Laboratorio de Dinámica Topológica Aplicada:

IDENTIFICACIÓN DINÁMICA LINEAL DE SISTEMAS MECÁNICOS

Fredy Vides
Scientific Computing Innovation Center, UNAH &
Centre for Analysis of Data-Driven Systems
E-mail: fredyvides@caddslab.com

ÍNDICE

OBJETIVOS

1. Clasificar modelos dinámicos estructurales planares en términos de sus caracteríticas mecánicas elementales.
2. Identificar el modelo dinámico lineal planar que mejor describe la deformación dinámica de un conjunto de nodos en una estructura dada, con a lo más dos grados de libertad por nodo.
3. Calcular numéricamente de forma eficiente la deformación dinámica aproximada de un conjunto de nodos en una estructura dada, con a lo más dos grados de libertad por nodo.

1. DINÁMICA ESTRUCTURAL EN CONJUNTOS DE NODOS CON DOS GRADOS DE LIBERTAD

1.1. Identificación de sistemas dinámicos estructurales. Considerando una estructura como la mostrada en la figura 1.1. El problema de identifiación dinámica estructural lineal **IDEL** es un problema de ingeniería inversa, que consiste en calcular el modelo matricial dinámico óptimo en el sentido de los mínimos cuadrados, que describe el comportamiento

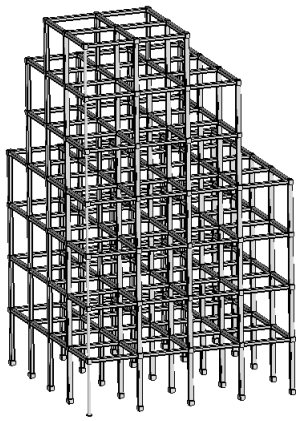

FIGURA 1.1. Estructura de acero de 7 niveles sujeta a cargas externas.

dinámico de la estructura en la figura 1.1, con base en secuencias de datos correspondientes al comportamiento dinámico observado de la estructura, medido a través de sensores localizados en puntos de interés específicos en la estructura.

(1.1)


Los modelos dinámicos que estudiaremos en este laboratorio son de la forma.

$$\text{(1.2)} \quad \begin{cases} x_{t+1} = Ax_t + c \\ y_t = Sx_t x_t = \arg\min \|y - X_t\|_2 \end{cases} , \ t \in \mathbb{Z}^+, X_t \in X([0,T]), T > 0$$

donde $\{X_t\}_{t\geq 1}^m$ es una muestra de tamaño m en una serie de tiempo $X([0,T]), T > 0$, de datos generados por una colección de sensores localizados en puntos de interés en la estructura de la figura 1.1.
Los modelos de matriciales de la forma (1.2) serán calculados aplicando los algoritmos y técnicas desarrollados por F. Vides en [3].

2. PROYECTO DE LABORATORIO 1

2.1. Modelos Dinámicos Planares Lineales Basados en Señales Mecánicas Lineales.

2.1.1. Modelos mecánicos dinámicos lineales. Considerando el sistema mecánico determinado por la deflexión de un nodo de interés en la estructura de la figura 1.1, cuya dinámica es aproximadamente descrita por la ecuación diferencial:

$$\text{(2.1)} \quad m\frac{d^2x}{dt^2} + \delta\frac{dx}{dt} + \omega^2 x = 0$$

donde se asume que $m, \omega, \delta \in \mathbb{R}$, $m > 0$ y $4m\omega^2 - \delta^2 > 0$

Es posible aplicar reducción de orden para verificar que el sistema (2.1) es equivalente al sistema de ecuaciones diferenciales de primer orden.

$$\frac{d}{dt}\begin{bmatrix}x\\y\end{bmatrix}=\begin{bmatrix}0 & 1\\ -\frac{\omega^2}{m} & -\frac{\delta}{m}\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} \tag{2.2}$$

El sistema (2.2) puede estudiarse utilizando el siguiente sistema genérico.

