la base EXFOR. Celui-ci est estimé à 3.9% sur le nombre total d'anomalies repérées, et 6.7% sur l'échantillon représentatif de la base, ce qui est faible. De plus, 7 cas sur ces 8 identifiés comme faux positifs ne constituent pas des anomalies importantes (la distance n'est égale qu'à 3 intervalles vides). En fait, seul le dernier cas constitue un « vrai » faux positif. Cette erreur provient d'une discrétisation trop fine pour ce cas, où les données sont peu nombreuses.

On remarque donc que le choix de la discrétisation des axes est un point essentiel : une discrétisation trop fine peut engendrer un nombre de faux positif important, alors qu'une discrétisation trop grossière peut « laisser passer » des anomalies. Un compromis doit donc être trouvé. Dans ce but, nous avons testé une discrétisation à 19 intervalles (20 bornes) pour les axes d'ordonnées et d'abscisses. Celle-ci fonctionne correctement dans la plupart des cas sauf pour

les « nuages des points » (qui représentent 5% des cas traités), ou pour des cas composés d'un faible nombre de mesures.

Difficultés liées aux particularités de la base EXFOR : ajustements de la méthode

La principale difficulté de l'étude tient à la grande diversité du type de réactions et du type d'anomalies : certains cas étaient prévisibles mais d'autres ont été identifiés au cours de la validation des résultats. La phase de validation a donc entraîné de nombreuses adaptations du code.

- Les mesures peuvent se présenter alignées verticalement si elles partagent la même valeur de x. Le principe de la méthode reste inchangé mais il a fallu adapter le code à ce cas.

- Il est fréquent de trouver des données à faible valeur avec des incertitudes très élevées ; ceci peut affecter la représentation graphique et donc la détection d'anomalies.

  Dans un premier temps, nous avons testé l'apport de la prise en compte des incertitudes dans la recherche de l'échelle optimale : l'implémentation de cette information supplémentaire ne permet pas d'améliorer la détection des anomalies, et augmente de manière significative le temps de calcul. La figure suivante illustre ce point :

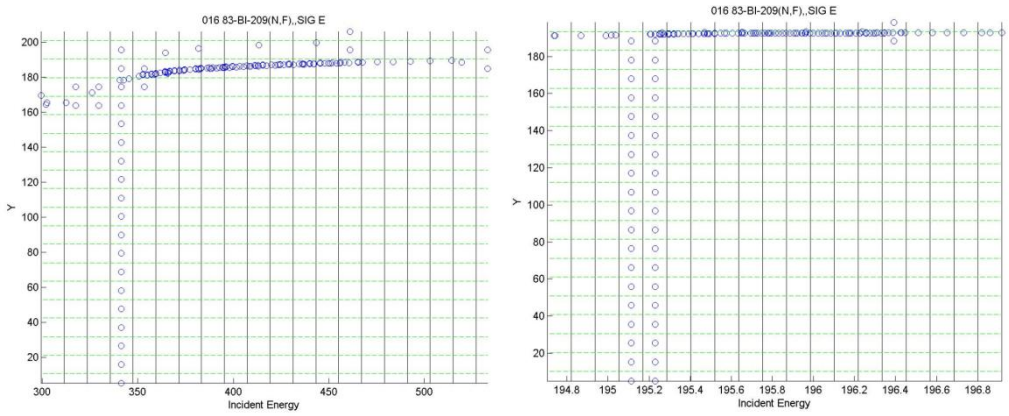


Figure 13 : Représentation de la réaction `016 83-BI-209(N,F),,SIG E` avant et après la prise en compte des incertitudes dans la recherche de l'échelle optimale

  Afin de pallier cette difficulté, nous avons finalement limité l'étendue de l'incertitude : la prise en compte des incertitudes nous permet de ne pas identifier comme suspectes des données qui ne le sont pas du fait de leur incertitude. Le fait que l'incertitude d'une donnée soit tellement élevée qu'elle arrive à des niveaux où il n'y a pas d'autres valeurs ne nous apporte pas d'information supplémentaire. Nous avons donc limité le nombre d'intervalles de discrétisation sur lesquelles la donnée peut s'étaler.

  L'exemple suivant montre l'intérêt de cette limitation ; la figure de gauche représente la réaction avant limitation de l'étendue de l'incertitude, celle de droite après limitation :

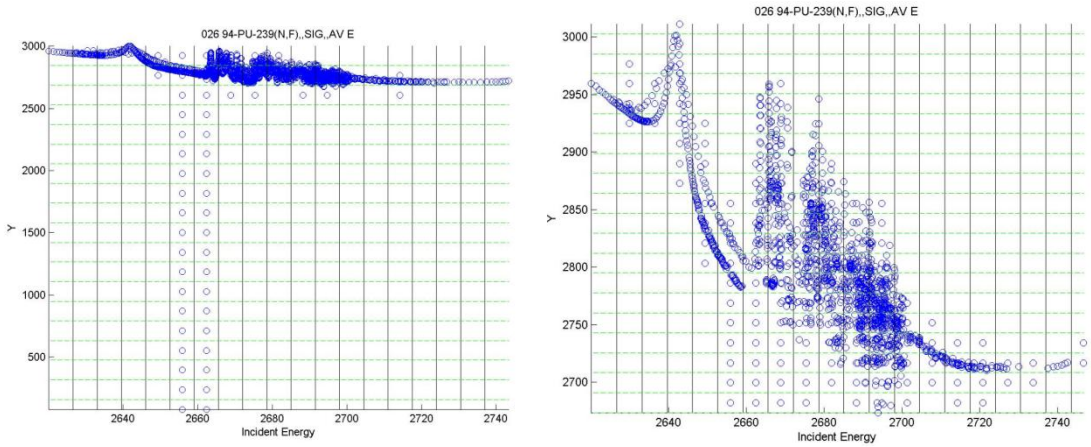


Figure 14 : Représentation de la réaction `026 94 -PU-239 (N,F),,SIG,,AV E` avant et après la limitation de l'étendue de l'incertitude

- Les mesures peuvent contenir des valeurs nulles. Le problème est identique à celui du point précédent : la représentation graphique de la réaction n'est pas optimale, comme on peut le voir sur l'exemple suivant :

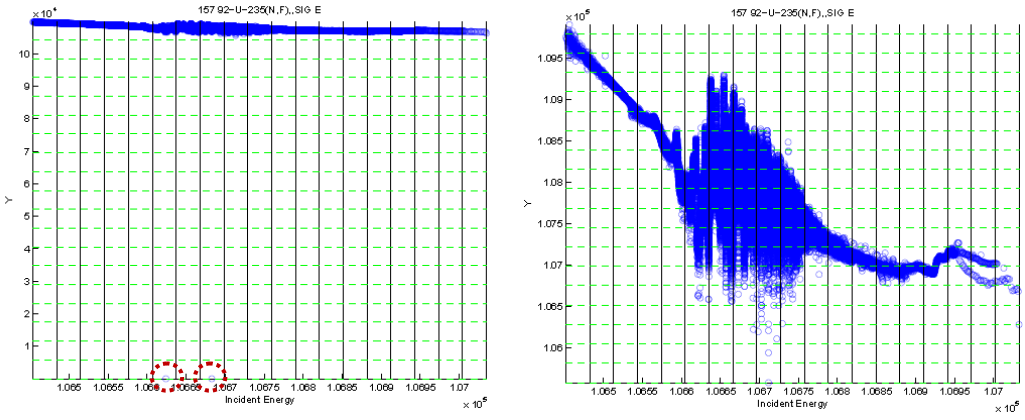


Figure 15 : Visualisation de la réaction `157 92-U-235(N,F),,SIG E` avec et sans données nulles

Le graphique à gauche représente la réaction avec les mesures de valeur nulle (entourées en rouge) alors qu'à droite, on représente cette même réaction en supprimant ces valeurs. Ces valeurs nulles ne sont pas aberrantes du fait de l'incertitude mais, elles peuvent nuire fortement à la qualité de la visualisation graphique, et donc à la détection des anomalies.

Pour résoudre ce problème, il n'y a pas de solution idéale. Nous avons choisi de supprimer les valeurs nulles des gros fichiers où la valeur d'alpha est très élevée.

- Il est possible de trouver des anomalies très éloignées des autres mesures. Comme pour les cas précédents, cela peut avoir un impact visuel sur la configuration du graphique : dans ce cas, on détectera cette anomalie mais on ne pourra pas identifier d'autres anomalies à plus courte distance. Afin d'éviter ceci, l'algorithme est appliqué deux fois en cas d'existence d'une valeur beaucoup plus faible que les autres. Toutefois, il faut définir un seuil en fonc-

tion de la discrétisation choisie. Ce seuil ne doit pas être trop faible pour éviter de faire tourner l'algorithme deux fois pour des cas qui ne sont pas nécessaires et ainsi identifier des faux positifs.

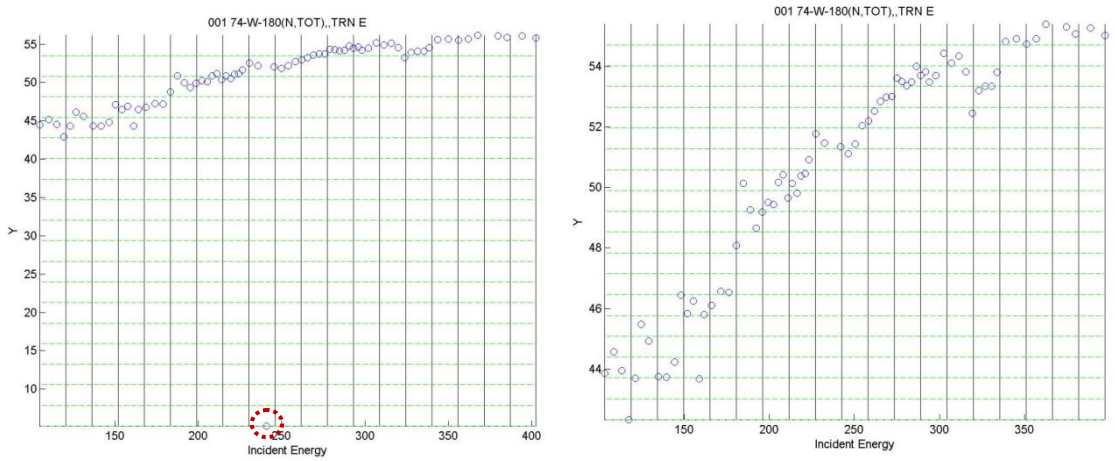


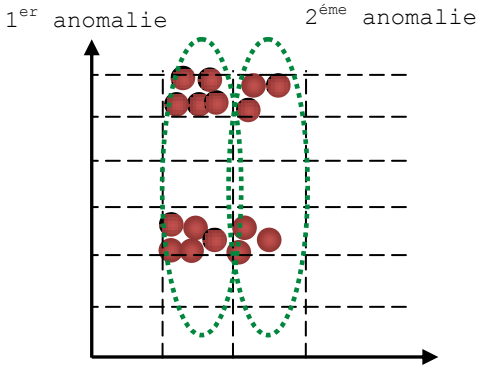
Figure 16 : Représentation de la réaction `001 74-W-180(N,TOT),,TRN E` *avant et après la suppression de la donnée aberrante*

- L'existence de valeurs négatives constitue un problème pour la détection de l'échelle optimale : si les données sont négatives et que nous cherchons l'échelle polynomiale optimale, nous obtiendrons des nombres complexes. Il a été décidé de supprimer les nombres négatifs, qui sont très peu nombreux. Cette solution n'est pas idéale, car on néglige une partie (très faible) de l'information, mais elle permet tout de même d'appliquer la méthode pour les autres données. En outre, les valeurs nulles et faibles sont toujours analysées mais, elles présentent souvent des incertitudes élevées. Afin d'éviter que ces données deviennent négatives, nous avons imposé que la valeur la plus faible (y compris l'incertitude) qu'une mesure peut prendre soit égale à zéro.

Limites de la méthode

La méthode développée peut être utilisée pour vérifier des fichiers de différente taille. Toutefois, la robustesse de la méthode, et donc le degré de confiance des résultats, seront liés au nombre de données ainsi qu'à la configuration. Par exemple, si nous ne disposons que de 10 mesures mais qu'elles présentent une tendance claire, la méthode fonctionnera correctement. Toutefois, si les données se présentent sous une configuration de nuage de points, l'identification des données aberrantes est plus complexe.

Dans le cas où deux tendances d'égale importance existent, la méthode ne permet que de signaler l'existence d'une anomalie, sans pouvoir identifier quelles sont les données aberrantes. Dans ce cas, l'outil développé identifie les deux groupes de poids égal à 0.5 comme suspects ; l'AEN devra ensuite approfondir l'analyse pour déterminer lequel est anormal.



	nombre carrés	poids	nb données aberrantes
1er anomalie	2	0.5	5
1er anomalie	2	0.5	5
2éme anomalie	2	0.5	3
2éme anomalie	2	0.5	3

Figure 17 : exemple d'anomalie avec un poids égal à 50 %

Comme évoqué plus haut, la méthode ne permet pas une détection optimale dans le cas de configurations de la forme « nuage de points » : cela signifie qu'on ne dispose que d'une certaine partie de la courbe, en « zoom ». La tendance peut alors être difficile à cerner, le nuage de points étant diffus. Ceci peut créer des faux positifs : le logiciel risque d'identifier des points comme étant des anomalies alors qu'ils ne le sont pas.

Pour ces cas particuliers, on peut chercher à optimiser la discrétisation des axes en fonction du nombre de points. Cependant, il n'existe pas de méthode a priori permettant de lier le nombre d'intervalles de discrétisation et le nombre de points. Il faut donc trouver cette relation dans la pratique et pour cela, une analyse de sensibilité s'impose.

Enfin, il ne faut pas oublier qu'il s'agit d'une méthode inspirée par la détection réalisée par l'œil humain : si un expert ne peut pas conclure, le logiciel n'y parviendra pas non plus. Néanmoins, ces cas sont peu nombreux sur la base EXFOR.

Améliorations possibles

Les voies d'améliorations sont multiples. Nous proposons les suivantes :

- Optimisation du temps de calcul : il est possible d'améliorer le programme développé en termes de temps de calcul, notamment pour la recherche de l'échelle optimale ;

- Approfondissement des cas difficiles : les configurations en forme de nuage de points ainsi que les cas composés d'un faible nombre de données peuvent être étudiés plus en détail. Comme indiqué précédemment, il faudrait réaliser plusieurs analyses de sensibilité afin de trouver une discrétisation des axes optimale pour traiter ces particularités.

- Développement de la prise en compte des incertitudes : il est possible d'intégrer les incertitudes des mesures sur l'axe des abscisses. Il suffit d'appliquer le même principe que pour la prise en compte des incertitudes en ordonnées. Toutefois, cette implémentation demande une modification profonde du code.

- Application de la méthode à de nouvelles bases de données : le principe de détection de données reste pertinent, il faut seulement adapter le code aux nouveaux cas particuliers. Par exemple, si la nouvelle base de données présente des données négatives, il faudrait peut-être inclure une autre échelle de visualisation des données à la place de l'échelle po-

lynomiale. En outre, la méthode peut être élargie aux cas 3D et autres cas multidimensionnels.

- Méthode complémentaire de détection d'anomalies : il est possible de développer d'autres méthodes afin d'identifier les données aberrantes. Par exemple, il serait intéressant d'analyser la continuité de la loi de probabilité. Pour cela, il faut étudier l'évolution de l'espérance : la moyenne de l'ensemble des mesures contenues dans la même tranche verticale. S'il existe une modification importante de cette valeur, on pourra alors signaler la présence d'une discontinuité horizontale dans la distribution des mesures. Cette méthode pourrait donc être utilisée en complément de la méthode actuelle. Elle est facilement implémentable et intégrable au code actuel.

Difficultés rencontrées

Comme le détaille la partie précédente, la principale difficulté tient à la grande diversité des types de réactions et des types d'anomalies. D'abord, cette particularité des données EXFOR contraint à ce que la méthode soit la plus générale et la plus robuste possible. Ensuite, la phase de validation a été très longue :

- Les fichiers à tester sont nombreux. De plus, certains contiennent un nombre de données très important ; le temps de calcul pour les plus gros fichiers peut dépasser les 24 heures ;

- Cette phase nous a permis d'identifier des cas particuliers nécessitant des adaptations du code : incertitudes très élevées, valeurs négatives, nulles, etc.

Sommaire

Table des illustrations

I. Problématique

L'Agence pour l'énergie nucléaire (AEN) est une agence spécialisée de l'Organisation de coopération et de développement économiques (OCDE). L'AEN a pour mission d'aider ses pays membres à maintenir et à approfondir les bases scientifiques, technologiques et juridiques en rapport avec l'énergie nucléaire.

L'AEN dispose ainsi de plusieurs bases de données, qu'elle enrichit constamment et met à la disposition de personnes intéressées. Il y en a une dizaine, notamment :

- International Criticality Safety Benchmark Evaluation Project ;
- Information System on Occupational Exposure ;
- OECD Piping Failure Data Exchange ;
- OECD/AEN Fire Incidents Records Exchange (FIRE) Project ;
- OECD/AEN Stress Corrosion Cracking and Cable Ageing Project ;
- EVA: Evaluated Nuclear Data ;
- CINDA: Bibliographical information on nuclear reaction data ;
- Le "red book" et "brown book", qui concernent les ressources et la demande en uranium ;
- La base EXFOR.

De manière générale, l'AEN souhaite que ces bases soient de la meilleure qualité possible. L'Agence veut donc disposer d'outils permettant la vérification des bases et la détection d'enregistrements erronés. Ces outils doivent être "flexibles", en ce sens que l'on doit pouvoir les utiliser sur les diverses bases.

Les sources d'erreurs sont multiples : elles peuvent résulter d'erreurs d'origine humaine lors de la mesure proprement dite ou lors du traitement des données (erreur d'unité, erreur dans la transcription de la mesure, etc), ou bien de l'instrument de mesure (mauvaise calibration). Ces erreurs se manifestent sur les bases sous forme de « singularités isolés » (valeurs isolées manifestement aberrantes), ou bien sous forme des « groupes de données » cohérents entre eux mais suspects par rapport l'ensemble des données.

Cette étude se concentre sur la base EXFOR, constituée de relevés de réactions nucléaires. Cette base représente plus de 19 000 expériences et contient plus de 130 000 séries des données. Elle contient principalement des données numériques et expérimentales ainsi que des informations bibliographiques sur les expériences de faibles et moyenne énergie (jusqu'à 1 GeV) pour des neutrons incidents, pour des particules chargées ($A \leq 12$) et pour des réactions de photon induites sur un large éventail d'isotopes, d'éléments naturels, et de composés. Cette base est interrogeable en ligne : http://www.nea.fr/dbdata/x4/.

Les méthodes mises en place doivent maitriser le taux de faux positif, c'est-à-dire le nombre de mesures définies comme anomalies à tort.

II. Données de l'étude

Les données sont regroupées sous la forme d'Ensembles Homogènes de Données (EHD) : il s'agit de données comparables entre elles (même réaction nucléaire, même quantité mesurée). La méthode s'applique à des bases de données avec un nombre de points minimum : nous ne traitons que les EHD constitués d'au moins 5 séries de mesure (5 expériences différentes) ou au moins 10 points de mesure (pouvant provenir d'une ou plusieurs expériences). Ceci représente au total 28 000 EHD de la base EXFOR.

Les données sont représentées sous la forme de nuages de points 2D (par exemple mesure de la section efficace (CS) en fonction de l'énergie incidente) ou 3D (distribution angulaire (DA), distribution en énergie (DE), rendement de fission (FY)…).

Ces EHD représentent des fonctions continues, pouvant comporter des zones de résonance. Les EHD peuvent être complètes (continuité du nuage de points), ou par morceaux. La quantité de données constituant un EHD est très variable ; certains contiennent un nombre de mesures très faible. Dans ce cas, la détection d'une anomalie peut être délicate.

A la plupart des mesures sont associées des incertitudes, par rapport aux différents paramètres (x, y, z). Celles-ci sont liées aux conditions d'expérimentation et doivent être prises en compte dans la détection des anomalies.

Nous disposons d'un échantillon de 125 cas représentatifs des anomalies existant dans la base de données. Parmi celles-ci, 113 cas représentent des mesures de section efficace en fonction de l'énergie incidente (celle-ci est produite par des protons ou par des neutrons). Il s'agit du cas le plus représenté dans la base EXFOR et en conséquence, la méthode développée a été ajustée en fonction de cet échantillon. L'échantillon est constitué de deux groupes de données :

- Le premier, composé de 55 cas test, est un échantillon représentatif de la base de données EXFOR ;
- Le second est composé de 64 cas, regroupant des anomalies très particulières, difficiles à traiter. L'objectif est de pouvoir utiliser la méthode de détection de données aberrantes même pour les cas les plus rares.

Cet échantillon nous a permis de mettre en place des méthodes de détection de configurations suspectes, permettant ainsi d'automatiser cette opération de vérification, réalisée avant manuellement par l'AEN.

Dans l'avenir, le principe de détection de données aberrantes, mise en place pour des problèmes en deux dimensions (2D), peut être étendue aux problèmes 3D.

III. Définition d'une anomalie

A. Définition

La première étape de l'étude consiste à définir ce qu'est une anomalie. Après l'étude des cas tests fournis par l'AEN, nous définissons une anomalie comme suit :

Une anomalie est une mesure ou un ensemble de mesures (série) qui se démarque de la tendance générale de l'EHD.

C'est par cette définition que les experts de l'AEN détectent visuellement les anomalies de la base ; la détection automatique par la méthode mise en place doit être basée sur les mêmes critères : une discontinuité dans la distribution des points composant l'EHD est une configuration suspecte. On s'aperçoit alors que l'aspect du graphique, et donc le choix de l'échelle de représentation des mesures, est très important : on peut passer à coté d'anomalies ou au contraire définir comme aberrantes des mesures qui ne le sont pas.

Nous présentons quelques exemples d'anomalies présentes dans la base EXFOR, afin d'illustrer cette définition :

Exemple d'une série de mesures aberrante, sans résonance

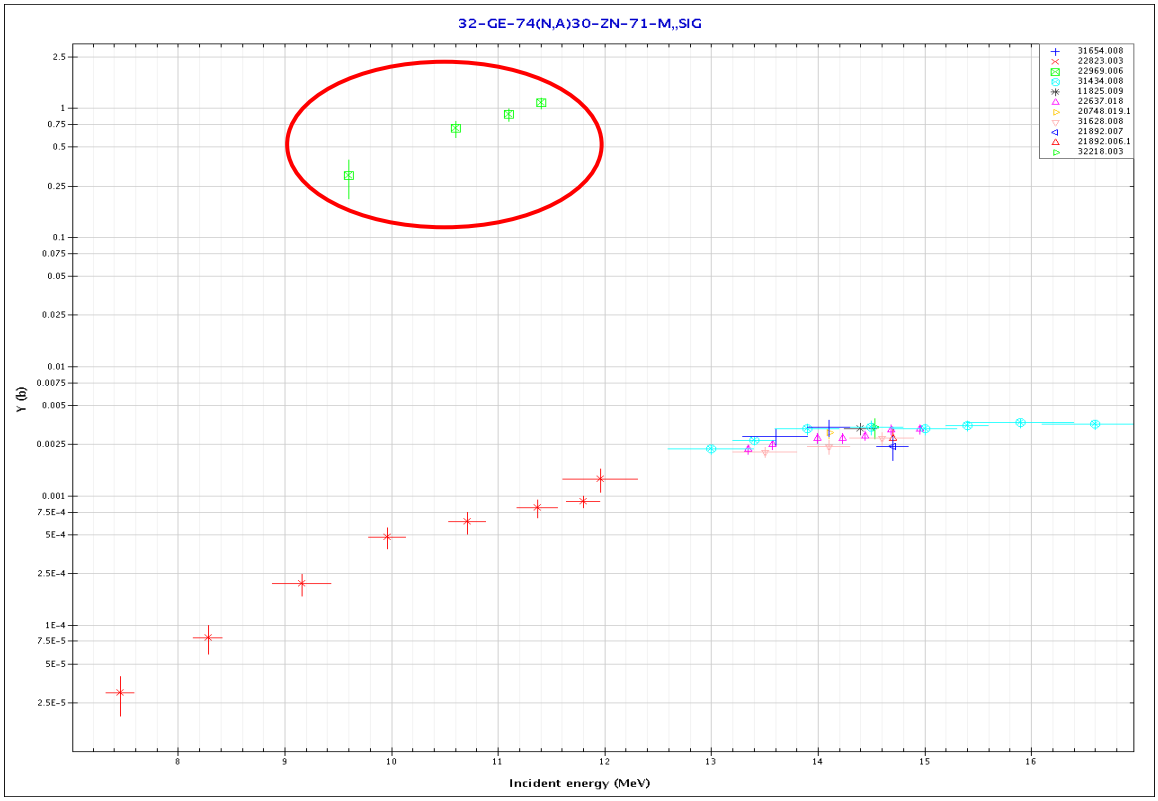


Figure 18 : Exemple d'une série de mesures aberrante

Exemple un seul point anomalie, avec zone de résonance

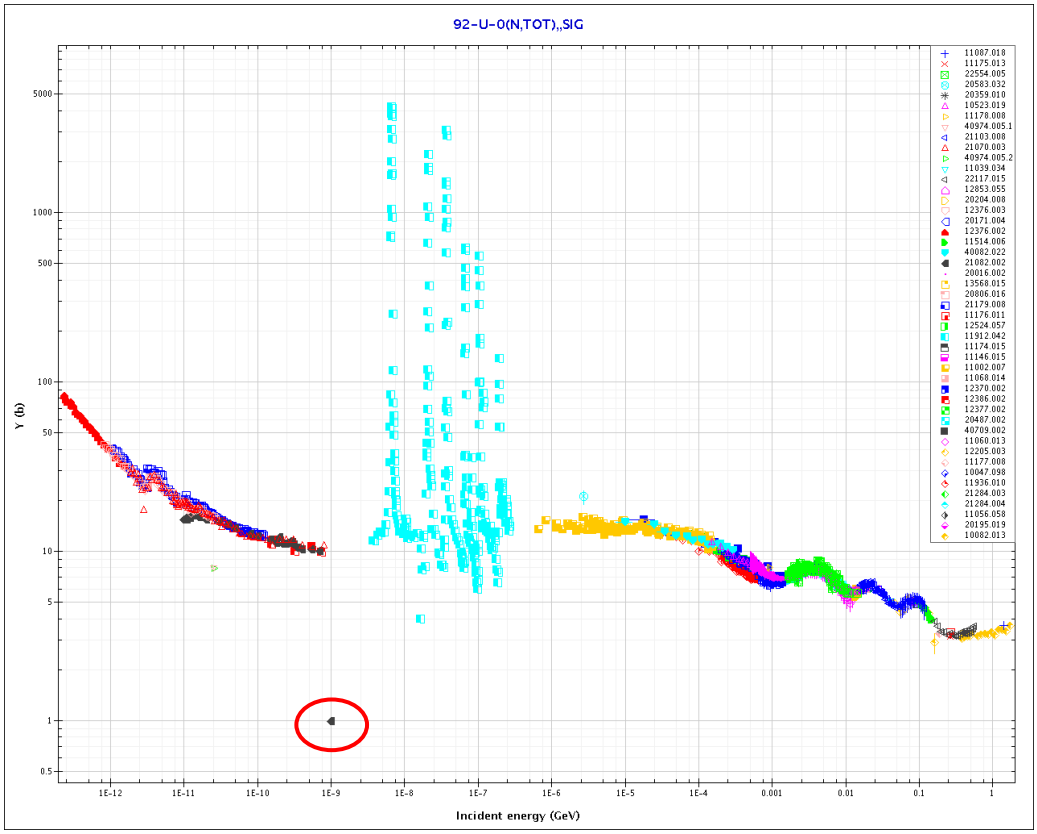


Figure 19 : Exemple d'un point anormal, dans une fonction comportant une zone de résonance

Exemple d'une série aberrante avec zone de résonance

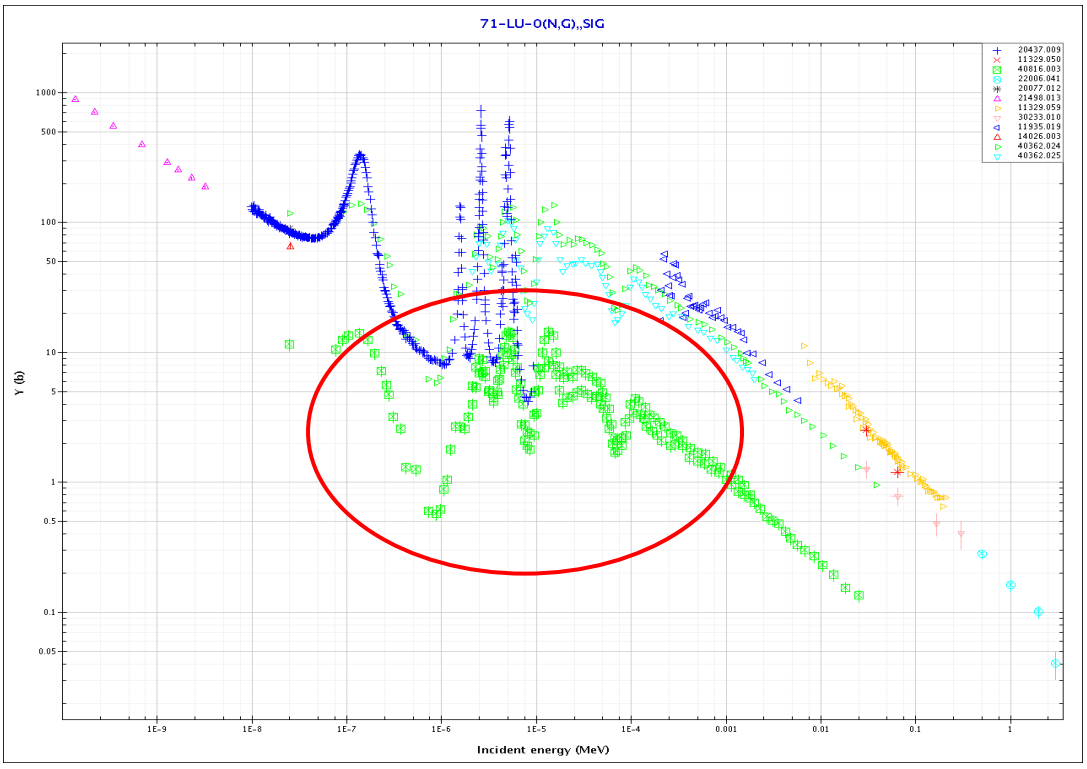


Figure 20 : Exemple d'une série aberrante avec zone de résonance

Exemple de mesures suspectes, mais non fausses compte tenu de l'incertitude

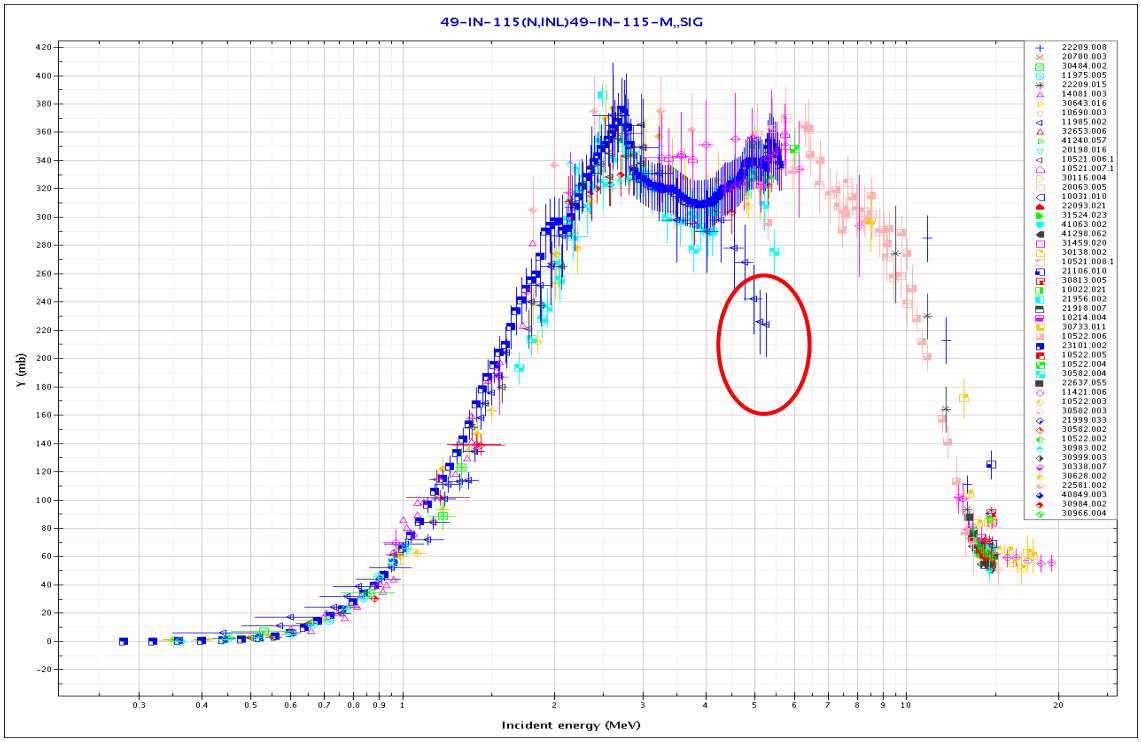


Figure 21 : exemple de mesures suspectes mais acceptables compte tenu des incertitudes

Exemple de séries de mesures non cohérentes

L'une des deux tendances est fausse, mais il n'est pas possible de distinguer la tendance anormale.

