HP-41 Module: CHAOS-101

Overview

As its name implies, this module includes a set of routines and applications dealing with a first-level approximation to chaotic systems. This intriguing subject has grown in popularity in the last couple of decades but wasn't much of a trending topic when the HP-41 reigned supreme in the calculator world – thus the conspicuous lack of material in the HP-41 literature.

The module is divided in four mayor sections as follows:

1. "-CHAOS 101" is the opening section with a few MCODE utility functions on the subject. Some are trivial examples but nevertheless they're helpful to understand the basic concepts dealt with in the module. Includes functions on the Logistics Map and the Hénon attractor.

2. Next follows the "-PENDULUM" section, which starts with a large-angle compatible version for the period of a single pendulum, and continues covering the double pendulum, the elastic pendulum, and a 3-magnet configuration of the magnetic pendulum as well. All cases are solved numerically using a 4th order Runge-Kutta method on a system of four ODEs. A set of MCODE functions is also provided to calculate the Lagrangian for any dynamic configuration of these systems.

3. The "-ATTRACTORS" section includes routines to study four of the more popular cases, such as the Lorentz, Röessler, Thomas and Sprott' systems of three ODEs; also solved numerically using a 4th order Runge-Kutta method. All systems of differential equations have been programmed as MCODE functions for increased accuracy and speed – albeit as you can already expect, "fast" is not a word that defines the operation of the routines. Using TURBO mode (on the 41CL or V41) is a real must.

4. Finally, the "-N-BODY PRB" section includes two methods to solve the gravitational n-body problem defined on an inertial frame of reference, using Runge-Kutta and Numerov's methodology. This section is a subset of the module with the same name, which also includes Heliocentric coordinates and adds a third resolution approach (7-th. degree Multi-step).

It's important that you adjust your expectations up front: you'll find no fancy phase diagrams or 3D representations of the equations of movement in the manual or done with the module – that's not the point of this project, as it "just" handles a modest numerical resolution of the system equations. No more, no less!

The Runge-Kutta programs and the core n-body problem routines were written by Jean-Marc Baillard. He also provided advise on the resolution of the Double pendulum examples.

© 2022 ÁNGEL MARTIN

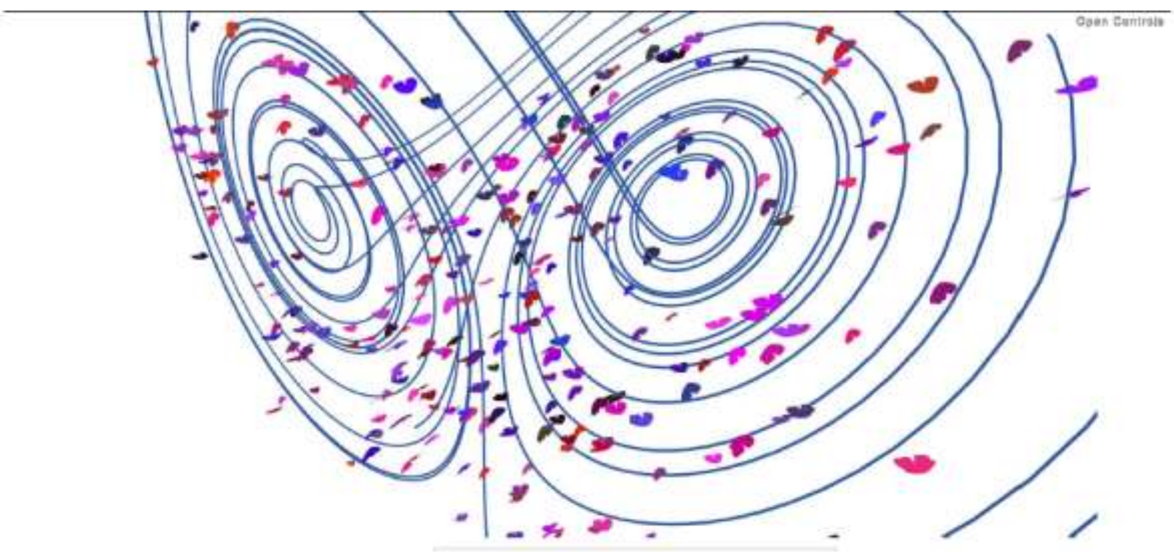
Module function table

Without further ado, see below the functions included in each section of the module.

XROM	Function	Description	Input	Author
31,00	-CHAOS 101	Section Header	n/a	*n/a*
31,01	**AGM**	Arithmetic-Geometric Mean	Values in X, Y	*Ángel Martin*
31,02	**CRITPT**	Lorenz Critical Points	σ,ρ,β, in {Z,Y,X}	*Ángel Martin*
31,03	**FEIG1**	Feigenbaum first constant	none	*Ángel Martin*
31,04	**FEIG2**	Feigenbaum second constant	none	*Ángel Martin*
31,05	**HENON**	Hénon Attractor	xn in X , yn in Y	*Ángel Martin*
31,06	**LGEQ**	Logistics Equation	r in Y, xn in X	*Ángel Martin*
31,07	**LGLIM**	LG Limit Values	r in Y, xn in X	*Ángel Martin*
31,08	**LYAPNV**	Lyapunov's Criteria	σ,ρ,β, in {Z,Y,X}	*Ángel Martin*
31,09	**SIGMD**	Sigmoid function	Argument in X	*Ángel Martin*
31,10	**STABL?**	Lorenz Stability Test	σ,ρ,β, in {Z,Y,X}	*Ángel Martin*
31,11	**ULAM**	Collatz Conjecture	Start point in X	*Ángel Martin*
31,12	-PENDULUM	Section Header	n/a	*n/a*
31,13	**"PEND**	Simple Pendulum Period	Prompts for data	*Ángel Martin*
31,14	**PLAG**	Single Pendulum Lagrangian	System Parameters	*Ángel Martin*
31,15	**"DBLPEN**	Double pendulum driver	Prompts for data	*Ángel Martin*
31,16	**"DP/DT**	Pendulum Dispatcher routine	User flags F00 – F02	*Ángel Martin*
31,17	**DBPLAG**	Double pendulum Lagrangian	System Parameters	*Ángel Martin*
31,18	**DW/DT**	Double P. Equations	System Parameters	*Ángel Martin*
31,19	**"ELTPEN**	Elastic Pendulum driver	Prompts for data	*Ángel Martin*
31,20	**ELPLAG**	Elastic pendulum Lagrangian	System Parameters	*M. Harwood*
31,21	**DE/DT**	Elastic P. Equations	System Parameters	*Ángel Martin*
31,22	**"MAGPEN**	Magnetic Pendulum driver	Prompts for data	*Ángel Martin*
31.23	**DM/DT**	Magnetic P. Equations	System Parameters	*Ángel Martin*
31.24	-ATTRACTORS	Section Header	n/a	*n/a*
31.25	**"LORENZ**	Lorenz Attractor driver	Prompts for data	*Ángel Martin*
31.26	**"DL/DT**	Dispatching routine	User flags F00 – F03	*Ángel Martin*
31.27	**dL/dT**	Lorenz system of ODEs	System Parameters	*Ángel Martin*
31.28	**"RSSLER**	Röessler Attractor driver	Prompts for data	*Ángel Martin*
31.29	**dR/dT**	RÓesless system of ODEs	System Parameters	*Ángel Martin*
31.30	**"THOMAS**	Thomas'Attractor driver	Prompts for data	*Ángel Martin*
31.31	**dT/dT**	Thomas' system of ODEs	System Parameters	*Ángel Martin*
31.32	**"SPROTT**	Sprott's Attractor driver	Prompts for data	*Ángel Martin*
31.33	**dS/dT**	Sprott's System of ODEs	System Parameters	*Ángel Martin*
31.34	**"?**	Auxiliary Prompt-1	Value in X; ALPHA text	*Ángel Martin*
31.35	**"=**	Auxiliary Prompt-2	Text in ALPHA	*Ángel Martin*

31.36	-N-BODY PRB	Section Header	n/a	*n/a*
31.37	**AINT**	Append Integer	x in X	*Firtz Fewerda*
31.38	**E3/E+**	Pointer builder	x in X	*Ángel Martin*
31.39	**<G>**	Gravitational Constant	none	*Ángel Martin*
31.40	**<K>**	Gaussian Grav. Constant	None	*Ángel Martin*
31.41	**">XM**	X-Mem Storage	See manual	*Ángel Martin*
31.42	**"GM**	Inertial FoR, Runge-Kutta	See manual	*JM Baillard*
31.43	**"GM+**	Driver for "GM	Prompts for data	*Ángel Martin*
31.44	**"GM2**	Inertial FoR, Numerov's	See manual	*JM Baillard*
31.45	**"GM2+**	Driver for "GM2	Prompts for data	*Ángel Martin*
31.46	**"RK4C**	System of 3 ODE's using RK-4	See manual	*JM Baillard*
31.47	**"RK4N**	System of N-ODE's using RK-4	See manual	*JM Baillard*
31.48	**"GM3+**	Driver for "GM3	Prompts for data	*Ángel Martin*

(*) Function "GM3+" is not enabled in the FAT



I. Chaos Utilities

The table below shows the utility functions from the opening section in the module:

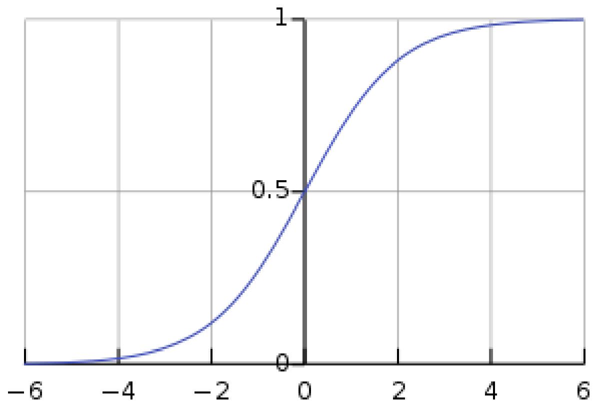
Function	Description	Input	Output
AGM	Arithmetic-Geometric Mean	Values in X, Y	AGM(x,y)
CRITPT	Lorenz Critical Points	σ,ρ,β, in {Z,Y,X}	(Px, Py, Pz) in {Z,Y,X}
FEIG1	Feigenbaum first constant	none	d=4.669201609
FEIG2	Feigenbaum second constant	none	a=2.502907875
HENON	Hénon Attractor	xn in X , yn in Y	Y(n+1), x(n+1) in {Y,X}
LGEQ	Logistics Equation	r in Y, xn in X	X(n+1) in X
LGLIM	LG Limit	r in Y, xn in X	Limit in X
LYAPNV	Lyapunov's Criteria	σ,ρ,β, in {Z,Y,X}	Lyapunov's factor in X
SIGMD	Sigmoid function	Argument in X	Result in X
STABL?	Lorenz Stability Test	σ,ρ,β, in {Z,Y,X}	Yes/No, skip if false
ULAM	Collatz Conjecture	Start point in X	Shows steps/final count

Here's a short description for them, starting with the two not related to the chaos theme.

SIGMD calculates the Sigmoid of the argument in x. This function is relevant in machine learning and data mining fields. It is defined as:

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} = 1 - S(-x).$$

The result is placed in X and the original argument is saved in LastX. Y,Z,T are untouched and no data registers are used either.



Examples:

1, XEQ "SIGMD => 0.731058579
2, XEQ "SIGMD => 0.880797078

The Sigmoid function is also known as the *Standard Logistics function*, which will appear linked to the Logistics Map in the discrete domain – therefore it very much belongs to the main subject of this module after all.

© 2022 ÁNGEL MARTIN

Derivative and Integral of the Sigmoid function.

The derivative is known as the density of the logistic distribution:

$$\frac{\mathrm{d}}{\mathrm{d}x}f(x) = \frac{e^x \cdot (1+e^x) - e^x \cdot e^x}{(1+e^x)^2} = \frac{e^x}{(1+e^x)^2} = f(x)\big(1-f(x)\big)$$

Conversely, its antiderivative can be computed by the substitution u = 1+e^x , since f(x) = u'/ u, so (dropping the constant of integration)

$$\int \frac{e^x}{1+e^x}\,dx = \int \frac{1}{u}\,du = \ln u = \ln(1+e^x).$$

In artificial neural networks, this is known as the *softplus* function and (with scaling) is a smooth approximation of the ramp function, just as the logistic function (with scaling) is a smooth approximation of the Heaviside step function.

Finally, SIGMD is a rather simple function. The MCODE listing is shown below.

Header	AEAC	084	"D"	
Header	AEAD	00D	"M"	**Sigmoid Function**
Header	AEAE	007	"G"	*sig = 1/(1 + e^-x)*
Header	AEAF	009	"I"	
Header	AEB0	013	"S"	*Ángel Martin*
SIGMD	**AEB1**	**0F8**	**READ 3(X)**	
	AEB2	361	?NC XQ	(includes SETDEC)
	AEB3	050	->14D8	[CHK_NO_S]
	AEB4	2BE	C=-C-1 MS	Sign change
	AEB5	044	CLRF 4	standard version (w/out "-1")
	AEB6	029	?NC XQ	e^-x
	AEB7	068	->1A0A	[EXP10]
	AEB8	001	?NC XQ	1+e^-x
	AEB9	060	->1800	[ADDONE]
	AEBA	239	?NC XQ	1/(1+e^-x)
	AEBB	060	->188E	[ON/X13]
	AEBC	331	?NC GO	Overflow, DropST, FillXL & Exit
	AEBD	002	->00CC	[NFRX]

Collatz conjecture. (see:: https://en.wikipedia.org/wiki/Collatz_conjecture)

ULAM shows the successive values in the Collatz conjecture, starting with the argument in X.
It is completely off-topic subject but it sorts of happened while preparing this manual – what an excuse, uh?