$$\frac{d}{dt}\begin{bmatrix}x\\y\end{bmatrix}=\begin{bmatrix}0 & 1\\ -\alpha & -\beta\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix} \tag{2.3}$$

para dos parámetros adimensionales α, β.
Es posible aplicar GNU Octave para resolver (2.3) para las condiciones iniciales $x(0) = 1$, $y(0) = x'(0) = 0$. Aplicando un método numérico de cuarto orden para calcular la curva solución aproximada $\hat{X}([0, 10]) \approx X([0, 10])$ correspondiente a la solución $X(t)$ de (2.1), con un orden de error relativo $\epsilon_r = \mathcal{O}(10^{-16})$, para las condiciones iniciales $x(0) = 1, y(0) = 0$, y para los valores paramétricos $m = 10, \omega = \sqrt{50}, \delta = 0$.
A continuación se presenta el código del programa `MechanicalSystem.m` en GNU Octave, que puede aplicarse para calcular $\hat{X}([0, 10])$ y generar un subconjunto del campo vectorial que controla la dinánica del sistema mecánico (2.1) con base en el modelo dinámico genérico (2.3).

```
% Copyright (C) 2020 Fredy Vides
% function [t,x,A,C,Th]=MechanicalSystem(m,k,d,T,x0)
% A mechanical system simulation

% Example: [t,x,A,C,Th]=MechanicalSystem(10,50,0,10,[1,0]);

% Author: Fredy Vides <fredy@HPCLAB>
% Scientific Computing Innovation Center
% Created: 2020-05-03
function [t,x,A,C,Th]=MechanicalSystem(m,k,d,T,x0)
  Fx=@(x,y)y;
  Fy=@(x,y)(1/m)*(-k*x+d*y);
  f=@(t,y)[y(2);(1/m)*(-k*y(1)+d*y(2))];
  opt=odeset ("RelTol", eps);
  [t,x] = ode45 (f, [0, T], x0,opt);
  cx=norm(max(abs(x)));
  [X,Y]=meshgrid(-cx:2*cx/30:cx);
  quiver(X,Y,Fx(X,Y),Fy(X,Y),'k');
  hold on;
  plot(x(:,1),x(:,2),'k','markersize',12);
  hold off;
  axis equal;
  N=size(x,1);
  h=T/N;
  C=[0 1;-k/m d/m];
  p=(h.^(4:-1:0))./(factorial(4:-1:0));
  A=polyvalm(p,C);
  Th=expm(h*C);
```

```
end
```

Es posible aplicar el programa `MechanicalSystem.m` para simular una serie de tiempo de deflexiones aproximadas $\hat{X}([0,10]) = \{x_1, \ldots, x_T\}$, para el elemento estructural representado por un nodo genérico de dos grados de libertad.
Una vez que se ha escrito el programa y se ha guardado en un directorio de ruta simple (sin dejar espacios entre los nombres de los sub-directorios), es posible utilizar el icono  para asegurarse de que el Directorio Actual/Current Directory de GNU Octave sea el directorio donde se guardó el archivo de programa `MechanicalSystem.m`.
Se puede ejecutar el programa `MechanicalSystem.m` en la ventana de comandos de GNU Octave con la siguiente secuencia de comandos:

```
>> [t,x,A,C,Th]=MechanicalSystem(10,50,0,10,[1,0]);
```

2.1.2. Cómputo de controladores universales para sistemas dinámicos estructurales. Ahora es posible desarrollar un método numérico que permita calcular una representación matricial de un controlador algebraico universal (**UAC**) con tres grados de libertad iniciales, basado en la muestra $\hat{X}([0,10])$ de la serie de tiempo $\hat{X}([0,T]), T > 0$, generada por el programa `MechanicalSystem.m`, en el sentido de [2], aplicando las técnicas presentadas en [2,3].
Es posible escribir un programa GNU Octave que permite calcular la representación matricial del UAC (estabilizado aproximado) del sistema. La representación matricial del UAC producirá un par (T_h, f) donde $T_h \in \mathbb{R}^{2\times 2}$ y $f \in \mathbb{R}^n$ cumplen (además de otras condiciones genéricas estudiadas en [2,3]) las restricciones:

$$\text{(2.4)} \quad \begin{cases} x_{t+1} = T_h x_t + f \\ x_t = \arg\min \|y - X_t\|_2 \end{cases}, \; t \in \mathbb{Z}^+, X_t \in X([0,T]), T > 0$$