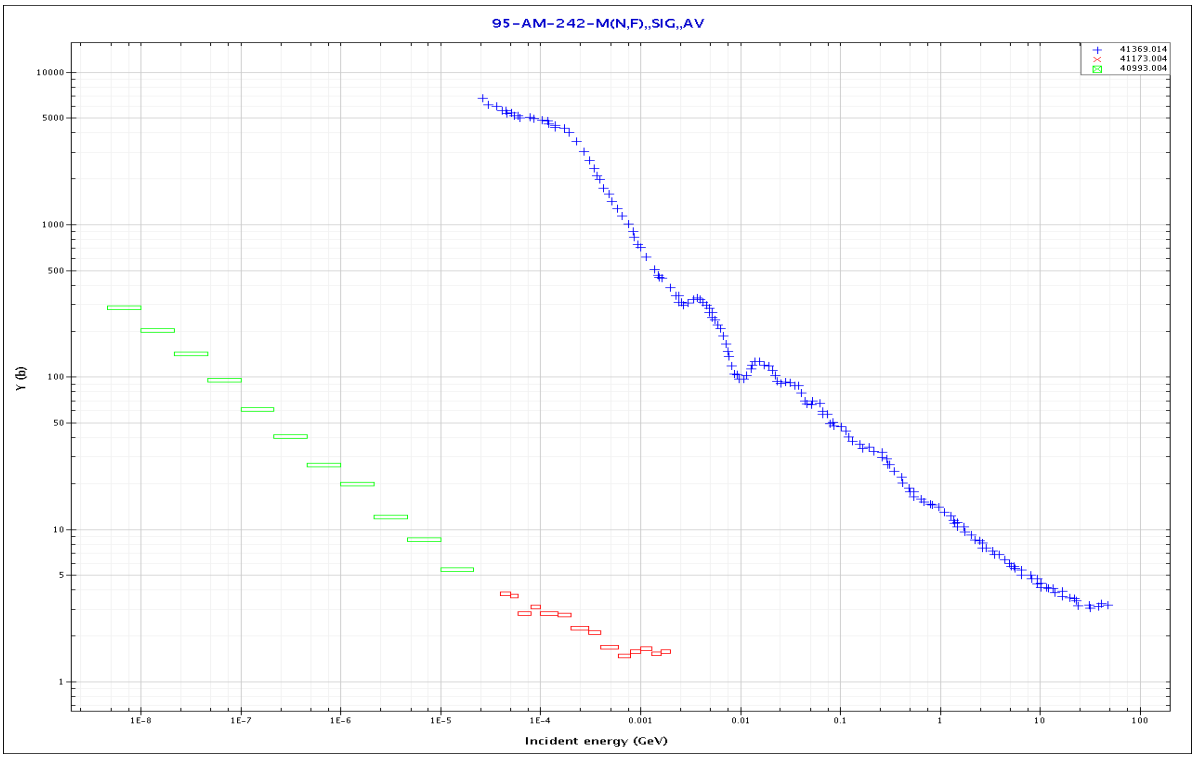


Figure 22 : Exemple de séries de mesures incohérentes

B. Particularités de la base de données

La méthode développée pour la détection de données aberrantes doit être la plus générale possible, l'objectif étant de traiter l'ensemble des données faisant partie de la base EXFOR (plus de 28 000 EHD). Pour cela, nous avons développé une méthode intuitive, permettant d'imiter les opérations manuelles réalisées par l'œil de l'expert.

Comme le montrent les quelques exemples précédents, les Ensembles Homogènes de Données représentent des fonctions continues très variées. Selon la configuration, les EHD peuvent :

- comporter des zones de résonances ;
- être complets (continuité du nuage de points), ou par morceaux ;
- présenter des incertitudes en abscisses et/ou en ordonnées ;
- présenter une résolution non optimale : le niveau de précision des instruments ainsi que la nature de la variable à mesurer peuvent affecter le niveau de résolution des observations réalisées ;
- contenir un nombre de mesures très faibles (10 points) ou très élevé (170 000 points) ;
- présenter une configuration en forme de nuage de points ;
- contenir une donnée, une série de données aberrantes, ou même les deux ;
- contenir des mesures observées pour une seule valeur d'énergie ;
- présenter des incertitudes en ordonnées tellement élevées qu'elles peuvent déformer le graphique ;
- contenir des valeurs négatives, nulles et en même temps des valeurs très élevées. La plupart des variables mesurées dans la base EXFOR (sections efficaces, distributions en énergie/angle, rendement de fission, etc.) présentent uniquement des valeurs positives ou nulles. Cependant, les données négatives ne sont pas forcement des erreurs. Dans la plupart des cas, il s'agit seulement de variations statistiques de la mesure autour d'une valeur centrale, positive ou nulle ;
- contenir une anomalie tellement éloignée du reste des mesures qu'elle déforme la représentation graphique ;
- contenir une anomalie, mais ne pas pouvoir identifier quelles sont les données aberrantes;
- contenir une anomalie contenant elle-même une nouvelle anomalie ;
- contenir une anomalie composée par une ou par plusieurs séries de données aberrantes ;

Les possibilités sont donc multiples. Au cours de ce projet, 113 cas test représentatifs des anomalies existant dans la base EXFOR ont été utilisées pour la construction et la validation de la méthode. Toutefois, il ne faut pas exclure la possibilité de trouver d'autres cas, sûrement rares, mais pour lesquels la méthode n'est pas totalement adaptée.

IV. Détermination de l'échelle optimale

La première étape consiste à déterminer l'échelle de représentation optimale du nuage de points. Nous présentons d'abord la méthode mise en place de façon synthétique, à travers un exemple, puis la théorie générale. Enfin, nous présentons également l'implémentation Matlab de l'ensemble des fonctions nécessaires pour cette première étape.

Le choix de l'échelle est essentiel pour visualiser les données de façon optimale. Par exemple, nous prenons le cas de la réaction *157 92-U-235(N,F),,SIG [E]*.

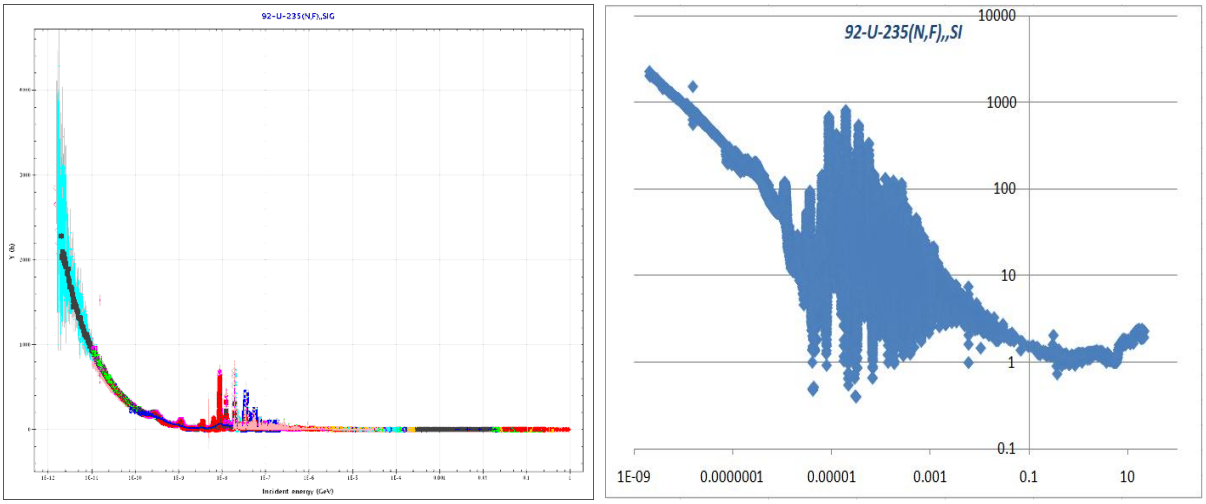


Figure 23 : Visualisation des données, réaction : 157 92-U-235(N,F),,SIG [E]

Les deux graphiques représentent le même jeu de données. Cependant, la représentation est très différente selon l'échelle choisie et en conséquence, les anomalies ne seront pas détectées de la même façon.

En effet, le choix de l'échelle n'affecte pas uniquement la visualisation de données, elle détermine aussi la discrétisation sur chacun des axes. En conséquence, les points ne sont pas également répartis lorsque l'on considère une échelle linéaire ou que l'on considère une échelle logarithmique.

A. Exemple simple

Dans cette section, nous illustrons avec un exemple la méthode développée afin de déterminer l'échelle optimale pour la représentation et le traitement postérieur des données.

On dispose d'un ensemble homogène de points M_k , dans un plan : soit x_k , y_k les coordonnées de M_k .

$$x_k = \{1, 10, 100, 1000, 10\,000, 100\,000\}$$

$$y_k = \{0, 10, 20, 30, 40, 50\}$$

Nous représentons l'ensemble de données M_k avec une échelle linéaire :

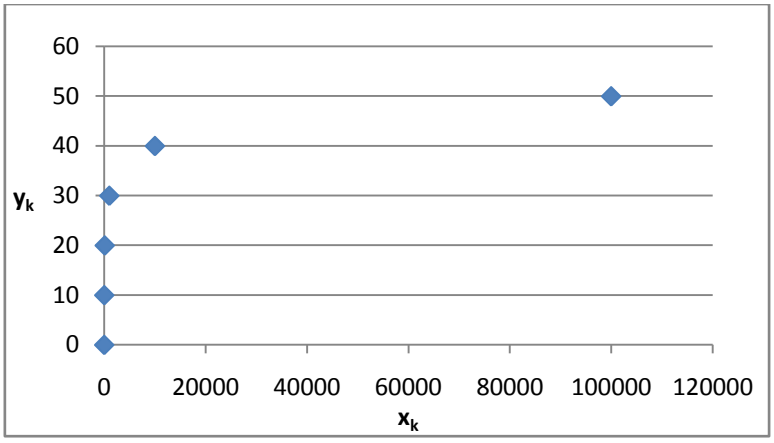


Figure 24 : Exemple- représentation de l'ensemble de points (x_k,y_k) avec une échelle linéaire

Posons maintenant la question de l'échelle de représentation : l'échelle linaire est-elle, dans cet exemple, l'échelle optimale ? Afin de répondre à cette question, nous allons étudier la configuration des points x_k et y_k.

On commence par normaliser et ranger par ordre croissant l'ensemble des points x_k et l'ensemble des points y_k: le maximum doit être égal à 1.

On détermine l'échelle optimale des axes x et y indépendamment. Commençons par nous intéresser à l'axe x. Après normalisation, le vecteur x_k vaut $x_k = \{0, 0.0001, 0.001, 0.01, 0.1, 1\}$. Testons d'abord une échelle linéaire. Pour ce faire, nous représentons les points (g_k, x_k) où $g_k = \frac{k}{N}$ (N représente le nombre de points). Dans notre cas, g_k vaut $g_k = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$.

Nous représentons l'ensemble de points x_k sur le plan.

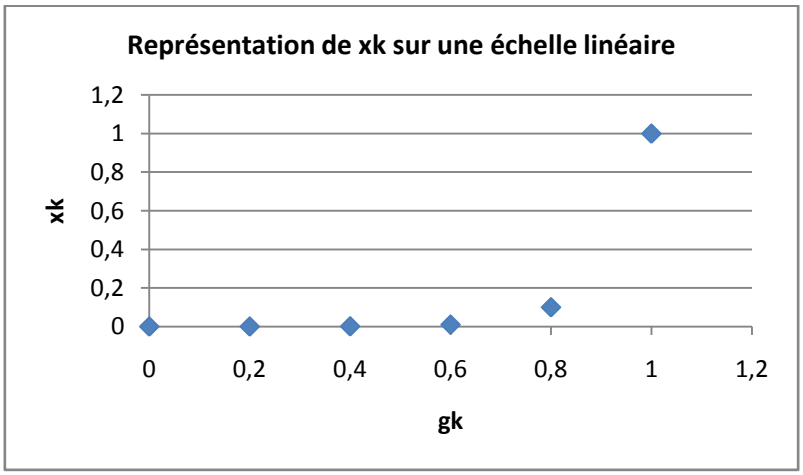


Figure 25 : Exemple - représentation de l'ensemble de points x_k avec une échelle linéaire

On observe que les valeurs de x_k ne sont pas régulièrement réparties dans le plan. Les points x_k croissent à pas constant par rapport à l'axe x mais irrégulièrement par rapport à l'axe y. L'échelle serait optimale si les points (g_k, x_k) étaient placés sur la première bissectrice du plan ; la configuration initiale des valeurs serait alors transformée de telle sorte que tous les

points seraient visualisés avec la même précision. Ce n'est pas le cas ici ; l'échelle linaire n'est pas idéale pour représenter les points x_k. Nous cherchons donc une autre échelle afin que la disposition des valeurs x_k soit parfaitement régulière sur le plan de coordonnées (g_k, x_k).

Nous testons ensuite une échelle logarithmique. Les points g_k valent alors $g_k = \{0, 0.0001, 0.001, 0.01, 0.1, 1\}$:

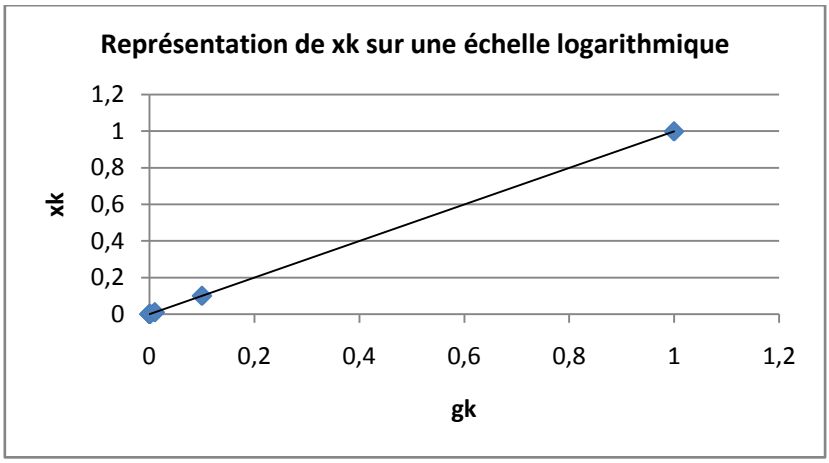


Figure 26 : Exemple - représentation de l'ensemble de points x_k pour une échelle logarithmique

Cette fois les points sont bien alignés sur la première bissectrice, ce qui permet une visualisation optimale. L'échelle logarithmique en base 10 est donc plus adaptée pour visualiser les données x_k .

Étudions à présent l'ensemble des points y_k. Après normalisation, le vecteur vaut $y_k = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. En le représentant avec une échelle linéaire ($g_k = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$), on obtient :

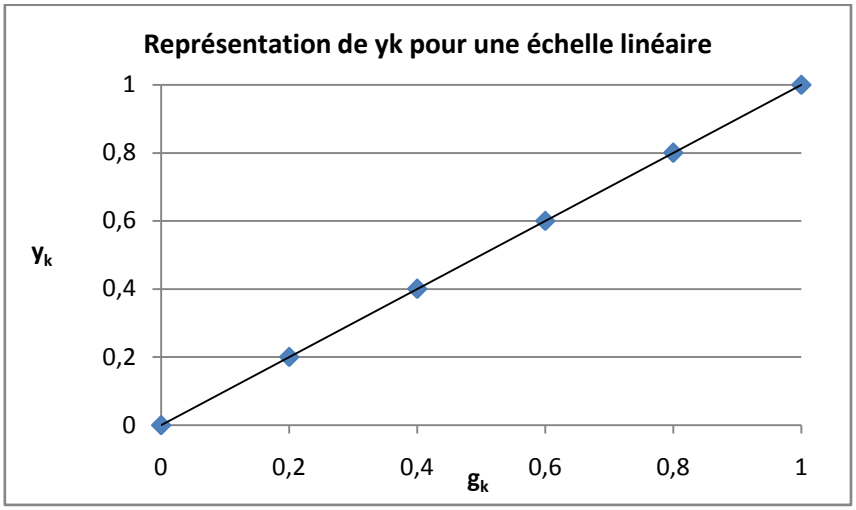


Figure 27 : Exemple - représentation de l'ensemble de points (x_{yk},y_{yk}) avec une échelle linéaire

Les points y_k sont bien rangés sur la première bissectrice, l'échelle linéaire est donc appropriée pour ce cas.

Enfin, nous pouvons représenter l'ensemble homogène de points M_k avec une échelle optimale : l'axe x est représentée avec une échelle logarithmique en base 10 et l'axe y avec une échelle linéaire.

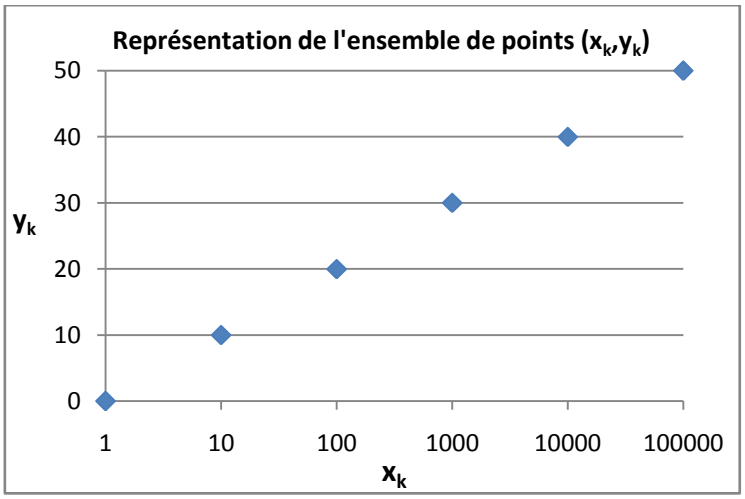


Figure 28 : Exemple - représentation de l'ensemble de points (x_k,y_k) avec une échelle optimale

B. Choix des types d'échelle considérés pour l'étude

En théorie, il existe de nombreuses échelles qu'il est possible d'utiliser. Par exemple :

- l'échelle linéaire $g_k = k$, pour les séries de données linéaires telles que $\{10, 20, 30, 40, 50, 60\}$;
- l'échelle logarithmique $g_k = \alpha^k$, pour les séries de données exponentielles telles que $\{10^1, 10^2, 10^3, 10^4, 10^5, 10^6\}$;
- l'échelle polynomiale $g_k = k^\alpha$, pour les séries de données polynomiales telles que $\{1^{10}, 2^{10}, 3^{10}, 4^{10}, 5^{10}, 6^{10}\}$;
- l'échelle exponentielle $g_k = log_\alpha(k)$, pour les séries de données logarithmiques telles que $\{\log(1), \log(2), \log(3), \log(4), \log(5), \log(6)\}$;

Après normalisation des données, ces échelles se présentent sous la forme :

- l'échelle linéaire $g_k = \frac{k}{N}$, pour les séries de données linéaires telles que $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$;
- l'échelle logarithmique $g_k = \frac{\alpha^k}{\alpha^N}$, pour les séries de données exponentielles telles que $\{0, 0.0001, 0.001, 0.01, 0.1, 1\}$;
- l'échelle polynomiale $g_k = \left(\frac{k}{N}\right)^\alpha$, pour les séries de données polynomiales telles que $\{0, 1.7^{-5}, 0.001, 0.017, 0.16, 1\}$;
- l'échelle exponentielle $g_k = \frac{log_\alpha(k)}{log_\alpha(N)}$, pour les séries de données logarithmiques telles que $\{0, 0.39, 0.61, 0.77, 0.89, 1\}$.

Représentons graphiquement ces séries de données :

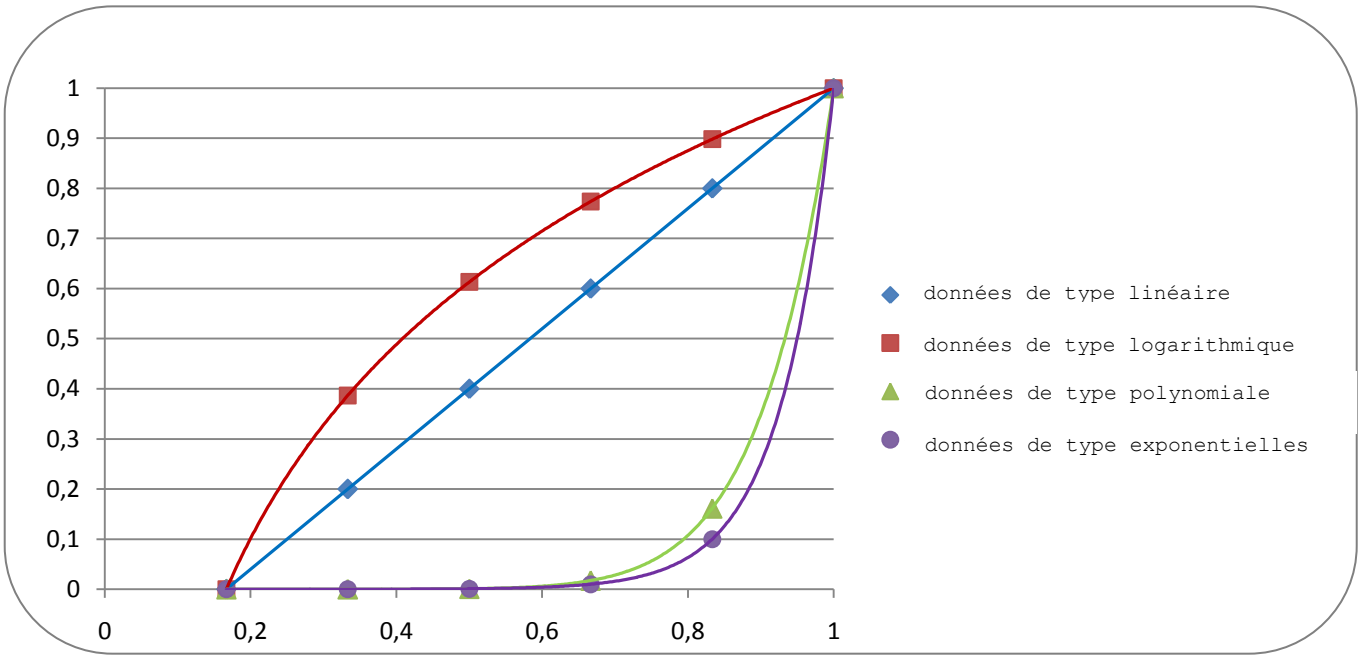


Figure 29 : Représentation graphique des séries de données linéaires, logarithmique, polynomiale et exponentielle

- L'échelle linéaire (en bleu) attribue la même importance à l'ensemble des valeurs. Cette échelle est donc adaptée lorsque les valeurs sont distribuées à des intervalles équidistants ;

- L'échelle polynomiale (en vert), similaire à l'échelle logarithmique (en violet), contracte les valeurs élevées et dilate les valeurs les plus faibles. De façon générale, l'effet de compression et de dilatation est moins fort lorsque l'échelle est polynomiale. Toutefois, après normalisation des données, ces différences ne sont pas significatives. Le choix de cette échelle est approprié lorsqu'il existe une concentration importante de valeurs faibles, qu'il y a peu de valeurs élevées et que celles-ci sont très dispersées. Les données provenant des réactions nucléaires présentent souvent ce type de configuration;

- L'échelle exponentielle (en rouge) contracte les valeurs les plus faibles et dilate les valeurs les plus élevées. Pour l'étude des réactions nucléaires, les échelles exponentielles ne sont pas adaptées car leur utilisation entrainerait une perte d'information provenant des données les plus faibles, en général très nombreuses.

Nous avons donc décidé de nous limiter à l'étude des échelles linéaires et polynomiales. L'équation suivante nous permet de prendre en compte simultanément les deux types d'échelles considérée : $g_k = \left(\frac{k}{N}\right)^{\alpha}$, qui prend une forme linéaire lorsque $\alpha = 1$ et une forme polynomiale lorsque $\alpha > 1$. Compte-tenu de la configuration des données provenant des réactions nucléaires, se limiter à ces deux options est justifié et permet de minimiser de façon considérable le temps de calcul du logiciel.

C. Méthode générale

Une étape préliminaire de mise en forme des données est nécessaire, avant de chercher l'échelle optimale pour la visualisation des mesures. Plus particulièrement, il faut trier les vecteurs x_k et y_k en ordre strictement croissant pour ensuite les normaliser. La formule utilisée est la suivante :

$$x_k = \frac{x_k - min}{max - min}$$

Dans la plupart des cas, les mesures sont positives sauf pour des variables spécifiques telles que les angles. Dans ce cas, la normalisation est la même, il faut seulement adapter cette équation lorsque les données sont alignées verticalement, c'est-à-dire lorsque l'ensemble des mesures partagent la même valeur de x :

$$x_k = \frac{x_k}{max}$$

Sans cet ajustement, les numérateur et dénominateur seraient nuls.

La méthode développée pour la visualisation des données consiste à tester différentes échelles possibles : linéaire $\left(g_k = \frac{k}{N}\right)$ et polynomiale $\left(g_k = \left(\frac{k}{N}\right)^{\alpha}\right)$ pour les axes x et y, pour différentes valeurs de α. Pour chaque axe, on place les points x_k (ou y_k) par rapport à l'échelle, et on détermine l'écart des points à la première bissectrice du plan.

Le fait de choisir la première bissectrice n'est pas trivial, nous cherchons une échelle g_k telle que la disposition des valeurs x_k (ou y_k) soit parfaitement régulière sur le plan des coordonnées (g_k, x_k) (ou sur le plan (g_k, y_k)). L'échelle permet ainsi de transformer la configuration initiale des valeurs de telle sorte qu'elle contracte ou dilate la distance entre les points afin de visualiser l'ensemble des valeurs avec la même précision.

L'échelle optimale est celle minimisant l'écart des données à la première bissectrice. Ceci se traduit mathématiquement par la minimisation de la quantité suivante :

$$C(\alpha) = \sum_k (x_k - g_k)^2$$

Pour trouver le α optimal, le plus simple est de calculer la valeur de la fonction $C(\alpha)$ lorsque $\alpha = 1, 2, 3, 4, 5, \ldots, 1000$. Il est également possible de tester des valeurs décimales pour la recherche d'alpha $\alpha = 0.1, 0.2, 0.3, \ldots, 1.1, 1.2, 1.3, \ldots, 1000$. Toutefois, le temps de calcul augmente considérablement et l'échelle ne s'améliore pas de façon perceptible. En effet, nous avons mené une analyse de sensibilité au regard de la discrétisation de α : nous avons testé un échantillon de 25 cas. Pour ces 25 cas, l'utilisation des valeurs décimales n'apporte aucune amélioration.

