

Péndulos Acoplados Amortiguados y Forzados

Generated by Doxygen 1.9.8

1 README	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 Params Struct Reference	7
4.1.1 Detailed Description	7
4.1.2 Member Data Documentation	8
4.1.2.1 F0	8
4.1.2.2 g	8
4.1.2.3 gamma	8
4.1.2.4 kappa	8
4.1.2.5 l	8
4.1.2.6 m	8
4.1.2.7 omega	8
4.2 Pendulo Class Reference	8
4.2.1 Detailed Description	9
4.2.2 Constructor & Destructor Documentation	10
4.2.2.1 Pendulo()	10
4.2.3 Member Function Documentation	10
4.2.3.1 get_omega()	10
4.2.3.2 get_theta()	11
4.2.3.3 individual_energy()	11
4.2.3.4 set_omega()	11
4.2.3.5 set_theta()	11
4.2.4 Member Data Documentation	11
4.2.4.1 omega_	11
4.2.4.2 theta_	11
5 File Documentation	13
5.1 include/Pendulo.h File Reference	13
5.1.1 Function Documentation	13
5.1.1.1 calculate_accel()	13
5.2 Pendulo.h	14
5.3 README.md File Reference	14
5.4 src/main.cpp File Reference	14
5.4.1 Function Documentation	14
5.4.1.1 get_coupling_energy()	14
5.4.1.2 get_user_input()	15
5.4.1.3 main()	15

5.5 src/Pendolo.cpp File Reference	15
Index	17

Chapter 1

README

Péndulos Acoplados Amortiguados y Forzados

****Curso:** Física Computacional II**

Descripción del Proyecto

Este proyecto implementa la simulación numérica de ****dos péndulos acoplados con amortiguamiento, forzamiento externo y una interacción no lineal cuadrática****. El sistema se resuelve usando ****Runge-Kutta de 4º orden (RK4)**** o ****Euler-Cromer****, permitiendo al usuario configurar todos los parámetros de forma interactiva.

El software, desarrollado en C++17, permite explorar fenómenos como la ****transferencia de energía****, ****sincronización****, ****resonancia**** y ****caos****.

Modelo Físico

El sistema está gobernado por las ecuaciones:

$$\backslash$$

$$\backslash \begin{cases}$$

$$\ddot{\theta}_1 + \gamma \dot{\theta}_1 + \frac{g}{l} \sin(\theta_1) + \kappa (\theta_1 - \theta_2)^2 = F_0 \cos(\omega t), \backslash \backslash$$

$$\ddot{\theta}_2 + \gamma \dot{\theta}_2 + \frac{g}{l} \sin(\theta_2) + \kappa (\theta_2 - \theta_1)^2 = F_0 \cos(\omega t).$$

$$\backslash \end{cases}$$

$$\backslash$$

Estructura del Proyecto

pendulos_proyecto/ +- include/ | +- [Pendulo.h](#) +- src/ | +- [Pendulo.cpp](#) | +- [main.cpp](#) +- scripts/ | +- plot.py +- results/ +- documents/ | +- analysis.tex | +- flowchart.png +- Makefile +- Doxyfile +- [README.md](#)

1. Compilar el proyecto

```
make

\###2. Ejecutar la simulación

\###El programa te guiará para introducir los parámetros.

make run

\###Los datos se guardan en results/simulation\_data.csv.

\###3. Generar gráficos

make plot

\###Crea theta\_vs\_t.png, phase\_space.png y energy.png en results/.

\###4. Generar documentación

make doc

\###Abre docs/html/index.html.

\###5. Generar reporte PDF

\###Edita documents/analysis.tex y compila:

make report

---

\#### \*\*ARCHIVO: `include/Pendolo.h`\*\*
cpp
#ifndef PENDULO_H
#define PENDULO_H
#include <cmath>
/**
*
```

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Params	Almacena los parámetros físicos y de simulación del sistema	7
Pendulo	Representa un péndulo individual en el sistema acoplado	8

