Proyecto Fin de Carrera
Ingeniería Electrónica Robótica y Mecatrónica

# Integración de herramientas computacionales para sistemas dinámicos

Autor: Pablo Pardo López

Tutor: Jorge Galán Vioque

**Dpto. Matemática Aplicada II**
**Escuela Técnica Superior de Ingeniería**
**Universidad de Sevilla**

Sevilla, 2019

Proyecto Fin de Carrera
Ingeniería Electrónica Robótica y Mecatrónica

# Integración de herramientas computacionales para sistemas dinámicos


Autor:
Pablo Pardo López

Tutor:
Jorge Galán Vioque
Catedrático

Dpto. Matemática Aplicada II
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla



Sevilla, 2019

Proyecto Fin de Carrera: Integración de herramientas computacionales para sistemas dinámicos

Autor: Pablo Pardo López
Tutor: Jorge Galán Vioque

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

# Resumen

La herramienta empleada para caracterizar sistemas dinámicos continuos en el tiempo bajo la idea de estudiarlos discretizando esta dimensión es conocida como *sección de Poincaré*. El objetivo de este TFG es implementar dicha herramienta en *Matlab* y, como ejemplo, emplearla para caracterizar y estudiar posteriormente (mediante el *gui* de *Matlab*) un sistema complejo conocido como el péndulo-muelle.

Este sistema es el resultante de la combinación de dos elementos simples: péndulo ideal y de un muelle funcionando como oscilador armónico.

# Abstract

The tool employed to characterize dynamic systems continuous in time under the idea of studying them discretizing this dimension is known as *Poincaré map*. The aim of this TFG is to implement this tool in matlab and, as an example, employ it to characterize and study afterwards (using the *Matlab gui*) a complex system known as spring-pendulum.

This system is the result of the combination of two simple elements: ideal pendulum and a spring working as a harmonic oscillator.

# Índice

# 1 Introducción

Tanto en la naturaleza, como en la industria y en cualquier otro ámbito existen sistemas que evolucionan con el tiempo. Estos, son conocidos como sistemas dinámicos. Su evolución está gobernada por un conjunto de reglas que especifican su estado a cada valor del tiempo. Estos sistemas pueden ser modelados matemáticamente mediante ecuaciones discretas y, por tanto, ser estudiados.

Los sistemas pueden llegar a ser muy sensibles tanto a la variación de las condiciones iniciales como a la variación de sus parámetros. Cambios en los parámetros pueden llegar a causar modificaciones en el comportamiento del sistema haciendo que aparezcan y desaparezcan soluciones, que estas cambien de estabilidad o incluso rupturas de simetría. Estos puntos en los que ocurren dichos cambios suelen ser de gran interés y se les conoce como puntos de bifurcación, son estudiables mediante un *Diagrama de Bifurcación*.

Sin embargo, en este proyecto no se realizará ningún estudio de bifurcación, la herramienta elegida para estudiar la dinámica del sistema en cuestión es la *Sección de Poincaré*. Con ella se podrá estudiar la dinámica del sistema para distintas condiciones iniciales para un valor de los parámetros dados y, observando la variación de las secciones al variar los parámetros, podremos detectar valores en los que se producen bifurcaciones.

El sistema que se estudiará en el proyecto es el *péndulo-muelle*. Un sistema que, a pesar de ser la composición de otros dos muy simples, da lugar a una dinámica mucho más compleja, rica e interesante.

## 1.1 Objetivos

En este proyecto se plantean los siguientes objetivos:

- Implementación computacional de la Sección de Poincaré.
- Crear una herramienta de visualización para estudiar los resultados.
- Y, como objetivo principal, el análisis del comportamiento dinámico para todos los valores de los parámetros y todas las condiciones iniciales así como identificar los valores para los que hay un cambio cualitativo.

Este proyecto podría verse como la continuación de [1] y, en especial de [2], los cuales de distinta forma implementan la Sección de Poincaré y caracterizan el mismo sistema sin llegar a estudiarlo.

## 1.2 Contenido y metodología

El proyecto quedará dividido en dos partes fundamentales: la primera, cuya resolución consta de la mayor parte del tiempo del proyecto, consiste en todo el problema computacional con respecto a la obtención de las diversas secciones de Poincaré; y una segunda, que como se ha explicado en los objetivos, consiste en el análisis de parte de la dinámica del sistema.

Todo el proyecto será realizado mediante un único software: *Matlab*. A pesar del enorme coste computacional para obtener las secciones, lo cual haría más adecuado a un lenguaje de programación compilado, se ha decidido usar un intérprete debido a su flexibilidad, a las capacidades gráficas que nos ofrece este en concreto y, para poder tener todo el proyecto integrado en un único software.

# 2 Modelo dinámico

Previo a la realización del apartado computacional es necesario obtener un modelo dinámico del sistema. Para ello es importante entender, en primera instancia, los sistemas simples que lo conforman.

## 2.1 Péndulo simple y ley de Hook

El péndulo simple, también conocido como péndulo matemático o ideal, es un sistema constituido por un hilo unido a un punto fijo y, en el otro extremo a una masa suspendida (realizando un movimiento por un único plano vertical) . Este péndulo solo es accesible teóricamente pues, el hilo se considera inextensible y sin masa, y la masa del extremo puntual. Posee un grado de libertad que es el ángulo que forma el hilo con la vertical.

Sin embargo, hay casos en los que tanto esta deformación, como la masa del hilo son despreciables (la extensión con respecto a la longitud del propio hilo, y la del peso con respecto a la masa que cuelga del extremo). Dando lugar a péndulos físicos con comportamientos prácticamente idénticos al simple.

Este tipo de péndulos son los que aprovechó *Galileo* para descubrir que el período de oscilación de estos no dependía de la amplitud de oscilación, si no de la longitud del hilo dando a lugar a una frecuencia de oscilación:

$$w_p = \sqrt{\frac{g}{l}},$$

con $g$ la gravedad y $l$ la longitud del péndulo.

Por otro lado, si sustituimos el hilo por un resorte de compresión y fijamos el ángulo que forma este con la vertical, estamos ante un oscilador armónico regido por la *ley de Hook*. El movimiento generado por este sistema es de un único grado de libertad y lo realiza sobre la recta que forma la masa con el muelle.

La *ley de Hook* establece una fuerza proporcional y de sentido contrario a la extensión causada sobre el muelle, tal que:

$$F = -k\delta,$$

con $k$ la constante elástica del muelle y $\delta$ la elongación.

Dando lugar a una oscilación de frecuencia angular:

$$w_m = \sqrt{\frac{k}{m}}$$

## 2.2 Péndulo-muelle

Partiendo de estos dos sistemas y fusionándolos daría lugar al péndulo-muelle. O lo que es lo mismo, sustituyendo en el péndulo el hilo inextensible y sin masa por un resorte sin masa obtendríamos el péndulo-muelle, un sistema que a priori podría parecer sencillo como los que lo conforman y que sin embargo, resulta un sistema complejo con dos grados de libertad: el ángulo que forma el muelle con la vertical, y la distancia entre el punto fijo y la masa (la longitud del muelle).

Para modelarlo hay que tener en cuenta las siguientes hipótesis:

- El sistema está completamente equilibrado, así pues, toda trayectoria del péndulo muelle se encontraría dentro del plano $Oxy$.
- La energía del sistema se conserva, no hay pérdidas.
- Supondremos que la masa del sistema es puntual y se encuentra en el extremo final, es decir, la masa del muelle es despreciable con respecto al peso que cuelga del mismo (péndulo simple).

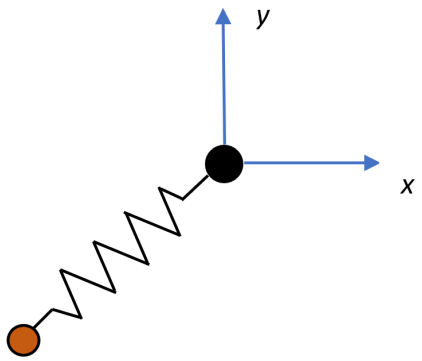


**Figura 2.1** Péndulo-muelle.

Tal y como se indica en la Figura 2.1: el eje $y$ se ha escogido para que sea paralelo al muelle en estado de reposo; el eje $x$ para que sea perpendicular al mismo y forme junto con el eje $y$ el plano de movimiento del sistema; y, el origen, será el punto del que cuelga el muelle.

### 2.2.1 Ecuaciones de Newton

Así pues mediante las ecuaciones de *Newton* obtenemos las siguientes ecuaciones de movimiento:

$$\ddot{x} = -\frac{k}{m}x\left(1 - \frac{l_0}{\sqrt{x^2+y^2}}\right), \tag{2.1}$$

$$\ddot{y} = -g - \frac{k}{m}y\left(1 - \frac{l_0}{\sqrt{x^2+y^2}}\right). \tag{2.2}$$

En donde podemos observar que es un sistema autónomo, es decir, no depende explícitamente del tiempo.

### 2.2.2 Parámetros

Los parámetros que aparecen en las ecuaciones son:

- $k$ : constante elástica del resorte.

- $l_0$ : longitud natural del resorte, siendo esta la que mide el resorte cuando se encuentra sin deformar debido a la acción de una fuerza externa.
- $g$ : gravedad.
- $m$ : Masa del sistema.

### 2.2.3 Energía

La energía del sistema quedaría:

$$E = \frac{1}{2}m\left(\dot{x}^2+\dot{y}^2\right)+mgy+\frac{1}{2}k\left(\sqrt{x^2+y^2}-l_0\right)^2 \tag{2.3}$$

### 2.2.4 Adimensionalización

Como podemos observar aparecen hasta cinco parámetros (contando la energía) lo cual hace del estudio del sistema algo verdaderamente tedioso pues, para poder estudiar la dinámica al completo habría que estudiar todas las combinaciones posibles de los valores de dichos parámetros. La técnica empleada para solucionar este problema será la adimensionalización. En este caso el parámetro adimensionalizado sería:

$$\lambda = \frac{kl_0}{mg} \qquad con \quad \lambda \geq 0, \tag{2.4}$$

ya que todos los parámetros que la conforman, por naturaleza, son estrictamente reales y no negativos. Las variables adimensionales del sistema escogidas son:

$$q_1 = \frac{k}{mg}x, \tag{2.5}$$

$$q_2 = \frac{k}{mg}y, \tag{2.6}$$

$$p_1 = \dot{x}\frac{1}{g}\sqrt{\frac{k}{m}}, \tag{2.7}$$

$$p_2 = \dot{y}\frac{1}{g}\sqrt{\frac{k}{m}}. \tag{2.8}$$

#### Hamiltoniano

Estas variables han sido elegidas para que el *Hamiltoniano*, que en este caso es la energía del sistema adimensionalizadada, quede de la siguiente forma:

$$H = \frac{p_1^2}{2}+\frac{p_2^2}{2}+\frac{1}{2}\left(\sqrt{q_1^2+q_2^2}-\lambda\right)^2+q_2+\lambda+1/2 \tag{2.9}$$

En donde: $H = \frac{E}{(m^2g^2)/k}+\lambda+\frac{1}{2}$. En este caso, podemos observar que se le ha añadido un término constante a la energía (que corresponde al punto elegido en donde la energía potencial sea cero), esto es para que $\forall\lambda : H \in [0,+\infty)$. Es decir, para que, en el punto de equilibrio estable, que es el punto de mínima energía, se dé siempre en $H = 0$.

De esta forma, son dos los parámetros que conforman el sistema: $\lambda$ y $H$. Siendo el primero una característica del sistema; y el segundo la energía, que está relacionada con las condiciones iniciales.

#### Ecuaciones de Newton adimensionalizadas

Partiendo del *Hamiltoniano* podemos obtener directamente el modelo dinámico teniendo en cuenta que: $\dot{q} = \frac{\partial H}{\partial p}$; y $\dot{p} = -\frac{\partial H}{\partial q}$. Quedándonos así el siguiente sistema de ecuaciones:

$$\left.\begin{aligned} \dot{q_1} &= p_1, \\ \dot{q_2} &= p_2, \\ \dot{p_1} &= -q_1 + \lambda \frac{q_1}{\sqrt{q_1^2 + q_2^2}}, \\ \dot{p_2} &= -q_2 + \lambda \frac{q_2}{\sqrt{q_2^2 + q_2^2}} - 1. \end{aligned}\right\} \tag{2.10}$$

De esta forma se queda un sistema ODE (ecuación diferencial ordinaria) de primer orden con cuatro ecuaciones, listo para el integrador.

### 2.2.5 Equilibrios

En el péndulo muelle hay dos puntos de equilibrio que dependen del valor de $\lambda$, ambos situados sobre el eje vertical: uno estable, que es el punto de mínima energía; otro inestable, siendo este el punto de mínima energía para el que $q_2 > 0$ y $q_1 = 0$. Si sustituimos las ecuaciones (2.10) por las condiciones de equilibrio ($\dot{q} = 0; \dot{p} = 0$). Obtenemos que los puntos de equilibrio son:

- Estable $(0, -\lambda - 1)$.
- Inestable $(0, \lambda - 1)$ con $\lambda > 1$ y para $H = 2\lambda$.

# 3 Herramientas Computacionales

En este capítulo se explicarán todas las herramientas computacionales que han sido necesarias para realizar las simulaciones y obtener los datos del modelo del sistema, y para su posterior análisis.

El apartado computacional quedará dividido en dos programas principales:

- El primero se encargará de obtener de manera automática y para un valor fijo de $\lambda$, una serie de secciones de Poincaré (que evolucionan con el valor de H) y de guardar los datos en un archivo.
- El segundo como se vio en los objetivos del proyecto, es la herramienta de visualización, que nos permita leer los archivos creados por el primero, y analizarlos.

A la hora de crear dichas herramientas se han tenido en cuenta dos aspectos fundamentales en cada uno de los apartados:

- Coste computacional. El coste de resolver el sistema es muy grande, convirtiendo este problema en el de mayor prioridad.
- Memoria. La cantidad de datos necesarios para caracterizar el sistema es enorme, por lo que reducir estos al mínimo necesario es una tarea importante.

## 3.1 Sección de Poincaré

Antes de realizar cualquier simulación y obtener los datos es fundamental saber cómo deseamos representar esa información para así poder discernir entre qué datos nos interesan y cuáles no.

Sea $\dot{x} = f(x), \quad x \in \mathbb{R}^n$ un sistema diferencial de ecuaciones ordinario, podemos definir un subespacio de $n-1$ dimensiones en donde el flujo del sistema sea transversal a dicho subespacio. Esta transversalidad es necesaria pues teniendo una solución en el sistema, estás órbitas cortarían a través del plano (o mapa de Poincaré).

El principal problema de la *sección de Poincaré* es obtener un subespacio tal que se pueda caracterizar por completo el sistema pues, no existe un método generalizado para este problema y hay varias soluciones (varios planos) posibles. Así, el objetivo principal será encontrar un plano tal que todas las órbitas de interés intersequen al mismo. [4].

Para entender mejor la correspondencia entre el sistema original y la sección de Poincaré como ejemplo podríamos ver que:

- Una órbita periódica simple del sistema dinámico original se convierte en un único punto en la sección de Poincaré.
- Una órbita cuasiperiódica en una curva cerrada.
- Una trayectoria caótica en un conjunto de puntos inconexos.
- Una órbita cuasiperiódica con n-multiplicación de periodo en n curvas cerradas.

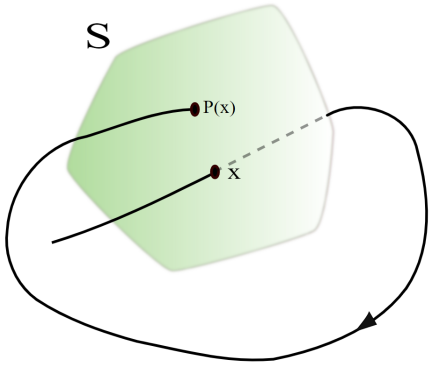


**Figura 3.1** Aplicación de Poincaré.

Con esto se puede notar el potencial de esta herramienta debido a que reducir la dimensión de un sistema facilita la visualización del mismo. En nuestro caso, partimos de un sistema $\mathbb{R}^4$ el cual convertiremos en uno $\mathbb{R}^2$: una debido al plano; y otra, como se explica en el siguiente subapartado, debido a la condición de la conservación de la energía del sistema.

### 3.1.1 Reducción de dimensionalidad

En el capítulo del modelo dinámico obtuvimos que $H = H(q, p, \lambda)$ siendo H la energía, y por tanto (como se indica en las hipótesis), constante durante la simulación de una trayectoria. A partir de esa ecuación podemos y obtener:

$$p_1 = g(q_1, q_2, p_2, \lambda, H) \tag{3.1}$$

En (3.1) podemos ver cómo una de las cuatro variables puede ser escrita en función de las otras quedando un sistema $\mathbb{R}^3$.

El último paso es definir el subespacio, en este caso, cambiar de un sistema continuo en el tiempo a un sistema discreto. Esto significa que pasaremos de un sistema en donde $t \in (-\infty, +\infty)$ a otro en que $t = t_1, t_2, t_3 ... t_k ...$ En nuestro caso utilizaremos $q_1 = 0$ como evento discreto, pudiendo así conseguir una representación bidimensional en la que tendríamos los valores de $q_2$ y $p_2$ cuando se produce un paso por la verical. Lo cual sucede cuando el valor de $q_1$ cambia de signo. Para dicho evento solo se ha decidido seleccionar los puntos en los que el valor de $q_1$ pasa de negativo a positivo, esto es debido a que ambos (salvo alguna excepción puntual) dan el mismo tipo de información y, por tanto, representando ambos casos solo conseguiríamos superposicionarlos en la sección.

Hemos escogido este evento discreto debido a que por naturaleza toda dinámica del péndulo-muelle pasa por ese plano, pudiéndose así analizar todas las órbitas existentes para un valor concreto de la energía y del parámetro $\lambda$ estudiando la sección de Poincaré.

Sin embargo, hay una excepción, esta ocurre cuando el sistema se comporta exclusivamente como un muelle, debido a que esta solución no interseca con el plano vertical, sino que que se desplaza dentro de este.

### 3.1.2 Ejemplo

En el ejemplo de la Figura 3.2 podemos observar que hay una región en dónde hay valores de $q_2$ y $p_2$ no son válidos. La curva que delimita ambas regiones se puede obtener de la ecuación (3.1).

## 3.2 Integración del sistema de ecuaciones diferenciales

Una vez que se han conseguido las ecuaciones y se ha escogido el método en el que se quiere representar al sistema, el siguiente paso es seleccionar un integrador y configurarlo para que lo resuelva. En nuestro

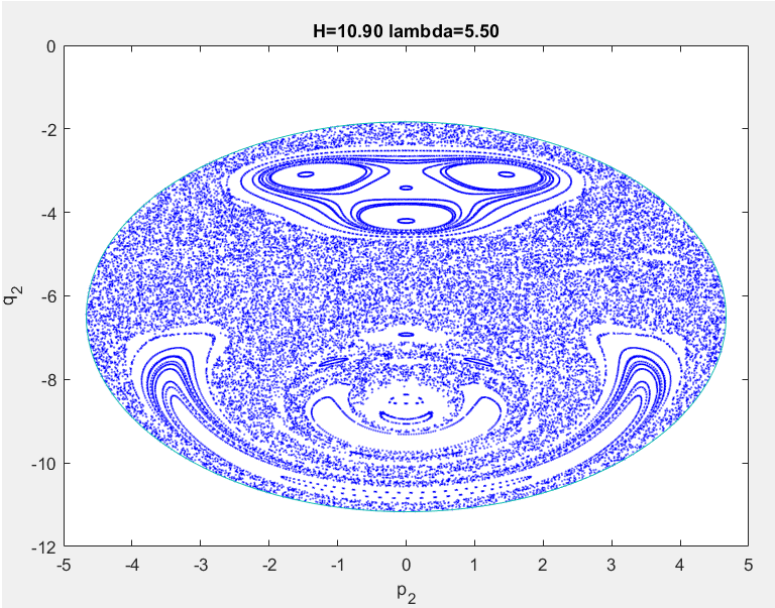


**Figura 3.2** Ejemplo de una Sección de Poincaré.

caso el integrador en cuestión es ODE45, uno de los solver que posee el software de *Matlab* para ecuaciones diferenciales no rígidas, el cual se basa en el método de Dormand-Prince [3].

Aquí es donde encontramos una de las mayores ventajas de haber escogido *Matlab* para la realización del proyecto. Y es que, no es solo que disponga de un grupo de integradores completamente funcionales y fáciles de usar, sino que también, estos poseen la opción de detección de eventos. Esto quiere decir que con tal de activar los eventos en el apartado '*option*'de ODE45 y de definir cuándo se produce el evento, queda resuelto el problema de detectar un paso por cero del eje $x$ y de interpolar las soluciones. Este problema surge porque las soluciones se obtienen en un paso de $t$ y, por ello $q_1$ nunca vale exactamente cero; así pues, aunque detectar entre qué puntos se produce el paso por cero sea trivial, si deseamos obtener resultados de alta precisión habrá que realizar una interpolación más compleja que la lineal.

### 3.2.1 Configuración del integrador

Nuestro integrador quedaría:

**Código 3.1** Código integrador.

```
options=odeset('events','on','RelTol',TOL,'AbsTol',TOL);
[t,y,te,ye,ie]=ode45('pendulum', tspan, y0,options,lambda,0,1);
```

En donde en la primera linea se indican las opciones, y en la segunda se ejecuta el integrador. Los datos suministrados al integrador (de izquierda a derecha) son:

- Función pendulum. Es un archivo con la extensión *.m, en el cual principalmente está el sistema ODE, ecuaciones (2.10), y en el que también se define el evento.
- tspan. Ventana de tiempo en la que queremos obtener las soluciones del sistema.
- y0. Vector de condiciones iniciales.
- options. En el cual activamos los eventos e indicamos la tolerancia del error.
- El resto son argumentos para la función pendulum.

El integrador a la salida suministra los puntos que conforman la solución para dichas condiciones iniciales y durante una ventana de tiempo *tspan*. También entrega los valores de tiempo y de $p$ y $q$ en los que se produce un evento, estos últimos formarán parte de la sección de Poincaré.

#### Función pendulum

**Código 3.2** Fichero pendulum.m.

```
function [out1,out2,out3] = pendulum(t,y,flag,lambda,para,dir)
if nargin < 3 | isempty(flag)
out1 = zeros(4,1);
out1(1) = y(2);
out1(2) = -y(1)+(lambda*y(1))/(sqrt(y(1)^2+y(3)^2));
out1(3) = y(4);
out1(4) = -1+(lambda*y(3))/(sqrt(y(1)^2+y(3)^2))-y(3);
else
switch(flag)
case 'events'
q1 = y(1);
out1 = [q1];
out2 = [para];
out3 = [dir];
otherwise
error(['Unknown flag '])
end
end
```

En el caso de Código 3.2 definimos los eventos junto con las ecuaciones del sistema. Otra opción completamente equivalente a esta sería definir una función de eventos independiente.

### 3.2.2 Resolución del integrador

Para poder obtener una imagen de la *Sección de Poincaré* con una buena calidad dos aspectos son importantes:

- Una buena precisión en la integración.
- Un tiempo de simulación apropiado para poder obtener un número de puntos suficiente.

El primero de ellos corresponde a la tolerancia, y el segundo al *tspan*. Sin embargo, un aumento desmesurado de cualquiera de ellos podría provocar un tiempo de computación excesivo, por lo que es importante escoger los valores menos exigentes que cumplan el requisito de calidad. Para este proyecto en los experimentos principalmente se ha utilizado una tolerancia de $10^{-8}$ y un tiempo de simulación de 9.000.

Si escogemos unas condiciones iniciales tal que el comportamiento del muelle sea una trayectoria cuasiperiódica, lo cual correspondería una curva cerrada en su sección correspondiente, los efectos de una tolerancia o tiempo de simulación insuficientes serían:

- Engrosamiento de la curva (lo que corresponde al error) llegando incluso a salirse de la misma en caso de una tolerancia muy pequeña.
- Curva abierta, incompleta, debido a un tiempo de simulación insuficiente. Este es más común encontrarlo puesto que por cuestión de periodicidad a veces, a pesar de tener un tiempo alto los puntos tienden a estar en los mismos sitios.

#### Ejemplos

Los ejemplos de a continuación se han realizado para un mismo valor de $\lambda$ y $H$, y para las mismas condiciones iniciales: $p_2 = -1.621; q_2 = -2.325$.

