CHAOS IN ROBOTIC CONTROL EQUATIONS

ED5314 Parallel Manipulators Course Project

Adithya Ramesh

ED11B002

The control equations of a parallel manipulator by applying PD control after feedback linearization, are given by,

α q''+ β = α hat Fdash + β hat

where α = M,

$\beta = J \eta q^{T} A^{-1} J \eta q' q'' + B(C q' + G)$,

Fdash= $q_d'' + Kv\,(q_d' - q') + Kp\,(q_d - q) = q_d'' + Kv\, e' + Kp\, e$,

and α hat, β hat represent the estimates of α , β respectively.

When the estimates α hat, β hat are exactly equal to the actual quantities α , β ,we have,

e'' + Kv e'+ Kp e = 0,

which is a set of uncoupled linear second order ODEs which for positive definite gain matrices Kp and Kv is asymptotically stable as $e -> 0$ as $t -> Infinity$.

However,in practice this does not happen because

(i) We do not know the model parameters such as mass,inertia,legnths exactly.Even the best values for these parameters from measurement will inevitably contain errors compared to the true values of these parameters.

(ii)The state q whose value we use to calculate α hat, β hat to cancel non linearities at a particular instant will contain sensor errors and will also have a finite delay.

.In such a situation we have a set of highly coupled nonlinear ODEs whose behaviour is not as easy to predict as the previous case.

$e'' + Kv.\ e' + Kp.e = \alpha\text{hat}^{-1}((\alpha-\alpha\text{hat})q''+(\beta-\beta\text{hat}))$

These equations being non linear and coupled can possibly exhibit chaos,understanding which is the purpose of this study.

Chaotic Dynamics is characterized mainly by the following properties :

a) They exhibit sensitive dependence to initial conditions,that is,chaotic trajectories locally diverge away from each other and small changes in starting conditions build up exponentially fast into large changes in evolution.

b) They are bounded random like steady state trajectories.They converge to a set in the phase space,called a strange attractor,which is not a simple manifold like a point,circle or torus but has a complex(fractal) geometrical structure with a Fractional Hausdorrf dimension.

c)They exhibit *Topological mixing* (or *topological transitivity*) which means that the system will evolve over time so that any given region or open set of its phase space will eventually overlap with any other given region

d)Density of Periodic Orbits which means that every point in the space is approached arbitrarily closely by periodic orbits.

A consequence of sensitivity to initial conditions is that if we start with only a finite amount of information about the system (as is usually the case in practice), then beyond a certain time the system will no longer be predictable.Also,disturbances can take the system far away from its intended configuration quickly.

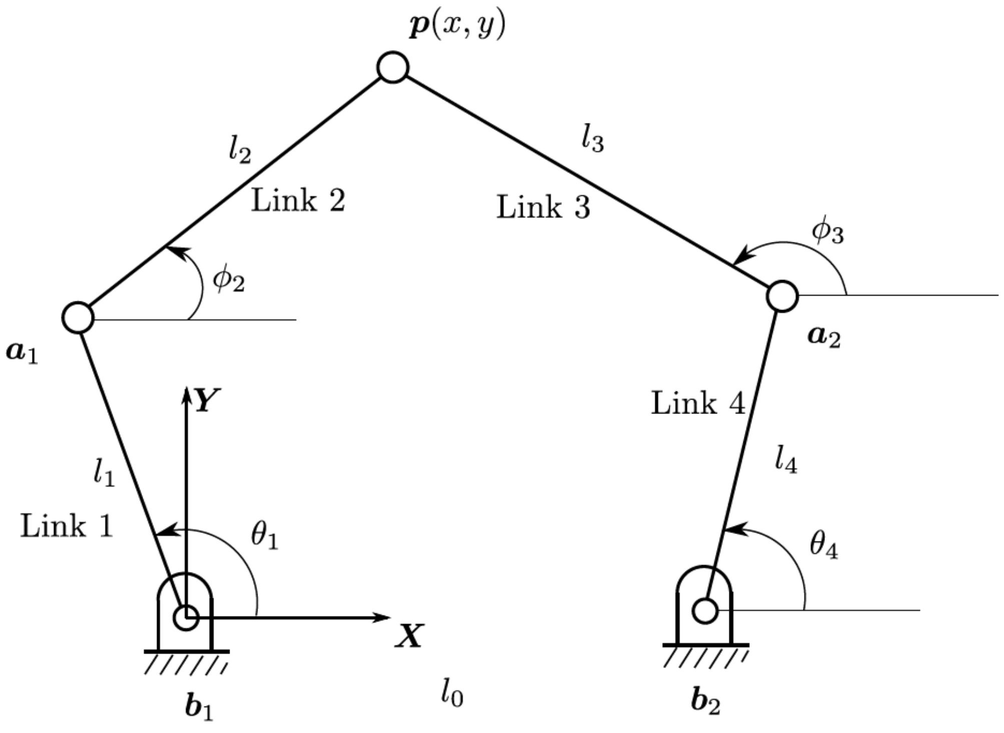
These are situations we would rather want to avoid while operating a robotic manipulator.Hence it is important to understand the conditions under which Chaos occurs so that they can be suitably avoided.

Lyapunov Exponents provide a quantitative measure of the diverge or convergence of nearby trajectories for a dynamical system.A positive Lyapunov exponent implies chaotic dynamics.

While tracking a periodic trajectory, a well behaved system follows closed curves in the Phase Space.Whereas for Chaotic systems,the phase space plots fill an entire region.

We shall thus calculate Lyapunov Exponents and obtain Phase Space Plots of our system and use these as tools to identify chaotic dynamics.

We shall consider a representative parallel manipulator,a 5 bar for the purpose of this study



(a) The planar five-bar manipulator

The true mass and inertia parameters used are as follows

numdata = {$g \to 9.806$, ic1 → 0.05881061447520093, ic2 → 0.024810727981725386, ic3 → 0.024810727981725386, ic4 → 0.05881061447520093, l0 → 1, l1 → $\frac{4}{5}$, l2 → $\frac{3}{5}$, l3 → $\frac{3}{5}$, l4 →

$\frac{4}{5}$, m1 → 1.1026990214100174, m2 → 0.827024266057513, m3 → 0.827024266057513, m4 → 1.1026990214100174, mp → 1}

The approximate mass and inertia parameters used are as follows

numdatahat= {ic1 → 0.05881061447520093(1 + ϵ), ic2 → 0.024810727981725386(1 + ϵ), ic3 → 0.024810727981725386(1 + ϵ), ic4 → 0.05881061447520093(1 + ϵ), l0 → 1 + ϵ, l1 → $\frac{4(1+\epsilon)}{5}$, l2 → $\frac{3(1+\epsilon)}{5}$, l3 → $\frac{3(1+\epsilon)}{5}$, l4 → $\frac{4(1+\epsilon)}{5}$, m1 → 1.1026990214100174(1 + ϵ), m2 → 0.827024266057513(1 + ϵ), m3 → 0.827024266057513(1 + ϵ), m4 → 1.1026990214100174(1 + ϵ), mp → 1 + ϵ, g → 9.806}

Here ϵ represents the mismatch between the true model and the approximate model.A positive ϵ represents overestimation and a negative ϵ represents underestimation.

The reference trajectory we are trying to track is,

```
θ1d=π/6*Sin[2*t]
θ4d=π/3*Sin[2*t]
```

From **α** q''+ **β** = **α hat Fdash** + **β hat** we obtain q'' as,

q''= **α^{-1}(** **α hat Fdash** + **β hat** - **β** **).**

Define an augmented state vector Q consisting of q,q'and t as,

Q={q,q',t}.

For the 5 bar we are considering,

q = {θ1[t], ϕ2[t], ϕ3[t], θ4[t]}

Therefore Q = {θ1[t], ϕ2[t], ϕ3[t], θ4[t], θ1′[t], ϕ2′[t], ϕ3′[t], θ4′[t], t}.

Our n = 9 dimensional smooth dynamical system whose state vector is Q, is described by the differential equation

Q ' = F(Q)

The purpose of including t in the State Vector Q is that we want an autonomous system.Consequently the state t evolves according to the trivial relation

t' = 1

Thus,

Q ' = F(Q)= $\{\theta1'[t], \phi2'[t], \phi3'[t], \theta4'[t], \theta1''[t], \phi2''[t], \phi3''[t], \theta4''[t], t'\}$.

= $\{Q[[5]], Q[[6]], Q[[7]], Q[[8]], q''[[1]], q''[[2]], q''[[3]], q''[[4]], 1\}$.

where q'' as derived previously is simply a function of q , q' which are themselves contained in Q.

Thus given Q we can estimate F(Q) in a straightforward manner.

