Modélisation des incertitudes en mécanique par simulation de Monte Carlo: propagation et hiérarchisation des incertitudes, calcul de courbes de fragilité

Résumé :

Ce document donne les éléments pour la mise en œuvre de simulations numériques de Monte Carlo à partir du fichier de commande et d'opérateurs de génération aléatoire. Les quatre principaux ingrédients sont :

- Une boucle Python,
- Un générateur de variables aléatoires (GENE_VARI_ALEA) et/ou un générateur de matrices aléatoires (GENE_MATR_ALEA) pour la dynamique, et/ou un générateur de fonctions aléatoires (GENE_FONC_ALEA),
- Le calcul d'estimateurs statistiques classiques (CALC_FONCTION) et estimation de paramètres de courbes de fragilité (POST_DYNA_ALEA) (avec les arbres de défaillances et les courbes d'aléa sismiques, les courbes de fragilité constituent les principaux ingrédients pour les études probabilistes de sûreté (EPS) sismique),
- Hiérarchisation des sources d'incertitude.

Le domaine d'application principal présenté ici est l'analyse sismique, mais les méthodes dépassent bien évidemment ce cadre restreint.

Table des Matières

1 Introduction

La méthode numérique de Monte Carlo permet de calculer différentes grandeurs statistiques d'une variable aléatoire ou d'un processus stochastique. Dans le contexte d'un calcul mécanique (ou thermo-mécanique, …), le principe est d'obtenir ns réalisations de cette variable aléatoire ou de ce processus stochastique puis d'en déduire les estimations statistiques recherchées. Les quatre principales étapes de la méthode de Monte Carlo sont :

- Modélisation des incertitudes : Génération d'un échantillon de ns réalisations des données aléatoires d'entrée du modèle mécanique,
- Propagation des incertitudes : Calcul des ns grandeurs résultats correspondant à ces données,
- Calcul des estimateurs statistiques des grandeurs recherchées.
- Hiérarchisation des incertitudes

Dans la méthode de Monte Carlo simple ou directe que l'on utilise, chacun des *ns* calculs peut être fait indépendamment des autres. Afin de réduire la taille mémoire nécessaire, les *ns* générations et calculs sont donc effectuées séquentiellement dans une boucle avec destructions des résultats intermédiaires inutiles.

Les méthodes d'hiérarchisation des incertitudes permettent de « classer » les différentes sources d'incertitudes selon leur importance. On présente ici la méthode la plus générale (applicable dans le cas de modélisations comportant aussi bien des variables aléatoires que des processus stochastiques et des matrices aléatoires) s'appuyant sur la décomposition de variance [bib2].

Les objectifs principaux d'un calcul de sensibilité sont généralement les suivants :

- Identification des variables peu ou pas influentes afin de réduire la taille du modèle (ces variables ne feront alors plus objet d'une modélisation probabiliste).
- Identification des variables les plus influentes afin d'affiner leur modélisation ou de réduire les incertitudes si possible (par exemple en collectant de nouvelles données afin de réduire le manque de connaissance initial). Ceci conduit finalement à une réduction des incertitudes sur la grandeur d'intérêt.
- Amélioration de la connaissance sur le modèle, évaluation de la qualité du modèle (les incertitudes sur la grandeur d'intérêt, relèvent-elles surtout du manque de connaissance lié aux paramètres du modèle ou par exemple sur l'excitation sismique, par nature aléatoire ?).

1.1 Courbes de fragilité sismique

Cette documentation présente une méthodologie numérique permettant de déterminer les paramètres d'une courbe de fragilité sismique. L'estimation des paramètres d'une courbe de fragilité se situe dans le cadre du post-traitement statistique (le point 3 ci-dessus). Dans ce qui suit, on évoque brièvement le cadre dans lequel sont utilisées ces courbes.

Les courbes de fragilité constituent l'un des ingrédients d'une étude probabiliste de sûreté (EPS) sismique. L'objet des EPS sismique est l'étude du risque de défaillance d'un système (en l'occurrence l'installation nucléaire) composé de sous-structures, équipements, etc. La probabilité de défaillance de l'installation nucléaire est alors déterminée au moyen d'un arbre de défaillance recensant toutes les combinaisons possibles d'événements qui conduisent à la défaillance. Afin de déterminer le risque de défaillance des composants (sous-structures et équipements), il est nécessaire de déterminer les courbes de fragilité associées qui donnent la probabilité conditionnelle de défaillance en fonction d'un niveau de paramètre représentatif de l'action sismique (en général, on prend l'accélération maximale du sol). La probabilité de défaillance du composant est obtenue après convolution avec la courbe d'aléa sismique. Pour un site donné, la courbe d'aléa sismique représente la probabilité annuelle du dépassement de valeurs du paramètre représentatif de l'action sismique.

2 Modélisation des incertitudes

Dans le cadre de la simulation de Monte Carlo, la modélisation des incertitudes passe par la génération de variables aléatoires et éventuellement de processus stochastiques. Dans ce qui suit, on suppose que l'utilisateur ait choisi des lois caractérisant bien les paramètres incertains. Pour ce faire, on peut avoir recours à des résultats d'essais, le jugement d'expert ou encore le principe du maximum d'entropie (voir [R4.03.05]).

Remarquons que la terminologie de variables aléatoire doit être ici prise au sens large. Ces variables aléatoires peuvent être à valeurs scalaires ou matricielles. Le *Code_Aster* est capable de générer des variables aléatoires, des matrices aléatoires ainsi que des processus stochastiques gaussiens stationnaires, respectivement par les commandes GENE_VARI_ALEA, GENE_MATR_ALEA et GENE_FONC_ALEA.

A moins d'une indication contraire à l'aide du mot clé INIT_ALEA, toutes les valeurs générées par les trois commandes GENE_VARI_ALEA, GENE_MATR_ALEA et GENE_FONC_ALEA sont statistiquement indépendantes entre elles à l'intérieur d'une même exécution de *Code_Aster*. A contrario, d'une exécution à l'autre, un fichier de commande strictement identique (mêmes appels aux trois commandes dans le même ordre avec les mêmes arguments) fournira exactement les mêmes résultats. Ceci est dû au fait que le générateur de variables aléatoires utilisé par *Code_Aster* est toujours initialisé à la même valeur. Si l'on souhaite générer des résultats différents d'une exécution à l'autre, alors il faut utiliser le mot-clé INIT_ALEA avec des valeurs majorant le nombre de termes utilisés dans les exécutions antérieures.