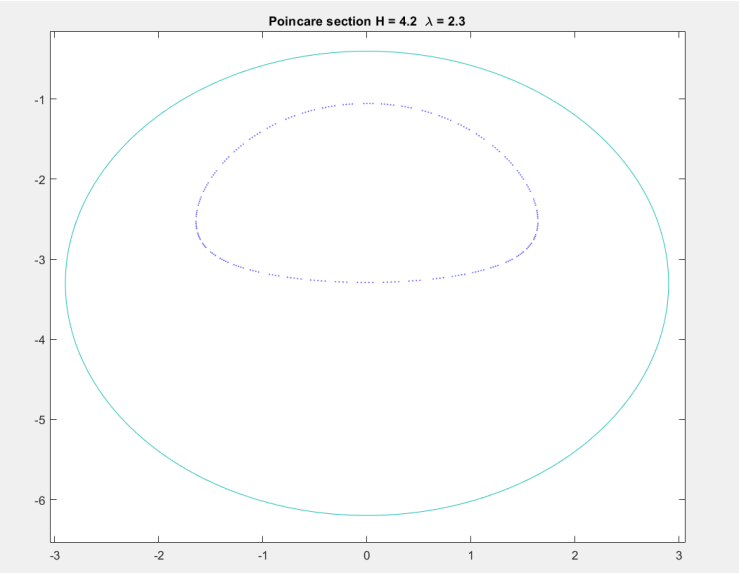


**Figura 3.3** $TOL = 10^{-8}$ $tf = 2.000$. Curva incompleta debido a un tiempo de simulación insuficiente.

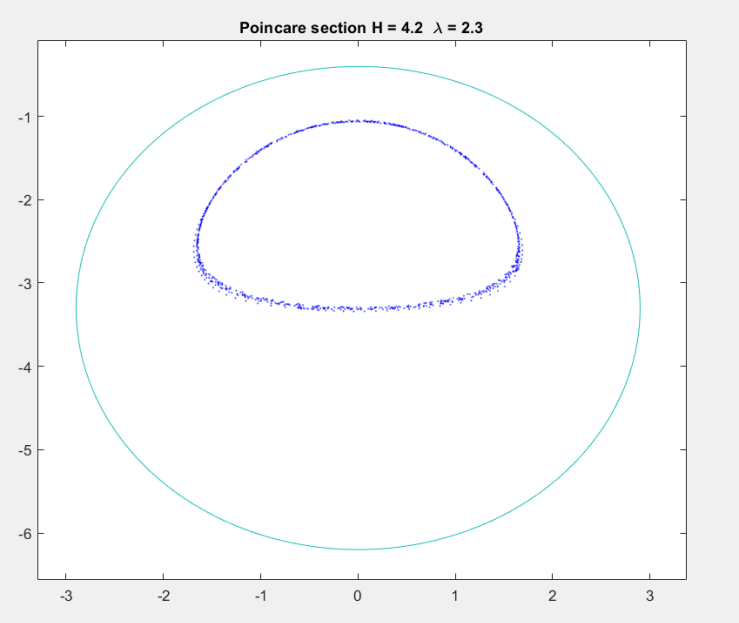


**Figura 3.4** $TOL = 10^{-4}$ $tf = 9.000$. Baja resolución dando lugar a errores notables en la solución (engrosamiento de la curva).

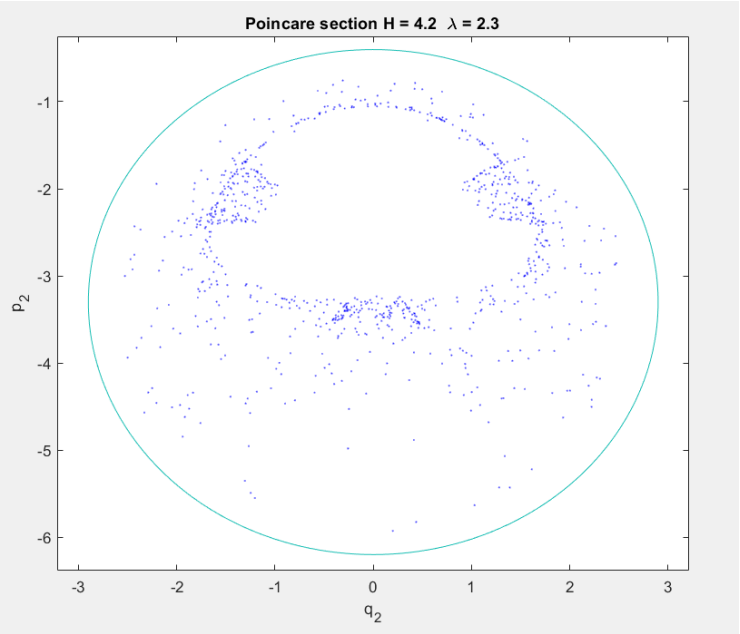


**Figura 3.5** $TOL = 10^{-3}$ $tf = 9.000$. Resolución muy baja dando lugar a errores tan grandes que la solución se separa de la órbita.

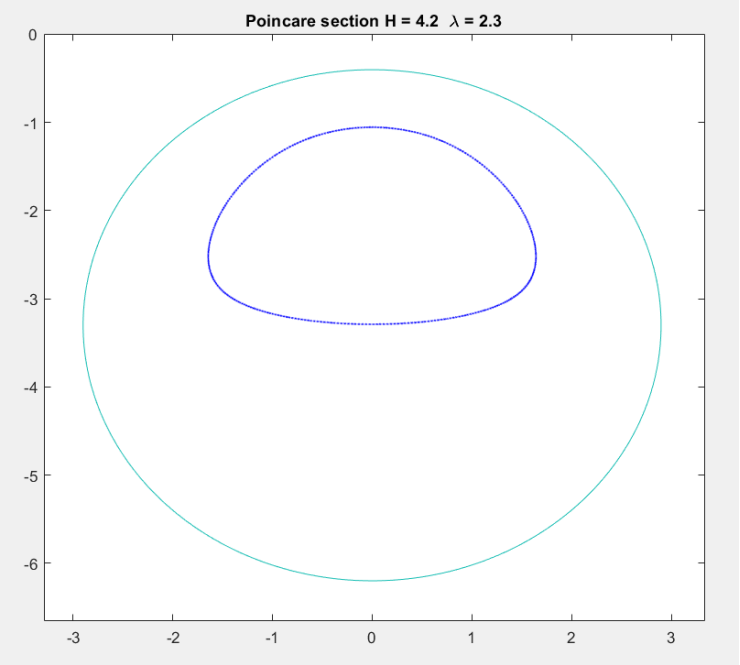


**Figura 3.6** $TOL = 10^{-8}$ $tf = 9.000$. Tiempo de simulación y resolución suficientes.

## 3.3 Mallado

Dados un valor de energía y de $\lambda$ el sistema en cuestión es muy sensible al cambio en las condiciones iniciales habiendo dinámicas completamente distintas. El objetivo de este apartado es escoger de forma automática un grupo de condiciones iniciales tal que quede completamente caracterizado la dinámica, es decir, obtengamos una *Sección de Poincaré* completa.

Como ya se indicado anteriormente, el parámetro $\lambda$ es el que nos define el sistema, mientras que H es otra condición inicial. Así pues, estando interesados en un valor concreto de $\lambda$ es importante estudiarlo $\forall H$ (una sección por cada H diferente). Y, dependiendo de esta, el rango de valores de las variables cambia. De esta forma, el problema principal es que cuando se quiera obtener una serie de secciones, a priori no se sabe el rango de las variables ni el paso óptimo del mallado.

### 3.3.1 Rango del mallado

El tipo de mallado, por sencillez, será rectangular, entre otras cosas es debido a que en un principio no deberíamos de saber la forma de la sección y si hay alguna manera óptima de realizar el mallado.

Para evitar cálculos innecesarios es necesario acotar las posibles soluciones. Para ello, se obtienen los valores máximos y mínimos de las variables $p_2$ y $q_2$. Y, dentro de este rango de valores, se realiza un mallado y se comprueban cuales entran dentro de la región de condiciones iniciales válidas. Aquellas que son válidas se incluirían en un vector de condiciones iniciales el cual se le pasaría al integrador (una ejecución de este por cada conjunto de condiciones iniciales).

Para obtener los valores máximo y mínimo (los extremos) de $p_2$ y $q_2$ hay dos formas de realizarlo:

- Solución numérica.
- Solución analítica.

Para este proyecto se ha decidido optar por la solución analítica debido a que las ecuaciones no son complejas.

#### Cálculo de los extremos

Previo al cálculo de los extremos hay que definir la curva. Esta delimita dos regiones: aquella cuyo conjunto de condiciones iniciales no son reales de las que sí, es decir, aquellos valores de $p_2$ y $q_2$ ($q_1 = 0$) que en la ecuación (3.1) den una solución real. En este caso:

$$p_1 = \pm\sqrt{2H - p_2^2 - \left(\sqrt{q_1^2 + q_2^2} - \lambda\right)^2 - 2q_2 - 2\lambda - 1} \tag{3.2}$$

En donde obtenemos que la curva queda definida por:

$$\mathscr{C} \equiv 2H - p_2^2 - (|q_2| - \lambda)^2 - 2q_2 - 2\lambda - 1 = 0, \tag{3.3}$$

puesto que $(3.3) > 0 \Leftrightarrow p_1 \in \mathbb{R}$ y si $(3.3) < 0 \Leftrightarrow p_1 \in \mathbb{C}$.

El cálculo analítico de los extremos es un problema de mínimos y máximos, así pues:

$$Si \quad \frac{\partial \mathscr{C}}{\partial p_2} = 0 \quad \Longrightarrow \quad p_2 = 0, \tag{3.4}$$

$$Si \quad \frac{\partial \mathscr{C}}{\partial q_2} = 0 \quad y \quad q_2 \leq 0 \quad \Longrightarrow \quad q_2 = -\lambda - 1, \tag{3.5}$$

$$Si \quad \frac{\partial \mathscr{C}}{\partial q_2} = 0 \quad y \quad q_2 > 0 \quad \Longrightarrow \quad q_2 = \lambda - 1 \qquad \text{si } \lambda \geq 1. \tag{3.6}$$

Sustituyendo dichos valores en (3.3) se obtienen las coordenadas de los extremos de la curva:

$$p_{2_{max,min}} = \pm\sqrt{2H}, \tag{3.7}$$

$$q_{2_{min}} = -\lambda - 1 - \sqrt{2H} \tag{3.8}$$

$$q_{2_{max}} = \begin{cases} -\lambda - 1 + \sqrt{2H} & \text{si } 0 < H < 2\lambda \\ \lambda - 1 + \sqrt{2(H - 2\lambda)} & \text{si } H \geq 2\lambda \end{cases} \tag{3.9}$$

### 3.3.2 Paso del mallado

Por último, hay que escoger el paso para realizar el mallado. La importancia que cobra este apartado vuelve a ser el coste computacional, esto es debido a que un paso demasiado pequeño en el mallado provocaría la obtención de un vector de condiciones iniciales muy grande, y, por tanto, demasiadas ejecuciones del integrador. Mientras que un paso demasiado grande implicaría que obtendríamos una sección en la que habría dinámicas sin caracterizar, o lo que es lo mismo, que se estudiará el sistema para un conjunto demasiado pequeño de condiciones iniciales. El mayor problema del paso es que el tamaño de las regiones que contienen las diferentes dinámicas aumenta con el tamaño de la sección, lo que quiere decir que el paso óptimo varía con H y que no se puede saber a priori.

Una posible solución para escoger el paso es obtenerlo en función del tamaño del rango de valores que tomen las variables, es decir, escoger como paso en el *eje*$q_2$ un porcentaje de $q_{2_{max}} - q_{2_{min}}$, y de la misma manera en el *eje* $p_2$. De esta forma, se consigue un número similar de condiciones iniciales para distintos valores de H. Aun así, este método, aunque sea mejor que escoger un paso fijo, no funciona del todo bien puesto que conforme evoluciona el valor de H, un mayor conjunto de condiciones iniciales es requerido.

Así pues, el método escogido para determinar el paso es el uso de un polinomio de primer grado en el que el término constante toma mayor importancia para los valores bajos de energía y el del primer grado para valores más grandes. Y en donde los coeficientes se ajustan experimentalmente, dependiendo estos del valor de $\lambda$.

### 3.3.3 Ejemplo

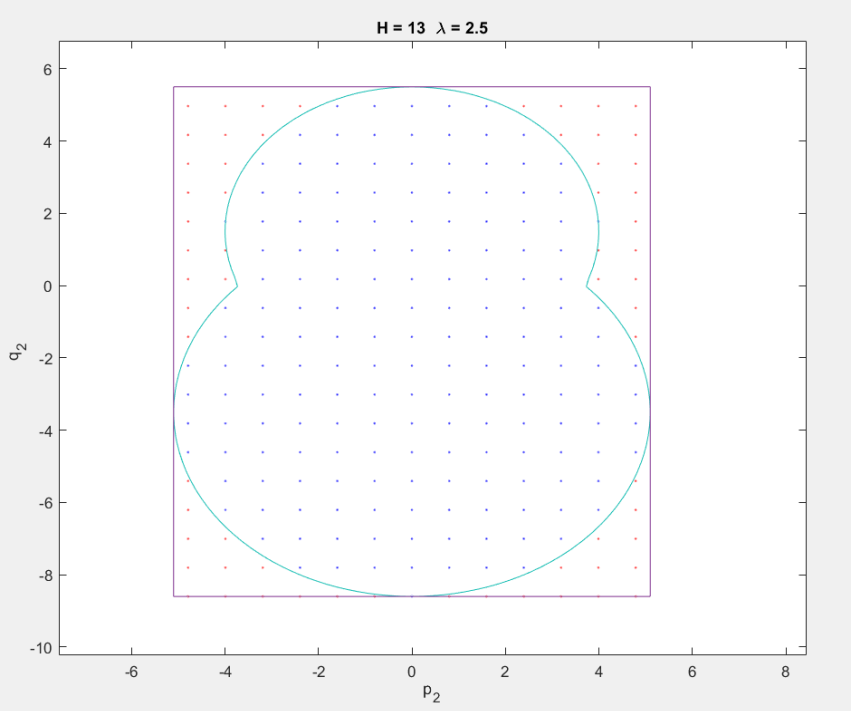


**Figura 3.7** Ejemplo de mallado.

En donde podemos ver la región rectangular en la que se traza el mallado, y con puntos rojos aquellos conjuntos de condiciones iniciales no válidas, mientras que pintados de azul están el conjunto de condiciones iniciales que se le entregaría al integrador.

### 3.3.4 Simetría

Toda dinámica del péndulo-muelle es simétrica en el eje$p_2$ con respecto al eje$q_2$, eso significa que la *sección de Poincaré* descrita será simétrica $\forall H, \forall \lambda$. Esto quiere decir que, a la hora del mallado podemos escoger

solo la mitad negativa (o positiva) de valores de $p_2$. Pues, habiendo un punto que forma parte de una órbita cualquiera, existe su simétrico y forma parte de la misma órbita.

Aprovechando así las propiedades simétricas de la *sección de Poincaré* se reducen a la mitad el conjunto de condiciones iniciales y, por tanto, el coste computacional. Cabe decir que esta reducción implicaría obtener un menor número de soluciones por lo que habría que compensar con un mayor tiempo de simulación.

Otra ventaja de esta propiedad es la capacidad de reducir a la mitad la cantidad de memoria necesaria para almacenar los datos. Si la sección de Poincaré es estrictamente simétrica, a la hora de almacenar los datos también podemos decidir escoger solo la mitad negativa (o positiva) de valores de $p_2$, y a la hora de representar la *sección de Poincaré*, duplicar los datos para rellenar la otra mitad de la sección. Con esto también quedaría resuelto el problema de la rotura de simetría.

#### Rotura de simetría

Hay ciertas órbitas muy concretas que no cumplen con lo anteriormente dicho. En ellas se produce una rotura de simetría, debido a esto, el mallado escogido no representaría su órbita simétrica. Aunque como se puede observar en Figura 3.8, esto se podría utilizar para detectar estas roturas de simetría.

Sobre este tema se entrará más en detalle en el apartado dinámico.

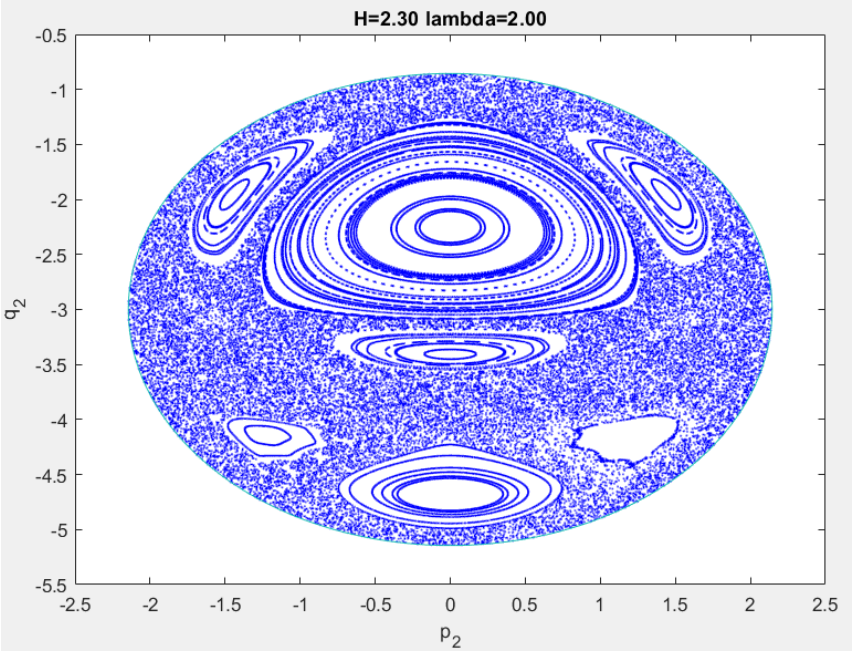


**Figura 3.8** Ejemplo de rotura de simetría.

## 3.4 Paralelización

Durante todo el capítulo se ha hecho hincapié en el coste computacional. Esto es debido a que, dependiendo de la precisión que se desee y del paso en H, la caracterización del sistema para un único valor de $\lambda$ podría tomar semanas incluso en un buen procesador. Por ello, para la realización de los experimentos es fundamental paralelizar el proceso.

Una vez obtenido el vector con las condiciones iniciales para un valor de H, se ejecuta el integrador en bucle para resolver el sistema para cada cuádrupla de valores. Estas iteraciones son independientes las una de las otras haciendo así paralelizable al bucle, es decir, podría haber n-procesadores integrando a la vez. Y, si tenemos en cuenta que la integración asume prácticamente en su totalidad el coste computacional del problema, podríamos decir que el tiempo de ejecución del proceso se reduce n veces, siendo n el número de procesadores ejecutándose en paralelo.

En la Tabla C.1 del Apéndice C encontramos una comparación de los tiempos de simulación entre el proceso paralelizado y sin paralelizar en diferentes máquinas.

En este caso, a diferencia de los proyectos [2] y [1], al realizar esta paralelización en *Matlab*, este paso es trivial. Por lo general, basta con cambiar el bucle *for* por uno de tipo *parfor* (salvo por posibles pequeños cambios extras en variables compartidas, también triviales, en este caso todos los procesadores escriben en la misma variable). Al realizar el *parfor*, Matlab se encarga de forma automática de reservar un *pool de workers* para el usuario en cuestión y de gestionar los procesos generados al paralelizar.

# 4 Dinámica del péndulo-muelle

Una vez creadas las herramientas computacionales necesarias y de obtener los datos, el siguiente paso es el estudio de las *secciones de Poincaré* para poder caracterizar el sistema.

En este capítulo principalmente se estudiará el sistema para un valor concreto de $\lambda$ y, a partir de este, se extrapolará lo aprendido para comprender al sistema con un caracter más general.

## 4.1 Dinámica para $\lambda = 2.9$

Como se indicó previamente en este proyecto, este sistema tiene dos modos normales: péndulo y muelle. A diferencia del muelle, el péndulo no es ármonico simple, su espectro en frecuencia incluye más tonos. Sin embargo, para pequeña amplitud (un ángulo menor a quince grandos), podemos observar que un único tono es relevante haciendo así, bajo estas condiciones, al péndulo un movimiento armónico. Este único tono sería la frecuencia natural de cada uno de los sistemas. Así pues, si la frecuencia de una es múltiplo de la otra e interactúan entre sí estaríamos ante un caso de resonancia y en una zona en la que el traspaso de energía entre ambos modos es mayor. Dando lugar así, a una dinámica que presumiblemente será más rica que a otros valores de $\lambda$ en los que no se produzca dicho efecto.

Las frecuencia del péndulo simple (a pequeña amplitud) y del muelle son:

$$\begin{aligned} w_m &= \sqrt{\frac{k}{m}}, \\ w_p &= \sqrt{\frac{g}{l_p}} \qquad con \quad l_p = l_0 + \frac{mg}{k} \end{aligned}$$

En donde la longitud del resorte cuando el sistema se comporta como un péndulo no es su longitud natural, sino que hay que añadirle la elongación producida por el peso de la masa.

Que adimensionalizando con la escala de tiempo usada en (2.5) queda:

$$\begin{aligned} w_m &= 1, \\ w_p &= \frac{1}{\sqrt{\lambda+1}} \implies \lambda = n^2 - 1 \quad \text{con } \ n \in \mathbb{N}, \end{aligned} \tag{4.1}$$

en donde $n = \frac{w_m}{w_p}$. Y, por tanto, la solución de la ecuación (4.1) son los valores de $\lambda$ para los que una frecuencia es múltiplo de la otra.

En este apartado se estudiará para un caso cercano a $1:2$ ($\lambda = 2.9$), y se estudiará incrementando la energía partiendo desde su mínimo valor.

### 4.1.1 Hill y Valley

A lo largo de los distintos valores de $\lambda$ y a energías no muy grandes, los tipos principales de órbitas que dominan en el sistema son de tipo valley y de tipo hill. Esta terminología, acuñada de [5], se refiere a que la trayectoria descrita se asemeja a un valle y una colina, respectivamente. Es decir, describen una trayectoria con forma de 'U'(valley) y de 'U' invertida (hill).

Como se vio en el apartado de modelado dinámico, el punto de mínima energía (el que corresponde al equilibrio del péndulo colgante y con el muelle estirado para compensar la gravedad), se produce en $H = 0$. Así pues, si escogemos un valor particularmente bajo de $H$, en este caso $H = 0.25$, nos encontramos con que la sección se divide en dos regiones con curvas (órbitas cuasiperiódicas) concéntricas, tal y como podemos observar en la Figura 4.1. Mientras que la región superior corresponde con el tipo hill, la región inferior lo hace con el tipo valley.

Los puntos centrales de ambas regiones corresponden con las órbitas periódicas de sus mismos tipos. Aunque no podamos escoger el punto exacto donde encontramos una órbita periódica de tipo hill/valley, como vemos en las gráficas que conforman las Figura 4.2 y Figura 4.7, tenemos una trayectoria con una única componente frecuencial y un rango de valores de $q_2$ y $p_2$ para $q_1 = 0$ prácticamente nulo; de forma que la diferencia con la órbita periódica sea despreciable.

Conforme nos alejamos de este centro, las frecuencias en $q_1$ y $q_2$ dejan de estar conmensuradas y ganan otras componentes haciendo así, que la solución no se cierre nunca. Lo cual se traduce en un mayor rango de valores de $q_2$ y $p_2$ y, por tanto, un engrosamiento en su trayectoria. Tal y como podemos ver en las Figura 4.3 a Figura 4.11. Se conocen como órbitas cuasiperiódicas.

Estas curvas concéntricas son el resultado de las intersecciones de toros en un espacio tridimensional, con el plano $q_1 = 0$. Estos toros, como era de esperar, son los planos descritos por las parejas de valores $q_2$ y $p_2$ durante una trayectoria.

En la Figura 4.12 están marcadas en rojo las órbitas usadas como ejemplo.

Como se verá en el próximo apartado, cuando tenemos dos regiones diferentes y vecinas en la sección (los límites de una marcan el comienzo de la región del otro sin haber nada más en medio), si escogemos una órbita de cada una de las dos regiones cerca de los límites de estas, por proximidad, las trayectorias que describirían ambas serían similares a pesar de ser de distinto tipo. Así pues, en este caso, con las regiones de tipo hill y valley obtendríamos el resultado de las figuras Figura 4.13 y Figura 4.14.

