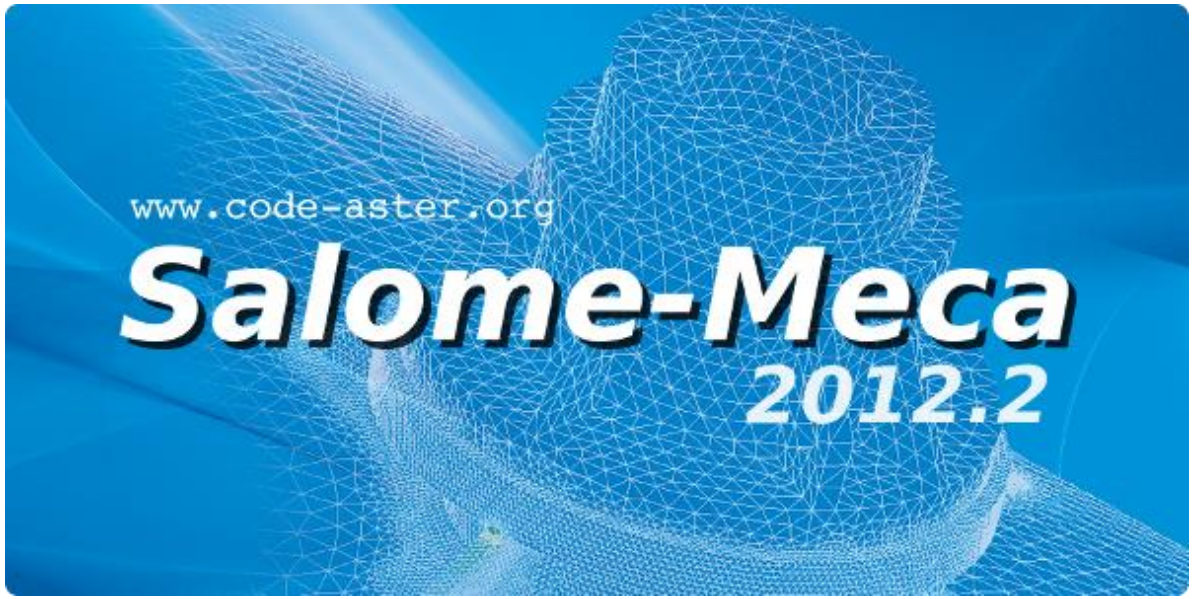
モーダル解析



Code_Aster, Salome-Meca course material

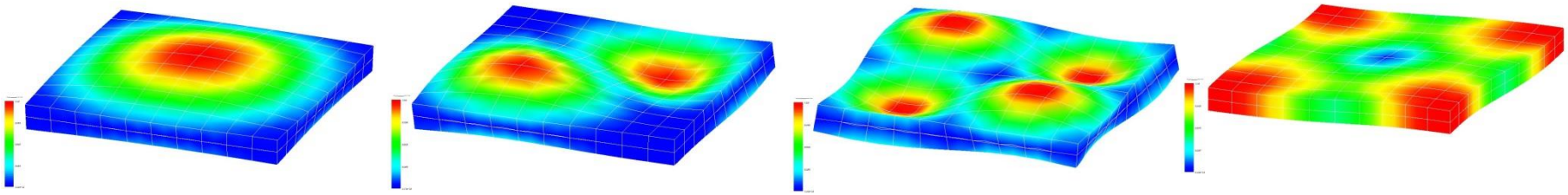
概要

- 導入
- 固有値問題とは
- 解法
- Code_Asterでの実装
- ヒント

導入

モーダル解析とは

- 固有値モードとは



- モーダル解析を行う目的

固有値問題とは

- 2種類の問題
  - 一般化された固有値問題(減衰・外力がない状態)

$$M\ddot{u} + Ku = 0 \quad \Rightarrow \quad (K - \lambda M)\,\Phi = 0$$

  - 2次固有値問題(減衰・外力を考慮)

$$M\ddot{u} + C\dot{u} + Ku = 0 \quad \Rightarrow \quad \underbrace{(K + \lambda C + \lambda^2 M)\,\Phi = 0}_{\text{線形化}}$$

$$\underbrace{\left(\begin{bmatrix} C & K \\ K & 0 \end{bmatrix}\right.}_{\mathbf{K}_{quad}} + \lambda \underbrace{\left.\begin{bmatrix} M & 0 \\ 0 & -K \end{bmatrix}\right)}_{\mathbf{M}_{quad}} \begin{bmatrix} \lambda\Phi \\ \Phi \end{bmatrix} = \mathbf{0}$$