Los grados de libertad en este contexto se refieren a que en la matriz inicial $A \in \mathbb{R}^{2\times 2}$ generada por el algoritmo, aunque las cuatro entradas están restringidas por la información dinámica estructural del sistema, solo tres entradas son libres para tomar valores factibles no preestablecidos, la matriz estructurada A es luego **estabilizada** para generar la matriz T_h, de tal manera que el sistema (2.4) sea **localmente estable** en el sentido de sistemas dinámicos lineales de tiempo discreto, es decir, la deflexión de los elementos estructurales se mantiene dentro del margen de tolerancia estableciado para la estructura o sistema estructural en estudio, durante un periodo de tiempo determinado. El código del programa `TMatrixID.m` de GNU Octave que calcula el par (T_h, f) que cumple las restricciones (2.4), se muestra a continuación:

```
% Copyright (C) 2020 Fredy Vides
% function [A,Ap,f]=TMatrixID(x,m)
% A universal algebraic controller
% comptutation
% This method is based on the UAC method
% presented by F. Vides in the article:
% "Universal algebraic controllers and
% system identification"

% See also: MechanicalSystemID.
% Author: Fredy Vides <fredy@HPCLAB>
% Scientific Computing Innovation Center
```

```
% Created: 2020-05-03
function [A,Ap,f]=TMatrixID(x,m)
  N=size(x,1);
  m=min([m N-1]);
  x0=x(1:m,1);
  x1=x(2:(m+1),1);
  y0=x(1:m,2);
  y1=x(2:(m+1),2);
  E=ones(m,1);
  c1=[y0 E]\(x1-x0);
  c2=[x0 y0 E]\y1;
  A=[1 c1(1);c2(1:2).'];
  f=[c1(2);c2(3)];
  [v,a]=eig(A);
  a=diag(a);
  a=diag(a./abs(a));
  Ap=real(v*a/v);
end
```

Aplicando el programa `TMatrixID.m`, calcular el par UAC (T_h, f) para el sistema mecánico, basado en la muestra $\hat{X}([0, 10])$, para un tamaño de submuestreo $m = 120$, ejecutando el programa `TMatrixID.m` en la ventana de comandos de GNU Octave utilizando la siguiente secuencia de comandos.

```
>> [A,Th,f]=TMatrixID(x,120)
```

2.1.3. Identificación lineal de dinámica estructural basada en señales mecánicas lineales. Ahora se puede desarrollar un algoritmo de identificación que permite calcular el modelo lineal UAC de tres grados de libertad que mejor aproxima la dinámica del sistema mecánico en el sentido de mínimos cuadrados (con respecto a la métrica Euclidiana d_2 en $\mathbb{R}^2$), con base en la muestra $\hat{X}([0, 10])$ de la serie de tiempo $\hat{X}([0, T]), T > 0$. Es posible escribir un programa en GNU Octave que permite implementar un algoritmo identificación para sistemas basados en señales mecánicas, con base en modelos genéricos de la forma (2.3). El código GNU Octave del programa `MechanicalSystemID.m` para la identificación de sistemas mecánicos estructurales lineales se muestra a continuación.

```
% Copyright (C) 2020 Fredy Vides
% function [Th,Thp,C]=MechanicalSystemID(m)
% A sytem identificacion computational
% implementation

% Example: [Th,Thp,C]=MechanicalSystemID(120)

% Author: Fredy Vides <fredy@HPCLAB>
% Scientific Computing Innovation Center
% Created: 2020-05-03
function [Th,Thp,C]=MechanicalSystemID(m)
[t,x,A,C,Th]=MechanicalSystem(10,50,0,10,[1,0]);
hold on;
[Ap,Thp,f]=TMatrixID(x,m);
```

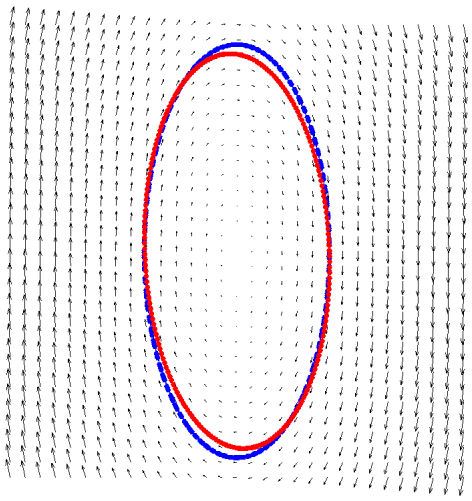

FIGURA 2.1. Salida gráfica producida por el algoritmo de identificación para $m = 120$.

```
N=size(x,1);
Yp=[1;0];
Y=Yp;
for k=1:3*N
    Y=[Y Th*Y(:,k)];
    Yp=[Yp Thp*Yp(:,k)+f];
end
plot(x(:,1),x(:,2),'k');
plot(Y(1,:),Y(2,:),'b.-','markersize',12);
plot(Yp(1,:),Yp(2,:),'r.-','markersize',12);
axis equal;
hold off;
end
```