De esta forma, si nos acercamos aun más al límite se hace indistinguible si nos encontramos ante un caso de tipo hill o tipo valley llegando incluso a encontrar un caso en el que a ciertas condiciones iniciales, la curva que dibuja en la sección parece formar parte de ambas regiones. Sin embargo, aunque la gráfica de la trayectoria parezca idéntica, esta no lo es. Si comparamos la Figura 4.15 y Figura 4.16 (esta última correspondería a una hill), hay una diferencia fundamental: la frecuencia. Mientras que la segunda en la frecuencia de $q_1$ presenta una serie de componentes en frecuencia, la primera es muy ruidosa. Esto significa que el comportamiento de ambas es muy diferente, siendo la primera bastanta caótica.

Esta pequeña región aumentará de relevancia con la energía siendo así la ruta al caos del sistema.

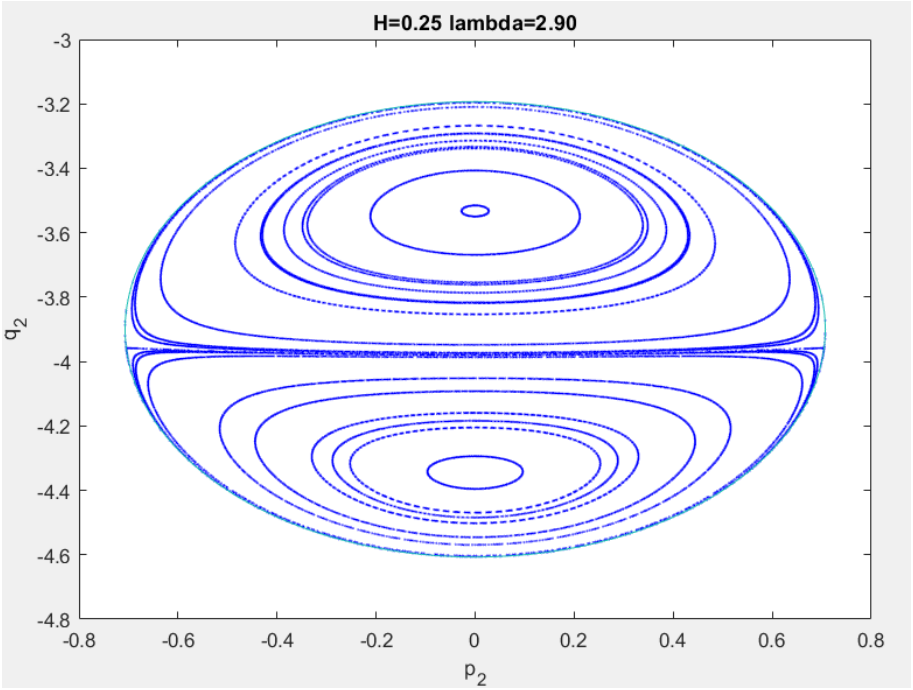


**Figura 4.1** Sección Poincaré $H = 0.25$ $\lambda = 2.90$.

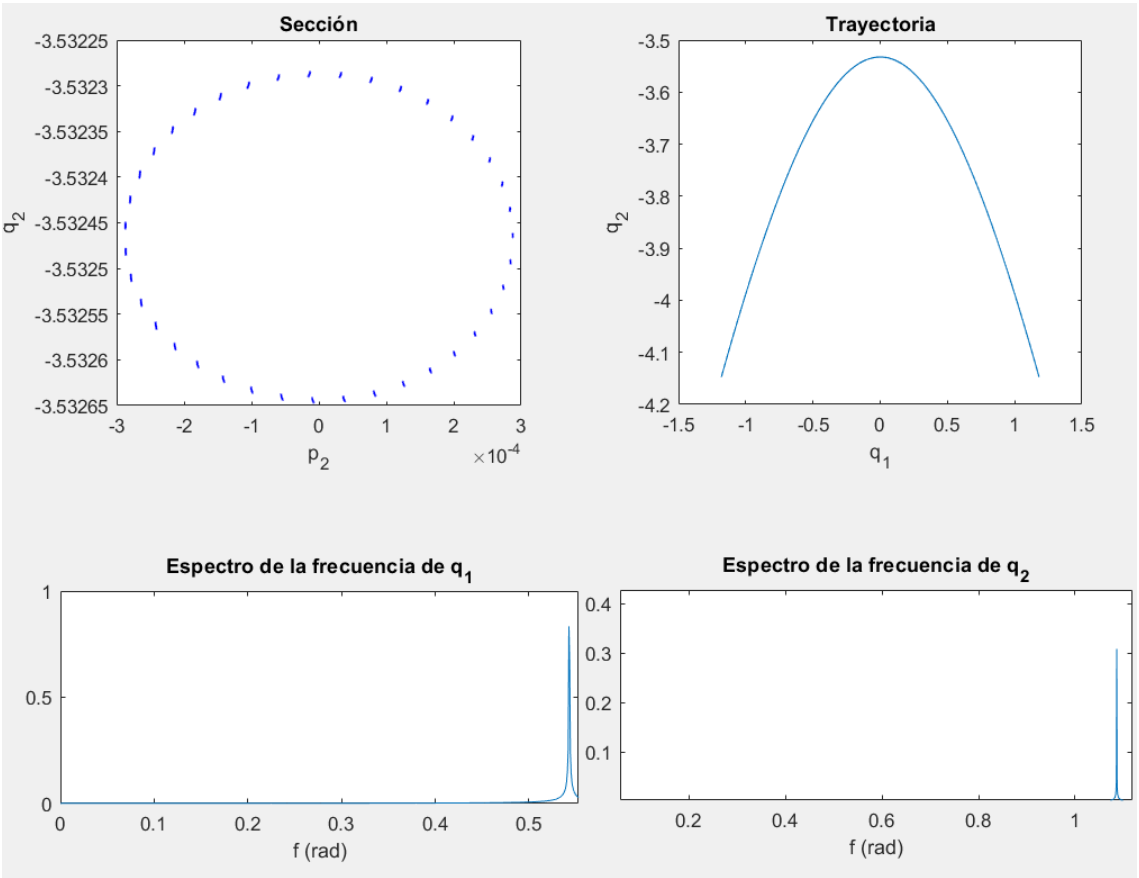


**Figura 4.2** Órbita periódica tipo hill.

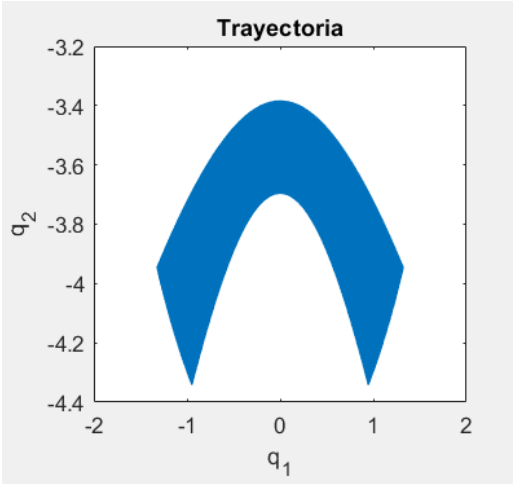


**Figura 4.3** Trayectoria cuasiperiódica tipo hill (1).

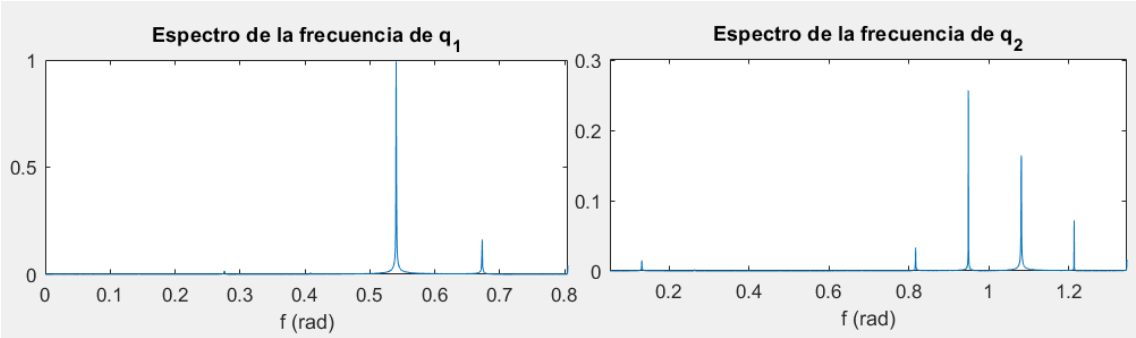


**Figura 4.4** Frecuencia cuasiperiódica tipo hill (1).

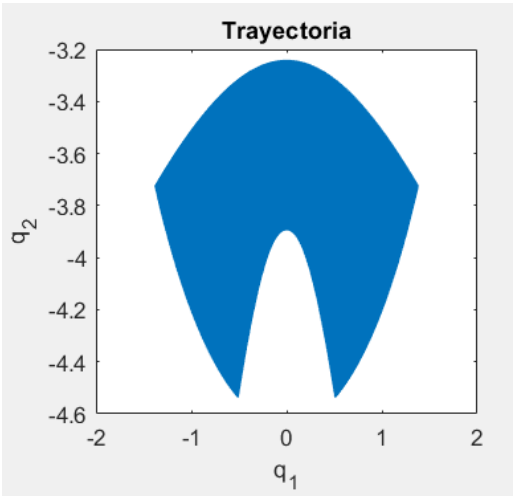


**Figura 4.5** Trayectoria cuasiperiódica tipo hill (2).

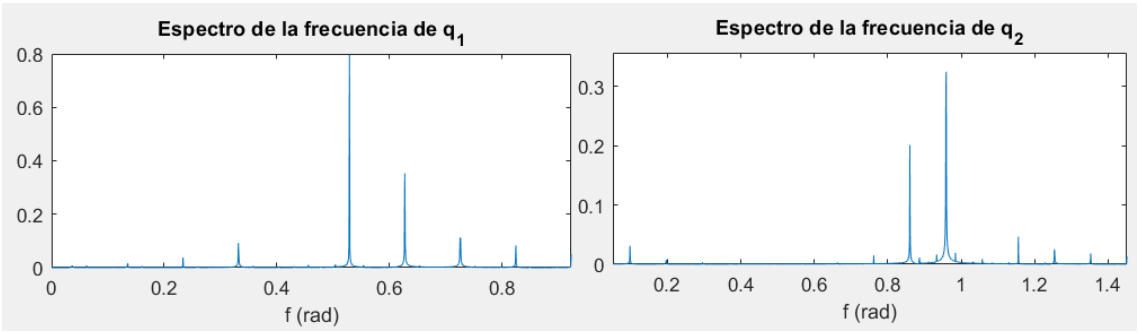


**Figura 4.6** Frecuencia cuasiperiódica tipo hill (2).

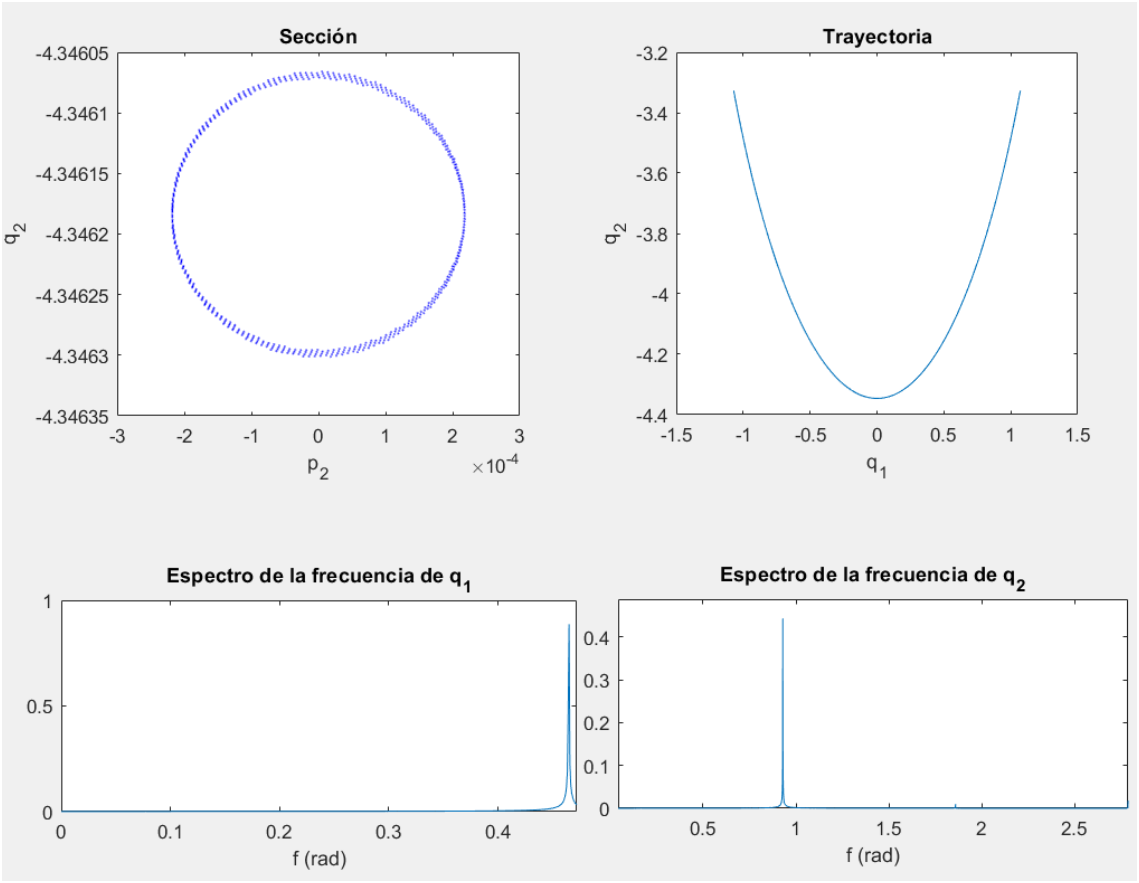


**Figura 4.7** Órbita periódica tipo valley.

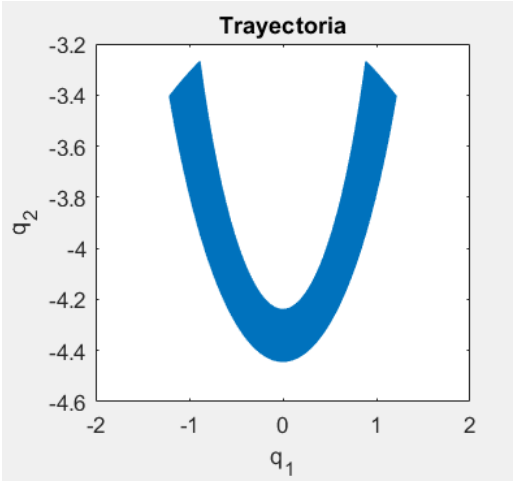


**Figura 4.8** Trayectoria cuasiperiódica tipo valley (1).

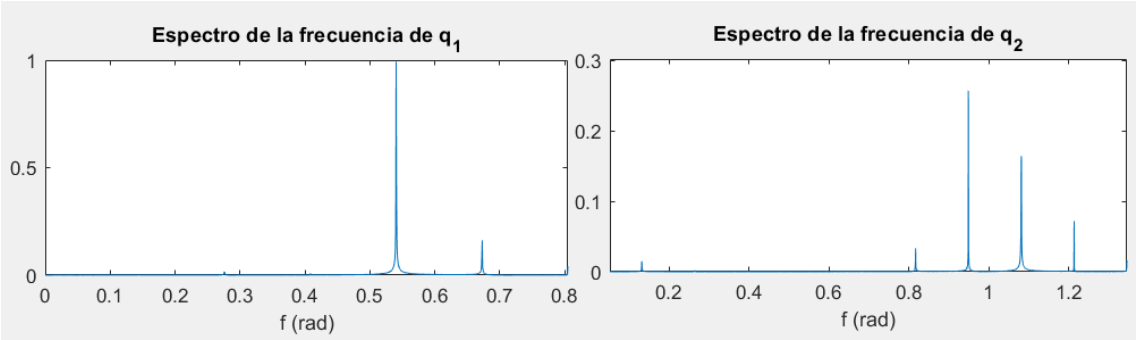


**Figura 4.9** Frecuencia cuasiperiódica tipo valley (1).

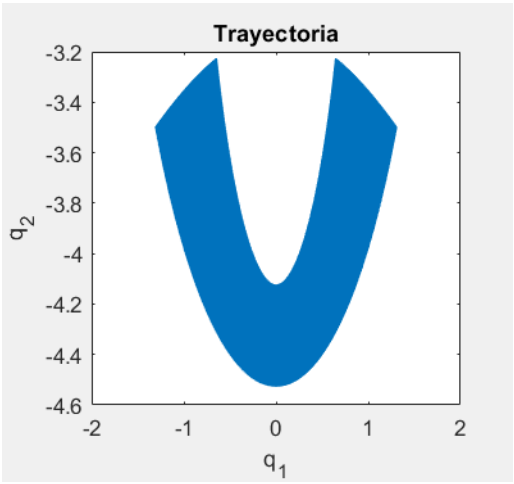


**Figura 4.10** Trayectoria cuasiperiódica tipo valley (2).

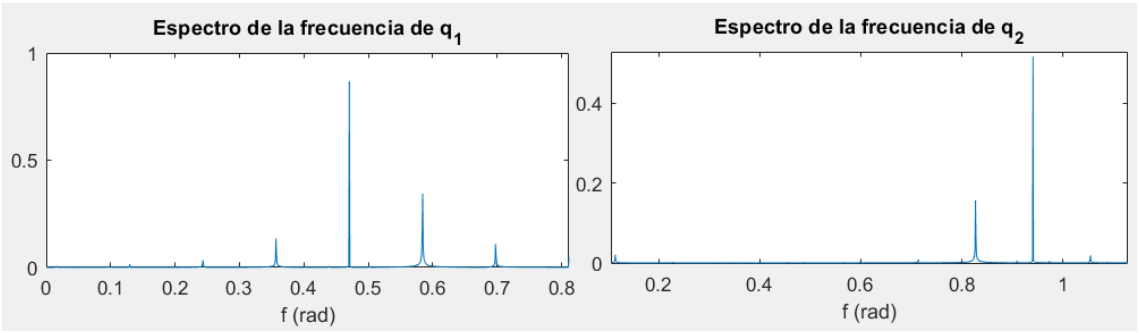


**Figura 4.11** Frecuencia cuasiperiódica tipo valley (2).

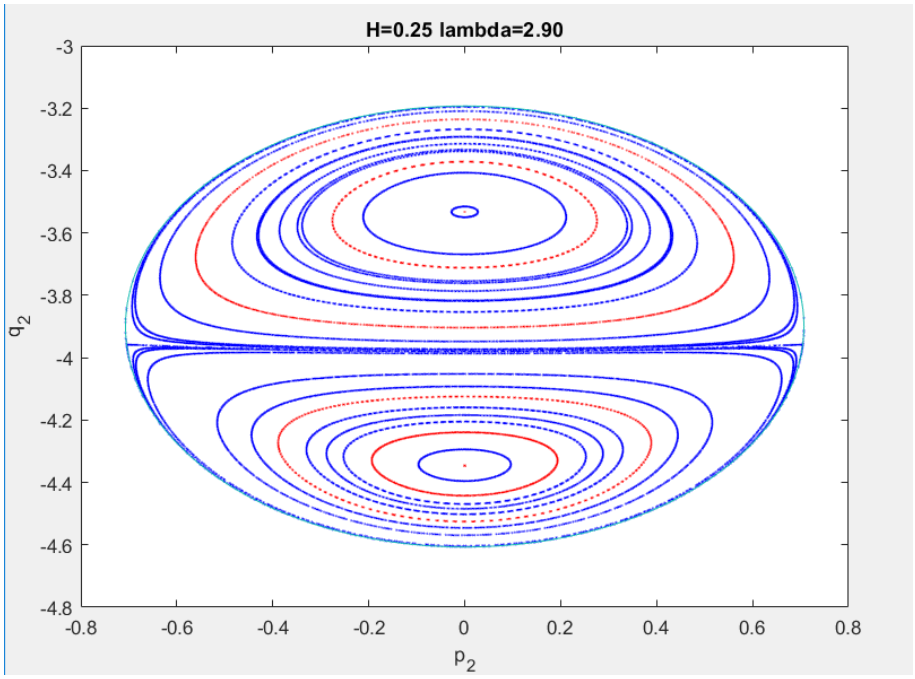


**Figura 4.12** SP órbitas ejemplo.

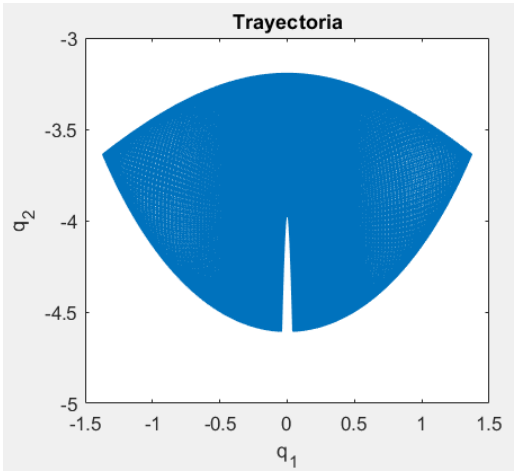


**Figura 4.13** Trayectoria cuasiperiódica tipo hill (4).

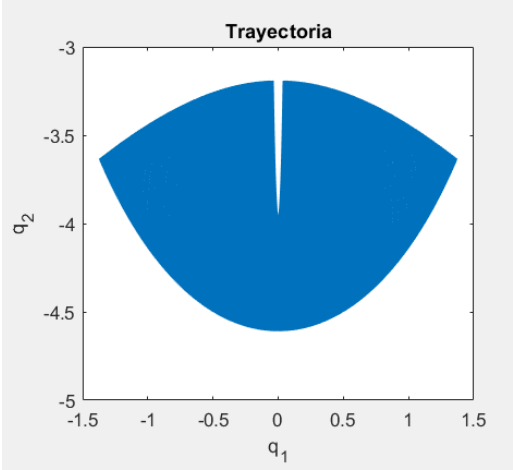


**Figura 4.14** Trayectoria cuasiperiódica tipo valley (4).

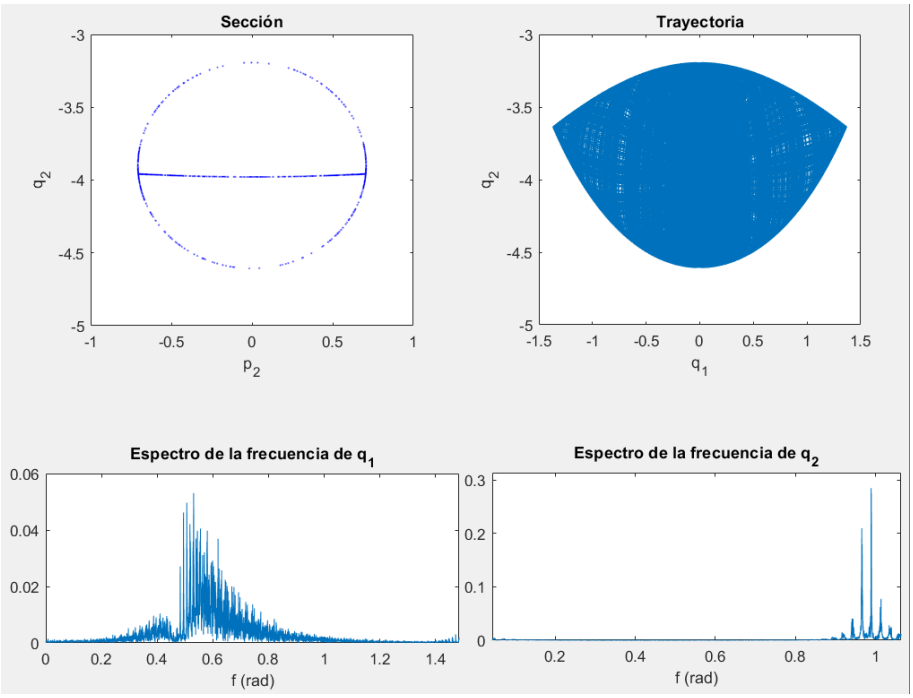


**Figura 4.15** Trayectoria tipo caótica.

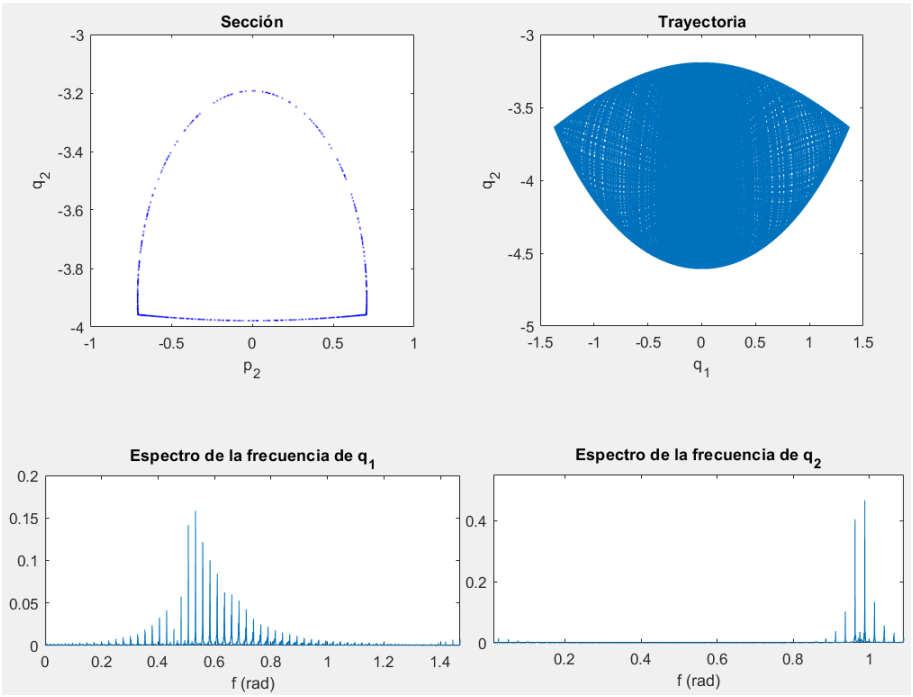


**Figura 4.16** Cuasiperiódica tipo hill (5).

### 4.1.2 Bifurcaciones subarmónicas

Conforme aumentamos la energía, aparecen trayectorias diferentes, pero que no dejan de ser tipo valley y hill. Para $H = 0.85$ nos encontramos con dos más.

