Introduire une nouvelle loi non-linéaire localisée dans DYNA_VIBRA

Résumé :

DYNA_VIBRA permet de réaliser un calcul de réponse transitoire ou harmonique.

Ce document décrit la méthode pour introduire une nouvelle loi non-linéaire localisée dans cet opérateur.

Cette possibilité est uniquement disponible sur un calcul transitoire sur base réduite appelée par abus de langage une base modale.

Table des Matières

1 Introduction....3
2 Rappel de la démarche de calcul sur base réduite....3
3 Démarche de programmation....3
3.1 Préparation des données....4
3.2 Evaluation de la force, résolution et archivage des résultats à chaque pas de temps....4
4 Introduction d'une nouvelle loi non-linéaire....4
5 Règles de programmation....5
6 Activation de la nouvelle loi non linéaire....6

1 Introduction

L'opérateur DYNA_VIBRA permet de réaliser un calcul de réponse transitoire ou harmonique sur base physique ou base réduite (appelée base modale par abus de langage).
Il est dédié aux modèles linéaires, mais une particularité a été introduite pour l'étude des structures possédant des non-linéarités localisées où on fait l'analyse sur une base réduite, censée représenter le comportement de la structure dans la bande de fréquence d'intérêt.
La prise en compte de cette non-linéarité est uniquement disponible pour un calcul de réponse transitoire sur base modale. Dans *Code_Aster*, cela correspond à BASE_CALCUL = 'GENE' et TYPE_CALCUL = 'TRAN'.

2 Rappel de la démarche de calcul sur base réduite

La démarche pour le calcul de réponse transitoire sur base modale est la suivante.

- Définition de la base de projection
- Projection des matrices du système sur cette base
- Evaluation de la force non-linéaire (force extérieure : second membre)
- Projection de la force extérieure sur la base de réduction
- Résolution du système sur base réduite
- Restitution sur base physique

La particularité de la résolution se situe au niveau de la projection de la force sur la base réduite. En effet, dans le cas non-linéaire, cette force peut dépendre du déplacement, de la vitesse voire de l'accélération à l'instant de calcul. Ici, on a donc besoin, à chaque pas de calcul, de récupérer ces informations définies sur la base physique. Cela nécessite alors un retour sur base physique à partir des coordonnées sur base réduite et ensuite une projection de l'effort non-linéaire calculé sur la base de réduction.

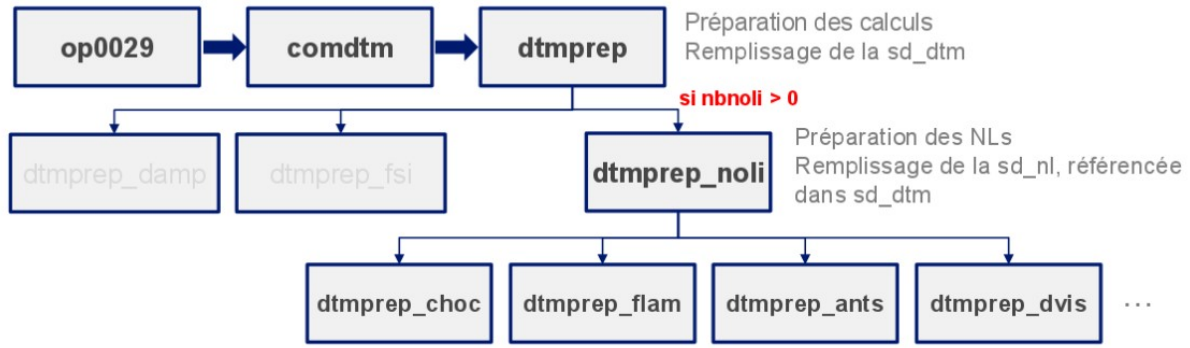
3 Démarche de programmation

La démarche de programmation se déroule en deux étapes.
- La première étape consiste à saisir tous les paramètres nécessaires pour le calcul.
- La deuxième étape concerne l'estimation de la force non-linéaire. Cette estimation est réalisée à chaque pas de temps de calcul en sommant toutes les forces non-linéaires définies.

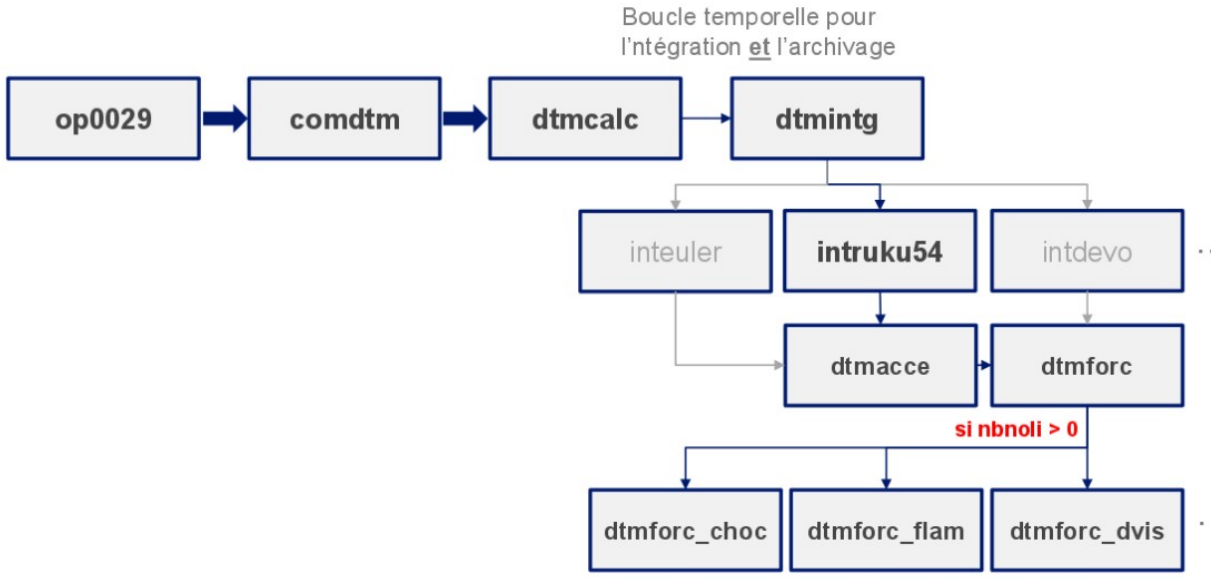
Pour chaque type de non-linéarité, on saisit des données nécessaires pour le calcul (première étape) avec dtmprep_noli_*. Le calcul de la force non-linéaire et l'archivage des variables internes nécessaires pour le traitement sont réalisés avec dtmforc_*.

