

# TP4 – MODELISATION NUMERIQUE DE LA TOUR DE TAIPEI

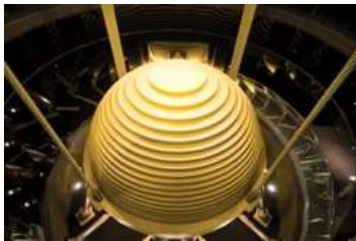
## OBJECTIFS :

- Résoudre un problème conduisant à la résolution approchée d'une équation différentielle du second ordre à l'aide d'un schéma d'Euler explicite.
- Mettre en évidence l'impact du pas de discrétisation sur la qualité des résultats.
- Comparer cette résolution avec les résultats obtenus en utilisant une bibliothèque d'intégration numérique.
- Mettre en place une interface graphique

## 1. .PRESENTATION

La Tour Taipei 101 de 508 mètres a été inaugurée à Taiwan début 2004 et est restée le plus haut gratte-ciel du monde jusque fin 2007 à l'inauguration du Burj Dubaï (512,10 m). Cette tour qui pèse 700 000 tonnes a été définie par ses concepteurs comme « un majestueux bambou bleu turquoise ». Le but étant que cette tour soit suffisamment souple pour résister aux typhons et aux tremblements de terre fréquents sur l'île de Taiwan.

En outre la tour de 101 étages a été équipée d'une boule d'acier de 660 tonnes suspendue au 92e étage de la tour. Afin d'amortir 30 à 40 % des mouvements de l'édifice qui pourraient être causés par des vents violents dus aux typhons, un tremblement de terre ou une collision avec un aéronef léger la sphère peut se déplacer jusqu'à une amplitude maximum de 1,5 mètre.

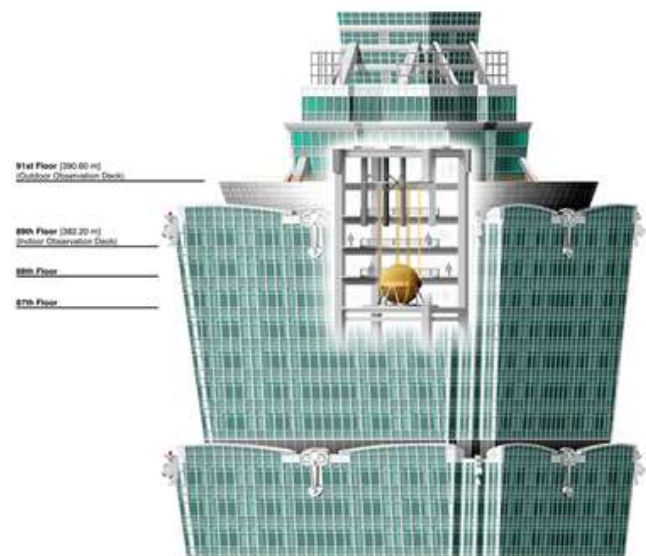
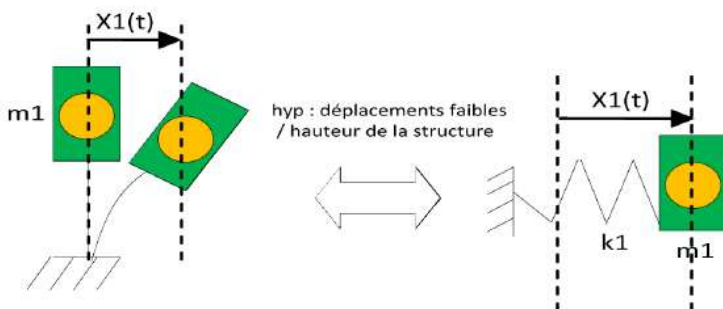


En effet la tour a été étudiée pour résister à des vents soufflant à 216 kilomètres/heure et à un tremblement de terre de 7 sur l'échelle de Richter. Lors de vents très violents le dernier étage se déplace jusqu'à 3m. Dans ce TP un modèle de la tour va être réalisé avec et sans amortisseur. On pourra alors comparer les deux et vérifier que pour une sollicitation donnée les vibrations sont bien amorties.