In the attached Mathematica File,I have defined a Module numF which computes F(Q) given Q numerically.

```
 (*Module to compute F(Q) given Q numerically  *)
numF[Q_?(VectorQ[#, NumericQ] &), kp_, ϵ_] :=
  Module[{q, qdash, t1, numJηq, numJηqhat, numJηqdot, numJηqdothat, tempC, tempChat, tempG,
    tempGhat, A, Ahat, B, Bhat, α, αhat, β, βhat, Fdash, qddot, tempF, Kp, Kv, numdatahat},
   (*Numerical Data along with mismatch ϵ*)
   numdatahat = ({ic1 → (1 + ϵ) (ic1 /. numdata), ic2 → (1 + ϵ) (ic2 /. numdata), ic3 → (1 + ϵ) (ic3 /. numdata),
      ic4 → (1 + ϵ) (ic4 /. numdata), l0 → (1 + ϵ) (l0 /. numdata), l1 → (1 + ϵ) (l1 /. numdata),
      l2 → (1 + ϵ) (l2 /. numdata), l3 → (1 + ϵ) (l3 /. numdata), l4 → (1 + ϵ) (l4 /. numdata),
      m1 → (1 + ϵ) (m1 /. numdata), m2 → (1 + ϵ) (m2 /. numdata), m3 → (1 + ϵ) (m3 /. numdata),
      m4 → (1 + ϵ) (m4 /. numdata), mp → (1 + ϵ) (mp /. numdata), g → 9.806});


  (*Gains chosen for errors to die out in a Critically damped manner*)
  Kp = kp * IdentityMatrix[4];
  Kv = 2 (kp)^0.5 * IdentityMatrix[4];

  q = {Q[[1]], Q[[2]], Q[[3]], Q[[4]]};
  qdash = {Q[[5]], Q[[6]], Q[[7]], Q[[8]]};
  t1 = Q[[9]];

  numJηq = (Jηq /. (Thread[{θ1[t], ϕ2[t], ϕ3[t], θ4[t]} → q]) /. numdata);
  numJηqhat = (Jηq /. (Thread[{θ1[t], ϕ2[t], ϕ3[t], θ4[t]} → q]) /. numdatahat);
  numJηqdot =
   (Jηqdot /. (Thread[{θ1[t], ϕ2[t], ϕ3[t], θ4[t]} → q]) /.
      (Thread[{θ1'[t], ϕ2'[t], ϕ3'[t], θ4'[t]} → qdash]) /. numdata);
  numJηqdothat =
   (Jηqdot /. (Thread[{θ1[t], ϕ2[t], ϕ3[t], θ4[t]} → q]) /.
      (Thread[{θ1'[t], ϕ2'[t], ϕ3'[t], θ4'[t]} → qdash]) /. numdatahat);
  α = massmat[q, t1, numdata];
  tempC = cmat[q, qdash, t1, numdata];
  tempG = gvec[q, t1, numdata];
  A = numJηq.Inverse[α].Transpose[numJηq];
  B = IdentityMatrix[4] - Transpose[numJηq].Inverse[A].numJηq.Inverse[α];
  β = Transpose[numJηq].Inverse[A].numJηqdot.qdash + B.(tempC.qdash);
  Fdash = ((D[qd, {t, 2}]) /. {t → t1}) + Kv.(((D[qd, t]) /. {t → t1}) - qdash) + Kp.((qd /. {t → t1}) - q);

  αhat = massmat[q, t1, numdatahat];
  tempChat = cmat[q, qdash, t1, numdatahat];
  tempGhat = gvec[q, t1, numdatahat];
  Ahat = numJηqhat.Inverse[αhat].Transpose[numJηqhat];
  Bhat = IdentityMatrix[4] - Transpose[numJηqhat].Inverse[Ahat].numJηqhat.Inverse[αhat];
  βhat = Transpose[numJηqhat].Inverse[Ahat].numJηqdothat.qdash + Bhat.(tempChat.qdash);

  qddot = Inverse[α].(αhat.Fdash + βhat - β);
  tempF = {qdash[[1]], qdash[[2]], qdash[[3]], qdash[[4]],
    qddot[[1]], qddot[[2]], qddot[[3]], qddot[[4]], 1};
  Return[tempF];
 ];
```

Define some utilities :

Module RKStepnum : Performs numerical integration of the function numf[y] by the Runge-Kutta Method for a time dt from an initial state y0 and returns the final state.

```
RKStepnum[numf_, y_, y0_, dt_, kp_, ϵ_] :=
Module[{k1, k2, k3, k4},
  k1 = dt N[numf[(y /. Thread[y -> y0]) , kp, ϵ]];
  k2 = dt N[numf[ (y /. Thread[y -> y0 + k1 / 2]) , kp, ϵ]];
  k3 = dt N[numf[ (y /. Thread[y -> y0 + k2 / 2] ) , kp, ϵ]];
  k4 = dt N[numf[ (y /. Thread[y -> y0 + k3]) , kp, ϵ]];
  y0 + (k1 + 2 k2 + 2 k3 + k4) / 6 ]
```

Module RKPointnum : Performs numerical integration of the function numf[y] by the Runge-Kutta Method for a time t1 in steps of dt from an initial state y0 and returns the final state.

```
RKPointnum[numf_, y_, y0_, {t1_, dt_} , kp_, ϵ_] :=
  Nest[ RKStepnum[numf, y, #, N[dt], kp, ϵ] &, N[y0],
   Round[N[t1 / dt]] ]
```

Module RKListnum: Performs numerical integration of the function numf[y] by the Runge-Kutta Method for a time t1 in steps of dt from an initial state y0 and returns the entire sequence of states.

```
RKListnum[numf_, y_, y0_, {t1_, dt_}, kp_, ϵ_] :=
  NestList[ RKStepnum[numf, y, #, N[dt], kp, ϵ] &, N[y0],
   Round[N[t1 / dt]] ]
```

Obtaining Phase Space Plots of the system

Module PhaseSpaceCnum : Simulates the system over the transient period " trans" using RKPointnum which returns the final state.Use this state as initial condition and simulate for the required time "time" in steps of " stepsize" using RKListnum which ultimately returns the entire sequence of states.Feed this sequence to showOrbit1 to plot the phase space trajectory.

```
PhaseSpaceCnum[numf_, initcond_, time_, trans_:0,
     stepsize_:0.02, patt_List, kp_, ϵ_] :=
Module[{n, a, x, x0, sol, i},
  n = Length[initcond];
  x = Array[a, n];
  x0 = RKPointnum[numf, x, initcond, {trans, stepsize}, kp, ϵ];
  sol = RKListnum[numf, x, x0, {time, stepsize}, kp, ϵ];
  showOrbit1[sol, patt, Line]
 ]
```

Module showOrbit1 : Takes sequence of states as input and plots the corresponding Phase Space Trajectories

```
showOrbit1[sol_, patt_List, f_] :=
Module[{n, orbit1, orbit2, orbit3, orbit4, x, z},
  n = Length[First[sol]];

    orbit1 = Map[#[[patt[[1]]]] &, sol] ;
   orbit2 = Map[#[[patt[[2]]]] &, sol] ;
   orbit3 = Map[#[[patt[[3]]]] &, sol] ;
   orbit4 = Map[#[[patt[[4]]]] &, sol] ;
    Show[Graphics[f[orbit1]], PlotRange -> All, Axes -> Automatic, AxesLabel → {"θ1[t]", "θ1'[t]"}]
    Show[Graphics[f[orbit2]], PlotRange -> All, Axes -> Automatic, AxesLabel → {"ϕ2[t]", "ϕ2'[t]"}]
   Show[Graphics[f[orbit3]], PlotRange -> All, Axes -> Automatic, AxesLabel → {"ϕ3[t]", "ϕ3'[t]"}]
    Show[Graphics[f[orbit4]], PlotRange -> All, Axes -> Automatic, AxesLabel → {"θ4[t]", "θ4'[t]"}]
 ]
```

Calculating Lyapunov Exponents :

(The following procedure of Numerically Calculating Lyapunov Exponents is attributed to Reference #3,Marco Sandri,Italy)

Consider two nearby points x_0 and $x_0 + u_0$ in the Phase Space M,where u_0 is a small perturbation of the initial point x_0.After a time t,their images under the flow will be $f^t(x_0)$ and $f^t(x_0 + u_0)$ and the perturbation u_t will be

$$u_t = f^t(x_0 + u_0) - f^t(x_0) = (D_{x0} f^t(x_0)).\, u_0$$

which is obtained by linearizing $f^t(x_0)$around x_0.

Therefore the average exponential rate of divergence or convergence of the two trajectories is given by

$$\lambda(x_0 , u_0) = \lim_{t\to Infinity} \left(\frac{1}{t}\right) \ln(|| u_t ||/|| u_0||) = \lim_{t\to Infinity} \left(\frac{1}{t}\right) \ln(D_{x0}\, f^t(x_0)).$$

Thus if $\lambda(x , u) > 0$,one has exponential divergence of nearby orbits.It can be shown that,under very weak smoothness conditions on the dynamical system,the limit exists and is finite for all points $x_0 \in M$,and for almost all tangent vectors u_0 it is equal to the largest LCE λ_1.

So far we have defined LCEs of vectors i.e LCEs of order 1.A useful generalization is to define LCEs of order p , $1 \leq p \leq n$,which describes the mean rate of growth of a p-dimensional volume in the tangent space.

Consider a parallelopiped U_0 in the tangent space whose edges are the p vectors u_1,... u_p ,LCEs of order p are then defined by

$\lambda p(x_0 , U_0)= \quad \lim_{t\to Infinity} \left(\frac{1}{t}\right) \ln(Vol^p\ [\ Dx_0\ f^t(U_0)]\ )$.

where Vol^p is the p dimensional volume defined in the tangent space.Oseledec 1968 shows that one can find p linearly independent vectors

$\lambda p(x_0 , U_0)= \lambda_1+......+ \lambda p$.

That is,each LCE of order p is equal to the sum of the p largest LCEs of order 1.

For p = n,we obtain the mean exponential rate of growth of the phase space volume.

Estimation of LCEs

The tangent vector u_t,defined earlier evolves in time satisfying the so-called variational equation

$\Phi_t'(x_0)= D_x F(f^t(x_0)).\ \Phi_t(x_0)\ ,\ \Phi_0(x_0)= I$

where $\Phi_t(x_0)$ is the derivative with respect to x_0 of f^t at x_0, that is $\Phi_t(x_0)= D_{x0}\ f^t(x_0)$.

The previous equation is a matrix valued time varying linear DE whose coefficients depend on the evolution of the original system.

Therefore to calculate the trajectory,we must integrate the combined system

$x' = F(x)$

$\Phi'=D_x F(x).\ \Phi$

with the Initial conditions, $x(t_0)=x_0$, $\Phi(t_0) = I$.

In the attached Mathematica File,I have defined a **Module numjacF** to compute the jacobian of F w,r,t Q at Q given Q numerically.

```
(*Module to compute Jacobian of F given Q numerically *)

numjacF[Q_ ? (VectorQ[#, NumericQ] &), kp_, ϵ_] := Module[{tempJ, i, j, h, ΔQ},
   tempJ = ConstantArray[0, {9, 9}];
   h = 0.001;
   For[i = 1, i ≤ 9, i++,
    For[j = 1, j ≤ 9, j++,
     ΔQ = ConstantArray[0, 9];
     ΔQ[[j]] = h;
     tempJ[[i, j]] = (((numF[Q + ΔQ, kp, ϵ])[[i]]) - ((numF[Q - ΔQ, kp, ϵ])[[i]])) / (2 * h);
    ]];
   Return[tempJ];
  ];
```

Let the combined system be represented by the augmented state X ={Q , Φ}.

Define a module numG to return the state transition function of the combined system,

```
(*Module to return the State transition function of the combined system*)
numG[X_, kp_, ϵ_] := Module[{Q, ϕ, tempg},
   Q = X[[1]];
   ϕ = X[[2]];
   tempg = ϕ.Transpose[numjacF[Q, kp, ϵ]];

   Return[{numF[Q, kp, ϵ], tempg}];
  ];
```

Calculating the entire LCE spectrum

We adopt an algorithm which relies on the calculation of order-p LCEs and the repeated application of the Gram Schmidt orthonormalization procedure.

Given a set $\{u_1,......,u_p\}$ of p linearly independent vectors in R^n,the Gram-Schmidt procedure generates an orthonormal set $\{v_1,......,v_p\}$ of vectors which spans the same subspace spanned by $\{u_1,......,u_p\}$.