En la Figura 4.18 podemos ver una dinámica cuya trayectoria es asimétrica con respecto al eje$q_1$. Como la sección de Poincaré es simétrica, tiene que existir otra en versión espejo de la primera (Figura 4.19). La región correspondiente a una de estas está marcada en Figura 4.17; y, forma parte de la región de tipo valley.

En la región tipo hill hay una zona que en vez de haber una única curva hay un grupo de seis. Esto se conoce como una bifurcación subarmónica, n-periódica (siendo n el número de curvas). El resultado se puede ver en la Figura 4.20, en donde podemos observar como la trayectoria pasa en seis puntos por $q_1 = 0$ lo cual, evidentemente, coincide con el número de 'islas'.

Sin embargo, si pasamos al integrador una de estas islas como condición inicial solo aparecerían la mitad de ellas en la sección (como está indicado en la Figura 4.21). Esto se debe a dos razones: como se indicó anteriormente, solo produce un evento al paso por $q_1 = 0$ en un sentido; y, a que la curva que define la trayectoria es cerrada. Esto último se debe a que si la trayectoria fuera abierta, ambos extremos de esta serían puntos de parada, es decir, la velocidad se anula en ellos permitiendo así que se invierta el sentido en el que la recorre. Sin embargo, en las curvas cerradas esto no sucede, de forma que, aunque se pueda recorrer en ambos sentidos, una vez iniciada la trayectoria solo lo hace en uno cortando n veces en el $q_1 = 0$ pero tan solo en la mitad lo hace en sentido positivo.

Debido a que estas trayectorias son simétricas con respecto al eje $q_1$, si el número de veces que pasan por dicho eje es par, la curva que describa la trayectoria será cerrada; si es impar, la única posibilidad es que sea abierta. Así pues, como se verá más adelante, tan solo con observar la *sección de Poincaré* se puede más o menos entender con qué tipo de órbitas nos encontramos.

Esta última, como se ha indicado en el principio, no deja de ser una dinámica de tipo hill. Al fin y al cabo, es el resultado de una bifurcación armónica de esta. Para que quede más claro se ha simulado con un par de órbitas muy cercanas: una con multiplicación de periodo y otra sin ello. El resultado se puede comparar en la Figura 4.22 y en la Figura 4.23.

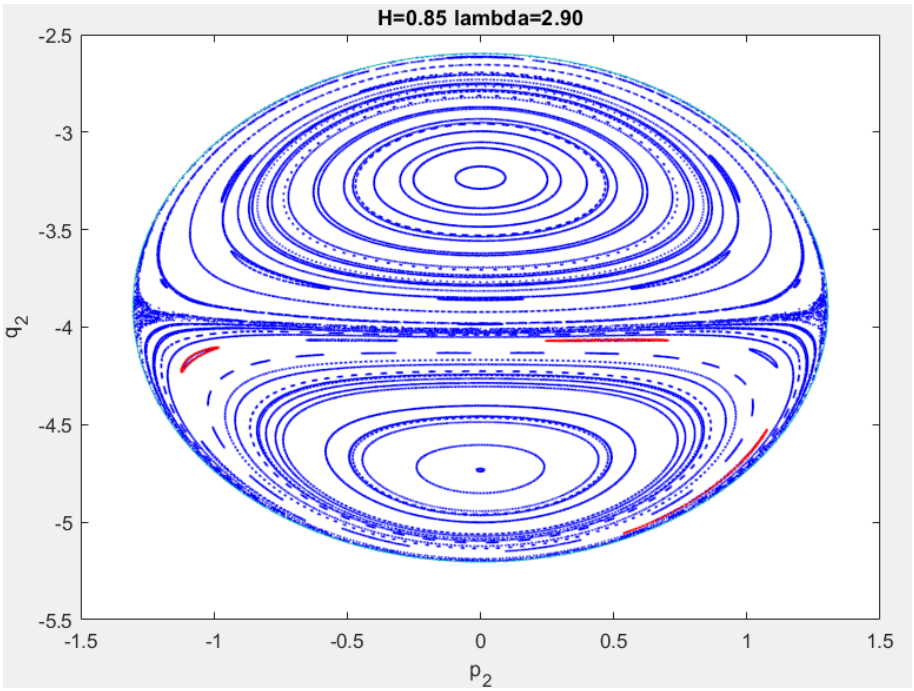


**Figura 4.17** SP asimetría tipo A.

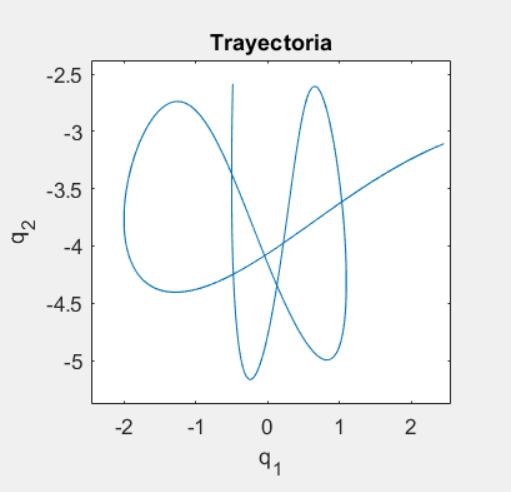


**Figura 4.18** Asimetría tipo A (1).

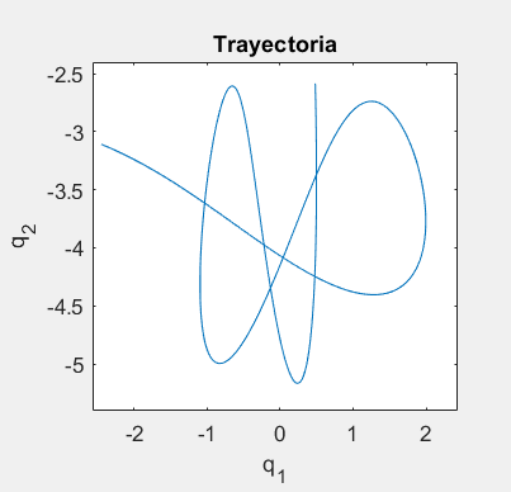


**Figura 4.19** Asimetría tipo A (2).

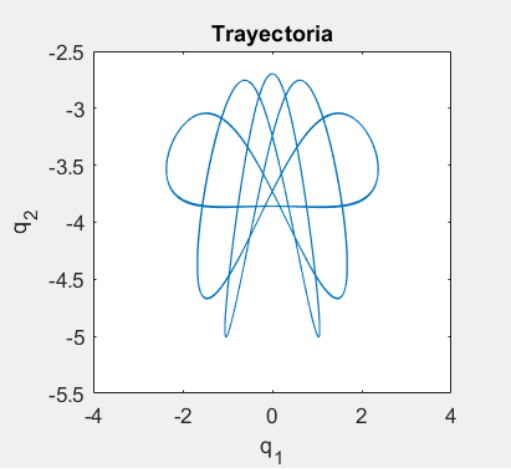


**Figura 4.20** 6-periódico tipo hill.

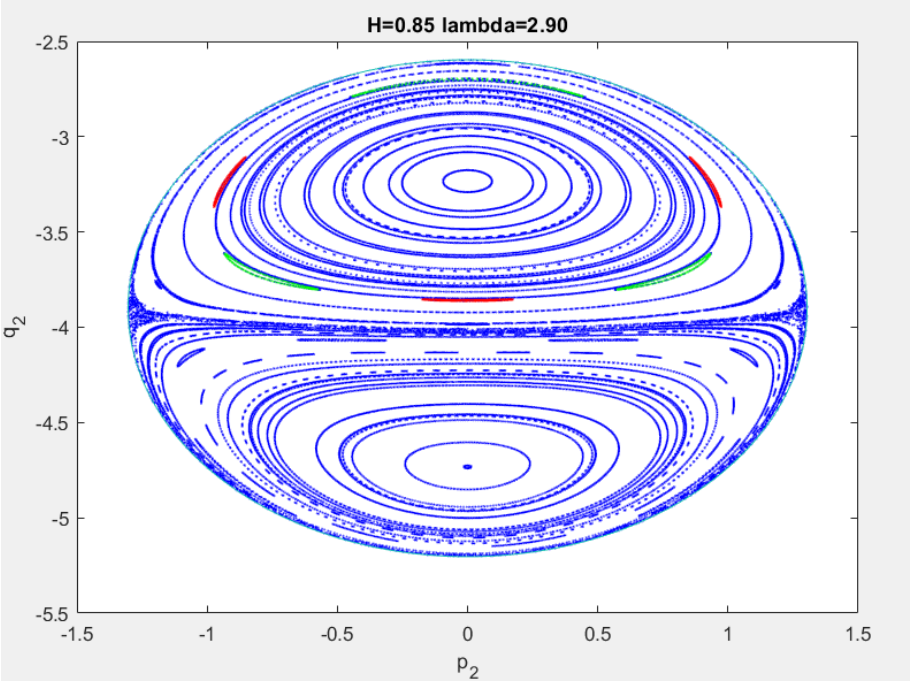


**Figura 4.21** SP 6-periódico.

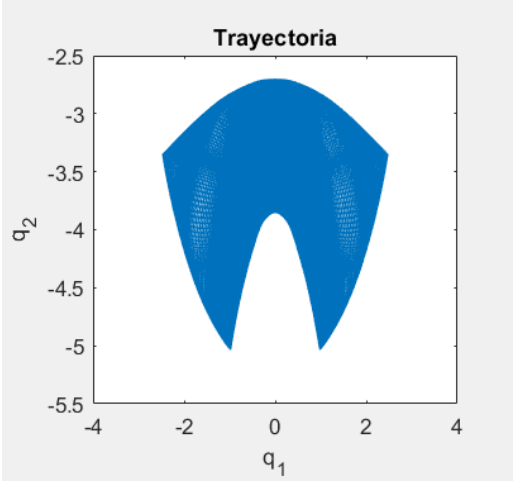


**Figura 4.22** Comparación hill-subarmónica (hill).

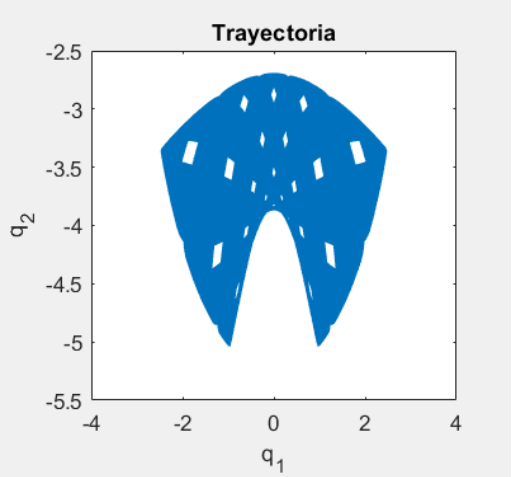


**Figura 4.23** Comparación hill-subarmónica (subarmónica).

### 4.1.3 Bifurcaciones subarmónicas en cascada

Para valores moderados de energía (entorno a $0.8 < H < 4.5$) el sistema es muy rico en dinámica. Estas nacen y, por lo general, conforme aumenta $H$ se van alejando de la órbita periódica de tipo hill/valley y volviéndose cada vez más pequeñas hasta desaparecer. Para poder entender mejor esta evolución del sistema, una buena opción es, con un barrido en $H$, observar cómo varían las distintas secciones.

Para $H = 1.5$ (Figura 4.24), en la región tipo hill nos encontramos ante dos órbitas subarmónicas: una de cuatro, y otra de cinco. Y, como se vio en el apartado anterior, la primera daría lugar a una trayectoria cerrada (Figura 4.25), mientras que la segunda a una abierta (Figura 4.26).

Justo en los límites de esta región se encuentra la bifurcación subarmónica del apartado anterior, sobre la cual, podemos encontrar que se ha producido otra bifurcación subarmónica dando lugar a las Figura 4.28. Esto se conoce como bifurcaciones subarmónicas en cascada. Si comparamos esta con aquella en la que no se ha producido una segunda bifurcación (Figura 4.27), se puede apreciar que la diferencia es que la trayectoria se realiza en seis trazos. Cada isla se ha partido en seis, eso significa que por cada vez que hiciera un paso por cero, lo realizará seis veces. Este fenómeno también se irá repitiendo a lo largo de todo el sistema.

Esto último también sucede en la zona de tipo valley (Figura 4.31 - Figura 4.32), en la que si comparamos sus frecuencias, se aprecia que tal y como indica su nombre, una bifurcación subarmónica tiene como consecuencia una trayectoria con las mismas frecuencias principales pero con una serie de armónicos extra.

También nos encontramos ante una nueva asimetría (Figura 4.29), la del apartado anterior con una bifurcación 2-periódica (Figura 4.30), y una órbita 3-periódica (Figura 4.33).

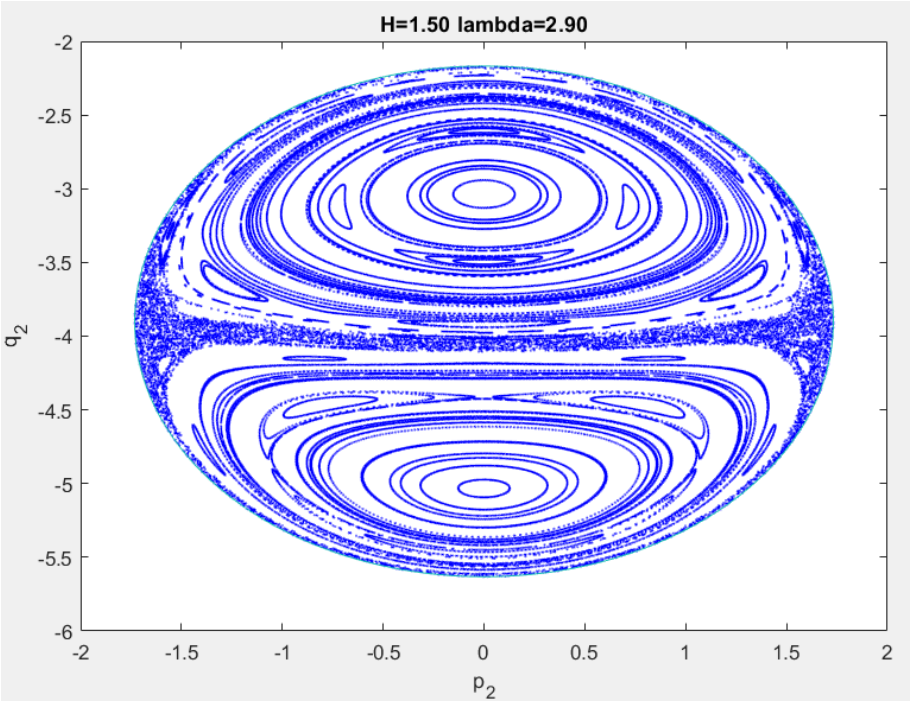


**Figura 4.24** Sección para $H = 1.50$.

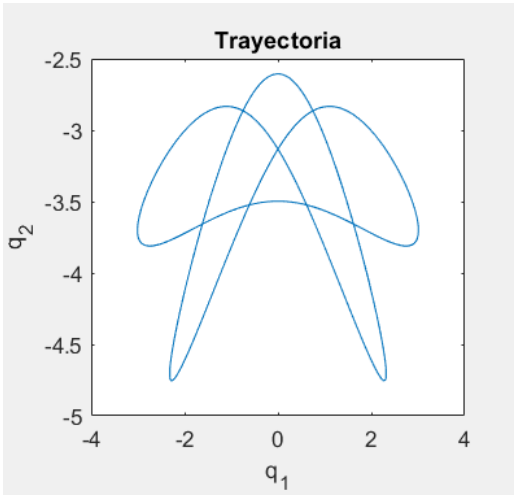


**Figura 4.25** 4-periódico tipo hill.

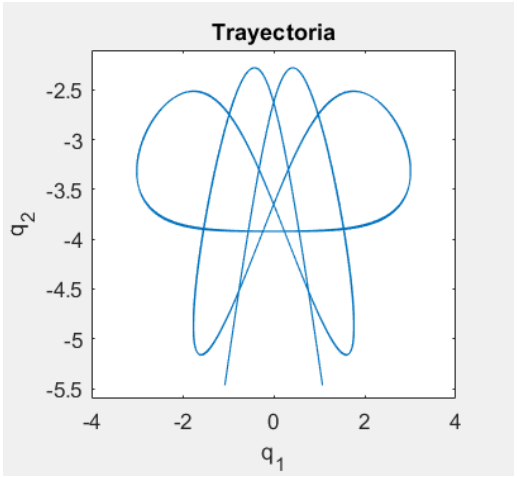


**Figura 4.26** 5-periódico tipo hill.

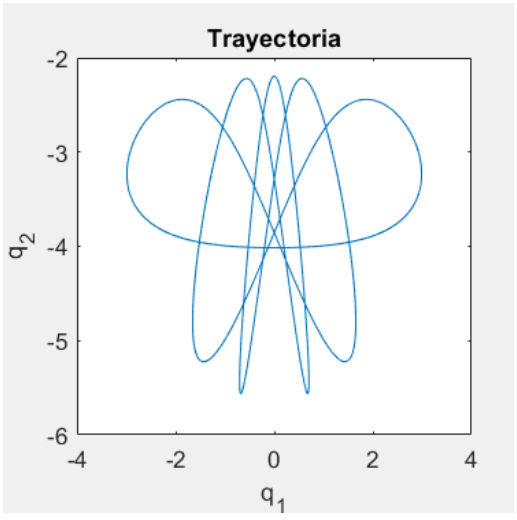


**Figura 4.27** 6-periódico tipo hill.

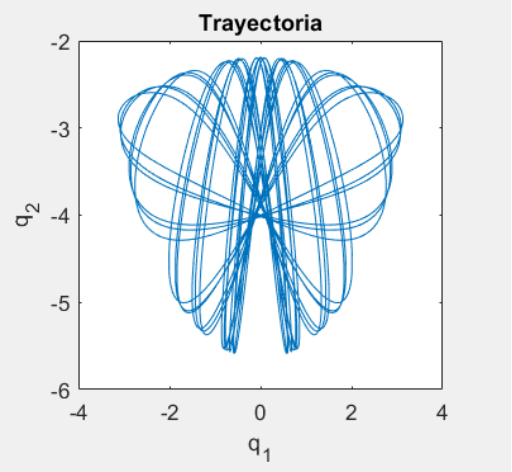


**Figura 4.28** 6-6-periódico tipo hill (bifurcación subarmónica en cascada).

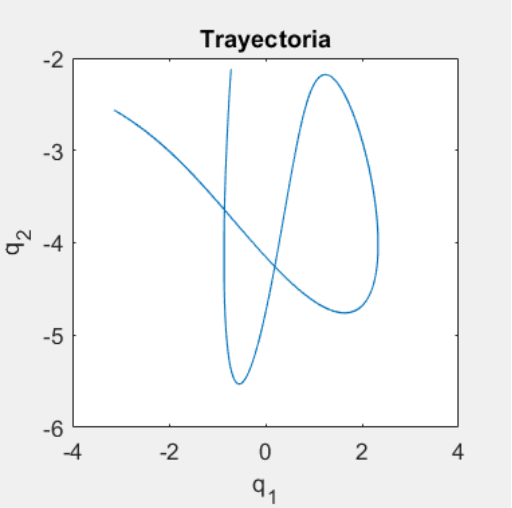


**Figura 4.29** Asimetría tipo B.

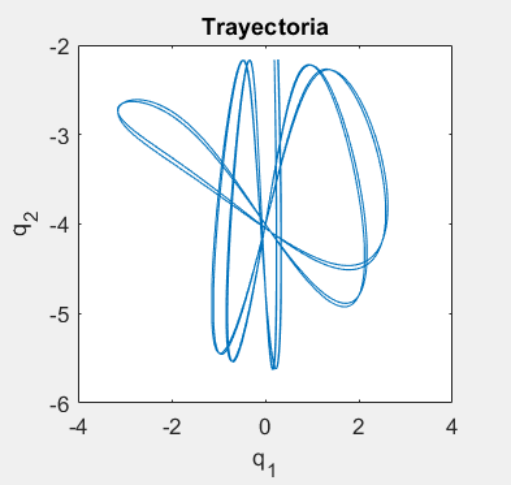


**Figura 4.30** 2-2-periódico asimetría tipo A.

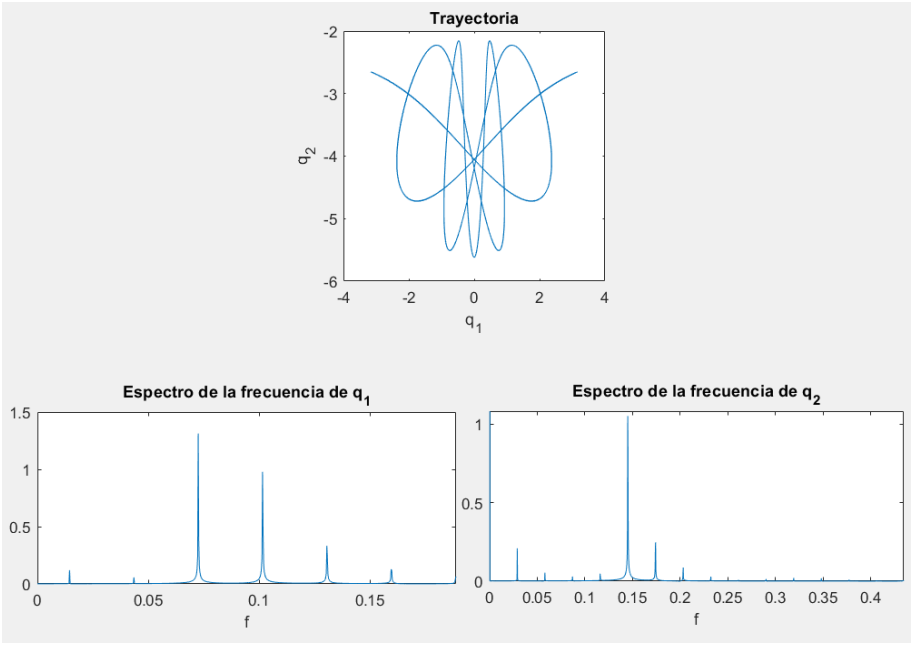


**Figura 4.31** 5-periódico tipo valley.

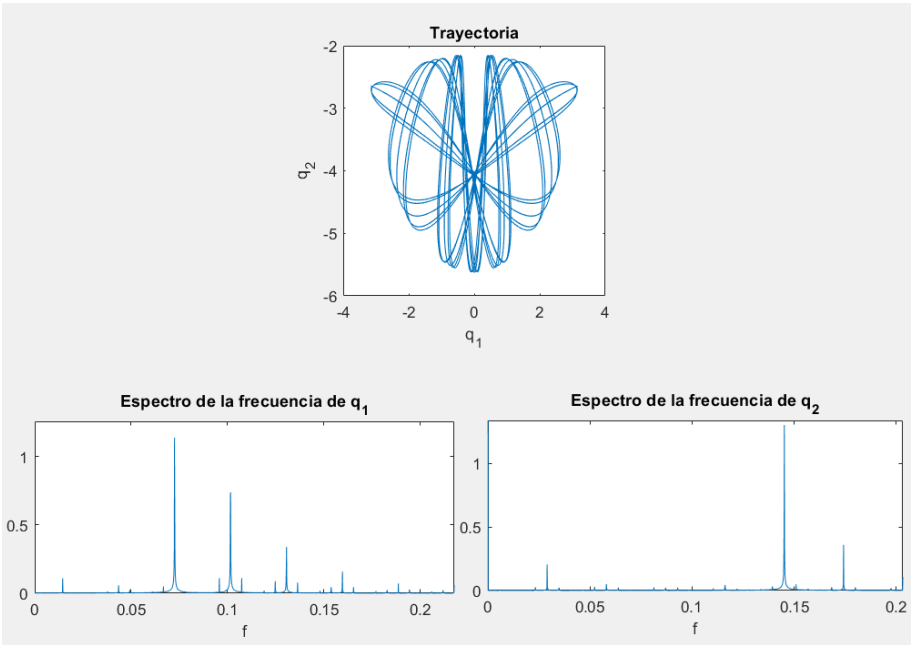


**Figura 4.32** 5-5-periódico tipo valley (bifurcación subarmónica en cascada).

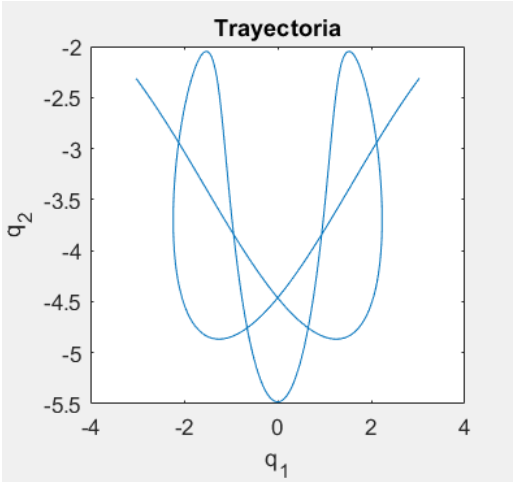


**Figura 4.33** 3-periódico tipo valley.

### 4.1.4 Rotura de simetría

Entre $H = 1.92$ y $H = 1.93$ se produce una rotura de simetría en la región de tipo valley (entorno a $q_2 = -5.25$). Esta en concreto se puede apreciar claramente cómo nace y evoluciona con $H$.