Attention :

En général, on peut supposer que les différentes sources d'incertitudes sont indépendantes. On peut alors effectuer des tirages aléatoires

2.1 Variables et matrices aléatoires

Les variables aléatoires peuvent être des paramètres du modèle éléments finis (paramètres matériaux, valeurs d'un jeu, d'une raideur de butée élastiques, d'un module d'Young, etc). Dans ce cas on modélise les incertitudes de modélisation par une approche probabiliste paramétrique et on utilise alors GENE_VARI_ALEA [U4.36.07].

En dynamique des structures sur base modale, ces variables aléatoires peuvent aussi être les matrices généralisées de masse, de raideur et d'amortissement et/ou les paramètres locaux du modèle aux éléments finis. Dans ce cas, on modélise à la fois les incertitudes de modèle et de modélisation par une approche probabiliste non-paramétrique, et on utilise GENE_MATR_ALEA [U4.36.06].
Ces variables aléatoires à valeurs scalaires ou matricielles suivent des lois de probabilités construites par l'utilisation du principe du maximum d'entropie et de l'information disponible (voir [R4.03.05]). On peut consulter le cas-test SDNS01 [V5.06.001] pour l'utilisation dans un calcul de simulation de Monte Carlo.

Attention :

Les matrices généralisées obtenues ne sont pas diagonales et nécessitent donc un stockage plein.

Remarque :

Le module "random" de Python fournit une alternative à la commande GENE_VARI_ALEA *pour générer des variables aléatoires dont les densités ne sont pas disponibles dans cette commande.*

2.2 Processus stochastiques

L'opérateur GENE_FONC_ALEA [U4.36.05] permet de générer des trajectoires d'un processus stochastique stationnaire gaussien multi-varié mono-dimensionnel (i.e. à plusieurs composantes et indexé sur une seule variable) à partir de sa densité spectrale de puissance. Dans le cas d'un calcul dynamique transitoire, on peut ainsi générer des chargements temporels stationnaires connues par leur matrice interspectrale. On peut consulter le cas-test ZZZZ180a [V1.01.180] pour un exemple d'utilisation de GENE_FONC_ALEA dans une boucle de Monte Carlo.

Dans d'autres cas, on peut disposer d'une base de données contenant des signaux qui modélisent un phénomène physique aléatoire. C'est le cas en analyse sismique, où on peut être amené à modéliser l'excitation sismique par un processus stochastique non stationnaire via la donnée d'accélérogrammes. Ces accélérogrammes peuvent être mesurés ou produits par un logiciel dédié.

3 Propagation des incertitudes par simulation de Monte Carlo

La simulation de Monte Carlo consiste à tirer *ns* réalisations des variables, matrices et processus aléatoires ce qui permet de faire *ns* calculs. Dans *Code_Aster* ceci est possible à l'aide d'une boucle Python dans le fichier de commande Aster (cf. [U1.03.01]).

3.1 Utilisation de boucles en Python

La boucle python en elle-même commence par la commande for, et englobe toutes les lignes de même indentation

```
for k in range(1,1000) :
    COMMANDE1
    for m in range(1,500):
       COMMANDE2
    COMMANDE3
```

Dans cet exemple, on trouve deux boucles python emboîtées. La première, sur la variable k , permet d'exécuter 999 fois les instructions COMMANDE1, la deuxième boucle python, et COMMANDE3. La deuxième boucle python interne, sur la variable m permet à COMMANDE2 d'être exécuté 499 fois pour chaque k allant de 1 à 499.

Attention :

1) Il n'y a pas d'instruction de fin de boucle. Les indentations seules marquent le corps de l'instruction « for ».

2) *Si l'utilisateur n'indique rien d'autre, la numérotation des indices dans Python commence à partir de zéro et non à partir de un. Si on écrit par exemple* `for k in range(1000)` *, alors k prendra les valeurs 0,1,…,999. On exécute alors 1000 fois la commande.*

4 Post-traitement statistique

4.1 Estimateurs statistiques classiques

D'un échantillon de ns réalisations de la quantité d'intérêt, on peut déduire les estimations de grandeurs statistiques comme la moyenne, l'écart type … La documentation [R7.10.01] donne plus d'éléments sur le post-traitement statistique de résultats de calcul. Dans ce qui suit, on note $E(X)$ l'espérance mathématique (la moyenne) d'une variable aléatoire X et $V(X)$ sa variance.

Prenons par exemple un échantillon de spectres d'oscillateurs $\left\{SRO_k(\omega,\xi)\right\}$, $1\le k\le ns$, pour lequel nous souhaitons calculer, pour chaque pulsation ω , les moments d'ordre 1 (l'estimateur de la moyenne) et d'ordre 2.

Remarque :

En général, on note par θ le paramètre (par exemple l'espérance mathématique $E(X)$) et par $\hat{\theta}$ son estimateur (par exemple la moyenne empirique $\hat{E}(X)$).

Ces moments ont pour expression :

$$\hat{E}\left(SRO(\omega,\xi)\right)=\hat{m}_1(SRO(\omega,\xi))=\frac{1}{ns}\sum_{k=1}^{ns}SRO_k(\omega,\xi)\ ,$$

$$\hat{E}\left(SRO(\omega,\xi)^2\right)=\hat{m}_2(SRO(\omega,\xi))=\frac{1}{ns}\sum_{k=1}^{ns}SRO_k(\omega,\xi)^2\ .$$

Dans ce qui suit, on écrira $\hat{m}_l(\omega,\xi)\equiv\hat{m}_l\left(SRO(\omega,\xi)\right)$, $l=1,2$ pour simplifier les notations.

Les deux sommes ci-dessus sont aisément calculables par une formule itérative, où on met à jour l'estimateur au fil des simulations de Monte Carlo. Il est seulement nécessaire de différencier le cas de l'initialisation et puis la mise à jour de l'estimation.