The vectors v_i are given by

$w_1 = u_1,\ v_1= w_1/|| w_1||$

$w_2 = u_2 - < u_2,v_1>v_1 ,\ v_2= w_2/|| w_2||,$

$w_p = u_p - \ \Sigma < u_p,v_i>v_i ,\ \ v_p= w_p/|| w_p||$

The volume of the parallelopiped spanned by $\{u_1,......,u_p\}$ is

$\text{Vol}\ \{u_1,......,u_p\} = || w_1||..........|| w_p||.$

We start by choosing an initial condition x_0 and a n x n matrix $U_0 =[u_1^0,......, u_n^0]$

Using the Gram-Schmidt procedure we calculate the corresponding matrix of orthonormal vectors $V_0 =[v_1^0,......, v_n^0]$ and integrate the variational equation from $\{x_0,V_0\}$ for a short interval T,to obtain

$x_1=f^T(x_0)$ and

$U_1 =[u_1^1,......, u_n^1]=D_{x0}\, f^T(U_0)= \Phi_T(x_0) \, . \, [u_1^0,......, u_n^0]$

Again we calculate the orthonormalized version of U_1 and integrate the equation from $[x_1,V_1]$ for T seconds to obtain x_2 and U_2.We repeat this integration - orthonormalization procedure k times.

During the k-th step,the p dimensional volume Vol^p defined earlier increases by a factor of $|| w_1^k||..........|| w_p^k||$,where $\{ w_1^k,......, w_p^k\}$ is the set of orthogonal vectors calculated from U_k.

Thus,

$\lambda^p(x_0,U_0) = \lim_{k\to Infinity} \left(\frac{1}{k\,T}\right) \sum_{i=1}^{k} ln\ (|| w_1^i||..........|| w_p^i||)$

Subtracting λ^{p-1} from λ^p and using the fact that $\lambda^p (x_0 , U_0)= \lambda_1+......+ \lambda_p$,

we obtain the p-th LCE of order 1 :

$\lambda_p = \lim_{k\to Infinity} \left(\frac{1}{k\,T}\right) \sum_{i=1}^{k} ln\ (|| w_p^i||)$

This method gives us an easy way to calculate the entire Lyapunov Spectrum.

Thus choosing a suitable value of T,we calculate the quantities

$\lambda_1 = \left(\frac{1}{K\,T}\right) \sum_{i=1}^{K} ln\ (|| w_1^i||)$, ... , $\lambda_n = \left(\frac{1}{K\,T}\right) \sum_{i=1}^{K} ln\ (|| w_n^i||)$

until they show convergence/until a maximum iteration count is reached.

This procedure has been implemented in Mathematica

as part of the **Module LCEsCnum**

```
LCEsCnum[numf_, numg_, initcond_, T_, K_, trans_, stepsize_, opts___Rule, kp_, ϵ_] :=
Module[{opt, n, a, b, x, x0, Phi, DPhi, V0, s,
     W, norms, lcelist, lces},
  opt = LCEsPlot /. {opts} /. {LCEsPlot -> False};
  n = Length[initcond];
  x = Array[a, n];
  Phi = Transpose[Array[b, {n, n}]];
   (*Simulate upto the end of the transient period*)
  x0 = Nest[RKStepnum[numf, x, #, N[stepsize], kp, ϵ] &, N[initcond],
    Round[N[trans / stepsize]] ];;
  V0 = IdentityMatrix[n];
   (*Now start with x0,V0 as initial conditions and simulate for K steps*)
  s = {};
  Do[
   {x0, V0} = IntVarEqnum[numg, {x, Phi}, {x0, V0}, {T, stepsize}, kp, ϵ];
   W = GramSchm[V0];
   norms = Map[Norm, W];
   s = Append[s, norms];
   V0 = W / norms,
   {K}];
  lces = Sort[Apply[Plus, Log[s]] / (T K), Greater];
  Return[lces];

 ]
```

The above module uses a few other modules such as

1. IntVarEqnum : Integrates the variational equation of the combined system

```
IntVarEqnum[numg_, y_, y0_,
   {t1_, dt_}, kp_, ϵ_] :=
Module[{ yt},
  yt = Nest[ RKStepnum2[numg, y, #, N[dt], kp, ϵ] &, N[y0],
     Round[N[t1 / dt]] ];

 Return[yt];
  ]
```

2. RKStepnum2 : Performs numerical integration of the combined state transition function G by the Runge-Kutta Method

```
(*Module to perform Numerical Integration of G by the Runge-
 Kutta method *)
RKStepnum2[numg_, y_, y0_, dt_, kp_, ϵ_] :=
Module[{ k1, k2, k3, k4 },

  k1 =
   dt
    N[ numg[{(y[[1]] /. Thread[y[[1]] -> y0[[1]]]),
       (y[[2]] /. Thread[Flatten[y[[2]]] → Flatten[ y0[[2]]]])},
      kp, ϵ ]];

  k2 =
   dt
    N[ numg[{(y[[1]] /. Thread[y[[1]] -> y0[[1]] + k1[[1]] / 2]),
       (y[[2]] /. Thread[Flatten[y[[2]]] ->
           Flatten[y0[[2]] + k1[[2]] / 2]])}, kp, ϵ ]];
  k3 =
   dt
    N[ numg[{(y[[1]] /. Thread[y[[1]] -> y0[[1]] + k2[[1]] / 2]),
       (y[[2]] /. Thread[Flatten[y[[2]]] ->
           Flatten[y0[[2]] + k2[[2]] / 2]])}, kp, ϵ ]];

  k4 =
   dt
    N[ numg[{(y[[1]] /. Thread[y[[1]] -> y0[[1]] + k3[[1]]]),
       (y[[2]] /. Thread[Flatten[y[[2]]]  ->
           Flatten[y0[[2]] + k3[[2]]]])}, kp, ϵ ]];

  y0 + (k1 + 2 k2 + 2 k3 + k4) / 6 ]
```

3.

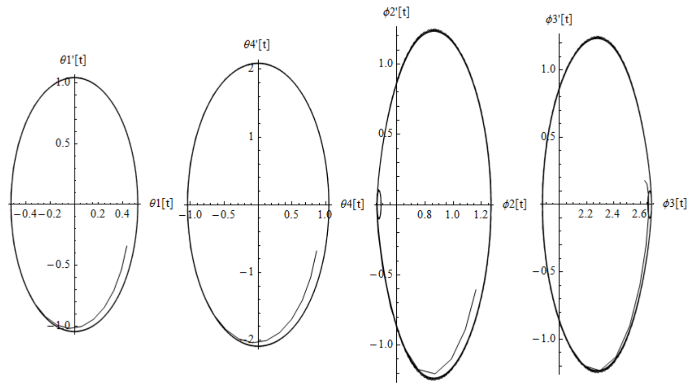
```
 (* Project *)
 prj[v1_, v2_] := (v2.v1) v2 / (v2.v2)
 multiprj[v1_, vecs_] :=
   Plus @@ Map[prj[v1, #] &, vecs]
 (* Gram-Schmidt Orthogonalization *)
 GramSchm[vecs_] :=
   Fold[Join[#1, {#2 - multiprj[#2, #1]}] &, {}, vecs];
```

Observation

Phase Space Plots of the system were obtained for different values of kp,ϵ by simulating the system for a total time of 20s with a transient period of 1s and step size of 0.1s. While tracking a periodic trajectory, a well behaved system follows closed curves in the Phase Space.Whereas for Chaotic systems,the phase space plots fill an entire region.

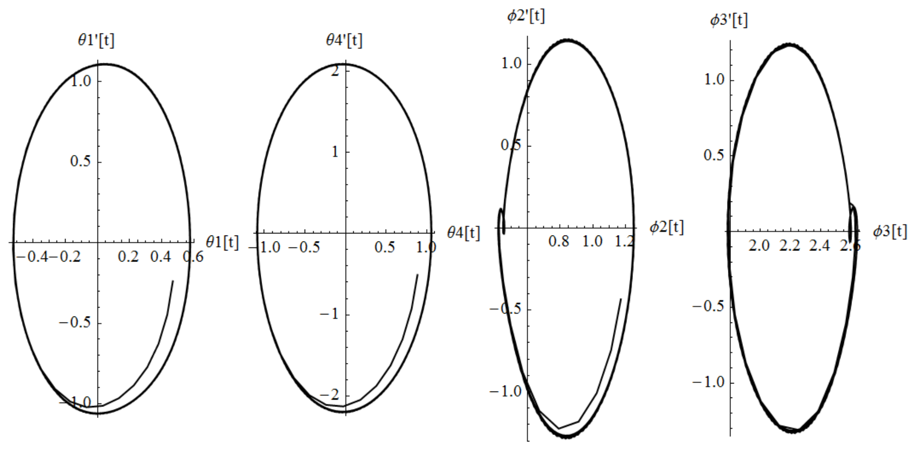
```
(*For kp=10,ϵ=0.
  This is the ideal case where the nonlinearities in the model are exactly cancelled and we have set of uncoupled linear
  ODEs.In this case we expect the system to track the reference trajectory well.*)

PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0]
```



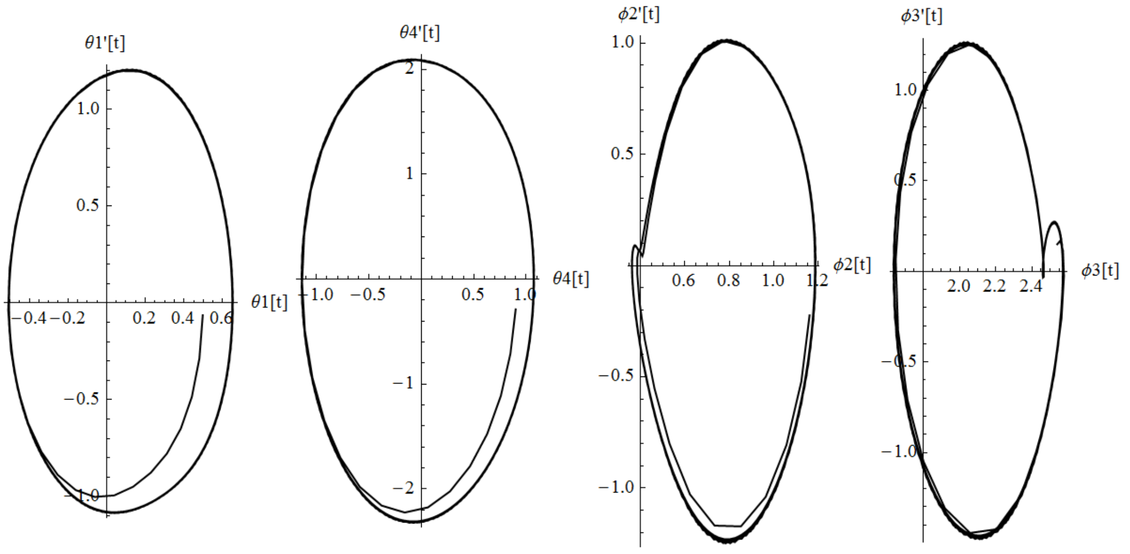
```
(*We observe that in this case each of the phase space plots are closed curves within numerical errors as expected.*)
```

```
(*Now we study the behaviour of the system for underestimations of the model i.e ϵ<0 *)
(*For kp=10,ϵ=-0.1*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.1]
```



