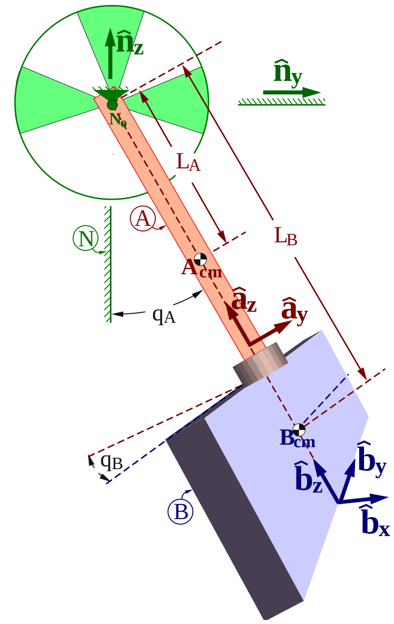
Motivating example (MIPSI): Babyboot



Model

The figure to the right is a schematic representation of a swinging babyboot attached by a shoelace to a rigid support. The mechanical model of the babyboot consists of a thin uniform rod A attached to a fixed support N by a revolute joint, and a uniform plate B connected to A with a second revolute joint so that B can rotate freely about A's axis.
Note: The revolute joints' axes are ***perpendicular***, not parallel.

- **Bodies**: The rod and plate are rigid (inflexible/undeformable).
- **Connections**: The revolute joints are ideal (massless, frictionless, with no slop or flexibility).
- **Force**: Earth's gravity is uniform and constant. Other contact forces (e.g., air resistance, solar/light pressure) and distance forces (e.g., electromagnetic, other gravitational) are negligible.
- **Newtonian reference frame**: Earth

Identifiers

Right-handed sets of unit vectors $\hat{\mathbf{n}}_x$, $\hat{\mathbf{n}}_y$, $\hat{\mathbf{n}}_z$; $\hat{\mathbf{a}}_x$, $\hat{\mathbf{a}}_y$, $\hat{\mathbf{a}}_z$; and $\hat{\mathbf{b}}_x$, $\hat{\mathbf{b}}_y$, $\hat{\mathbf{b}}_z$ are fixed in N, A, and B, respectively, with $\hat{\mathbf{n}}_x = \hat{\mathbf{a}}_x$ parallel to the revolute axis joining A to N, $\hat{\mathbf{n}}_z$ vertically upward, $\hat{\mathbf{a}}_z = \hat{\mathbf{b}}_z$ parallel to the rod's long axis (and the revolute axis joining B to A), and $\hat{\mathbf{b}}_z$ perpendicular to plate B.

Quantity	Symbol	Type	Value
Earth's gravitational constant	g	Constant	9.81 m/s^2
Distance between N_o and A_{cm}	L_A	Constant	7.5 cm
Distance between N_o and B_{cm}	L_B	Constant	20 cm
Mass of A	m^A	Constant	0.01 kg
Mass of B	m^B	Constant	0.1 kg
A's moment of inertia about A_{cm} for $\hat{\mathbf{a}}_x$	I^A	Constant	0.05 kg∗cm^2
B's moment of inertia about B_{cm} for $\hat{\mathbf{b}}_x$	I^B_x	Constant	2.5 kg∗cm^2
B's moment of inertia about B_{cm} for $\hat{\mathbf{b}}_y$	I^B_y	Constant	0.5 kg∗cm^2
B's moment of inertia about B_{cm} for $\hat{\mathbf{b}}_z$	I^B_z	Constant	2.0 kg∗cm^2
Angle from $\hat{\mathbf{n}}_z$ to $\hat{\mathbf{a}}_z$ with $+\hat{\mathbf{n}}_x$ sense	q_A	Dependent variable	Varies
Angle from $\hat{\mathbf{a}}_y$ to $\hat{\mathbf{b}}_y$ with $+\hat{\mathbf{a}}_z$ sense	q_B	Dependent variable	Varies
Time	t	Independent variable	Varies

Physics

Physics from www.MotionGenesis.com ⇒ Get Started ⇒ Chaotic Pendulum (Babyboot).