Se puede aplicar el programa `MechanicalSystemID.m` para calcular y visualizar diversas identificaciones aproximadas de sistemas mecánicos estructurales de la forma (2.3), correspondientes a diferentes valores de submuestreo m. En particular para $m = 120$, se pueden utilizar las siguientes secuencias de comandos para calcular la identificación del sistema correspondiente.

```
>> [Th,Thp,C,T,Tp]=MechanicalSystemID(120);
```

La salida gráfica producida por el programa se muestra en la figura 2.1.

3. PROYECTO DE LABORATORIO 2

3.1. Modelos Dinámicos Planares Lineales Basados en Señales Mecánicas No Lineales.

3.1.1. Modelos mecánicos dinámicos no lineales. Considerando el sistema mecánico determinado por la deflexión de un nodo de interés en la estructura de la figura 1.1, cuya dinámica

es aproximadamente descrita por el sistema genérico (3.1),

$$\frac{d}{dt}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} y \\ -\alpha x & -\beta x^3 \end{bmatrix} \tag{3.1}$$

para dos parámetros adimensionales α, β.
Es posible aplicar GNU Octave para resolver (2.3) para las condiciones iniciales $x(0) = 1$, $y(0) = x'(0) = 0$. Aplicando un método numérico de cuarto orden para calcular la curva solución aproximada $\hat{X}([0, 10]) \approx X([0, 10])$ correspondiente a la solución $X(t)$ de (2.1), con un orden de error relativo $\epsilon_r = \mathcal{O}(10^{-16})$, para las condiciones iniciales $x(0) = 1, y(0) = 0$, y para los valores paramétricos $m = 10, \omega = \sqrt{50}, \delta = 0$.
A continuación se presenta el código del programa `NLMechanicalSystem.m` en GNU Octave, que puede aplicarse para calcular $\hat{X}([0, 10])$ y generar un subconjunto del campo vectorial que controla la dinánica del sistema mecánico (3.1).

```
% Copyright (C) 2020 Fredy Vides
% function [t,x]=NLMechanicalSystem(m,k,d,T,x0)
% A mechanical system simulation

% Example: [t,x]=NLMechanicalSystem(10,50,0,10,[1,0]);

% Author: Fredy Vides <fredy@HPCLAB>
% Scientific Computing Innovation Center
% Created: 2020-05-03
function [t,x]=NLMechanicalSystem(m,k,d,T,x0)
  Fx=@(x,y)y;
  Fy=@(x,y)(1/m)*(-k*x+d*x.^3);
  f=@(t,y)[y(2);(1/m)*(-k*y(1)+d*y(1).^3)];
  opt=odeset ("RelTol", eps);
  [t,x] = ode45 (f, [0, T], x0,opt);
  cx=norm(max(abs(x)));
  [X,Y]=meshgrid(-cx:2*cx/30:cx);
  quiver(X,Y,Fx(X,Y),Fy(X,Y),'k');
  hold on;
  plot(x(:,1),x(:,2),'k','markersize',12);
  hold off;
  axis equal;
end
```

Es posible aplicar el programa `NLMechanicalSystem.m` para simular una serie de tiempo de deflexiones aproximadas $\hat{X}([0, 10]) = \{x_1, \ldots, x_T\}$, para el elemento estructural representado por un nodo genérico de dos grados de libertad.
Una vez que se ha escrito el programa y se ha guardado en un directorio de ruta simple (sin dejar espacios entre los nombres de los sub-directorios), es posible utilizar el icono  para asegurarse de que el Directorio Actual/Current Directory de GNU Octave sea el directorio donde se guardó el archivo de programa `MechanicalSystem.m`.
Se puede ejecutar el programa `NLMechanicalSystem.m` en la ventana de comandos de GNU Octave con la siguiente secuencia de comandos:

```
>> [t,x]=NLMechanicalSystem(10,50,0,10,[1,0]);
```