Voici, par exemple, un fichier de commandes épuré permettant d'évaluer $\hat{m}_2(\xi,\omega)$:

Génération d'une réalisation aléatoire du spectre d'oscillateur

Calcul de l'estimateur $\hat{m}_2^k(\xi,\omega)$ à l'itération k

```
for k in range(1,ns+1) :
  MATM=GENE_MATR_ALEA(MATR_MOYEN=MASSE,DELTA=0.2)
  MATK=GENE_MATR_ALEA(MATR_MOYEN=RIGID,DELTA=0.2)
  MATD=GENE_MATR_ALEA(MATR_MOYEN=AMORT,DELTA=0.2)
  DYNA =DYNA_TRAN_MODAL( MASS_GENE=MATM,
                         RIGI_GENE=MATK,
                         AMOR_GENE=MATD,
                            ... )
  ACC1=RECU_FONCTION(RESU_GENE = DYNA, ... )
  SRO= CALC_FONCTION(SPEC_OSCI=_F(NATURE='ACCE',FONCTION=ACC1,...))
  if k==1:
     M2_3= CALC_FONCTION(PUISSANCE=_F(FONCTION=SRO, EXPOSANT=2),)
  else:
     M2_0 = CALC_FONCTION(PUISSANCE=_F(FONCTION=SRO, EXPOSANT=2),)
     M2_1 = CALC_FONCTION( COMB=(_F(FONCTION=M2_0,COEF= k**-1),
                                 _F(FONCTION=M2_3,COEF=(k-1)*k**-1)) )

           DETRUIRE(CONCEPT=_F(NOM=(M2_3,M2_0)))
           M2_3=  CALC_FONCTION(COMB=_F(FONCTION=M2_1, COEF=1.), )
           DETRUIRE(CONCEPT=_F(NOM=(M2_1)))
       DETRUIRE CONCEPT=_F(NOM=(MATM,MATK,MATD,DYNA,SRO,ACC1)))
```

Lorsque `k==1`, on initialise la fonction `M2_3` avec le carré de la première réalisation produite. La puissance énième d'une fonction est effectuée par le mot clé `PUISSANCE` de la commande `CALC_FONCTION`. La première estimation $\hat{m}_2^1(\xi,\omega)$ est égale à la valeur SRO calculé à l'étape k=1 :

$$\hat{m}_2^1(\xi,\omega)=SRO_1(\xi,\omega)^2$$

Ensuite, on utilise une formulation récursive afin d'évaluer la nouvelle valeur de l'estimateur à chaque itération *k* :

$$\hat{m}_2^k(\xi,\omega)=\frac{SRO_k(\xi,\omega)^2}{k}+\frac{k-1}{k}\hat{m}_2^{k-1}(\xi,\omega);$$

Il n'est alors pas nécessaire de stocker tous les *ns* résultats de calcul. La nouvelle estimation est obtenue au fur est à mesure que les réalisations sont produites. Elle est stocké dans M2_3 à l'aide des mots clés PUISSANCE et COMB de la commande CALC_FONCTION [U4.32.04], des fonctions intermédiaires M2_0 et M2_1 et de la commande DETRUIRE [U4.14.01]. Tous les différents concepts produits (MATM, MATK, MATD, DYNA, SPO, ACC1.) doivent être détruits à la fin de chaque itération à l'exception de M2_3, bien entendu.

Remarque 1:

Il peut être utile de stocker les estimations intermédiaires $\hat{m}_2^k(\xi,\omega)$ *pour vérifier la convergence de l'estimation finale. Pour ce faire on crée, en début du ficher hors la boucle, une liste vide de taille ns,* `M2_1=[None]*(ns-1)`*, qu'on remplira au fur et à mesure* `M2_1[k]= CALC_FONCTION ....`

L'estimateur de la variance se détermine à partir de l'estimation des moments $\hat{m}_2(\xi,\omega)\equiv\hat{m}_2^{ns}(\xi,\omega)$ et $\hat{m}_1(\xi,\omega)\equiv\hat{m}_1^{ns}(\xi,\omega)$ par la formule classique (variance empirique corrigée):

$$\hat{V}\left(SRO(\xi,\omega)\right)=\frac{ns}{ns-1}\left(\hat{m}_2(\xi,\omega)-\hat{m}_1(\xi,\omega)^2\right) .$$

Remarque 2:

Le calcul de spectres de réponse et de leur enveloppe peut également se faire par la macro-commande `MACR_SPECTRE` *[U4.32.11.].*

4.2 Détermination de courbes de fragilité

Les courbes de fragilité donnent la probabilité conditionnelle de défaillance d'une structure ou d'un composant en fonction du niveau d'excitation sismique. La modélisation et la propagation des incertitudes décrites ci-dessus permettent également la détermination des courbes de fragilité. Il suffit d'introduire un critère de défaillance et de vérifier à chaque calcul de Monte Carlo si la défaillance est atteinte ou non.

Les ingrédients principaux pour l'établissement de courbes de fragilité par simulation numérique sont les suivants :

- Détermination de l'excitation sismique à considérer (base de données d'accélérogrammes),
- Définition de critères de défaillance,
- Modélisation des incertitudes et propagation par simulation numérique (Monte Carlo)
- Estimation des paramètres (médiane et écart-type logarithmique) de la courbe de fragilité associée au critère de défaillance.

4.2.1 Modèle log-normal pour les courbes de fragilité

La courbe de fragilité d'un composant est définie à partir de la notion de "capacité". La capacité d'un composant est la valeur du paramètre représentatif de l'action sismique à partir de laquelle le composant est défaillant. L'approche courante consiste à modéliser la capacité par une variable aléatoire suivant une loi log-normale, telle que $A=A_m\varepsilon$, où A_m est la capacité médiane et ε désigne une variable aléatoire log-normale de médiane unité et d'écart-type logarithmique β . Aussi , la capacité A d'un composant (et A_m ainsi sa courbe de fragilité), est caractérisée par deux paramètres qui sont la médiane (« capacité médiane ») et l'écart-type β .
Ainsi, la probabilité de ruine pour un niveau d'accélération *a* donné peut s'écrire [bib3]:

$$P_{f|a}(a)=\int_0^a \underbrace{p(x)}\, dx = \Phi\left(\frac{\ln(a/A_m)}{\beta}\right) \qquad (1)$$

Distribution log-normale

où $\Phi(.)$ désigne la fonction de répartition d'une variable aléatoire gaussienne centrée réduite.

4.2.2 Estimation des paramètres d'une courbe de fragilité par maximum de vraisemblance

Dans ce qui suit, on considère que l'accélération maximale a été choisie pour caractériser le niveau d'excitation sismique et donc la capacité. La démarche suivie consiste alors à modéliser le résultat des expériences numériques par une variable aléatoire de Bernoulli *X* . En effet, pour chaque simulation numérique i, on a deux issues possibles : soit on a atteint le niveau critique et on a défaillance ($x_i=1$) soit on n'a pas défaillance ($x_i=0$). De même pour chaque simulation, on peut déterminer la valeur de l'accélération maximale a_i .