Cuando se produce dicha rotura, la dinámica (de tipo valley) se bifurca en tres ramas: dos estables las cuales son las asimetrías (Figura 4.36); y una rama inestable en medio que es simétrica (Figura 4.35), esta se puede ver señalada en la sección de la Figura 4.34.

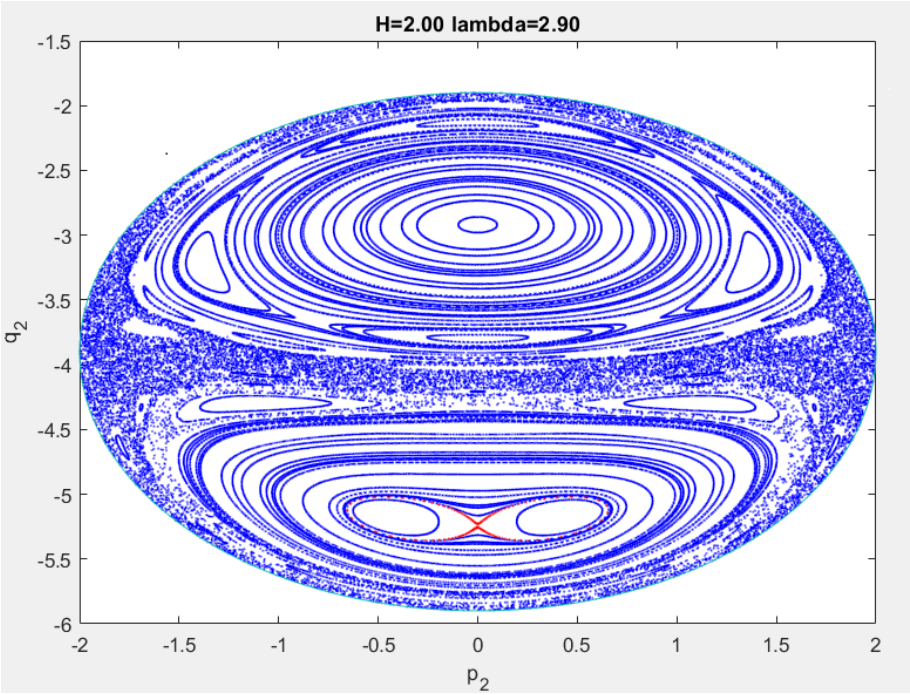


**Figura 4.34** Sección Poincaré BP.

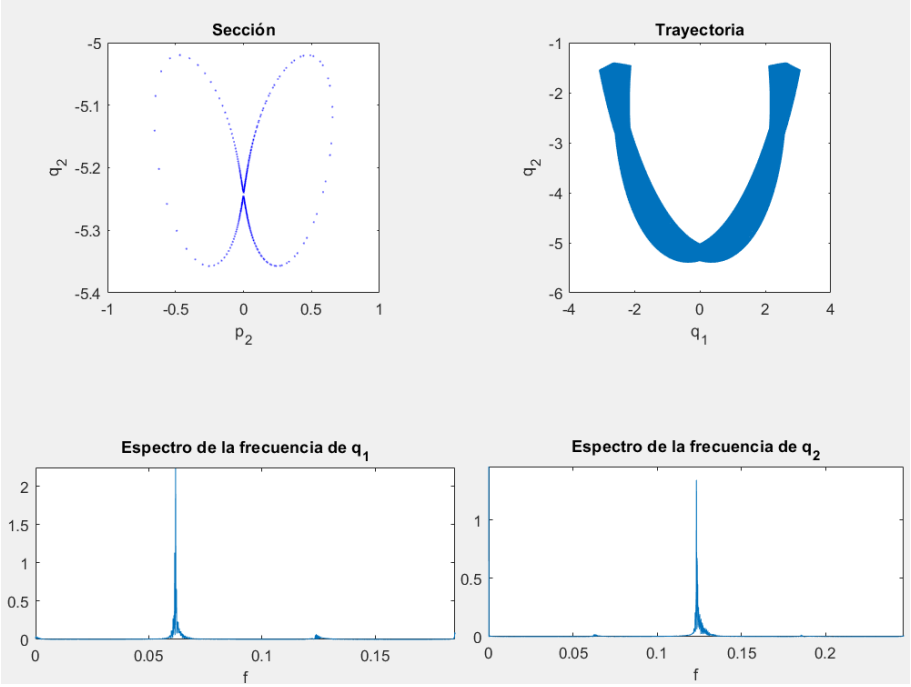


**Figura 4.35** Rama simétrica BP.

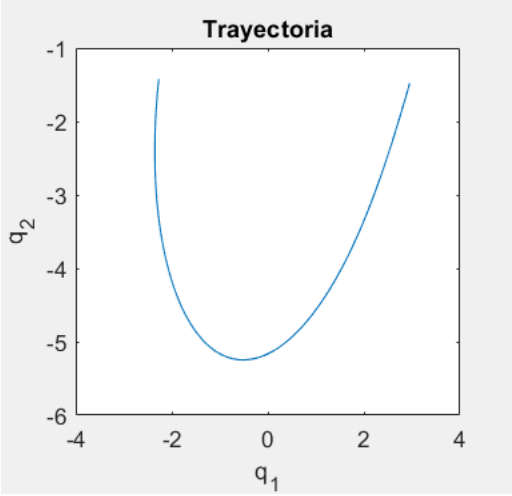


**Figura 4.36** Asimetría tipo C.

### 4.1.5 Evolución de las órbitas con $H$

Una vez visto todo lo anterior es posible entender casi a su totalidad las diferentes *secciones de Poincaré* hasta aproximadamente el valor $H = 2\lambda$. Pues, como se vio en la ecuación (3.9), es a partir de ese valor de $H$ que el sistema ya tiene suficiente energía como para obtener valores positivos de $q_2$ y pasar por $q_1 = 0$. Y, por tanto, que aparezcan nuevos tipos de dinámicas y comportamientos.

Conforme aumenta la energía aumenta el rango de valores de las variables. Y, junto a ello el de las distintas secciones, en las cuales cada vez domina más la región cuyo comportamiento es caótico en detrimento de las de tipo hill y valley.

Las diferentes órbitas subarmónicas, y la asimetría evolucionan como se explicó anteriormente dando como resultado las secciones de las Figura 4.37-Figura 4.42:

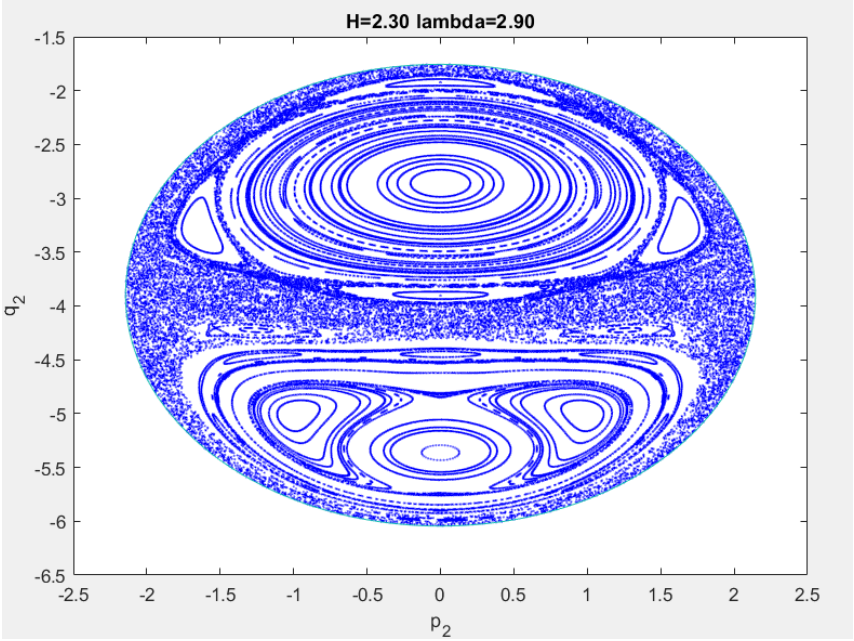


**Figura 4.37** SP $H = 2.30$.

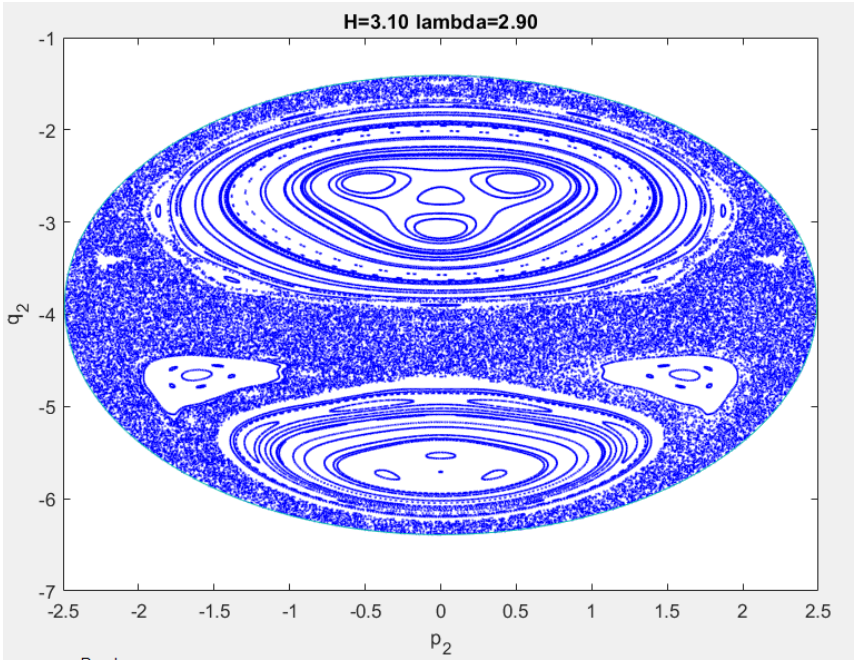


**Figura 4.38** SP $H = 3.10$.

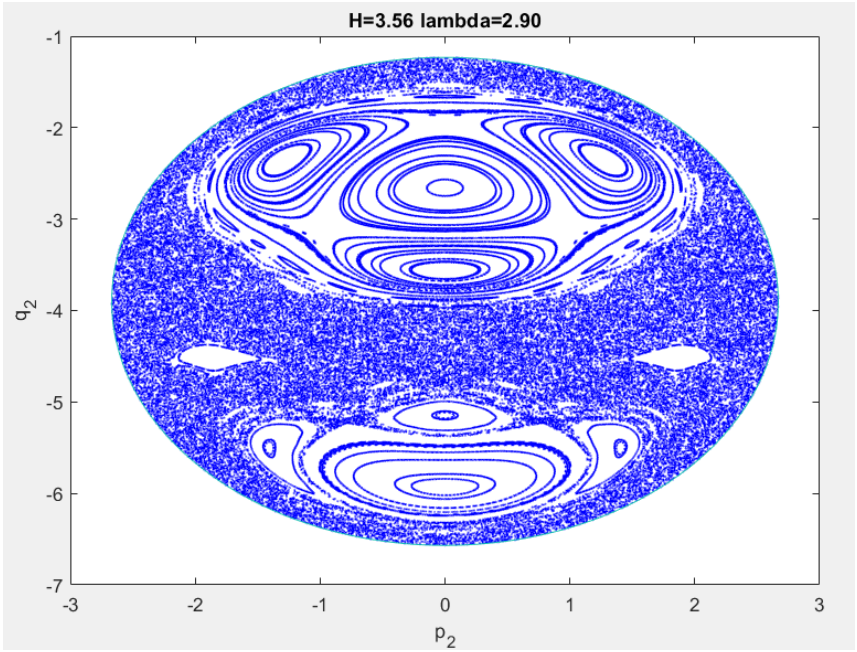


**Figura 4.39** SP $H = 3.56$.

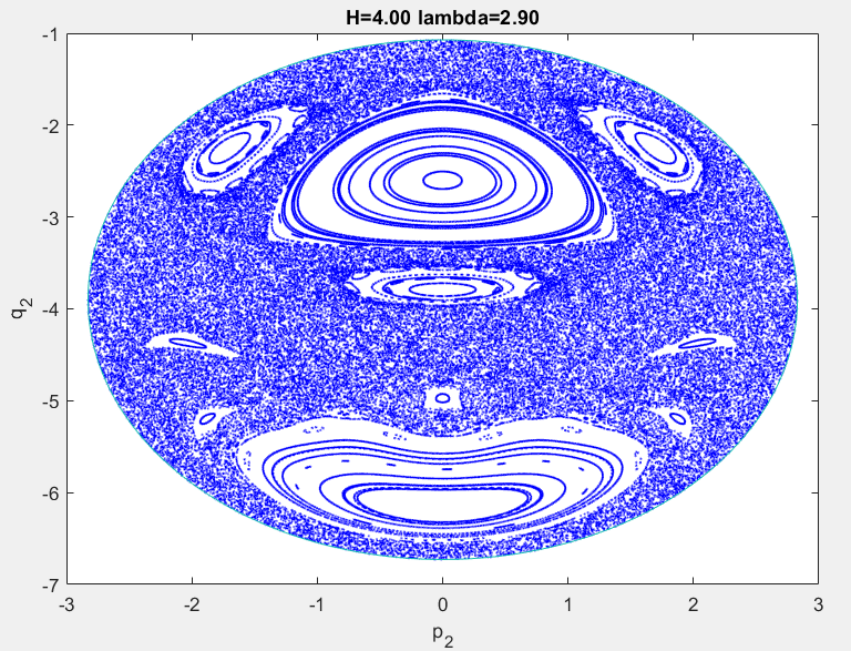


**Figura 4.40** SP $H = 4.00$.

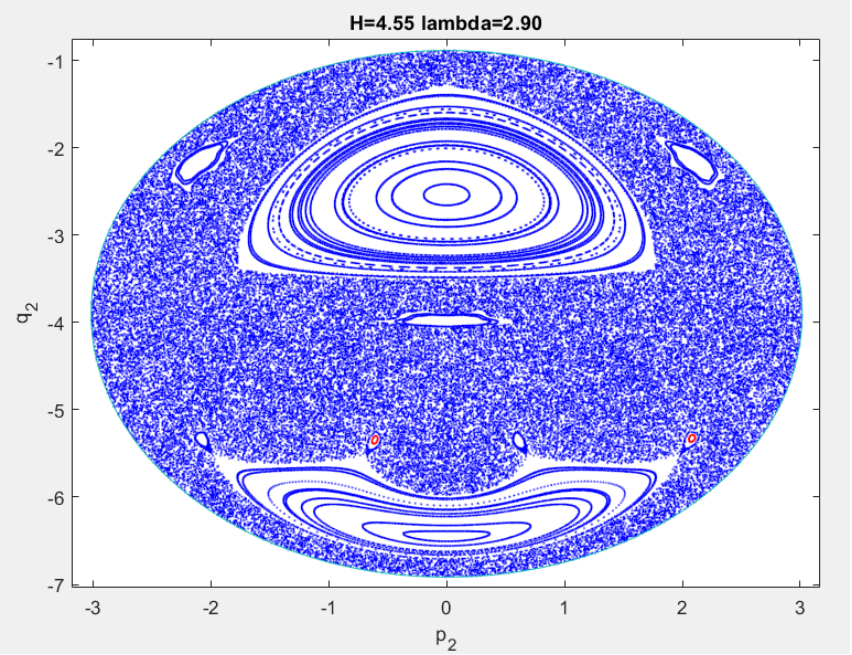


**Figura 4.41** SP $H = 4.55$.

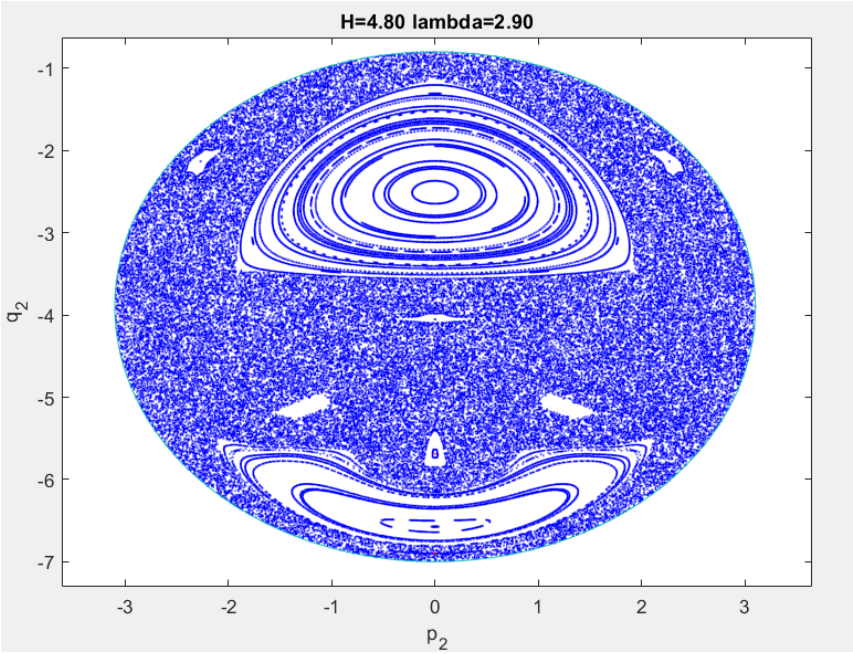


**Figura 4.42** SP $H = 4.80$.

En donde cabe destacar:

- Como es de esperar, conforme la asimetría se aleja de la región de tipo valley, esta disparidad se va acrecentando. Como se puede apreciar en la Figura 4.43 en contraposición de la Figura 4.36
- En la sección de la Figura 4.41, en rojo está señalado una nueva asimetría cuya presencia es efímera y cuya trayectoria viene representada por la Figura 4.44.

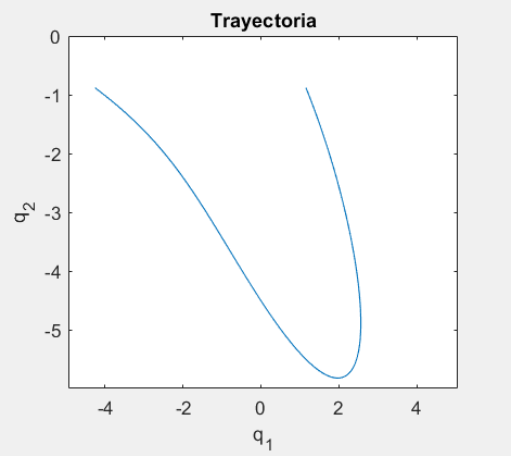


**Figura 4.43** Asimetría tipo C ($H = 3.56$).

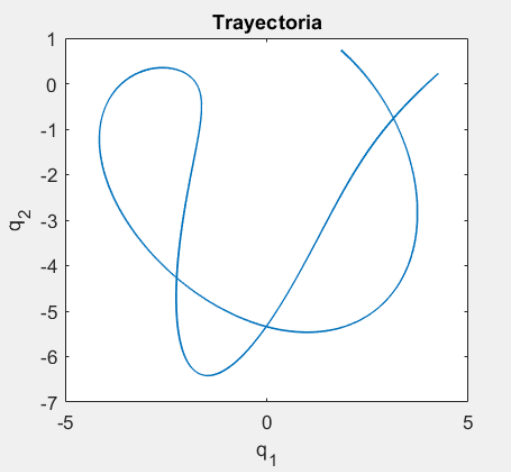


**Figura 4.44** Asimetría tipo D ($H = 4.55$).

### 4.1.6 Dinámicas para $q_2 > 0$

Un poco antes de que el sistema tenga la energía suficiente como para alcanzar valores positivos de $q_2$ con $q_1 = 0$ podemos encontrar un nuevo tipo de trayectoria con forma de herradura (llamado horseshoe por [5]), tal y como se puede ver en la Figura 4.46. Este tipo de dinámica aunque se pueda parecer a la de tipo hill difiere en gran medida de esta; tal y como se puede observar en la figura, presenta un comportamiento oscilatorio que tiende a cerrar la curva, solo que no posee energía suficiente para hacerlo.

Dicho esto, al aumentar la energía del sistema, este tipo de dinámica desaparece dando lugar a una cuya trayectoria descrita es como la anterior salvo la diferencia de que la curva consigue cerrarse. Esto significa pues, que cortará dos veces con el eje $q_1$ lo cual se traduce en:

- Los toros del espacio 3D dibujarán dos curvas en la *sección de Poincaré*
- A diferencia de la anterior, como es una curva cerrada, el movimiento que presenta no será oscilatorio, sino de rotación, y este, evidentemente, se podrá recorrer en sentido horario o antihorario, pero no ambos sentidos en una misma trayectoria. Y, por tanto, como se vio en apartados anteriores, es una curva cerrada y ambos sentidos de rotación se identifican por separado. Este efecto se puede apreciar en Figura 4.47.

Este tipo de trayectoria lo podemos ver en la Figura 4.48.