The **ULAM** function does a complete path starting with the value in X, all the way until the end when "1" is reached using the well-known Ulam's (or Collatz's) algorithm:

- If odd, multiply by three and add one
- If even, divide by two

$$f(n) = \begin{cases} \frac{n}{2} & \text{if } n \equiv 0 \pmod{2} \\ 3n+1 & \text{if } n \equiv 1 \pmod{2}. \end{cases}$$

The function will take the integer part of the absolute value of the number in X. Then all intermediate values are briefly shown, and the total number of "nodes" is left in X upon completion. The starting number is left in X.

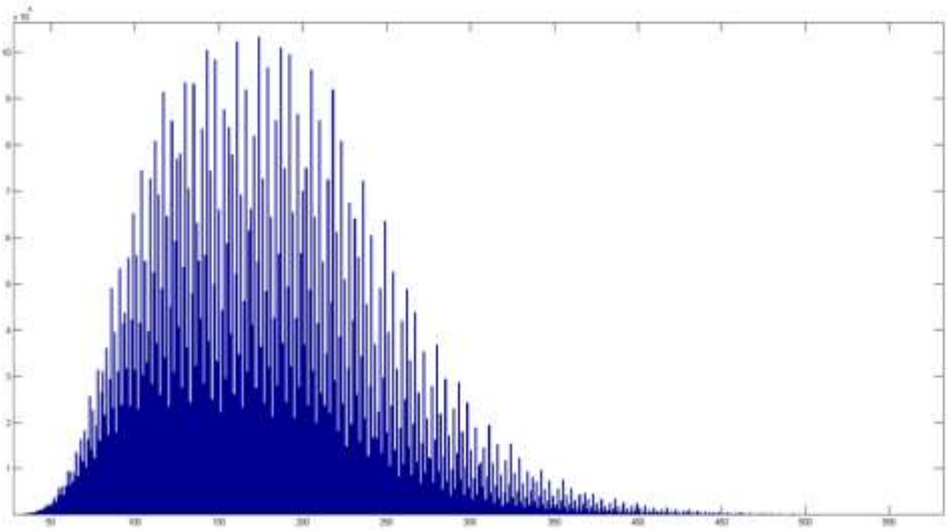
Examples:

41, XEQ "ULAM" -> generates a sequence of 109 numbers

22, ULAM -> generates a sequence of 15 numbers

The sequence for n = 27, listed below, takes 111 steps (41 steps through odd numbers), climbing as high as 9232 before descending to 1.

27, 82, 41, 124, 62, 31, 94, 47, 142, 71, 214, 107, 322, 161, 484, 242, 121, 364, 182, 91, 274, 137, 412, 206, 103, 310, 155, 466, 233, 700, 350, 175, 526, 263, 790, 395, 1186, 593, 1780, 890, 445, 1336, 668, 334, 167, 502, 251, 754, 377, 1132, 566, 283, 850, 425, 1276, 638, 319, 958, 479, 1438, 719, 2158, 1079, 3238, 1619, 4858, 2429, 7288, 3644, 1822, 911, 2734, 1367, 4102, 2051, 6154, 3077, 9232, 4616, 2308, 1154, 577, 1732, 866, 433, 1300, 650, 325, 976, 488, 244, 122, 61, 184, 92, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1 (sequence A008884 in the OEIS)



Histogram of total stopping times for the numbers 1 to 108. Total stopping time is on the x axis, frequency on the y axis.

© 2022 ÁNGEL MARTIN

MCODE listing

Header	AEBF	*08D*	*"M"*	
Header	AEC0	*001*	*"A"*	***Collatz Conjecture***
Header	AEC1	*00C*	*"L"*	
Header	AEC2	*015*	*"U"*	*Ángel Martin*
ULAM	**AEC3**	**0F8**	**READ 3(X)**	
	AEC4	128	WRIT 4(L)	
	AEC5	*149*	*?NC XQ*	*Integer & Positive*
	AEC6	*134*	*->4D52*	*[CHKZI]*
	AEC7	268	WRIT 9(Q)	
	AEC8	04E	C=0 ALL	
	AEC9	0E8	WRIT 3(X)	*reset the counter*
LOOP1	AECA	00E	A=0 ALL	
	AECB	35C	PT= 12	*Builds "1" in A*
	AECC	162	A=A+1 @PT	
	AECD	278	READ 9(Q)	
	AECE	36E	?A#C ALL	*end of the path?*
	AECF	**3A0**	**?NC RTN**	*yes, end here.*
	AED0	0F8	READ 3(X)	
	AED1	2A0	SETDEC	
	AED2	*01D*	*?NC XQ*	*increase counter*
	AED3	*060*	*->1807*	*[AD2_10]*
	AED4	0E8	WRIT 3(X)	*update value*
	AED5	278	READ 9(Q)	*get current n*
	AED6	*3CD*	*?NC XQ*	*C= MOD[int(C),2]*
	AED7	*100*	*->40F3*	*[MOD2]*
	AED8	2EE	?C#0 ALL	*it is odd?*
	AED9	***02F***	***JC +05***	*yes, skip*
	AEDA	278	READ 9(Q)	
EVEN	AEDB	*3CD*	*?NC XQ*	*{A,B} = {C} /2*
	AEDC	*13C*	*->4FF3*	*[DIVTWO]*
	AEDD	***053***	***JNC +10d***	*show result*
ODD	AEDE	04E	C=0 ALL	
	AEDF	35C	PT= 12	
	AEE0	0D0	LD@PT- 3	
	AEE1	10E	A=C ALL	
	AEE2	278	READ 9(Q)	
	AEE3	*135*	*?NC XQ*	*3*n*
	AEE4	*060*	*->184D*	*[MP2_10]*
	AEE5	*001*	*?NC XQ*	*3*n+1*
	AEE6	*060*	*->1800*	*[ADDONE]*
MERGE	AEE7	268	WRIT 9(Q)	
	AEE8	*099*	*?NC XQ*	*Sends C to display - sets HEX*
	AEE9	*02C*	*->0B26*	*[DSPCRG]*
	AEEA	*1FD*	*?NC XQ*	*wait a little - CL compatible*
	AEEB	*12C*	*->4B7F*	*[WAIT4L] - Enables RAM*
	AEEC	*1FD*	*?NC XQ*	*wait a little - CL compatible*
	AEED	*12C*	*->4B7F*	*[WAIT4L] - Enables RAM*
	AEEE	*2E3*	*JNC -36d*	[LOOP1]

The calls to [WAIT4L] ensure compatibility with the SY-41CL – slowing down the output for the user to catch a glimpse of the enumerated values.

Feigenbaum constants. (see: https://en.wikipedia.org/wiki/Feigenbaum_constants)

In mathematics, specifically bifurcation theory, the **Feigenbaum constants** are two mathematical constants which both express ratios in a bifurcation diagram for a non-linear map. They are named after the physicist Mitchell J. Feigenbaum.

Their values are shown below:

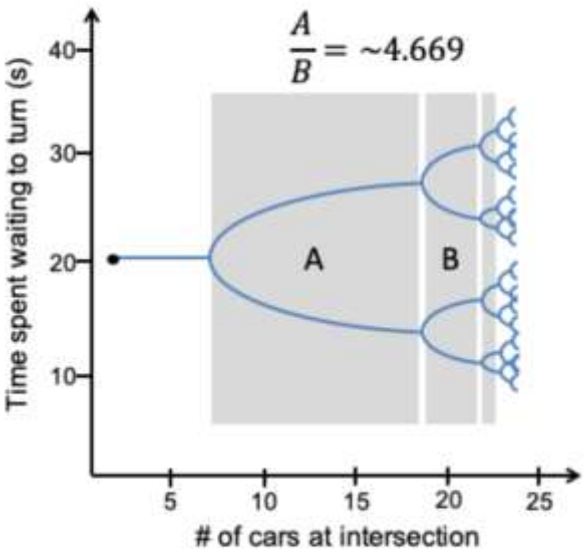
$$\delta = \lim_{n\to\infty} \frac{a_{n-1} - a_{n-2}}{a_n - a_{n-1}} = 4.669\,201\,609\ \ldots,$$

$$\alpha = 2.502\,907\,875\,095\,892\,822\,283\,902\,873\,218\ldots,$$

As you can expect, the functions **FEIG1** and **FEIG2** place the corresponding constant in the X register, lifting the stack if CPU flag 11 is set.

Feigenbaum originally related the first constant to the period-doubling bifurcations in the logistic map, but also showed it to hold for all one-dimensional maps with a single quadratic maximum. As a consequence of this generality, every chaotic system that corresponds to this description will bifurcate at the same rate. Feigenbaum made this discovery in 1975, and he officially published it in 1978

Feigenbaum constant δ expresses the limit of the ratio of distances between consecutive bifurcation diagram on L_i/L_{i+1}



Both numbers are believed to be transcendental, although they have not been proven to be so. There is also no known proof that either constant is irrational.

Examples:

XEQ "FEIG1" => 4.669201609

XEQ "FEIG2" => 2.502907875

MCODE listing.

A trivial implementation but shown here for completion sake and as a reference for future use. Note the call to [R_SUB] – notwithstanding F11 status - to lift the stack prior to entering the constant's value in the X register.

Header	AE47	0B2	"2"	
Header	AE48	007	"G"	Feigenbaum 2nd. Constant
Header	AE49	009	"I"	a=2.502907875
Header	AE4A	005	"E"	
Header	AE4B	006	"F"	Ángel Martin
FEIG2	AE4C	18C	?FSET 11	
	AE4D	3B5	?C XQ	Stack lift
	AE4E	051	->14ED	[R_SUB]
	AE4F	04E	C=0 ALL	
	AE50	35C	PT= 12	
	AE51	090	LD@PT- 2	
	AE52	150	LD@PT- 5	
	AE53	010	LD@PT- 0	
	AE54	090	LD@PT- 2	
	AE55	250	LD@PT- 9	
	AE56	010	LD@PT- 0	
	AE57	1D0	LD@PT- 7	
	AE58	210	LD@PT- 8	
	AE59	1D0	LD@PT- 7	
	AE5A	150	LD@PT- 5	
	AE5B	0AB	JNC +21d	
Header	AE5C	0B1	"1"	
Header	AE5D	007	"G"	Feigenbaum 1st. Constant
Header	AE5E	009	"I"	d=4.669201609
Header	AE5F	005	"E"	
Header	AE60	006	"F"	Ángel Martin
FEIG1	AE61	18C	?FSET 11	
	AE62	3B5	?C XQ	Stack lift
	AE63	051	->14ED	[R_SUB]
	AE64	04E	C=0 ALL	
	AE65	35C	PT= 12	
	AE66	110	LD@PT- 4	
	AE67	190	LD@PT- 6	
	AE68	190	LD@PT- 6	
	AE69	250	LD@PT- 9	
	AE6A	090	LD@PT- 2	
	AE6B	010	LD@PT- 0	
	AE6C	050	LD@PT- 1	
	AE6D	190	LD@PT- 6	
	AE6E	010	LD@PT- 0	
	AE6F	250	LD@PT- 9	
EXIT	AE70	0E8	WRIT 3(X)	
	AE71	3E0	RTN	

Arithmetic-Geometric Mean - Revisited { AGM }

In mathematics, the arithmetic–geometric mean (AGM) of two positive real numbers x and y is defined as follows: First compute the arithmetic mean of x and y and call it a1. Next compute the geometric mean of x and y and call it g1; this is the square root of the product xy:

$$a_1 = \frac{1}{2}(x+y)$$
$$g_1 = \sqrt{xy}$$

Then iterate this operation with a1 taking the place of x and g1 taking the place of y. In this way, two sequences (an) and (gn) are defined:

$$a_{n+1} = \frac{1}{2}(a_n + g_n)$$
$$g_{n+1} = \sqrt{a_n g_n}$$

These two sequences converge to the same number, which is the arithmetic–geometric mean of x and y; it is denoted by M(x, y), or sometimes by agm(x, y).

Stack	Input	Output
Y	a0	Z
X	b0	**agm**(a0,b0)
L	-	b0

Note that "DATA ERROR" will be triggered when one of the arguments is negative (but not if both are).

Example 1:

To find the arithmetic–geometric mean of a0 = 24 and g0 = 6, simply input:

24, ENTER^, 6, XEQ "AGM" → 13,45817148

Example 2. Gauss Constant.

The reciprocal of the arithmetic–geometric mean of 1 and the square root of 2 is called Gauss's constant, after Carl Friedrich Gauss. Calculate it using AGM:

2, SQRT, 1, XEQ "AGM" → 1,198140235; 1/X → 0,834626842

A piece of trivia: the Gauss constant is a transcendental number, and appears in the calculation of several integrals such as those below:

$$\frac{1}{G} = \int_0^{\pi/2} \sqrt{\sin(x)}dx = \int_0^{\pi/2} \sqrt{\cos(x)}dx$$
$$G = \int_0^{\infty} \frac{dx}{\sqrt{\cosh(\pi x)}}$$

—

Example 3.- Complete Elliptic Integral of 1st Kind.