The ODEs (ordinary differential equations) governing the motion of this mechanical system are[11]

$$\ddot{q}_A = \frac{2\,\dot{q}_A\,\dot{q}_B\,\sin(q_B)\,\cos(q_B)\,\left(\mathrm{I}^B_x - \mathrm{I}^B_y\right) \;-\; \left(m^A\,L_A + m^B\,L_B\right)g\sin(q_A)}{\mathrm{I}^A \;+\; m^A\,L_A^2 \;+\; m^B\,L_B^2 \;+\; \mathrm{I}^B_x\cos^2(q_B) \;+\; \mathrm{I}^B_y\sin^2(q_B)}$$

$$\ddot{q}_B = \frac{-\dot{q}_A{}^2\,\sin(q_B)\,\cos(q_B)\,(\mathrm{I}^B_x - \mathrm{I}^B_y)}{\mathrm{I}^B_z}$$

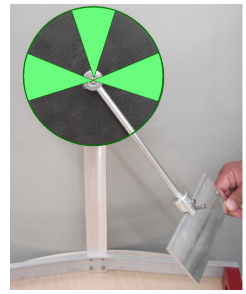
[11] Four methods for forming equations of motion are: ***Free-body diagrams*** of A and B (which is inefficient as it introduces up to 10 unknown force/torque measures); D'Alembert's method (***road maps*** of Section 22.6) which efficiently forms the two equations shown for $\ddot{q}_A$ and $\ddot{q}_B$ (but require a clever selection of systems, points, and unit vectors); ***Lagrange's equations*** (an energy-based method that automates D'Alembert's cleverness); ***Kane's equations*** (a modern efficient blend of D'Alembert and Lagrange).

Simplify and solve

Computers has revolutionized the solution of differential equations. There are a many numerical algorithms for solving nonlinear, coupled, variable coefficient, ODEs (ordinary differential equations) including Euler's method, predictor-corrector, Runga-Kutta, etc. In addition, there are many programs (MATLAB®, **M**otion**G**enesis, WolframAlpha, etc.) that make it easy to solve ODEs.

Computer (numerical) solution of ODEs with MotionGenesis (with plotting)

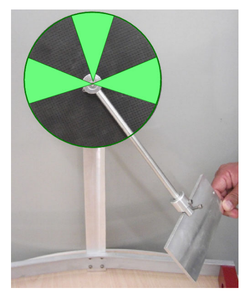
```
Variable  qA'', qB''      % Angles and first/second time-derivatives.
%--------------------------------------------------------------------
qA'' = 2*( 508.89*sin(qA) - sin(qB)*cos(qB)*qA'*qB' ) / (-21.556 + sin(qB)^2)
qB'' = -sin(qB)*cos(qB)*qA'^2
%--------------------------------------------------------------------
Input  tFinal = 10 sec, tStep = 0.02 sec, absError = 1.0E-07
Input  qA = 90 deg,  qB = 1.0 deg,  qA' = 0.0 rad/sec,  qB' = 0.0 rad/sec
OutputPlot  t sec,  qA degrees,  qB degrees
%--------------------------------------------------------------------
ODE() solveBabybootODE
Quit
```



Interpret

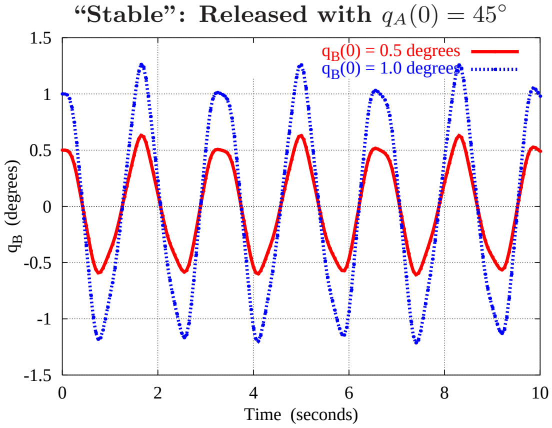
The solution to these differential equations reveals this simple system has strange, non-intuitive motion.[12] For certain initial values of q_A, the motion of plate B is well-behaved and "stable". Alternately, for other initial values of q_A, B's motion is **"chaotic"** – meaning that a small variation in the initial value of q_B or numerical integration inaccuracies lead to dramatically different results (these ODEs are used to test the accuracy of numerical integrators – the plots below required a numerical integrator error of absError $= 1 \times 10^{-7}$).