## 2. BATIMENT SANS AMORTISSEUR

### 2.1. Modélisation

On considère dans un premier temps que la boule est fixe au sommet de la tour ; elle ne joue pas le rôle d'amortisseur dans ce cas :



La tour est un système complexe dont les vibrations peuvent être très variées. Cependant la réduction à un modèle de type masse ressort permet d'obtenir rapidement une estimation de la vibration de la tour pour sa fréquence fondamentale (déplacements les plus importants). Il correspondra au degré de liberté du système masse ressort équivalent. On choisit pour cela le déplacement du sommet de la tour comme déplacement représentatif de l'ensemble de la structure.

La tour sans la boule est assimilée à une poutre encastree dans le sol et qui travaille en flexion : on peut montrer que pour de petits déplacements elle a un comportement proche d'un ressort de raideur  $k_1$ . Ainsi en notant  $m_1$  la masse équivalente du bâtiment et  $x_1$  le déplacement transversal du sommet de la tour, nous obtenons le modèle simplifié suivant par application du PFD à la masse ponctuelle :  $m_1 \cdot \ddot{x}_1(t) = -k_1 \cdot x_1(t)$

On modélise une bourrasque de vent très violent par un déplacement initial  $x_1(t=0) = x_0$  du sommet de la tour et une vitesse nulle  $\dot{x}_1(t=0) = 0$ . On choisit comme valeur du déplacement initial, la valeur donnée de 0.25 m. On verra dans la suite que la sphère atteint son maximum d'amplitude pour cette valeur. En fait en cas de dépassement du déplacement maximum de la sphère, un système de blocage empêche la sphère de heurter les murs. On cherche alors à déterminer l'évolution de la position de la tour au cours du temps après cette sollicitation.

On va donc procéder à une intégration numérique afin de résoudre l'équation différentielle.

On ne représente pas le bâtiment par une masse ponctuelle de 700000 tonnes à une hauteur de 508m car cette masse est répartie sur toute la hauteur de la structure donc la masse équivalente est plus faible. On choisit  $m_1 \approx 264000t$  et  $k_1 \approx 225000000N \cdot m^{-1}$  afin que le modèle ait un comportement équivalent au système étudié (ces valeurs sont justifiées en annexe).

Q1. En posant un vecteur  $X = \begin{pmatrix} x_1 \\ \dot{x}_1 \end{pmatrix}$  montrer que l'équation obtenue peut se mettre sous la forme matricielle  $X' = K \cdot X$  (on pourrait également le mettre sous la forme  $X' = F(X)$  avec  $F$  une fonction de  $\mathbb{R}^2$  dans  $\mathbb{R}^2$ )

## 2.2. Résolution numérique avec un schéma d'Euler explicite

**Créer un fichier TaipeiExercice1.py.**

Q2. La première partie du programme permet d'importer les deux bibliothèques nécessaires à la suite :

```
import numpy as np #Bibliothèque utilisée pour le calcul
import pylab as pl #Bibliothèque utilisée pour les affichages

#Equations différentielles ordinaires (EDO) (bibliothèque directement intégrée)
from scipy.integrate import odeint
```

Notamment NUMPY qui servira à simplifier le calcul matriciel.

Q3. Mettre la liste des paramètres et notamment :

```
temps1=240 # Temps maximum d'étude
nbpt=2000 # Nombre de points pour discrétiser le temps
```

Q4. Afin de pouvoir réutiliser la définition de la dérivée plusieurs fois, définir une fonction pour calculer la dérivée  $X'$  à partir de  $X$

```
def dx(X, t):
    #fonction qui calcule la dérivée et la dérivée seconde (...à l'aide du np.dot(K,X))
```

On utilisera `np.array([])` pour créer des matrices. ATTENTION, ici le vecteur  $X$  sera noté comme une matrice ligne sur python  $x=[x_1, \dot{x}_1]$

Tester d'abord tester la fonction `np.dot` sur une matrice exemple :

```
M=np.array([[0,1],[2,3]])
X=np.array([0,1])
B=np.dot (M[1,:],X)
np.dot(M[1:],0.1)
```

Avec le type `array` il est donc possible d'extraire des sous-matrices, notamment des lignes et des colonnes d'une matrice et `np.dot` permet de multiplier la matrice avec un nombre ...

