Travaux Pratiques / Harpe sympatique

L. Le Marrec*

October 13, 2025

1 Objectif général

Dans le code de référence que vous pouvez trouvez en section §3.1, un premier code vous a été proposé. L'objectif pour vous est de proposer à partir de cette référence, une nouvelle version du code qui

- est mieux structurée : c'est à dire avec des fonctions.
- plus efficace : réduction du temps de calcul, réduction des entrées/sorties.
- plus interactif : ou plusieurs options sont proposées en fonction de paramètres identifiés par l'utilisateur.
- propose des extensions ou des applications originales.

1.1 Point de départ

Pour réaliser cet objectif, on va partir d'une version déjà structurée du code... mais vide. Elle est proposée en §3.2. On voit que ce programme principal fait appel à plusieurs fonctions

- `ParamInit.m` Comme son l'indique cette fonction a pour vocation de fournir les variables d'entrée de la modélisation. On y retrouve les caractéristiques fondamentale de la structure, des conditions initiales, et de la modélisation.
- `ParamInter.m` Pour estimer les paramètres intermédiares, qui sont souvent celles exploitées dans les calcul.
- `DomaineModal.m` On introduit les grandeurs modales
- `DomaineSpatial.m` On introduit les grandeurs permettant d'évaluer le domaine spatial.
- `DomaineTemporel.m` On introduit les grandeurs permettant d'évaluer le domaine temporel.
- `ModePropre.m` Pour calculer la fonction $Y_n(S)$. La variable `Aff` peut servir à définir si l'on souhaite avoir un affichage associé ou nom.
- `AmplitudeModale.m` L'évaluation des amplitudes modales.
- `FctTemporelle.m` Pour définir les fonctions $T_n(t)$ (ou évntuellement $q_n(t)$.
- `FctDeplacement.m` Pour déterminer $u(S,t)$
- `Illustration.m` Cette fonction permet de proposer plusieurs illustrations finales. En fonction de la variable `Type` on peut ainsi choisir le type d'illustration qui est présenté.

La première étape est donc tout simplement de *remplir ces fonctions*. Dans bien des cas, il suffit en réalité de faire des copié collé judicieux à partir du code de référence (§3.1). A chaque étape il est bien sûr nécessaire de s'assurer que le code compile sans erreurs.

*IRMAR, 02 23 23 66 86, loic.lemarrec@univ-rennes.fr

1.2 Améliorations modestes

Il y'a donc plusieurs options qui sont proposées et qui sont contrôlées par des variables `Aff`, ou `Type`. A vous de choisir pertinament les valeurs de ces variables. Par exemple et de manière progressive :

1. dans `ParamInit` vous pouvez définir `Aff=1` et le mettre en variable de sortie de cette fonction.
2. dans les fonctions qui utilisent `Aff`, comme `AmplitudeModale.m`, vous pouvez choisir de faire un affichage si `Aff>0` ou non si `Aff=0`. Pour cela vous pouvez utiliser des commande `if` ou `switch case` (regardez dans l'aide comment cela fonctionne)
3. Une fois que cela fonctionne, vous pouvez définir plusieurs valeurs de `Aff` dans `ParamInit`. Par exemple en définissant un vecteur : `Aff=[0,2,4,1]`. Ainsi dans chaque fontion qui utilise cette variable, vous pouvez spécifier une ligne spécifique. Bref un truc du genre :

```
Y=ModePropre(kn,s,Nw,Aff(1));
[an,bn]=AmplitudeModale(L,el,kn,wn,n,H,Aff(2));
T=FctTemporelle(Nw,wn,an,bn,t,Aff(3));
```

Un autre point qui rend le code môche est qu'il y a des variables qui pourraient très bien être évaluées en interne. Par exemple celle de type `tmax`, `dt`, ou encore `Nt` (idem pour les modes ou l'espace). En fait une fois que l'on a évalué `t` dans `DomaineTemporel.m` alors finalement cette variable suffit en effet pas la peine de mettre les détails en entrée des autres fonctions car `dt=t(2)-t(1)`, `tmax=max(t)` et `Nt=length(t)`. Donc il suffit d'utiliser ce genre de petites lignes de codes dans les fonctions qui utilisent le temps. Cela permet de réduire le transfert d'information : cela permet d'augmenter la lisibilité du code. On obtient d'un côté `[t]=DomaineTemporel(Per,L)` et de l'autre (pour l'espace) `Y=ModePropre(kn,s,Aff)`.