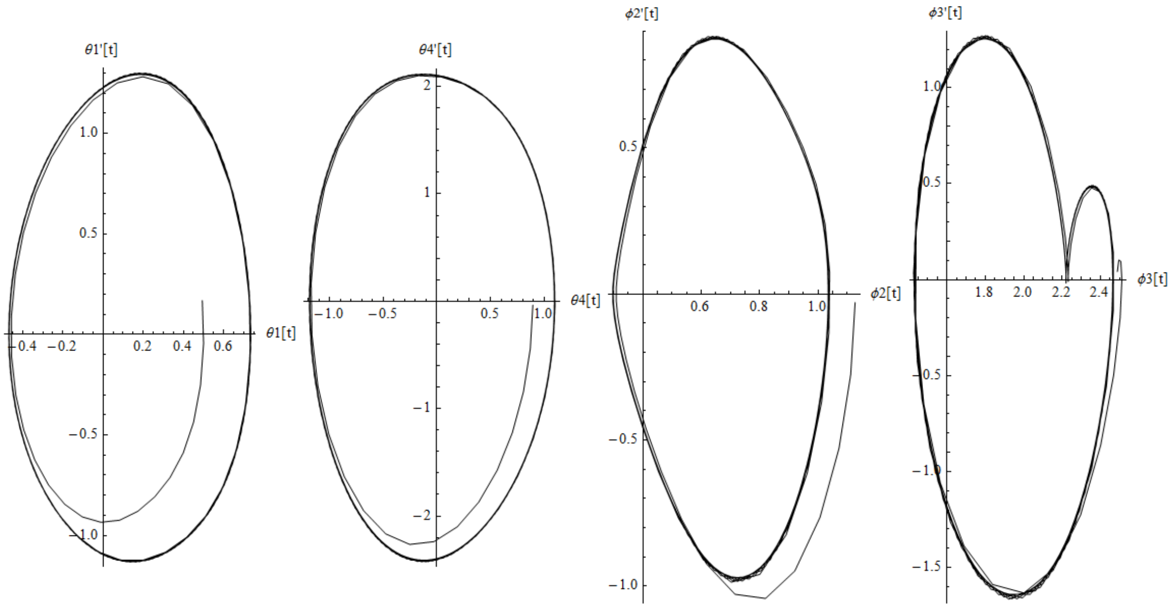
```
(*We observe that the phase space plots are still closed curves.*)
```

```
(*For kp=10,ϵ=-0.2*)
PhaseSpaceCnum[numF, Qinit, 20, 1,  0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.2]
```



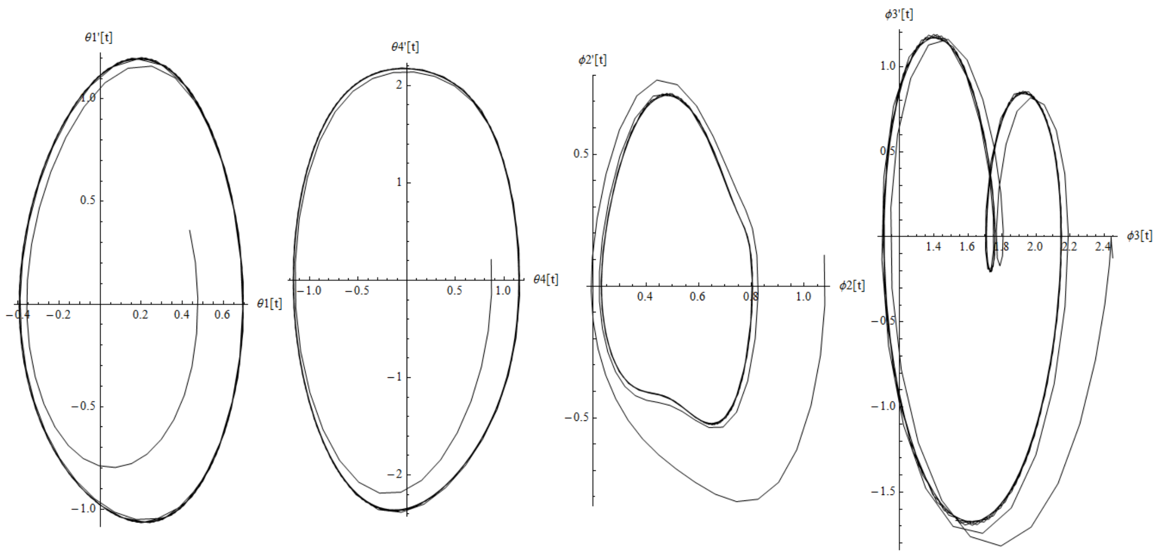
```
(*We observe that the phase space plots are still closed curves.*)
```

```
(*For kp=10,ϵ=-0.3*)
PhaseSpaceCnum[numF, Qinit, 20, 1,  0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.3]
```



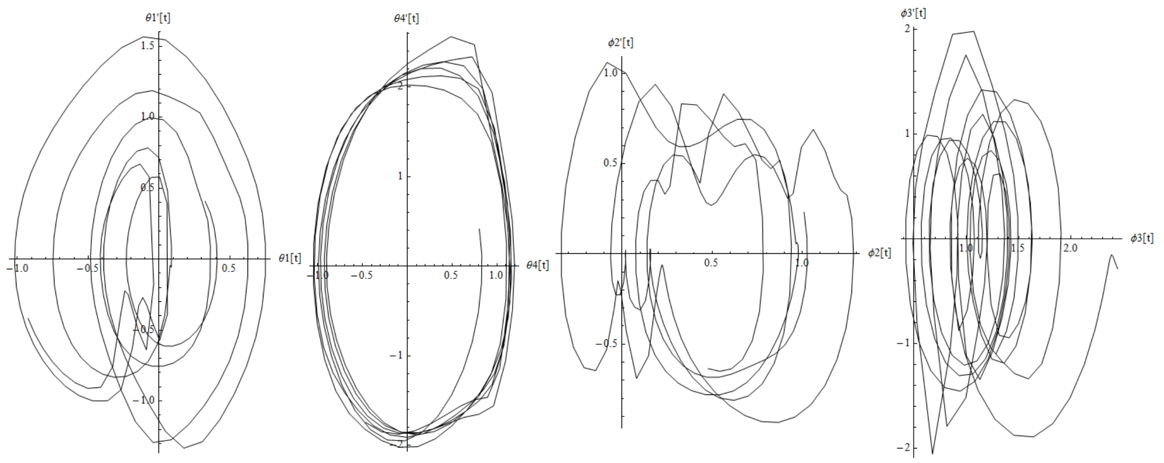
```
(*We observe that the phase space plots are still closed curves but are getting distorted.*)
```

```
(*For kp=10,ϵ=-0.4*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.4]
```



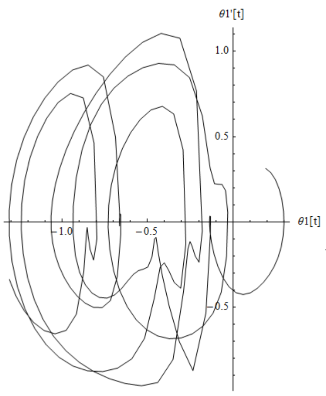
```
(*We observe that the phase space plots are getting more distorted .*)
```

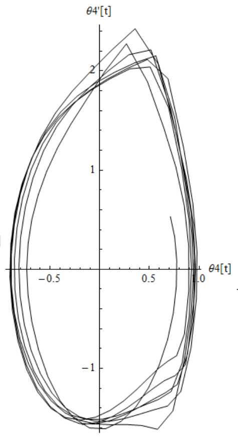
```
(*For kp=10,ϵ=-0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.5]
```

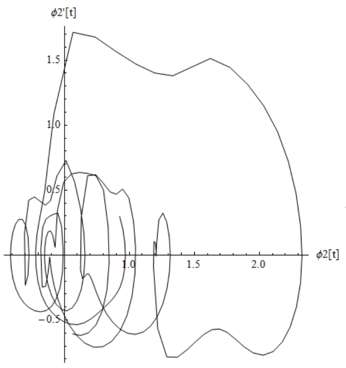


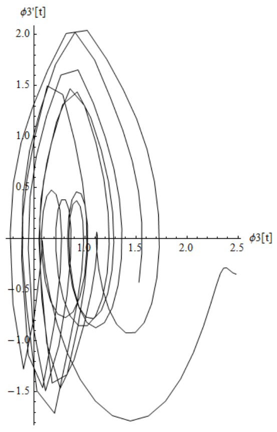
```
(*At this point we can see that the phase space plots are very different from the closed curves we want.
  The phase space plots can be seen to fill an entire region of the phase space(which is called a strange attractor).
   This is one of the hallmarks of Chaos! *)
```

```
(*For kp=10,ϵ=-0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.6]
```



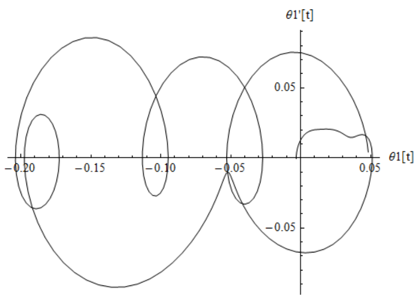


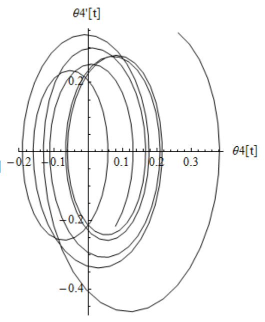


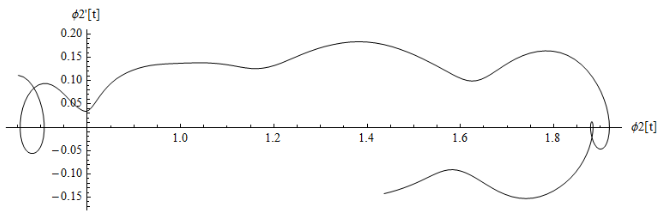


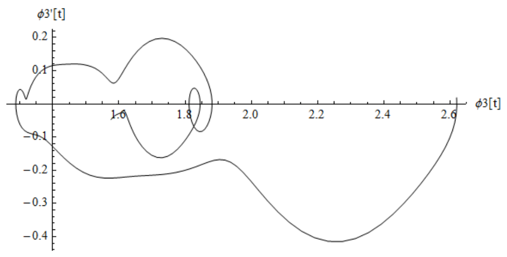
```
(*Again we observe that the Phase Space Plots fill an entire region*)
```

```
(*For kp=10,ϵ=-0.9.
*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.9]
```