The following chart and figure to the right shows this system's regions of stability (**black**) and instability (**green**). Notice the **"chaotic"** plot below shows q_B is ***very*** sensitive to initial values. A 0.5° change in the initial value of $q_B(0)$ results in more than a 2000° difference in the value of $q_B(t = 10)$!

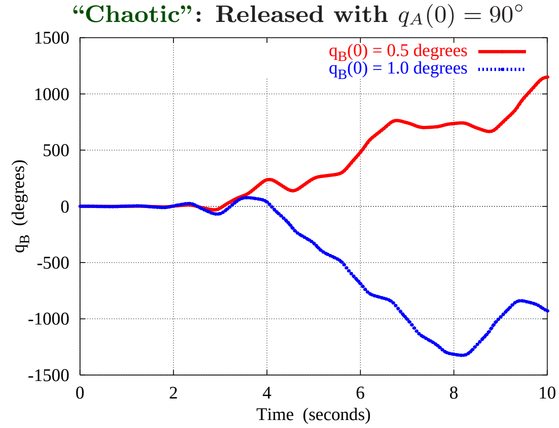


Initial value of q_A			Stability	
0°	$\leq q_A(0) \leq$	71.3°	**Stable**	**black**
71.4°	$\leq \mathbf{q_A(0)} \leq$	**111.77°**	**Unstable**	**green**
111.78°	$\leq q_A(0) \leq$	159.9°	**Stable**	**black**
160.0°	$\leq \mathbf{q_A(0)} \leq$	**180.0°**	**Unstable**	**green**

"Stable": Released with $q_A(0) = 45°$



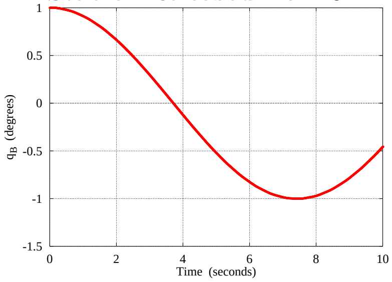
"Chaotic": Released with $q_A(0) = 90°$



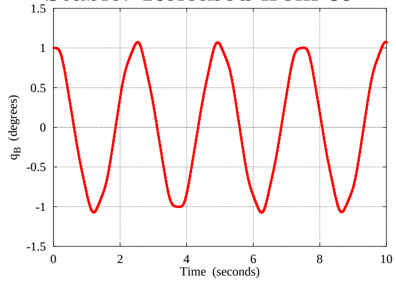
[12] More information about this problem is in "Mechanical Demonstration of Mathematical Stability and Instability", *International Journal of Engineering Education (Journal of Mechanical Engineering Education)*, Vol. 2, No. 4, 1974, pp. 45-47, by Thomas R. Kane. Or visit www.MotionGenesis.com ⇒ Get Started ⇒ Chaotic Pendulum (Babyboot).