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

include/ Pendulo.h	13
src/ main.cpp	14
src/ Pendulo.cpp	15

Chapter 4

Class Documentation

4.1 Params Struct Reference

Almacena los parámetros físicos y de simulación del sistema.

```
#include <Pendulo.h>
```

Public Attributes

- double `g` = 9.81
Aceleración gravitacional (m/s^2)
- double `l` = 9.81
Longitud del péndulo (m)
- double `m` = 1.0
Masa del péndulo (kg)
- double `kappa` = 0.5
Constante de acoplamiento no lineal.
- double `gamma` = 0.1
Coefficiente de amortiguamiento.
- double `F0` = 0.0
Amplitud del forzamiento.
- double `omega` = 1.0
Frecuencia angular del forzamiento.

4.1.1 Detailed Description

Almacena los parámetros físicos y de simulación del sistema.

```
*  
*/  
struct Params {  
    // Parámetros básicos  
    double g = 9.81; ///Aceleración gravitacional ( $m/s^2$ )  
    double l = 9.81; ///Longitud del péndulo (m)  
    double m = 1.0; ///Masa del péndulo (kg)  
    double kappa = 0.5; ///Constante de acoplamiento no lineal  
    // Parámetros del modelo extendido  
    double gamma = 0.1; ///Coefficiente de amortiguamiento  
    double F0 = 0.0; ///Amplitud del forzamiento  
    double omega = 1.0; ///Frecuencia angular del forzamiento  
};  
/**  
*
```

4.1.2 Member Data Documentation

4.1.2.1 F0

```
double Params::F0 = 0.0
```

Amplitud del forzamiento.

4.1.2.2 g

```
double Params::g = 9.81
```

Aceleración gravitacional (m/s^2)

4.1.2.3 gamma

```
double Params::gamma = 0.1
```

Coefficiente de amortiguamiento.

4.1.2.4 kappa

```
double Params::kappa = 0.5
```

Constante de acoplamiento no lineal.

4.1.2.5 l

```
double Params::l = 9.81
```

Longitud del péndulo (m)

4.1.2.6 m

```
double Params::m = 1.0
```

Masa del péndulo (kg)

4.1.2.7 omega

```
double Params::omega = 1.0
```

Frecuencia angular del forzamiento.

The documentation for this struct was generated from the following file:

- [include/Pendolo.h](#)

4.2 Pendulo Class Reference

Representa un péndulo individual en el sistema acoplado.

```
#include <Pendulo.h>
```

Public Member Functions

- [Pendulo](#) (double theta0, double omega0)
Constructor de la clase [Pendulo](#).
- double [get_theta](#) () const
- double [get_omega](#) () const
- void [set_theta](#) (double new_theta)
- void [set_omega](#) (double new_omega)
- double [individual_energy](#) (const [Params](#) &p) const
Calcula la energía total de este péndulo (cinética + potencial gravitacional).

Private Attributes

- double `theta_`
Ángulo actual en radianes.
- double `omega_`
Velocidad angular actual en rad/s.

4.2.1 Detailed Description

Representa un péndulo individual en el sistema acoplado.

```

*
*/
class Pendulo {
private:
    double theta_; ///< Ángulo actual en radianes.
    double omega_; ///< Velocidad angular actual en rad/s.
public:
    /**
    * Constructor de la clase Pendulo.
    *

```

Parameters

<code>theta0</code>	Ángulo inicial.
---------------------	-----------------

```

*

```

Parameters

<code>omega0</code>	Velocidad angular inicial.
---------------------	----------------------------

```

*/
Pendulo(double theta0, double omega0);
// Getters y Setters
double get_theta() const { return theta_; }
double get_omega() const { return omega_; }
void set_theta(double new_theta) { theta_ = new_theta; }
void set_omega(double new_omega) { omega_ = new_omega; }
/**
* Calcula la energía total de este péndulo (cinética + potencial gravitacional).
*