Dans l'avenir, il est possible de simplifier le temps de calcul nécessaire pour la recherche d'alpha optimal. Dans ce but, l'AEN a d'ailleurs suggéré l'utilisation d'un algorithme de type « Golden section search ». Celui-ci permet de trouver les extrema (minimum or maximum) d'une fonction lorsqu'elle est unimodale. Toutefois, cette méthode ne doit être intégrée que suite à une analyse de la base de données, car elle n'est applicable que dans les cas remplis-

sant cette condition. Autrement dit, il est nécessaire de vérifier que les fonctions $C(\alpha)$ sont bien unimodales quelle que soit la réaction et les variables mesurées (par exemple, $C(\alpha)$ peut être unimodale pour les réactions CS mais ne pas l'être pour les DA) et cela pour l'ensemble des cas (y compris les cas particuliers).

D. Implémentation

Cette méthode a été implémentée sous Matlab. Pour choisir l'échelle idéale en utilisant cette méthode, le logiciel procède de la façon suivante :

OBJECTIF	FONCTION
Les valeurs non numériques de x_k et y_k sont remplacées par zéro.	remplacer_NaN_zero
Les séries de valeurs x_k et y_k sont triées en ordre croissant et normalisées.	normalisation_echelle
Pour chaque axe, on recherche d'alpha optimal afin de choisir l'échelle linéaire ou polynomiale la plus adaptée au jeu des données.	recherche_alpha
Appels des fonctions « remplacer_zeros », « normalisation_échelle » et « recherche-alpha ». Représentation graphique des valeurs x_k et y_k par rapport à leurs échelles respectives.	detection_anomalies_main

Nous donnons ci-dessous l'implémentation Matlab de l'ensemble de ses fonctions :

1. La fonction `remplacer_NaN_zero` permet de remplacer les valeurs non numériques d'un vecteur par zéro. En théorie, il n'existe plus de données non numériques dans la base de données. Il est néanmoins préférable d'ajouter cette condition dans le cas d'un oubli. Cela évitera l'arrêt du logiciel lors de l'exécution du code.

En entrée la fonction reçoit :

- x : le vecteur initial ;

En sortie :

- x : le nouveau vecteur.

```
function x=remplacer_NaN_zero(x)

x(isnan(x)==1)=0;
x=x;
```

2. La fonction `normalisation_echelle` permet de normaliser les valeurs d'un vecteur.

En entrée la fonction reçoit :

- x : le vecteur à normaliser ;

En sortie :

- x : le vecteur normalisé.

```
function x_normal=normalisation_echelle(x)

x=sort(x);
m=min(x);
M=max(x);

if m<M
        x=x-m;
        x=x/(M-m);
elseif m==M
        x=x/M;
else
    display ('erreur dans la base de données');
end

x_normal=x;
```

3. La fonction `recherche_alpha` permet de trouver la valeur optimale de α afin de choisir la bonne échelle.

En entrée la fonction reçoit :

- N : le nombre de points
- x : la valeur de chaque point (vecteur)
- alpha : les valeurs d'alpha
- Nb : le nombre de valeurs d'alpha à tester

En sortie : α optimal

```
function alpha_optimal=recherche_alpha(N,x,Nb,alpha)

new_result=10^1000000000000;
alpha_optimal=0;
indice=0;

% chercher alpha optimal pour une échelle linéaire ou polynomiale.

for i =1:Nb

    result(i)= [(x(1)-(1/N).^(alpha(i))).^2];

    for k=2:N
        result(i)=result(i)+[(x(k)-(k/N).^(alpha(i))).^2];
    end
```

```
% identifier le résultat minimal et l'indice d'alpha

    if result(i)<new_result
        indice=i;
    else
        indice=indice;
    end

    new_result=min(result);

end

% Choisir entre une échelle linéaire ou polynomiale.

figure = plot (alpha,result);

if alpha(indice)==1
    display ('léchelle de visualisation est linéaire')
else
    display ('léchelle de visualisation est polynomiale')
end

alpha_optimal=alpha(indice);
```

4. La deuxième étape de la fonction `detection_anomalies_main` permet de choisir l'échelle idéale et de la visualiser.

```
%------------ETAPE 2--------------------------------
%---------Choix de l'échelle-----------------------

% x_k vecteur de valeurs comprises entre 0 et 1
x_normal=normalisation_echelle(x);

% y_k vecteur de valeurs comprises entre 0 et 1
y_normal=normalisation_echelle(y);

% Trouver alpha optimal pour choisir l'échelle x
alpha_optimal_x=recherche_alpha(N,x_normal,Nb,alpha);

% Trouver alpha optimal pour choisir l'échelle y
 alpha_optimal_y=recherche_alpha(N,y_normal,Nb,alpha);

% Visualisation de l'échelle optimale par rapport l'axe x et par rapport l'axe y
plot((k/N).^alpha_optimal_x,x_normal,'k');
title(filename);
hold on;
plot((k/N).^alpha_optimal_y,y_normal,'r');

x=x.^(1./alpha_optimal_x).*N;
y=y.^(1./alpha_optimal_y).*N;
```

V. Méthode de détection d'anomalies

L'approche s'inspire de la méthode employée jusqu'à présent par l'AEN : la détection des anomalies se fait par la visualisation des résultats des différentes séries sur un graphique (2D). La méthode mise en place permet d'automatiser cette détection, sans faire appel à l'œil des experts.

A. Méthode de détection

1. Théorie générale

Une fois l'échelle optimale déterminée, on discrétise le plan (x, y) en rectangles réguliers.

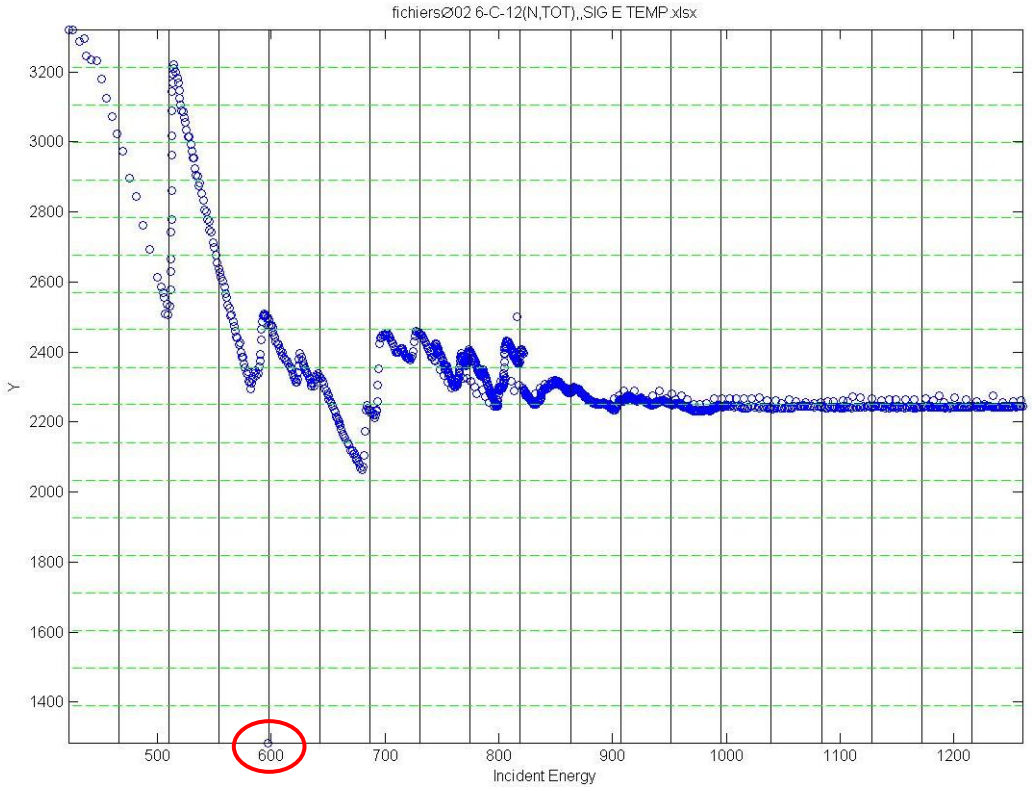


Figure 30 : Discrétisation en rectangles réguliers

Sur chaque tranche verticale $[x_i, x_{i+1}[$, on construit l'histogramme des points de mesure : on compte le nombre de points contenus dans chaque intervalle$[y_j, y_{j+1}[$. On dispose ainsi de la loi de probabilité des mesures dans la tranche $[x_i, x_{i+1}[$. L'étude de ces lois de probabilité nous permet de détecter les anomalies.

On considère que l'EHD présente une anomalie lorsqu'au moins une loi de probabilité présente une (ou plusieurs) discontinuité(s), c'est-à-dire lorsque la loi est composée de deux ou plusieurs ensembles distincts, séparés par un certain nombre d'intervalles vides. Plus cette « distance » (nombre d'intervalles vides séparant les ensembles de données) est élevée, plus la probabilité de détecter une vraie anomalie est importante.

Par exemple, sur le graphique suivant, qui représente la quatrième tranche verticale de la figure précédente, on remarque une anomalie car il existe un point séparé du reste de la distribution par un nombre important d'intervalles vides (8).

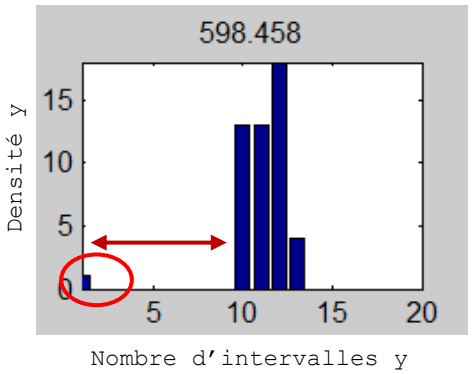


Figure 31 : Loi de probabilité des mesures sur la quatrième tranche verticale

La première figure ci-dessous représente la troisième tranche verticale, juste avant l'anomalie et la deuxième représente la dernière tranche verticale. Sur ces deux graphes, on constate que la distribution est continue. En conséquence, on ne détecte pas d'anomalies.

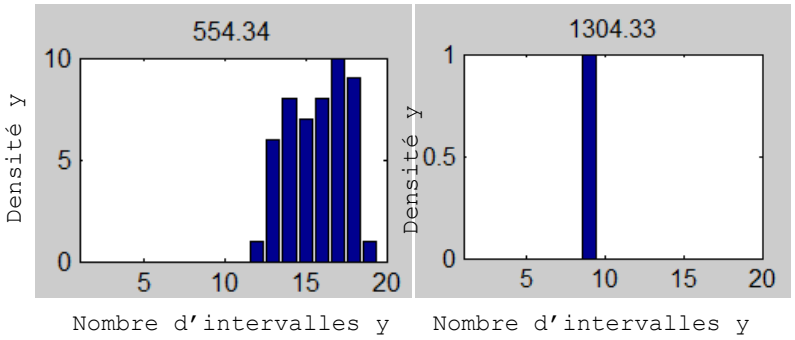


Figure 32 : Lois de probabilité des mesures sur les troisième et cinquième tranches verticales

On remarque que le choix de la discrétisation des axes est un point important : une discrétisation trop fine engendrerait un nombre de faux positif trop important, alors qu'une discrétisation trop grossière peut « laisser passer » des anomalies. Un compromis doit donc être trouvé. Dans ce but, nous avons testé une discrétisation à 19 intervalles (20 bornes) pour les axes d'ordonnées et d'abscisses. Celle-ci fonctionne correctement dans la plupart des cas (92 % des cas testés) sauf pour les « nuages des points », souvent composés par un faible nombre de mesures.

Pour ces cas particuliers, on peut chercher à optimiser la discrétisation des axes en fonction du nombre de points. Toutefois, il ne faut pas oublier qu'il s'agit d'une méthode basée sur la détection réalisée par l'œil humain : si manuellement ce n'est pas possible, le logiciel n'y parviendra pas non plus.

Cette méthode est adaptée aux particularités propres aux réactions nucléaires ; en particulier, elle permet une identification robuste des anomalies, même si on ne dispose pas de la totalité des données ou s'il existe des zones de résonance d'amplitude très importantes en comparaison au reste de la tendance. De plus, la présence de ces zones de résonance impose l'étude de

l'histogramme complet : sur les autres parties des fonctions, l'étude des indicateurs de dispersion classiques (variance, écart-type) permettrait de détecter la présence d'anomalies ; elle n'est pas pertinente dans les zones de résonance, où il est normal que la dispersion soit importante.

2. Indicateurs de la présence d'une anomalie et du degré de suspicion

La détection d'anomalies est réalisée à l'aide de plusieurs indicateurs. Le premier est la distance (nombre d'intervalles) séparant les points constituant l'anomalie du reste des points dans l'histogramme. Cet indicateur est pertinent : plus sa valeur est élevée, plus la probabilité que le ou les points repérés soient effectivement une anomalie est élevée.

Toutefois, cet indicateur n'est pas suffisant. En effet, dans la base de données EXFOR, il est courant d'observer des séries des données décalées de seulement quelques intervalles vides (2 ou 3), mais dont la tendance est clairement différente de celle de l'ensemble des points. Il s'agit donc bien d'anomalies, dont l'importance ne sera pas quantifiée de manière satisfaisante par l'indicateur de la distance. Afin d'identifier ces cas, nous avons ajouté un nouvel indicateur qui indique le nombre de fois qu'une même série est identifiée comme suspecte pour la même réaction.

Par exemple, le graphique suivant représente une EHD ou réaction composée de plusieurs séries de données. A gauche, chaque série est indiquée avec une couleur différente.

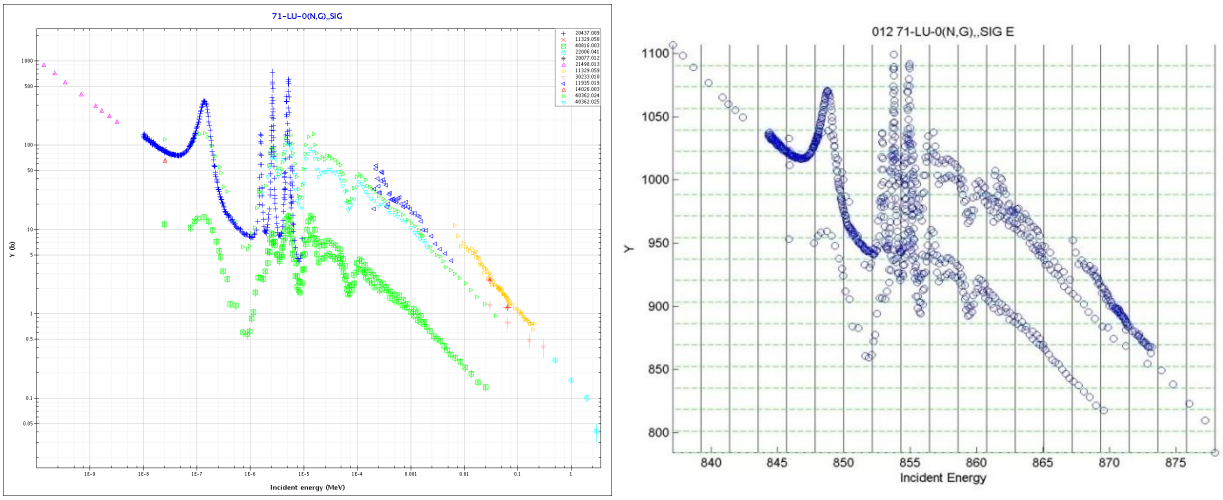


Figure 33 : Représentation d'une EHD composée de plusieurs séries de données

A droite, cette même réaction est représentée après recherche de l'échelle optimale et discrétisation des axes. Nous pouvons observer que le nombre des cases vides n'est pas supérieur à deux, quelle que soit la tranche verticale considérée. Toutefois, l'existence d'une anomalie est incontestable : la même série est identifiée comme aberrante 8 fois (Figure 34).

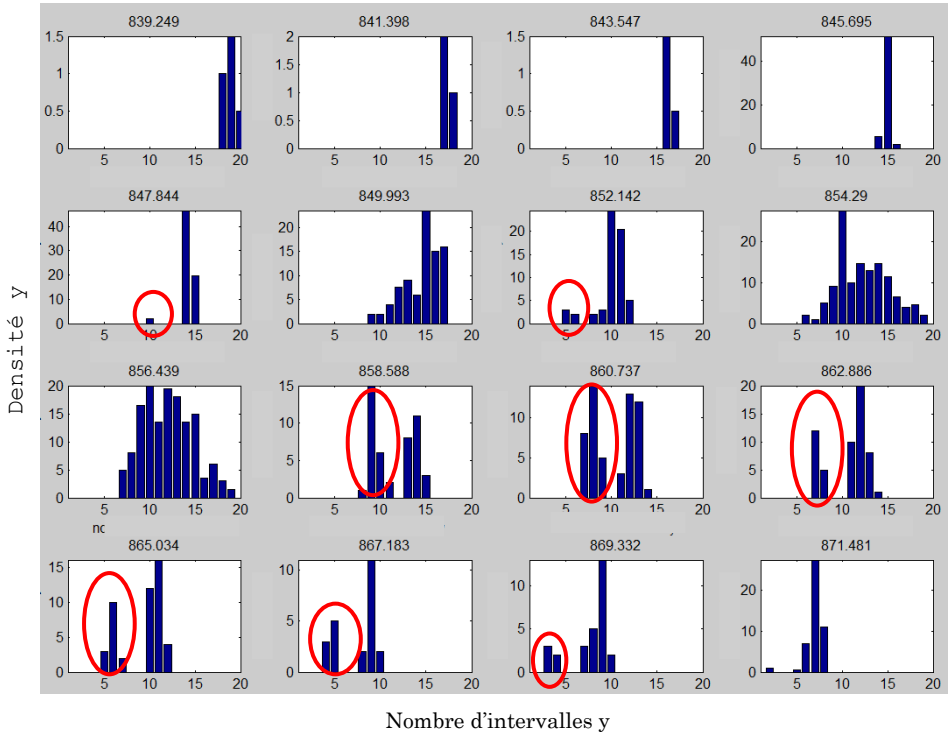


Figure 34 : Lois de probabilité des mesures sur les seize premières tranches verticales

Les résultats finaux doivent donc être triés en fonction de la distance *nombre_carrés* mais aussi en fonction de l'indicateur *nb_serie_aberrante*. Par exemple, si $nombre\_carrés = 2$ mais $nb\_serie\_aberrante = 8$, il y a sûrement une anomalie. L'utilisateur pourra alors traiter les cas les plus urgents (c'est-à-dire ceux dont la probabilité d'être aberrants est la plus élevée) en priorité, laissant les possibles faux positifs à la fin de la liste (distance et nombre de fois où la série est identifiée comme aberrante faibles).

Cette méthode présente l'avantage de permettre de détecter des anomalies même dans les zones de résonance, à condition bien sûr que la résonance soit suffisamment représentée (plusieurs points de mesure), ce qui est en général le cas dans les EHD considérés.

Enfin, il est important de signaler que la nature de ces deux indicateurs est différente. L'indicateur de distance représente une valeur quantitative, c'est-à-dire que les données identifiées comme suspectes ont plus de chance d'être aberrantes lorsque cet indicateur augmente. Par exemple, si $nombre\_carrés = 3$, on ne pourra pas conclure quant à l'existence d'une anomalie. Si cet indicateur vaut 5, on pourra affirmer qu'il existe une anomalie mais qu'elle n'est pas très grave. Par contre, si l'indicateur est égal à 15, on saura que l'anomalie est très grave et qu'il faut la traiter en priorité. Ces exemples sont donnés sur la base d'une discrétisation des deux axes à 19 intervalles.

Par contre, le deuxième indicateur, qui compte le nombre de fois qu'une série est identifiée comme suspecte, a une nature plutôt qualitative. Autrement dit, il existe un effet de seuil ; il n'y a donc pas de différence si $nb\_serie\_aberrante = 5$ ou si $nb\_serie\_aberrante = 8$. L'information est la même car la probabilité que la série soit aberrante dans le première cas est déjà très élevée. Ce seuil doit être établi suite à une analyse de sensibilité. Le traitement des 113 cas types avec une discrétisation à 19 intervalles en abscisses et en ordonnées, permet

de suggérer que ce seuil est situé autour de 3 ou 4. Ceci reste toutefois à affiner par l'étude de cas supplémentaires.

Comme suggéré par l'AEN, il est possible de créer un indicateur mixte à partir de ces deux indicateurs. Toutefois, il s'agit d'une tâche complexe car leur nature est différente. De même, il faut réfléchir au poids qu'il faut attribuer à chacun dans la construction du nouvel indicateur. La méthode à développer doit donc être bien justifiée et une analyse de sensibilité s'impose.

3. Exemples de détection d'anomalies

Sur la figure ci-dessous, on repère une anomalie sur la sixième tranche verticale : le point est isolé du reste des autres mesures. Ceci est confirmé par l'analyse de la distribution (voir figure à droite).

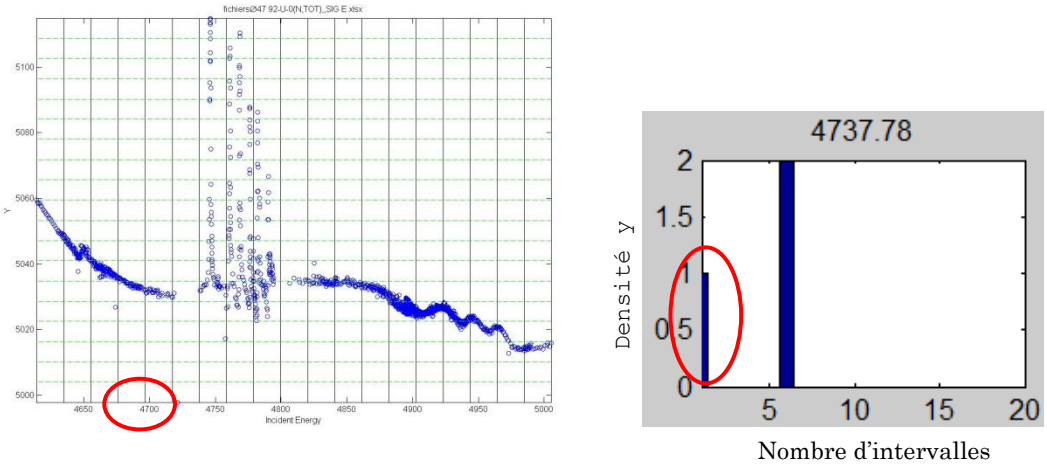


Figure 35 : Exemple de graphique présentant une anomalie et histogramme de la sixième tranche verticale

Sur cette autre figure, on repère plusieurs données aberrantes, qui font partie de la même série : celles-ci semblent cohérentes entre elles, mais très éloignées du reste des autres points.

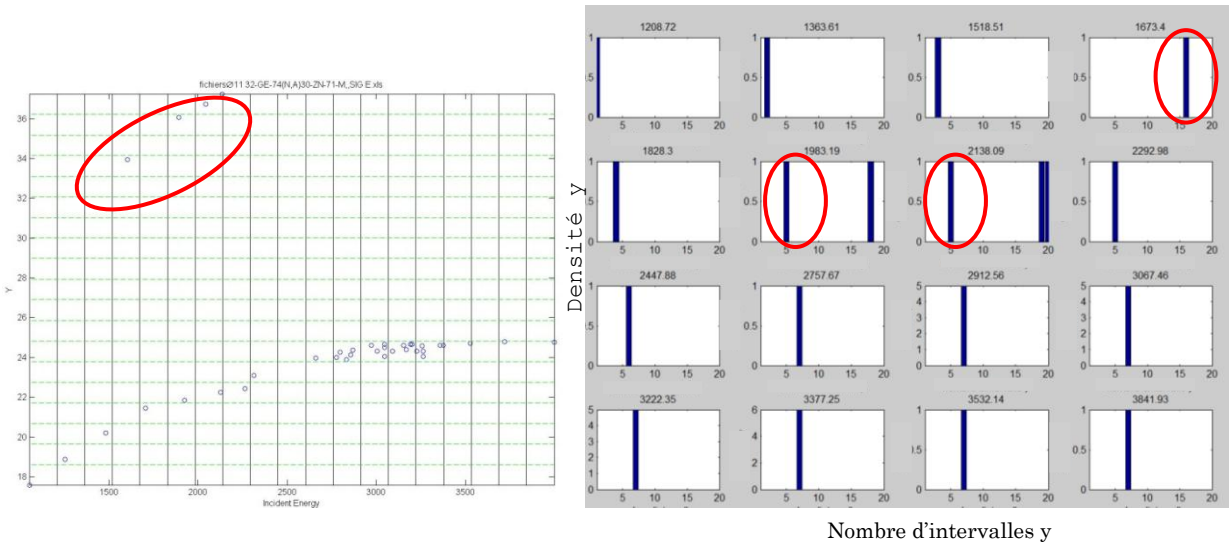


Figure 36 : Exemple de graphique présentant une série suspecte et histogrammes verticaux associés

B. Prise en compte des incertitudes et de la résolution des données

Cette première approche permet de détecter les anomalies lorsque les mesures sont connues avec certitude et lorsque la résolution est optimale : les valeurs sont représentées par un seul point sur le graphique. Or la plupart des mesures collectées par l'AEN sont entachées d'une incertitude et, en outre, la résolution n'est pas toujours optimale (les valeurs x_{min} et x_{max} peuvent être différentes). La méthode a donc été approfondie afin de prendre en compte les incertitudes et la résolution propres aux mesures.

Les sources d'incertitude affectant une donnée sont potentiellement nombreuses :

- incertitude sur la masse de l'échantillon ;
- variation du nombre et/ou de l'énergie des particules incidentes sur l'échantillon au cours de la mesure ;
- incertitude statistique liées aux taux de comptage ;
- incertitude des corrections appliquées à la mesure brute, e.g. temps mort de l'électronique, absorption des particules dans l'échantillon, efficacité des détecteurs…
- normalisation de la mesure.

En général, ces contributions sont supposés être indépendantes ce qui permet d'utiliser le théorème de la limite centrale pour justifier que l'incertitude totale peut être représentée par une loi normale. Cependant, cette information n'est pas nécessaire pour la détection des anomalies ; seul le support de la loi importe. Pour des raisons de simplification des algorithmes et diminution des temps de calcul, nous choisissons d'utiliser des lois uniformes. Cela signifie que l'on considère la mesure peut être égale à n'importe quelle valeur entre les bornes de l'intervalle, avec une même probabilité pour toutes les valeurs. La modélisation est plus grossière mais les résultats de la détection d'anomalies sont identiques.

Sans incertitude, la méthode consiste à construire les histogrammes des points de mesure, sur chaque tranche verticale : lorsqu'un point est contenu dans une case de la discrétisation en y, on ajoute 1 à l'histogramme de cette case. Le principe de détection reste le même lorsque l'on prend en compte les incertitudes : si la loi uniforme représentant l'incertitude de la valeur croise une case de discrétisation en y, alors on ajoute $\frac{1}{N}$ à l'histogramme de cette case, N étant le nombre de cases sur lesquelles s'étend l'intervalle d'incertitude.

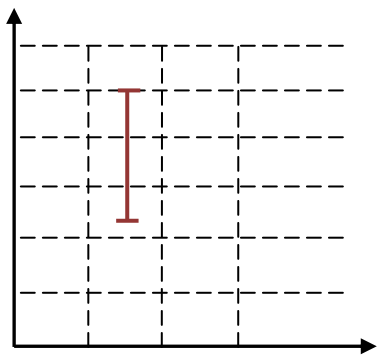


Figure 37 : Exemple de mesure avec incertitude

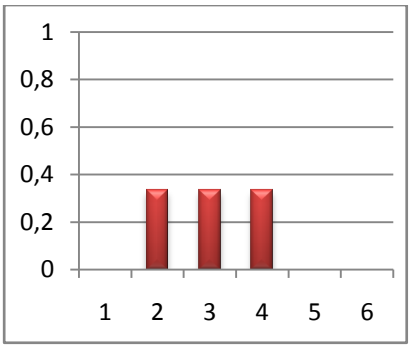


Figure 38 : Loi de probabilité des mesures sur la deuxième tranche des abscisses

Si l'incertitude porte sur x, le principe est le même : la mesure sera prise en compte dans la loi de probabilité des tranches verticales que croise la mesure incertaine.

La résolution des mesures dépend du niveau de précision des instruments et de la variable à mesurer. Lorsque la résolution est optimale, la mesure est représentée par un seul point sur la graphique ($x_{min} = x_{max}$ and $y_{min} = y_{max}$). Dans le cas contraire, la mesure est représentée par un trait vertical ($y_{min} \neq y_{max}$), horizontal ($x_{min} \neq x_{max}$) ou bien par un rectangle ($x_{min} \neq x_{max}$ and $y_{min} \neq y_{max}$).

La méthode développée intègre également cette notion. Pour identifier les intervalles de discrétisation occupés par une mesure, on procède de la façon suivante :

$$new\_x_{min} = x_{min} - incertitude_{Xmin}$$
$$new\_x_{max} = x_{max} + incertitude_{Xmax}$$

En conclusion, pour le même degré d'incertitude, le nombre de cases occupées par une mesure est supérieur lorsque la résolution est faible. L'AEN nous a indiqué que les incertitudes en y sont indiquées « à 1 σ » dans la base de données ; le degré de confiance dans la mesure est donc de 68 %. En effet, pour chaque tranche verticale, les mesures sont supposées suivre une distribution normale. Afin d'augmenter le niveau de confiance de ces données à 95 %, nous exprimons l'incertitude sur l'axe y « à 2 σ ». Par contre, l'incertitude sur l'axe x n'a pas à être modifiée car elle représente le niveau de précision des instruments de mesure. Il n'y donc pas de notion de distribution normale des données.