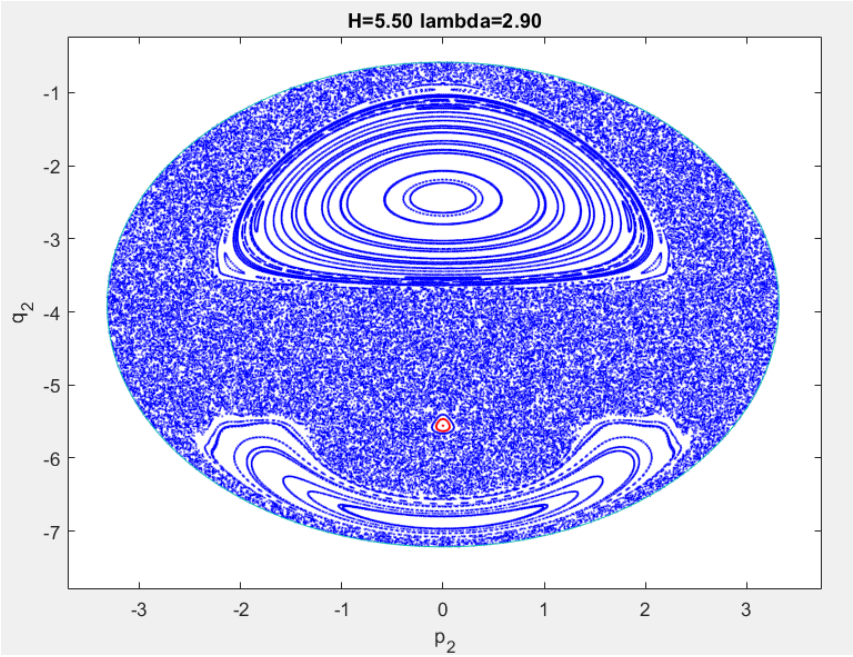


**Figura 4.45** SP $H = 5.50$.

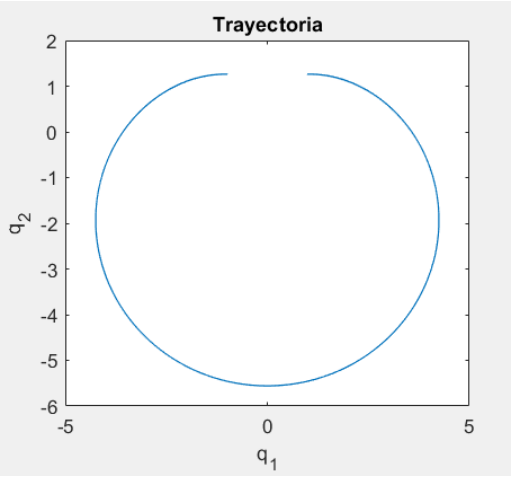


**Figura 4.46** Trayectoria tipo aro.

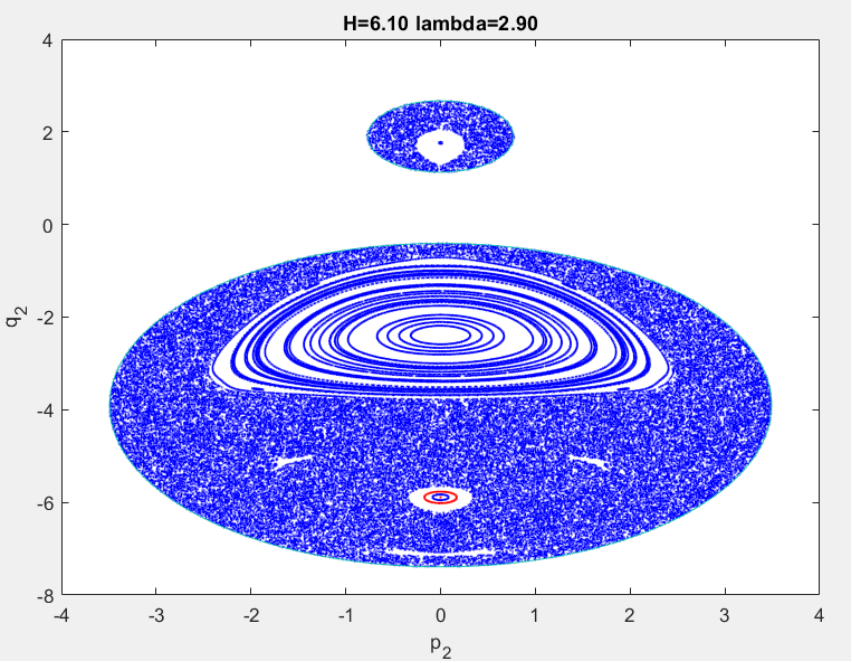


**Figura 4.47** SP $H = 6.10$.

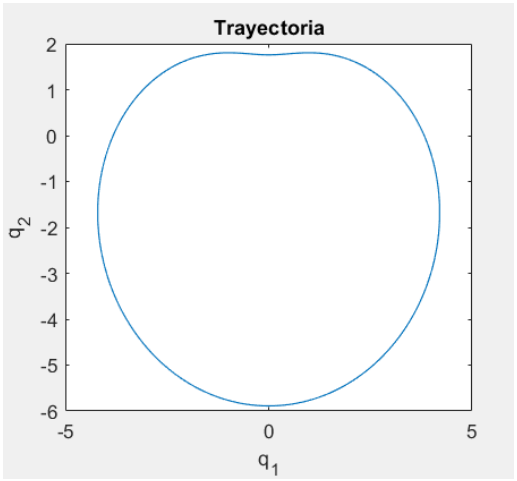


**Figura 4.48** Trayectoria tipo circular.

Para $H = 8$ (Figura 4.49), si la comparamos con la sección anterior, se pueden apreciar una serie de cambios, las dos regiones de condiciones iniciales válidas se han unido y, encontramos cuatro tipos de trayectorias:

- La región de tipo hill.
- La región de tipo valley que antes de desaparecer vuelve a aumentar su tamaño.
- La trayectoria de tipo rotación aumenta considerablemente su tamaño.
- Hay unas cinco 'islas'que surgen de la zona caótica y parece un tipo de trayectoria subármonica dando lugar a la Figura 4.50.

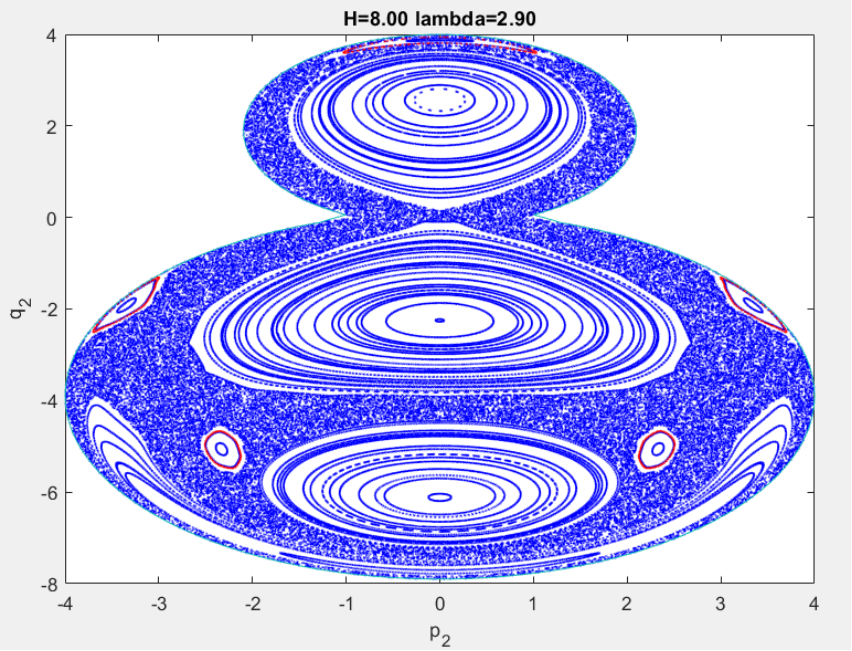


**Figura 4.49** SP $H = 8$.

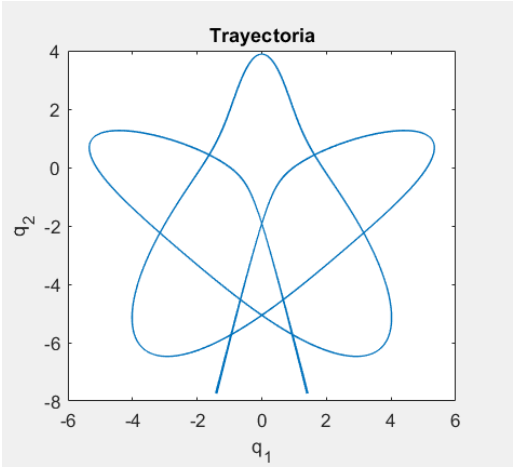


**Figura 4.50** Trayectoria en forma de estrella para $H = 8$.

A partir de aquí el comportamiento del sistema se vuelve más complejo:

- Aparece de la zona caótica otra trayectoria del último tipo entorno a $H = 9$.
- Entorno a $H = 10.5$ aparecen también dentro de la región de tipo caótico una asimetría y una subármonica de la Figura 4.50, las cuales se van fusionando con esta (Figura 4.52-Figura 4.53).
- Este proceso también sucede en la región de tipo rotación. Partiendo desde $H = 8.7$ aparece una 6-armónica de esta en la región exterior y, conforme aumenta la energía, esta se adentra hacia la solución periódica para posteriormente desaparecer. Siendo esto lo contrario a lo que sucedía con las regiones tipo hill y valley.
- Aun así, este tipo de soluciones, entorno a $H = 10$ parecen a empezar a surgir solo en las zonas externas a la región de tipo hill y rotación. De hecho, si nos fijamos con detalle, estas órbitas subarmónicas parecen una mezcla de entre los dos tipos anteriores por un par de razones. Primero, rodean a ambas regiones. Segundo, para $H = 10.5$ encontramos las trayectorias de la Figura 4.54 y la Figura 4.55, una 11-periódica y otra 8-periódica casi al exterior sus regiones, hill y de rotación, respectivamente; si miramos para $H = 10.75$ nos encontramos una que parece venir de las anteriores dando lugar a la trayectoria de la Figura 4.56, la cual, si la comparamos con las anteriores, esta parece una mezcla de ambas. Como ejemplo, la parte de la trayectoria que parece de tipo hill, se mueve en el mismo rango de valores que la 11-periódica de este mismo tipo.

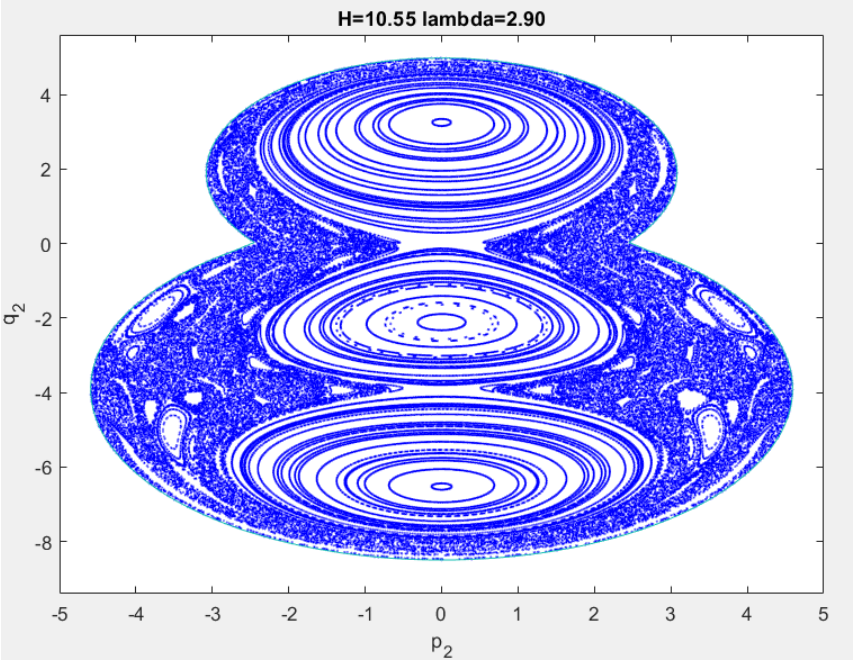


**Figura 4.51** SP $H = 10.55$.

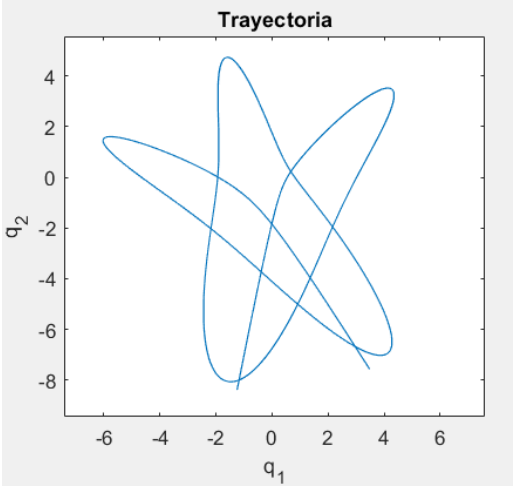


**Figura 4.52** Trayectoria con forma de estrella asimétrica para $H = 9$.

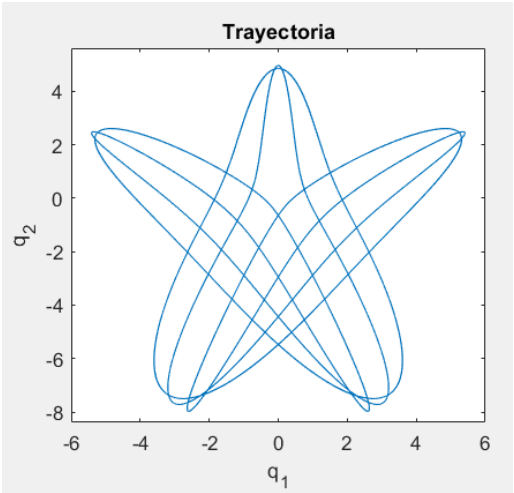


**Figura 4.53** Trayectoria con forma de estrella simétrica para $H = 9$.

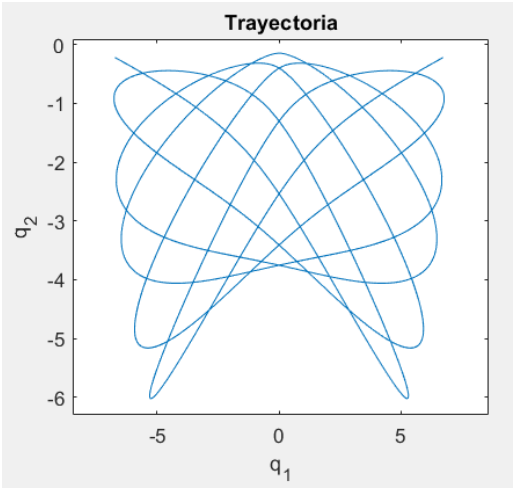


**Figura 4.54** 11-periódica tipo hill para $H = 10.5$.

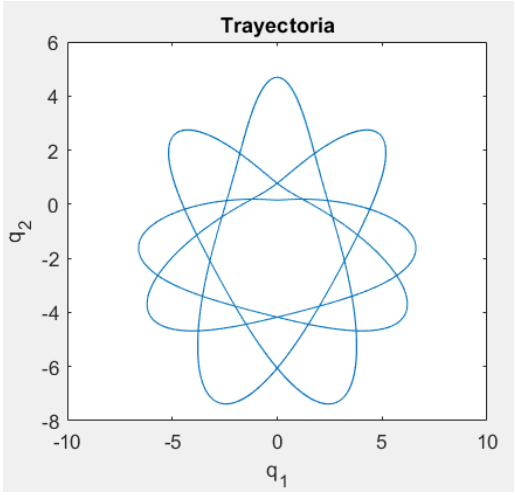


**Figura 4.55** 8-periódica tipo rotación para $H = 10.5$.

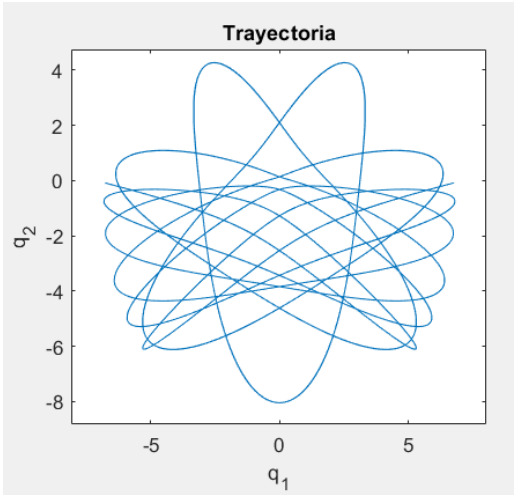


**Figura 4.56** 17-periódica para $H = 10.75$.

### 4.1.7 Dinámica valores altos de energía

A partir de cierto valor de $H$ la evolución del sistema con esta se vuelve bastante plano. La región de tipo caótico va desapareciendo haciendo tender al sistema, una vez más, al orden; la región de tipo valley desaparece y, mientras a la de tipo hill le sucede lo mismo, parece ser que alrededor de este se generan las dinámicas anteriormente mentadas, mientras todas ellas van desplazándose hacia el exterior. Esto sucede incluso cuando la región de tipo hill a desaparecido por completo. Para apreciarlo mejor, es necesario ver una secuencia de secciones con una precisión adecuada y un paso en $H$ relativamente pequeño. En las Figura 4.57-Figura 4.60 se muestra parte de esta evolución.

Una región bastante interesante es la que queda cuando desaparece la de tipo hill y sus alrededores pues, esas curvas convergen en el punto en el que había trayectorias de tipo hill. Si se observa con mayor precisión se aprecia que esta es una zona inestable; pues, si se integra para unas condiciones iniciales cercanas a este punto y vemos los que genera en su correspondiente sección (Figura 4.61) se aprecia como hay una rama que se aleja de este punto y, sin embargo, no vuelve a ingresar en ella. Como un punto en donde puede haber un equilibrio inestable, en función de por donde se aproxime a este es por donde se alejará la rama (en el ejemplo los valores de $p_2$ y $q_2$ son menores que este punto, por lo que se aleja disminuyendo el valor de estas).

Así pues, si para las mismas condiciones iniciales se reduce el tiempo de simulación se puede apreciar que la dinámica inestable, como cabía esperar, es de tipo hill. Esto, y cómo la trayectoria se va deformando conforme aumenta el tiempo se puede apreciar en las Figura 4.62-Figura 4.64.