```

Parameters

<code>p</code>	Estructura con los parámetros físicos.
----------------	--

```

*

```

Returns

La energía individual del péndulo.

```

*/
double individual_energy(const Params& p) const;
};
/**
* Calcula la aceleración angular de un péndulo en un instante dado.
*

```

Parameters

<i>theta_this</i>	Ángulo del péndulo para el cual se calcula la aceleración.
-------------------	--

*

Parameters

<i>omega_this</i>	Velocidad angular del mismo péndulo.
-------------------	--------------------------------------

*

Parameters

<i>theta_other</i>	Ángulo del otro péndulo (para el acoplamiento).
--------------------	---

*

Parameters

<i>t</i>	Tiempo actual (para el forzamiento).
----------	--------------------------------------

*

Parameters

<i>p</i>	Estructura con los parámetros físicos del sistema.
----------	--

*

Returns

La aceleración angular (theta_ddot).

```
*/
double calculate_accel(double theta_this, double omega_this, double theta_other, double t, const Params& p);
#endif // PENDULO_H
```

4.2.2 Constructor & Destructor Documentation**4.2.2.1 Pendulo()**

```
Pendulo::Pendulo (
    double theta0,
    double omega0 )
```

Constructor de la clase [Pendulo](#).

Parameters

<i>theta0</i>	Ángulo inicial.
<i>omega0</i>	Velocidad angular inicial.

4.2.3 Member Function Documentation**4.2.3.1 get_omega()**

```
double Pendulo::get_omega ( ) const [inline]
```

4.2.3.2 get_theta()

```
double Pendulo::get_theta ( ) const [inline]
```

4.2.3.3 individual_energy()

```
double Pendulo::individual_energy (
    const Params & p ) const
```

Calcula la energía total de este péndulo (cinética + potencial gravitacional).

Parameters

p	Estructura con los parámetros físicos.
-----	--

Returns

La energía individual del péndulo.

4.2.3.4 set_omega()

```
void Pendulo::set_omega (
    double new_omega ) [inline]
```

4.2.3.5 set_theta()

```
void Pendulo::set_theta (
    double new_theta ) [inline]
```

4.2.4 Member Data Documentation

4.2.4.1 omega_

```
double Pendulo::omega_ [private]
```

Velocidad angular actual en rad/s.

4.2.4.2 theta_

```
double Pendulo::theta_ [private]
```

Ángulo actual en radianes.

The documentation for this class was generated from the following files:

- include/[Pendulo.h](#)
- src/[Pendulo.cpp](#)

Chapter 5

File Documentation

5.1 include/Pendulo.h File Reference

```
#include <cmath>
```

Include dependency graph for Pendulo.h: This graph shows which files directly or indirectly include this file:

Classes

- struct [Params](#)
Almacena los parámetros físicos y de simulación del sistema.
- class [Pendulo](#)
Representa un péndulo individual en el sistema acoplado.

Functions

- double [calculate_accel](#) (double theta_this, double omega_this, double theta_other, double t, const [Params](#) &p)
Calcula la aceleración angular de un péndulo en un instante dado.

5.1.1 Function Documentation

5.1.1.1 calculate_accel()

```
double calculate_accel (  
    double theta_this,  
    double omega_this,  
    double theta_other,  
    double t,  
    const Params & p )
```

Calcula la aceleración angular de un péndulo en un instante dado.

Parameters

<i>theta_this</i>	Ángulo del péndulo para el cual se calcula la aceleración.
<i>omega_this</i>	Velocidad angular del mismo péndulo.
<i>theta_other</i>	Ángulo del otro péndulo (para el acoplamiento).
<i>t</i>	Tiempo actual (para el forzamiento).
<i>p</i>	Estructura con los parámetros físicos del sistema.

Returns

La aceleración angular (θ_{ddot}).