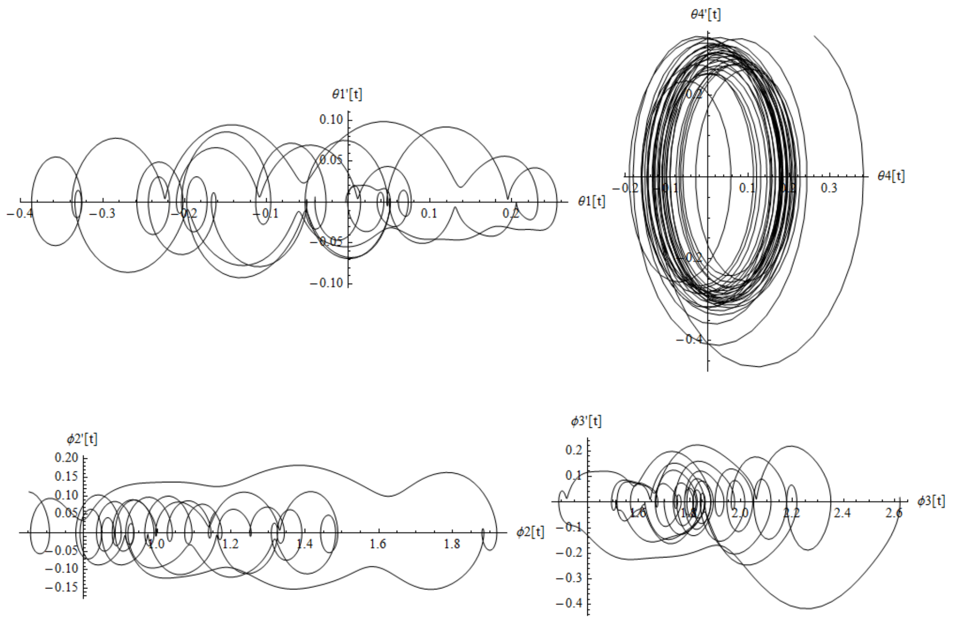




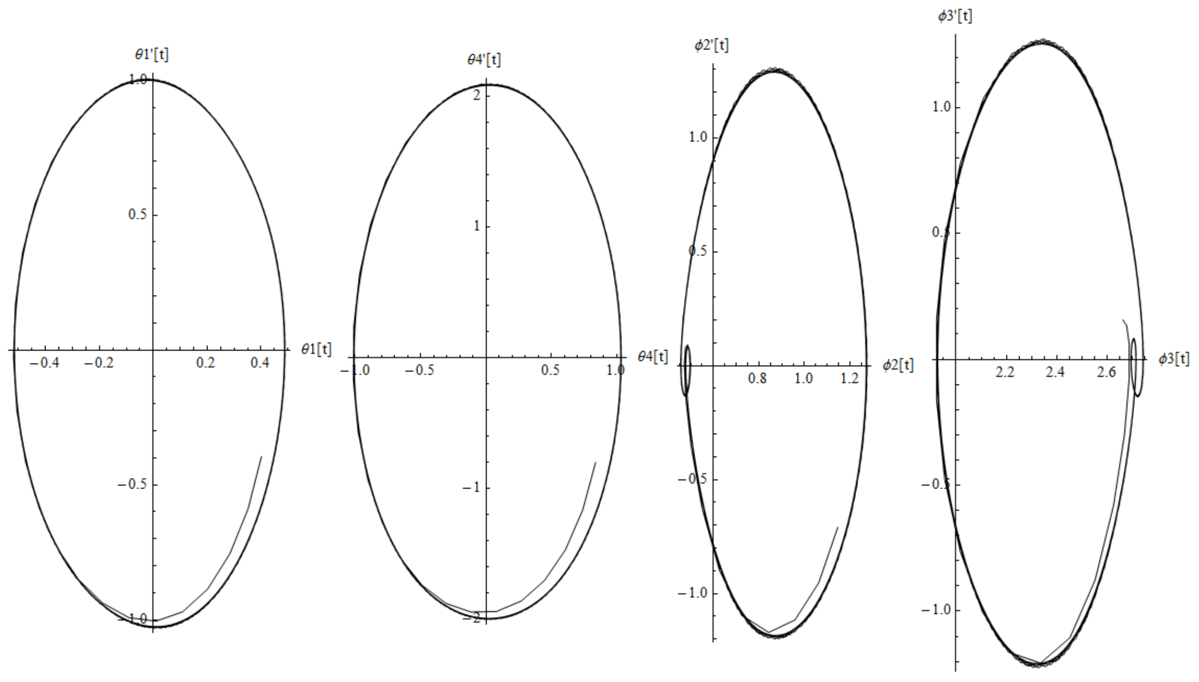


```
(*Again we observe that the Phase Space Plots fill an entire region*)
```

```
(*For the same values kp=10,ϵ=-0.9 we increase the simulation time to 100s to obtain a clearer picture*)
PhaseSpaceCnum[numF, Qinit, 100, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, -0.9]
```

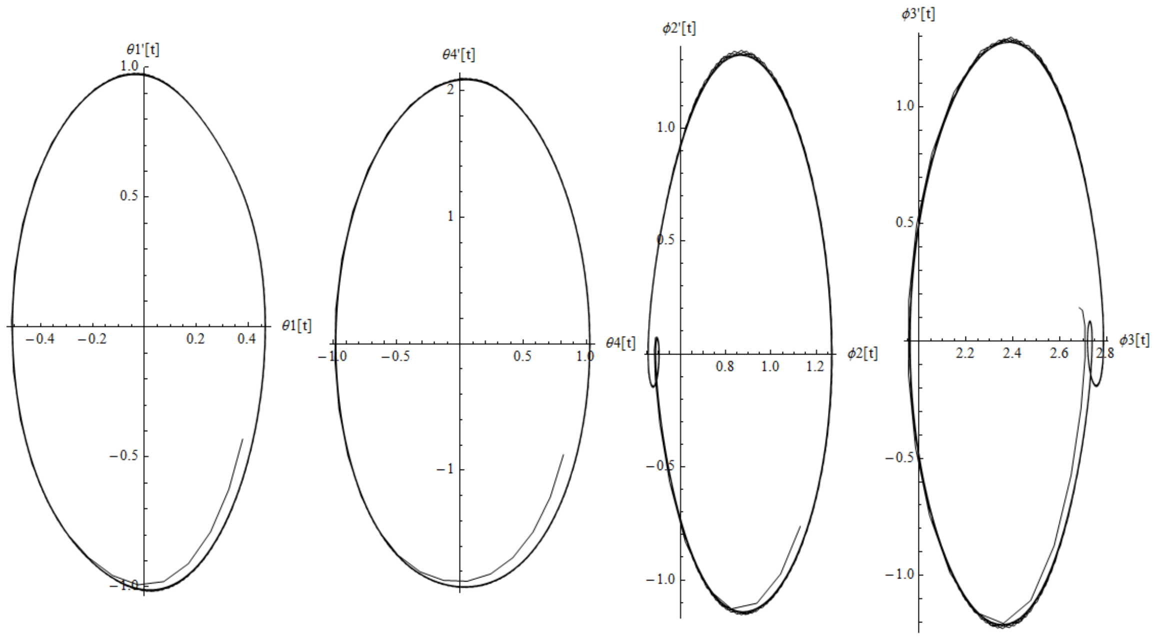


```
(*Next we study the behaviour of the system for overestimations of the model i.e ϵ>0 *)
(*For kp=10,ϵ=0.1*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0.1]
```



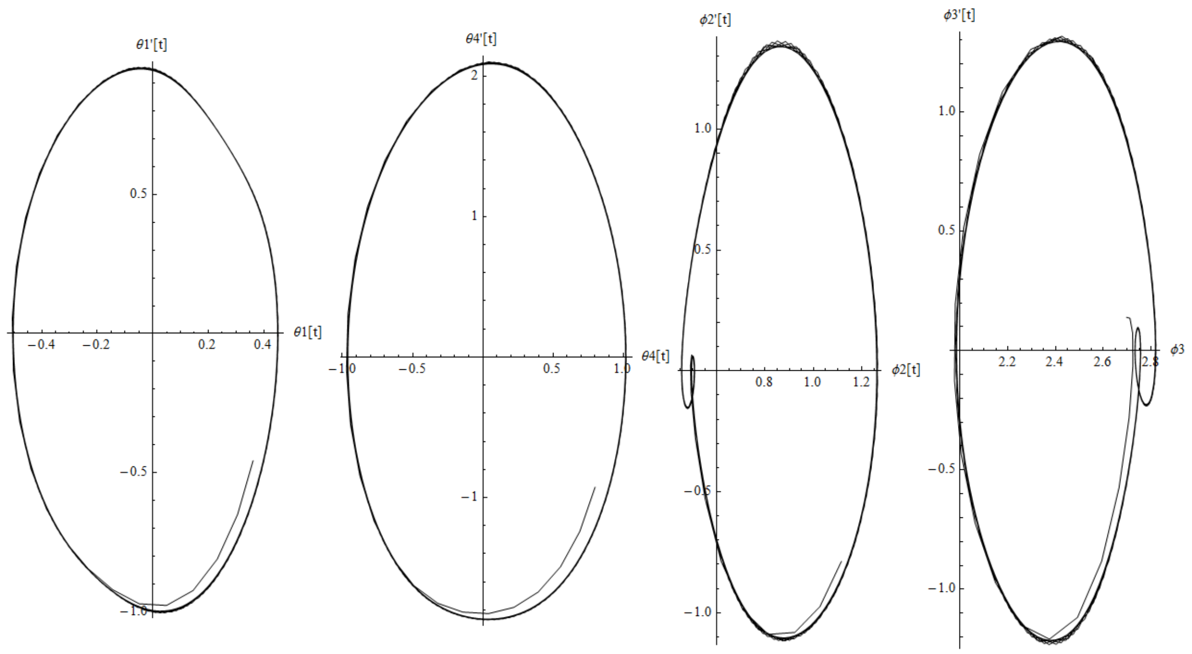
```
(*We observe that the phase space plots are closed curves.*)
```

```
(*For kp=10,ϵ=0.2*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0.2]
```