L'estimation des paramètres d'une courbe de fragilité peut se faire par la méthode du maximum de vraisemblance. La fonction de vraisemblance à maximiser pour ce problème s'écrit :

$$L=\prod_{i=1}^{N}\left(P_{f|a}(a_i)\right)^{x_i}\left(1-P_{f|a}(a_i)\right)^{1-x_i}$$

Dans cette expression, la réalisation x_i de X prend donc la valeur 1 si on a défaillance ou 0 s'il n'y a pas défaillance pour le chargement (l'accélération maximale) a_i . Ces événements arrivent avec la probabilité $P_{f|a}$ donnée par l'expression (1). Les estimations des paramètres β et A_m sont ceux qui minimisent $-\ln(L)$:

$$(\beta_C^e, A_m^e)=\arg\min_{\beta, A_m}\left[-\ln(L)\right] .$$

Cette étape peut être mise en œuvre en post-traitement des résultats de calculs. Dans le fichier de commande *Code_Aster* il faut, pour chaque simulation, vérifier si on a défaillance ou non. Si le critère de défaillance consiste en une contrainte admissible, on vérifie si la contrainte maximale calculée dépasse cette contrainte.

Puis on fait écrire, à laide de CREA_TABLE et CALC_TABLE, les résultats des simulations dans une table de résultats. Cette table doit contenir deux colonnes, PARA_NOCI et DEFA, qui renseignent respectivement la valeur a_i (la valeur PGA ou tout autre indicateur) et la valeur x_i (1 ou 0). L'opérateur POST_DYNA_ALEA [U4.84.04] permet d'estimer paramètres de la courbe de fragilité à partir de cette table.

Dans ce qui suit, on reproduit les commandes permettant de déterminer les paramètres d'une courbe de fragilité selon le modèle log-normal présenté ci-dessus.

# on effectue ns simulations de Monte Carlo
for k in range(ns):

On tire au hasard l'un des accélérogrammes dans la base de données dont on détermine le PGA via la commande INFO_FONCTION(NOCI_SEISME=….

```
MAX=INFO_FONCTION(NOCI_SEISME=_F(    FONCTION=ACCE,
                                     OPTION='TOUT',
                                     AMOR_REDUIT = 0.1,
                                     PESANTEUR=9.81,))
                                     PGA=MAX['ACCE_MAX',1]
```

Puis, on calcule l'excitation sismique pour cet accélérogramme et on effectue le calcul mécanique Aster … (on calcule notamment la contrainte maximale Smax).

```
# On teste si on a eu défaillance
```

```
      # (la contrainte maximale Smax est supérieure à la contrainte admissible
      Sadm)
      xi=   0
      #  si on observe défaillance
                  if Smax >= Sadm :
             xi=1;
if k==0:
TAB1=CREA_TABLE(LISTE=(
                        _F(PARA='PARA_NOCI',LISTE_R = PGA,),
                        _F(PARA='DEFA',LISTE_I =xi ,),
                        ),);
else:
                      TAB1 = CALC_TABLE(reuse=TAB1, TABLE= TAB1 ,
                         ACTION=_F(OPERATION='AJOUT',
                               NOM_PARA=('PARA_NOCI', 'DEFA'),
                               VALE=(PGA ,xi ),),)
```

On peut ensuite déterminer les paramètres de la courbe de fragilité à l'aide de l'opérateur POST_DYNA_ALEA en choisissant le mot-clé FRAGILITE, voir [U4.84.04].

```
TAB_POST=POST_DYNA_ALEA( FRAGILITE=(_F(TABL_RESU=TAB1,
                                    LIST_PARA=lr,
                                    AM_INI  =0.3 ,
                                    BETA_INI=0.1 ,
                                    FRACTILE = (0.0,0.05,0.5,0.95,1.0),
                                    NB_TIRAGE =ns,
                                  ),),
                           TITRE = 'courbe 1',
                           INFO=2,);
```

On donne des valeurs initiales des paramètres A_m et β à estimer (point de démarrage pour l'algorithme d'optimisation) via AM_INI et BETA_INI.
Si l'on renseigne le mot-clé FRACTILE, on détermine également les fractiles (intervalles de confiance) demandés par une méthode de rééchantillonnage (dite méthode de « bootstrap »). Dans l'exemple, on détermine donc les courbes enveloppes (fractiles 1.0 et 0.0) ainsi que les fractiles à 0.05 et 0.95. Le rééchantillonnage consiste à tirer de nouveaux échantillons à partir des valeurs de l'échantillon original. Ces tirages s'effectuent avec remise, voir [bib1] pour plus de détails. Ensuite, on détermine les paramètres de la courbe de fragilité pour chaque échantillon « bootstrap », ce qui donne un échantillon de courbes de fragilités. On détermine alors les fractiles pour l'échantillon de courbes de fragilités obtenu.
En général, on tire autant d'échantillons « bootstrap » qu'on dispose de valeurs dans l'échantillon original. Ceci est fortement conseillé pour avoir des résultats fiables. Il est néanmoins possible de travailler avec un nombre de tirages inférieur en renseignant NB_TIRAGE (par défaut le nombre de tirages correspond à la taille de l'échantillon original, ici *ns*).
Dans le graphique ci-dessous, on donne un exemple de courbe de fragilité pour $A_m = 0.68$ et $\beta = 0.25$:

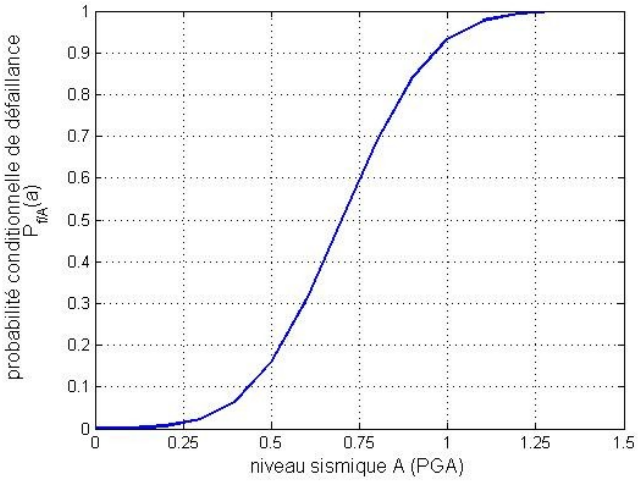

Figure 4.2.2-a : Probabilité conditionnelle de défaillance en fonction du PGA.