Dans le cas où la résolution de la mesure en y n'est pas optimale ($y_{min} \neq y_{max}$) et qu'il existe une incertitude en y, l'AEN recommande de ne pas prendre en compte l'incertitude. Ce cas est cependant très rare dans la base de données.

C. Implémentation

La méthode de détection des anomalies a été implémentée sous Matlab.

Cette section explique les différentes fonctions utilisées afin d'incorporer dans la méthode l'incertitude et la résolution des mesures lorsqu'ils portent sur l'axe y. Ensuite, nous expliquons la construction de la loi de probabilité des mesures sur chaque tranche verticale et la détection des discontinuités dans la loi de probabilité permettant d'identifier les données suspectes. Enfin, nous donnons aussi les fonctions utilisées pour l'automatisation des calculs, l'importation et l'exportation des données.

1. Prise en compte de l'incertitude et de la résolution des données

La première étape du code sert à importer les données sur le logiciel pour ensuite créer les variables nécessaires. La deuxième phase, expliquée précédemment, permet de chercher l'échelle optimale pour visualiser les mesures. La troisième étape, expliquée par la suite, permet la prise en compte des incertitudes et de la résolution des données. Celle-ci est nécessaire afin de créer de nouvelles variables pour le calcul d'histogrammes (quatrième étape).

Au cours de cette phase, nous calculons trois variables :

- `intervalle_y` : fait référence à la valeur moyenne de la mesure pour l'axe `y` en sachant également l'intervalle `x` de discrétisation auquel la mesure appartient ;
- `interv_incertitude_ymin` : fait référence à la valeur minimale de `y` que la mesure peut pendre du fait de l'incertitude. De même, on connait l'intervalle `x` de discrétisation auquel la mesure appartient ;
- `interv_incertitude_ymax` : fait référence à la valeur maximale de `y` que la mesure peut atteindre du fait de l'incertitude. De même, on connait l'intervalle `x` de discrétisation auquel la mesure appartient.

Toutefois, cette information n'est pas suffisante, nous avons besoin d'une nouvelle variable permettant de regrouper l'ensemble des valeurs possibles que la mesure peut prendre en ordonnées. Cette variable, nommée `intervalle_newy`, constitue le principal résultat de sortie de cette étape. Si la mesure occupe 5 intervalles, la nouvelle variable `intervalle_newy` doit contenir 5 valeurs possibles de `y` pour la mesure. De même, il faut une autre variable `compteur_newy` qui renseigne sur la probabilité d'appartenance à chacun de ces intervalles. Puisque nous considérons des lois uniformes, la probabilité d'appartenance à un intervalle sera égale à l'inverse du nombre de cases de discrétisation occupées par la mesure.

Au cours de cette étape, nous avons développé les fonctions suivantes :

OBJECTIF	FONCTION
Identification des intervalles de discrétisation occupés par une mesure. Création d'une nouvelle variable qui tient compte de l'ensemble des valeurs possibles de y que la mesure peut prendre Appels des fonctions.	detection_anomalies_main (étape 3)
Discrétisation de l'axe x	calcul_bornes
Identification des valeurs de y qui correspondent à chaque tranche Δx	matrice_distribution
Identification si la mesure s'étale sur plusieurs intervalles de discrétisation. Si oui, quantification du nombre de cases de séparation et de la borne inférieure de la mesure	distrib_incertitude

La fonction `detection_anomalies_main` représente le cœur du code. Cette fonction est organisée en plusieurs étapes, détaillées ci-après.

A. *Mise en forme préliminaire des données : incertitudes et échelle optimale*

Le tableau d'entrée présente pour chaque mesure les renseignements suivants :

x min	x max	incertitude x min	incertitude x max	y min	y max	incertitude y min	incertitude y max

Compte tenu de l'incertitude en *y* (1) et du niveau de résolution des données (2), une mesure peut être étalée sur plusieurs intervalles. En pratique, nous identifions ces phénomènes si :

- `incertitude_ymin` ≠ `0` ou `incertitude_ymax` ≠ `0` (1);
- `ymin` ≠ `ymax` (2);

Afin d'augmenter le niveau de confiance de ces données à 95 %, les valeurs correspondant à l'incertitude sur l'axe *y* sont multipliés par deux. Toutefois, si la résolution n'est pas optimale, on ne tient pas compte de l'incertitude sur l'axe *y*.

Nous avons ainsi créé deux variables `incertitude_ymin` et `incertitude_ymax` qui présentent la valeur minimale et maximale de `y`. Ensuite, ces variables sont transformées en utilisant `alpha_optimal`, calculé dans l'étape 2.

```
%----------------------------------ETAPE 3-------------------------------------
%----Prise en compte de l'incertitude et de la résolution des données--------

% 1. Mise en forme préliminaire des données

for i=1:N
    if ymin(i)==ymax(i) % si la résolution est optimale
        incertitude_ymin(i)=ymin(i)-2*incertitude_ymin(i);
        incertitude_ymax(i)=ymax(i)+2*incertitude_ymax(i);
    else % si la résolution n'est pas optimale
        incertitude_ymin(i)=ymin(i);
        incertitude_ymax(i)=ymax(i);
    end
end

% remplacer les valeurs négatives par zéro
incertitude_ymin=remplacer_negatif_zero(incertitude_ymin);
incertitude_ymax=remplacer_negatif_zero(incertitude_ymax);

%transformer les valeurs (prise en compte d'alpha)
incertitude_ymin=incertitude_ymin.^(1./alpha_optimal_y).*N;
incertitude_ymax=incertitude_ymax.^(1./alpha_optimal_y).*N;
incertitude_ymin=tranfor_valeurs(incertitude_ymin);
incertitude_ymax=tranfor_valeurs(incertitude_ymax);
incertitude_ymin=remplacer_incert_min(incertitude_ymin,y,minimum);
incertitude_ymax=remplacer_incert_min(incertitude_ymax,y,minimum);

ymin=ymin.^(1./alpha_optimal_y).*N;
ymax=ymax.^(1./alpha_optimal_y).*N;
ymin=tranfor_valeurs(ymin);
ymax=tranfor_valeurs(ymax);
```

Cette phase de mise en forme des données fait appel à la fonction `remplacer_negatif_zero`, qui permet de remplacer les valeurs négatives par zéro. Cette fonction limite ainsi l'étendue de l'incertitude à des valeurs strictement positives ou égales à zéro.

En entrée la fonction reçoit :

- `x` : le vecteur initial ;

En sortie :

- `x` : le nouveau vecteur.

Voici le code :

```
function x=remplacer_negatif_zero(x)

for i=1:size(x,1)
    if x(i)<0
        x(i)=0;
    end
end
```

La fonction `tranfor_valeur` permet de remplacer les valeurs nulles par des valeurs très faibles. L'objectif d'une telle fonction est de permettre à Matlab de différencier les données vraiment nulles des données manquantes.

En entrée la fonction reçoit :

- `x` : le vecteur initial ;

En sortie :

- `x` : le nouveau vecteur.

Voici le code :

```
function x=tranfor_valeurs(x)

for i=1:size(x,1)
    if x(i)==0
        x(i)=0.0000001;
    end
end
```

Enfin, la fonction `remplacer_incert_min` permet de limiter l'étendu de l'incertitude des mesures lorsque celle-ci nuit à la visibilité du graphique. Pour cela, cette fonction remplace la valeur de l'incertitude par la valeur minimale des mesures observées.

La fonction reçoit en entrée :

- `y_incertitude`: la valeur du vecteur avec l'incertitude ;
- `y` : la valeur du vecteur sans incertitude ;
- `minimum`: la valeur minimale qui peut prendre xmoy (sans incertitude)

En sortie :

- `y_incertitude`: le nouveau vecteur

Voici le code :

```
function y_incertitude=remplacer_incert_min(y_incertitude,y,minimum)

for i=1:size(y,1)
    if (y_incertitude(i)<minimum & y_incertitude(i)~=y(i))
        y_incertitude(i)=minimum;
    end
end
```

B. *Identifier les valeurs de y qui correspondent à chaque tranche verticale*

Au cours de cette deuxième étape de calcul, nous allons :

- discrétiser l'axe des abscisses x. Pour cela, on fait appel à la fonction `calcul_bornes` ;
- identifier les valeurs de `y` qui correspondent à chaque tranche verticale. Il faut réaliser cette opération pour les trois variables `incertitude_ymin, incertitude_ymax` et `y`. Pour cela, on fait appel à la fonction `matrice_distribution` ;

La deuxième étape de la fonction `detection_anomalies_main` se présente sous cette forme :

```
% 2. Identifier les valeurs de y qui correspondent à chaque tranche Δx

bornesx=calcul_bornes(x,pasx);

interv_incertitude_ymin =matrice_distribution(x,incertitude_ymin,N,pasx,bornesx);% mini-
mum

interv_incertitude_ymax=matrice_distribution (x,incertitude_ymax,N,pasx,bornesx); % maxi-
mum

intervalle_y=matrice_distribution (x,y,N,pasx,bornesx); % moyenne
```

La première ligne du code fait appel à la fonction `calcul_bornes,` qui permet de réaliser la discrétisation de l'axe des abscisses.

Pour cela, la fonction reçoit en entrée :

- `x` : le vecteur à discrétiser ;
- `pas` : le nombre de pas utilisés pour la discrétisation de l'axe `x`. Plus le nombre de pas est élevé, plus la discrétisation est fine.

En sortie :

- `bornes` : les bornes qui permettent de découper le vecteur `x` en plusieurs intervalles.

```
function bornes=calcul_bornes(x,pas)
bornes=zeros(pas+1,2);
for j=1:pas+1
       bornes(j,1)= [(max(x)-min(x))/pas].*(j-1)+ min(x);
       bornes(j,2)= [(max(x)-min(x))/pas].*(j)+ min(x);
end
```

Ensuite, l'objectif est d'identifier les valeurs qui appartiennent à chaque tranche de Δx. Dans ce but, nous avons programmé la fonction `matrice_distribution`.

La fonction reçoit en en entrée :

- `x`: la valeur de la mesure en abscisses. Ce vecteur sert à identifier les cas où l'ensemble des mesures est aligné verticalement, c'est-à-dire, l'ensemble des données présente la même valeur `x`;
- `y`: la valeur de la mesure en ordonnées. Nous allons identifier les valeurs de `y` qui appartient à chaque tranche verticale;
- `N`: la taille des vecteurs `x` et `y`;
- `pas` : le nombre de pas utilisés pour la discrétisation de l'axe `x`;
- `bornes`: les bornes utilisées pour la discrétisation de l'axe `x`. En théorie, le nombre de `bornes = pas + 1`;

En sortie :

- `intervalle_y` : les valeurs de `y` pour chaque tranche verticale. Nous faisons tourner cette fonction trois fois, nous aurons donc trois résultats :
    - `interv_incertitude_ymin` lorsqu'on considère les valeurs minimales de `y (incertitude_ymax);`
    - `interv_incertitude_ymax` lorsqu'on considère les valeurs maximales de `y (incertitude_ymax);`
    - `intervalle_y` lorsqu'on considère les valeurs moyennes de `y (y);`

La fonction réalise les opérations suivantes :

- Elle crée une matrice :

$$\begin{bmatrix} x(i_1), & y(i_1), & j(i_1) \\ x(i_2), & y(i_2), & j(i_2) \\ x(i_3), & y(i_3), & j(i_3) \\ \dots & \dots & \dots \\ x(i_N), & y(i_N), & j(i_N) \end{bmatrix}$$

Pour chaque mesure $[x(i),\ y(i)]$, on identifie la tranche de x à laquelle la mesure appartient (exprimée par `j`). Par exemple, si nous avons 20 tranches de x, la valeur de j peut être égale à une des valeurs parmi les suivantes : $\{1, 2, 3, \ldots, 20\}$. De plus, il faut tenir compte du cas particulier où les données sont alignées verticalement : les mesures appartiendront à la même tranche de x, le numéro de la tranche sera forcement égal à 1 : $j = 1$;

- Ensuite, les valeurs de $y(i)$ possédant la même valeur de j sont rangées ensemble. L'objectif est de construire une matrice :

$$\begin{bmatrix} y(i_1), & y(i_2), & y(i_3), & \ldots \\ y(i_4), & y(i_5), & y(i_6), & \ldots \\ y(i_7), & y(i_8), & y(i_9), & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ y(i_{N-2}), & y(i_{N-1}), & y(i_N), & \ldots \end{bmatrix} \text{ pour chaque } \begin{bmatrix} j(1) \\ j(2) \\ j(3) \\ \ldots \\ j(n) \end{bmatrix}$$

`n` étant le nombre de bornes,
`N` étant le nombre des mesures,

Cette matrice permet de repérer les mesures pour chaque tranche verticale $[x_i, x_{i+1}[$. Le nombre de colonnes est égal au nombre de bornes et le nombre de lignes au nombre maximum de mesures appartenant à une des tranches. Il faut également considérer le cas où l'ensemble des données partage la même valeur de x. La matrice précédente n'aurait alors qu'une seule ligne :

$$[y(i_1),\quad y(i_2),\quad y(i_3),\ \ldots]\ \text{pour}\ j(1)$$

Le code est le suivant :

```
% 1.Créer une matrice [x(i) y(i) j], où [j] est le numéro d'intervalle

function intervalle_y=matrice_distribution (x,y,N,pas,bornes)
taille_C=0;

% l'ensemble des points présentent le même x
if min(x)==max(x)

    for i=1:N
        C(i,1:3)=[x(i),y(i),1];
    End

taille_C=N;
end

% dans le cas contraire
for i=1:N
    for j=1:pas+1

        if (x(i)>= bornes(j,1)& x(i)< bornes(j,2) & isnan(y(i)) == 0)
            taille_C=taille_C+1 ;
            C(taille_C,1:3)=[x(i), y(i),j];
        end

    end
```

```
end
C=sortrows(C,3);

% 2.Créer une matrice avec les différents valeurs de [y] pour chaque intervalle
% de [j]: [y(i11) y(i12) y(i13) ...] pour j1

% l'ensemble des points présentent le même x
intervalle_y=zeros(1,pas+1);

if min(x)==max(x)
    s=1;
    for i=1:taille_C
        if C(i,3)==1
            intervalle_y(s,1)=C(i,2);
            s=s+1;
        end
    end

else % dans le cas contraire
    for j=1:pas+1
        s=1;
        for i=1:taille_C
            if C(i,3)==j
               intervalle_y(s,j)=C(i,2);
               s=s+1;
            end
        end
    end
end
```

Ensuite, on compte le nombre de cases occupées par les mesures pour chaque tranche verticale. Cette variable, nommée `nb_points`, sera utilisée dans la suite du programme.

```
% calculer le nombre des points pour chaque intervalle
nb_points=zeros(1,pasx+1);
for j=1:size(intervalle_y,2)
    for i=1:size(intervalle_y,1)
        if intervalle_y(i,j)>0
            nb_points(j)=nb_points(j)+1;
        end
    end
end
```

C. *Identifier si, compte-tenu de l'incertitude en y, la mesure s'étale sur plusieurs intervalles de discrétisation en y*

Au cours de cette troisième phase, nous allons :

- discrétiser l'axe des ordonnées. Pour cela, on fait de nouveau appel à la fonction `calcul_bornes` ;
- identifier si, compte tenu de l'incertitude en `y`, la mesure occupe plusieurs cases de discrétisation, c'est-à-dire, si pour une seule et même valeur de `x`, on trouve plusieurs valeurs possibles de `y`. Si oui :
  - on quantifie le nombre de cases de séparation entre `ymin` et `ymax` (exprimée par la variable `distance_bornes`) ;
  - on identifie le numéro de la borne inférieure occupée par la mesure (exprimée par la variable `nb_bornes_inf`).

  Pour cela, on fait appel à la fonction `distrib_incertitude`.

La troisième étape de la fonction `detection_anomalies_main` se présente sous la forme suivante :

```
% 3. Identifier si "incertitude_ymin" et "incertitude_ymax" sont localisés sur
% le même carré y. En cas contraire quantifier la distance de séparation

bornesy=calcul_bornes(incertitude_y,pasy);

if min(x)==max(x); % l'ensemble des points présente le même x

[distribution,nb_bornes]=
distrib_incertitude(interv_incertitude_ymin(1:size(interv_incertitude_ymin,1),1)
, interv_incertitude_ymax(1:size(interv_incertitude_ymax,1),1),
size(interv_incertitude_ymin,1),pasy,bornesy);

       distance_bornes (:,1)=distribution(:,1);
       nb_bornes_inf(:,1)=nb_bornes(:,1);

else
       for i=1:pasx+1

[distribution,nb_bornes]=
distrib_incertitude(interv_incertitude_ymin(1:size(interv_incertitude_ymin,1),i),
interv_incertitude_ymax(1:size(interv_incertitude_ymax,1),i),
size(interv_incertitude_ymin,1),pasy,bornesy);

       distance_bornes (:,i)=distribution(:,1);
       nb_bornes_inf(:,i)=nb_bornes(:,1);

       end
end
```

Le programme fait appel à la fonction `distrib_incertitude` , qui reçoit en entrée :

- `x1` : les valeurs du premier vecteur. Dans ce cas, ce vecteur correspond à la variable `interv_incertitude_ymin`;
- `x2` : les valeurs du deuxième vecteur. Dans ce cas, ce vecteur correspond à la variable `interv_incertitude_ymax`;
- `N` : taille des vecteurs `x1` et `x2`;
- `pas` : le nombre de pas. Dans ce cas, cette variable correspond au nombre d'intervalles utilisés pour discrétiser l'axe des ordonnées ;
- `bornes` : la valeur des bornes. Dans ce cas, cette variable correspond à la valeur des bornes utilisées pour discrétiser l'axe des ordonnées ;

En sortie, nous obtenons deux résultats :

- `distance_bornes`: quantifie le nombre d'intervalles de séparation entre `ymin` et `ymax` ;
- `nb_bornes_inf`: indique le numéro de la borne inférieure occupée par la mesure;

La mesure présente une seule valeur de `x` et donc appartient à une tranche verticale déterminée. La fonction `distrib_incertitude` permet alors de vérifier si la mesure s'étale sur plusieurs cases de discrétisation sur l'axe d'ordonnées y :
$[y_i, y_{i+1}[+ [y_{i+1}, y_{i+2}[+ [y_{i+2}, y_{i+3}[+ \text{etc} + [y_{n-1}, y_n[$, n étant le dernier intervalle sur lequel la mesure s'étale.

En effet, la vraie valeur de `y` peut appartenir à n'importe quel intervalle parmi les précédents et plus particulièrement, elle peut être égale à n'importe quelle valeur entre les bornes de ces intervalles et tout cela avec une probabilité identique.

Dans ce but, la fonction `distrib_incertitude` identifie le nombre de l'intervalle de discrétisation contenant la variable `interv_incertitude_ymin` et la variable `interv_incertitude_ymax`. Ces variables correspondent aux valeurs extrêmes que la mesure peut prendre.

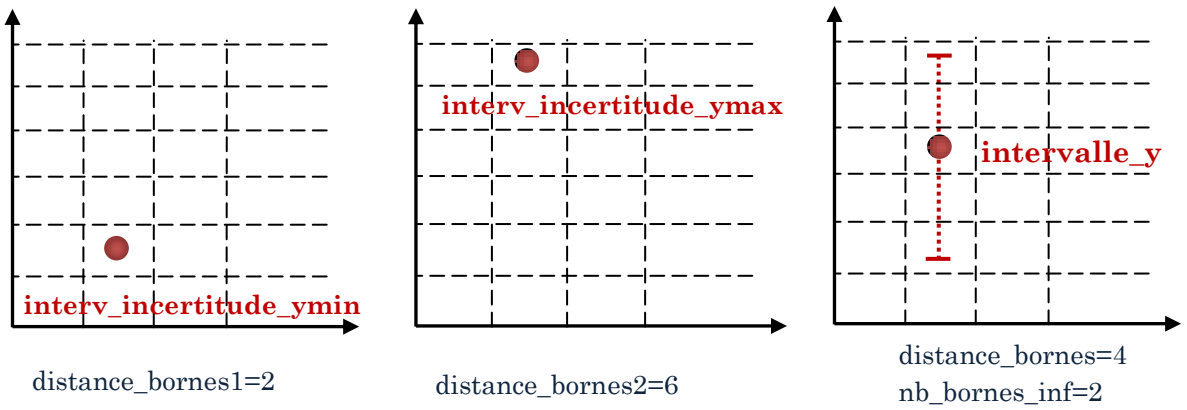


Figure 39 : Illustration des trois variables : interv_incertitude_ymin, interv_incertitude_ymas et intervalle_y.

La variable `distance_bornes1` identifie le rang de l'intervalle inférieur de discrétisation occupée par la mesure. Dans la Figure 39, la variable `interv_incertitude_ymin` occupe le

deuxième intervalle, en conséquence `distance_bornes1=2`. De même, la variable `distance_bornes2` informe sur le rang de l'intervalle supérieur de discrétisation occupée par la mesure. Dans la Figure 39, la variable `interv_incertitude_ymax` occupe le sixième intervalle, en conséquence `distance_bornes2=6`.

Si la mesure est entachée d'une incertitude en `y`, et n'est pas contenue dans un unique intervalle ($distance_{bornes1} \neq distance_{bornes2}$), la fonction calcule :

- `distance_bornes` : le nombre d'intervalles séparant l'intervalle inférieur de l'intervalle supérieur de la mesure ;
- `nb_bornes_inf` : le numéro de l'intervalle inférieur présentant la mesure.

Pour les mesures appartenant à un seul intervalle de discrétisation, ces deux variables : `distance_bornes` et `nb_bornes_inf` seront égales à zéro.

Enfin, il faut également tenir compte du cas particulier `min(x)=max(x)` : le principe de calcul est le même mais on considère un seul intervalle de discrétisation $[x_i, x_{i+1}[$.

La fonction `distrib_incertitude` est codée comme suit :

```
function [distance_bornes,nb_bornes_inf]=distrib_incertitude (x1,x2,N,pas,bornes)

distance_bornes1=zeros(1);
distance_bornes2=zeros(1);
distance_bornes=zeros(N,1);
nb_bornes_inf=zeros(N,1);

for i=1:N
    distance=0;
    distance_bornes1=0;
    distance_bornes2=0;
    for j=1:pas+1

       if (x1(i)>= bornes(j,1)& x1(i)< bornes(j,2))
           distance_bornes1=j;
       end
       if (x2(i)>= bornes(j,1)& x2(i)< bornes(j,2))
           distance_bornes2=j;
       end

       distance=distance_bornes2-distance_bornes1;

   end
   distance_bornes(i,1)=distance;
   nb_bornes_inf(i,1)=distance_bornes1;

end

distance_bornes = distance_bornes(1:N,1);
nb_bornes_inf=nb_bornes_inf(1:N,1);
```

D. *Créer une nouvelle variable "intervalle_newy" qui comprend l'ensemble des valeurs possibles que y peut prendre.*

Actuellement, nous connaissons la valeur moyenne de la mesure (`intervalle_y`) et ses extremas (`interv_incertitude_ymin` et `interv_incertitude_ymax`). L'objectif de cette étape est la création d'une nouvelle variable (`intervalle_newy`) qui regroupe l'ensemble des valeurs possibles que `y` peut prendre.

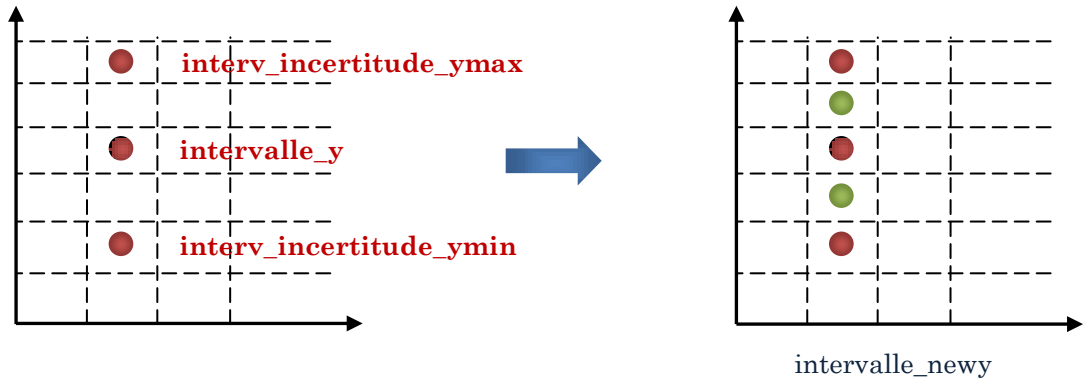


Figure 40 : Création d'une nouvelle variable « intervalle_newy »

Pour cela, nous distinguons les cas où la variable `distance_bornes` est strictement inférieure à 1 – c'est-à-dire que la mesure occupe une seule case de discrétisation en y – des cas où la variable `distance_bornes` est supérieure ou égale à 1 (la mesure occupe au moins une case de discrétisation).

Dans le premier cas, la variable `intervalle_newy` conserve la valeur de `intervalle_y`. Dit autrement, en sachant que les extremas appartiennent à la même case de discrétisation, la valeur de y sera égale à la moyenne de `ymin` et de ymax.

Pour le deuxième cas, la fonction crée autant de nouvelles lignes que de cases occupées en plus à cause de l'incertitude de la mesure. Dans la figure 40 par exemple, il y 5 cases occupées par la mesure ; le programme créera alors 4 nouvelles lignes, ce qui fait un total de 5 lignes pour la mesure `(nb_lignes=5)`. Ensuite, on attribue à chaque case occupée par la mesure la valeur moyenne de la case de discrétisation :

```
intervalle_newy(1) = bornes_meany(2);
intervalle_newy(2) = bornes_meany(3);
intervalle_newy(3) = bornes_meany(4);
intervalle_newy(4) = bornes_meany(5);
intervalle_newy(5) = bornes_meany(6);
```

Cela signifie que la mesure peut être présente dans n'importe lequel de ces 5 intervalles occupés par l'incertitude. Toutefois, il faut faire attention à ne pas compter 5 fois la même mesure. Afin d'éviter cela, nous avons créé une variable $\_newy = \frac{1}{nb\_lignes}$. Si la mesure occupe une seule case de discrétisation `compteur_newy`=1, et si, comme dans l'exemple, la mesure occupe 5 cases de discrétisation `compteur_newy`=0.2. Cette variable fait référence à la probabilité de la mesure de se retrouver dans n'importe quelle case du fait de l'incertitude de la mesure.

Enfin, l'objectif pratique de la quatrième étape de calcul est d'obtenir en sortie ces deux variables :

- `intervalle_newy` : l'ensemble des valeurs possibles de y ;
- `compteur_newy` : la probabilité de la mesure de se trouver dans les différentes cases de discrétisation.

Voici le code :

```
% 4. Créer un nouveau vecteur "intervalle_new_y" afin de tenir compte de
% l'incertitude en y

% calculer la valeur moyenne de chaque borne
bornes_meany=mean(bornesy,2);
intervalle_newy=zeros(1,pasy+1);
compteur_newy=zeros(1,pasy+1);

for j=1:size(distance_bornes,2)
nb_lignes=1;
    for i=1:size(distance_bornes,1)

       % si interv_incertitude_ymin ~= interv_incertitude_ymax
      if distance_bornes(i,j)>=1

             %nb_intervalles = distance_bornes+1
            for indice=1:distance_bornes(i,j)+1

                    intervalle_newy(nb_lignes,j)=
                    bornes_meany(nb_bornes_inf(i,j)+indice-1);

                    compteur_newy(nb_lignes,j)=(1/(distance_bornes(i,j)+1));
                    nb_lignes=nb_lignes + 1;
            end

      % si interv_incertitude_ymin = interv_incertitude_ymax
       else
             % moyenne de ymin et ymax
            intervalle_newy(nb_lignes,j)=intervalle_y(i,j);
            if intervalle_y(i,j)==0
                compteur_newy(nb_lignes,j)=0;
            else
                compteur_newy(nb_lignes,j)=1;
            end
            nb_lignes=nb_lignes+1;
        end
    end
end

intervalle_newy=intervalle_newy(:,1:pasy+1);
compteur_newy=compteur_newy(:,1:pasy+1);
```

2. Construction des histogrammes des points de mesure

La quatrième étape de la fonction `detection_anomalies_main` permet de construire l'histogramme des points de mesure. Ces histogrammes constituent la base pour ensuite identifier les données suspectes : on observe une anomalie lorsque l'on identifie un trou, c'est-à-dire des intervalles vides, dans l'histogramme.

Au cours de cette phase, nous calculons également les histogrammes cumulés des points de mesures. Ces histogrammes servent ensuite à calculer le poids de données suspectes.

Pour le calcul des histogrammes, nous avons programmé les fonctions suivantes :

OBJECTIF	FONCTION
Appel des fonctions.	detection_anomalies_main (étape 4)
Construction d'histogramme des points de mesure.	calcul_distribution
Construction d'histogramme cumulé des points de mesure.	calcul_cumul
Mise en forme de la matrice « distrib», « cumul_haut » et « cumul_bas ».	mise_forme

Voici le code correspondant à cette quatrième étape:

```
%---------------------------------ETAPE 4---------------------------------------
%-------------- Construction d'histogramme des points de mesure ----------------

distrib=zeros(pasx+1,pasy+1);
cumul_haut=zeros(pasx+1,pasy+1);
cumul_bas=zeros(pasx+1,pasy+1);

for i=1:pasx+1
    distribution=calcul_distribution (intervalle_newy(1:size(intervalle_newy,1),i),
    size(intervalle_newy,1),pasy,bornesy,compteur_newy(1:size(compteur_newy,1),i));
    distrib(i,1:pasy+1)=distribution(1,1:pasy+1);

    [distrib_cumul_haut,distrib_cumul_bas]=
    calcul_cumul (intervalle_newy(1:size(intervalle_newy,1),i),size(intervalle_newy,1),
    pasy,bornesy,compteur_newy(1:size(compteur_newy,1),i),nb_points(i));
    cumul_haut(i,1:pasy+1)=distrib_cumul_haut(1,1:pasy+1);
    cumul_bas(i,1:pasy+1)=distrib_cumul_bas(1,1:pasy+1);
end

% mise en forme des tableaux "distrib" et "cumul"
distrib=mise_forme(distrib,pasy,pasx);
cumul_haut=mise_forme(cumul_haut,pasy,pasx);
cumul_bas=mise_forme(cumul_bas,pasy,pasx);
```

Au cours de l'étape précédente, nous avons discrétisée le plan (x, y) en rectangles réguliers. L'objectif de cette quatrième étape est la construction d'histogrammes des points de mesure : pour chaque tranche verticale, on compte le nombre de points contenus dans chaque intervalle $[y_j, y_{j+1}[$.