Il y'a des variables qui ne sont pas utilisées `Def`, `Freq`, ou très localement comme `Note`. Surtout au début. Il serait bien de réorganiser cela pour avoir qu'une seule fonction dans le `MAIN.m` par exemple `[L,C,H,el,Nw,Aff]=Param` qui utiliserait *à l'intérieur* `[L,R,E,ro,Note,H,el,Nw,Aff]=ParamInit`. On pourrait même envisager une variable d'option pour `ParamInit` : en fonction de cette variable, on pourrait choisir plusieurs matériaux par exemple. Au final, on aurait un code de la sorte :
au début du `MAIN.m` on trouverait `[L,C,H,el,Nw,Aff]=Param(TypeCorde)` et dans cette fonction on utiliserait `[L,R,E,ro,Note,H,el,Nw,Aff]=ParamInit(TypeCorde)` qui sortirait des propriétés différentes en fonction de la variable `TypeCorde` en utilisant judicieusement `swith case` ou `if`.

2 Pour aller plus loin

2.1 Quelques mises en applications

On peut exploiter les idées proposées en fin de code. Il est facile de trouver l'aide pour réaliser un film, une video, ou d'enregistrer et d'écouter la dynamique présente en un point S de la corde. Pour cela, l'aide fournie par Matlab ou sur internet est suffisament riche.

2.2 Tenir compte d'autres conditions aux limites

Pour des conditons aux limites de type Dirichlet-Neuman $u(0,t) = 0$ & $u'(L,t) = 0$. Le polycopié donne également des solutions très faciles. Vous pouvez les intégrer ou mieux les mettre en choix au début de votre code.

2.3 Conditons aux limites avec un ressort ou une masse

Nous avons vu cela en TD et un code a été fourni. Il est donc possible maintenant d'intégrer cela comme vous l'avez fait pour ce problème plus facile. Cela nécessite néanmoins de créer de nouvelles fonciton et de bien se familiariser avec la méthode de travail et de programmation.

2.4 Changer les conditions initiales

Nous avons vu d'autres conditions initiales et les solutions a_n & b_n associées. Cela peut donc aussi assez facilement être intégré à votre code.

2.5 Valorisation du travail

Vous disposez du code source à la fois matlab mais également du fichier LaTeX de ce document. Vous pouvez donc compléter et améliorer ce document pour faire un compte rendu de votre TP. Vous pouvez également choisir de réaliser un Beamer pour valoriser ce travail. La encore les sources et informations sont très nombreuses sur le web.