Using **AGM** it's a convenient way to calculate the Complete Elliptic Integral of the first kind, **ELIPK** (k), by means of the following relationship (where M(x,y) represents the AGM):

$$M(x,y) = \frac{\pi}{2} \Big/ \int_0^{\pi/2} \frac{d\theta}{\sqrt{x^2\cos^2\theta + y^2\sin^2\theta}} \quad = \frac{\pi}{4}(x+y) \Big/ K\left(\frac{x-y}{x+y}\right)$$

where K(k) is the Complete Elliptic Integral of the first kind:

$$K(k) = \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-k^2\sin^2\theta}} = \int_0^1 \frac{dt}{\sqrt{(1-t^2)(1-k^2t^2)}}.$$

As usual the conventions used for the input parameters get in the way – so paying special attention to this, we can re-write the expression using the **In**complete Elliptic Integral instead, as follows:

ELIPF { π/2 | (a-b)/(a+b) } = π (a+b) / 4 **AGM**(a,b), which is the same as:

ELIPF { π/2, [(a-b)/(a+b)]^2 } = π (a+b) / 4 **AGM**(a,b)

The idea is to find two values a,b derived from the argument: x = [(a-b)/(a+b)]^2

The easiest approach is to choose a=1, and therefore: b= [1-sqr(x)] / [1+sqr(x)]

Here's the FOCAL program used for the calculation. - Note the first step needed to get the square root of the argument, to harmonize both conventions used.

1	LBL "ELIPK"		7	E		13	4		19	E
2	SQRT		8	+		14	*		20	+
3	E		9	/		15	1/X		21	*
4	X<>Y		10	RCL X		16	PI		22	END
5	-		11	E		17	*			
6	LASTX		12	AGM		18	X<>Y			

And here are some results comparing the values obtained using **ELIPF** in the SandMath module. As you can expect, the execution is substantially faster using the **AGM** approach.

x	ELIPK(x)	ELIPF (π/2, x)	% Delta
0.1	1.612441348	1.612441348	0
0.2	1.659623599	1.659623598	6.02546E-10
0.3	1.713889448	1.713889447	5.83468E-10
0.4	1.777519373	1.777519371	1.12516E-09
0.5	1.854074677	1.854074677	0
0.6	1.949567749	1.949567749	0
0.7	2.075363134	2.075363135	-4.81843E-10
0.8	2.257205326	2.257205326	0
0.9	2.578092113	2.578092113	0

MCODE Listing

See below the MCODE for this function – a quick implementation using an iterative loop repeated until both values are equal (i.e. their difference is less than 1 E-9).

Header	ABDA	08D	"M"	Arithmetic-Geometric Mean
Header	ABDB	007	"G"	
Header	ABDC	001	"A"	Ángel Martin
AGM	ABDD	1A5	?NC XQ	Check for valid entries
	ABDE	100	->4069	[CHKST2] - sets DEC mode
	ABDF	128	WRIT 4(L)	
LOOP	ABE0	0F8	READ 3(X)	bn
	ABE1	05E	C=0 MS	make it positive
	ABE2	10E	A=C ALL	
	ABE3	0B8	READ 2(Y)	an
	ABE4	05E	C=0 MS	make it positive
	ABE5	36E	?A#C ALL	
	ABE6	0E3	JNC +28d	
	ABE7	01D	?NC XQ	an+bn
	ABE8	060	->1807	[AD2_10]
	ABE9	3D9	?NC XQ	{A,B} = {A,B} /2
	ABEA	13C	->4FF6	[DIVBY2]
	ABEB	070	N=C ALL	a(n+1)
	ABEC	089	?NC XQ	{A,B} -> {Q,+}
	ABED	064	->1922	[STSCR]
	ABEE	0F8	READ 3(X)	bn
	ABEF	10E	A=C ALL	
	ABF0	0B8	READ 2(Y)	an
	ABF1	135	?NC XQ	
	ABF2	060	->184D	[MP2_10]
	ABF3	305	?NC XQ	
	ABF4	060	->18C1	[SQR13]
	ABF5	0E8	WRIT 3(X)	b(n+1)
	ABF6	2BE	C=-C-1 MS	-b(n+1)
	ABF7	11E	A=C MS	13-digit too
	ABF8	0B0	C=N ALL	
	ABF9	0A8	WRIT 2(Y)	
	ABFA	0D1	?NC XQ	partial result
	ABFB	064	->1934	[RCSCR]
	ABFC	031	?NC XQ	a(n+1)-b(n+1)
	ABFD	060	->180C	[AD2-13]
	ABFE	351	?NC XQ	9-decimal digits precision
	ABFF	128	>4AD4	[TOLER4]
	AC00	2FE	?C#0 MS	is (delta-err) <0 => d<e?
	AC01	2FB	JNC -33d	no, keep at it
EXIT	AC02	3A5	?NC XQ	roll_down
	AC03	050	->14E9	[RDNSUB]
	AC04	3C1	?NC GO	finish off...
	AC05	002	->00F0	[NFRPU]

Logistic Equation. (see: https://en.wikipedia.org/wiki/Logistic_map)

The **logistic map** is a polynomial mapping (equivalently, recurrence relation) of degree 2, often cited as an archetypal example of how complex, chaotic behaviour can arise from very simple non-linear dynamical equations. The map was popularized in a 1976 paper by the biologist Robert May,[1] in part as a discrete-time demographic model analogous to the logistic equation written down by Pierre François Verhulst.

Mathematically, the logistic map is written as:

$$x_{n+1} = rx_n (1 - x_n)$$

where x_n is a number between zero and one, that represents the ratio of existing population to the maximum possible population

The usual values of interest for the parameter "r" are those in the interval [0, 4], so that x_n remains bounded on [0, 1]. The $r = 4$ case of the logistic map is a nonlinear transformation of both the bit-shift map and the $\mu = 2$ case of the tent map. If $r > 4$ this leads to negative population sizes. (This problem does not appear in the older Ricker model, which also exhibits chaotic dynamics.) One can also consider values of r in the interval [−2, 0], so that x_n remains bounded on [−0.5, 1.5].

Chaos and the logistic map

The relative simplicity of the logistic map makes it a widely used point of entry into a consideration of the concept of chaos. A rough description of chaos is that chaotic systems exhibit a great sensitivity to initial conditions—a property of the logistic map for most values of r between about 3.57 and 4. A common source of such sensitivity to initial conditions is that the map represents a repeated folding and stretching of the space on which it is defined. In the case of the logistic map,

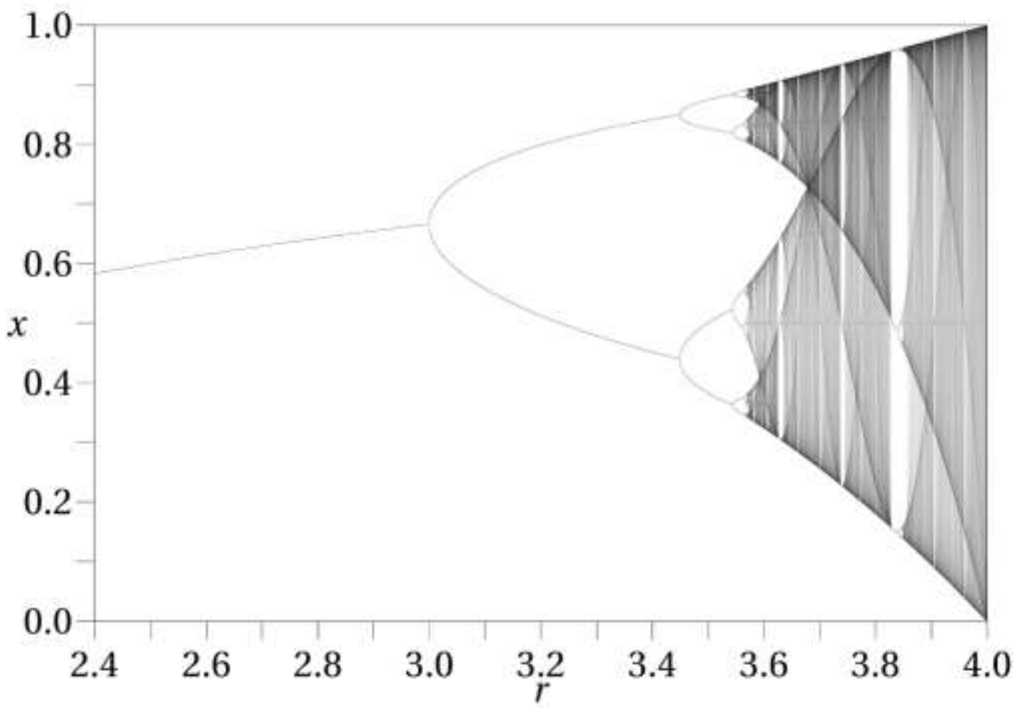
By varying the parameter r, the following behavior is observed:

- With r between 0 and 1, the population will eventually die, independent of the initial population.
- With r between 1 and 2, the population will quickly approach the value $r - 1/r$, independent of the initial population.
- With r between 2 and 3, the population will also eventually approach the same value $r - 1/r$, but first will fluctuate around that value for some time. The rate of convergence is linear, except for $r = 3$, when it is dramatically slow, less than linear.
- With r between 3 and $1 + \sqrt{6} \approx 3.44949$ the population will approach permanent oscillations between two values. These two values are dependent on r and given by:

$$x_{\pm} = \frac{1}{2r}\left(r + 1 \pm \sqrt{(r-3)(r+1)}\right).$$

- With r between 3.44949 and 3.54409 (approximately), from almost all initial conditions the population will approach permanent oscillations among four values. The latter number is a root of a 12th degree polynomial (sequence A086181 in the OEIS).
- With r increasing beyond 3.54409, from almost all initial conditions the population will approach oscillations among 8 values, then 16, 32, etc. The lengths of the parameter intervals that yield oscillations of a given length decrease rapidly; the ratio between the lengths of two successive bifurcation intervals approaches the Feigenbaum constant $\delta \approx 4.66920$. This behavior is an example of a period-doubling cascade.

- At $r \approx 3.56995$ (sequence A098587 in the OEIS) is the onset of chaos, at the end of the period-doubling cascade. From almost all initial conditions, we no longer see oscillations of finite period. Slight variations in the initial population yield dramatically different results over time, a prime characteristic of chaos.
- Most values of r beyond 3.56995 exhibit chaotic behavior, but there are still certain isolated ranges of r that show non-chaotic behavior; these are sometimes called *islands of stability*. For instance, beginning at 1 + √8 (approximately 3.82843) there is a range of parameters r that show oscillation among three values, and for slightly higher values of r oscillation among 6 values, then 12 etc.



LGLIM calculates the convergence limit for the provided parameter r in the stack Y register – starting with the value in X, which becomes the starting point of the limit calculation, and because of the chaotic nature of the equation also has an impact in the calculation of the limit in those regions where there isn't s finite number of possible results.

Examples:

1.5, ENTER^, 0.5, LGLIM => 0.333333333

3.3, ENTER^, 0.5, LGLIM => either: 0.823603283 or: 0.479503910

Note that the functions don't check for valid range of the input parameters, so always make sure the xn values are between 0 and 1.

MCODE listing.

See on next page the actual implementation for both **LGEQ** and **LGLIM**. The latter is just a repeated execution of the former until there's no meaningful difference between two consecutive values.