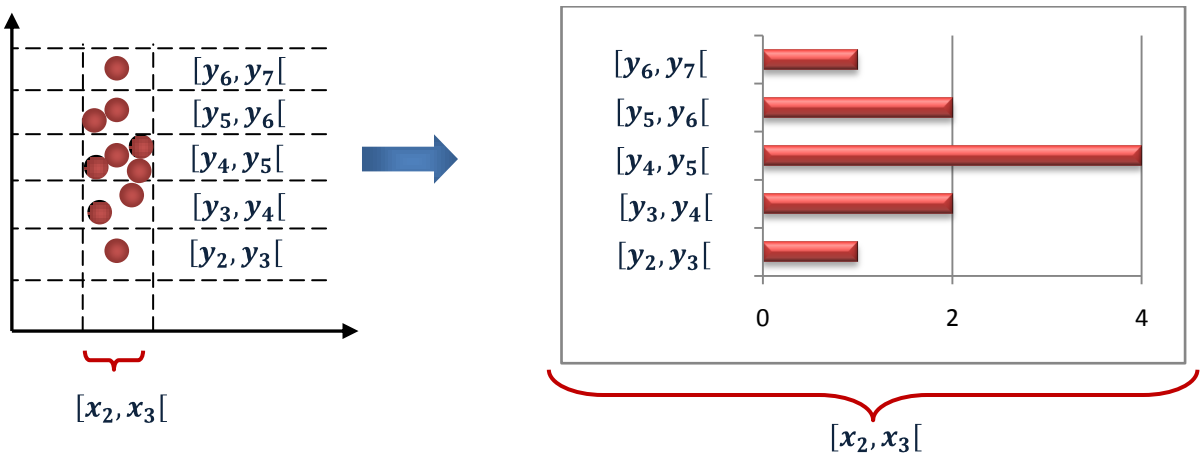


Figure 41 : Exemple de construction d'histogramme des points de mesure

On connait donc la distribution des mesures dans chaque tranche verticale$[x_i, x_{i+1}[$; leur étude nous permettra ensuite de détecter les anomalies. Pour la construction d'histogrammes, on fait appel à la fonction `calcul_distribution`.

En entrée, la fonction reçoit :

- `x` : la valeur de la mesure. Dans ce cas, `x` correspond à la variable `inter-valle_newy` ;
- `N` : taille de la variable `x` ;
- `pas` : le nombre d'intervalles de discrétisation. Dans ce cas, `pas` désigne les intervalles de discrétisation en `y` ;
- `bornes` : la valeur des bornes. Dans ce cas, `bornes` désigne les bornes en `y` ;
- `compteur_newy` : ce paramètre permet de prendre en compte la probabilité que la mesure appartienne à différentes cases de discrétisation.

En sortie :

- `distribution` : la distribution des mesures pour une tranche donnée $[x_i, x_{i+1}[$.

La fonction `calcul_distribution` est codée comme suit :

```
function distribution=calcul_distribution (x,N,pas,bornes,compteur_newy)

distribution=zeros(pas+1,1);
for i=1:N
   for j=1:pas+1
          if (x(i)>= bornes(j,1)& x(i)< bornes(j,2)& x(i)~=0)
                 distribution(j)=distribution(j)+1*compteur_newy(i);
          elseif (x(i)>= bornes(j,1)& x(i)== bornes(j,2)& x(i)~=0)
                 distribution(j)=distribution(j)+1*compteur_newy(i);
          end
   end
end
distribution=distribution(1:pas+1)';
```

D'autre part, une fois les données suspectes identifiées, nous avons besoin de connaître leur poids. Dans ce but, nous construisons les histogrammes cumulés des points de mesure : pour chaque tranche verticale, on compte le nombre de points supérieurs ou égal à la borne inférieure de la tranche $[y_j, y_{j+1}[$.

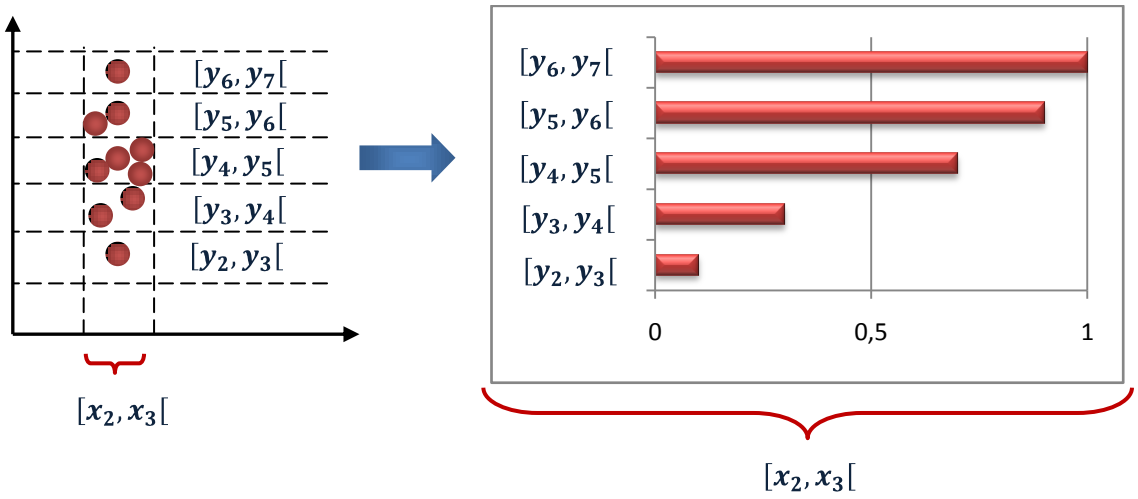


Figure 42 : Exemple de construction d'histogramme cumulé des points de mesure en direction haut → bas (variable : cumul_haut)

On connait donc la distribution cumulée des mesures dans la tranche $[x_i, x_{i+1}[$; leur étude nous permettra ensuite de calculer le poids des données suspectes. La variable est nommée `cumul_haut` car l'histogramme cumulé va dans le sens : haut→bas.

Toutefois, on peut également construire un histogramme cumulé dans le sens bas→ haut . Dans ce cas, pour chaque tranche verticale, on compte le nombre de points inférieurs ou égaux à la borne supérieure de la tranche $[y_j, y_{j+1}[$. La variable ainsi construite est nommée `cumul_bas`.

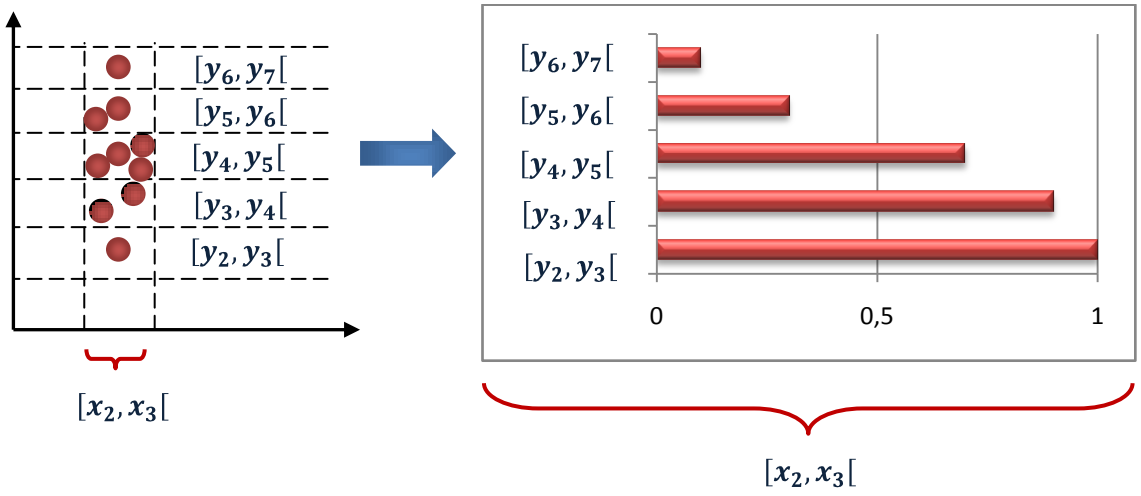


Figure 43 : Exemple de construction d'histogramme cumulé des points de mesure en direction bas → haut (variable : cumul_bas)

Pour la construction d'histogrammes cumulés, on fait appel à la fonction `calcul_cumul`.

En entrée, la fonction reçoit :

- `x` : la valeur de la mesure. Dans ce cas, x correspond à la variable `intervalle_newy` ;
- `N` : taille de la variable `x` ;
- `pas` : le nombre d'intervalles de discrétisation. Dans ce cas, pas désigne les intervalles de discrétisation en y ;
- `bornes` : la valeur des bornes. Dans ce cas, bornes désigne les bornes en `y` ;
- `compteur_newy` : ce paramètre permet de prendre en compte la probabilité que la mesure appartienne à différentes cases de discrétisation ;
- `nb_points` : ce paramètre désigne pour chaque tranche $[x_i, x_{i+1}[$, le nombre de cases occupées par des mesures. Il sert à normaliser l'histogramme cumulé afin d'obtenir la loi de probabilité cumulée.

En sortie :

- `cumul_haut`: la distribution cumulée des mesures pour une tranche donnée $[x_i, x_{i+1}[$, direction haut→ bas;
- `cumul_bas` : la distribution cumulée des mesures pour une tranche donnée $[x_i, x_{i+1}[$, direction bas→haut.

La fonction `calcul_cumul` est codée comme suit :

```
function [cumul_haut, cumul_bas]=
calcul_cumul (x,N,pas,bornes,compteur_newy,nb_points)

% Calcul de la distribution cumulée en direction haut -> bas

cumul_haut=zeros(pas+1);
for i=1:N
   for j=1:pas+1
       if (x(i)>= bornes(j,1)& x(i)~=0)
           cumul_haut(j)=cumul_haut(j)+1*compteur_newy(i);
       end
   end
end

if nb_points~=0
    cumul_haut=cumul_haut(1:pas+1)/nb_points;
end

% Calcul de la distribution cumulée en direction bas -> haut

cumul_bas=zeros(pas+1);
for i=1:N
   for j=0:pas
       if (x(i)<= bornes(pas+1-j,2)& x(i)~=0)
           cumul_bas(pas+1-j)=cumul_bas(pas+1-j)+1*compteur_newy(i);
       end
   end
end

if nb_points~=0
    cumul_bas=cumul_bas(1:pas+1)/nb_points;
end
```

Enfin, la fonction `mise_forme` sert à mettre en forme les trois matrices : `distrib`, `cumul_haut` et `cumul_bas`. Nous ajoutons une première colonne qui sert à indiquer le nombre d'intervalle y de discrétisation auxquelles les mesures appartiennent. Voici le code :

```
function matrice=mise_forme(matrice,pasy,pasx)

a=zeros(pasy+1,1);
for i =1:pasy+1
    a(i)=i;
    i=i+1;
end
matrice=matrice(1:pasx+1,1:pasy+1);
matrice=matrice';
matrice(1:pasy+1,2:pasx+2)=matrice(1:pasy+1,1:pasx+1);
matrice(1:pasy+1,1)=a(1:pasy+1,1);
```

3. Représentation graphique

Cette cinquième étape permet de visualiser les données avec la bonne échelle avant et après la prise en compte des incertitudes. De même, nous pouvons visualiser les histogrammes des points mesurés pour chaque tranche verticale.

Cette étape de calcul n'est pas obligatoire mais elle permet de vérifier les résultats obtenus. En fait, la représentation graphique est surtout intéressante afin de vérifier les différent cas test (113 réactions). Toutefois, lorsqu'on applique la méthode de détection des anomalies à la totalité de la base de données, ce module graphique est très gourmand au niveau du temps de calcul, il peut donc être utile de le désactiver.

Voici le code :

```
%------------ETAPE 5--------------------------------
%---Représentation graphique-------------------

% représentation de la graphique avec l'échelle idéale
graph_echelle_ideale(x,y,bornesx,bornesy,nom_fichier);
saveName=(['fig', num2str(nb_reaction)]);
saveas(gcf, saveName, 'jpg');
hold on;
figure (2);

% représentation de l'incertitude sur le graphique précédent
[newx, newy]=calcul_graph_incertitude (x,y,pasx,bornesx,bornesy,intervalle_newy);
graph_echelle_ideale(newx,newy,bornesx,bornesy,nom_fichier);
saveName=(['fig_bis', num2str(nb_reaction)]);
saveas(gcf, saveName, 'jpg');

% représentation de la distribution
graph_distrib(distrib,pasx,bornesx);
```

La fonction `graph_echelle_ideale` permet d'obtenir la représentation graphique de la réaction avec la bonne échelle. Pour cela, la fonction reçoit en entrée :

- `x` : la valeur moyenne de la mesure en abscisses ;
- `y` : la valeur moyenne de la mesure en ordonnées ;
- `bornesx` : la valeur de la borne minimale et maximale pour tranche $[x_i, x_{i+1}[$;
- `bornesy` : la valeur de la borne minimale et maximale pour tranche $[y_j, y_{j+1}[$;
- `nom_fichier` : ce vecteur renseigne sur le nom du fichier. Cette variable est utilisée ensuite pour le titre de la graphique.

En sortie : la représentation graphique de la réaction.

Voici le code :

```
function graph_echelle_ideale(x,y,bornesx,bornesy,nom_fichier)

scatter(x, y);
title(nom_fichier);
xlabel('Incident Energy');
ylabel('Y');
hold on;

if min(x)~=max(x)
    Axis([min(x) max(x) min(y) max(y)]);
    bornes(:,1)=bornesx(:,1);
    bornes(end+1,1)=bornesx(end, 2);
    plot([bornes(:,1) bornes(:,1)],[min(bornesy(:,1)) max(bornesy(:,2))], 'k-');
    clear bornes;
elseif min(x)==max(x)
    Axis([(min(x)-min(x)/10) (max(x)+max(x)/10) min(y) max(y)])
end

bornes(:,1)=bornesy(:,1);
bornes(end+1,1)=bornesy(end, 2);
plot([min(bornesx(:,1)) max(bornesx(:,2))],[bornes(:,1) bornes(:,1)],  'g--' );
clear bornes;
```

Ensuite, la fonction `calcul_graph_incertitude` permet de calculer les nouvelles valeurs des mesures `(x,y)` après la prise en compte des incertitudes. En fait, lors de la phase précédente, nous avons vu qu'une mesure peut être représentée par plusieurs points, c'est-à-dire pour une même valeur de `x` nous pouvons avoir plusieurs valeurs de `y`.

Si la mesure n'a pas d'incertitude, elle conserve les valeurs initiales en `x` et `y`. Si par contre, la mesure s'étale sur plusieurs cases de discrétisation en y, la fonction va attribuer à chaque `x` la valeur moyenne de l'intervalle. On suppose un point au milieu de la case de discrétisation.

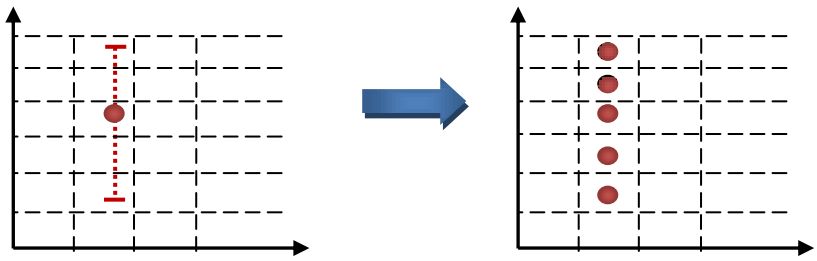


Figure 44 : Calcul des nouvelles valeurs des mesures (x,y)

Pour cela, la fonction reçoit en entrée :

- `x` : la valeur moyenne de la mesure en abscisses ;
- `y` : la valeur moyenne de la mesure en ordonnées ;
- `pasx` : le nombre de pas en `x`;
- `bornesx` : la valeur de la borne minimale et maximale pour tranche verticale $[x_i, x_{i+1}[$;
- `bornesy` : la valeur de la borne minimale et maximale pour tranche horizontale $[y_j, y_{j+1}[$;
- `intervalle_newy` : pour chaque tranche $[x_i, x_{i+1}[$, ce vecteur renseigne sur les nouvelles valeurs de la mesure en ordonnées après la prise en compte des incertitudes

En sortie :

- `newx` : la nouvelle valeur de la mesure en abscisses après la prise en compte des incertitudes ;
- `newy` : la nouvelle valeur de la mesure en ordonnées après la prise en compte des incertitudes.

Voici le code :

```
% Cette fonction permet de calculer les vecteur newx et newy nécessaires
% pour ensuite réaliser la graphique

function [newx, newy]=calcul_graph_incertitude (x,y,pasx,bornesx,bornesy,intervalle_newy)

% calculer la valeur moyenne de chaque borne
bornes_meanx=mean(bornesx,2);

% Calcul de newy
newy=[intervalle_newy(:,1);intervalle_newy(:,2)];
for i=3:size(intervalle_newy,2)
    newy=[newy(:,1);intervalle_newy(:,i)];
end

% Calcul de newx
indice=1;
for j=1:size(intervalle_newy,2)
    for i=1:size(intervalle_newy,1)
        newx(indice)=j;
        indice=indice+1;
    end
end
newx=newx';
[newy,newx]=suppr_lignes(newy,newx,size(newy,1));

bornes_xi=identifier_bornes(x,pasx,bornesx);
for i=1:size(y,1)
    for indice=1:size(newy,1)
        if (min(x)==max(x)& y(i)==newy(indice))
            newx(indice)=x(i);
        elseif (y(i)==newy(indice)& newx(indice)==bornes_xi(i) )
            newx(indice)=x(i);
        end
    end
```

```
end

for i=1:size(y,1)
    for indice=1:size(newy,1)
        if (newx(indice)>0 & newx(indice)<21 & newx(indice)==round(newx(indice)))
            newx(indice)=bornes_meanx(newx(indice),1);
        end
    end
end
```

Enfin, la fonction `graph_distrib` permet de représenter l'histogramme des mesures pour chaque tranche verticale $[x_i, x_{i+1}[$. Pour cela, la fonction reçoit en entrée :

- `vecteur` : la valeur de la variable à représenter. Dans ce cas, il s'agit de la distribution ;
- `pasx` : le nombre de pas en x ;
- `bornesx` : la valeur de la borne minimale et maximale pour tranche $[x_i, x_{i+1}[$;

En sortie : on obtient les histogrammes.

Voici le code :

```
function graph_distrib(vecteur,pasx,bornesx);

nb_figure=0;
nb_reconst =0;
nb_graph=0;

for i=2: pasx+2

    if(max(vecteur(:,i))~=0)

    % nb_reconst : compteur du nombre de reconstructions effectué au total
    nb_reconst=nb_reconst+1;
    % Le choix fait est d'afficher 4 graphiques par fenêtre, une fois la fenêtre remplie
       on en crée une autre.

    if ((nb_reconst-1)/16)==0+nb_figure
        figure
        nb_figure=nb_figure+1;
        nb_graph=0;
    end

    % Compteur du nombre de graphiques présents dans la fenêtre
    nb_graph=nb_graph+1;

        subplot(4,4,nb_graph);
        bar(vecteur(:,1),vecteur(:,i));
        title([bornesx(i-1,2)])
        xlabel('nombre dintervalle y');
        ylabel('densité y');
        Axis([min(vecteur(:,1)) max(vecteur(:,1)) min(vecteur(:,i)) max(vecteur(:,i))]);
    end

end
```

4. Identification des données suspectes

La sixième étape de la fonction `detection_anomalies_main` permet enfin d'identifier les données suspectes. En résumé, le programme réalise au cours de cette phase les opérations suivantes :

- détection des données suspectes ;
- calcul du poids des données suspectes ;
- recherche de l'intervalle de discrétisation en ordonnées et en abscisses auquel la mesure appartient ;
- recherche sur la base de données originale du numéro de la ligne présentant des données suspectes.

Au cours de cette phase, nous avons programmé les fonctions suivantes :

OBJECTIF	FONCTION
Appel des fonctions ; Transformation des mesures afin de retrouver les valeurs initiales présentes dans la base de données.	detection_anomalies_main (étape 6)
Création d'une nouvelle variable « transfor » à partir de « cumul_haut » et « cumul_bas ». Cette variable permettra d'identifier le poids de données suspectes.	transfor_cumul
Détection des données suspectes ; Calcul du poids de données suspectes ; Recherche de l'intervalle de discrétisation en ordonnées et en abscisses auquel la mesure appartient.	identifier_anomalies
Recherche dans la base de données originale du numéro de la ligne présentant des données suspectes.	calcul_ligne

Voici le code correspondant à cette dernière étape de calcul:

```
%-------------------------------------ETAPE 6---------------------------------
%-----------------------------Détection des anomalies-------------------------
% modifier la matrice de "poids" afin d'obtenir la matrice "poids_transfor"

transfor=zeros(pasx+1,pasy+2);
transfor=transfor_cumul(cumul_haut,cumul_bas,transfor);
donnees_aberrantes=identifier_anomalies(distrib,cumul_haut,transfor,
pasx,pasy,bornesx,bornesy);


% Chercher les données compris entre l'intervallex_min et l'intervallex_max
% et entre l'intervalley _min et l'intervalley_max

if (donnees_aberrantes(1,5)==0)
    nom_serie='-';
    name_serie_aberrante=0;

elseif donnees_aberrantes(1,5)~=0
```

```
    [nom_serie don
nees_aberrantes]=calcul_ligne(donnees_aberrantes,x,incertitude_ymin,incertitude_ymax, N,
serie, index);


    % calculer le nombre de fois qu'une série est identifiée comme aberrante
    % Supprimer tous les répétitions

    new_name=nom_serie(1);
    compteur=1;

    for i=1:size(nom_serie,1)-1
        if (isempty(nom_serie{i})==0 & isempty(nom_serie{i+1})==0 & ise-
qual(nom_serie{i},nom_serie{i+1})==0)
            compteur=compteur+1;
            new_name(compteur)=nom_serie(i+1);
        end
    end

    nom_serie_aberrante(compteur)=0;


    % calculer pour chaque série, le nombre des fois qu'elle est indiquée comme
    % aberrante

    for j=1: compteur
        for i=1:size(nom_serie,1)
            if (isempty(nom_serie{i})==0 & isequal(nom_serie{i},new_name{j})==1)
                nom_serie_aberrante(j)=nom_serie_aberrante(j)+1;
            end
        end
    end



    % créer un nouveau vecteur avec tant des lignes que nom_serie et dans le
    % même ordre

    for j=1: compteur
        for i=1:size(nom_serie,1)
            if (isempty(nom_serie{i})==0 & isequal(nom_serie{i},new_name{j})==1)
                name_serie_aberrante(i)=nom_serie_aberrante(j);
            end
        end
    end
    name_serie_aberrante=name_serie_aberrante';

end



% remplacer y et x par les vraies valeurs

for indice=1:size(donnees_aberrantes,1)

    donnees_aberrantes(indice,1)=
    (donnees_aberrantes(indice,1)./N).^(alpha_optimal_x);
    donnees_aberrantes(indice,2)=
    (donnees_aberrantes(indice,2)./N).^(alpha_optimal_x);
    donnees_aberrantes(indice,3)=
    (donnees_aberrantes(indice,3)./N).^(alpha_optimal_y);
```

```
    donnees_aberrantes(indice,4)=
    (donnees_aberrantes(indice,4)./N).^(alpha_optimal_y);

end
```

La deuxième ligne du code fait appel à la fonction `transfor_cumul`. Celle-ci permet de créer une nouvelle variable nommée `transfor`, qui sera utilisée ensuite pour le calcul du poids des données suspectes. Cette variable est créée à partir des distributions cumulées `cumul_haut` et `cumul_bas`.

La fonction reçoit en entrée :

- `cumul_haut` : la matrice cumulée direction: `haut(valeur=1)`→`bas(valeur=0);`
- `cumul_bas` : la matrice cumulée direction: `bas(valeur=1)`→`haut(valeur=0);`
- `transfor` : la matrice à transformer.

En sortie, on obtient :

- `transfor` : la matrice transformée.

Calculer le poids des données suspectes n'est pas trivial, il faut savoir quel sens choisir pour comptabiliser leur poids. Cela peut être en direction centre→haut : la variable `transfor` sera égale à `cumul_bas` ou bien, en direction centre→bas : la variable `transfor` sera égale à `cumul_haut`.

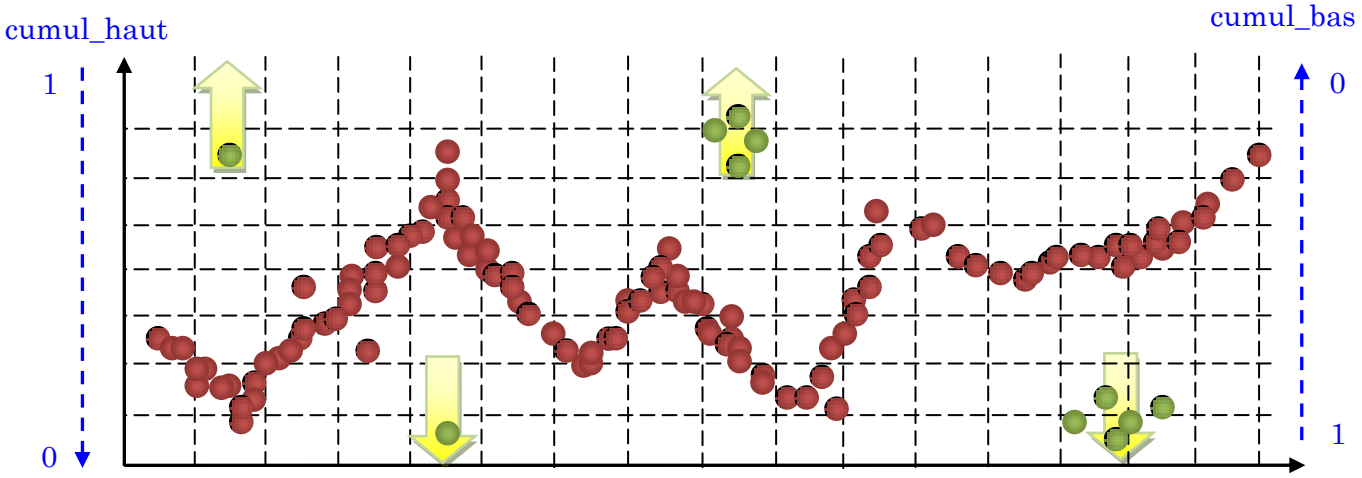


Figure 45 : Création d'une nouvelle variable transfor à partir de cumul_haut et cumul_bas

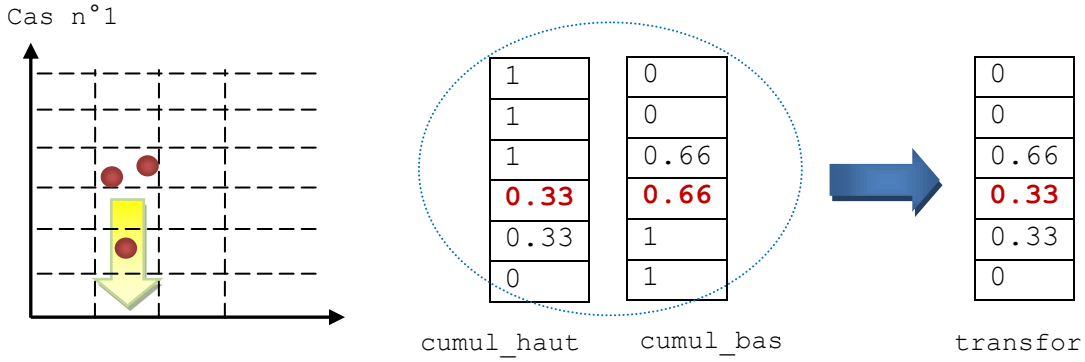
Afin de créer la variable `transfor`, on identifie la médiane pour les deux distributions, c'est-à-dire lorsque `cumul_haut=0.5` et `cumul_bas=0.5`. En théorie, si `cumul_haut>0.5`, alors `cumul_bas<0.5` et vice-versa. La nouvelle variable `transfor` est créée à partir des valeurs inférieures à `0.5` provenant de `cumul_haut` et de `cumul_bas` :

1. `Si, (cumul_haut<0.5 & cumul_bas>0.5), alors transfor=cumul_haut;`
2. `Si, (cumul_haut>0.5 & cumul_bas<0.5), alors transfor=cumul_bas;`

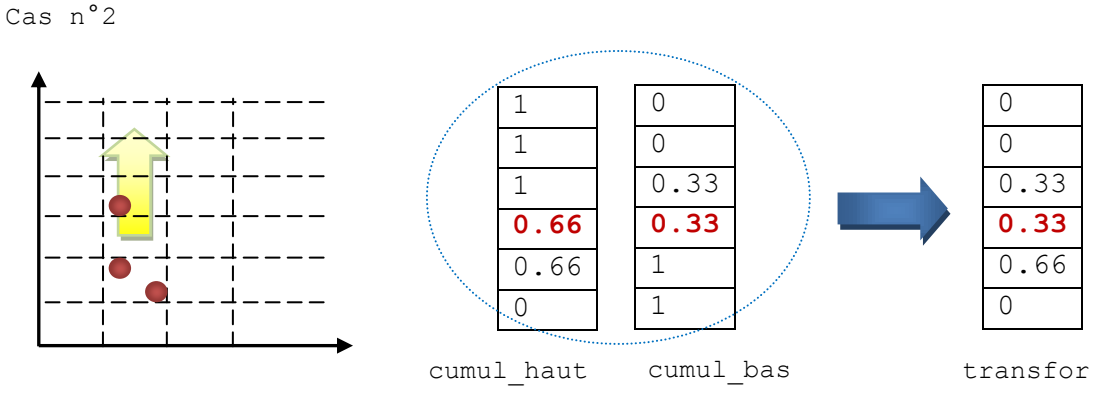
Ces deux premières conditions correspondent aux cas les plus courants. Toutefois, il est également possible de trouver que le poids est de 50% pour cumul_haut et cumul_bas. Dans ce cas, transfor est forcement égal à 0.5 :

3. Si, (cumul_haut=0.5 & cumul_bas=0.5), alors transfor=0.5;

A priori, le poids de n'importe quelle case vide détectée comme une anomalie peut être calculé à partir de ces trois conditions. Voici, une illustration de ces trois cas, où les tableaux à droite de la figure représentent les valeurs de cumul_haut, cumul_bas et transfor :



Par exemple pour ce premier cas, on compte deux données dans le quatrième intervalle de discrétisation et une donnée dans le second. La variable cumul_haut, qui représente la matrice cumulée en direction haut→bas, prend la valeur 1 pour les trois derniers intervalles (6,5 et 4), ensuite deux données sur trois ne sont plus comptées, cumul_haut prend alors la valeur 0.33 pour les intervalles 2 et 1. Enfin, cumul_haut est égale à zéro pour le premier intervalle car il n'y a plus de données. La variable cumul_bas est construite de la même façon, mais en sens inverse. Enfin, la matrice transfor est créée à partie de cumul_haut et cumul_bas et pour cela, il faut tenir compte des conditions spécifiées précédemment.