Remarque : La fonction `dx` n'a pas besoin de l'argument  $t$  mais il sera nécessaire lors de l'utilisation de Scipy

Pour résoudre le problème  $X' = K \cdot X$  on va utiliser dans la suite **un schéma d'Euler explicite**. Il va être nécessaire de calculer à chaque itération le terme au pas suivant. Pour cela un calcul de la dérivée pour chacun des différents pas sera nécessaire. Celle-ci fait appel à la notation matricielle définie précédemment.

*Q5. Créer un tableau (linspace ou une liste) qui découpe l'intervalle de temps avec un pas régulier en utilisant le nombre de points défini au départ.*

On va stocker les résultats dans une matrice. Les données de chaque point sont stockées sur une ligne. Pour chaque point on stocke la position et sa dérivée dans deux colonnes différentes et on définit la position initiale.

Pour utiliser le schéma d'Euler explicite on définit la position au pas  $i + 1$  à partir de la position au pas  $i$  et de sa dérivée en  $i$ .

*Q6. Créer la matrice de zéros  $x$*   
*Q7. Puis calculer  $x[i+1, :]$*

**On souhaite afficher les résultats dans la dernière partie**

*Q8. Utiliser la fonction **plot** pour afficher les positions en fonction du temps.*

*Q9. Faire une résolution sur 60 secondes avec un nombre de points de 200000, 20000 et 2000 et conclure (stabilité, précision...).*

Pour comparer la précision à la méthode Scipy de Python on pourra utiliser les instructions suivantes

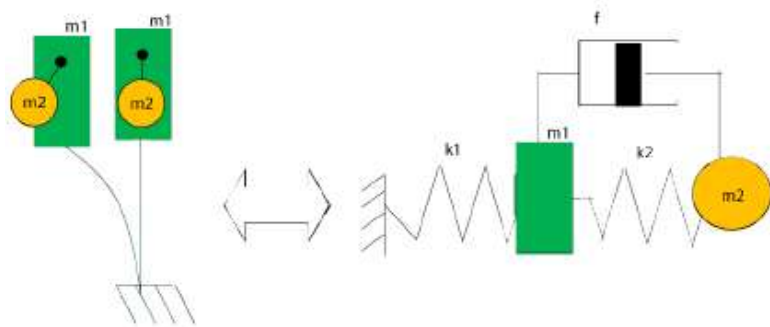
```
# Résolution avec Scipy . integrate pour comparer la méthode d'intégration propre à Python
x0_scipy = [x0, 0]
x_scipy = odeint(dx, x0_scipy, t)
```

*Q10. Puis tracer la courbe d'une autre couleur ...*

### 3. BATIMENT AVEC AMORTISSEUR

#### 3.1. Modélisation

On considère un deuxième modèle du bâtiment avec son amortisseur : la boule de masse  $m_2$  n'est plus fixe au sommet de la tour mais liée par un ressort de raideur  $k_2$  mis en parallèle avec un amortisseur de coefficient de frottement visqueux  $f$ .



Les données choisies pour que le modèle ait un comportement équivalent au système étudié sont justifiées en annexes. On choisit  $m_2 \approx 660t$ ,  $k_2 \approx 510000N \cdot m^{-1}$  et  $f \approx 52000 Kg \cdot s^{-1}$

La tour sans la boule est toujours représentée par un système de masse  $m_1$  et ressort de raideur  $k_1$ . Les déplacements  $x_1$  et  $x_2$  sont alors liés au déplacement initial  $x_0$  par le système d'équations suivant :

$$\begin{cases} m_1 \ddot{x}_1 = -k_1 \cdot x_1 + k_2 \cdot (x_2 - x_1) \\ m_2 \ddot{x}_2 = -k_2 \cdot (x_2 - x_1) - f \cdot (\dot{x}_2 - \dot{x}_1) \end{cases}$$

Les conditions initiales sont :  $x_1(t = 0) = x_0, x_2(t = 0) = 0, \dot{x}_1(0) = \dot{x}_2(0) = 0$

*Q11. Montrer que ce système d'équation peut se mettre sous la forme matricielle ci-contre.*

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \ddot{x}_1(t) \\ \ddot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{-(k_1+k_2)}{m_1} & \frac{k_2}{m_1} & 0 & 0 \\ \frac{k_2}{m_2} & \frac{-k_2}{m_2} & \frac{f}{m_2} & \frac{-f}{m_2} \end{bmatrix} \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \\ \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix}$$

### 3.2. Résolution numérique

#### ouvrir le fichier TaipeiExercice2.py

Les premières lignes (partie <— Importation des bibliothèques —>) qui permettent d'importer les bibliothèques nécessaires à la suite sont les mêmes que précédemment.