3.1.2. Identificación lineal de dinámica estructural basada en señales mecánicas no lineales. Aplicando el programa Octave `TMatrixID.m` desarrollado en §2.1.2, se puede desarrollar un algoritmo de identificación que permite calcular el modelo lineal UAC de tres grados de libertad que mejor aproxima la dinámica del sistema mecánico en el sentido óptimo de mínimos cuadrados, con base en la muestra $\hat{X}([0, 10])$ de la serie de tiempo $\hat{X}([0, T]), T > 0$. Es posible escribir un programa en GNU Octave que permite implementar un algoritmo identificación para sistemas basados en señales mecánicas, con base en modelos genéricos de la forma (3.1). El código GNU Octave del programa `NLMechanicalSystemID.m` para la identificación de sistemas mecánicos estructurales lineales se muestra a continuación.

```
% Copyright (C) 2020 Fredy Vides
% function [Thp,f,t,x,Y]=NLMechanicalSystemID(m)
% A sytem identificacion computational
% implementation

% Example: [Thp,f,t,x,Y]=NLMechanicalSystemID(180)

% Author: Fredy Vides <fredy@HPCLAB>
% Scientific Computing Innovation Center
% Created: 2020-05-03
function [Thp,f,t,x,Yp]=NLMechanicalSystemID(m)
subplot(211);
[t,x]=NLMechanicalSystem(2,.1,-4,30,[1,0]);
hold on;
[Ap,Thp,f]=TMatrixID(x,m);
N=size(x,1);
Yp=[1;0];
for k=1:(N-1)
    Yp=[Yp Thp*Yp(:,k)+f];
end
plot(x(:,1),x(:,2),'b.-','markersize',12);
plot(Yp(1,:),Yp(2,:),'r.-','markersize',12);
axis square;
grid on;
hold off;
subplot(212);
plot(t,x(:,1),'k',t,x(:,2),'r');
hold on;
plot(t,Yp(1,:),'k.-','markersize',12);
plot(t,Yp(2,:),'r.-','markersize',12);
legend('x','dx/dt')
axis equal;
grid on;
end
```

Se puede aplicar el programa `NLMechanicalSystemID.m` para calcular y visualizar diversas identificaciones aproximadas de sistemas mecánicos estructurales de la forma (3.1), correspondientes a diferentes valores de submuestreo m. En particular para $m = 140$, se

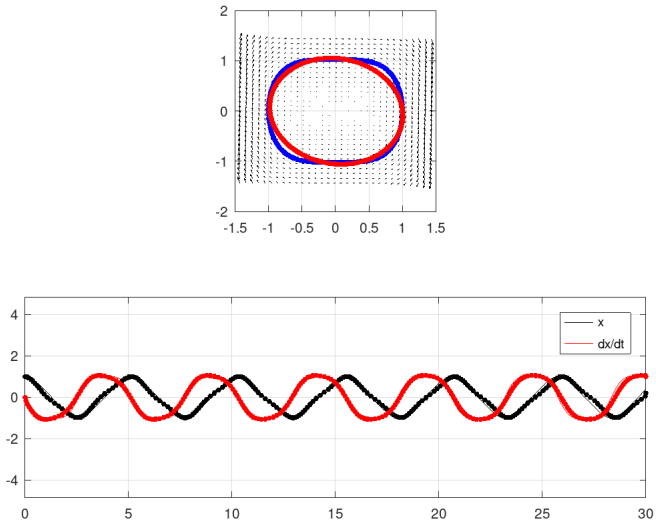

FIGURA 3.1. Salida gráfica producida por el algoritmo de identificación para un tamaño de submuestreo de la serie de tiempo del sistema de $m = 140$. La sub-figura superior muestra el diagrama de fase aproximado del sistema mecánico. La sub-figura inferior muestra las gráficas de posición y velocidad correspondientes.

pueden utilizar las siguientes secuencias de comandos para calcular la identificación del sistema correspondiente.

```
>> [Thp,Tp,t,x,Y]=NLMechanicalSystemID(140);
```

La salida gráfica producida por el programa se muestra en la figura 3.1.

REFERENCIAS

[1] Edwards, C. H., Penney, D. E (2000). Ecuaciones Diferenciales. 4a Ed. Prentice Hall.

[2] Vides, F (2019). On uniform connectivity of algebraic matrix sets. Banach J. Math. Anal., 13(4):918-943, 2019.

[3] Vides, F (2020). Universal algebraic controllers and system identification. In press. 2020. Also available at https://arxiv.org/pdf/2001.11133.pdf