Remarque :

La valeur de contrainte admissible peut également être considérée comme une variable aléatoire. Dans ce cas, on peut tirer des réalisation de cette variable à l'aide de `GENE_VARI_ALEA`.

Remarque :

L'approche présentée ici possède l'avantage de nécessiter un nombre réduit de simulations par rapport à un calcul direct des probabilités de défaillance. En effet, dans un calcul direct, sans hypothèse de loi a priori, il faut déterminer des probabilités d'évènements rares (les probabilités en queue de distribution), ce qui demande un nombre de simulation de Monte Carlo très importants. Dans l'approche proposée ici, on utilise non seulement l'information fournit par les simulations aboutissant à la défaillance, mais aussi l'information des calculs sans défaillance. Il est néanmoins clair qu'il faut avoir observé un certain nombre de cas de défaillance pour que l'estimation des paramètres A_m *et* β *par la résolution d'un problème d'optimisation soit pertinente (et que l'algorithme d'optimisation converge).*

5 Hiérarchisation des sources d'incertitude

5.1 Définition des indices de sensibilités globaux

L'analyse de sensibilité globale s'intéresse à la variabilité de la réponse en fonction de la variabilité des variables d'entrée. Cette démarche permet de hiérarchiser les différentes sources d'incertitudes en fonction de leur contribution à la variabilité totale de la réponse.
Dans ce qui suit, on suppose que les différentes sources d'incertitudes peuvent être considérées statistiquement indépendantes.

Remarque :

L'hypothèse d'indépendance est souvent justifiée en mécanique. Par exemple, si l'on considère des incertitudes sur la géométrie, par exemple le jeu, un paramètre mécanique comme le module d'Young et sur le chargement (par exemple le séisme) on peut dire que le phénomène aléatoire qu'est le séisme, la tolérance de fabrication et de montage ainsi que les propriétés du matériau utilisée ne dépendent pas l'un de l'autre.

Dans ce qui suit et pour simplifier les notations, on suppose que les variables d'entrée incertaines sont des variables aléatoires (v.a.), notées $X_1,\ldots,X_m$, même si toute autre source d'incertitudes (matrice, processus) peut être prise en compte. La variable de sortie, c'est-à-dire la grandeur d'intérêt, $Y=f(X)$ est alors également une v.a. La fonction *f* est évaluée à l'aide de calculs *Code_Aster*.

On note $E(X)$ l'espérance mathématique d'une variable aléatoire X et $V(X)$ sa variance. Par ailleurs, on note par $\hat{E}^{ns}$ et respectivement $\hat{V}^{\mathrm{ns}}$ les estimations statistiques de ces quantités à partir d'un échantillon de taille *ns*.

On cherche donc à déterminer la part de la variance de Y expliquée par une variable d'entrée X_i ou un groupe de données d'entrée. Les indices de sensibilité globaux s'obtiennent par une décomposition de la variance totale. Si on introduit au total *m* sources d'incertitude différentes, on peut écrire :

$$V(Y)=\sum_i V_i+\sum_i\sum_{i<j} V_{ij}+\sum_i\sum_{i<j<k} V_{ijk}+\cdots+V_{12\ldots m}$$

où [bib2] :

$$V_i=V\left[E(Y|X_i)\right],\quad V_{ij}=V\left[E(Y|X_i,X_j)\right]-V_i-V_j\ ,\quad V_{ijk}=\cdots \qquad (0)$$

A partir de cette formulation, on peut déterminer les indices de sensibilité du premier ordre ou encore les indices totaux. La définition de ces indices ainsi que leur mode d'utilisation sont expliqués dans ce qui suit.

5.1.1 Indices de sensibilité d'ordre 1

Le classement des incertitudes à l'aide des indices du premier ordre permet de répondre à la question de savoir quelle variable (ou matrice…) il faut fixer pour avoir la plus grande réduction de variance. Il s'agit en gros de la quantité de variance qui serait enlevée de la variance totale si on connaissait la « vraie » valeur de X_i.

Les indices de sensibilité du premier ordre $S_i,\ i=1,\ldots m$ peuvent s'exprimer comme :

$$S_i=V_i/V(Y)$$

Cet indicateur est utile si l'on souhaite identifier les variables les plus influentes afin d'affiner leur modélisation ou de réduire les incertitudes sur ces variables (par exemple en collectant de nouvelles données afin de réduire le manque de connaissance initial). Ceci conduit finalement à une réduction des incertitudes sur la grandeur d'intérêt.

5.1.2 Indices de sensibilité totaux

D'autre part, les indices de sensibilité totaux S_{Ti} s'expriment comme :

$$S_{Ti}=1-\frac{V_{-i}}{V(Y)}\ ,$$
$$\text{où}\ \ V_{-i}=V(Y)-\left(V_i+\sum_{i<j}V_{ij}+\sum_{i<j<k}V_{ijk}+\cdots\right).$$

Ils mesurent l'effet total de la variable X_i, à savoir la quantité de variance qui subsisterait si toutes les variables sauf X_i étaient connues. Ainsi, si l'indice total est faible, alors le fait de modéliser cette source d'incertitude apporte peu ou rien pour le modèle. Au contraire, la taille du modèle peut être réduite en fixant le paramètre X_i à une valeur « raisonnable ». En effet, on peut fixer une

variable identifiée comme non-influente, à n'importe quelle valeur dans son domaine de définition sans changer la variance totale de manière significative et sans perte d'information.

Remarque1 :

L'indice de sensibilité totale pour la variable X_i s'écrit encore:

$$S_{Ti} = S_i + \sum_{i<j} S_{ij} + \sum_{i<j<k} S_{ijk} + \cdots .$$

Remarque2 :

Dans un cadre déterministe, on peut effectuer une analyse de sensibilité via le calcul de gradients, tel que :

$$\frac{\frac{\partial Y_j}{\partial X_i}(\bar{x})\,\mathrm{Var}(X_i)}{\mathrm{Var}(Y)}$$

C'est l'approche la plus classique. Un certain nombre de ces dérivées, $\frac{\partial Y_j}{\partial X_i}$ *, peut être calculé analytiquement par Code_Aster, voir [U2.08.02] et aussi [R4.03.04]. Il s'agit néanmoins d'une analyse de sensibilité locale, autour d'une valeur de référence, ici la valeur « best-estimate » (ou l'espérance mathématique* $\bar{x}$ *) des variables incertaines, sans prendre en compte l'interaction entre les variables. On voit alors aisément que cette approche peut être satisfaisante dans le cas linéaire mais se révèle insuffisante si le modèle exhibe un comportement non linéaire. D'autre part, elle ne permet de calculer la sensibilité que pour des variables aléatoires (paramètres du modèle) et non pour des processus stochastiques ou matrices aléatoires.*

5.2 Calcul numérique des indices de sensibilité

Dans ce qui suit, on présente une méthode efficace (nécessitant un nombre réduit de simulations) pour l'approximation numérique des expressions S_i ainsi que S_{Ti} [bib2].