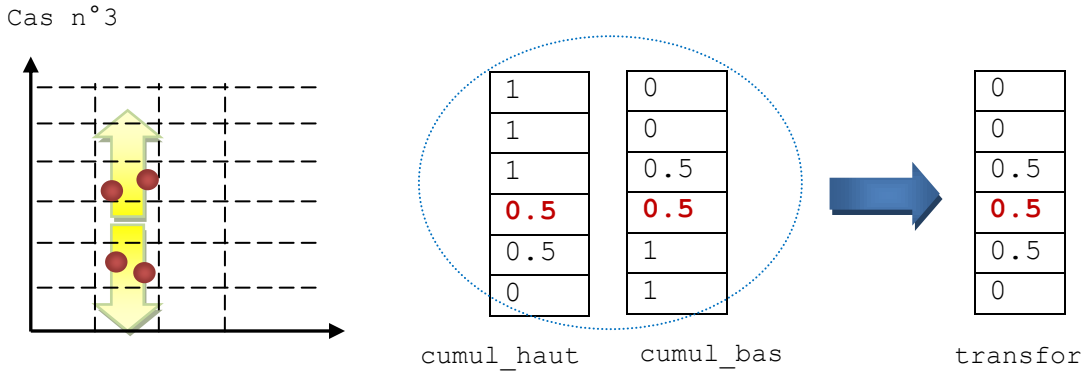


Figure 46 : Création d'une nouvelle variable `transfor`

Par définition, le poids des données suspectes peut être au maximum égal à 0,5. Dans ce cas, on est uniquement capable d'indiquer l'existence d'une anomalie mais on ne peut pas identifier quelles sont les données aberrantes.

Toutefois, il faut souligner que la `matrice_transfor` peut présenter des poids supérieurs à 0.5 car elle renseigne également sur le poids de l'ensemble des cases de discrétisation (pas uniquement celles identifiées comme suspectes). Voici les autres conditions qui complètent la liste des possibilités :

4. Si, (cumul_haut>0.5 & cumul_bas=0.5), alors transfor=cumul_haut;
5. Si, (cumul_haut=0.5 & cumul_bas>0.5), alors transfor=cumul_bas;
6. Si, (cumul_haut = 0.5 & cumul_bas=1), alors transfor =0.5;
7. Si, (cumul_haut=1 & cumul_bas=0.5), alors transfor=0.5;
8. Si, (cumul_haut=cumul_bas), alors transfor=cumul_haut;
9. Si, (cumul_haut>0.5 & cumul_bas>0.5 & cumul_haut> cumul_bas), alors transfor=cumul_bas;
10. Si, (cumul_haut>0.5 & cumul_bas>0.5 & cumul_haut< cumul_bas), alors transfor=cumul_haut;
11. Pour les autres cas, transfor=0.5;

Voici le code :

```
function transfor=transfor_cumul(cumul_haut, cumul_bas, transfor);

cumul_haut=round(cumul_haut*10000)/10000;
cumul_bas=round(cumul_bas*10000)/10000;

for j=2:size(transfor,2)
    for i=1:size(transfor,1)
        if (cumul_haut(i,j)==0.5 & cumul_bas(i,j)==1)
           transfor(i,j)=0.5;
        elseif (cumul_haut(i,j)==1 & cumul_bas(i,j)==0.5)
           transfor(i,j)=0.5;
        elseif (cumul_haut(i,j)==0.5 & cumul_bas(i,j)==0.5)
            transfor(i,j)=0.5;
```

```
        elseif cumul_haut(i,j)==cumul_bas(i,j)
            transfor(i,j)=cumul_haut(i,j);
        elseif (cumul_haut(i,j)<0.5 & cumul_bas(i,j)>0.5)
           transfor(i,j)=cumul_haut(i,j);
        elseif (cumul_haut(i,j)>0.5 & cumul_bas(i,j)<0.5)
            transfor(i,j)=cumul_bas(i,j);
        elseif (cumul_haut(i,j)==0.5 & cumul_bas(i,j)>0.5)
           transfor(i,j)=cumul_bas(i,j);
        elseif (cumul_haut(i,j)>0.5 & cumul_bas(i,j)==0.5)
            transfor(i,j)=cumul_haut(i,j);
        elseif (cumul_haut(i,j)>0.5 & cumul_bas(i,j)>0.5 & cumul_haut(i,j)> cu
      mul_bas(i,j) )
            transfor(i,j)=cumul_bas(i,j);
        elseif (cumul_haut(i,j)>0.5 & cumul_bas(i,j)>0.5 & cumul_haut(i,j)< cu
      mul_bas(i,j) )
            transfor(i,j)=cumul_haut(i,j);
        else
            transfor(i,j)=0.5;
        end
    end
end

transfor(1:end,1)=cumul_haut(1:end,1);
transfor(1:end,2:end)=transfor(1:end,2:end);
```

La fonction `identifier_anomalies` réalise les opérations suivantes :

- détection des données suspectes ;
- calcul du poids des données suspectes ;
- recherche de l'intervalle de discrétisation en ordonnées et en abscisses auquel la mesure appartient ;

Pour cela, la fonction reçoit en entrée :

- `distrib` : la matrice de distribution ;
- `cumul_haut` : la matrice de distribution cumulée, dans le sens `haut(valeur=1)`→`bas(valeur=0);`
- `pasx` : le nombre de pas en `x` ;
- `pasy` : le nombre de pas en `y` ;
- `bornesy` : le nombre des bornes en `y` ;
- `bornesx` : le nombre des bornes en `x`.

En sortie, on obtient :

- `donnees_aberrantes` : ce matrice renseigne sur :
  - l'intervalle $[x_i, x_{i+1}[$ auquel la mesure appartient ;
  - l'intervalle $[y_j, y_{j+1}[$ auquel la mesure appartient ;
  - le nombre de carrés vides ;
  - le poids des données suspectes.

L'objectif de la première étape de cette fonction est de créer une nouvelle variable nommée `compteur`. Cette variable, créée à partir de `distrib`, permet d'identifier pour chaque tranche $[x_i, x_{i+1}[$, les cases de discrétisation en `y` qui présentent des données. Par exemple :

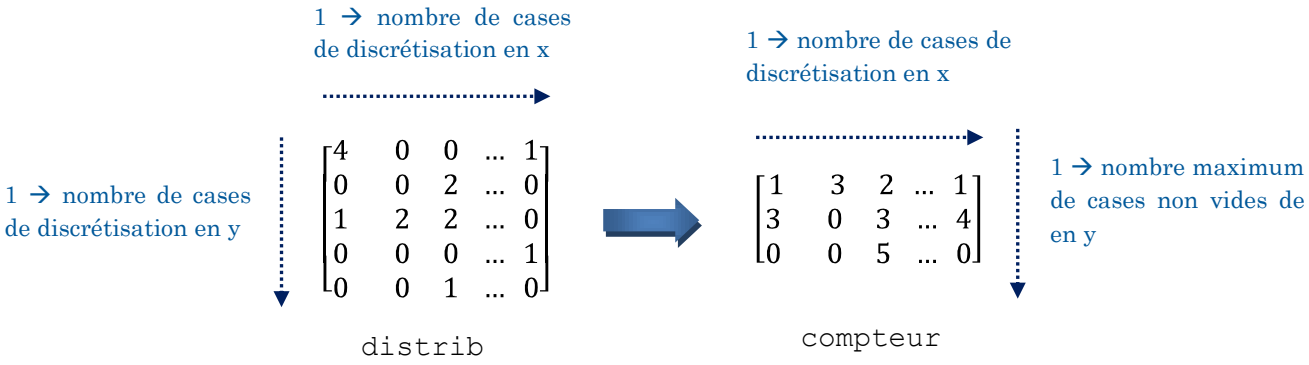


Figure 47 : Création de la variable `compteur`

Cette phase est codée comme suit :

```
function donnees_aberrantes=
identifier_anomalies(distrib,cumul_haut,transfor,pasx,pasy,bornesx,bornesy)

% A. chercher dans la matrice de "distrib" les valeurs non nulles. Ces valeurs non nulles
correspondent à des carres contenant des données

% 1. critère de distance: matrice "compteur"

compteur=zeros(1,pasx+1);
poids=compteur;
poids_transfor=compteur;
for j=2:size(distrib,2);
nb_compteur=1;
    for i=1:size(distrib,1)
        if distrib(i,j)~=0
            compteur(nb_compteur,j-1)=distrib(i,1); % critére de distance
            nb_compteur=nb_compteur+1;
        end
    end
end
```

Au cours de la deuxième étape de cette fonction, on identifie les anomalies présentes dans l'histogramme. Nous avons défini une anomalie comme une discontinuité dans la distribution des mesures, c'est-à-dire que l'histogramme est composé d'au moins deux ensembles distincts, séparés par un certain nombre d'intervalles vides. Nous allons donc compter le nombre de cases vides parmi les cases contenant des mesures. Pour cela, on utilise la variable `compteur` pour créer une variable temporaire `a`, définit comme suit :

```
a=compteur(nb+1,j)-compteur(nb,j)
```

Ensuite, à partir de cette variable temporaire on calcule le `nb_carres`, défini par `nb_carres=a-1`. Nous pouvons trouver trois cas différents en fonction des valeurs de `a`:

(1) `a=0-3=-3` → `a<1` → `nb_carres=0`

(2) `a=2-1=1` → `a=1` → `nb_carres=0`

(3) `a=3-1=2` → `a>1` → `nb_carres=2-1=1`

Nous nous intéressons uniquement au troisième cas. On considère qu'il y une discontinuité lorsqu'il y au moins un carré vide, c'est-à-dire `a>1`. Par exemple :

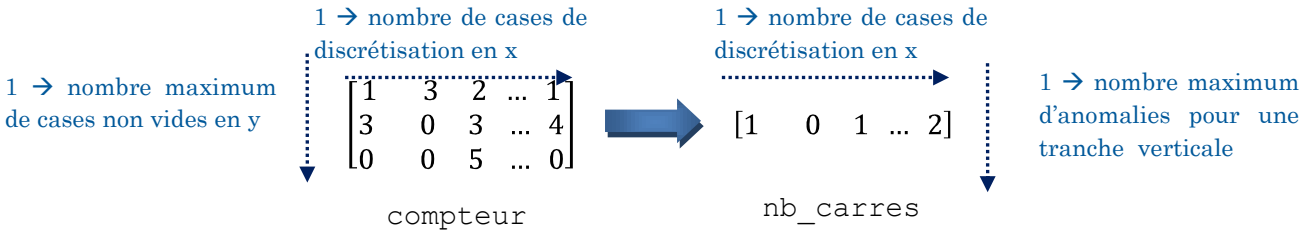


Figure 48 : Création de la variable `nb_carres`

Nous allons également chercher le poids des données aberrantes dans les matrices `cumul_haut` et `transfor` calculées lors des étapes précédentes. Pour cela, on doit identifier dans l'histogramme les cases vides parmi les cases contenant des données. Dans ce but, on combine la variable `compteur`, qui indique les cases de discrétisation ayant des données, avec la variable temporaire `a` : la condition `a>1` indique qu'il existe une anomalie. Si `a>1`, le numéro de ligne qui présente une case suspecte est égale à `compteur+1`, Par exemple :

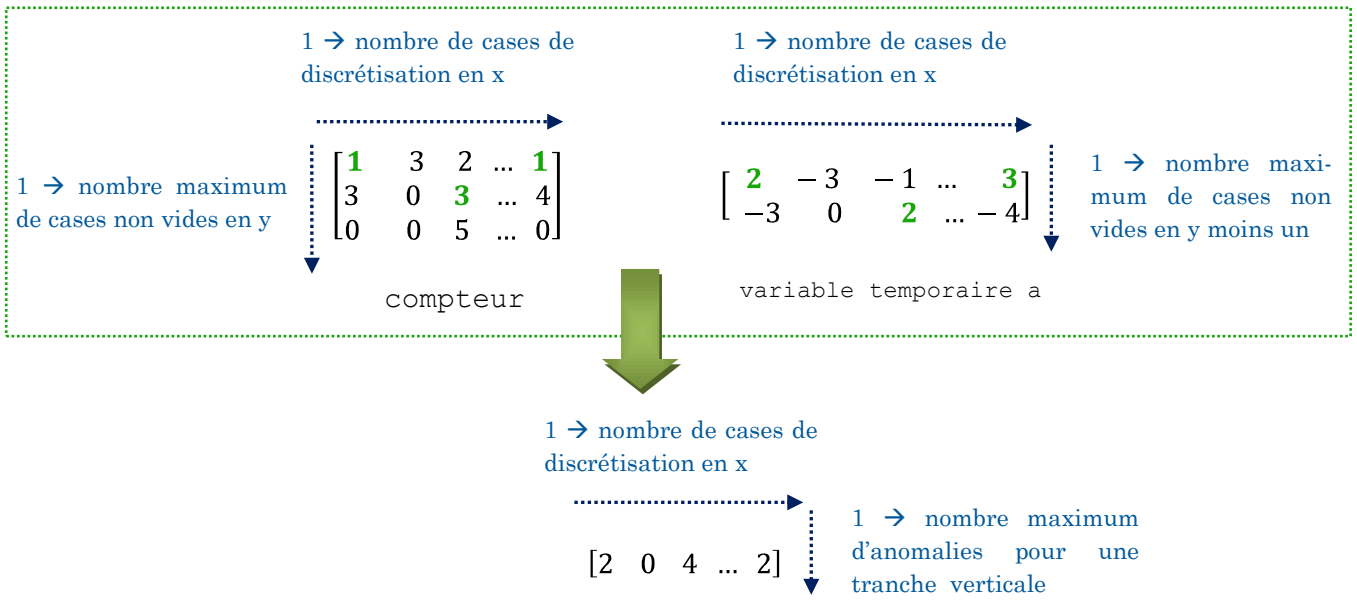


Figure 49 : Identification des cases vides parmi les cases contenant des données

Ensuite, on crée les variables `matrice_poids` et `matrice_cumul`, qui représentent le poids pour l'ensemble des anomalies. Par exemple :

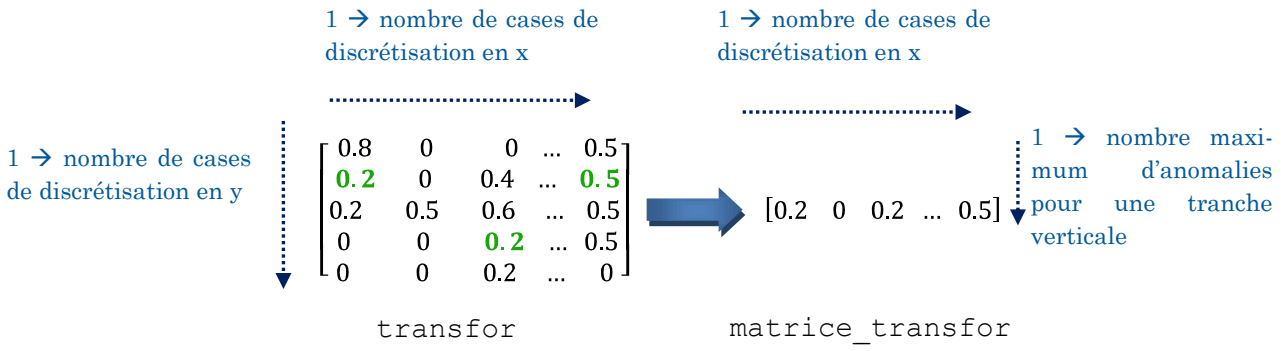


Figure 50 : Création de la variable `matrice_cumul`

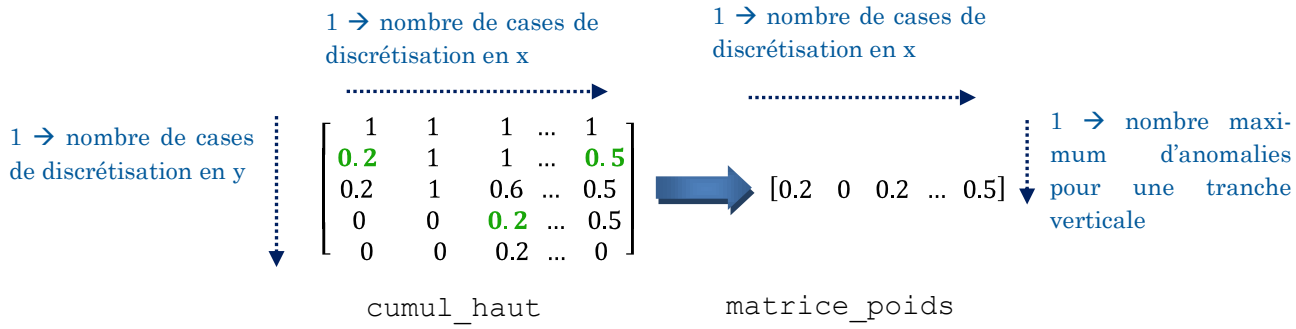


Figure 51 : Création de la variable matrice_poids

Au cours de cette phase, nous définissons aussi la variable `carre_nb_intervalley_min`, qui permettra à terme d'identifier la borne inférieure de la case de discrétisation `y` à laquelle la mesure appartient. Cette variable est actuellement égale à `compteur`. Cela veut dire qu'elle indique le numéro de la dernière case de discrétisation contenant des mesures avant l'anomalie. Cette variable est modifiée lors de la prochaine phase afin qu'elle indique la vraie valeur de la borne.

Voici le code :

```
% initialiser les matrices
nb_carres=zeros(1,pasx+1);              % 1. matrice de distance
matrice_transfor=nb_carres;             % 2. matrice de poids transformée
matrice_poids=nb_carres;                % 3. matrice de poids
carre_nb_intervallex_min=nb_carres;     % matrice d'intervalles x min
carre_nb_intervallex_max=nb_carres;     % matrice d'intervalles x max
carre_nb_intervalley_min=nb_carres;     % matrice d'intervalles y min


% B. chercher dans la matrice "compteur" les valeurs aberrantes. On considère qu'une va-
leur est aberrante à partir d'un carré de distance

% 1. comptabiliser le nombre des carrés vides entre les carrés avec des
% points [nb_carres]
% 2. creer la matrice de poids: elle représente le poids des données
% aberrantes [matrice_transfor]
% 3. identifier les intervalles min y [carre_nb_intervalley_min]
```

```
for j=1:size(compteur,2);
indice=1;
    for nb=1:size(compteur,1)-1
        a=compteur(nb+1,j)-compteur(nb,j);
            if a>1 % il y a plus d'un carré de distance
                nb_carres(indice,j)=a-1;
                matrice_poids(indice,j)=cumul_haut(compteur(nb,j)+1,j+1);
                matrice_transfor(indice,j)=transfor(compteur(nb,j)+1,j+1);
                carre_nb_intervalley_min(indice,j)= compteur(nb,j);
                indice=indice+1;
            end
    end
end

carre_nb_intervalley_max=zeros(size(carre_nb_intervalley_min,1),size(carre_nb_intervalley
_min,2));
```

Au cours de la troisième étape de calcul, la fonction identifie les intervalles auxquels les données aberrantes appartiennent. Pour cela, on crée les 4 variables suivantes :

- `carre_nb_intervallex_min` : indique la borne inférieure de la tranche verticale $[x_i, x_{i+1}[$;
- `carre_nb_intervallex_max` : indique la borne supérieure de la tranche verticale $[x_i, x_{i+1}[$;
- `carre_nb_intervalley_min` : indique la borne inférieure de l'intervalle $[y_j, y_{j+1}[$;
- `carre_nb_intervalley_max` : indique la borne supérieure de l'intervalle $[y_j, y_{j+1}[$.

La fonction cherche la valeur de `carre_nb_intervalle_xmin` et `carre_nb_intervalle_xmax` dans la variable `bornesx` définie précédemment, ainsi que la valeur de `carre_nb_intervalle_ymin` et `carre_nb_intervalle_ymax` dans la variable `bornesy`. Par exemple :

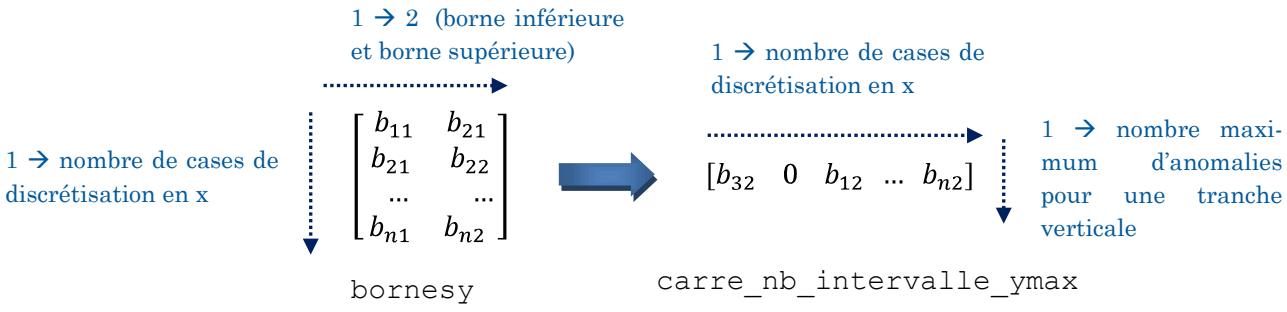
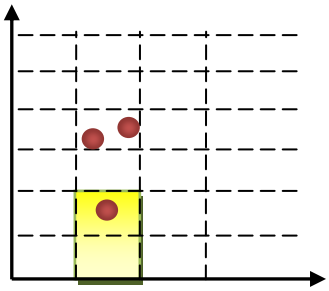


Figure 52 : Création de la variable `carre_nb_intervalle_ymax`

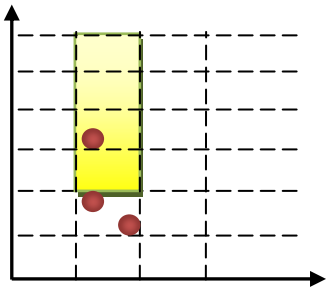
La recherche de la valeur des bornes en abscisse est directe : si par exemple on se situe dans la troisième colonne de `carre_nb_intervallex_min (j=3)`, la borne inférieure sera b_{31} et la supérieure b_{32}. Par contre, pour la recherche de la valeur des bornes en ordonnées, il faut tenir compte de la direction. On peut donc distinguer trois cas :

`Cas n°1` : du centre vers le bas



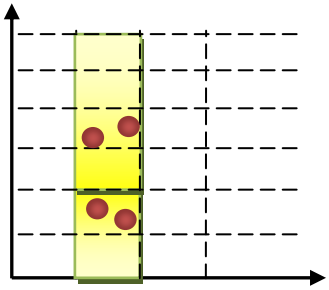
La borne inférieure est égale à b_{11} et la borne supérieure à b_{22}. Dit autrement, `carre_nb_intervalle_ymax` est égal à la borne supérieure de la dernière case avant l'anomalie.

`Cas n°2` : du centre vers le haut



La borne inférieure est égale à b_{31} et la borne supérieure à b_{55}. Dit autrement, `carre_nb_intervalle_ymin` est égal à la borne inférieure de la première case suspecte. Nous aurions pu être plus exacts dans le calcul de cette variable : elle aurait pu être égale à b_{41} comme le montre la figure à gauche. Toutefois, la programmation est plus simple de cette façon et les résultats sont toujours corrects car l'intervalle en `y` comprend toujours la variable suspecte.

`Cas n°3` : du centre vers le bas et vers le haut



Dans ce cas, on considère deux intervalles car il existe deux ensembles des données suspectes.

Figure 53 : Création de `carre_nb_intervallex_min` et `carre_nb_intervallex_max`

Voici le code :

```
% C. identifier les bornes d'intervalles correspondants aux données aberrantes
for j=1:size(nb_carres,2)
    for indice=1:size(nb_carres,1)
        a=carre_nb_intervalley_min(indice,j);
        if a>0
            carre_nb_intervallex_min(indice,j)=bornesx(j,1);
            carre_nb_intervallex_max(indice,j)=bornesx(j,2);
            carre_nb_intervalley_min(indice,j)=bornesy(a,1);
            carre_nb_intervalley_max(indice,j)=bornesy(a,2);

            % transformer les intervalles de y
            % 1. si matrice_poids=matrice_transfor et la valeur > 0.5, il faut
```

```
            % regarder les deux sens
            % 2. si matrice_poids est inférieur à 0.5 on regarde la matrice en
              direction du bas au centre
            % 3. si matrice_poids est supérieur à 0.5 on regarde la matrice en
              direction du haût au centre

            if (matrice_poids(indice,j)>=0.5 & matrice_transfor(indice,j)>=0.5 &
           matrice_poids(indice,j)== matrice_transfor(indice,j))
                carre_nb_intervalley_min(indice,j)=bornesy(a+1,2);
                carre_nb_intervalley_max(indice,j)=bornesy(pasy+1,2);

            elseif (matrice_poids(indice,j)>0.5 & matrice_transfor(indice,j)<0.5
                carre_nb_intervalley_min(indice,j)=bornesy(1,1);

            elseif (matrice_poids(indice,j)==0.5 & matrice_transfor(indice,j)>0.5)
                carre_nb_intervalley_min(indice,j)=bornesy(a+1,2);
                carre_nb_intervalley_max(indice,j)=bornesy(pasy+1,2);

            elseif (matrice_poids(indice,j)==0.5 & matrice_transfor(indice,j)<0.5)
                carre_nb_intervalley_min(indice,j)=bornesy(1,1);

            elseif matrice_poids(indice,j)<0.5
                carre_nb_intervalley_min(indice,j)=bornesy(a+1,2);
                carre_nb_intervalley_max(indice,j)=bornesy(pasy+1,2);

            end

        else
            carre_nb_intervallex_min(indice,j)=0;
            carre_nb_intervallex_max(indice,j)=0;
            carre_nb_intervalley_min(indice,j)=0;
            carre_nb_intervalley_max(indice,j)=0;
        end
    end
end
```

Au cours de la quatrième phase de calcul, la fonction `identifier_anomalies` crée la variable `donnees_aberrantes`. A cette stade, cette variable est une matrice composée de six colonnes et elle renseigne sur :

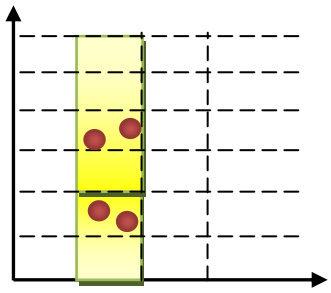
1. la borne inférieure de l'intervalle de discrétisation en ordonnées auquel la mesure appartient ;
2. la borne supérieure de l'intervalle de discrétisation en ordonnées auquel la mesure appartient ;
3. la borne inférieure de l'intervalle de discrétisation en abscisses auquel la mesure appartient ;
4. la borne supérieure de l'intervalle de discrétisation en abscisses auquel la mesure appartient ;
5. le nombre des cases de discrétisation vides constituant la discontinuité dans l'histogramme ;
6. le poids des données suspectes ;

Le calcul de cette variable est codé comme suit :

```
% 4. créer une matrice [ (intervalle_minx) (intervalle_maxx) (intervalle_miny)
% (intervalle_maxy) (nb_carres) (poids)]

donnees_aberrantes=zeros(1,6);
indice=1;
for j=1:size(nb_carres,2)
    for i=1:size(nb_carres,1)
        if nb_carres(i,j)~=0
            donnees_aberrantes(indice,1)=carre_nb_intervallex_min(i,j);
            donnees_aberrantes(indice,2)=carre_nb_intervallex_max(i,j);
            donnees_aberrantes(indice,3)=carre_nb_intervalley_min(i,j);
            donnees_aberrantes(indice,4)=carre_nb_intervalley_max(i,j);
            donnees_aberrantes(indice,5)=nb_carres(i,j);
            donnees_aberrantes(indice,6)=matrice_transfor(i,j);
            indice=indice+1;
        end
    end
end
```

Enfin, on doit considérer le cas particulier où le poids des données suspectes est égal à `0.5`. Nous savons qu'il y a une anomalie mais, on ne peut pas identifier quelles sont les données aberrantes.



En sortie, on aura autant de lignes que d'ensembles de données suspectes. Dans cet exemple, il y aura deux lignes. En outre, cet algorithme permet d'identifier l'intervalle $[y_j, y_{j+1}[$ auquel appartient chaque ensemble de données suspectes.