**Q12.** Ajouter les données concernant l'amortisseur, dans la partie <— Mise en donnée avec  $m_2, k_2$  et  $f$ .

Le problème à résoudre n'est plus le même, donc l'expression de la **fonction  $dx$**  donnant la dérivée est à modifier (Partie <— Écriture de la dérivée —>).

**Q13.** Compléter le vecteur  $X1$  et la matrice  $K$  puis retourner les dérivées et dérivées secondes  $dx1, dx2, ddx1, ddx2$  en sortie de fonction.

Dans la partie <— Résolution —> le schéma de résolution ne change pas cependant la matrice permettant de stocker les déplacements change de taille. En effet pour chaque pas  $i$  on stocke le déplacement de la tour et sa dérivée, mais aussi le déplacement de l'amortisseur et sa dérivée [ $x_1(i), x_2(i), \dot{x}_1(i), \dot{x}_2(i)$ ].

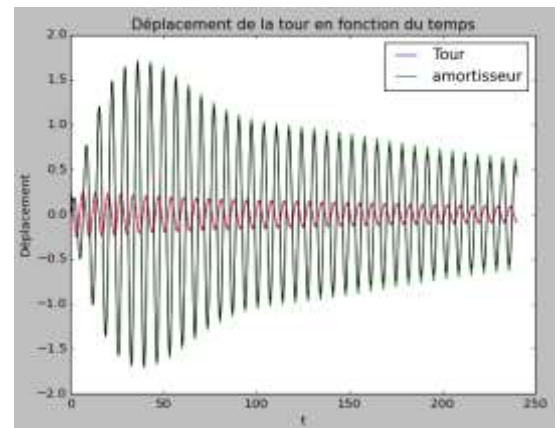
# Résolution avec euler explicite  
< A COMPLETER >

# Résolution avec Scipy . integrate pour comparer la méthode d'intégration propre à Python  
< A COMPLETER >

**Q14.** Compléter les lignes indiquées de la partie résolution

Dans la partie <— Affichage des résultats —> les premières lignes permettent de réaliser l'affichage de la position de la tour au cours du temps déterminée avec le schéma d'Euler mais aussi avec **Scipy.integrate**.

**Q15.** Afin de réaliser un second affichage illustrant l'évolution de la position de l'amortisseur au cours du temps (avec Euler et **Scipy.integrate**), compléter les deux lignes correspondantes. Utiliser une autre couleur et une légende (label)



**Q16.** Faire une résolution sur 240 secondes avec un nombre de points de 200000, 20000 et 2000 et conclure.

## 4. INTERFACE GRAPHIQUE

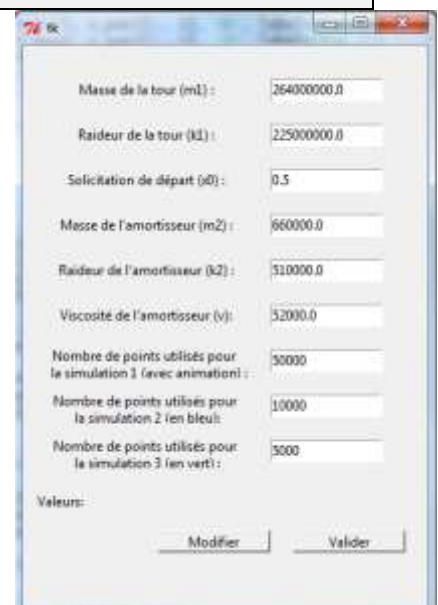
#### Ouvrir InterfaceTaipei.py

On crée un fichier séparé permettant de définir la fenêtre graphique (emplacement des boutons, champs) afin de ne pas surcharger l'écriture du code pour la résolution.

**Q17.** Expliquer les différentes parties de la classe **Interface** et expliquer à quoi sert la méthode **modifier**

Si l'on prend l'exemple de la variable  $m_1$  (ligne 31 à 34), pour mettre le texte demandant  $m_1$  on crée un label. Celui-ci va permettre d'afficher du texte (qui ne sera pas modifié) dans notre fenêtre. On crée alors le champ ("Entry") correspondant à  $m_1$  (ligne 35 à 40). On donne la variable associée ( $\text{textvariable}=\text{self.m}_1$ ), on le place puis on donne sa valeur initiale.