5.2 Pendulo.h

[Go to the documentation of this file.](#)

```
00001 #ifndef PENDULO_H
00002 #define PENDULO_H
00003
00004 #include <cmath>
00005
00010 struct Params {
00011     // Parámetros básicos
00012     double g = 9.81;
00013     double l = 9.81;
00014     double m = 1.0;
00015     double kappa = 0.5;
00016
00017     // Parámetros del modelo extendido
00018     double gamma = 0.1;
00019     double F0 = 0.0;
00020     double omega = 1.0;
00021 };
00022
00027 class Pendulo {
00028 private:
00029     // --- ESTAS LÍNEAS FALTABAN ---
00030     double theta_;
00031     double omega_;
00032
00033 public:
00039     Pendulo(double theta0, double omega0);
00040
00041     // Getters y Setters
00042     double get_theta() const { return theta_; }
00043     double get_omega() const { return omega_; }
00044     void set_theta(double new_theta) { theta_ = new_theta; }
00045     void set_omega(double new_omega) { omega_ = new_omega; }
00046
00052     // --- ESTA DECLARACIÓN FALTABA ---
00053     double individual_energy(const Params& p) const;
00054 };
00055
00065 double calculate_accel(double theta_this, double omega_this, double theta_other, double t, const
Params& p);
00066
00067 #endif // PENDULO_H
```

5.3 README.md File Reference

5.4 src/main.cpp File Reference

```
#include <iostream>
#include <vector>
#include <fstream>
#include <string>
#include "Pendulo.h"
```

Include dependency graph for main.cpp:

Functions

- void [get_user_input](#) (int &method, double &dt, double &t_max, [Params](#) &p, double &th1_0, double &w1_0, double &th2_0, double &w2_0)
- double [get_coupling_energy](#) (const [Pendulo](#) &p1, const [Pendulo](#) &p2, const [Params](#) &p)
- int [main](#) ()

5.4.1 Function Documentation

5.4.1.1 [get_coupling_energy\(\)](#)

```
double get_coupling_energy (
```

```
const Pendulo & p1,  
const Pendulo & p2,  
const Params & p )
```

5.4.1.2 get_user_input()

```
void get_user_input (   
    int & method,  
    double & dt,  
    double & t_max,  
    Params & p,  
    double & th1_0,  
    double & w1_0,  
    double & th2_0,  
    double & w2_0 )
```

5.4.1.3 main()

```
int main ( )
```

5.5 src/Pendulo.cpp File Reference

```
#include "Pendulo.h"
```

Include dependency graph for Pendulo.cpp:

Index

- calculate_accel
 - Pendolo.h, [13](#)
- F0
 - Params, [8](#)
- g
 - Params, [8](#)
- gamma
 - Params, [8](#)
- get_coupling_energy
 - main.cpp, [14](#)
- get_omega
 - Pendolo, [10](#)
- get_theta
 - Pendolo, [10](#)
- get_user_input
 - main.cpp, [15](#)
- include/Pendolo.h, [13](#), [14](#)
- individual_energy
 - Pendolo, [11](#)
- kappa
 - Params, [8](#)
- l
 - Params, [8](#)
- m
 - Params, [8](#)
- main
 - main.cpp, [15](#)
- main.cpp
 - get_coupling_energy, [14](#)
 - get_user_input, [15](#)
 - main, [15](#)
- omega
 - Params, [8](#)
- omega_
 - Pendolo, [11](#)
- Params, [7](#)
 - F0, [8](#)
 - g, [8](#)
 - gamma, [8](#)
 - kappa, [8](#)
 - l, [8](#)
 - m, [8](#)
 - omega, [8](#)
- Pendolo, [8](#)
 - get_omega, [10](#)
 - get_theta, [10](#)
 - individual_energy, [11](#)
 - omega_, [11](#)
 - Pendolo, [10](#)
 - set_omega, [11](#)
 - set_theta, [11](#)
 - theta_, [11](#)
- Pendolo.h
 - calculate_accel, [13](#)
- README, [1](#)
- README.md, [14](#)
- set_omega
 - Pendolo, [11](#)
- set_theta
 - Pendolo, [11](#)
- src/main.cpp, [14](#)
- src/Pendolo.cpp, [15](#)
- theta_
 - Pendolo, [11](#)