Voici le code :

```
% Ajouter une nouvelle ligne si le poids = 0.5 et % modifier les
% intervalles ymin et ymax

for indice=1:size(donnees_aberrantes,1)
    numero=1;%il compte le nombre des lignes qu'il faut ajouter à cause des valeurs=0.5
avec des nom des séries différents

    if (donnees_aberrantes(indice,6)==0.5)
        donnees_aberrantes(numero+size(donnees_aberrantes,1),1:6)=
         donnees_aberrantes(indice,1:6);
        numero=numero+1;

        if round(donnees_aberrantes(size(donnees_aberrantes,1),3)
         *1000000)/1000000==round(bornesy(1,1)*1000000)/1000000
            donnees_aberrantes(size(donnees_aberrantes,1),3)=
              donnees_aberrantes(indice,4);
            donnees_aberrantes(size(donnees_aberrantes,1),4)=bornesy(pasy+1,2);
        elseif round(donnees_aberrantes(size(donnees_aberrantes,1),4)
         *1000000)/1000000==round(bornesy(pasy+1,2)*1000000)/1000000;
            donnees_aberrantes(size(donnees_aberrantes,1),3)=bornesy(1,1);
            donnees_aberrantes(size(donnees_aberrantes,1),4)=
              donnees_aberrantes(indice,3);
```

```
        end

    end

end
```

Enfin, la fonction `calcul_ligne` permet de chercher le numéro de la ligne de la base de données originale présentant les données identifiées comme aberrantes. En outre, cette fonction identifie également le nom de la série à laquelle appartiennent les données suspectes. Dans ce but, la fonction reçoit en entrée :

- `donnees_aberrantes` : la variable que l'on vient de créer grâce à la fonction `identifier_anomalies` ;
- `x` : la valeur moyenne de la mesure en abscisses ;
- `incertitude_ymin` : la valeur minimale de la mesure en ordonnées ;
- `incertitude_ymax` : la valeur maximale de la mesure en ordonnées ;
- `N` : la taille du vecteur `x`, c'est-à-dire, le nombre des mesures ;
- `Serie` : cette variable renseigne sur le nom de la série de l'ensemble des mesures.

En sortie, on obtient :

- `nom_serie` : le nom de la série présentant des données suspectes ;
- `données_aberrantes` : la variable d'entrée complété du numéro de la ligne présentant des données suspectes.

La fonction `identifier_anomalies` nous a permis d'identifier les intervalles en ordonnées et en abscisses comportant des données suspectes. La fonction `calcul_ligne` cherche la ou les mesures comprises entre la borne inférieure et supérieure de ces intervalles et présentes en sortie, le numéro de la ligne pour chaque donnée identifiée comme aberrante. De plus, elle renseigne sur le nom de la série de ces données.

Enfin, il faut tenir compte du cas particulier où l'ensemble des données partagent la même valeur en abscisses. L'algorithme reste analogue mais adapté : la valeur de x est identique à celle de l'intervalle. La fonction est codée comme suit :

```
function [nom_serie donnees_aberrantes]=
calcul_ligne(donnees_aberrantes,x,incertitude_ymin,incertitude_ymax,N,serie)

if min(x)==max(x)

    for indice=1:size(donnees_aberrantes,1)
        compteur=1;% il compte le nombre des données aberrantes
        for i=1:N

            if ((donnees_aberrantes(indice,1)==x(i))&
```

```
              (donnees_aberrantes(indice,3)<=incertitude_ymin(i)) &
              (donnees_aberrantes(indice,4)>incertitude_ymax(i)))

                 donnees_aberrantes(indice,7)=compteur;
                 donnees_aberrantes(indice,7+compteur)=i+1;
                 nom_serie(indice)=serie(i);
                 compteur=compteur+1;
             end
         end
     end

else

     for indice=1:size(donnees_aberrantes,1)
         compteur=1;% il compte le nombre des données aberrantes
         for i=1:N

             if((donnees_aberrantes(indice,1)<=x(i))& donnees_aberrantes(indice,2)>x(i)) &
               (donnees_aberrantes(indice,3)<=incertitude_ymin(i)) &
               (donnees_aberrantes(indice,4)>incertitude_ymax(i)))

                 donnees_aberrantes(indice,7)=compteur;
                 donnees_aberrantes(indice,7+compteur)=i+1;
                 nom_serie(indice)=serie(i);
                 compteur=compteur+1;
             end
         end
     end
end
```

La partie suivante du code permet d'ajouter une ligne dans le tableau des résultats pour chaque série identifiée dans une anomalie. Par exemple, si une anomalie présente 5 mesures dont 4 appartiennent à des séries différentes, le tableau des résultats sera composé par 4 lignes (une pour chaque série).

```
% Supprimer tous les répétitions
new_name=nom_serie(1:end,1);
compteur(1:size(donnees_aberrantes,1))=1;
for indice=1:size(donnees_aberrantes,1)
    for i=1:size(nom_serie,2)-1
        if (isempty(nom_serie{indice,i})==0 & isempty(nom_serie{indice,i+1})==0 & ise-
qual(nom_serie{indice,i},nom_serie{indice,i+1})==0)
            compteur(indice)=compteur(indice)+1;
            new_name(indice,compteur(indice))=nom_serie(indice,i+1);
        end
    end
end
nom_serie=new_name;
clear new_name;
clear compteur;

% Calculer le nombre des données aberrantes et le numéro de la ligne sur la
% base des données originale

taille=size(donnees_aberrantes,1);
numero(1:size(nom_serie,1),1:size(nom_serie,2))=0;
ajouter_ligne=0;
```

```
if min(x)==max(x)

    for indice=1:size(donnees_aberrantes,1)
        compteur(1:size(nom_serie,2))=1;% il compte le nombre des données aberrantes pour
chaque série indentifiée comme suspecte
        for i=1:N

            if ((donnees_aberrantes(indice,1)==x(i))& (don-
nees_aberrantes(indice,3)<=incertitude_ymin(i)) & (don-
nees_aberrantes(indice,4)>incertitude_ymax(i)))

                for j=1:size(nom_serie,2)
                    if (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 & j==1)
                        donnees_aberrantes(indice,7)=compteur(1);
                        donnees_aberrantes(indice,7+compteur(1))=i+1;
                        compteur(1)=compteur(1)+1;
                    elseif (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 )
                        if numero(indice,j)==0
                            % Créer des nouvelles lignes
                            numero(indice,j)=numero(indice,j)+1;
                            ajouter_ligne=ajouter_ligne+1;
                            don-
nees_aberrantes(ajouter_ligne+taille,1:6)=donnees_aberrantes(indice,1:6);
                        end
                        donnees_aberrantes(ajouter_ligne+taille,7)=compteur(j);
                        donnees_aberrantes(ajouter_ligne+taille,7+compteur(j))=i+1;
                        compteur(j)=compteur(j)+1;
                    end
                end

            elseif ((donnees_aberrantes(indice,1)==x(i))& (don-
nees_aberrantes(indice,4)==incertitude_ymax(i)))

                for j=1:size(nom_serie,2)
                    if (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 & j==1)
                        donnees_aberrantes(indice,7)=compteur(1);
                        donnees_aberrantes(indice,7+compteur(1))=i+1;
                        compteur(1)=compteur(1)+1;
                    elseif (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 )
                        if numero(indice,j)==0
                            % Créer des nouvelles lignes
                            numero(indice,j)=numero(indice,j)+1;
                            ajouter_ligne=ajouter_ligne+1;
                            don-
nees_aberrantes(ajouter_ligne+taille,1:6)=donnees_aberrantes(indice,1:6);
                        end
                        donnees_aberrantes(ajouter_ligne+taille,7)=compteur(j);
                        donnees_aberrantes(ajouter_ligne+taille,7+compteur(j))=i+1;
                        compteur(j)=compteur(j)+1;
                    end
                end

            end
        end
    end
```

```
else

    for indice=1:size(donnees_aberrantes,1)
        compteur(1:size(nom_serie,2))=1;% il compte le nombre des données aberrantes pour
chaque série indentifiée comme suspecte
        for i=1:N

            if ((donnees_aberrantes(indice,1)<=x(i)) & (don-
nees_aberrantes(indice,2)>x(i)) & (donnees_aberrantes(indice,3)<=incertitude_ymin(i)) &
(donnees_aberrantes(indice,4)>incertitude_ymax(i)))

                for j=1:size(nom_serie,2)
                    if (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 & j==1)
                        donnees_aberrantes(indice,7)=compteur(1);
                        donnees_aberrantes(indice,7+compteur(1))=i+1;
                        compteur(1)=compteur(1)+1;
                    elseif (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 )

                        if numero(indice,j)==0
                            % Créer des nouvelles lignes
                            numero(indice,j)=numero(indice,j)+1;
                            ajouter_ligne=ajouter_ligne+1;
                            don-
nees_aberrantes(ajouter_ligne+taille,1:6)=donnees_aberrantes(indice,1:6);
                        end
                        donnees_aberrantes(ajouter_ligne+taille,7)=compteur(j);
                        donnees_aberrantes(ajouter_ligne+taille,7+compteur(j))=i+1;
                        compteur(j)=compteur(j)+1;
                    end
                end

            elseif ((donnees_aberrantes(indice,1)<=x(i)) & (don-
nees_aberrantes(indice,2)>x(i)) & (donnees_aberrantes(indice,4)==incertitude_ymax(i)))

                for j=1:size(nom_serie,2)
                    if (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 & j==1)
                        donnees_aberrantes(indice,7)=compteur(1);
                        donnees_aberrantes(indice,7+compteur(1))=i+1;
                        compteur(1)=compteur(1)+1;
                    elseif (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 )

                        if numero(indice,j)==0
                            % Créer des nouvelles lignes
                            numero(indice,j)=numero(indice,j)+1;
                            ajouter_ligne=ajouter_ligne+1;
                            don-
nees_aberrantes(ajouter_ligne+taille,1:6)=donnees_aberrantes(indice,1:6);
                        end
                        donnees_aberrantes(ajouter_ligne+taille,7)=compteur(j);
                        donnees_aberrantes(ajouter_ligne+taille,7+compteur(j))=i+1;
                        compteur(j)=compteur(j)+1;
                    end
                end
```

```
            elseif ((donnees_aberrantes(indice,2)==x(i)) & (don-
nees_aberrantes(indice,3)<=incertitude_ymin(i)) & (don-
nees_aberrantes(indice,4)>incertitude_ymax(i)))

                for j=1:size(nom_serie,2)
                    if (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 & j==1)
                        donnees_aberrantes(indice,7)=compteur(1);
                        donnees_aberrantes(indice,7+compteur(1))=i+1;
                        compteur(1)=compteur(1)+1;
                    elseif (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 )

                        if numero(indice,j)==0
                            % Créer des nouvelles lignes
                            numero(indice,j)=numero(indice,j)+1;
                            ajouter_ligne=ajouter_ligne+1;
                            don-
nees_aberrantes(ajouter_ligne+taille,1:6)=donnees_aberrantes(indice,1:6);
                        end
                        donnees_aberrantes(ajouter_ligne+taille,7)=compteur(j);
                        donnees_aberrantes(ajouter_ligne+taille,7+compteur(j))=i+1;
                        compteur(j)=compteur(j)+1;
                    end
                end

            elseif ((donnees_aberrantes(indice,2)==x(i)) & (don-
nees_aberrantes(indice,4)==incertitude_ymax(i)))

                for j=1:size(nom_serie,2)
                    if (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 & j==1)
                        donnees_aberrantes(indice,7)=compteur(1);
                        donnees_aberrantes(indice,7+compteur(1))=i+1;
                        compteur(1)=compteur(1)+1;
                    elseif (isempty(nom_serie{indice,j})==0 & ise-
qual(serie{i},nom_serie{indice,j})==1 )

                        if numero(indice,j)==0
                            % Créer des nouvelles lignes
                            numero(indice,j)=numero(indice,j)+1;
                            ajouter_ligne=ajouter_ligne+1;
                            don-
nees_aberrantes(ajouter_ligne+taille,1:6)=donnees_aberrantes(indice,1:6);
                        end
                        donnees_aberrantes(ajouter_ligne+taille,7)=compteur(j);
                        donnees_aberrantes(ajouter_ligne+taille,7+compteur(j))=i+1;
                        compteur(j)=compteur(j)+1;
                    end
                end


            end
        end
    end

end
```

Cette partie du code est utile lorsque le logiciel tourne deux fois. Elle permet d'identifier dans la base de données originale le numéro de la ligne qui présente les données aberrantes.

```
% Si nous avons supprimé une ligne lors de la première fois que nous avons
% fait tourner le logiciel pour cette réaction, on la tient en compte
% maintenant pour le calcul du numéro de la ligne

for i=1:size(donnees_aberrantes,1)
    for j=8:size(donnees_aberrantes,2)
        for indice=1:size(index,1)
            if donnees_aberrantes(i,j)>=index(indice)
                donnees_aberrantes(i,j)=donnees_aberrantes(i,j)+1;
            end
        end
    end
end


compteur=0;

for j=1:size(nom_serie,2)
    for i=1:size(nom_serie,1)
        if isempty(nom_serie{i,j})==0
            compteur=compteur+1;
            name_serie(compteur)=nom_serie(i,j);
        end
    end
end
name_serie=name_serie';
nom_serie=name_serie;
clear name_serie;
```

La dernière partie de la fonction permet de trier les résultats, nommés `donnees_aberrantes`, en fonction du nombre de carrés vides. Plus sa valeur est élevée, plus la donnée suspecte sera considérée comme aberrante. Toutefois, cet algorithme est gourmand en temps de calcul ; nous avons donc limité le tri aux cas où le nombre de lignes ayant des données aberrantes est inférieur à 15. Pour les cas dépassant ce seuil, les résultats ne seront pas triés. Il serait intéressant de tester le temps de calcul de cet algorithme dans les outils de l'AEN.

```
Nombre=size(donnees_aberrantes,1);

if (Nombre>1 & Nombre<15)
    while compteur<factorial(Nombre)/factorial(2);
        for i=1:size(donnees_aberrantes,1)-1
            if donnees_aberrantes(i,5)<donnees_aberrantes(i+1,5)
                %trie nom_serie
                nom=nom_serie(i,1);
                nom_serie(i,1)=nom_serie(i+1,1);
                nom_serie(i+1,1)=nom;

                %trie données_aberrantes
                data=donnees_aberrantes(i,1:end);
                donnees_aberrantes(i,1:end)=donnees_aberrantes(i+1,1:end);
                donnees_aberrantes(i+1,1:end)=data;
```

```
            end
            compteur=compteur+1;
        end
    end
end
```

5. Automatisation des calculs

L'objectif de ce module est de permettre d'automatiser les phases d'importation, de calcul et d'exportation des données. L'utilisateur doit uniquement indiquer les réactions qu'il souhaite analyser et le logiciel réalise les opérations nécessaires.

Pour cela, nous avons développé la fonction `automatisation`, qui permet d'importer les fichiers des données. Ensuite, cette fonction fait appel à la fonction principale `Détection_anomalies_main` , permettant l'identification de données suspectes.

Voici le code :

```
% Automatisation-détection d'anomalies
clear;

source='fichier_csv\automatisation.xls';
[data, name]=xlsread(source,1);
taille_data=size(name);
avant='fichier_csv\';
apres='.csv';
nom_reaction=name(2:taille_data(1,1),2);
nom_reaction=strcat(avant,nom_reaction,apres);
nom_feuille=1;
nb_repetitions=0;

for nb_reaction=1:125
    if nb_reaction>1
        clearex ('nom_reaction','nb_reaction','nom_feuille','nb_repetitions');
    end

    filename=nom_reaction(nb_reaction);
    filename=sprintf('%s',filename{:}); % convert cell in string

    % importer le fichier data
    format long;

    % lire csv files
    % importer numbers
    delimiter=';';
    donnee=dlmread(filename,delimiter,1,2);
    index=10^20;
    repetition=0;
    [don-
nees_aberrantes,nom_serie,name_serie_aberrante,nom_feuille]=Detection_anomalies_main(file
name,nb_reaction,nom_feuille,donnee, index, nb_repetitions,repetition );
```

```
    %----------Recherche une nouvelle fois des anomalies-----------------
    for indice=1:size(donnees_aberrantes,1)
        if (donnees_aberrantes(indice,5)>15 & max(name_serie_aberrante)<2)

        % nombre de répétitions, indiquer s'il s'agit de la deuxième fois
        nb_repetitions=nb_repetitions+1;
        repetition=1;
        % identifier les lignes à supprimer
        compteur=0;
            for k=1:size(donnees_aberrantes,1)
                for j=8:size(donnees_aberrantes,2)
                    if donnees_aberrantes(k,7)>0
                        compteur=compteur+1;
                        index(compteur)=donnees_aberrantes(k,j)-1;
                    else
                        compteur=1;
                        index(compteur)=0;
                    end
                end
            end

            % supprimer la donnée aberrante pour laquelle le nb_carre>10
            new_donnee=zeros(1,size(donnee,2));
            indice=0;
            for i=1:size(donnee,1)
                for j=1:compteur
                    if (index(j)~=i)
                        indice=indice+1;
                        new_donnee(indice,1:size(donnee,2))=donnee(i,1:size(donnee,2));
                    end
                end
            end

            donnee=new_donnee;

            % et faire tourner une deuxième fois
            nom_feuille=nom_feuille+1;
            [don-
nees_aberrantes,nom_serie,name_serie_aberrante,nom_feuille]=Detection_anomalies_main(file
name,nb_reaction,nom_feuille,donnee,index,nb_repetitions, repetition);
        end
    end
    nom_feuille=nom_feuille+1;
end
```

6. Traitement des fichiers lourds (>1000 Ko)

Le temps de calcul pour ces fichiers peut être très élevé (parfois plus de 24 heures). Afin d'accélérer ce processus, la représentation graphique de la réaction avec la prise en compte de l'incertitude a été supprimée. En outre, la recherche du nom de la série et de l'indicateur « nombre de fois que la série est identifiée comme aberrantes » a été également supprimée. Ces deux informations, même si elles sont intéressantes, ne sont pas indispensables pour la détection des données aberrantes et ces suppressions nous permettent de diviser le temps de calcul par 20.

Le code utilisé pour traiter ces fichiers est identique, seulement la fonction `automatisation` et `detection_anomalies_main` changent. A la place, le logiciel utilise les fonctions `automatisation_lourd` et `detection_anomlaies_main_lourd`. Ces fonctions seront également présentes dans le bon de livraison destiné à l'AEN.

VI. Sorties de l'outil

L'outil identifie la présence de données suspectes parmi les mesures d'une réaction donnée, et fournit, pour chaque anomalie détectée (ou pour chaque série dont les points font partie d'une anomalie), les indicateurs suivants :

- Le nombre d'intervalles vides délimitant la discontinuité dans la loi de probabilité. Plus ce nombre (supérieur ou égal à 1) est élevé, plus la mesure est suspecte ;
- Le nombre de points de mesure formant l'anomalie ;
- Le poids des données identifiées comme suspectes, par rapport à l'ensemble des données contenues dans la tranche verticale. Cette information, combinée avec le nombre de points, est intéressante car elle peut permettre d'identifier la nature de l'anomalie : si le poids des données suspectes et le nombre de points sont très faibles, il s'agira probablement d'un problème ponctuel. Par contre, un poids et un nombre de points élevés suggèrent un problème récurrent touchant plusieurs mesures;
- Le nom de la série et le nombre de fois que la série a été identifiée comme suspecte. Le principe de la méthode de détection ne change pas : le logiciel identifie les mesures qui font partie d'une anomalie et calcule leur poids. Toutefois, afin d'affiner cette recherche, il identifie aussi les séries qui font partie de l'anomalie. Le logiciel attribue donc une ligne pour chaque série de données identifiée comme suspecte. Cette façon de procéder permet aussi de calculer le nombre de fois que la série est identifiée comme suspecte. Plus ce dernier indicateur est élevé et plus la série a des chances d'être suspecte.

Ces indicateurs renseignent ainsi sur le degré de suspicion des points repérés.

Nous avons également développé un module de recherche qui permet de repérer les données suspectes dans la base de données originale (sous format Excel). Le logiciel identifie les intervalles sur l'axe x et sur l'axe y qui comprennent des données suspectes. Ensuite, il recherche sur la base de données originale les données comprises dans les intervalles suspects. En sortie, le logiciel indique le numéro de ligne de la base de données originale qui contient des données suspectes.

Le format des résultats est le suivant :

nom fichier	nom serie	nombre de fois série suspecte	inter-valle x minimum	inter-valle x maximum	inter-valle y minimum	inter-valle y maximum	nombre carrés	poids	nombre données aberrantes	numéro ligne

Tableau 3 : Format des résultats

Ensuite, les données doivent être triées en ordre décroissant en fonction de deux critères : la distance (nombre d'intervalles vides définissant une anomalie) et le nombre de fois que la série est identifiée comme suspecte. Les premières lignes du tableau de résultats présentent donc les données suspectes ayant la plus grande probabilité d'être aberrantes, laissant les « faux positifs » possibles à la fin de la liste.

VII. Résultats de la validation, limites et voies d'améliorations

La méthode a été validée à partir de l'échantillon de 113 cas représentatifs des anomalies existant dans la base de données. Après la mise en place et l'implémentation de la méthode, cet échantillon nous a permis de tester son efficacité sur des cas représentatifs de la base EXFOR, ainsi que sur des cas particuliers. Nous donnons ici les résultats de cette validation, ainsi que les ajustements nécessaires en raison des particularités présentes dans la base.

A. Résultats de la validation et limites

Le tableau ci-dessous présente les résultats de la validation. Nous avons considéré que les données suspectes sont aberrantes lorsque la distance ou le nombre de fois que la série apparait comme suspecte dépassait au moins le seuil de 2 (la valeur d'un de ces deux indicateurs doit être strictement supérieure à deux). Cette supposition fait suite à une analyse de sensibilité, au cours de laquelle nous avons testé une discrétisation à 19 intervalles.

Type	taille	Nb fichiers	Nbre cas aberrants	Cas de nuage de points	Nbre total d'anomalies	Nbre de faux positifs
CS cas représentatifs	<500 ko	50	15	5	118	6
CS cas représentatifs	>500 ko	5	0	1	0	2
CS cas difficiles	<500 ko	50	23	0	86	0
CS cas difficiles	>500 ko	8	3	0	3	0
Total	***nombre***	***113***	***41***	***6***	***207***	***8***
	%		***36%***	***5.3%***		***3.9%***

Tableau 4 : Résultats de la validation

Nous avons ainsi identifié 41 cas comportant des anomalies aberrants parmi les 113 cas test réalisés, soit 36% des deux échantillons réunis. Cependant, ce pourcentage est plus faible pour les CS représentatifs (27.3%) que pour les CS cas difficiles (44.8%).

Le nombre de faux positifs doit être le plus faible possible, notamment sur l'échantillon représentatif de la base totale : il permet de prévoir si la méthode sera efficace sur toute la base EXFOR. Celui-ci est estimé à 3.9% sur le nombre total d'anomalies repérées, et 6.7% sur l'échantillon représentatif de la base, ce qui est faible. De plus, 7 cas sur ces 8 identifiés comme faux positifs ne constituent pas des anomalies importantes (la distance n'est égale qu'à 3 intervalles vides). En fait, seul le dernier cas constitue un « vrai » faux positif. Cette erreur provient d'une discrétisation trop fine pour ce cas, où les données sont peu nombreuses.

On remarque donc que le choix de la discrétisation des axes est un point essentiel : une discrétisation trop fine peut engendrer un nombre de faux positif important, alors qu'une discrétisation trop grossière peut « laisser passer » des anomalies. Un compromis doit donc être trouvé. Dans ce but, nous avons testé une discrétisation à 19 intervalles (20 bornes) pour les axes d'ordonnées et d'abscisses. Celle-ci fonctionne correctement dans la plupart des cas sauf pour les « nuages des points » (qui représentent 5% des cas traités), ou pour des cas composés d'un faible nombre de mesures.

En général, nous rencontrons cette configuration de la forme « nuage de points » lorsque le nombre de données et donc la plage occupée par celles-ci est limitée. En termes pratiques, cela signifie qu'on ne dispose que d'une certaine partie de la courbe, en « zoom ». La tendance peut alors être difficile à cerner, le nuage de points étant diffus. Ceci peut créer des faux positifs : le logiciel peut identifier des points comme étant des anomalies alors qu'ils ne le sont pas.

Pour ces cas particuliers, on peut chercher à optimiser la discrétisation des axes en fonction du nombre de points. Cependant, il n'existe pas de méthode a priori permettant de lier le nombre d'intervalles de discrétisation et le nombre de points. Il faut donc trouver cette relation dans la pratique et pour cela, une analyse de sensibilité s'impose.

Enfin, il ne faut pas oublier qu'il s'agit d'une méthode inspirée par la détection réalisée par l'œil humain : si un expert ne peut pas conclure, le logiciel n'y parviendra pas non plus. Néanmoins, ces cas sont peu nombreux sur la base EXFOR.

B. Ajustements nécessaires liées aux particularités de la base de données

La principale difficulté de l'étude tient à la grande diversité du type de réactions et du type d'anomalies : certains cas étaient prévisibles mais d'autres ont été identifiés au cours de la validation des résultats. La phase de validation a été la plus longue du projet et a entraîné de nombreuses adaptations du code. Ces différentes particularités sont expliquées par la suite.

- Les mesures peuvent se présenter alignées verticalement si elles partagent la même valeur de x. Le principe de la méthode reste inchangé mais il a fallu adapter le code à ce cas. Autrement, nous pourrions rencontrer des cas étranges dans les résultats ou des erreurs dans l'exécution du code.

- Il est courant de trouver des données à faible valeur avec des incertitudes très élevées. Cela peut affecter la configuration du graphique et donc, la détection d'anomalies ne sera pas efficace.

  Dans un premier temps, nous avons testé l'apport de la prise en compte des incertitudes dans la recherche de l'échelle optimale : l'implémentation de cette information supplémentaire ne permet pas d'améliorer la détection des anomalies, et augmente de manière significative le temps de calcul. La figure suivante illustre ce point :

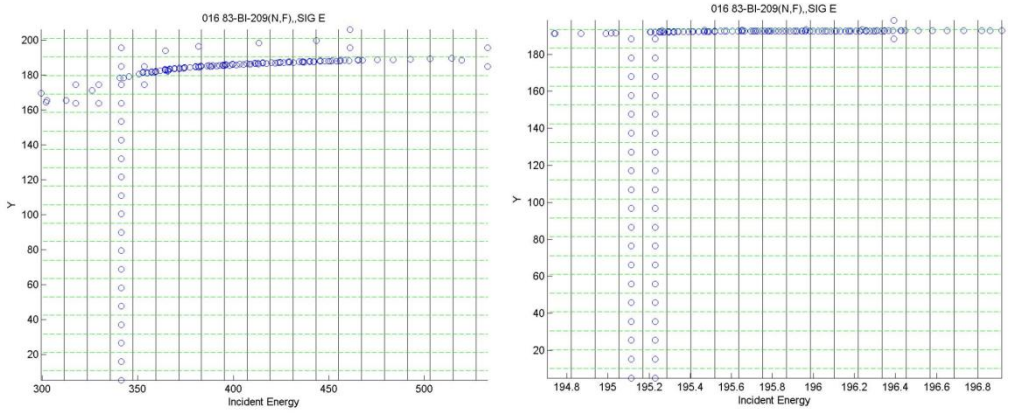


Figure 54 : Représentation de la réaction `016 83-BI-209(N,F),,SIG E` avant et après la prise en compte des incertitudes dans la recherche de l'échelle optimale

Afin de pallier cette difficulté, nous avons finalement limité l'étendue de l'incertitude : la prise en compte des incertitudes nous permet de ne pas identifier comme suspectes des données qui ne le sont pas du fait de leur incertitude. Le fait que l'incertitude d'une donnée soit tellement élevée qu'elle arrive à des niveaux où il n'y a pas d'autres valeurs ne nous apporte pas d'information supplémentaire. Nous avons donc limité le nombre d'intervalles de discrétisation sur lesquelles la donnée peut s'étaler.

Les exemples suivants montrent l'intérêt de cette limitation ; la figure de gauche représente la réaction avant limitation de l'étendue de l'incertitude, celle de droite après limitation :

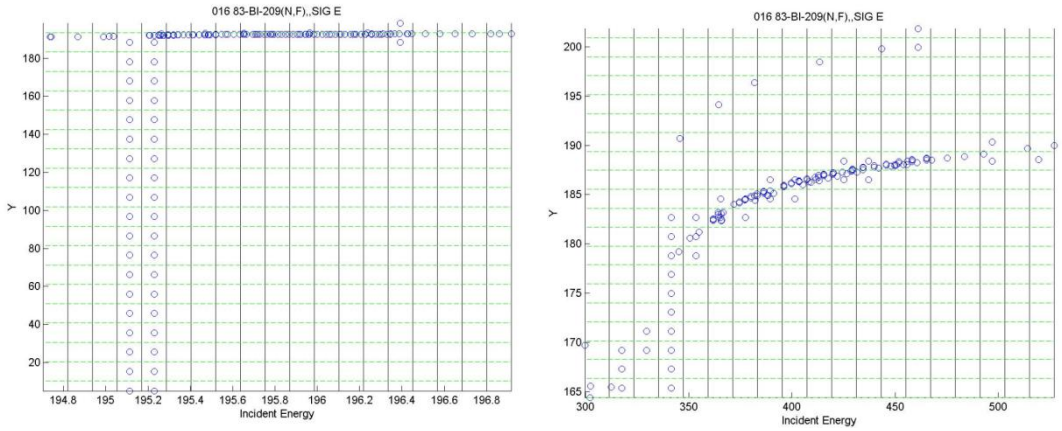


Figure 55 : Représentation de la réaction `016 83-BI-209(N,F),,SIG E` avant et après la limitation de l'étendue de l'incertitude

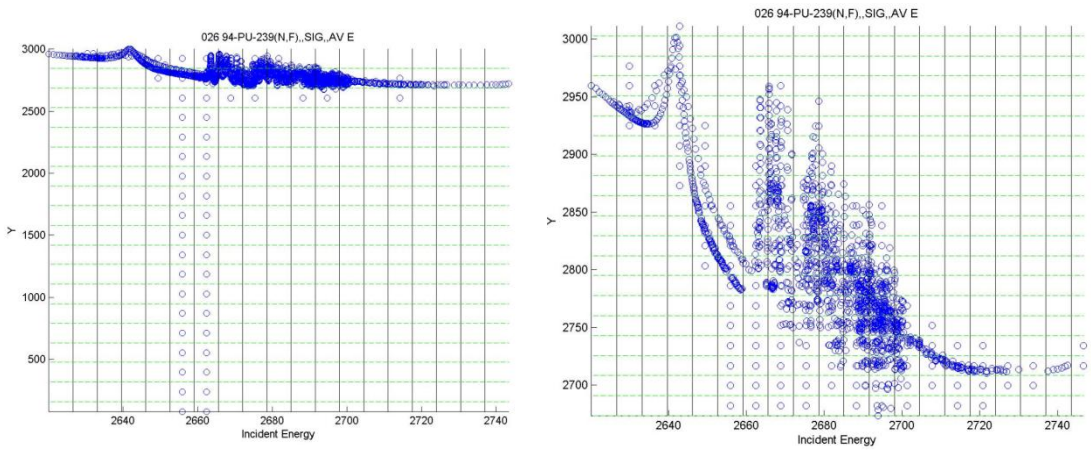


Figure 56 : Représentation de la réaction 016 83-BI-209(N,F),,SIG E avant et après la limitation de l'étendue de l'incertitude

Cette solution permet de centrer le graphique sur les vraies mesures observées au lieu de se focaliser sur leur incertitude. La configuration reste ainsi optimale pour la recherche d'anomalies.