**Q18.** De la même façon, compléter le fichier **InterfaceTaipei.py** afin d'ajouter le champ **Raideur** associé à la variable  $k_1$  (ligne 42 à 47).



**Ouvrir TaipeiExercice3.py.**

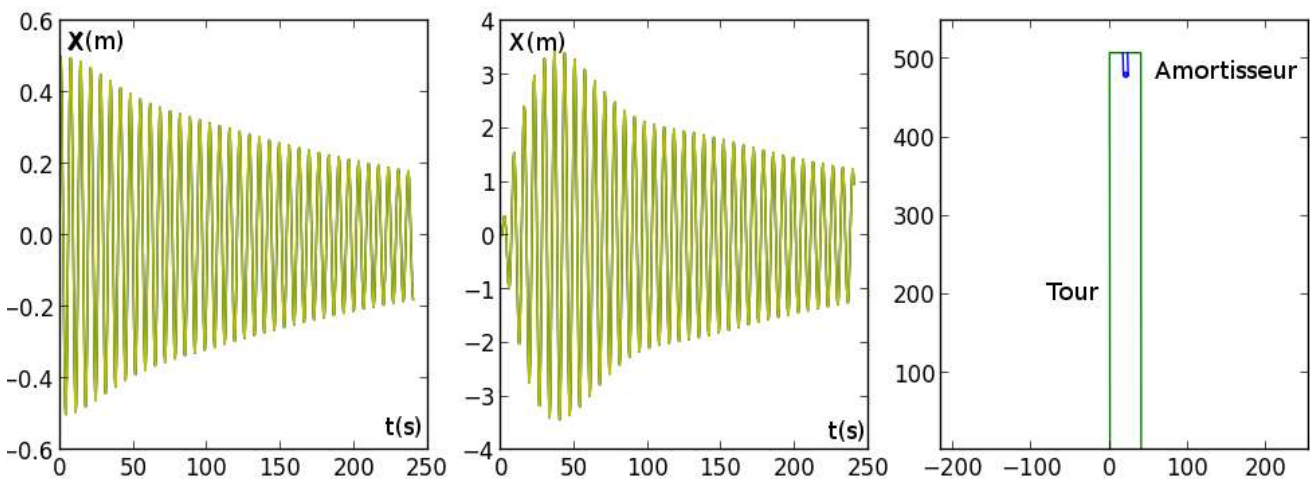
Q19. Copier coller le fichier InterfaceTaipei avant <— Mise en donnée —>  
 Q20. Commenter la partie <— Mise en donnée —>

La partie <— Écriture de la dérivée —> ne change pas ici car on résout le même problème.

Afin de pouvoir superposer le tracé pour trois pas de temps différents (comme on l'a fait manuellement à la fin de chacune des parties précédentes) trois résolutions sont prévues d'office dans le fichier TaipeiExercice3.py (partie <— Résolution —> lignes 62 à 89).

Q21. Soit  $nbpt1$ ,  $nbpt2$  et  $nbpt3$  les nombres de points qui vont être entrés dans l'interface graphique. Compléter les lignes indiquées (80 à 95) dans le fichier de la résolution avec le pas 1, pas2 et pas 3.

Q22. Lancer alors le programme Taipei\_exercice3.py, et remplir les champs de l'interface graphique.



Résultat final : Évolution de la position de la tour pour trois valeurs de pas dans le premier cadre, évolution de la position de l'amortisseur pour trois valeurs de pas dans le deuxième et animation correspondant à la première valeur de pas dans le troisième cadre

En utilisant les trois champs permettant de régler le nombre de points définir le nombre de points minimum pour garantir la stabilité du schéma.

Pour une même sollicitation comparer l'amplitude maximum de déplacement du bâtiment avec et sans amortisseur.