5.2.1 Expression numérique pour le calcul des indices du premier ordre

En reprenant les notations du paragraphe précédent, on peut déterminer les indices du premier ordre par l'expression :

$$\hat{S}_i = \frac{\hat{V}_i}{\hat{V}(Y)}$$

où la moyenne et la variance empiriques s'obtiennent par les estimateurs classiques

$$\hat{E}(y) = \frac{1}{N}\sum_{r=1}^{N} f(x_1^r, x_2^r, \ldots, x_m^r) , \qquad \hat{V}(y) = \frac{1}{N-1}\sum_{r=1}^{N} f(x_1^r, x_2^r, \ldots, x_m^r)^2 - \hat{E}(y)^2 ,$$

et la variance du premier ordre V_i peut être estimée par la formule [bib2] :

$$\hat{V}_i = \hat{U}_i - \hat{E}(Y)^2$$

avec

$$\hat{U}_i = \frac{1}{N-1}\sum_{r=1}^{N} f(x_1^r, x_2^r, \ldots, x_m^r)\, f(x_1^{r'}, x_2^{r'}, \ldots, x_{i-1}^{r'}, x_i^r, x_{i+1}^{r'}, \ldots, x_m^{r'}) .$$

Le calcul pratique des $\hat{U}_i$ par simulation de Monte Carlo est décrite plus en détail dans § 5.2.3..

5.2.2 Expression numérique pour le calcul des indices totaux

On peut déterminer l'indice total par l'expression :

$$\hat{S}_{Ti} = 1 - \frac{\hat{V}_{-i}}{\hat{V}(Y)}$$

où [bib2]

$$\hat{V}_{-i} = \hat{U}_{-i} - \hat{E}(Y)^2$$

et

$$\hat{U}_{-i} = \frac{1}{N-1}\sum_{r=1}^{N} f(x_1^r, x_2^r, \ldots, x_m^r)\, f(x_1^r, x_2^r, \ldots, x_{i-1}^r, x_i^{r'}, x_{i+1}^r, \ldots, x_m^r)$$

Le calcul pratique des $\hat{U}_{-i}$ par simulation de Monte Carlo est décrite plus en détail dans § 5.2.3.
Comme pour les indices du premier ordre, la moyenne $\hat{E}(Y)$ et la variance empiriques $\hat{V}(Y)$ s'obtiennent par leurs estimateurs classiques (voir ci-dessus).

5.2.3 Evaluation des expressions numériques par simulation de Monte Carlo

La démarche suivante peut être adoptée pour le calcul pratique des indices S_i et S_{Ti}. Dans un premier temps, il est nécessaire de choisir le nombre *ns* de simulations qu'on souhaite effectuer (c'est la taille de l'échantillon de base).

Remarque :

Dans le choix de la taille de base de l'échantillon (ns) il convient de vérifier que, au moins l'estimation de la variance $\hat{V}(y)$ a convergée.

Ensuite dans une boucle, on effectue à itération l, $l = 1, \ldots, ns$:

- Génération de deux réalisations de l'ensemble des variables d'entrée X_i, $i = 1, \ldots, m$ et définition de deux vecteurs contenant chacun un ensemble de m données :

$$A^l = \left(x_1^{(1)}, \cdots, x_i^{(1)}, \ldots, x_m^{(1)}\right), \qquad B^l = \left(x_1^{(2)}, \cdots, x_i^{(2)}, \ldots, x_m^{(2)}\right).$$

- Définition d'un vecteur M_i^l, contenant toutes les valeurs de A^l sauf la i[ieme] valeur qu'on prend de du vecteur B^l, et d'un vecteur M_{Ti}^l avec la i[ieme] valeur de A^l et toutes les autres valeurs comme dans B^l :

$$M_i^l = \left(x_1^{(1)}, \cdots, x_i^{(2)}, \ldots, x_m^{(1)}\right), \qquad M_{Ti}^l = \left(x_1^{(2)}, \cdots, x_i^{(1)}, \ldots, x_m^{(2)}\right).$$

- Evaluation de la fonction $f(x)$ pour les données A^l : $y^l = f(A^l)$. De même, on évalue les expressions $y_i^l = f(M_i^l)$ et $y_{Ti}^l = f(M_{Ti}^l)$ à partir de M_i et respectivement de M_{Ti}.

- Calcul des valeurs

$$\hat{U}_i^l = \frac{l-1}{l}\hat{U}_i^{l-1} + \frac{1}{l}y^l y_i^l, \qquad \hat{U}_{Ti}^l = \frac{l-1}{l}\hat{U}_{Ti}^{l-1} + \frac{1}{l}y^l y_{Ti}^l$$

ainsi que des estimateurs de la moyenne et du moment d'ordre 2 pour l'itération l :

$$\hat{E}^l = \frac{l-1}{l}y^l, \qquad \hat{m}_2^l = \frac{l-1}{l}\hat{m}_2^{l-1} + \frac{1}{l}(y^l)^2$$

Enfin, au bout des *ns* calculs, on peut évaluer la moyenne comme $\hat{E}(y) \equiv \hat{E}^{ns}$, la variance empirique totale peut être estimée par l'expression $\hat{V}(y) = \frac{ns}{ns-1}\left(\hat{m}_2^{ns} - \hat{E}(y)^2\right)$.