- Les mesures peuvent contenir des valeurs nulles. Le problème est identique à celui du point précédent : la représentation graphique de la réaction n'est pas optimale, comme on peut le voir sur l'exemple suivant :

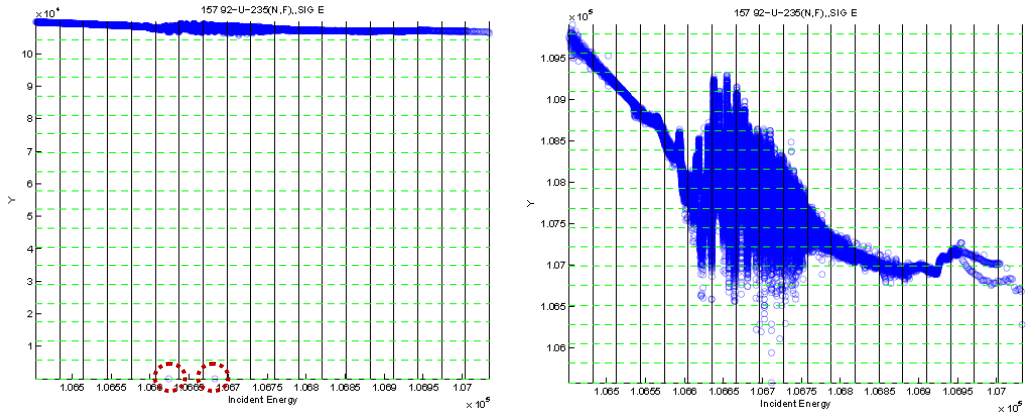


Figure 57 : Visualisation de la réaction `157 92-U-235(N,F),,SIG E` avec et sans données nulles

Le graphique à gauche représente la réaction avec les mesures de valeur nulle (entourées en rouge) alors qu'à droite, on représente cette même réaction en supprimant ces valeurs. Ces valeurs nulles ne sont pas aberrantes du fait de l'incertitude mais, elles peuvent nuire fortement à la qualité de la visualisation graphique, et donc à la détection des anomalies.

Pour résoudre ce problème, il n'y a pas de solution idéale. En fait, lors de la transformation des valeurs pour la recherche de l'échelle idéale, même les valeurs faibles deviennent très grandes. Toutefois, la valeur de zéro restera toujours zéro et donc éloignés des autres mesures non nulles. Nous avons choisi de supprimer les valeurs nulles des gros fichiers où la valeur d'alpha est très élevée.

- L'existence de valeurs négatives constitue un problème pour la détection de l'échelle optimale : si les données sont négatives et que nous cherchons l'échelle polynomiale optimale, nous obtiendrons des nombres complexes. Il a été décidé de supprimer les nombres négatifs, qui sont très peu nombreux. Cette solution n'est pas idéale, car on néglige une partie (très faible) de l'information, mais elle permet tout de même d'appliquer la méthode pour les autres données. En outre, les valeurs nulles et faibles sont toujours analysées mais, elles présentent souvent des incertitudes élevées. Afin d'éviter que ces données deviennent négatives, nous avons imposé que la valeur la plus faible (y compris l'incertitude) qu'une mesure peut prendre soit égale à zéro.

  Au cours de la dernière phase de validation, nous avons analysé d'autres bases de données présentant de nouvelles variables telles que les angles. Dans ce cas, les données négatives sont courantes et normales. Dans ce cas, nous ne rencontrons pas de problème, car il s'agit de configurations simples pour lesquelles l'échelle linéaire semble suffisante. Toutefois, si par la suite on souhaite élargir cette méthode à d'autres bases de données, il faudra ajouter d'autres échelles capables de gérer les nombres négatifs.

- Il est possible de trouver des anomalies très éloignées des autres mesures. Comme pour les cas précédents, cela peut avoir un impact visuel sur la configuration du graphique : dans ce cas, on détectera cette anomalie mais on ne pourra pas identifier d'autres anomalies à plus courte distance. Afin d'éviter ceci, l'algorithme est appliqué deux fois en cas d'existence d'une valeur beaucoup plus faible que les autres. Toutefois, il faut définir un seuil en fonction de la discrétisation choisie. Ce seuil ne doit pas être trop faible pour éviter de faire tourner l'algorithme deux fois pour des cas qui ne sont pas nécessaires et ainsi identifier des faux positifs. Actuellement, ce seuil est fixé arbitrairement à 10 intervalles

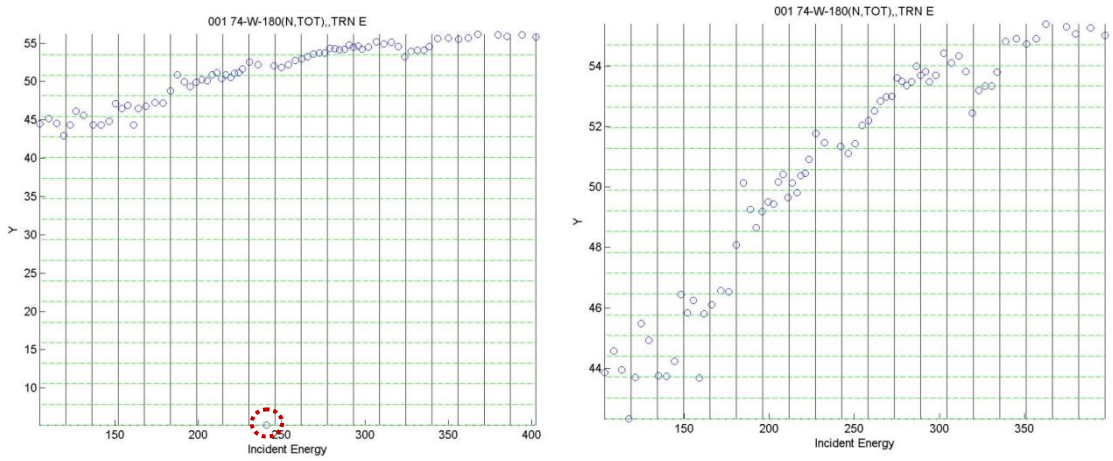
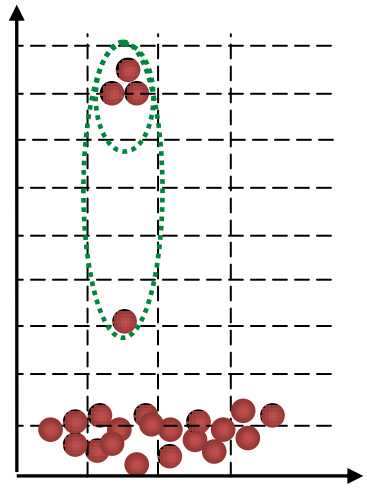


Figure 58 : Représentation de la réaction `001 74-W-180(N,TOT),,TRN E` avant et après la suppression de la donnée aberrante

- Le logiciel peut détecter pour la même tranche verticale plusieurs anomalies. Dans ce cas, nous trouverons sur le tableau de résultats une ligne pour chaque anomalie détectée (et chaque série). L'utilisateur peut ensuite, en fonction du poids des données aberrantes et de la distance, identifier l'anomalie principale. Par exemple, cela peut être utile pour ces types de configuration :

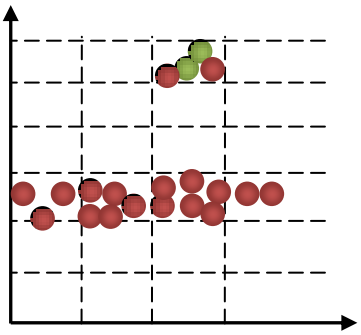


	nombre carrés	poids	nb données aberrantes
1er anomalie	1	0.4	4
2éme anomalie pour la même tranche verticale	4	0.3	3

Figure 59 : plusieurs anomalies dans la même tranche verticale

La première anomalie est négligeable du fait de la distance : le nombre d'intervalles vides est seulement égal à 1. Toutefois, la deuxième anomalie est plus importante, la distance est plus élevée. En conséquence, les données faisant partie de cette deuxième anomalie ont plus de chances d'être aberrantes.

- Les anomalies peuvent être composées de mesures appartenant à différentes séries de données. Afin d'identifier correctement le nom de la série et le nombre de fois qu'elle apparait comme suspecte, le tableau de résultats présente une ligne pour chaque série de données identifiée comme suspecte.



	nom de la série	nombre carrés	poids	nb données aberrantes pour chaque série
1er anomalie	rouge	2	0.4	2
1er anomalie	verte	2	0.4	2

Figure 60 : exemple de plusieurs anomalies dans la même tranche verticale

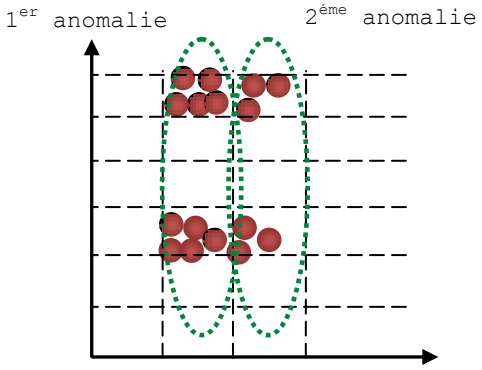
Seule la mise en forme des résultats est modifiée, et non le principe de la méthode. L'identification du nom de la série et des points qui font partie de la série est réalisée après la détection de l'anomalie. Dit autrement, le poids et la distance sont propres à l'anomalie identifiée (ensemble des données suspectes). Seuls le nombre de points et l'identification du numéro de la ligne font référence à la série suspecte.

Enfin, il ne faut pas exclure la possibilité de trouver d'autres cas, sûrement rares, pour lesquels le code devra être adapté.

C. Limites de la méthode

La méthode développée peut être utilisée pour vérifier des fichiers de différente taille. Toutefois, la robustesse de la méthode, et donc le degré de confiance des résultats, seront liés au nombre de données ainsi qu'à la configuration. Par exemple, si nous ne disposons que de 10 mesures mais qu'elles présentent une tendance claire, la méthode fonctionnera correctement. Toutefois, si les données se présentent sous une configuration de nuage de points, l'identification des données aberrantes est plus complexe.

Dans le cas où deux tendances d'égale importance existent, la méthode ne permet que de signaler l'existence d'une anomalie, sans pouvoir identifier quelles sont les données aberrantes. Dans ce cas, l'outil développé identifie les deux groupes de poids égal à 0.5 comme suspects ; l'AEN devra ensuite approfondir l'analyse pour déterminer lequel est anormal.



	nombre carrés	poids	nb données aberrantes
1er anomalie	2	0.5	5
1er anomalie	2	0.5	5
2éme anomalie	2	0.5	3
2éme anomalie	2	0.5	3

Figure 61 : exemple d'anomalie avec un poids égal à 50 %

Comme évoqué plus haut, la méthode ne permet pas une détection optimale dans le cas de configurations de la forme « nuage de points » : cela signifie qu'on ne dispose que d'une certaine partie de la courbe, en « zoom ». La tendance peut alors être difficile à cerner, le nuage de points étant diffus. Ceci peut créer des faux positifs : le logiciel risque d'identifier des points comme étant des anomalies alors qu'ils ne le sont pas.

Pour ces cas particuliers, on peut chercher à optimiser la discrétisation des axes en fonction du nombre de points. Cependant, il n'existe pas de méthode a priori permettant de lier le nombre d'intervalles de discrétisation et le nombre de points. Il faut donc trouver cette relation dans la pratique et pour cela, une analyse de sensibilité s'impose.

Enfin, il ne faut pas oublier qu'il s'agit d'une méthode inspirée par la détection réalisée par l'œil humain : si un expert ne peut pas conclure, le logiciel n'y parviendra pas non plus. Néanmoins, ces cas sont peu nombreux sur la base EXFOR.

D. Pistes d'amélioration

Dans l'avenir, les voies d'améliorations sont multiples. Nous proposons les suivants :

- Optimisation du temps de calcul : il est possible d'améliorer le programme développé en termes de temps de calcul. Dans ce but, l'AEN a d'ailleurs suggéré l'utilisation d'un algorithme du type « Golden section search » pour la recherche d'alpha optimal. Toutefois, cette méthode ne peut être intégrée que suite à une analyse de la base de données, car elle n'est applicable qu'aux fonctions remplissant la condition de fonction unimodale.

  Avant de modifier le code, il est donc nécessaire de vérifier que les fonctions sont bien unimodales quelle que soit la réaction et les variables mesurées et cela même pour les cas particuliers. D'ailleurs, il est possible qu'un tel algorithme fonctionne pour certaines réactions, mais pas pour la totalité.

- Mise en place d'un nouvel indicateur : les résultats sont actuellement triés en fonction de deux indicateurs : la distance, c'est-à-dire le nombre d'intervalles vides séparant deux ensembles homogènes de données, et le nombre de fois que la série a été identifiée comme aberrante. L'AEN a suggéré la création d'un indicateur mixte à partir de ces deux derniers, qui constituerait le seul critère de tri. Toutefois, il s'agit d'une tâche complexe car la nature de ces deux indicateurs est différente. De même, il faut réfléchir au poids qu'il faut attri-

buer à chacun dans la construction du nouvel indicateur. La méthode à développer doit donc être bien justifiée et une analyse de sensibilité s'impose ensuite.

- Approfondissement des cas difficiles : les configurations en forme de nuage de points ainsi que les cas composés d'un faible nombre de données peuvent être étudiés plus en détails. Comme indiqué précédemment, il faudrait réaliser plusieurs analyses de sensibilité afin de trouver une discrétisation des axes optimale pour traiter ces particularités.

- Prise en compte des incertitudes sur l'axe horizontal : il est possible d'intégrer les incertitudes des mesures sur l'axe des abscisses. Il suffit d'appliquer le même principe que pour la prise en compte des incertitudes en ordonnées. Toutefois, leur implémentation demande une modification profonde du code. Compte tenu du temps disponible et de l'intérêt à priori limité d'une telle incorporation, nous avons préférer d'orienter nos efforts vers d'autres priorités.

  Cette ajout n'est intéressant que dans certains cas rares d'EHD, comme dans l'exemple suivant :

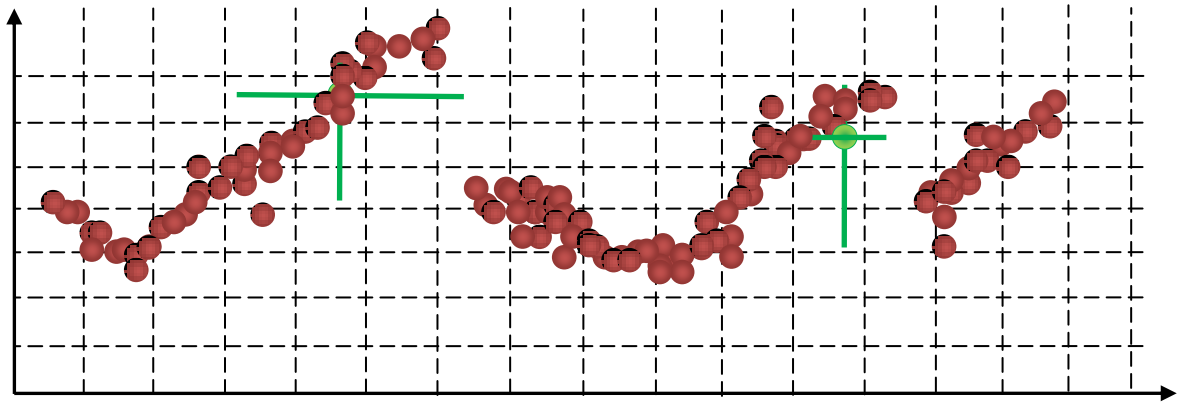


Figure 62 : Prise en compte de l'incertitude pour l'axe x

  La prise en compte de l'incertitude en x est intéressante si elle permet d'étaler les données sur des cases de discrétisation vides entourées (en haut et en bas) par deux ensembles de données. La mesure occupera donc une case de discrétisation qui autrement aurait été identifiée comme un intervalle vide composant une anomalie. Cependant, il ne faut pas négliger que si le nombre d'intervalles vides est 1 ou 2, l'anomalie identifiée ne sera pas importante. Il faudrait donc que la donnée dispose aussi d'une incertitude importante en y pour que la prise en compte de l'incertitude en x soit réellement utile.

  Les outils développés pour l'axe des ordonnées sont également applicables à l'axe des abscisses. Il est donc possible pour l'AEN de tester les bénéfices d'une telle incorporation.

- Application de la méthode à de nouvelles bases de données : le principe de détection de données reste pertinent, il faut seulement adapter le code aux nouveaux cas particuliers. Par exemple, si la nouvelle base de données présente des données négatives, il faudrait peut-être inclure une autre échelle de visualisation des données à la place de l'échelle polynomiale. En outre, la méthode peut être élargie aux cas 3D et autres cas multidimensionnels.

  Au cours de ce contrat, nous avons étudié les cas tests de nature multidimensionnelle (5 exemples de distribution angulaires, 3 exemples de distribution d'énergie et 3 exemples de

rendement de fission) : nous avons testé la méthode développée sur ces fichiers en ne prenant en compte que deux variables à la fois. Le logiciel traite correctement ces nouvelles variables. Néanmoins, les données identifiées comme aberrantes sont à prendre avec précaution car nous n'avons travaillé qu'en 2D. En fait, cet essai a servi à tester la méthode sur des variables autres que l'énergie et la section efficace mais, il n'a pas servi à tester la méthode en 3D.

- Méthode complémentaire de détection d'anomalies : il est possible de développer d'autres méthodes afin d'identifier les données aberrantes. Par exemple, il serait intéressant d'analyser la continuité de la loi de probabilité. Pour cela, il faut étudier l'évolution de l'espérance : la moyenne de l'ensemble des mesures contenues dans la même tranche verticale. S'il existe une modification importante de cette valeur, on pourra alors signaler la présence d'une discontinuité horizontale dans la distribution des mesures. Cette méthode pourrait donc être utilisée en complément de la méthode actuelle. Elle est facilement implémentable et intégrable au code actuel.

Annexes

Résultats de la validation pour l'échantillon de 50 cas test représentatifs de la majorité des anomalies pouvant être rencontrées dans la base EXFOR.

No	nom_fichier	Commentaires SCM	visuellement	logiciel
1	001 1-H-2(N,NON),,SIG E	*ok*	0	0
2	001 6-C-12(N,G)6-C-13,,SIG,,MXW KT	*ok*	0	0
3	001 8-O-16(N,P+D)6-C-14,SEQ,SIG,,,CALC E	*ok*	0	0
4	001 26-FE-57(N,G)26-FE-58,,SIG,,AV E	peu des données	3	3
5	001 47-AG-0(N,SCT)47-AG-0,,SIG E	*ok*	0	0
6	001 62-SM-148(N,2N)62-SM-147,,SIG E	*ok*	0	0
7	001 81-TL-203(N,TOT),,SIG E	nuage de points, changer la discrétisation		
8	002 14-SI-29(N,G)14-SI-30,,SIG E	peu de données, nuage de points		
9	001 92-U-238(N,TOT),,TRN,,SPA E	cas particulier avec paramètres cachés		
10	002 96-CM-244(N,TOT),,SIG E	*ok*	0	0
11	005 52-TE-122(N ,2N) 52-TE-121-G, SIG E	*ok*	0	0
12	006 66-DY-161(N,G)66-DY-162,,SIG E	*ok*	0	0
13	008 41-NB-93(N,A)39-Y-90,,SIG E	*ok*	0	0
14	010 51-SB-121(N,G)51-SB-122,,SIG E	*ok*	0	0
15	012 39-Y-89(N,INL)39-Y-89-M,,SIG E	*ok*	0	0
16	019 32-GE-70(N,2N)32-GE-69,,SIG E	*ok*	1	1
17	024 45-RH-103(N,G)45-RH-104,,SIG E	*ok*	0	0
18	034 47-AG-0(N,G),,SIG E	possible « faux positif » nb_carres=3	0	1
19	057 3-LI-6(N,T)2-HE-4,,SIG E	*ok*	0	0
20	084 28-NI-58(N,P)27-CO-58,,SIG E	*ok*	0	0
21	001 1-H-3(P,G)2-HE-4,,SIG E	*ok*	0	0
22	001 4-BE-10(P,N)5-B-10,,SIG E	nuage de points, problème de discrétisation		
23	001 20-CA-40(P,X)17-CL-36,,SIG E	je ne sais pas s'il y a une anomalie, c'est un cas limite : nb_carres=2 nom_serie=3	0	2
24	001 28-NI-0(P,X)24-CR-51,,SIG,,,EXP E	*ok*	0	0
25	001 34-SE-78(P,2N)35-BR-77-G,IND_over_M+,SIG,,,EXP E	*ok*	0	0
26	001 40-ZR-90(P,X)39-Y-86-G,M+,SIG,,,EXP E	*ok*	0	0
27	001 50-SN-117(P,N)51-SB-117,IND,SIG E	*ok*	0	0
28	001 73-TA-181(P,N+P)73-TA-180-M,,SIG E	je ne sais pas s'il y a une anomalie, c'est dans un cas : nb_carres=3 nom_serie=3	3	3
29	001 82-PB-0(P,X)54-XE-132,CUM,SIG E	*ok*	0	0
30	001 95-AM-241(P,2N)96-CM-240,,SIG,,,EXP E	*ok*	0	0
31	002 13-AL-27(P,X)0-NN-1,,SIG E	*ok*	1	1
32	002 27-CO-59(P,4N+P)27-CO-55,,SIG E	possible "faux positifs" c'est dans la limite	0	1
33	002 35-BR-0(P,X)36-KR-79,,SIG E	peu de données, nuage des points		
34	002 50-SN-0(P,X)51-SB-118-M,,SIG E	*ok*	0	0
35	002 82-PB-0(P,X)47-AG-110-M,IND,SIG E	*ok*	0	0
36	002 94-PU-244(P,F)57-LA-142,(CUM),SIG E	*ok*	0	0
37	003 25-MN-55(P,X)4-BE-10,,SIG E	*ok*	0	0
38	003 35-BR-81(P,N+P)35-BR-80-G,,SIG E	peu de données, problème de discrétisation	0	2
39	003 56-BA-0(P,X)55-CS-136-G,M+,SIG E	*ok*	0	0

40	003 95-AM-241(P,F),,SIG E	*ok*	0	0
41	004 28-NI-62(P,N)29-CU-62,,SIG E	*ok*	0	0
42	004 79-AU-197(P,X)37-RB-83,CUM,SIG E	*ok*	0	0
43	005 27-CO-59(P,N)28-NI-59,,SIG E	*ok*	0	0
44	006 13-AL-27(P,N+3P)11-NA-24,,SIG E	peu de données. Je ne sais pas s'il y a une anomalie		
45	007 22-TI-0(P,X)19-K-43,CUM,SIG E	*ok*	0	0
46	008 29-CU-0(P,X)25-MN-54,,SIG E	*ok*	0	0
47	010 30-ZN-68(P,2N)31-GA-67,,SIG,,,EXP E	peu des données, je ne sais pas il 'y a une anomalie mais ce cas est dans la limite	0	1
48	012 92-U-0(P,F),,SIG,,,EXP E	*ok*	0	0
49	016 26-FE-0(P,X)27-CO-56,,SIG E	*ok*	0	0
50	022 29-CU-63(P,N)30-ZN-63,,SIG,,,EXP E	*ok*	0	0

Résultats détaillés de la validation pour l'échantillon de 50 cas test représentatifs d'anomalies très rares. Ce tableau résume les résultats pour les fichiers ayant un poids faible (moins de 500 Ko).

No	nom_fichier	Commentaires SCM	visuellement	logiciel
1	001 74-W-180(N,TOT),,TRN E	le logiciel tourne 2 fois, nb_carres>15 pb de faux positif	1	6
2	002 6-C-12(N,TOT),,SIG E TEMP	*ok*		
3	003 95-AM-242-M(N,F),,SIG,,AV E	*ok*	4	4
4	003 97-BK-249(N,F),,SIG E	*ok*	0	0
5	004 82-PB-206(N,3N)82-PB-204-M,,SIG E	*ok*	0	0
6	005 53-I-129(N,2N)53-I-128,,SIG E	*ok*	3	3
7	006 28-NI-58(N,EL)28-NI-58,,SIG E	*ok*	0	0
8	006 66-DY-164(N,A)64-GD-161,,SIG E	*ok*	1	1
9	006 82-PB-0(N,2N),,SIG E	*ok*	0	0
10	006 92-U-235(N,ABS),,ALF,,RES EN-RES	nuage de points, problème de discrétisation		
11	007 40-ZR-90(N,2N)40-ZR-89-M_over_G,,SIG_over_RAT E	*ok*	1	1
12	007 94-PU-239(N,TOT),,SIG,,RES EN-RES	*ok*	1	1
13	008 1-H-1(N,EL)1-H-1,,SIG E	*ok*	0	0
14	009 42-MO-95(N,P)41-NB-95,,SIG E	*ok*	0	0
15	010 11-NA-23(N,EL)11-NA-23,,SIG E	*ok*	0	0
16	011 32-GE-74(N,A)30-ZN-71-M,,SIG E	*ok*	4	2
17	012 71-LU-0(N,G),,SIG E	*ok*	8	8
18	015 32-GE-76(N,2N)32-GE-75,,SIG E	*ok*	3	2
19	016 58-CE-140(N,2N)58-CE-139-M,,SIG E	*ok*	6	6
20	016 73-TA-0(N,G),,SIG E	*ok*	0	0
21	016 83-BI-209(N,F),,SIG E	*ok*	6	6
22	018 38-SR-88(N,P)37-RB-88,,SIG E	*ok*	0	0
23	019 28-NI-58(N,P)27-CO-58-M,,SIG E	*ok*	0	0
24	021 42-MO-92(N,2N)42-MO-91-M,,SIG E	*ok*	0	0
25	021 92-U-235(N,F),,SIG,,AV E	*ok*	0	0
26	022 92-U-238(N,2N)92-U-237,,SIG E	*ok*	0	0
27	024 29-CU-65(N,G)29-CU-66,,SIG E	*ok*	0	0

28	024 49-IN-115(N,2N)49-IN-114-M,,SIG E	*ok*	**0**	**0**
29	025 4-BE-9(N,EL)4-BE-9,,SIG E	*ok*	**0**	**0**
30	026 30-ZN-64(N,P)29-CU-64,,SIG,,FIS EN-DUMMY	*ok*	**0**	**0**
31	026 94-PU-239(N,F),,SIG,,AV E	*ok*	**0**	**0**
32	027 40-ZR-90(N,A)38-SR-87-M,,SIG E	*ok*	**1**	**1**
33	034 27-CO-59(N,P)26-FE-59,,SIG E	*ok*	**4**	**4**
34	035 25-MN-55(N,2N)25-MN-54,,SIG E	*ok*	**0**	**0**
35	036 12-MG-24(N,P)11-NA-24,,SIG E	**il faut vérifier le fichier de données**	**3**	**3**
36	036 74-W-186(N,G)74-W-187,,SIG E	*ok*	**0**	**0**
37	042 90-TH-232(N,G)90-TH-233,,SIG E	*ok*	**0**	**0**
38	051 49-IN-115(N,INL)49-IN-115-M,,SIG E	*ok*	**0**	**0**
39	081 13-AL-27(N,A)11-NA-24,,SIG E	*ok*	**0**	**0**
40	092 13-AL-27(N,P)12-MG-27,,SIG E	*ok*	**0**	**0**
41	003 25-MN-55(P,N+P)25-MN-54,,SIG E	**peu de données**	**1**	**1**
42	004 45-RH-103(P,3N)46-PD-101,,SIG E	**possible "faux positif"**	**0**	**2**
43	004 47-AG-109(P,N)48-CD-109,,SIG E	*ok*	**0**	**0**
44	006 30-ZN-67(P,2N)31-GA-66,,SIG E	*ok*	**7**	**7**
45	009 26-FE-0(P,X)24-CR-51,CUM,SIG E	*ok*	**1**	**1**
46	010 22-TI-0(P,X)21-SC-44-M,,SIG E	*ok*	**2**	**2**
47	011 29-CU-0(P,X)30-ZN-62,,SIG E	*ok*	**4**	**4**
48	013 26-FE-0(P,X)27-CO-56,IND,SIG,,,EXP E	*ok*	**0**	**0**
49	019 92-U-238(P,F),,SIG E	*ok*	**6**	**6**
50	016 5-B-0(N,ABS),,SIG E	*ok*	**1**	**1**