In[∘]:=

```
SetDirectory[NotebookDirectory[]];
establece direct… directorio de cuaderno
Print[
escribe
  "================================================================================
   ================================="];
(** MODELO DE DICKE MODEL**)

(* Classical energy. *)
```

$$H[p\_, q\_, P\_, Q\_] := \frac{\omega o}{2}\left(Q^2+P^2\right) - \omega o + \frac{\omega}{2}\left(q^2+p^2\right) + 2\,\gamma\, q\, Q\,\sqrt{1-\left(\frac{Q^2+P^2}{4}\right)};$$

```
(* Parameters of the Hamiltonian, energy shell and time of integration. *)
ω = 1;
ωo = 1.0;
ϵ = -1.4;
γ = 2 γc;
T = 5000;
γc = Sqrt[ω ωo] / 2;
(* Dynamical System: Hamilton equations *)
F[{p_, q_, P_, Q_}] :=
  {-∂q H[p, q, P, Q], ∂p H[p, q, P, Q], -∂Q H[p, q, P, Q], ∂P H[p, q, P, Q]};
f = γ / γc;
(* Initial condition *)
pi = 0;
(*𝒫i=0;
𝒬i=0.707;*)
𝒫i = 0.6481;
𝒬i = -1.371;
qs1 = Solve[H[pi, x, 𝒫i, 𝒬i] == ϵ, x];
       resuelve
(*We select one the two roots*)
qi = x /. qs1[[2]];
(**********************************
 INFORMATION ************************************)
Print["Initial condition in the variables (p,q,P,Q):  pi = ",
escribe
  pi, ", qi = ", qi, ", Pi = ", 𝒫i, ", Qi = ", 𝒬i];
                              número pi
Print["The system has energy: E = ", H[pi, qi, 𝒫i, 𝒬i], ", and coupling γ = ", f , "γc"];
escribe                            número e
```

```
========================================================================================
  =====================
```

Initial condition in the variables (p,q,P,Q): pi = 0, qi = 2.0981, Pi = 0.6481, Qi = -1.371

The system has energy: E = -1.4, and coupling γ = 2.γc

In[∘]:=

```
Print[
escribe
```

```
escribe
  "-----------------------------------------------------------------------------
    "];
Timing[
cronometra
  (*Differential equations of the system defined by: p, q, P, Q *)
  Dinam = Table[D[yk[t], {t, 1}] == F[{y1[t], y2[t], y3[t], y4[t]}][[k]], {k, 1, 4}];
          tabla  deriva
  (*Initial conditions*)
  CI = {y1[0] == pi, y2[0] == qi, y3[0] == 𝒫i, y4[0] == 𝒬i};
  (*Jacobian Matrix *)
    jacobiano
  JacobianMatrix[f_List, v_List] := Outer[D, f, v];
  matriz jacobiana  lista  lista  produ··  deriva
  (*Variables: y1=p, y2=q, y3=P, y4=Q*)
    variables
  J = JacobianMatrix[F[{y1[t], y2[t], y3[t], y4[t]}], {y1[t], y2[t], y3[t], y4[t]}];
      matriz jacobiana
  (*Monodromy matrix 4x4*)
  Φ = Table[{yk[t], yk+1[t], yk+2[t], yk+3[t]}, {k, 5, 20, 4}];
      tabla
  (*Derivative of the Monodromy*)
    derivada
  DPhi = Flatten[J.Φ];
         aplana
  (* Ecuaciones Differential eqautions of the monodromy matrix *)
  Var = Table[D[yk[t], {t, 1}] == DPhi[[k - 4]], {k, 5, 20}];
        tabla  deriva
  (*Initial condition of the variational matrix must be the identity *)
  CIVar = Table[yk[0] == If[Mod[k, 5] == 0, 1, 0], {k, 5, 20}];
          tabla           si  operación módulo
  (*sol=NDSolve[Join[Dinam,CI,Var,CIVar],Table[yk[t],{k,1,20}],{t,0,T},
         resolved··· junta                 tabla
     StepMonitor:→Sow[x],Method→ Automatic,MaxStepSize→0.1,AccuracyGoal→ 15];*)
     monitor de pa···  siembra  método  automático  máximo tamaño de ···  objetivo de exactitud
  sol = NDSolve[Join[Dinam, CI, Var, CIVar], Table[yk[t], {k, 1, 20}],
        resolved··  junta                   tabla
    {t, 0, T}, Method → {"ExplicitRungeKutta", "DifferenceOrder" → 8},
               método                          orden de diferencias
    StepMonitor :→ Sow[x], MaxSteps → 10^9, AccuracyGoal → 30];
    monitor de pasos  siembra  máximo de pasos  objetivo de exactitud
  (*error limit*)
  error = 10^-6;
  Do[If[Evaluate[error > Abs[ϵ - H[y1[t], y2[t], y3[t], y4[t]]] /. sol[[1]]], tmax = t],
  r···  si  evalúa         valor absoluto
   {t, 0, T, 1}];
  Print["Numerical solution acceptable at time t: ",
  escribe
   tmax, " with error: 10^", Log10[error], " of tolerance."];
                             logaritmo en base 10
```

```
                                                    logaritmo en base 10
  T = tmax;
  (*Monodromy matrix at time*)
  PhiT = Table[{y_k[t], y_{k+1}[t], y_{k+2}[t], y_{k+3}[t]}, {k, 5, 20, 4}] /. sol /. t → T;
          tabla
  (*************************************
   LYAPUNOV EXPONENT ******************************)
  VV = Transpose[PhiT[[1]]].PhiT[[1]];
       transposición
  λ = (1 / (2 T)) Log[Norm[VV]];
                  lo… norma
  Print["Lyapunov exponent: λ = ", λ];
  escribe
  Print[
  escribe
   "--------------------------------------------------------------------------
     --"];
 ];
tCPU = %;
Print["Time used: ", tCPU[[1]] / 60, " minutes."];
escribe
Print[
escribe
  "==========================================================================
    =================================="];
```

```
--------------------------------------------------------------------------------

Numerical solution acceptable at time t: 5000 with error: 10^-6 of tolerance.

Lyapunov exponent: λ = 0.051226

--------------------------------------------------------------------------------

Time used: 0.0046875 minutes.

=========================================================================================
  ======================
```

In[ ]:=

```
datOrb = Table[{t, y1[t], y2[t], y3[t], y4[t]} /. sol[[1]] /. t → i, {i, 0, 300, 0.05}];
         tabla
ListPlot[datOrb[[1 ;; -1, {3, 2}]], Joined → True, AspectRatio → 1]
representación de lista                        unido     verd··· cociente de aspecto
```

Out[ ]=