Tester maintenant le mouvement obtenu en donnant la sollicitation maximum prévue pour le bâtiment soit  $x_0 = 3m$  et conclure quant à l'amortissement mis en place ici

## 5. ANNEXE

### MASSE EQUIVALENTE :

La masse ponctuelle équivalente à l'ensemble de la masse du bâtiment doit être telle que l'énergie cinétique de la masse ponctuelle en mouvement soit équivalente à l'énergie cinétique de l'ensemble du bâtiment. On trouve par le calcul  $m_1 = \frac{3}{8} \cdot m_{tour} + m_{sph} = 263160000 = 263000t$ . On peut donc négliger, dans le modèle où l'amortisseur est fixe, la masse de cet amortisseur.

Classiquement l'amortisseur est compris entre 1.5% et 0.25% de la masse totale du bâtiment pour être efficace. Ici compte tenu de la difficulté de réalisation pratique de la sphère on peut se placer dans le cas d'une sphère de taille minimum soit 0.25% de la masse totale. On obtiendrait alors  $m_1 = 660000 \cdot \frac{100}{0.25} = 264000000K g$ . On retrouve bien le même ordre de grandeur, on prendra  $m_1 = 264000t$ .

RAIDEUR ÉQUIVALENTE DE LA TOUR :

La période d'oscillation du premier mode de la structure est de 6.8s [2]. On peut donc déterminer  $k_1 \approx 225000000.N.m^{-1}$  par la relation  $\omega = \sqrt{k_1/m_1}$ .

RAIDEUR DE L'AMORTISSEUR :

La pulsation de l'amortisseur doit être proche de celle du premier mode ( $\omega = \sqrt{k_1/m_1}$ ), ce qui connaissant  $m_2 = 660t$  permet de déterminer la raideur dont on a besoin pour la masse choisie.  $k_2 \approx 510000.N.m^{-1}$ .

VISCOSITÉ DE L'AMORTISSEUR :

D'après [2] l'amortissement de  $\xi = 4,7\%$  est tel que  $f = 2.\xi.\sqrt{k_2.m_2} \approx 52000.Kg.s^{-1}$ .

DÉMONSTRATION MASSE ÉQUIVALENTE :

*Cette démonstration est tirée de [1]. La justification est hors programme pour certaines filières. Elle a uniquement pour but de justifier une formule utilisée et est uniquement à destination des collègues.*

La tour de  $L = 500m$  de hauteur de module d'Young  $E$  et de masse volumique  $\rho$  supporte l'amortisseur considéré comme une masse ponctuelle au sommet ( $M = 660t$ ). L'ensemble est soumis à un effort au sommet normal à l'axe de la tour. Les déplacements possibles de la tour sont de faible amplitude (3m devant 500m) et assimilables à une translation horizontale.

Dans cette tour, le moment fléchissant est :  $M_f(x) = F(L-x)$ . En utilisant la relation de comportement, la courbure est :

$$\frac{\partial^2 v}{\partial x^2} = \frac{F}{EI}(L-x)$$

Donc le déplacement perpendiculaire à la ligne moyenne (avec la condition de rotation et de déplacement nulle (encastrement) en  $x = 0$ ) est :

$$v(x) = \frac{F}{EI} \left( L \cdot \frac{x^2}{2} - \frac{x^3}{6} \right)$$

Soit  $v(L) = \frac{FL^3}{3EI}$ .

La vitesse latérale en tout point de la tour est donnée en fonction de la vitesse en bout ( $x=L$ ) par :

$$\dot{v}(x) = \dot{v}(L) \cdot \left( \frac{3x^2}{2L^2} - \frac{x^3}{2L^3} \right) = \dot{v}(L) \cdot \phi(x)$$

L'équivalence en énergie énergie cinétique donne :

$$\frac{1}{2} m_{eq} \dot{v}(L)^2 = \frac{1}{2} \int_0^L \dot{v}(x)^2 \rho S dx + \frac{1}{2} M \dot{v}(L)^2 = \frac{1}{2} \int_0^L (\dot{v}(L) \cdot \phi(x))^2 \rho S dx + \frac{1}{2} M \dot{v}(L)^2$$

qui après calcul donne :

$$M_{eq} = \frac{3}{8} \cdot m_{tour} + M = 263160000$$

## 6. REFERENCES

- [1] L. Champaney, Exemples de réduction d'un système complexe à un système à un degré de liberté, Notes de cours de Dynamique des Constructions.
- [2] Loannis Kourakis, Structural Systems and Tuned Mass Dampers of Super-Tall Buildings : Case Study of Taipei 101, Massachusetts Institute of Technology juin 2007.