Header	AE73	08D	"M"	
Header	AE74	009	"I"	**Logistics Equation LIMIT**
Header	AE75	00C	"L"	r in Y
Header	AE76	007	"G"	xn in X
Header	AE77	00C	"L"	Ángel Martin
LGLIM	**AE78**	1A5	?NC XQ	Check Data in {X,Y}
	AE79	100	->4069	[CHKST2] - sets DEC
	AE7A	128	WRIT 4(L)	LastX value
	AE7B	070	N=C ALL	**xn**
LOOP	AE7C	3C8	CLRKEY	reset KY
	AE7D	3CC	?KEY	wait for a keypress
	AE7E	360	?C RTN	
	AE7F	375	PORT DEP:	Calculate Next value
	AE80	03C	XQ	xn+1
	AE81	2A0	->AEA0	[LGEQ#]
	AE82	0E8	WRIT 3(X)	Xn+1 in X
	AE83	0F0	C<>N ALL	Xn+1 to N, recall xn
	AE84	158	M=C ALL	save it in M
	AE85	0F0	C<>N ALL	Xn+1
	AE86	099	?NC XQ	Sends C to display - sets HEX
	AE87	02C	->0B26	[DSPCRG]
	AE88	2A0	SETDEC	
	AE89	198	C=M ALL	xn
	AE8A	2BE	C=-C-1 MS	
	AE8B	10E	A=C ALL	-xn in A
	AE8C	0F8	READ 3(X)	x(n+1)
	AE8D	070	N=C ALL	preserve in N
	AE8E	01D	?NC XQ	error = x(n+1) - xn
	AE8F	060	->1807	[AD2_10]
	AE90	351	?NC XQ	9-digits precision
	AE91	128	>4AD4	[TOLER4]
	AE92	2FE	?C#0 MS	is (error - 1E-9) <0 <=> err < 1 E
	AE93	34B	JNC -23d	no, -> [LOOP]
	AE94	3E0	RTN	
Header	AE95	091	"Q"	**Logistics Equation**
Header	AE96	005	"E"	r in Y
Header	AE97	007	"G"	xn in X
Header	AE98	00C	"L"	Ángel Martin
LGEQ	**AE99**	1A5	?NC XQ	Check Data in {X,Y}
	AE9A	100	->4069	[CHKST2] - sets DEC
	AE9B	375	PORT DEP:	Calculate Next value
	AE9C	03C	XQ	
	AE9D	2A0	->AEA0	[LGEQ#]
	AE9E	331	?NC GO	Overflow, DropST, FillXL & Exit
	AE9F	002	->00CC	[NFRX]
LGEQ#	AEA0	2A0	SETDEC	
	AEA1	0F8	READ 3(X)	xn
	AEA2	2BE	C=-C-1 MS	Sign change
$(1-x_n)$	AEA3	000	NOP	- xn
	AEA4	1E1	?NC XQ	1-xn
	AEA5	100	->4078	[INCC10]
	AEA6	0F8	READ 3(X)	xn
	AEA7	13D	?NC XQ	xn(1-xn)
	AEA8	060	->184F	[MP1_10]
	AEA9	0B8	READ 2(Y)	r
	AEAA	13D	?NC GO	r.xn.(1-xn)
	AEAB	062	->184F	[MP1_10]

Hénon Attractor. (see: https://en.wikipedia.org/wiki/H%C3%A9non_map)

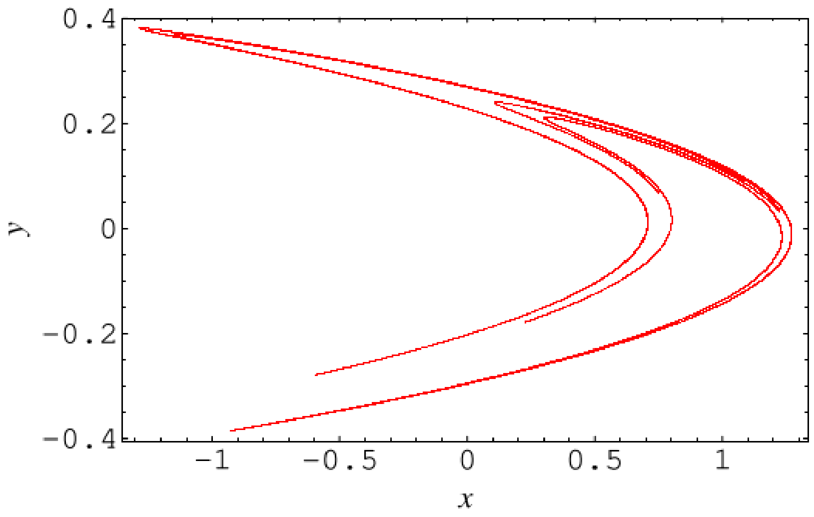
The **Hénon map**, sometimes called **Hénon-Pomeau attractor/map**,[1] is a discrete-time dynamical system. It is one of the most studied examples of dynamical systems that exhibit chaotic behavior. The Hénon map takes a point (x_n, y_n) in the plane and maps it to a new point given by:

$$\begin{cases} x_{n+1} = 1 - a x_n^2 + y_n \\ y_{n+1} = b x_n . \end{cases}$$

The map depends on two parameters, *a* and *b*, which for the **classical Hénon map** have values of *a* = 1.4 and *b* = 0.3. For the classical values the Hénon map is chaotic. For other values of *a* and *b* the map may be chaotic, intermittent, or converge to a periodic orbit. An overview of the type of behavior of the map at different parameter values may be obtained from its orbit diagram.

The figure below shows the Hénon attractor for *a* = 1.4 and *b* = 0.3. As you can see it maps two points into themselves: these are the *invariant points*. For the classical values of *a* and *b* of the Hénon map, one of these points is on the attractor:

x = 0.631354477 ;
y = 0.189406343 ;



The function **HENON** uses the classic map (thus a,b are fixed) and expects the x,y values in stack registers X,Y respectively.

Examples:

1, ENTER^, XEQ "HENON" => 0.3000000000
X<>Y => 0.6000000000

Note that the initial x,y values are replaced by the new ones – and that *only x is saved in LastX* (thus the initial yn will be lost.

MCODE listing

HENON is a very straightforward function, easy to put together using the OS math routines and a couple more borrowed from the Library#4 for further convenience.

Header	AC50	08E	"N"	
Header	AC51	00F	"O"	Hénon's Attractor
Header	AC52	00E	"N"	xn in X; yn in Y
Header	AC53	005	"E"	
Header	AC54	008	"H"	Ángel Martin
HENON	AC55	1A5	?NC XQ	Check for valid entries
	AC56	100	->4069	[CHKST2] - sets DEC mode
	AC57	128	WRIT 4(L)	xn saved in LastX
	AC58	04E	C=0 ALL	
	AC59	35C	PT=12	C = 0.3
	AC5A	0D0	LD@PT-3	
	AC5B	266	C=C-1 S&X	
	AC5C	10E	A=C ALL	
	AC5D	0F8	READ 3(X)	xn
	AC5E	135	?NC XQ	0.3. xn
	AC5F	060	->184D	[MP2_10]
	AC60	070	N=C ALL	
	AC61	0F8	READ 3(X)	xn
	AC62	10E	A=C ALL	
	AC63	135	?NC XQ	xn^2
	AC64	060	->184D	[MP2_10]
	AC65	04E	C=0 ALL	
	AC66	2DC	PT= 13	
	AC67	250	LD@PT- 9	C = -1.4
	AC68	050	LD@PT- 1	
	AC69	110	LD@PT- 4	
	AC6A	13D	?NC XQ	-1,4,xn^2
	AC6B	060	->184F	[MP1_10]
	AC6C	089	?NC XQ	-1.4.xn^2
	AC6D	064	->1922	[STSCR]
	AC6E	0B8	READ 2(Y)	yn
	AC6F	1E1	?NC XQ	1+yn
	AC70	100	->4078	[INCC10]
	AC71	0D1	?NC XQ	-1.4.xn^2
	AC72	064	->1934	[RCSCR]
	AC73	031	?NC XQ	1+yn - 1.4,xn^2
	AC74	060	->180C	[AD2_13]
	AC75	0A8	WRIT 2(Y)	replaces xn, yn with the
	AC76	0B0	C=N ALL	new x(n+1), y(n+1)
	AC77	0E8	WRIT 3(X)	
	AC78	3E0	RTN	done.

Lorenz Attractor Stability and Equilibrium Points

These three functions apply the general concepts of stability and critical points to the particular case of the Lorenz attractor. They use the same inputs in the stack registers. The result returned by **CRITPT** and **LYAPNV** is placed in X and the original value of β is saved in LastX.

Function	Description	Input	Output
CRITPT	Lorenz Critical Points	σ,ρ,β, in {Z,Y,X}	(Px, Py, Pz) in {Z,Y,X}
LYAPNV	Lyapunov's Criteria	σ,ρ,β, in {Z,Y,X}	Lyapunov's Dimension in X
STABL?	Lorenz Stability Test	σ,ρ,β, in {Z,Y,X}	Yes/No, skip if false

One normally assumes that the parameters σ, ρ, and β are positive. Lorenz used the values $\sigma = 10$, $\beta = 8/3$ and $\rho = 28$. The system exhibits chaotic behavior for these (and nearby) values.

If $\rho < 1$ then there is only one equilibrium point, which is at the origin. This point corresponds to no convection. All orbits converge to the origin, which is a global attractor, when $\rho < 1$.

A pitchfork bifurcation occurs at $\rho = 1$, and for $\rho > 1$ two additional critical points appear at

$$\left(\sqrt{\beta(\rho-1)}, \sqrt{\beta(\rho-1)}, \rho-1\right) \quad \text{and} \quad \left(-\sqrt{\beta(\rho-1)}, -\sqrt{\beta(\rho-1)}, \rho-1\right)$$

These correspond to steady convection. This pair of equilibrium points is stable only if

$$\rho < \sigma\frac{\sigma+\beta+3}{\sigma-\beta-1},$$

which can hold only for positive ρ if $\sigma > \beta + 1$. At the critical value, both equilibrium points lose stability through a subcritical Hopf bifurcation.

When $\rho = 28$, $\sigma = 10$, and $\beta = 8/3$, the Lorenz system has chaotic solutions (but not all solutions are chaotic). Almost all initial points will tend to an invariant set – the Lorenz attractor – a strange attractor, a fractal, and a self-excited attractor with respect to all three equilibria. Its Hausdorff dimension is estimated from above by the Lyapunov dimension (Kaplan-Yorke dimension) as 2.06±0.01,[16] and the correlation dimension is estimated to be 2.05±0.01.

The exact *Lyapunov dimension formula* of the global attractor can be found analytically under classical restrictions on the parameters:

$$3 - \frac{2(\sigma+\beta+1)}{\sigma+1+\sqrt{(\sigma-1)^2+4\sigma\rho}}$$

Example. Using the de-facto standard Lorenz parameters $\rho = 28$, $\sigma = 10$, and $\beta = 8/3$

CRITPT => 27.00000000; RDN => 4.153303439; RDN => 4.153303439

XEQ "LYAPNV" => 2.401312764

XEQ "STABL?" => "NO"

See the MCODE Listings if you're interested in the actual implementation of the functions. As usual, a simple utilization of the available OS routines given the straightforward nature of the formulas involved.

© 2022 ÁNGEL MARTIN

Header	ADCA	094	"T"	
Header	ADCB	010	"P"	**Critical Points**
Header	ADCC	014	"T"	Z - σ
Header	ADCD	009	"I"	Y - ρ
Header	ADCE	012	"R"	X - β
Header	ADCF	003	"C"	Ángel Martin
CRITPT	ADD0	199	?NC XQ	Checks XYZ - sets DEC mode
	ADD1	100	->4066	[CHKST3]
	ADD2	0B8	READ 2(Y)	ρ
	ADD3	1FD	?NC XQ	{A,B} = C-1
	ADD4	100	->407F	[DECC10]
	ADD5	0F0	C<>N ALL	
	ADD6	0F8	READ 3(X)	β
	ADD7	13D	?NC XQ	
	ADD8	060	->184F	[MP1_10]
	ADD9	305	?NC XQ	
	ADDA	060	->18C1	[SQR13]
	ADDB	068	WRIT 1(Z)	
	ADDC	0A8	WRIT 2(Y)	
	ADDD	0F0	C<>N ALL	
	ADDE	331	?NC GO	Overflow, DropST, FillXL & Exit
	ADDF	002	->00CC	[NFRX]
Header	ADE1	0BF	"?"	
Header	ADE2	00C	"L"	**Stability Check**
Header	ADE3	002	"B"	Z - σ
Header	ADE4	001	"A"	Y - ρ
Header	ADE5	014	"T"	X - β
Header	ADE6	013	"S"	Ángel Martin
STABL?	ADE7	199	?NC XQ	Checks XYZ - sets DEC mode
	ADE8	100	->4066	[CHKST3]
	ADE9	0F8	READ 3(X)	β
	ADEA	2BE	C=-C-1 MS	sign change
	ADEB	10E	A=C ALL	
	ADEC	078	READ 1(Z)	σ
	ADED	01D	?NC XQ	
	ADEE	060	->1807	[AD2-10]
	ADEF	009	?NC XQ	
	ADF0	060	->1802	[SUBONE]
	ADF1	089	?NC XQ	
	ADF2	064	->1922	[STSCR]
	ADF3	0F8	READ 3(X)	β
	ADF4	10E	A=C ALL	
	ADF5	078	READ 1(Z)	σ
	ADF6	01D	?NC XQ	
	ADF7	060	->1807	[AD2-10]
	ADF8	04E	C=0 ALL	
	ADF9	35C	PT=12	builds "3" in C
	ADFA	0D0	LD@PT- 3	
	ADFB	025	?NC XQ	
	ADFC	060	->1809	[AD1_10]
	ADFD	0D1	?NC XQ	
	ADFE	064	->1934	[RCSCR]
	ADFF	275	?NC XQ	{A,B} / {M, C}
	AE00	060	->189D	[DV2_13]
	AE01	078	READ 1(Z)	σ
	AE02	13D	?NC XQ	
	AE03	060	->184F	[MP1_10]
	AE04	070	N=C ALL	
	AE05	0B8	READ 2(Y)	ρ
	AE06	10E	A=C ALL	
	AE07	3E1	?NC GO	
	AE08	056	->15F8	[XX>Y?]