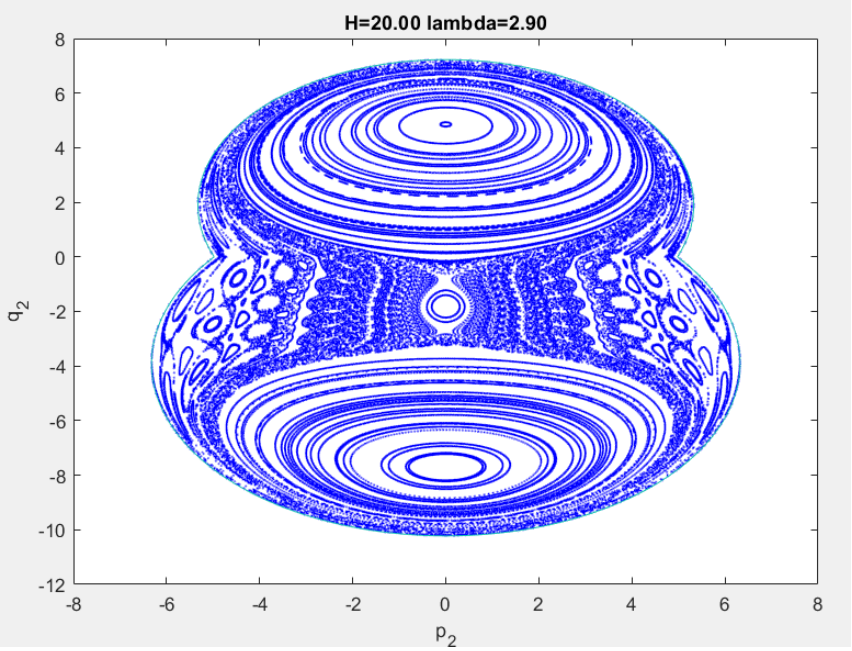


**Figura 4.57** SP $H = 20$.

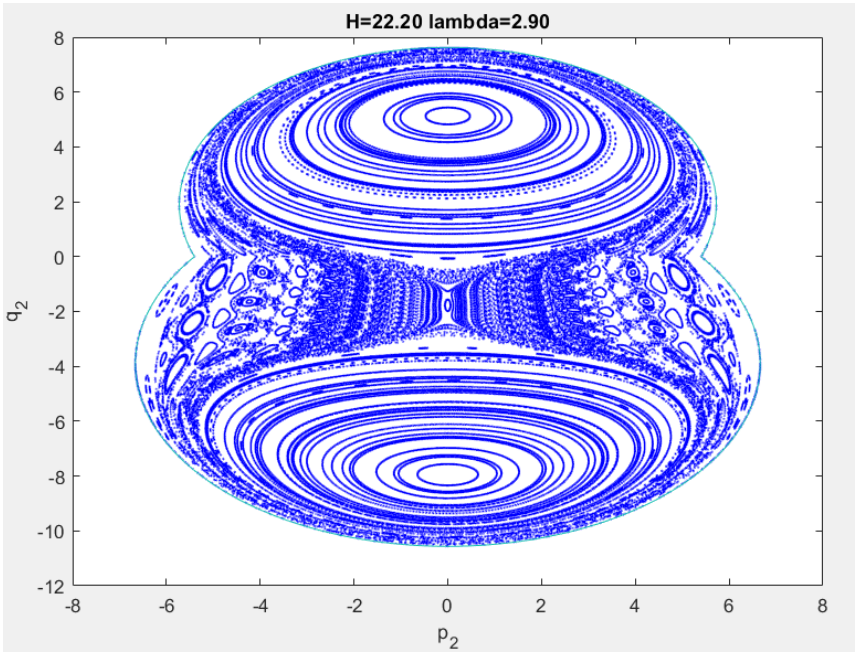


**Figura 4.58** SP $H = 22.2$.

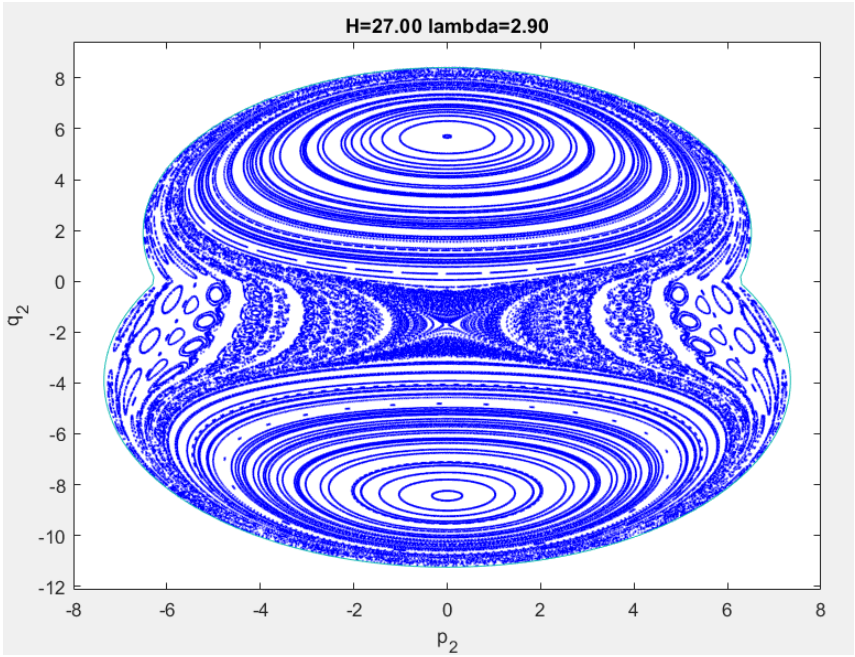


**Figura 4.59** SP $H = 27$.

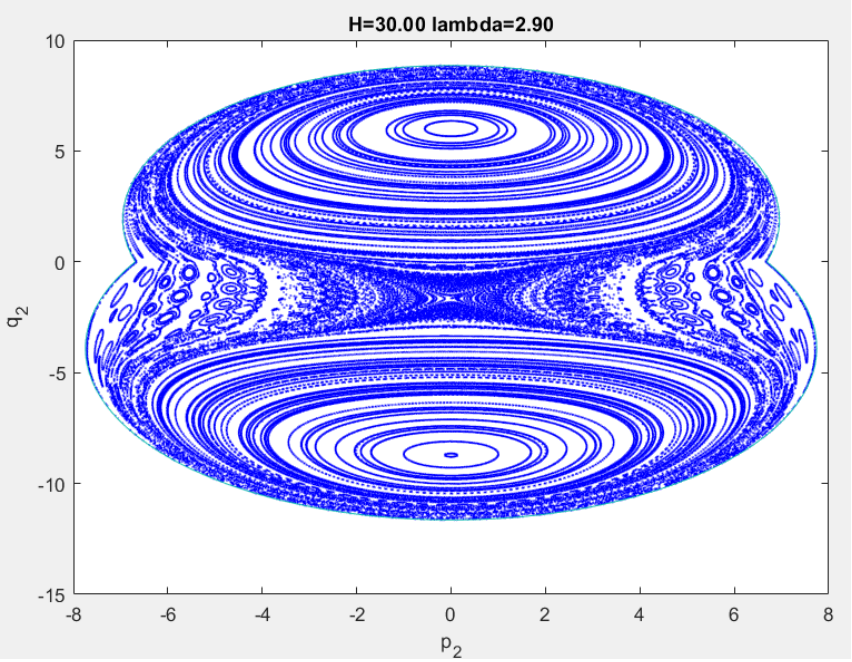


**Figura 4.60** SP $H = 30$.

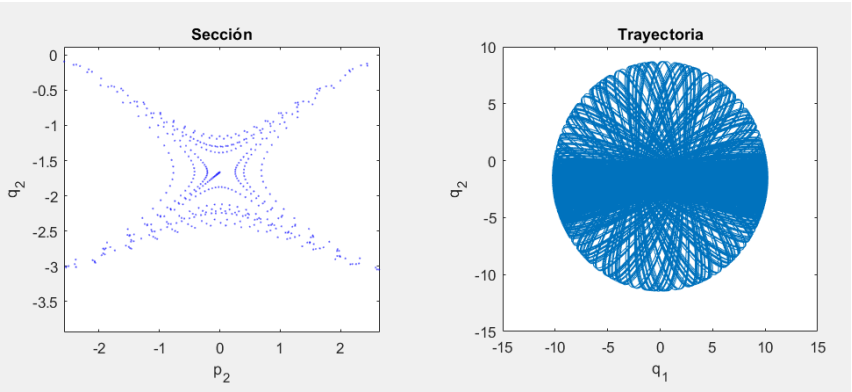


**Figura 4.61** $H = 30, tf = 5000$. Cond. ini.: $q_2 = -1.664$, $p_2 = -10^{-3}$.

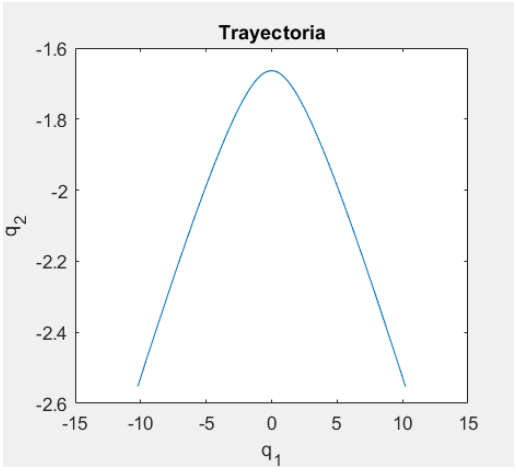


**Figura 4.62** $H = 30, tf = 20$. Cond. ini.: $q_2 = -1.664$, $p_2 = -10^{-3}$.

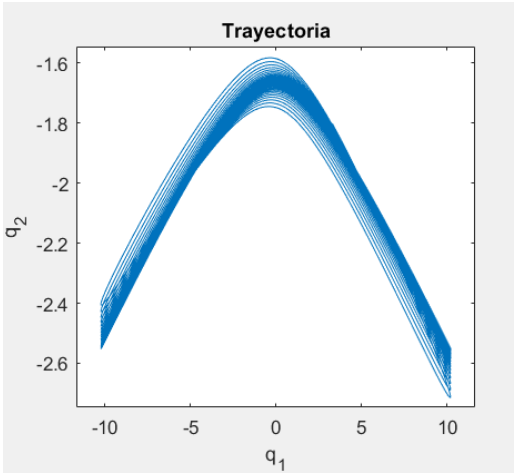


**Figura 4.63** $H = 30$, $tf = 330$. Cond. ini.: $q_2 = -1.664$, $p_2 = -10^{-3}$.

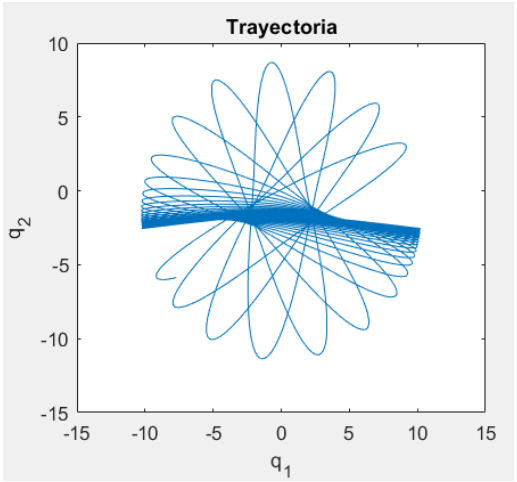


**Figura 4.64** $H = 30$, $tf = 500$. Cond. ini.: $q_2 = -1.664$, $p_2 = -10^{-3}$.

## 4.2 Efecto de la variación de $\lambda$

La mayor parte de lo aprendido para el caso de $\lambda = 2.9$ es extrapolable para el resto de valores del parámetro. Principalmente debido a un par de factores:

- A grandes rasgos podemos encontrar fundamentalmente el mismo tipo de dinámicas. Cuatro principales: hill, valley, herradura y circular; y derivadas: subarmónicos y asimetrías.
- Estas dinámicas evolucionan al aumentar $H$ tal y como se explicó en la sección anterior. De hecho, normalmente el sistema sigue un esquema orden-caos-orden [5] (como en el caso anterior) en donde nos encontramos comportamientos más interesantes en las dos transiciones entre cada una de las fases.

Esto nos permite llegar a entender en gran parte la dinámica para distintos valores de $\lambda$ tan solo con ver las secciones. Sin embargo, si se estudia con mayor detenimiento otros casos de $\lambda$, podemos observar las siguientes diferencias en función de su valor:

- El sistema no evoluciona a la misma velocidad con $H$. Cuanto más alto sea $\lambda$, más lenta será la evolución del sistema como podemos ver en la Figura 4.65, en donde para $H = 29$ no han desaparecido las regiones ni de tipo valley ni hill. También sabemos que la región superior aparece para $H = 2\lambda$, es decir, la evolución con $H$ depende de $\lambda$.
- Las regiones tipo hill y valley no tienen el mismo peso. A medida que aumenta $\lambda$ domina más la región de tipo valley, tal y como podemos ver en la Figura 4.66 y en la Figura 4.67; de esta misma forma, para $\lambda = 5.5$ desaparece ya un poco antes la región de tipo hill que la de tipo valley (representada en rojo por la Figura 4.68).
- Igualmente, si disminuye $\lambda$ se produce el efecto contrario tal y como sucede en Figura 4.69 y Figura 4.70, en donde la región tipo hill dominaría. En particular, como se aprecia en la figura anteriormente mentada, tenemos que para $\lambda \leq 1$ la región superior nace sobre la inferior. Y, en este caso (para $\lambda = 0.8$), la región de tipo rotación no surge hasta después de los subarmónicos, y lo hace a partir de la de tipo hill (Figura 4.71, Figura 4.72).
- Si se estudia con detalle las particularidades del sistema, este evidentemente presenta grandes diferencias. Es decir, distintos subarmónicos a diferentes energías, etc.

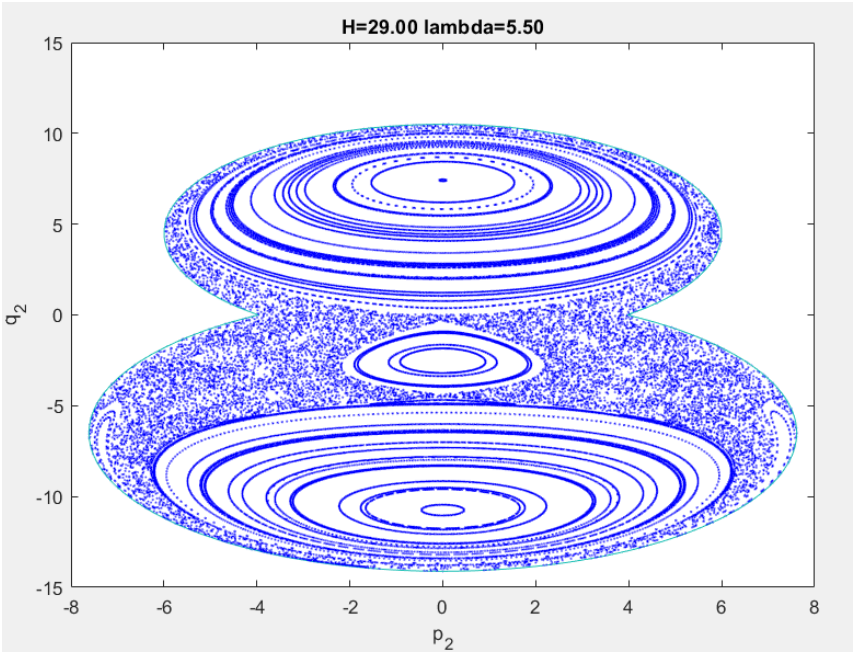


**Figura 4.65** SP para $H = 29$ y $\lambda = 5.5$ (con $tf = 9000$).

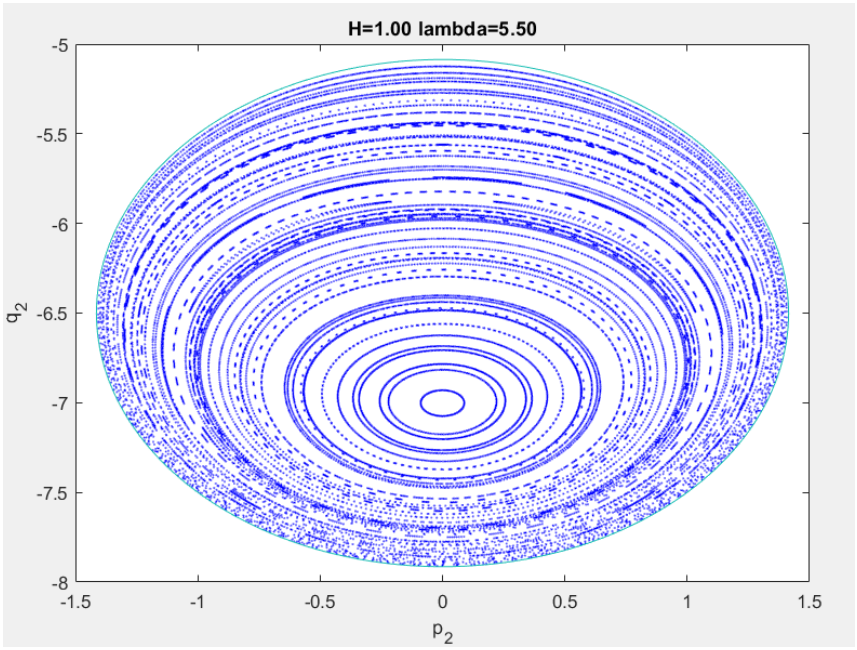


**Figura 4.66** SP para $H = 1$ y $\lambda = 5.5$ (con $tf = 9000$).

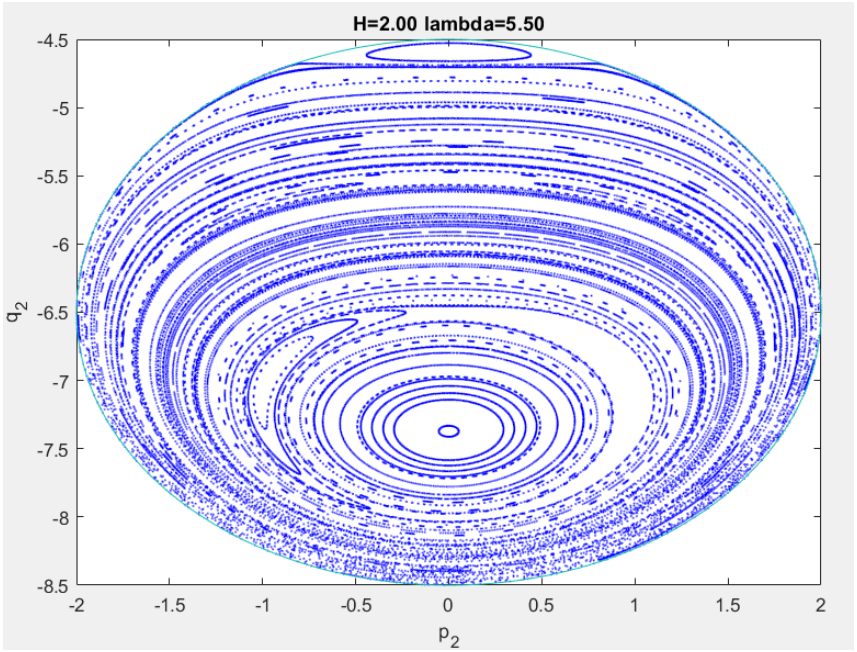


**Figura 4.67** SP para $H = 2$ y $\lambda = 5.5$ (con $tf = 9000$).

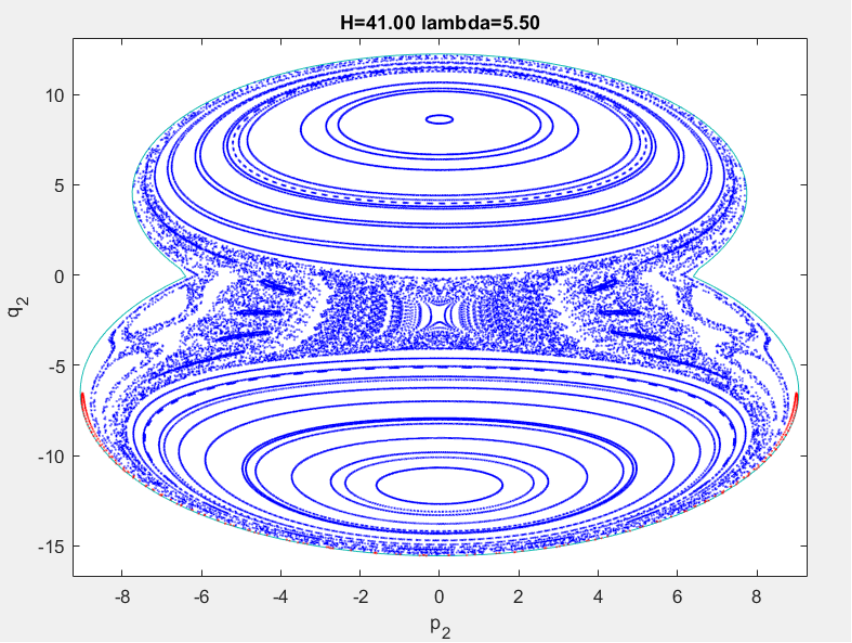


**Figura 4.68** SP para $H = 41$ y $\lambda = 5.5$ (con $tf = 9000$).

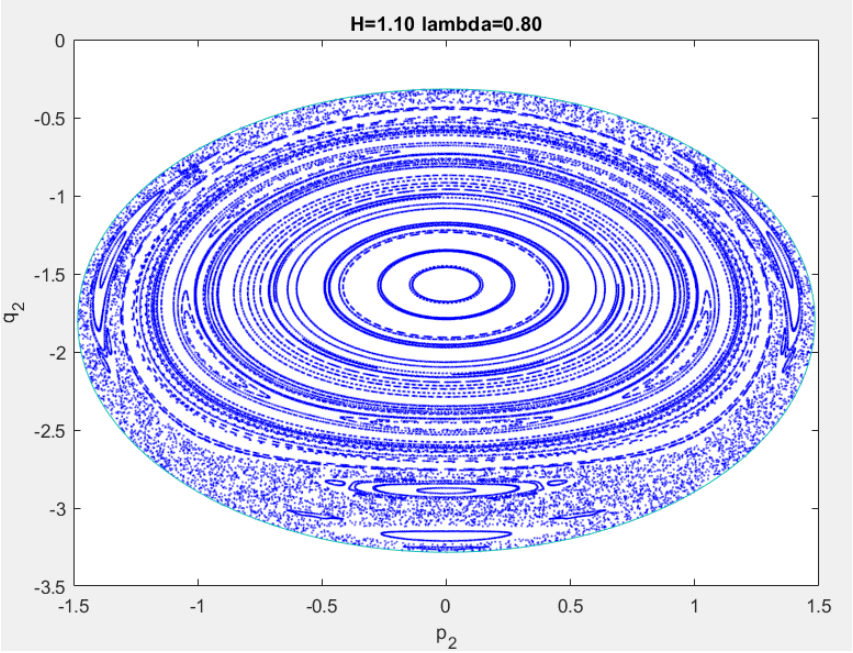


**Figura 4.69** SP para $H = 1.1$ y $\lambda = 0.8$ (con $tf = 9000$).

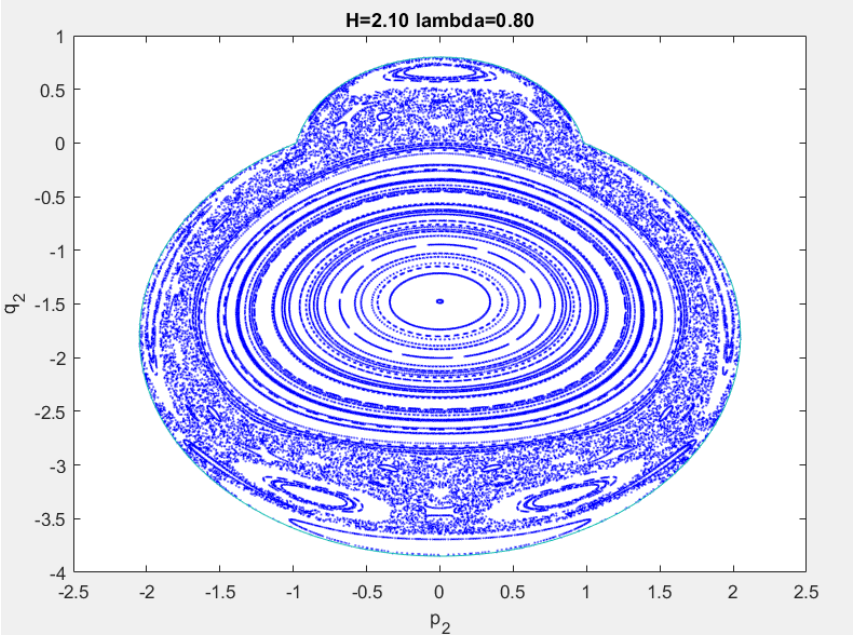


**Figura 4.70** SP para $H = 2.1$ y $\lambda = 0.8$ (con $tf = 9000$).

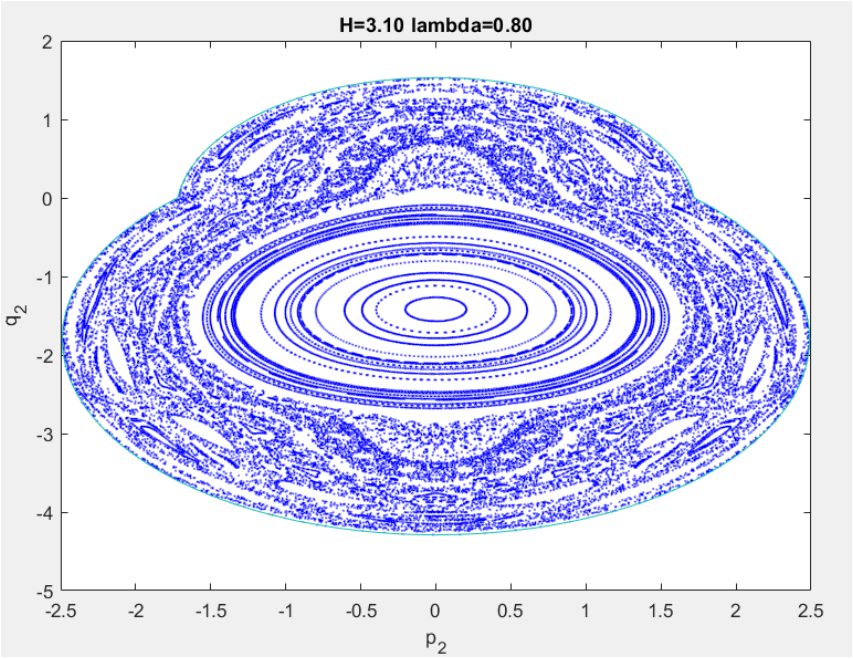


**Figura 4.71** SP para $H = 3.1$ y $\lambda = 0.8$ (con $tf = 9000$).

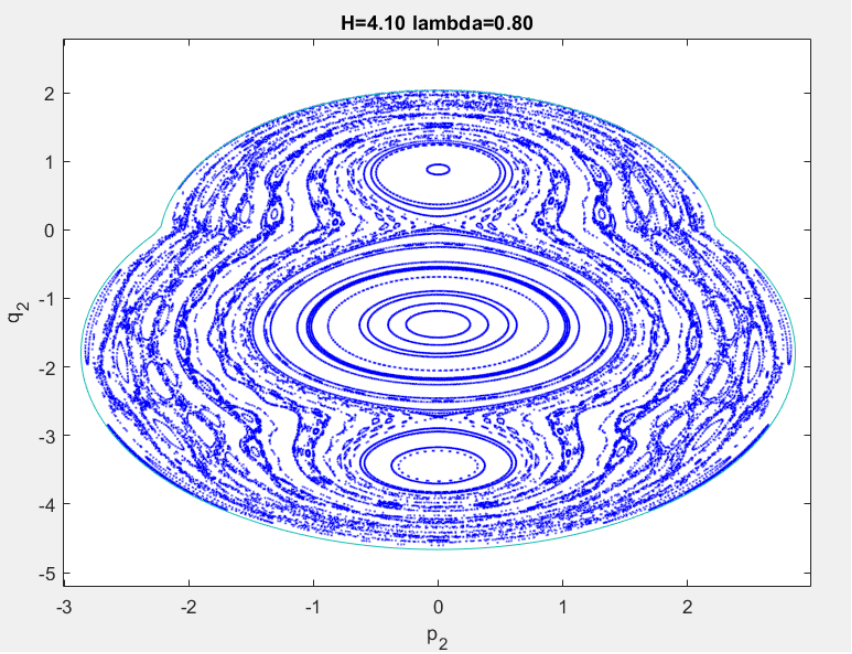


**Figura 4.72** SP para $H = 4.1$ y $\lambda = 0.8$ (con $tf = 9000$).

## 4.3 Caso límite: $\lambda = 0$

Para el caso concreto de $\lambda = 0$ se produce una particularidad. Esto se debe a que al sustituir $\lambda$ en el Hamiltoniano (2.9), este pasa de no ser lineal, a convertirse en un sistema lineal tal que:

$$H = \frac{p_1^2}{2} + \frac{p_2^2}{2} + \frac{q_1^2}{2} + \frac{q_2^2}{2} + q_2 - 1/2 \tag{4.2}$$

Así pues, la *sección de Poincaré* da un único punto para cada condición inicial como se puede ver en la Figura 4.73 y la Figura 4.74. Esto se debe a que todas las órbitas son periódicas y cuya solución (trayectoria) son elipses cuyas características son:

- Centradas en las coordenadas (0,-1).
- Escala en ambos ejes variables dependiendo de las condiciones iniciales y la energía del sistema.
- Giradas un cierto ángulo dependiendo de las condiciones iniciales.

Como ejemplo está la Figura 4.75.