- 未知数
  - 固有値
  - 固有モード

$$0 \le \lambda_1 \le \lambda_2 \le \ldots \le \lambda_n \quad (\lambda = \omega^2)$$

$$\Phi_1, \Phi_2, \Phi_3, \ldots, \Phi_n$$

解決方法

- モーダル計算方法は、4つの主なファミリーに区分されている:
  - Powers法（べき乗法）
    - 範囲：スペクトルの極値計算
    - 長所：シンプルで，小数回の反復で固有ベクトルの良好な推定が可能
    - 短所：収束の問題，複合モードの特定が貧弱
    - Code_Aster : **MODE_ITER_INV**
  - 部分空間法 (Jacobi, Lanczos, Arnoldi)
    - 範囲：スペクトルの一部の計算
    - 長所：問題のサイズを小さくし，メモリが少なくて済む
    - 短所：収束の問題，多くの処理と後処理が必要
    - Code_Aster : **MODE_ITER_SIMULT**
  - QR/QZ タイプの方法
    - 範囲：全スペクトルの計算
    - 長所：良好な収束性，ロバスト性
    - 短所：メモリとCPUコスト
    - Code_Aster : **MODE_ITER_SIMULT**
  - その他
    - 多かれ少なかれ経験的，専門的
    - Code_Aster : **MODE_ITER_SIMULT(bisection)**

Powers法（べき乗法）

- 標準的な問題 $Ax = \lambda x$
- 最大の固有値を決定する λ_1

アルゴリズム

Initial vector v_0

For i = 1, ...

$w = A v_{i-1}$

$\lambda^i = (Aw, w)$

$v_i = \frac{w}{\lambda^i}$ if $\lambda^i \neq 0$

$(\lambda^i, v_i) \rightarrow (\lambda_1, u_1)$

証明

固有モードと固有値

$(\lambda_j, u_j)_{j=1,n}$ eigenmodes and eigenvalue s

$|\lambda_1| > |\lambda_2| > |\lambda_3| > \ldots$

$v_0 = \sum_j (v_0, u_j) u_j$

$A^k v_0 = \lambda_1^k (v_0, u_1) u_1 + \sum_{j=2,n} \left(\frac{\lambda_j}{\lambda_1} \right)^k (v_0, u_j) u_j$

↑ 可能なら，$\neq 0$

↑ 収束率

Powers法（べき乗法）

- 最小固有値 A^{-1}
- σ に最も近い固有値 $(A-\sigma I)^{-1}$ → 移動と逆行列
- レイリー商

$$r(x)=\frac{x^T Ax}{x^T x}\ ;\ x\text{vp} \Rightarrow r(x)=\lambda$$

アルゴリズム

初期正規化ベクトル

Initial normalized vector v_0

For i = 0, ...

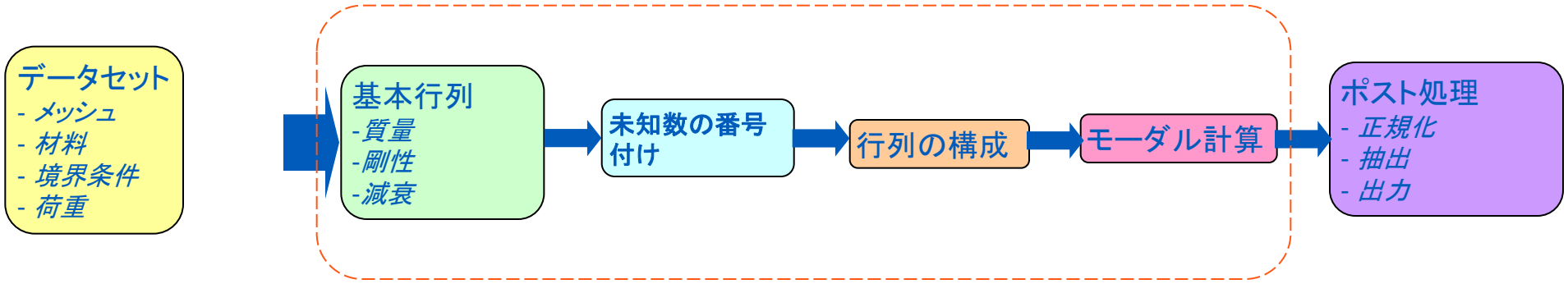
$\sigma_i = r(v_i)$;