Header	AE0A	096	"V"	
Header	AE0B	00E	"N"	Lyapunov
Header	AE0C	010	"P"	Z - σ
Header	AE0D	001	"A"	Y - ρ
Header	AE0E	019	"Y"	X - β
Header	AE0F	00C	"L"	Ángel Martin
LYAPNV	AE10	199	?NC XQ	Checks XYZ - sets DEC mode
	AE11	100	->4066	[CHKST3]
	AE12	0B8	READ 2(Y)	ρ
	AE13	10E	A=C ALL	
	AE14	078	READ 1(Z)	σ
	AE15	135	?NC XQ	σ.ρ
	AE16	060	->184D	[MP2_10]
	AE17	04E	C=0 ALL	
	AE18	35C	PT=12	builds "4" in C
	AE19	110	LD@PT- 4	
	AE1A	13D	?NC XQ	
	AE1B	060	->184F	[MP1_10]
	AE1C	089	?NC XQ	
	AE1D	064	->1922	[STSCR]
	AE1E	078	READ 1(Z)	σ
	AE1F	1FD	?NC XQ	{A,B} = C-1
	AE20	100	->407F	[DECC10]
	AE21	13D	?NC XQ	
	AE22	060	->184F	[MP1_10]
	AE23	0D1	?NC XQ	
	AE24	064	->1934	[RCSCR]
	AE25	031	?NC XQ	add term
	AE26	060	->180C	[AD2_13]
	AE27	305	?NC XQ	
	AE28	060	->18C1	[SQR13]
	AE29	001	?NC XQ	
	AE2A	060	->1800	[ADDONE]
	AE2B	078	READ 1(Z)	σ
	AE2C	025	?NC XQ	
	AE2D	060	->1809	[AD1_10]
	AE2E	089	?NC XQ	
	AE2F	064	->1922	[STSCR]
	AE30	078	READ 1(Z)	σ
	AE31	10E	A=C ALL	
	AE32	0F8	READ 3(X)	β
	AE33	01D	?NC XQ	σ+ β
	AE34	060	->1807	[AD2_10]
	AE35	001	?NC XQ	
	AE36	060	->1800	[ADDONE]
	AE37	025	?NC XQ	
	AE38	060	->1809	[AD1_10]
	AE39	0D1	?NC XQ	
	AE3A	064	->1934	[RCSCR]
	AE3B	275	?NC XQ	{A,B} / {M, C}
	AE3C	060	->189D	[DV2_13]
	AE3D	2BE	C=-C-1 MS	sign change
	AE3E	11E	A=C MS	ditto in 13-digit form
	AE3F	04E	C=0 ALL	
	AE40	35C	PT=12	builds "3" in C
	AE41	0D0	LD@PT- 3	
	AE42	025	?NC XQ	
	AE43	060	->1809	[AD1_10]
	AE44	331	?NC GO	Overflow, DropST, FillXL & Exit
	AE45	002	->00CC	[NFRX]

II. Pendulum Routines

Simple Pendulum - *see: https://en.wikipedia.org/wiki/Pendulum_(mechanics)*

Equations of motion:

$$\frac{d^2\theta}{dt^2} + \frac{g}{\ell}\sin\theta = 0, \quad ; \quad \frac{d\theta}{dt} = \sqrt{\frac{2g}{\ell}(\cos\theta - \cos\theta_0)}.$$

Real period:

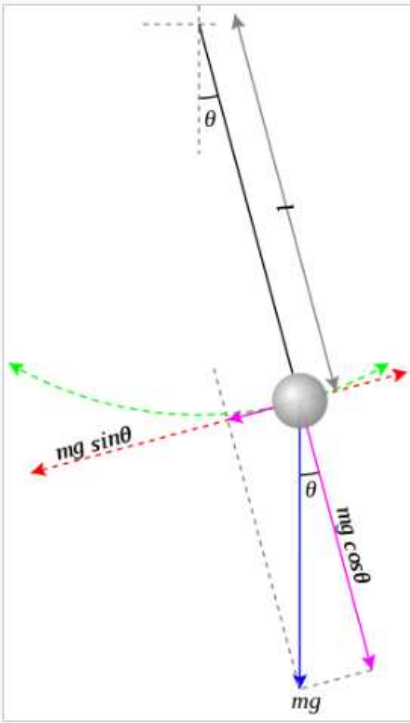
$$T = \frac{2T_0}{\pi}K(k), \qquad \text{where} \quad k = \sin\frac{\theta_0}{2}.$$

Here K(k) is the ***complete*** *elliptic integral of the first kind*, defined by:

$$K(k) = F\left(\frac{\pi}{2}, k\right) = \int_0^{\frac{\pi}{2}} \frac{du}{\sqrt{1 - k^2 \sin^2 u}}.$$

Which can be calculated based on the arithmetic-geometric mean AGM (or **M** in the formula below):

$$T = \frac{2\pi}{M\left(1, \cos\frac{\theta_0}{2}\right)}\sqrt{\frac{\ell}{g}}.$$



Below is the corresponding program as included in the module, based on the Arithmetic-Geometric Mean as the fastest surrogate for K(k). Note that the program prompts for the pendulum parameters and allows for repeat calculations at different initial angles:

1	LBL "PEND'	14	LBL C
2	DEG	15	STO 01
3	"L=? (M)"	16	2
4	PROMPT	17	/
5	9.81	18	COS
6	/	19	1
7	SQRT	20	**AGM**
8	PI	21	RCL 00
9	*	22	X<>Y
10	ST+X(3)	23	/
11	STO 00	24	"T="
12	"<)=? (DEG)"	25	ARCL X
13	PROMPT	26	PROMPT
		27	GTO C
		28	END

Lagrangian and Action

The Lagrangian of the movement of a simple pendulum with mass and length (m, L) at a given position defined by its angle α is given by the expression below:

$$L = m.l\,(2g\cos\alpha - \cos\alpha o)$$

Where αo is the initial position angle of the pendulum.

The function **PLAG** does the calculation of the Lagrangian of the movement using the input parameters from the stack as follows:

T: - m
Z: - L
Y: - θo
X: - θ

Examples for different values of θ on a pendulum with L=m=1 and θo=π/4

	θ	LGR
π/4	0.7854	2.6487 -10
π/5	0.6283	1.9995
π/6	0.5236	3.118
π/7	0.4488	3.8036
π/8	0.3927	4.2531
π/9	0.3491	4.5633
π/10	0.3142	4.7863

Using this function is possible to calculate the *Action between two pendulum positions*, simply using its definition:

$$S = \int_{t_1}^{t_2} L\,\mathrm{d}t\,,$$

The FOCAL routine below does the calculation, aided by the **FINTG** engine from the SandMath. Here the integration variable is not time, but the position angle instead, thus we're implicitly covering only pendulum movements in "not swinging back" trajectories, i.e. the angles are never repeated.

```
LBL "PACT"
"L^M=?"
PROMPT
STO 02
X<>Y
STO 01
"<)0=?"
PROMPT
LBL 00         ; new angle
STO 00         ; θo
ENTER^
CHS            ; θf = -θo
X<>Y           ; integral limits
"#A"           ; integrand's name
FINTG
"A="
ARCL X
PROMPT         ; show result
GTO 00         ; new angle in X
LBL "#A"       ; integrand
RCL 01         ; m
RCL 02         ; L
RCL 00         ; θo
R^             ; θ in X
PLAG
END
```

See below the MCODE listing for the PLAG function:

Header	AD82	087	"G"	T - m
Header	AD83	001	"A"	Z - L
Header	AD84	00C	"L"	Y - θo
Header	AD85	010	"P"	X - θ
PLAG	AD86	179	?NC XQ	Check Data in {X,Y}
	AD87	100	->405E	[CHKST4] - sets DEC
	AD88	0B8	READ 2(Y)	
s θ - cos θo)	AD89	070	N=C ALL	θo
	AD8A	3C4	ST=0	skips [TRGSET]
	AD8B	048	SETF 4	result in RAD
	AD8C	22D	?NC XQ	Cos θo - skipping [TRGSET]
	AD8D	048	->128B	[COS1]
	AD8E	2BE	C=-C-1 MS	
	AD8F	128	WRIT 4(L)	-Cos θo
	AD90	0F8	READ 3(X)	
	AD91	070	N=C ALL	θ
	AD92	3C4	ST=0	skips [TRGSET]
	AD93	048	SETF 4	result in RAD
	AD94	22D	?NC XQ	Cos θ - skipping [TRGSET]
	AD95	048	->128B	[COS1]
	AD96	138	READ 4(L)	
	AD97	025	?NC XQ	cos θ - cos θo
	AD98	060	->1809	[AD1_10]
	AD99	260	SETHEX	
	AD9A	349	PORT DEP:	multiplies {A,B} by g
	AD9B	08C	XQ	leaves result in {A,B}
	AD9C	35C	->A35C	[G*AB] - returns in DEC
	AD9D	025	?NC XQ	2.g.(cos θ - cos θo)
	AD9A	060	->1809	[AD1_10]
	AD9F	078	READ 1(Z)	l
	ADA0	13D	?NC XQ	2.g.L.(cos θ - cos θo)
	ADA1	060	->184F	[MP1_10]
	ADA2	046	C=0 S&X	
	ADA3	270	RAMSLCT	select Chip0
	ADA4	038	READATA	m
	ADA5	13D	?NC XQ	2.g.L.m.(cos θ - cos θo)
	ADA6	060	->184F	[MP1_10]
	ADA7	331	?NC GO	Overflow, DropST, FillXL & Exit
	ADA8	002	->00CC	[NFRX]
G*AB	A35C	2A0	SETDEC	
	A35D	04E	C=0 ALL	
	A35E	35C	PT= 12	
	A35F	250	LD@PT- 9	g = 9.81 m/s^2
	A360	210	LD@PT- 8	
	A361	050	LD@PT- 1	
	A362	13D	?NC GO	g.sin θ1 <or> g.cos θ1
	A363	062	->184F	[MP1_10]

Position and Lagrangian of Complex Pendulums

The table below summarizes the different routines used in the Pendulum section. Note how the three cases use the same FOCAL subroutine needed for the Runge-Kutta approximation. They are further discriminated by the user flags as shown on the table as well.

Type	**FOCAL Driver**	**FOCAL Sub**	**Flags**	**MCODE Equations**
Double	DBLPEN		SF 00	**dW/dT**
Elastic	ELTPEN	DP/DT	SF 01	**dE/dT**
Magnetic	MAGPEN		SF 02	**dM/dT**

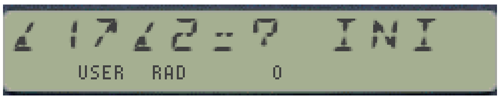
All cases use a system of two non-linear differential equations to define the equations of movement. They can be resolved using Runge-Kuta on a system of four linear ordinary differential equations (ODE), where two of them are simply the defining relationships between the angle and the angular speed, or the space and the velocity depending on the case.

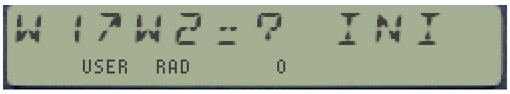
Pendulum Parameters

Magnitude	**Double**	**Elastic**	**Magnetic**
Length	L1, L1	L	d
Mass	m1, m2	m	-
Force Coeffs.	-	k	C, R
Geometry	-	-	x1,y1 ; x2,y2; x3,y3
Boundary	{θ1, θ2}; {w1, w2}	{ θ, r } ; { w, V }	{ x, y } ; { Vx, Vy }

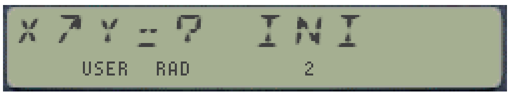
All appropriate parameters are entered at the prompts presented by the driver programs.

The driver program will also prompt for the initial boundary conditions of the problem to solve, as shown in the screen captures below:



,


,

,


The equations of movement are obtained by the application of the Euler-Lagrange equations (one per each generalized coordinate) below on the Lagrangian of the system, which is calculated first as the difference between kinetic and potential energies.