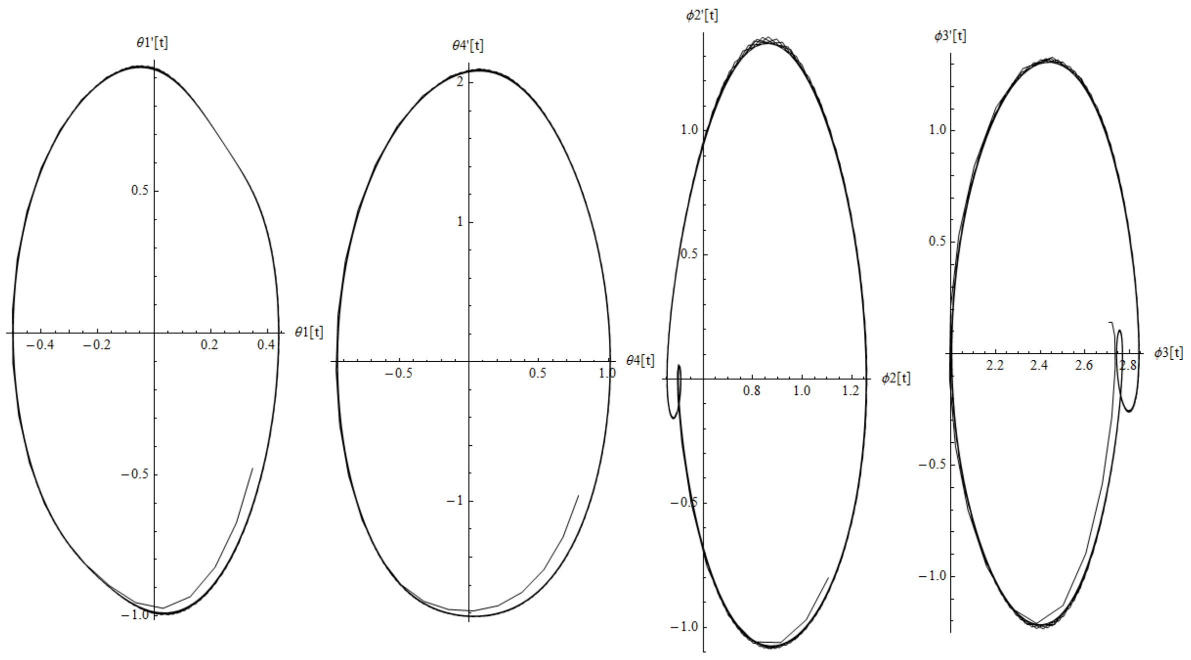
```
(*We observe that the phase space plots are still closed curves.*)

(*For kp=10,ϵ=0.3*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0.3]
```



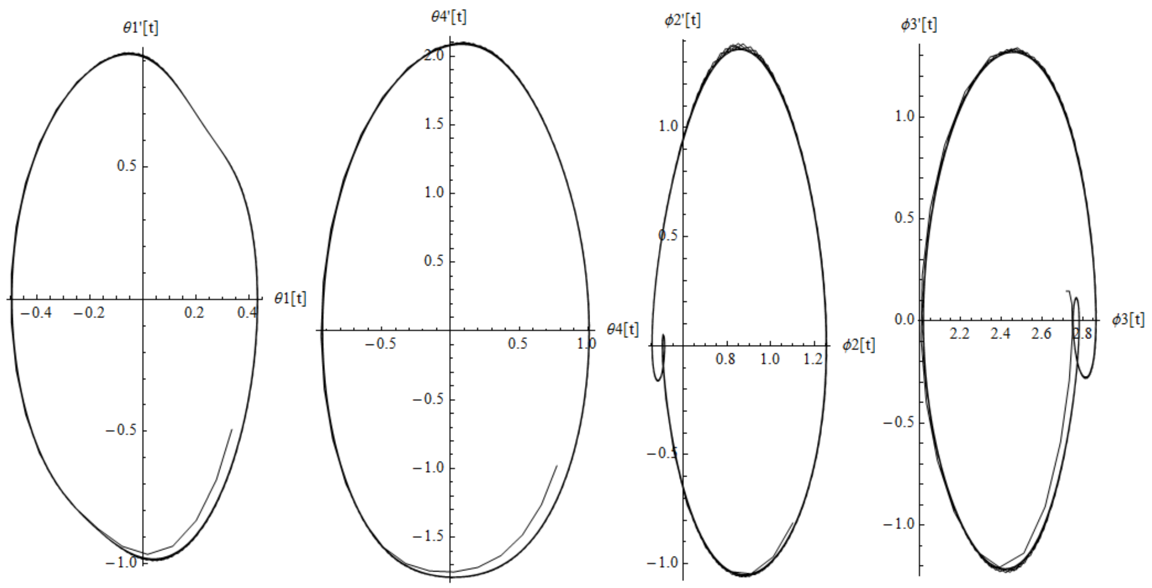
```
(*We observe that the phase space plots are still closed curves.*)
```

```
(*For kp=10,ϵ=0.4*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0.4]
```



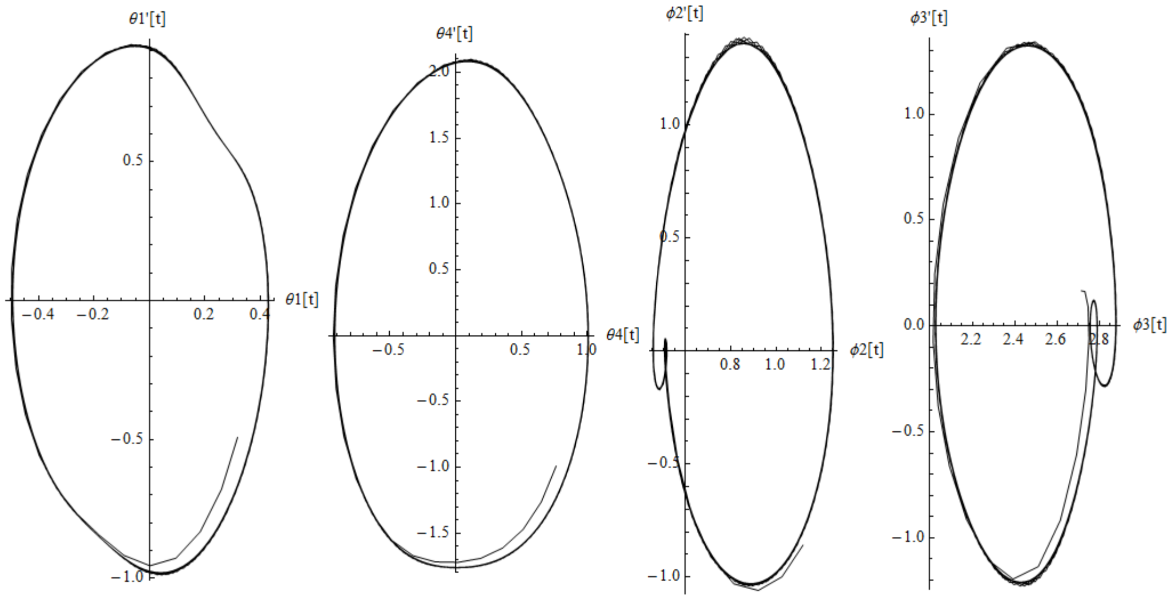
```
(*We observe that the phase space plots are still closed curves.*)
```

```
(*For kp=10,ϵ=0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0.5]
```



```
(*We observe that the phase space plots are still closed curves.*)
```

```
(*For kp=10,ϵ=0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 10, 0.6]
```

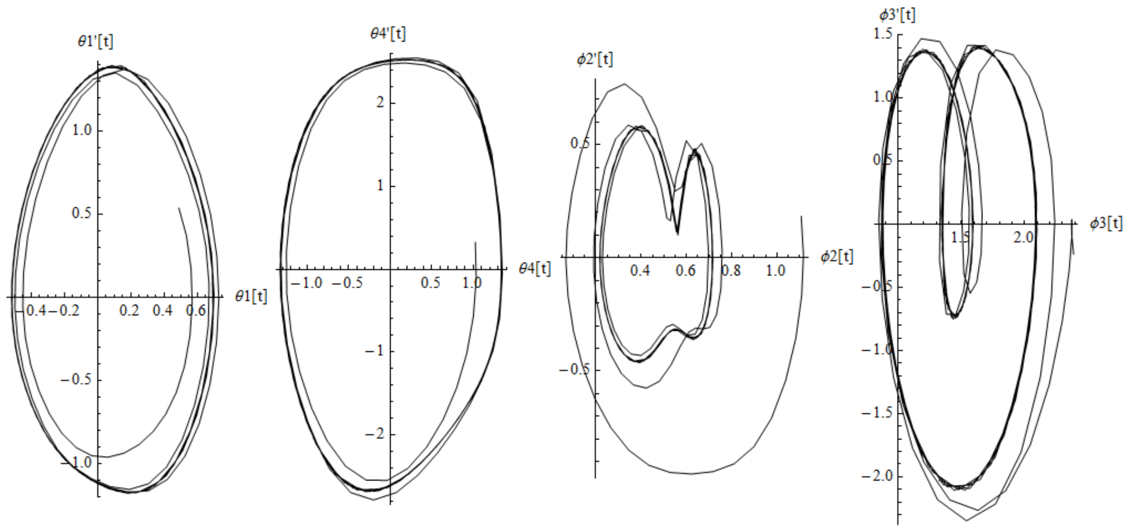


```
(*We observe that the phase space plots are still closed curves!.*)
```