Avec ces informations, on peut calculer les indices de sensibilité selon les formules introduites ci-dessus, à savoir

Indice du premier ordre : $\hat{S}_i = \frac{U_i^{ns} - \hat{E}(y)^2}{\hat{V}(y)}$

Indice total : $\hat{S}_{Ti} = 1 - \frac{U_{Ti}^{ns} - \hat{E}(y)^2}{\hat{V}(y)}$

5.2.4 Exemple de mise en œuvre avec *Code_Aster*

A titre d'exemple, on choisit comme variable d'intérêt le maximum du spectre de réponse oscillateur $\omega \to SRO(\omega, \xi)$ obtenu par un calcul dynamique transitoire, cf. §4.1. On a trois sources d'incertitudes, à savoir les matrices généralisées MATM, MATK, MATD, de sorte que *m=3*. On cherche à hiérarchiser ces sources d'incertitude par rapport à la variable d'intérêt. Plus précisément, on souhaite, si possible, réduire la variabilité des sources d'incertitudes contribuant le plus à la variance totale. C'est-à-dire qu'on s'intéresse aux indices du premier ordre S_{MATM} , S_{MATK} S_{MATD} .

Dans ce qui suit on reproduit les principales lignes d'un fichier de commandes (épuré donc) permettant d'évaluer les indices de sensibilités du premier ordre.

D'abord, on initialise:

```
MATM = [None]*2
MATK = [None]*2
MATD = [None]*2
U1   = [None]*m

E0 = 0.0
V0 = 0.0
```

…

Puis, on peut écrire

```
for k1 in range(1,ns+1) :
```

génération d'une réalisation des données d'entrée (A)

```
  MATM[0]=GENE_MATR_ALEA(MATR_MOYEN=MASSE,DELTA=0.2)
  MATK[0]=GENE_MATR_ALEA(MATR_MOYEN=RIGID,DELTA=0.2)
  MATD[0]=GENE_MATR_ALEA(MATR_MOYEN=AMORT,DELTA=0.2)
```

génération de la deuxième réalisation (B)

```
  MATM[1]=GENE_MATR_ALEA(MATR_MOYEN=MASSE,DELTA=0.2)
  MATK[1]=GENE_MATR_ALEA(MATR_MOYEN=RIGID,DELTA=0.2)
  MATD[1]=GENE_MATR_ALEA(MATR_MOYEN=AMORT,DELTA=0.2)
```

Calcul de la r éponse à partir de la réalisation (*A)*

```
  DYNA=DYNA_TRAN_MODAL( MASS_GENE=MATM[0],
                        RIGI_GENE=MATK[0]
                        AMOR_GENE=MATD[0],
                          ... )
  ACC1 =RECU_FONCTION(RESU_GENE = DYNA, ... )
  SRO=CALC_FONCTION(SPEC_OSCI=_F(NATURE='ACCE', FONCTION=ACC1,...))
```

Calcul du maximum SROMA=max(SRO) :

```
TSROMAX=INFO_FONCTION (MAX=_F(FONCTION = SRO))
SROMA = TSROMAX[ 'SRO',2]
```

Evaluation de la moyenne *E0* et de la variance *V0* empirique

```
 fk1=float(k1)
E0= SROMA/fk1+ ((fk1-1.)/fk1)*E0
V0= SROMA**2./fk1+ ((fk1-1.)/fk1)*V0

DETRUIRE (CONCEPT=_F(NOM=(MATM,MATK,MATD,DYNA,SRO,ACC1)))
```

Evaluation de U_{k2} à l'itération k1

```
for k2 in range(m) :
   L1=[0]*m
   L1[k2]=1

   DYNA=DYNA_TRAN_MODAL( MASS_GENE=MATM[L1[0]],
                         RIGI_GENE=MATK[L1[1]]
                         AMOR_GENE=MATD [L1[2]] ,
                           ... )
   ACC1 =RECU_FONCTION(RESU_GENE = DYNA, ... )
   SRO=CALC_FONCTION(SPEC_OSCI=_F(NATURE='ACCE',FONCTION=ACC1,...
))

   Calcul du maximum SROMAX=max(SRO)  comme décrit ci dessus . . .

   if k1==1:
      U1[k2]= SROMAX*SROMA
   else :
      U1[k2] = SROMAX*SROMA/fk1+((fk1-1.)/fk1)*U1[k2]


   DETRUIRE CONCEPT=_F(NOM=(MATM,MATK,MATD,DYNA,SRO,ACC1)))
```

Puis, à la fin des calculs, on peut évaluer les indices de sensibilité S_{k2} :

```
For k2 in range(m) :
   S[k2]=(U1[k2]-E0**2.)/V0
```

6 Exemple en dynamique transitoire

6.1 Principe du calcul déterministe

On s'appuie sur le cas-test SDNS01a concernant la réponse d'une plaque rectangulaire avec une butée élastique soumise à un chargement impulsionnel déterministe.
A la place de la charge impulsionnelle, on considère une excitation sismique, caractérisée par un accélérogramme. L'excitation sismique peut être définie via les commandes DEFI_FONCTION [U4.31.02] et CALC_CHAR_SEISME [U4.63.01] (voir aussi [R4.05.01] : « Réponse sismique par analyse transitoire » et le cas-test SDLL109a).

On construit la solution du modèle dynamique réduit moyen (déterministe) à l'aide d'un enchaînement classique d'opérateurs (ASSE_MATRICE , MODE_ITER_SIMULT , MACRO_PROJ_BASE …. voir SDNS01a).
On s'intéresse à la réponse du système calculée sur base modale par DYNA_TRAN_MODAL [U4.53.21], et plus exactement aux spectres de réponses normalisés et aux observations temporelles (champs de déplacement, vitesse, accélération, contraintes, etc).

6.2 Principe du calcul prenant en compte les incertitudes

Les raideurs de butées sont rendues aléatoires ainsi que les matrices généralisées de masse, de raideur et d'amortissement. Par ailleurs, l'excitation sismique est modélisée par un processus aléatoire (non stationnaire), caractérisé par la donnée de *na* accélérogrammes. On a donc *m=5* sources d'incertitudes.

Les réponses transitoires sont calculées par la méthode de simulation de Monte Carlo directe. Si on choisit un échantillon de base de taille *ns*, on doit effectuer *ns* tirages de la variable aléatoire et des matrices aléatoires. Ceci se fait au fur et à mesure, dans une boucle Python, comme décrit dans les chapitres précédents. Par ailleurs, on doit disposer d'une base de donnée d'accélérogrammes (excitation sismique) dont les éléments sont supposés équiprobables. Dans la boucle Python de la simulation de Monte Carlo, on tire « au hasard » l'un de ces accélérogrammes. Si on souhaite hiérarchiser les sources d'incertitudes, le nombre d'évaluation de la fonction (c'est-à-dire le nombre de calculs dynamiques) est néanmoins plus important, voir §5.2.3.