Investigation of stability: More simulation results

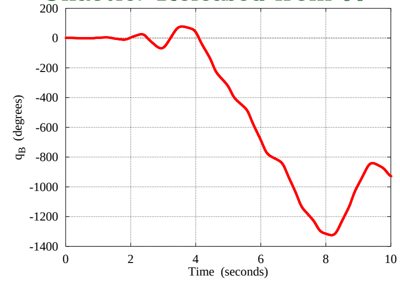
Stable: Released from 5°

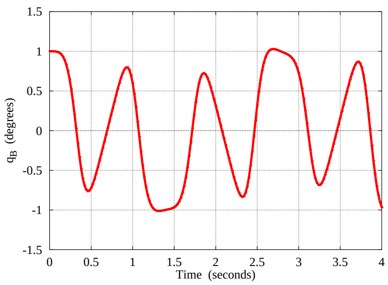


Stable: Released from 30°

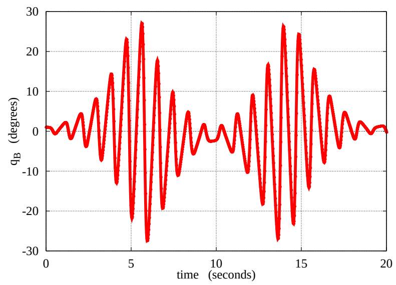


Chaotic: Released from 90°

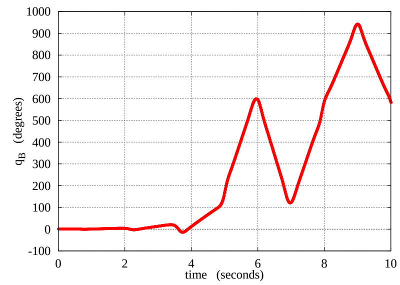




Stable: Released from 135°



Beat: Released from 158°



Chaotic: Released from 177°

Physics

Physics from www.MotionGenesis.com ⇒ Get Started ⇒ Chaotic Pendulum (Babyboot).

```
%     File:  BabybootWithDAlembertMethod.txt
%  Problem:  Analysis of 3D chaotic double pendulum
% Copyright (c) 2009 Motion Genesis LLC.  All rights reserved.
%--------------------------------------------------------------------
SetDigits( 5 )     % Number of digits displayed for numbers
%--------------------------------------------------------------------
NewtonianFrame  N
RigidBody  A       % Upper rod
RigidBody  B       % Lower plate
%--------------------------------------------------------------------
Variable   qA''         % Pendulum angle and its time-derivatives
Variable   qB''         % Plate angle and its time-derivative
Constant   LA = 7.5 cm  % Distance from pivot to A's mass center
Constant   LB = 20 cm   % Distance from pivot to B's mass center
A.SetMassInertia( mA =  10 grams,  IAx = 50 g*cm^2,  IAy,  IAz )
B.SetMassInertia( mB = 100 grams,  IBx = 2500 g*cm^2, IBy = 500 g*cm^2, IBz = 2000 g*cm^2 )
%--------------------------------------------------------------------
%       Rotational and translational kinematics
A.RotateX( N, qA )
B.RotateZ( A, qB )
Acm.Translate( No, -LA*Az> )
Bcm.Translate( No, -LB*Az> )
%--------------------------------------------------------------------
%       Add relevant forces
g> = -9.81*Nz>
System.AddForceGravity( g> )
%--------------------------------------------------------------------
%       Rotational equations of motion for B and A+B.
Dynamics[1] = Dot( B.GetDynamics(Bcm), Bz> )
Dynamics[2] = Dot( System(A,B).GetDynamics(No), Ax> )
Solve( Dynamics, qA'', qB'' )
%--------------------------------------------------------------------
%       Kinetic and potential energy
KE = System.GetKineticEnergy()
PE = System.GetForceGravityPotentialEnergy( g>, No )
Energy = KE + PE
%--------------------------------------------------------------------
%       Integration parameters and initial values.
Input  tFinal=10,  tStep=0.02,  absError=1.0E-07,  relError=1.0E-07
Input  qA = 90 deg,  qA' = 0.0 rad/sec,  qB = 1.0 deg,  qB' = 0.0 rad/sec
%--------------------------------------------------------------------
%       List output quantities and solve ODEs.
OutputPlot  t sec,  qA deg,  qB deg,  Energy N*m
ODE()  BabybootDAlembert
%--------------------------------------------------------------------
Save BabybootWithDAlembertMethod.all
Quit
```