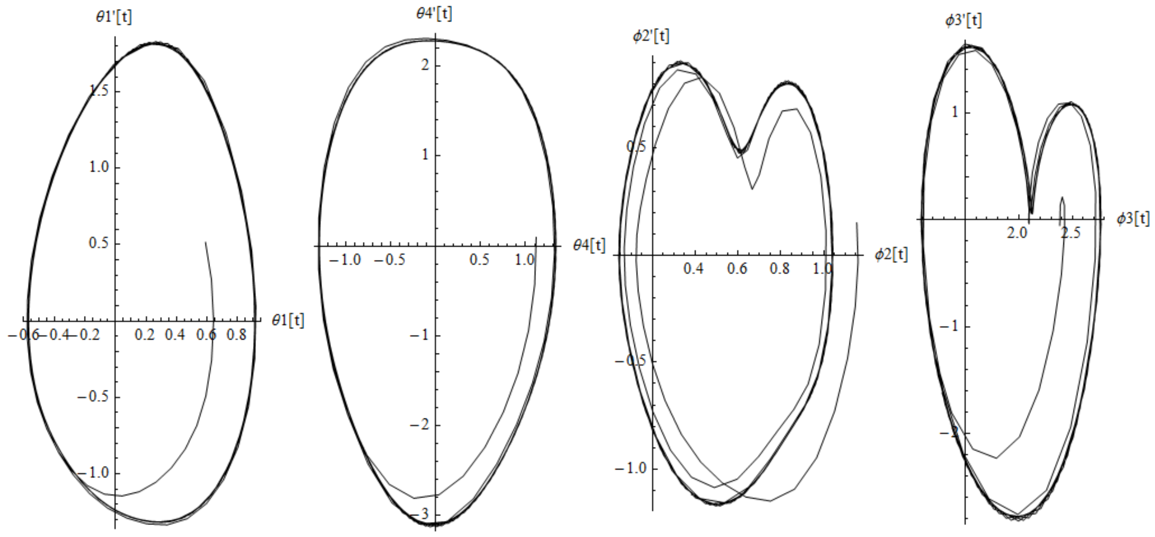
Thus we conclude that Chaotic Dynamics is more easily seen for underestimations of the model compared to overestimation.For this choice of kp, we were unable to observe chaos no matter how large we make the overestimation.Also greater the extent of underestimation,the more chaotic the system's dynamics are.

```
(*Next we turn our attention to kp.|
  For the same value of model mismatch ϵ,we try to vary kp and observe the effect*)
```

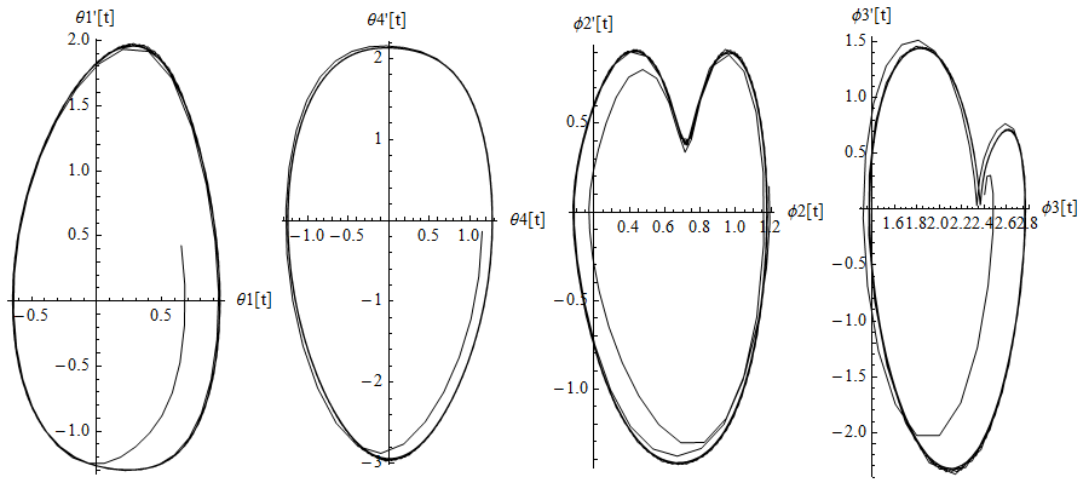
```
(*For kp=20,ϵ= -0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 20, -0.5]
```



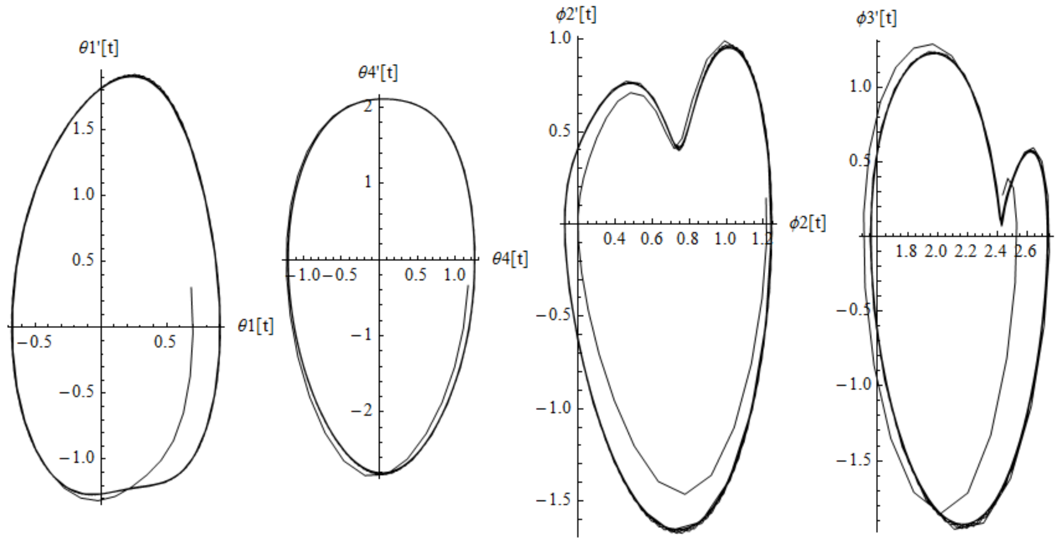
```
(*For kp=30,ϵ= -0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 30, -0.5]
```



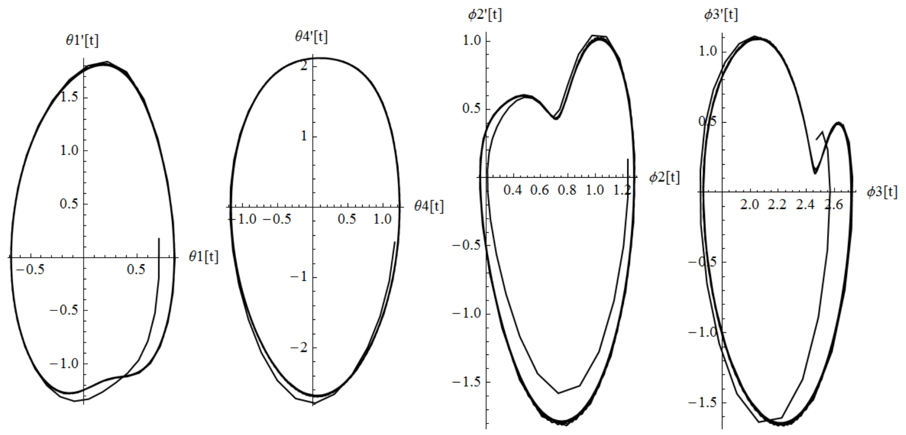
```
(*For kp=40,ϵ= -0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 40, -0.5]
```



```
(*For kp=50,ϵ= -0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 50, -0.5]
```

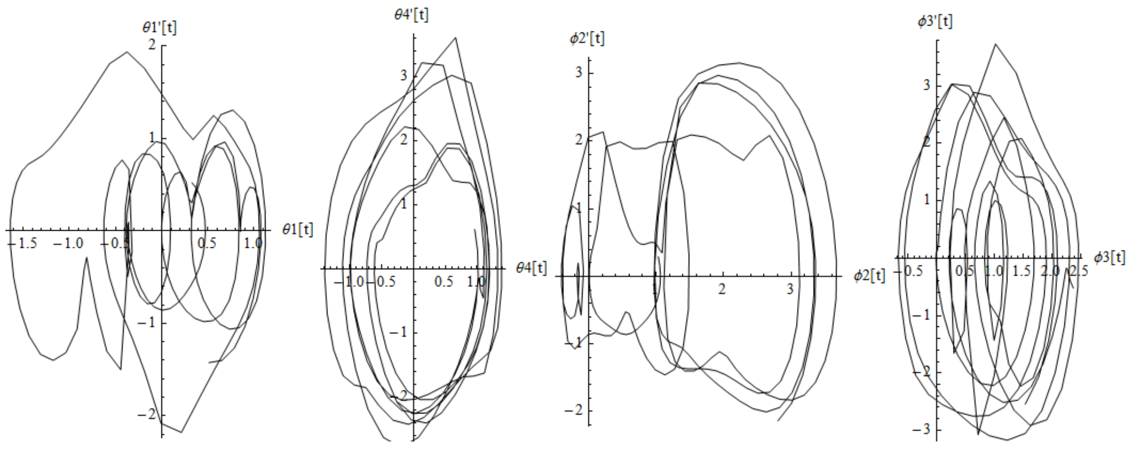


```
(*For kp=60,ϵ= -0.5*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 60, -0.5]
```

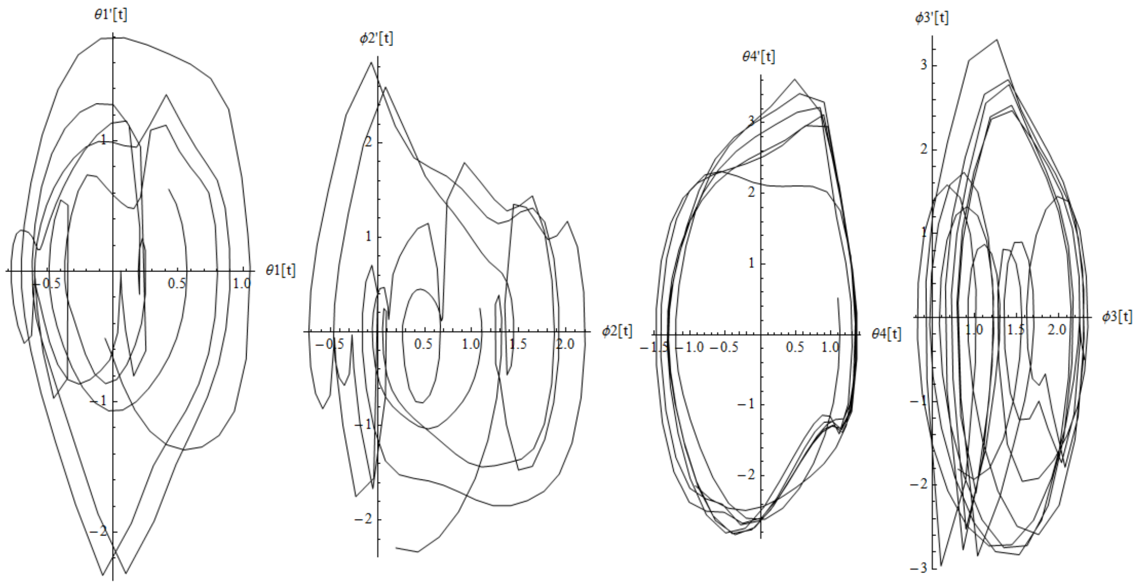


```
(*So far it appears that for the same model mismatch ϵ,
increasing kp seems to make the system less chaotic or more regular.
  We shall now increase the underestimation in the model i.e make ϵ more negative so that
 we can observe trends more clearly*)
```