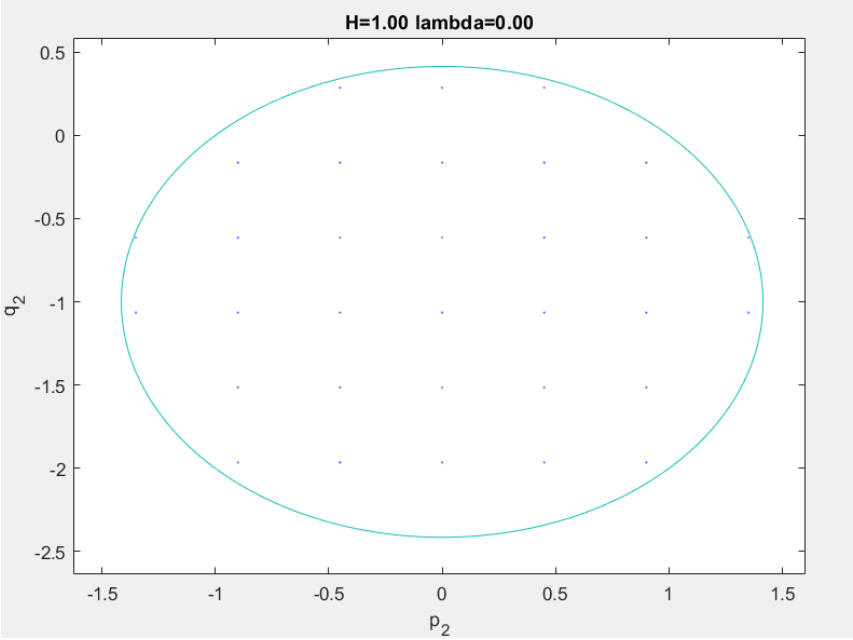


**Figura 4.73** SP para $H = 1$ y $\lambda = 0$.

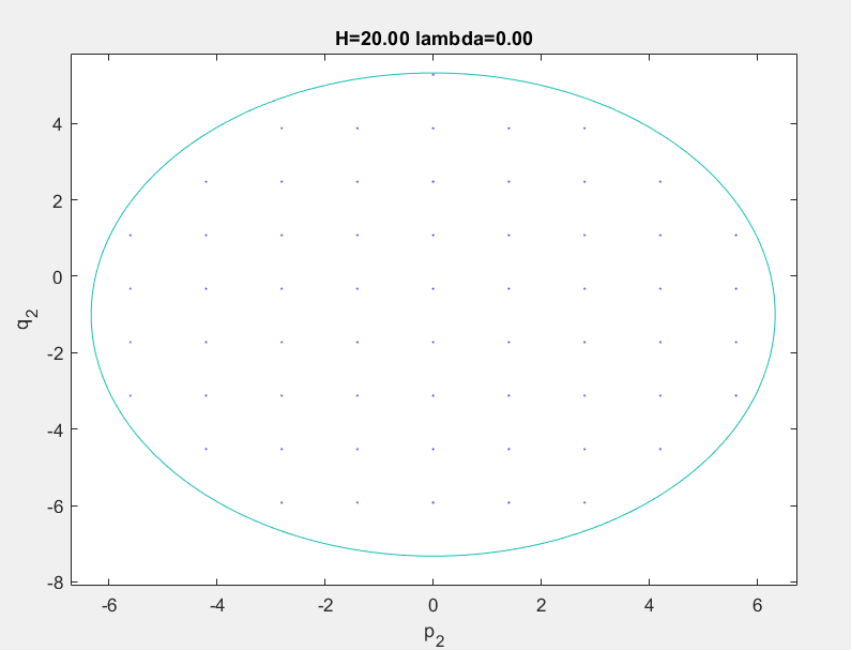


**Figura 4.74** SP para $H = 20$ y $\lambda = 0$.

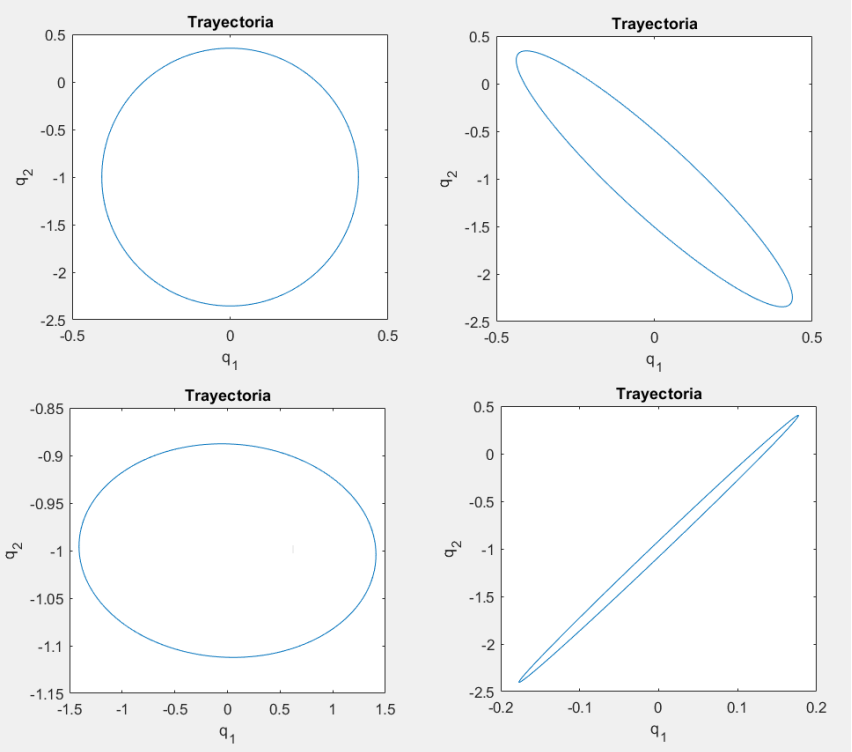


**Figura 4.75** Trayectorias para $H = 1$ y $\lambda = 0$, para distintas condiciones iniciales.

# 5 Conclusión

En el apartado de análisis dinámico se han podido comprobar las cualidades de la *sección de Poincaré*. Por un lado, estamos ante una herramienta que nos permite estudiar de manera realmente sencilla e intuitiva la dinámica de distintos tipos de sistemas. Sin embargo, obtener estas secciones en un sistema complejo de manera adecuada no es automático. Por ejemplo, para asegurar captar órbitas que ocupen una región muy pequeña haría falta un paso muy reducido que además de aumentarnos el coste computacional podría hacer que la sección diera una imagen no muy clara (como sucede con ciertas órbitas subarmónicas al estudiar el péndulo-muelle).

En general, el principal inconveniente de esta herramienta es que no existe una forma automatizada para escoger la sección y los parámetros a la hora de obtenerla.

Aun así, la *sección de Poincaré* no deja de ser una herramienta muy potente pues estudiar con ella un sistema y su evolución de una forma generalizada.

En cuanto al sistema péndulo-muelle es apreciable que un sistema compuesto, a pesar de partir de varios sistemas simples (dos en este caso), puede dar lugar a un sistema complejo el cual no se puede explicar simplemente a partir de las propiedades de los elementos que lo componen (cuando estos se estudian por separado). Esto se debe a que al combinar estos elementos surgen propiedades nuevas que se deben a las interacciones no lineales entre ellos.

Por ello, tal y como hemos podido estudiar, la combinación de dos elementos tan simples como el péndulo y el muelle da lugar a un sistema muy sensible a las condiciones iniciales. Pequeñas variaciones en estas pueden dar lugar a cambios radicales en la dinámica. Llegando incluso, a presentar un comportamiento impredecible, el cual es el que correspondería a la region de tipo caótico.

También se ha podido observar, a través del caso límite $\lambda = 0$, que para que surga este comportamiento no regular es una condición necesaria pero no suficiente la no linealidad del sistema.

# Apéndice A
# Código comentado

**Código A.1** Fichero Myguide.m.

```
function varargout = Myguide(varargin)
% MYGUIDE MATLAB code for Myguide.fig
%      MYGUIDE, by itself, creates a new MYGUIDE or raises the existing
%      singleton*.
%
%      H = MYGUIDE returns the handle to a new MYGUIDE or the handle to
%      the existing singleton*.
%
%      MYGUIDE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in MYGUIDE.M with the given input arguments.
%
%      MYGUIDE('Property','Value',...) creates a new MYGUIDE or raises the
%      existing singleton*. Starting from the left, property value pairs are
%      applied to the GUI before Myguide_OpeningFcn gets called. An
%      unrecognized property name or invalid value makes property application
%      stop. All inputs are passed to Myguide_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Myguide

% Last Modified by GUIDE v2.5 12-May-2019 12:53:14

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',   mfilename, ...
                   'gui_Singleton', gui_Singleton, ...
                   'gui_OpeningFcn', @Myguide_OpeningFcn, ...
                   'gui_OutputFcn', @Myguide_OutputFcn, ...
                   'gui_LayoutFcn', [] , ...
                   'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Myguide is made visible.
function Myguide_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject   handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)
% varargin  command line arguments to Myguide (see VARARGIN)

%Se inicializan los datos a 0 por defecto
%
%   H      : Vector de energia de las secciones
%   pe     : Celdas que contiene los puntos de las secciones (una celda
%            para cada valor de H)
%   lambda : Parametro de adimensionalizacion
%   n      : Numero de secciones
%   fid_ini: Booleano que indica si hay un archivo inicializado

handles.H=0;
handles.pe=0;
handles.lambda=0;
handles.n=0;
handles.fid_ini=0;
guidata(hObject, handles);

% Choose default command line output for Myguide
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Myguide wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Myguide_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject   handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in ginput_pushbutton.
function ginput_pushbutton_Callback(hObject, eventdata, handles)
% hObject   handle to ginput_pushbutton (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Este bloque nos permite escoger condiciones iniciales e integrar

if handles.fid_ini==0
    errordlg('No se han introducido datos');
    return;
end

H = str2double(get(handles.Hvalue_edit,'string'));
lambda = handles.lambda;
initaux = ginput(1);
x0 = 0; dy0 = initaux(1); y0 = initaux(2); % Cond. ini. ginput

TOL = str2double(get(handles.tol,'string'));
tf = str2double(get(handles.tf,'string')); tspan = [0,tf];

% Verifica si las cond. ini. estan dentro de los límites validos
aux=2*(H-lambda-0.5-y0-0.5*(sqrt(x0^2+y0.^2)-lambda).^2)-dy0.^2;
if (aux<0)
    errordlg('Valores no validos. Escoga otros.');
    return;
end
set(handles.Estado, 'ForegroundColor', 'red');set(handles.Estado,'string', '
    Calculando');pause(0.01);

dx0=sqrt(aux);
init=[x0 dx0 y0 dy0]';

% Solucion del sistema
options=odeset('events','on','RelTol',TOL,'AbsTol',TOL);
[t,sol,te,sole,ie]=ode45('pendulum', tspan, init,options,lambda,0,1);
ye=sole(:,3); dye=sole(:,4);


% Plot del espectro en frecuencia usando una ventana rectangular (FFT)
N = length(sol);
Fs = length(t)/tf;
f = Fs*(0:(N/2))/N;

fsol = abs(fft(sol))/N;
fsol = fsol(1:N/2+1,:);
fsol(2:end-1,:) = 2*fsol(2:end-1,:);
fmax_x = max(fsol(:,1));
fmax_y = max(fsol(2:length(fsol(:,3)),3));
for j=2:length(fsol)
    if fsol(j,1)>fmax_x/30
        k_x=f(j);
    end
     if fsol(j,3)>fmax_y/30
        k_y=f(j);
    end
end
plot(handles.freq_x,f,fsol(:,1));xlim(handles.freq_x,[0,k_x]);title(handles.
    freq_x,'Espectro de la frecuencia de q_1');xlabel(handles.freq_x,'f');
```

```
plot(handles.freq_y,f,fsol(:,3));xlim(handles.freq_y,[0,k_y]);title(handles.
    freq_y,'Espectro de la frecuencia de q_2');xlabel(handles.freq_y,'f');

% Plot de la solucion
plot(handles.x_y,sol(:,1),sol(:,3));title(handles.x_y,'Trayectoria');
xlabel(handles.x_y,'q_1');ylabel(handles.x_y,'q_2');
%plot(sol(:,1),sol(:,2));
%xlabel('x');ylabel('dx');
%plot(sol(:,3),sol(:,4));
%xlabel('y');ylabel('dy');
%plot(sol(:,2),sol(:,4));
%xlabel('dx');ylabel('dy');

%Plot de la seccion de Poincare

plot(handles.seccion,dye,ye,'b.','MarkerSize',2);
xlabel(handles.seccion,'p_2');ylabel(handles.seccion,'q_2');title(handles.
    seccion,'Sección');

hold(handles.poincare_axes,'on');
plot(handles.poincare_axes,dye,ye,'r.','MarkerSize',2);
xlabel(handles.poincare_axes,'p_2');ylabel(handles.poincare_axes,'q_2');
hold(handles.poincare_axes,'off');

set(handles.Estado, 'ForegroundColor', 'g');set(handles.Estado,'string', 'En
    espera');



% --- Executes on button press in todoh_pushbotton.
function todoh_pushbotton_Callback(hObject, eventdata, handles)
% hObject   handle to todoh_pushbotton (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Barrido de las secciones de Poincare en H
%
%   H_ini:  Valor de H inicial
%   H final: Valor de H final
%   tp:     Tiempo entre secciones

x0=0;
H=handles.H;
n=handles.n;
pe=handles.pe;
lambda=handles.lambda;

if handles.fid_ini==0
    errordlg('No se han introducido datos');
    return;
end

% Ajuste de las H de entrada a los valores disponibles mas cercanos
H_ini = str2double(get(handles.H_ini,'string'));
for i=2:n
    if H_ini-H(i)<0
        break;
```

```
    end
end
ini=i-1;
set(handles.H_ini,'string', num2str(H(i-1)));

H_fin = str2double(get(handles.H_fin,'string'));
for i=n-1:-1:2
    if H_fin-H(i)>0
        break;
    end
end
fin=i+1;
set(handles.H_fin,'string', num2str(H(i+1)));

t = str2double(get(handles.tp,'string'));

% Barrido de las secciones
for i=ini:fin
    allow = @(dy,y)2*(H(i)-lambda-0.5-y-0.5*(sqrt(x0^2+y.^2)-lambda).^2)-dy.^2;
    plot(handles.poincare_axes,pe{2,i},pe{1,i},'b.','MarkerSize',2); ax=axis(
        handles.poincare_axes);
    saux = sprintf('H=%4.2f lambda=%4.2f',H(i),lambda);title(handles.
        poincare_axes,saux);xlabel(handles.poincare_axes,'q_2');ylabel(handles.
        poincare_axes,'p_2');
    hold(handles.poincare_axes,'on');
    fcontour(handles.poincare_axes,allow,'LevelList',0);
    axis(handles.poincare_axes,ax);
    hold(handles.poincare_axes,'off');
    pause(t);
    set(handles.Hvalue_edit,'string', num2str(H(i)));
end



% --- Executes on button press in H_pushbutton.
function H_pushbutton_Callback(hObject, eventdata, handles)
% hObject   handle to H_pushbutton (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Muestra una seccion de un valor concreto H
%   newH: El valor de H a mostrar

n=handles.n;
H=handles.H;
pe=handles.pe;
lambda=handles.lambda;
x0=0;

if handles.fid_ini==0
    errordlg('No se han introducido datos');
    return;
end

% Ajusta H al valor mas proximo disponible
newH = str2double(get(handles.Hvalue_edit,'string'));
for i=2:n
```

```
    if newH-H(i)<0
        break;
    end
end
if newH<(H(i)+H(i-1))/2
    i=i-1;
end
set(handles.Hvalue_edit,'string', num2str(H(i)));
% Plot de la seccion
allow = @(dy,y)2*(H(i)-lambda-0.5-y-0.5*(sqrt(x0^2+y.^2)-lambda).^2)-dy.^2;
plot(handles.poincare_axes,pe{2,i},pe{1,i},'b.','MarkerSize',2); ax=axis(
    handles.poincare_axes);
saux = sprintf('H=%4.2f lambda=%4.2f',H(i),lambda);title(handles.poincare_axes,
    saux);xlabel(handles.poincare_axes,'p_2');ylabel(handles.poincare_axes,'q_2
    ');
hold(handles.poincare_axes,'on');
fcontour(handles.poincare_axes,allow,'LevelList',0);
axis(handles.poincare_axes,ax);
hold(handles.poincare_axes,'off');



function Data_Callback(hObject, eventdata, handles)
% hObject   handle to Data (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Data as text
%        str2double(get(hObject,'String')) returns contents of Data as a double

% Este bloque lee un archivo y carga los datos

fname = get(hObject,'String');

fid = fopen(fname,'r');
if fid==-1
    errordlg('No existe dicho archivo');
    return;
end
if handles.fid_ini==0
    handles.fid_ini=1;guidata(hObject, handles);
end
handles.Estado.String = 'Leyendo'; handles.Estado.ForegroundColor = 'r';pause
    (0.01);

inp = fscanf(fid,'%f %f\n',[2 inf])';
lambda = inp(1,1); n = inp(1,2); tf = inp(1,2); TOL = inp(2,2);
lec=3;pe=cell(2,n);
H=zeros(1,n);
for i=1:n
    H(i) = inp(lec,1);
    ny0=inp(lec,2);
    sole = inp(lec+1:ny0+lec,1:2);
    pe{1,i} = sole(:,1);
    pe{2,i} = sole(:,2);
    lec=lec+1+ny0;
end
```

```
fclose(fid);
handles.H=H;
handles.pe=pe;
handles.lambda=lambda;
handles.n=n;

set(handles.Estado, 'ForegroundColor', 'g');set(handles.Estado,'string', 'En
    espera');guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function H_ini_CreateFcn(hObject, eventdata, handles)
% hObject   handle to H_ini (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function tp_Callback(hObject, eventdata, handles)
% hObject   handle to tp (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of tp as text
%        str2double(get(hObject,'String')) returns contents of tp as a double


% --- Executes during object creation, after setting all properties.
function tp_CreateFcn(hObject, eventdata, handles)
% hObject   handle to tp (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function H_fin_Callback(hObject, eventdata, handles)
% hObject   handle to H_fin (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of H_fin as text
%        str2double(get(hObject,'String')) returns contents of H_fin as a
    double
```

```
% --- Executes during object creation, after setting all properties.
function H_fin_CreateFcn(hObject, eventdata, handles)
% hObject    handle to H_fin (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes during object creation, after setting all properties.
function Estado_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Estado (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% --- Executes during object creation, after setting all properties.
function Data_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Data (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function H_ini_Callback(hObject, eventdata, handles)
% hObject    handle to H_ini (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of H_ini as text
%        str2double(get(hObject,'String')) returns contents of H_ini as a
    double


function Hvalue_edit_Callback(hObject, eventdata, handles)
% hObject    handle to Hvalue_edit (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Hvalue_edit as text
%        str2double(get(hObject,'String')) returns contents of Hvalue_edit as a
     double


% --- Executes during object creation, after setting all properties.
function Hvalue_edit_CreateFcn(hObject, eventdata, handles)
```

```
% hObject   handle to Hvalue_edit (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function tf_Callback(hObject, eventdata, handles)
% hObject   handle to tf (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of tf as text
%        str2double(get(hObject,'String')) returns contents of tf as a double


% --- Executes during object creation, after setting all properties.
function tf_CreateFcn(hObject, eventdata, handles)
% hObject   handle to tf (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function tol_Callback(hObject, eventdata, handles)
% hObject   handle to tol (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of tol as text
%        str2double(get(hObject,'String')) returns contents of tol as a double


% --- Executes during object creation, after setting all properties.
function tol_CreateFcn(hObject, eventdata, handles)
% hObject   handle to tol (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles   empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
    defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
```

**Código A.2** Fichero P_function_auto.m.

```
function P_function_auto(lambda,ini)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Datos de entrada
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%    Se ha usado solo lambda e ini como argumentos de la funcion por sencillez
%    ini es numero de la seccion con la que se quiere empezar a calcular
%    (H(n)). De esta manera se puede dejar un calculo a medias o incluso
%    repartir aun mas la tarea entre varias computadoras.
%    Se puede añadir como argumento cualquier parametro (tf,TOL,a,b,H)


%H=[0.1:0.05:5 5.1:0.1:10 10.05:0.05:14 14.25:0.25:30];
H=[9.4:0.05:10];
tf=11000;
TOL=10^(-8);
tspan = [0 tf];

options=odeset('events','on','RelTol',TOL,'AbsTol',TOL);

num=length(H);
for n=ini:num

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Mallado de la seccion
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
listay0 = [];

ymin = -lambda-1-sqrt(2*H(n));
if H(n)<2*lambda
    ymax = -lambda-1+sqrt(2*H(n));
else
    ymax = lambda-1+sqrt(2*(H(n)-2*lambda));
end
dymin = -sqrt(2*H(n));

a=0.055;
b=0.04;

sim=floor(-0.5*dymin/(a+H(n)*b))*(a+H(n)*b)*2;
for i=ymin:2*(a+H(n)*b):ymax
    for j=-sim:(a+H(n)*b)*2:sim
        aux=2*(H(n)-lambda-0.5-i-0.5*(abs(i)-lambda)^2)-j^2;
        if aux>0 && i~=0
            dx0=sqrt(aux);
            init=[0 dx0 i j];
            listay0=[listay0;init];
        end
    end
end
```

```
%plot(listay0(:,4),listay0(:,3),'b.','MarkerSize',2);
size(listay0)
n
q1e=[];
q2e=[];
p1e=[];
p2e=[];

numit=length(listay0);


%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Integrador
%%%%%%%%%%%%%%%%%%%%%%%%%%%

parfor k=1:numit
    %disp(k)
    y0=listay0(k,:)';
    %[H k]
    [t,y,te,ye,ie]=ode45('pendulum', tspan, y0,options,lambda,0,1);
    q1e=[q1e;ye(:,1)]; p1e=[p1e;ye(:,2)]; q2e=[q2e;ye(:,3)]; p2e=[p2e;ye(:,4)];
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Imprimir en un archivo inp
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sole=[q1e p1e q2e p2e]';
if n==ini
    fname=sprintf('poincare_a%4.2f.inp',lambda);
    if n==1
        fid = fopen(fname,'w');
        fprintf(fid,'%.8f %.8f\n',lambda,num);
        fprintf(fid,'%.8f %.8f\n',tf,TOL);
    else
        %Si se quiere continuar un experimento parado a medias. Y, por tanto
        %no reescribir el archivo, sino continuarlo
        fid = fopen(fname,'a');
    end

end

fprintf(fid,'%.8f %.8f\n',H(n),length(sole));
fprintf(fid,'%.8f %.8f\n',sole(3:4,:));


clear p1e p2e q1e q2e listay0 sole
end
fclose(fid);
```

# Apéndice B
# Almacenamiento de los datos

Al final del Código A.2 se encuentra la metodología que se ha utilizado para almacenar los datos de los experimentos. El tipo de archivo escogido es un fichero *.inp. En él, solo se guardarán la pareja de datos necesarias para representar las *secciones de Poincaré*: $q_2$ y $p_2$ (para limitar el espacio de memoria necesario). Debido a esto se ha decidido almacenar todos los datos en el fichero en dos columnas, es decir, cada dos datos se introduce un salto de línea.

Para poder identificar a donde corresponde cada uno de los datos se crean líneas de datos adicionales a las de $q_2$ y $p_2$:

- Un par de líneas al inicio. La primera nos indica el valor de $\lambda$ al que pertenecen dichos datos, y el número $n$ de secciones en el archivo; la segunda nos informa del tiempo de simulación y de la tolerancia escogidas para el integrador.
- Para identificar a que sección pertenece cada uno de los datos es necesaria una línea previa a estos. Es decir, primero se escribe una línea que indica el valor de $H$ de la sección, y el número de par de datos que corresponden a dicha sección. Este último dato es necesario para poder indexar a la hora de leer el archivo, de esta forma sabremos que líneas son de este tipo, y cuales son valores de las variables.

# Apéndice C
# Tiempo computacional

Debido al alto coste computacional se ha hecho una prueba a diferentes máquinas a nuestra disposición para así, ver el tiempo que tardan estas en resolver la prueba tanto sin paralelización como con paralelización. Para la prueba se ha decidido resolver una serie de secciones y, el apartado de paralelización se ha realizado posteriormente a que matlab obtenga los permisos necesarios del sistema para paralelizar (para evitar que entre en el tiempo de computación).

**Tabla C.1** $\lambda = 2.9, tf = 1000, TOL = 10^{-6}$, 5 iteraciones en $H$.

| Computadora | Paralelo $t(s)$ | No paralelo $t(s)$ | Año | Nº Procesadores | CPU |
|---|---|---|---|---|---|
| MacPro home | 248 | 700 | 2009 | 8 | 2x2.26 GHz Quad Core Xeon |
| Imac | 184 | 643 | 2015 | 4 | 3,2 Intel Core I6 |
| MacPro IMUS (hilbert) | 54 | 528 | | 12 | |
| Anonumus915 | 290 | 1204 | | 12 | |
| Asus | | | 2015 | 8 | 4x2.59 GHz Intel Core i7-6700HQ |

Como se demuestran en los resultados experimentales, el uso de la paralelización es fundamental para impedir un tiempo de computación elevado. En principio, para obtener los distintos ficheros se ha utilizado la máquina 'Anonimus915'; sin embargo, como indica la Tabla C.1, la mejor máquina para ello es 'MacPro IMUS (hilbert)'.

Un detalle importante es tener en cuenta que el programa se ha ejecutado sobre un intérprete y que la computación no es en tiempo real. Esto quiere decir que, que el tiempo de computacion no es óptimo ni fijo. Depende, entre otras cosas, de cuán libres estén los procesadores.

# Índice de Figuras

# Índice de Tablas

# Índice de Códigos

# Bibliografía


[1] Thomas Douillet-Greiller, *Adaptación del métrodo de integración de Taylor en paralelo para el estudio de sistemas dinámicos*, ETSI, Junio 2012.

[2] Arnaud Kremer, *Herramientas computacionales para la visualización del comportamiento de sistemas dinámicos*, ETSI, Junio 2012.

[3] Dormand J., Prince P., *A family of embedded Runge–Kutta formulae. Journal of Computational and Applied Mathematics*, Volume 6, pages 19–26 (1980).

[4] Stephen Wiggins, *Introduction to applied nonlinear dynamical systems and chaos*, 2ª ed, ch. Poincaré Maps, Springer, 2003.

[5] J.P. van der Weele and E. de Kleine, *The order-chaos-order sequence in the spring pendulum*, Physica A, 228 (1996) 245-272.