3 Codes

3.1 Code de référence

```
%% ================================================================
%% INITIALISATION ================================================
clear;close all;clc;
% Geometrie : section cicrculaire
L=1;                % Longueur [m]
R=0.001;            % Rayon [m]
A=pi*R^2;           % Aire [m^2]
% Materiau : acier
E=210e9;            % Module de Young [Pa]
ro=7800;            % Masse volumique [kg/m^3]
% Tension : corde accordee sur le la-440
Note=440;           % Frequence fondamentale [Hz]
C=2*L*Note;         % Celerite [m/s]
N0=ro*A*C^2;        % Tension [N]
Def=N0/(E*A);       % Deformation [~]
% Excitation: corde pincee d'une hauteur H en s=el
H=L/5;              % Hauteur [m]
el=L/4;             % poistion [m]
% Domaine modal
nmax=10;            % Nombre maximal de mode considere
n=(1:nmax)';        % Indices modaux
Nw=nmax;
kn=n*pi/L;          % Nombres d'ondes [1/m] : corde fixee aux deux extremites
wn=C*kn;            % Pulsation [rad/s], relation de dispersion
Lamb=2*pi./kn;      % Longueur d'onde de chaque mode [m]
Per=2*pi./wn;       % Periode de chaque mode [s]
Freq=1./Per;        % Frequence de chaque mode [Hz]
% Domaine spatial
ds=min(Lamb)/20;% Pas en espace [m]
s=(0:ds:L);         % Echantillonage spatial [m]
Ns=length(s);       % Nombre de points d'echantillonages en espace
% Domaine temporel
dt=min(Per)/20; % Pas en temps [s]
tmax=max(Per)*2;% Temps maximum de la simulation [s]
t=0:dt:tmax;        % Echantillonage temporel [s]
Nt=length(t);       % Nombre de points d'echantillonages en temps
%% %% ============================================================
%%Rq% Dans une phase de bebeugage, il faut que [Nt,Ns,Nw] aient des valeurs
% raisonnables (<=1000) et si possible distinctes.
disp(['[Nt,Ns,Nw]=[' num2str([Nt,Ns,Nw]) ']'])

%% ================================================================
%% ANALYSE MODALE ================================================
% Modes propres
for in=1:Nw
    % Y_ij, avec i=>n et j=>s
    Y(in,:)=sin(kn(in)*s);
end
%=> visualisation de quelques modes propres
figure(1);
plot(s,Y([1:3 nmax],:),'LineWidth',2)
xlabel('s [m]')
legend('n=1','n=2','n=3','n=nmax')
set(gca,'FontSize',24)

% Amplitude modale
an=2*H./(n*pi)*L/(L-el).*sin(kn*el)./(kn*el);
bn=zeros(size(n));
%=> visualisation des amplitudes modales an
figure(2);
```

```
stem(wn,abs(an),'LineWidth',2)
xlabel('wn [rad/s]')
ylabel('|an| [m]')
set(gca,'FontSize',24)

% Fonction en temps
for in=1:nmax
    % T_ij, avec i=>n et j=>t
    T(in,:)=an(in)*cos(wn(in)*t)+bn(in)*sin(wn(in)*t)/wn(in);
end
%=> visualisation de T(t) pour quelques modes
figure(3);
plot(t,T([1:3 nmax],:),'LineWidth',2)
xlabel('t [s]')
legend('n=1','n=2','n=3','n=nmax')
set(gca,'FontSize',24)
%% ================================================================

%% ================================================================
%% SOLUTION GENERALE ==============================================
u=Y'*T;     % u_ij, avec i=>s et j=>t
%-> visualisation de u(s,t) a divers instants
figure(4);subplot(1,2,1)
plot(s,u(:,[1 10 20]),'LineWidth',2);
xlabel('s [m]');ylabel('u(s,t) [m]');
legend(['t=' num2str(t(1)) ],['t=' num2str(t(10)) ],['t=' num2str(t(20)) ])
axis equal
set(gca,'FontSize',24)
%-> visualisation de u(s,t) en divers point de la corde
figure(4);subplot(1,2,2)
plot(t,u([1 10 20],:),'LineWidth',2);
xlabel('t [m]');ylabel('u(s,t) [m]');
legend(['s=' num2str(s(1)) ],['s=' num2str(s(10)) ],['s=' num2str(s(20)) ])
set(gca,'FontSize',24)
%-> visualisation de u(s,t) au cours du temps
figure(5);
for j=1:Nt
    plot(s,u(:,j),'k','LineWidth',2);hold on
    plot(s([1 10 20]),u([1 10 20],j),'o','MarkerSize',8,'LineWidth',2)
    hold off
    xlabel('s [m]');ylabel('u(s,t) [m]');
    axis equal;axis([0,L,-H,H])
    set(gca,'FontSize',24);
    pause(0.1)
end
%% ================================================================
```

3.2 Code MAIN structuré

```
%% ================================================================
%% INITIALISATION =================================================
clear;close all;clc;
%% ================================================================
% Chargement des parametres
[L,R,E,ro,Note,H,el,Nw,Aff]=ParamInit;
% Parametres intermediaires
[A,C,N0,Def]=ParamInter(R,L,ro,E,Note);
% Domaine modal
[n,kn,wn,Lamb,Per,Freq]=DomaineModal(Nw,L,C);
% Domaine spatial
[ds,s,Ns]=DomaineSpatial(Lamb,L);
% Domaine temporel
[dt,t,Nt,tmax]=DomaineTemporel(Per,L);
% Rq : dans une phase de bebeugage, il faut que [Nt,Ns,Nw] aient des valeurs
% raisonnables (<=1000) et si possible distinctes.
disp(['[Nt,Ns,Nw]=[' num2str([Nt,Ns,Nw]) ']'])

%% ================================================================
%% ANALYSE MODALE =================================================
% Modes propres
Y=ModePropre(kn,s,Nw,Aff);
% Amplitude modale
```

```
[an,bn]=AmplitudeModale(L,el,kn,wn,n,H,Aff);
% Fonction en temps
T=FctTemporelle(Nw,wn,an,bn,t,Aff);
% Deplacement
u=FctDeplacement(Y,T);

%% ========================================================================
%% VALORISATION ============================================================
Type=1;Illustration(Type,u,s,t)
Type=2;Illustration(Type,u,s,t)
Type=3;Illustration(Type,u,s,t)
% D'autres valorisations peuvent etre envisagees, quelques propostion
% Film ?
% Son ?
% Autre ?
```