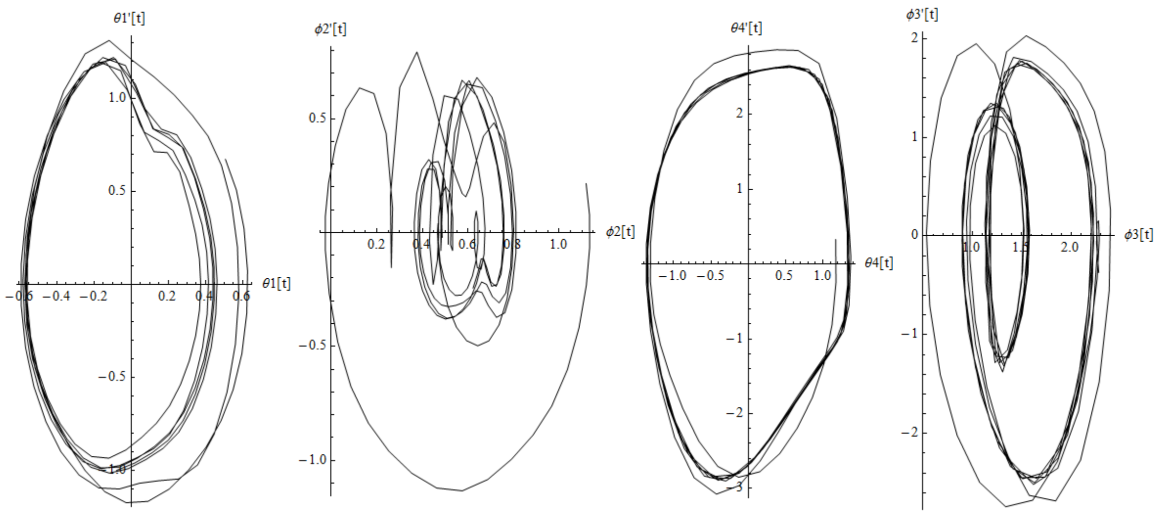
```
(*For kp=20,ϵ= -0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 20, -0.6]
```



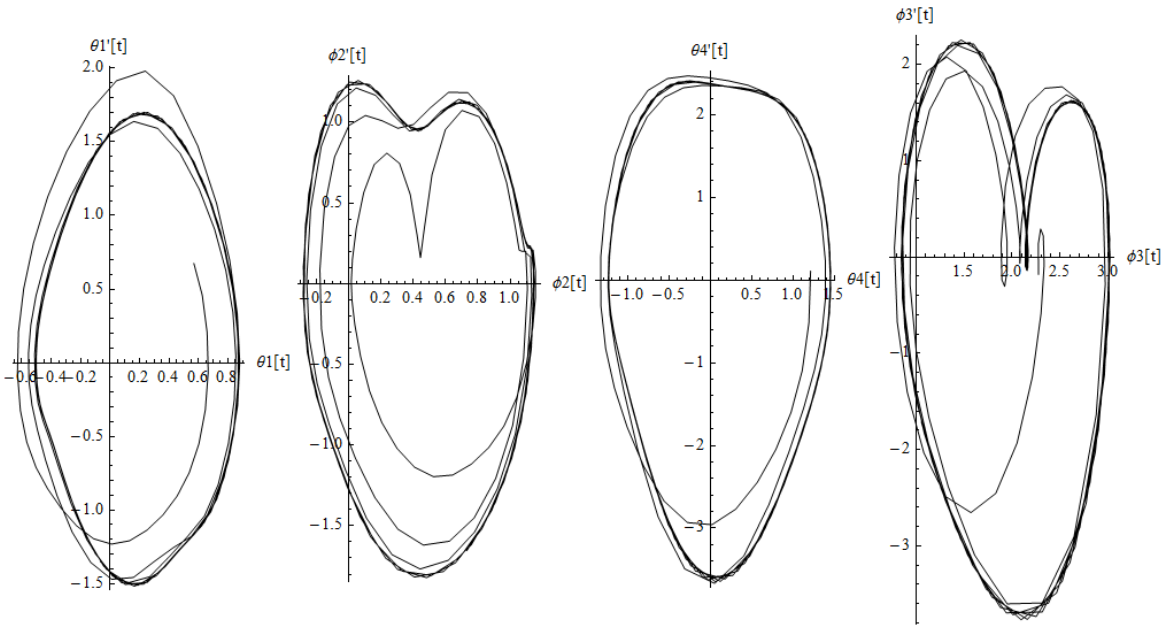
```
(*For kp=30,ϵ= -0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 30, -0.6]
```



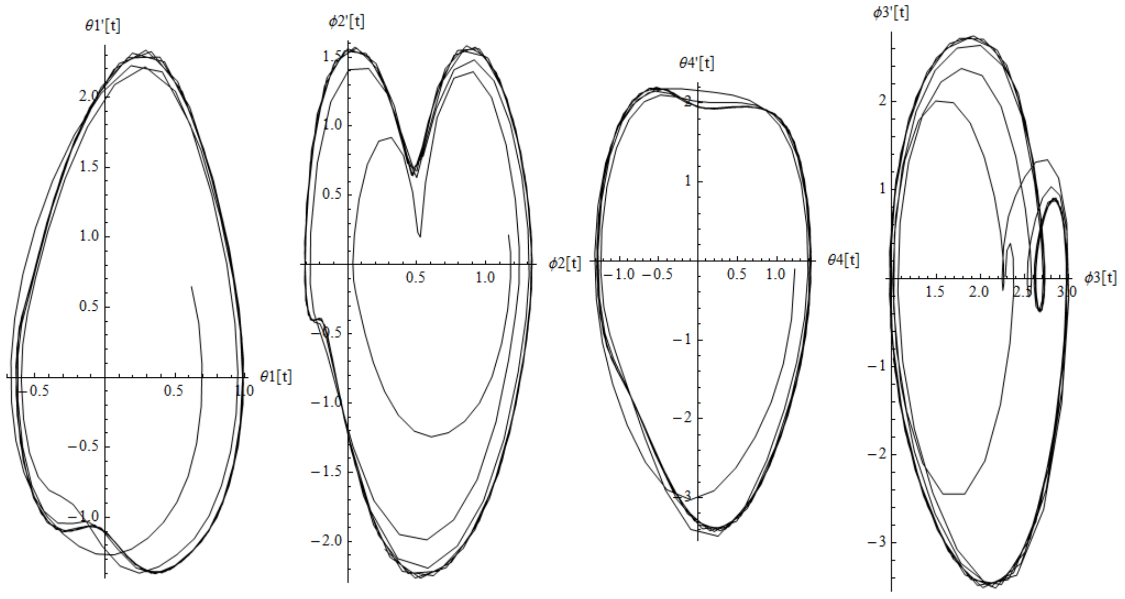
```
(*For kp=40,ϵ= -0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 40, -0.6]
```



```
(*For kp=50,ϵ= -0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 50, -0.6]
```



```
(*For kp=60,ϵ= -0.6*)
PhaseSpaceCnum[numF, Qinit, 20, 1, 0.1, {{1, 5}, {2, 6}, {3, 7}, {4, 8}}, 60, -0.6]
```



We thus observe that chaotic motions are more easily observed for lower values of kp than for higher values of kp for the same model mismatch ϵ.

Intutively we can expect the same result,as the higher the kp the better the system will follow the reference trajectory,hence the trajectory will be more periodic and less chaotic.

Now we attempt to quantitatively characterize chaos through the **Lyapunov Exponents**.

We shall calculate the entire Lyapunov Spectrum for different values of kp, ϵ and observe if any of the Lyapunov exponents are positive which if it happens is an indication of chaotic dynamics

The simulation conditions used are :

a)Step Size = 0.1

b)Length of one calculation cycle,T=1s

c)No of calculation cycles,K=10

We shall vary kp between 10 and 30 in steps of 10 and ϵ between -0.9 and 0.6 in steps of 0.3.

i.e kp=10,20,30 and ϵ =-0.9,-0.6,-0.3,0,0.3,0.6.

Note:The decision to vary kp, ϵ in such coarse steps,and kp only between 10 and 30,was taken because of the large computation time required to calculate even one complete set of Lyapunov Exponents.

It is observed that Positive Lyapunov exponents and hence Chaotic Dynamics is seen only for ϵ =-0.9,-0.6 among ϵ =-0.9,-0.6,-0.3,0,0.3,0.6.

For kp=10, ϵ =-0.9 , λ = 0.14369231643053867

kp=10, ϵ =-0.6 , λ = 0.12449954140888418

kp=20, ϵ =-0.9 , λ = 10.209688845627625

kp=20, ϵ =-0.6 , λ = 0.13014780626637962

kp=30, ϵ =-0.9 , λ = 13.746366414238118

kp=30, ϵ =-0.6 , λ = 0.05596911487555292

We observe that the results we obtain from the calculation of the Lyapunov exponents is consistent with our previous observations from the phase space plots.

The same values of ϵ= -0.9,-0.6 also gave us Phase Space Plots which filled an entire region.

Conclusions

Thus we have managed to observe and also quantify Chaotic Dynamics in Robotic Control Equations.It is seen that Chaos occurs only for certain values of Controller Gains and Model Mismatch.

Chaotic Dynamics is more easily seen for underestimations of the model compared to overestimation.And greater the extent of underestimation,the more chaotic the system’s dynamics are.

Chaotic motions are more easily observed for lower values of kp than for higher values of kp for the same model mismatch ϵ.

This study apart from being of mathematical interest, also helps in obtaining conditions for better trajectory tracking in feedback controlled Robots.

This study also suggests that robustness results in Robotic Control Literature need a fresh look since the possibility of Chaotic Motions have not been considered by researchers so far.

References

1. Chaos in Robot Control Equations,S.Lankalapalli and Ashitava Ghosal,April 1996

2.Nonlinear Dynamics and Chaotic Motions in Feedback Controlled Two and Three Degree of Freedom Robots,A.S.Ravishankarm,Ashitava Ghosal

3.Numerical Calculation of Lyapunov Exponents,Marco Sandri,University of Italy