Solve $(A - \sigma_i I)w = v_i$

$$v_{i+1} = \frac{w}{\|w\|}$$

Code_Asterの実装

- モーダル解析の主なステップ



- モーダル計算のためのCode_Aster 解法コマンド
  - **MODE_ITER_INV** 逆反復法を使用する
  - **MODE_ITER_SIMULT** 部分空間法を使用する
  - **MODE_ITER_CYCL** 周期的対称性を有する構造のモーダル計算
  - **MACRO_MODE_MECA** 実在する固有モードの計算
  - **CALC_MODAL** モーダル解析計算の完全な連鎖

解法コマンド：MODE_ITER_INV

- 文法

```
mode   = MODE_ITER_INV ( MATR_A = ...     MATR_B = ....
                 CALC_FREQ =_F ( OPTION = / 'PROCHE'   / 'SEPARE' / 'AJUSTE'
                                   FREQ = ( ....)
                                   NMAX_ITER_* =
                                   PREC_* =    )
                 CALC_MODE =_F ( OPTION =/ 'DIRECT' / 'RAYLEIGH'
                                   NMAX_ITER = ....
                                   PREC = ...                  )  )  ;
```

閉じる　分離　調整

レイリー波？

nmax_inv →

ε_{inv} →

解法コマンド：MODE_ITER_INV

- 備考
  - アルゴリズムが複雑：多くの分解
  - `OPTION = 'PROCHE'` は複合モードを計算しない (モーダル位置は不確実)
  - 固有値を計算するのに便利
  - 別のアルゴリズムによって見つけられた固有値（および固有ベクトル）の計算を精緻化
- 検証
  - 固有モードの誤差ノルム

解法コマンド：MODE_ITER_INV

- メッセージ出力ファイル

```
mode = MODE_ITER_INV (
            MATR_A = rigi ,  MATR_B = masse
            CALC_FREQ  =_F( OPTION  =  'AJUSTE'
            FREQ  = ( 5.,  10.,  15.,  20.,  24.,  27.,  30.,  32. ) ) )
```

周波数

VERIFICATION DU SPECTRE DE FREQUENCES (SUITE DE STURM)

周波数スペクトラム

LE NOMBRE DE FREQUENCES DANS LA BANDE (5.00000E+00, 3.20000E+01) EST 8

---バンドの周波数---

CALCUL MODAL: METHODE D'ITERATION INVERSE

モーダル計算：逆反復解析

			DICHOTOMIE	SECANTE		INVERSE	
NUMERO	FREQUENCE (HZ)	AMORTISSEMENT	NB_ITER	NB_ITER	PRECISION	NB_ITER	PRECISION
1	5.52739E+00	0.00000E+00	0	8	2.14267E-08	3	2.78546E-07
2	1.08868E+01	0.00000E+00	0	6	3.81407E-05	3	7.61278E-09
3	1.59155E+01	0.00000E+00	0	6	3.89777E-06	3	1.49518E-10
4	2.04606E+01	0.00000E+00	0	5	2.02615E-05	3	1.11688E-13
5	2.43840E+01	0.00000E+00	0	5	2.25410E-07	3	4.82059E-13
6	2.75664E+01	0.00000E+00	2	4	1.26633E-06	3	2.66454E-15
7	2.99113E+01	0.00000E+00	2	5	4.28255E-07	5	6.72507E-07
8	3.13474E+01	0.00000E+00	0	8	7.93951E-05	3	1.39337E-10

解法コマンド：MODE_ITER_SIMULT

- 最小周波数を計算するには
  - OPTION='PLUS_PETITE'
- またはバンド内のすべての周波数
  - OPTION='BANDE'
- 指定した周波数の近傍
  - OPTION='CENTER'
- 3つの方法
  - 'JACOBI'　Bathe and Wilson 法
  - 'TRI_DIAG'　Lanczos 法
  - 'SORENSEN'　Arnoldi 法の発展