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\alpha}_i} - \frac{\partial L}{\partial \alpha_i} = 0, \quad i = 1, 2.$$

© 2022 ÁNGEL MARTIN

Double Pendulum

Equations of motion and Lagrangian:

$$\omega_1' = \frac{-g\,(2\,m_1 + m_2)\sin\theta_1 - m_2\,g\sin(\theta_1 - 2\,\theta_2) - 2\sin(\theta_1 - \theta_2)\,m_2\,(\omega_2{}^2\,L_2 + \omega_1{}^2\,L_1\cos(\theta_1 - \theta_2))}{L_1\,(2\,m_1 + m_2 - m_2\cos(2\,\theta_1 - 2\,\theta_2))}$$

$$\omega_2' = \frac{2\sin(\theta_1 - \theta_2)\,(\omega_1{}^2\,L_1\,(m_1 + m_2) + g(m_1 + m_2)\cos\theta_1 + \omega_2{}^2\,L_2\,m_2\cos(\theta_1 - \theta_2))}{L_2\,(2\,m_1 + m_2 - m_2\cos(2\,\theta_1 - 2\,\theta_2))}$$

$$L \equiv T - V$$

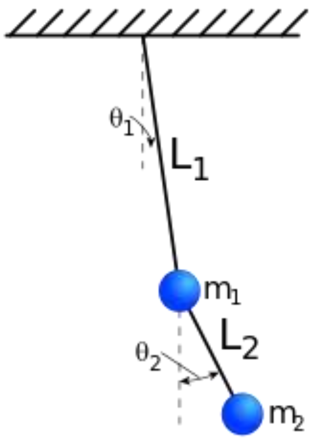
$$= \tfrac{1}{2}(m_1 + m_2)l_1^2\dot\theta_1^2 + \tfrac{1}{2}m_2 l_2^2\dot\theta_2^2 + m_2 l_1 l_2 \dot\theta_1 \dot\theta_2 \cos(\theta_1 - \theta_2)$$

$$+(m_1 + m_2)g l_1 \cos\theta_1 + m_2 g l_2 \cos\theta_2.$$

Example: L1= 3 ; L2 = 4 ; m1 = 1 ; m2 = 2 ; θ1 = p/4 ; θ2 = 0 ; w1 = 0 ; w2 = 0 { h=0.1 ; N = 10 }

t	θ1	θ2	w1	w2	LGR	Action (*)
0.0000	**0.7854**	**0.0000**	**0.0000**	**0.0000**	20.77	20.77
1.0000	-0.3483	0.5105	-0.5431	-0.4265	91.2979	112.07
2.0000	0.1923	-0.7249	-0.0732	-0.4885	0.2586	112.33
3.0000	-0.6805	0.0084	-0.2404	0.7318	-2.4384	109.89
4.0000	0.4984	-0.2370	0.8675	0.1483	-4.5138	105.37
5.0000	-0.0806	0.7789	-0.4737	0.4292	54.0948	159.47
6.0000	0.4574	-0.1206	0.5485	-1.2757	162.3469	321.82
7.0000	-0.5885	-0.0081	-1.1494	0.4365	131.3847	453.20
8.0000	0.0219	-0.6180	1.1791	-0.5518	164.2508	617.45
9.0000	-0.1906	0.3207	-0.8399	1.5275	-8.4357	609.02
10.0000	0.4885	0.2704	1.5553	-1.3140	-0.2547	608.76

Sketch:

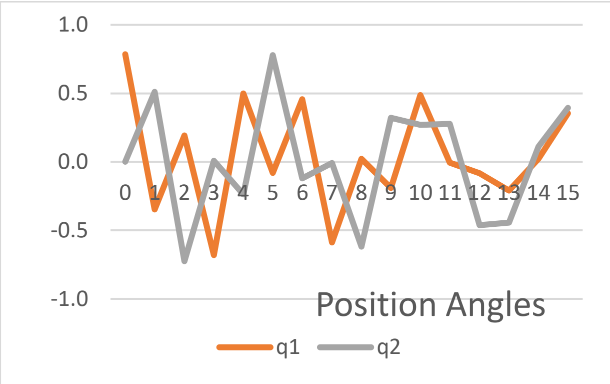


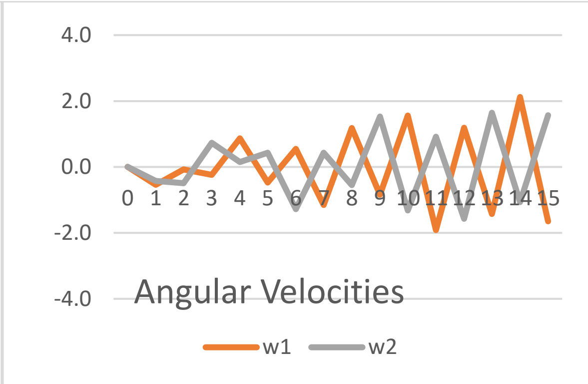
(*) Estimation of the Action.

As a first initial approximation the rectangular rule is used, taking time increments equal to one time unit (1) we'll assume the value of the Lagrangian to be constant within the corresponding time slice, thus the action is the cumulative sums of areas of each rectangle.

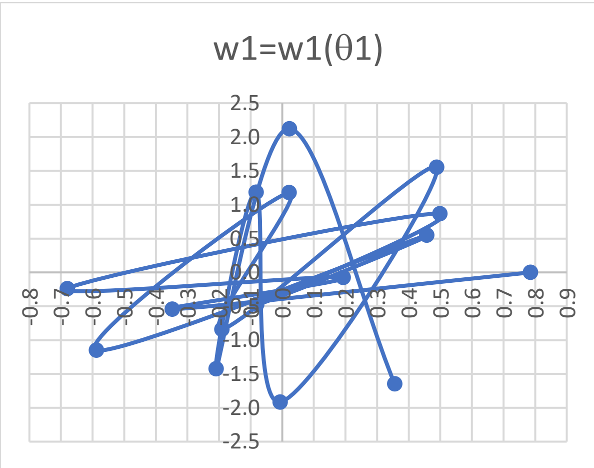
Graphics of the movement.

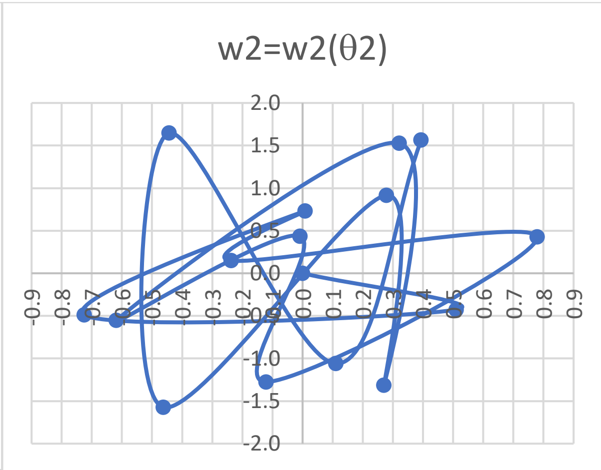
The top charts represent the position angles θ1 and θ2 as a function of time. The sawtooth shapes appear to indicate that a smaller step size should probably be used for a better resolution in the trends.





The bottom charts represent the phase charts for the main and secondary pendulum:





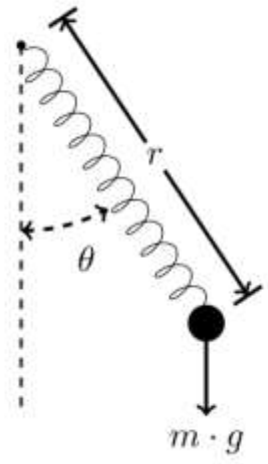
Elastic Pendulum

Equations of motion and Lagrangian:

$$\ddot{\theta} = -\frac{g}{l_0 + x}\sin\theta - \frac{2\dot{x}}{l_0 + x}\dot{\theta} \qquad \ddot{x} = (l_0 + x)\dot{\theta}^2 - \frac{k}{m}x + g\cos\theta$$

$$L[x, \dot{x}, \theta, \dot{\theta}] = \frac{1}{2}m(\dot{x}^2 + (l_0 + x)^2\dot{\theta}^2) - \frac{1}{2}kx^2 + gm(l_0 + x)\cos\theta$$

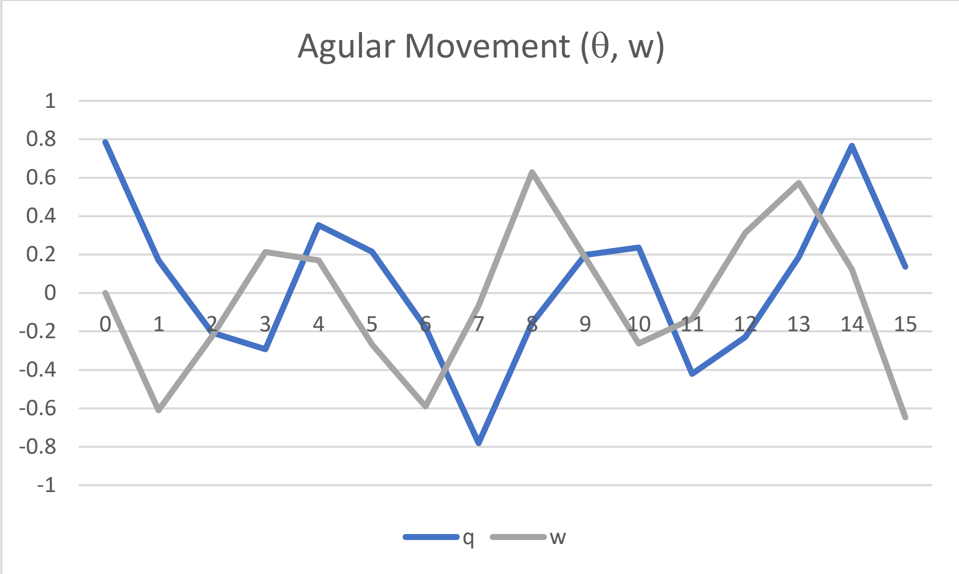
Sketch:

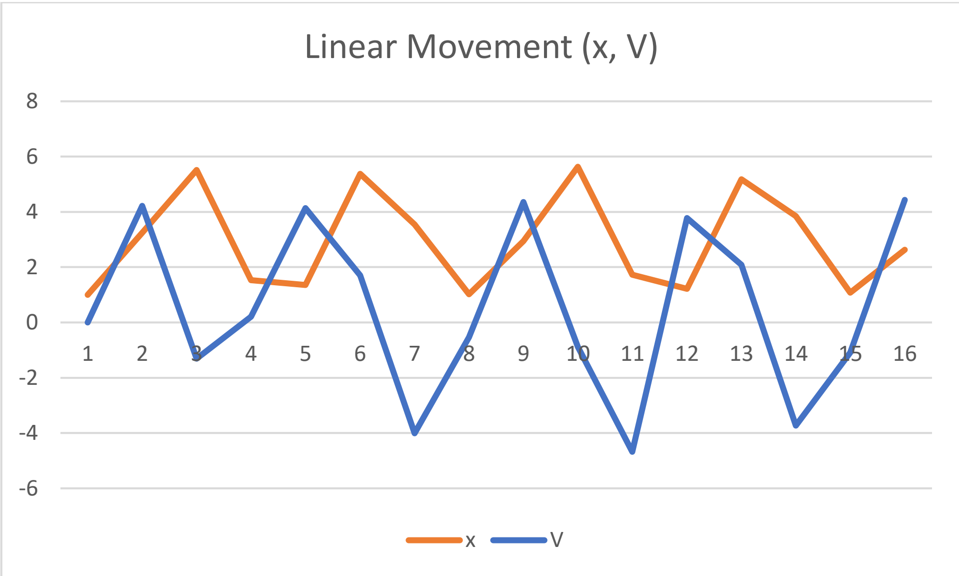


Example: K = 10 ; L = 2 ; Xo = 1 ; θo = π/4 ; ωo = 0 ; Vo = 0 ; { h=0.1 ; N = 10 }

t	θ	x	w	V	LGR	Action
0	**0.785**	**1**	**0**	**0**	-38.4735	-38.4735
1	0.1694	3.2502	-0.6095	4.2184	185.3589	146.8854
2	-0.2051	5.5203	-0.226	-1.309	662.4542	809.3396
3	-0.293	1.529	0.213	0.213	-102.8404	706.4992
4	0.353	1.361	0.171	4.134	-95.1558	611.3434
5	0.215	5.374	-0.267	1.707	2,055.79	2667.1358
6	-0.179	3.548	-0.591	-4.013	1,254.54	3921.6749
7	-0.7808	1.0183	-0.0654	-0.5321	81.5766	4003.2515
8	-0.156	2.9481	0.6287	4.3551	940.9042	4944.1557
9	0.1984	5.6323	0.1854	-0.8806	5,057.70	10001.855
10	0.2362	1.7251	-0.2629	-4.6787	-153.4522	9848.4031
11	-0.4201	1.2206	-0.1347	3.7752	-225.3752	9623.0279
12	-0.2289	5.1844	0.3135	2.0801	7,529.78	17152.81
13	0.1871	3.8495	0.5721	-3.7261	4,557.04	21709.847
14	0.7657	1.0725	0.1239	-1.0903	-473.0527	21236.794
15	0.1361	2.6353	-0.6473	4.4344	1,505.59	22742.385

Charts:


Agular Movement (θ, w)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
q
w


Linear Movement (x, V)
8
6
4
2
0
-2
-4
-6
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
x
V

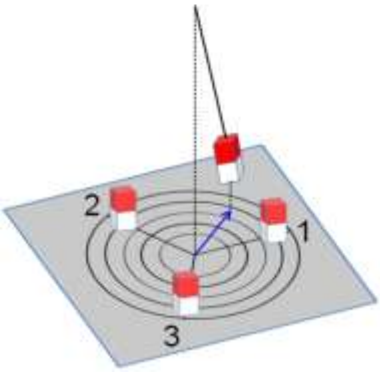
Magnetic Pendulum

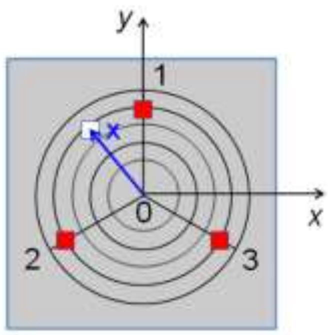
Equations of movement: R is for the air friction, C is proportional to the gravitation force, and D is the distance from the pendulum mass to the plane with the magnets.

$$x'' + Rx' - \sum_i \frac{x_i - x}{((x_i - x)^2 + (y_i - y)^2 + d^2)^{3/2}} + Cx = 0,$$

$$y'' + Ry' - \sum_i \frac{y_i - y}{((x_i - x)^2 + (y_i - y)^2 + d^2)^{3/2}} + Cy = 0.$$

Sketch of a system with 3 magnets:



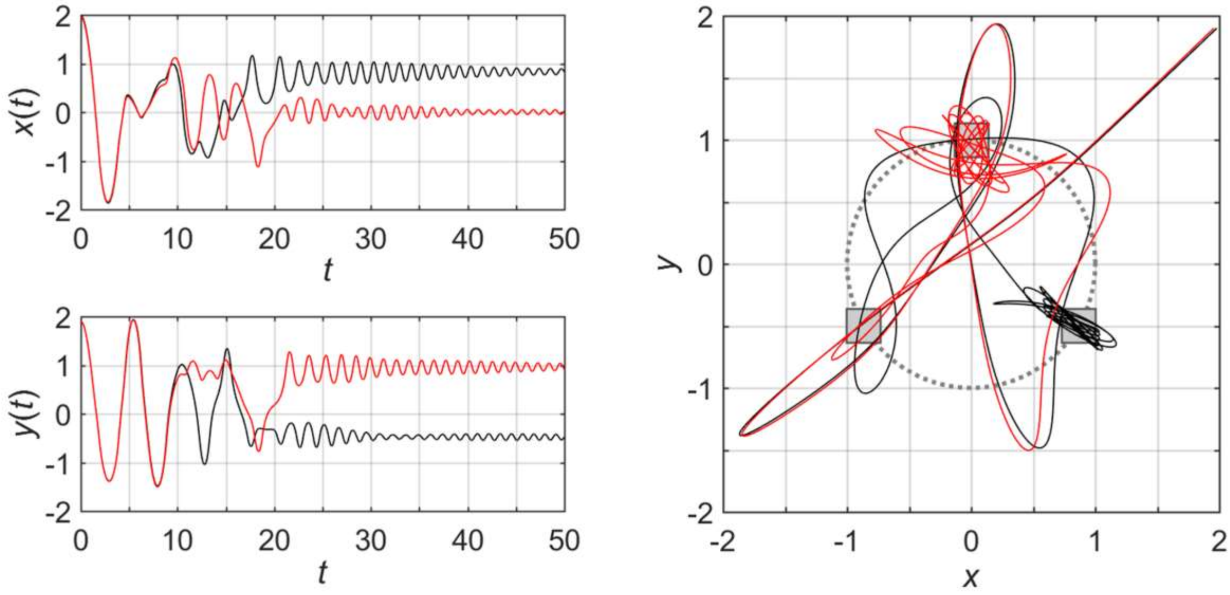


Example: C= 0.2 ; d = 0.2 ; k=0.7 ; with magnets separated 120 degrees

t	x	y	Vx	Vy
0	1	1	0	0
1	0.8911	0.6033	-0.2256	-0.6765
2	0.5396	-0.0619	-0.4783	-0.5615
3	-0.0585	-0.4617	-0.7097	-0.2408
4	-0.8526	-0.5465	-0.8593	0.066
5	-1.7288	-0.3473	-0.8642	0.3151
6	-2.5151	0.06	-0.6725	0.4911
7	-2.9995	0.632	-0.2594	0.6628
8	-2.9654	1.3661	0.3576	0.787
9	-2.2391	2.1374	1.1085	0.6882
10	-0.7451	1.9972	1.8678	-1.8069
11	1.4425	0.6232	2.4643	-1.1145
12	4.0645	-1.0358	2.7043	-1.3506
13	6.6715	-2.3959	2.4079	-1.4466
14	8.6604	-4.0205	1.4555	1.4555
15	9.3584	-6.056	-0.164	-2.2185
16	8.154	-8.3747	-2.3125	-2.361
17	4.6579	-10.6377	-4.6821	-2.0819
18	-1.1298	-12.3559	-6.8068	-1.2591
19	-8.6833	-12.9652	-8.1135	0.1306
20	-16.8876	-11.9412	-8.0162	1.9761
21	-24.0885	-8.9464	-6.0461	4.0145

$$\frac{d^2\mathbf{x}}{dt^2} + b\frac{d\mathbf{x}}{dt} + \mathbf{x} = \sum_{n=1}^{3} \frac{\mathbf{X}_n - \mathbf{x}}{\left(|\mathbf{X}_n - \mathbf{x}|^2 + h^2\right)^{5/2}}.$$

Example from https://chalkdustmagazine.com/features/the-magnetic-pendulum/



MCODE Routines in the Pendula cases.

At the core of the implementation lie the three MCODE routines that define the equations of movement for the three pendula cases: **dW.dT**, **dE/dT**, and **dM/dT**.

They're very long and hard to test but conceptually are rather simple. The main advantages are better accuracy (from using13-digit math routines) and faster execution speed, but the price paid is a much larger ROM space required than using equivalent FOCAL routines.

The driver program includes a triage subroutine (DP/DT) that dispatches the execution to the appropriate MCODE function, depending on the user flag status defined by the chosen case:

Type	FOCAL Driver	FOCAL Sub	Flags	MCODE Equations
Double	DBLPEN		SF 00	**dW/dT**
Elastic	ELTPEN	DP/DT	SF 01	**dE/dT**
Magnetic	MAGPEN		SF 02	**dM/dT**

Refer to the Appendix for the listing of the remaining MCODE routines.

Program listing.-

See below the driver program for all complex pendula cases.

```
01 *LBL "DBLPEN"
02 E
03 GTO 00
04 *LBL "ELTPEN"
05 2
06 GTO 00
07 *LBL "MAGPEN"
08 4
09 *LBL 00
10 X<>F
11 SIZE?
12 36
13 X>Y?
14 PSIZE
15 *LBL A
16 27
17 "L1"
18 FS? 01
19 "L"
20 FS? 02
21 "C"
22 XROM "?"
23 28
24 "L2"
25 FS? 01
26 "K"
27 FS? 02
28 "d"
29 XROM "?"
30 29
31 "M1"
32 FS? 01
33 "M"
34 FS? 02
35 "R"
36 XROM "?"
37 30
38 "M2"
39 FS? 00
40 XROM "?"
41 FC? 02
42 GTO B
43 "X1^Y1=?"
44 PROMPT
45 STO 31
46 X<>Y
47 STO 30
48 "X2^Y2=?"
49 PROMPT
50 STO 33
51 X<>Y
52 STO 32
53 "X3^Y3=?"
54 PROMPT
55 STO 35
56 RDN
57 STO 34
58 *LBL B
59 2
60 "ST. SIZE"
61 XROM "?"
62 3
63 "# STEPS"
64 XROM "?"
65 "a1^a2"
66 FS? 01
67 "a^R"
68 FS? 02
69 "X^Y"
70 >"=? INI"
71 PROMPT
72 STO 12
73 X<>Y
74 STO 11
75 "W1^W2"
76 FS? 01
77 "W^V"
78 FS? 02
79 "VX^VY"
80 >"=? INI"
81 PROMPT
82 STO 14
83 X<>Y
84 STO 13
85 0
86 STO 10
87 4
88 STO 01
89 *LBL C
90 RCL 10
91 "a1"
92 FS? 01
93 "a"
94 FS? 02
95 "X"
96 XROM "="
97 ARCL 11
98 PROMPT
99 "a2"
100 FS? 01
101 "R"
102 FS? 02
103 "Y"
104 XROM "="
105 ARCL 12
106 PROMPT
107 "W1"
108 FS? 01
109 "W"
110 FS? 02
111 "VX"
112 XROM "="
113 ARCL 13
114 PROMPT
115 "W2"
116 FS? 01
117 "V"
118 FS? 02
119 "VY"
120 XROM "="
121 ARCL 14
122 PROMPT
123 "DP/DT"
124 ASTO 00
125 XROM "RK4N"
126 GTO C
127 *LBL "DP/DT"
128 RCL 13
129 STO 15
130 RCL 14
131 STO 16
132 RCL 11
133 RCL 12
134 FS? 00
135 dW/dT
136 FS? 01
137 dE/dT
138 FS? 02
139 dM/dT
140 END
```

III. Attractors Routines

The table below summarizes the different routines used in the Attractors section. Note how the four cases use the same FOCAL subroutine needed for the Runge-Kutta approximation. They are further discriminated by the user flags as shown on the table as well.

Type	**FOCAL Driver**	**FOCAL Sub**	**Flags**	**MCODE Equations**
Lorenz	LORENZ	DL/DT	SF 00	**dL/dT**
Rössler	RSSLER		SF 01	**dR/dT**
Thomas	THOMAS		SF 02	**dT/dT**
Sprott	SPROTT		SF 03	**dS/dT**

All cases use a system of three non-linear differential equations to define the equations of movement. Therefore, they can be resolved using Runge-Kuta on a system of three linear differential equations.

Attractor Parameters

Magnitude	**Lorenz**	**Róssler**	**Thomas**	**Sprott**
Constants	s, r, b	A, B, C	b	a, b
Boundary	xo, yo, zo	xo, yo, zo	xo, yo, zo	xo, yo, zo

The systems are characterized by the following equations,

Lorenz:

$$\begin{cases} \frac{dx}{dt} = \sigma(-x+y) \\ \frac{dy}{dt} = -xz + \rho x - y \\ \frac{dz}{dt} = xy - \beta z \end{cases}$$

Roessler:

$$\begin{cases} \frac{dx}{dt} = -(y+z) \\ \frac{dy}{dt} = x + ay \\ \frac{dz}{dt} = b + z(x-c) \end{cases}$$

Thomas:

$$\begin{cases} \frac{dx}{dt} = \sin y - bx \\ \frac{dy}{dt} = \sin z - by \\ \frac{dz}{dt} = \sin x - bz \end{cases}$$

Sprott:

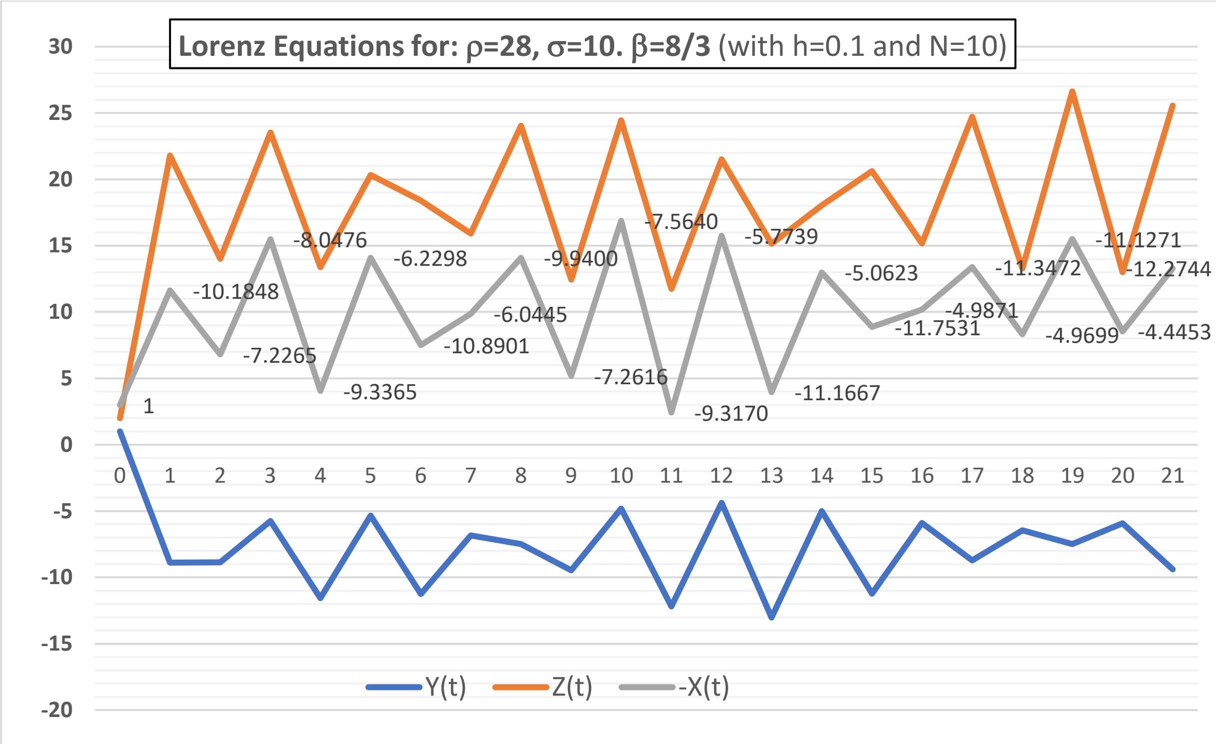
$$\begin{cases} \frac{dx}{dt} = y + axy + xz \\ \frac{dy}{dt} = 1 - bx^2 + yz \\ \frac{dz}{dt} = x - x^2 - y^2 \end{cases}$$

where σ is the Prandtl number, β is a rescaling of the Rayleigh number and ρ is an aspect ratio