Dans ce qui suit, on décrit l'enchaînement des calculs dans un fichier de commande *Code_Aster*. En ce qui concerne l'hiérarchisation des incertitudes, on détermine les estimateurs des indices du premier ordre uniquement (le calcul des indices totaux ne pose pas de problème supplémentaire). En total, on doit alors effectuer *ns*(m+1)* calculs numériques. Soit le maximum du spectre de réponse la grandeur d'intérêt, qui doit être scalaire.

I. On effectue les simulations à l'aide de boucles Python, dont la structure est :

Début boucle extérieure, pour *k* = 1, ... , *ns* :

Génération des deux réalisations A^k et B^k :

- Génération des *2* réalisations (A^k et B^k) des variables aléatoires modélisant les raideurs de butée à l'aide de GENE_VARI_ALEA.
- Génération des *2* réalisations (A^k et B^k) des matrices généralisées aléatoires de masse, de raideur et d'amortissement à l'aide de GENE_MATR_ALEA (approche non paramétrique).
- Tirage « au hasard » de 2 réalisations (A^k et B^k) des *ns* accélérogrammes dans la base de donné.

Boucle interne, pour k2=0,…,m (sur le nombre de sources d'incertitudes à considérer) :

Si k2==0 :

```
L1=[0]*m
```

(on prend les valeurs de la première réalisation (A^k) uniquement)

Sinon :

```
L1=[0]*m
L1[k2]=1
```

(on fait varier la source d'incertitude *k2* : on prend les valeurs de la réalisation A sauf pour la source d'incertitude *k2* pour laquelle on prend la valeur de la réalisation B^k)

1) Calcul de la $k^{ème}$ réalisation de $Q^k(t)$, solution du système matriciel classique avec les données d'entrée (matrices généralisées, raideurs, accélérogramme) précédemment définies.

1) Extraction des ddls physiques en déplacement (projection sur base physique via REST_BASE_PHYS) $x_k(t, j)$ pour un ensemble de ddls $j \in \Sigma$ prédéfinis **et construction des spectres de réponse** $\omega \to SRO_k(\omega, \xi, j)$ pour les ddls $j \in \Sigma$. Ceci peut se faire par l'enchaînement des opérateur RECU_FONCTION et CALC_FONCTION(SPEC_OSCI…) ou directement via la macro-commande MACR_SPECTRE .

Si k2==0 :

a) Evaluation à l'aide de l'opérateur CALC_FONCTION des contributions aux estimateurs des moyennes $\hat{m}_1^k(\xi,\omega;j)$ et des moments d'ordre deux pour les spectres normalisés :

$$\hat{m}_1^k(\xi,\omega;j)=\frac{SRO_k(\xi,\omega;j)}{k}+\frac{k-1}{k}\hat{m}_1^{k-1}(\xi,\omega;j);\quad \hat{m}_1^1(\xi,\omega;j)=SRO_1(\xi,\omega;j)$$

$$\hat{m}_2^k(\xi,\omega;j)=\frac{SRO_k(\xi,\omega;j)^2}{k}+\frac{k-1}{k}\hat{m}_2^{k-1}(\xi,\omega;j);\quad \hat{m}_2^1(\xi,\omega;j)=SRO_1(\xi,\omega;j)^2$$

b) Calcul du maximum du spectre de réponse en un point N1 $\omega\rightarrow SRO_k(\omega,\xi,N1)$ (c'est la grandeur d'intérêt pour l'hiérarchisation des incertitudes) via INFO_FONCTION et évaluation de la moyenne $\hat{E}^k$ et du moment d'ordre deux $\hat{m}_2^k$ (voir §5.2.3).

- c) On détermine l'indicateur caractérisant le niveau sismique (pour l'accélérogramme), ici le PGA, en utilisant la commande INFO_FONCTION(NOCI_SEISME=…).
- On détermine la contrainte maximale dans la section et on teste si la valeur admissible S_{adm} et dépassée ou non (c'est-à-dire qu'on vérifie si on a défaillance ou non). On pose valx=1 si on défaillance et valx=0 sinon.
- On écrit dans table de résultat :

```
Si k==1:
   TAB1=CREA_TABLE …
sinon:
   TAB1=CALC_TABLE(reuse =TAB1 …
```

Sinon :

Calcul du maximum du spectre de réponse en un point N1 $\omega\rightarrow SRO_k(\omega,\xi,N1)$ (c'est la grandeur d'intérêt pour l'hiérarchisation des incertitudes) via INFO_FONCTION et évaluation à l'aide de l'opérateur CALC_FONCTION des contributions aux estimateurs pour les indices de sensibilités U_{k2}^k (voir §5.2.3).

Fin de boucle interne k2 .

Fin de boucle extérieure k.

II. Evaluation la variance empirique totale $\hat{V}(y)=\frac{ns}{ns-1}\hat{m}_2^{ns}-\left(\hat{E}^{ns}\right)^2$, et des indices de sensibilité du premier ordre pour chaque source d'incertitude i: $\hat{S}_i=\frac{U_i^{ns}-\hat{E}(y)^2}{\hat{V}(y)}$.

III. En post-traitement, on peut alors déterminer les paramètres de la courbe de fragilité selon le modèle log-normale à l'aide de l'opérateur POST_DYNA_ALEA(FRAGILITE =(_F(TABL_RESU=TAB1…, (voir §4.2.1).

7 Bibliographie

[bib1] Saporta G., *Probabilités, analyse de données et statistique.* Editions Technip, 2006.
[bib2] Saltelli A., Tarantola S., Campolongo E., Ratto M, *Sensitivity analysis in practice. A guide to assessing scientific models* . W iley, 2004.
[bib3] EPRI, Seismic Probabilistic Risk Assessment Implementation Guide, Final Report 1002989, 2003.
[bib4] Lefebvre Y., de Rocquigny E., Dutfoy A., Delcoigne F., Sudret B., Cagnac A., *Guide Méthodologique pour le traitement des incertitudes*. Note EDF R&D HT-56-2007-01798, 2007.

[bib5] Cambier S. : *Quantification et hiérarchisation probabilistes des incertitudes dans la chaîne de calcul sismique d'un circuit primaire*. Note EDF R&D HT-61/06/017/A, 2006.

[bib6] Zentner I., *Méthodes probabilistes dans les analyses sismiques : Modélisation, propagation et hiérarchisation des incertitudes.* Communication au colloque AFPS 2007.