解法コマンド：MODE_ITER_SIMULT

- 部分空間における3つの反復法
  - **MODE_ITER_INV** は逆反復と同じ原理であるが，
    部分空間⇒ではArnoldi法の方が効率的

- 手法の選択
  - **SORENSEN**
    - 高速で，低CPUコスト
    - ロバスト
    - 複合モードを計算する
    - 剛体モードを得る (**OPTION = 'BANDE** で下限を0に)

解法コマンド：MACRO_MODE_MECA

- より小さな部分区間に分解することでバンド内の周波数を計算する
- CPU時間とメモリを節約し，収束を促進する
- **MASS_** が低い固有モードを除く

```
For i = 1, number of intervals
        mode_i = MODE_ITER_SIMULT ( ... ) ;
        mode_i = NORM_MODE ( reuse, ... ) ;
End loop
EXTR_MODE ( ...) ;
```

解法コマンド：CALC_MODAL

- コマンドで一連のモーダル解析を完遂する
  - **CALC_MATR_ELEM** (質量，剛性，減衰)
  - **NUME_DDL**
  - **CALC_MATR_ASSE** (質量，剛性，減衰)
  - **MODE_ITER_SIMULT** (ダイナミックモード)
- シンプルな文法と簡単なデータ設定
- 流体相互作用のない固体構造のモーダル計算に特化
- すべてのモードを含む一つの結果を生成

解法コマンド：CALC_MODAL

```
MELR=CALC_MATR_ELEM(OPTION='RIGI_MECA',
                    MODELE=MO,
                    CHAM_MATER=CHMAT,
                    CARA_ELEM=CARELEM,
                    CHARGE=CH1,);

MELM=CALC_MATR_ELEM(OPTION='MASS_MECA',
                    MODELE=MO,
                    CHAM_MATER=CHMAT,
                    CARA_ELEM=CARELEM,
                    CHARGE=CH1,);

MELA=CALC_MATR_ELEM(OPTION='AMOR_MECA',
                    MODELE=MO,
                    CHAM_MATER=CHMAT,
                    CARA_ELEM=CARELEM,
                    CHARGE=CH1,);

NUM=NUME_DDL(MATR_RIGI=MELR,);

MATASSR=ASSE_MATRICE(MATR_ELEM=MELR,
                                          NUME_DDL=NUM,);

MATASSM=ASSE_MATRICE(MATR_ELEM=MELM,
                                          NUME_DDL=NUM,);

MATASSA=ASSE_MATRICE(MATR_ELEM=MELA,
                                          NUME_DDL=NUM,);

MODES=MODE_ITER_SIMULT(MATR_A=MATASSR,
                                              MATR_B=MATASSM,
        MATR_C=MATASSA,
       METHODE='TRI_DIAG',

CALC_FREQ=_F(OPTION='CENTRE',

FREQ=20,

NMAX_FREQ=10),);
```

```
MODES=CALC_MODAL(MODELE=MO,
       CHAM_MATER=CHMAT,
                                CARA_ELEM=CARELEM,
                                CHARGE=CH1,
       AMORTISEMENT='OUI',
        METHODE='TRI_DIAG',

CALC_FREQ=_F(OPTION='CENTRE',

FREQ=20,

NMAX_FREQ=10),);
```

データ構造内の蓄積

- **NUME_ORDRE**
  - mode_mecaのデータ構造では，昇順にモードを列記
    NUME_ORDRE はデータ構造内での位置

- **NUME_MODE**
  - スペクトル全体のモード位置

NUME_ORDRE	NUME_MODE	FREQ
1	23	2.51972E+01
2	24	2.63652E+01
3	25	2.78854E+01
4	26	2.79978E+01
5	27	2.89328E+01

データ構造内の蓄積

- 一般化された質量または剛性

$$m_x = x^T M x k_x = x^T K x \omega = \frac{k_x}{m_x}$$

$$v(t) = \sum_i \alpha_i(t) x_i$$

$$M\ddot{v} + Kv = f \quad \Rightarrow \quad m_{x_i}\ddot{\alpha}_i + k_{x_i}\alpha_i = f_i$$

- 実効モーダル質量(unit) **MASS_EFFE_ (UN_) D ***

$\Re^3$の中で正規化されたベクトルU_d

$$m_{x,d} = \frac{\left(x^T M U_d\right)^2}{x^T M x}$$ 有効モーダル質量(レイリー商)