Les organigrammes suivants présentent la démarche adoptée.

3.1 Préparation des données



3.2 Evaluation de la force, résolution et archivage des résultats à chaque pas de temps



- A la sortie de dtmforc est le cumul des efforts externes (EXCIT) et internes venant des non linéarités localisées
- Les variables internes sont stockés dans un vecteur dédié de la sd_nl : INTERNAL_VARS

I 8

4 Introduction d'une nouvelle loi non-linéaire

Une structure de données temporaire `sd_nl` a été créée afin de pouvoir stocker tous les paramètres nécessaires au calcul.

Plusieurs utilitaires sont mis à disposition :
- `nlinivec` : permet d'initialiser un vecteur à zéro
- `nlsav` : permet de stocker les paramètres de calcul dans la structure de données `sd_nl`

- nlget : permet de récupérer les informations stockées dans sd_nl
- togene : projection d'un vecteur sur base réduite
- tophys : expansion d'un vecteur généralisé sur base physique
- tophys_ms : expansion sur base physique avec prise en compte du mouvement d'entrainement (calcul multi-appui)

Pour introduire une nouvelle loi non-linéaire, le développeur n'a donc besoin que d'écrire la routine de préparation des données : dtmprep_noli_*newf*.F90 et celle qui réalise le calcul et l'archivage des variables internes : dtmforc_*newf*.F90 (sans oublier d'écrire les fichiers d'interface dtmprep_noli_*newf*.h et dtmforc_*newf*.h).

Par exemple pour la définition de la relation effort – déplacement (RELA_EFFO_DEPL), on a seulement besoin de dtmprep_noli_rede.F90 et de dtmforc_rede.F90. On peut s'en inspirer pour l'écriture d'une nouvelle loi.

On met ensuite à jour :
- le fichier nldef.h pour la définition de la non-linéarité – associer un entier au type de la non-linéarité (NL_*NEWF*) et définir à priori le nombre de variables internes à stocker (NBVARINT_*NEWF*),
- le fichier nlinc.h pour la définition des paramètres de calcul (dans le cas où l'ajout de nouveaux paramètres dans la sd_nl s'avère nécessaire),
- le catalogue de commande dyna_vibra.capy (mot clé COMPORTEMENT/RELATION).

5 Règles de programmation

Si le développeur a le libre choix dans l'utilisation et le nommage de ses paramètres de travail dans la structure de données dédiée (sd_nl), il reste toutefois obligatoire de respecter les règles suivantes afin de permettre aux processus d'archivage et de reprise de calcul de fonctionner correctement avec la nouvelle non-linéarité.
- Le paramètre MX_LEVEL de la sd_nl doit être incrémenté dans la routine préparant les données dtmprep_noli_*newf*. Cela permet d'éviter l'écrasement de données lors de la combinaison de plusieurs non-linéarités.
- Le paramètre NB_NONL de la sd_dtm doit également être incrémenté dans dtmprep_noli_*newf*. Cela permet d'activer globalement le calcul de forces non linéaires et l'archivage des variables internes associées.
- Le paramètre NL_TYPE de la sd_nl doit être sauvegardé avec la valeur entière décrivant la non-linéarité traitée dans dtmprep_noli_*newf*. Cette valeur est définie dans le fichier nldef.h (par ex. NL_CHOC = 1, NL_BUCKLING = 2, etc.). Le paramètre NL_TYPE est requis dans dtmforc afin de dispatcher correctement vers la routine dtmforc_*newf*. L'utilisation d'un entier permet d'optimiser cette étape très répétitive dans la boucle d'intégration.
- Les variables internes (réels) sont concaténées pour toutes les non linéarités et stockées sous le paramètre global INTERNAL_VARS de la sd_nl. Afin de ne pas écraser les variables des autres non linéarités, il est nécessaire de récupérer l'indice de sauvegarde généré et stocké automatiquement sous le paramètre INTERNAL_VARS_INDEX. En pratique :

```
call nlget (sd_nl, _INTERNAL_VARS      , vr= vint )
call nlget(sd_nl, _INTERNAL_VARS_INDEX, vi=vindx)
start = vindx(nl_ind)
vint(start+0) = valeur1
vint(start+1) = valeur2
vint(start+2) = valeur3
```

6 Activation de la nouvelle loi non linéaire

Une fois les routines de préparation et de calcul de la force non linéaire créées (dtmprep_noli_newf et dtmforc_newf), il est nécessaire de les activer en rajoutant un « case » :

- Dans dtmprep_noli, dispatchant vers dtmprep_noli_newf en fonction de la valeur du mot-clé RELATION sous le mot clé facteur COMPORTEMENT de DYNA_VIBRA.
- Dans dtmforc, routine responsable du calcul de la force, en fonction de la valeur du paramètre NL_TYPE, permettant de passer par dtmforc_newf.