特性 : $\sum_i m_{x_i,d} = \text{mass}_{\text{structure}}$

$$\overline{m}_{x,d} = \frac{1}{\text{mass}_{\text{structure}}} m_{x,d}$$ 単位あたりの有効モーダル質量

データ構造内の蓄積

- 刺激係数 **FACT_PARTICI_D ***

$$p_{x,d} = \frac{\left(x^T M U_d\right)}{x^T M x} \quad \text{刺激係数}$$

$$\text{特性}: \sum_i \left(p_{x_i,d}\right)^2 m_{x_i} = \text{mass}_{\text{structure}}$$

検証

- 各手法の収束性
- 固有モードに対する事後ノルム誤差

$$\text{if } |\lambda| > \omega^2_{\text{solid body movement}}$$

$$error = \frac{\|Ax - \lambda Bx\|}{\|Ax\|} \text{ if } error \leq threshold \text{ OK}$$

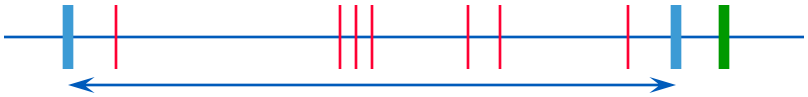
else < F >

else

$$erreur = \|Ax - \lambda Bx\| \text{ if } error \leq threshold \text{ OK}$$

else < F >

- Sturm検証



検証後の周波数範囲

ユーザが求める周波数

ユーザは求めなかったが，計算された周波数

後処理コマンド：NORM_MODE

- ノルム
  - デフォルトでは：最大物理的 DOF が 1 になる
  - **NORME = 'MASS_GENE'** $x^T M\ x = 1$
  - **NORME = 'RIGI_GENE'** $x^T K\ x = 1$
  - DOFのグループ内で得られた最大物理的 DOF が 1 になる
    - **NORME = 'TRAN'** **(DX,DY,DZ)**
    - **NORME = 'TRAN_ROTA'** **(DX,DY,DZ, DRX,DRY,DRZ)**
    - **NOEUD = no** and **NOM_CMP=DOF** **DOF(no) = 1**
    - **SANS_CMP = (CMP1, ...)** DOFを持たないグループ
    - **AVEC_CMP = (CMP1, ...)** DOFで構成されるグループ

後処理コマンド：IMPR_RESU

- モーダルパラメータのテーブルのみを出力

```
IMPR_RESU ( RESU = _F( RESULTAT =mode_meca
                       TOUT_PARA = 'OUI'
                       TOUT_CHAM = 'NON'   ) )
```

- すべての固有モードを出力

```
IMPR_RESU (     FORMAT='GMSH'
                 RESU =_F( RESULTAT = modes   ) )
```

前/後処理コマンド：IMPR_STURM

- バンド内の周波数の数を決定

```
IMPR_STURM      (
        MATR_A = RIGI    ,    MATR_B = MASSE
        FREQ_MIN = f1    ,    FREQ_MAX =f2
        )
```

- MACRO_MODE_MECA のためのサイズ間隔
  - 間隔：10 から 20 の周波数

後処理コマンド：IMPR_STURM

- MODE_ITER_ *によって生成された mode_meca データ構造を連結

- 以下を削除して，モードを並び替える：
  - MASS_GENE < しきい値
  - MASS_EFFE_UN < しきい値
  - NUME_ORDRE または NUME_MODE を指定

- MASS_ * の合計値を表示する

- 連結される mode_meca の中に存在するモードの検出

ヒント

- モデルの重さ (**POST_ELEM**)
- 周波数の数を見積る (**IMPR_STURM**)
- **'BANDE'** オプションを使用
- 計算する周波数の数が大きい場合は，**MACRO_MODE_MECA** を使用
- **<F>** または **<A>** となるエラーメッセージ，警告メッセージを考慮
  - ゼロピポッド，並進を検知
  - モードでの重要なノルム誤差
  - STURM検証と整合しない数の周波数が計算されている場合

End of presentation

Is something missing or unclear in this document?

Or feeling happy to have read such a clear tutorial?

Please, we welcome any feedbacks about Code_Aster training materials.

Do not hesitate to share with us your comments on the Code_Aster forum dedicated thread.