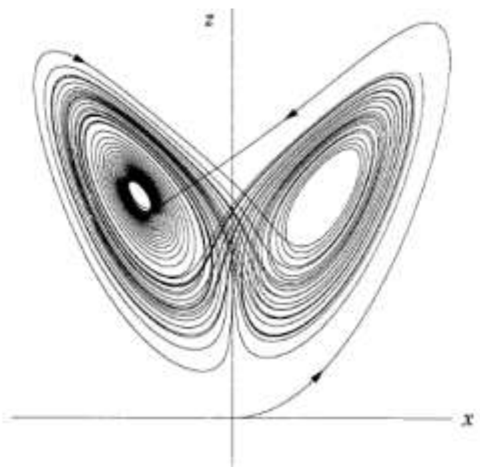
© 2022 ÁNGEL MARTIN

Lorenz Attractor

Example: x0 = y0 = z0 = 1 ; and the usual σ = 10, ρ = 28, β = 8/3



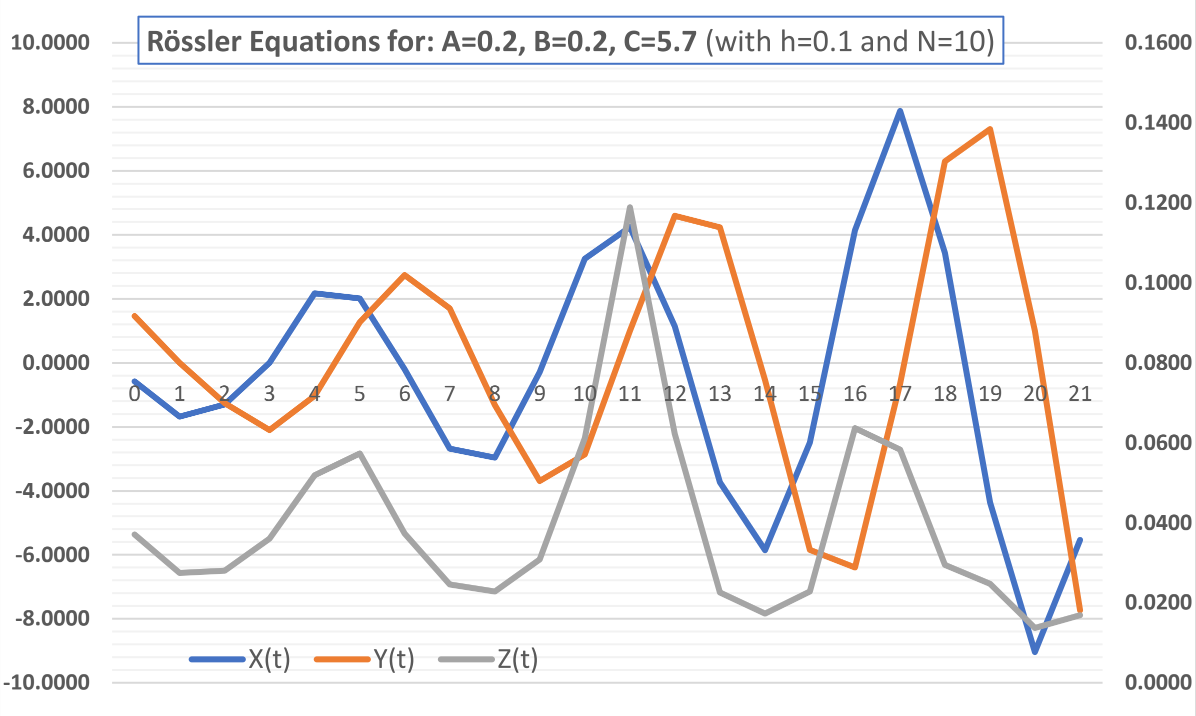
t	-X(t)	Y(t)	Z(t)
0	**1.0000**	**1.0000**	**1.0000**
1	-10.1848	-8.8793	30.6813
2	-7.2265	-8.8616	22.8818
3	-8.0476	-5.7442	29.2743
4	-9.3365	-11.5700	24.9635
5	-6.2298	-5.3340	25.6543
6	-10.8901	-11.2508	29.6540
7	-6.0445	-6.8391	22.7514
8	-9.9400	-7.4969	31.5292
9	-7.2616	-9.4601	21.9084
10	-7.5640	-4.8184	29.2625
11	-9.3170	-12.1841	23.9316
12	-5.7739	-4.3757	25.8939
13	-11.1667	-13.0256	28.1649
14	-5.0623	-5.0058	23.0550
15	-11.7531	-11.2207	31.8494
16	-4.9871	-5.9062	21.0660
17	-11.3472	-8.7287	33.4552
18	-4.9699	-6.4625	19.7479
19	-11.1271	-7.4905	34.1214
20	-4.4453	-5.9135	18.9023
21	-12.2744	-9.3946	34.9572



© 2022 ÁNGEL MARTIN

Rössler Attractor

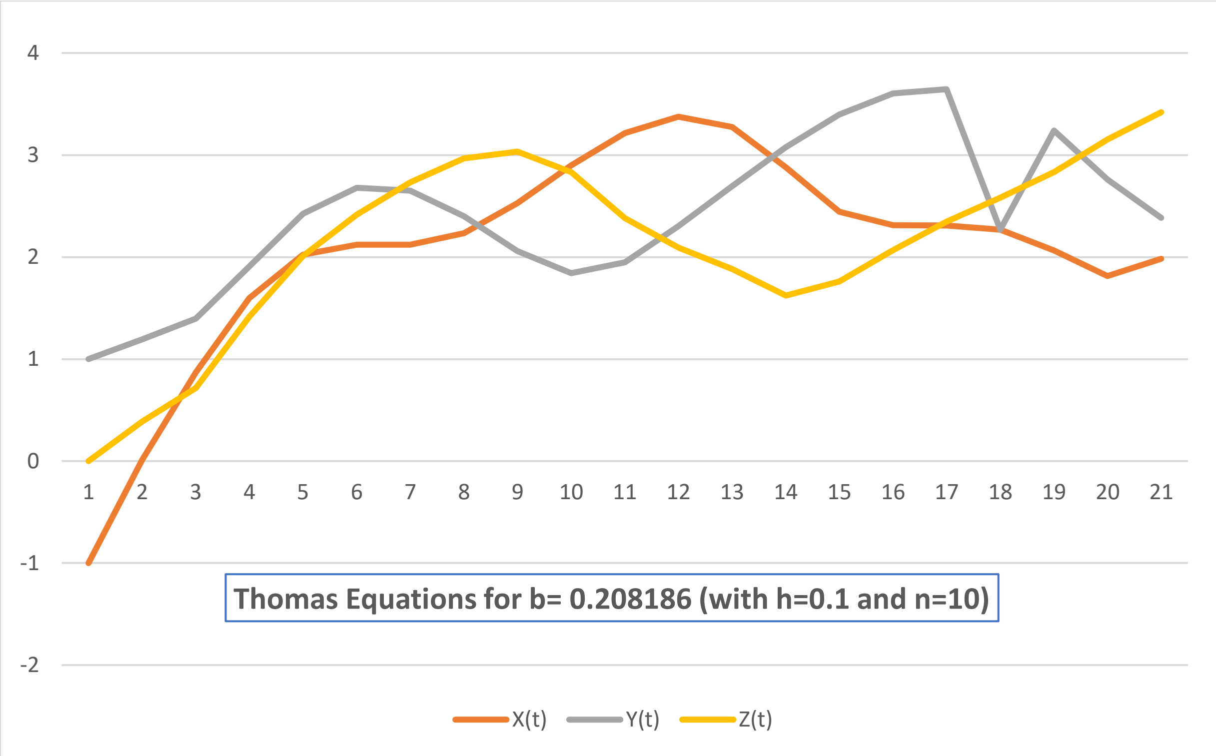
Example: x0 = y0 = z0 = 1 ; and the usual A = 0.2, B = 0.2, C = 5.7



t	X(t)	Y(t)	Z(t)
0	**1.0000**	**1.0000**	**1.0000**
1	-0.5791	1.4585	0.0371
2	-1.6783	0.4584	0.0275
3	-1.3034	-1.2583	0.0280
4	0.4830	-2.1034	0.0360
5	2.1683	-1.0319	0.0519
6	2.0164	1.2754	0.0573
7	-0.2017	2.7396	0.0373
8	-2.6792	1.7003	0.0246
9	-2.9659	-1.3259	0.0228
10	-0.2950	-3.6965	0.0308
11	3.2567	-2.8625	0.0612
12	4.2446	1.0048	0.1189
13	1.1409	4.5979	0.0622
14	-3.7319	4.2374	0.0226
15	-5.8614	-0.5417	0.0173
16	-2.4926	-5.8460	0.0228
17	4.1465	-6.3998	0.0637
18	7.8724	-0.6541	0.0583
19	3.4360	6.2973	0.0295
20	-4.3689	7.3045	0.0247
21	-9.0406	0.9971	0.0137
22	-5.5393	-7.7364	0.0169

Thomas Attractor

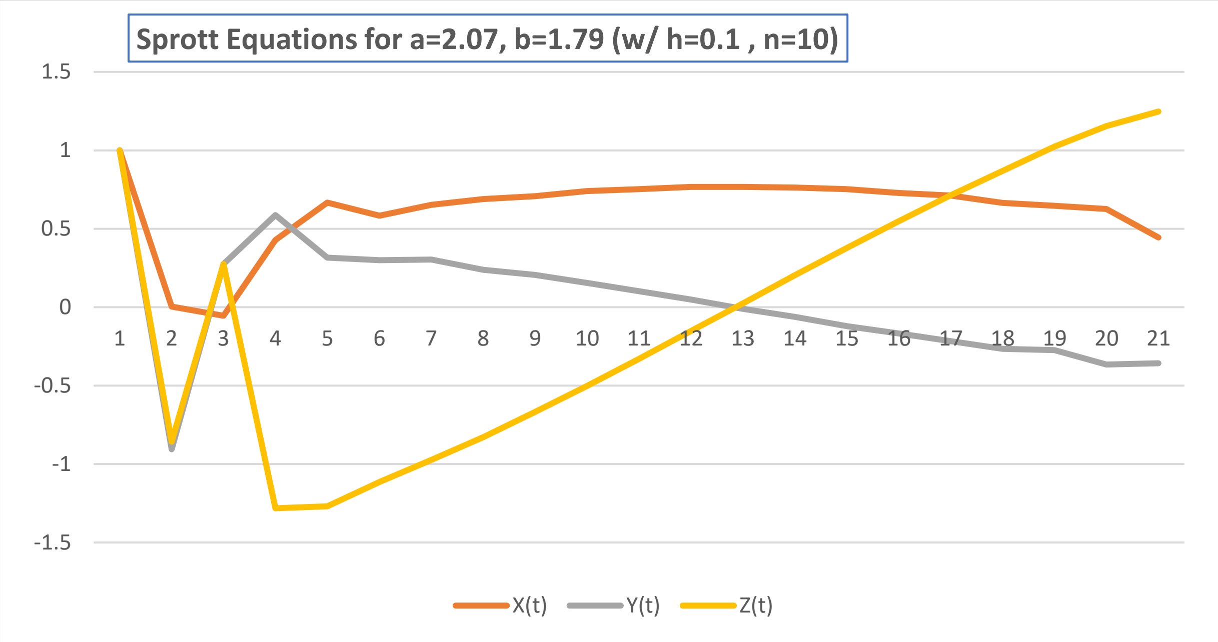
Example: x0 = -1, y0 = 1, z0 = 0 ; and the defaultl b = 0.208186



t	X(t)	Y(t)	Z(t)
0	**-1**	1	0
1	0.0100	1.1943	0.3861
2	0.8710	1.3961	0.7163
3	1.5978	1.9073	1.4154
4	2.0233	2.4253	2.0107
5	2.1220	2.6780	2.4147
6	2.1208	2.6506	2.7313
7	2.2348	2.4010	2.9658
8	2.5281	2.0597	3.0329
9	2.8975	1.8416	2.8310
10	3.2160	1.9498	2.3826
11	3.3754	2.3032	2.0920
12	3.2741	2.6940	1.8831
13	2.8785	3.0758	1.6226
14	2.4432	3.3957	1.7605
15	2.3130	3.6034	2.0667
16	2.3094	3.6449	2.3465
17	2.2694	2.2694	2.5802
18	2.0635	3.2392	2.8333
19	1.8133	2.7607	3.1525
20	1.9826	2.3836	3.4193

Sprott Attractor

Example: x0 = y0 = z0 = 1 ; and the usual a=2.07 ; b = 1.79



t	X(t)	Y(t)	Z(t)
0	**1.0000**	**1.0000**	**1.0000**
1	0.0045	-0.9059	-0.8568
2	-0.0546	0.2756	0.2756
3	0.4283	0.5871	-1.2818
4	0.6678	0.3150	-1.2690
5	0.5827	0.2992	-1.1143
6	0.6520	0.3032	-0.9745
7	0.6887	0.2391	-0.8290
8	0.7085	0.2057	-0.6666
9	0.7409	0.1546	-0.5013
10	0.7526	0.1010	-0.3288
11	0.7665	0.0495	-0.1522
12	0.7674	-0.0100	0.0250
13	0.7627	-0.0612	0.2036
14	0.7521	-0.1199	0.3780
15	0.7278	-0.1669	0.5493
16	0.7117	-0.2161	0.7147
17	0.6653	-0.2656	0.8681
18	0.6456	-0.2745	1.0230
19	0.6261	-0.3663	1.1537
20	0.4454	-0.3567	1.2471

Program listing.-

See below the driver program for all attractors cases.

```
01 *LBL "LORENZ"
02 E
03 GTO 00
04 *LBL "RSSLER"
05 2
06 GTO 00
07 *LBL "THOMAS"
08 4
09 GTO 00
10 *LBL "SPROTT"
11 8
12 *LBL 00
13 X<>F
14 SIZE?
15 15
16 X>Y?
17 PSIZE
18 *LBL A
19 12
20 "A"
21 FS? 00
22 "SIGMA"
23 XROM "?"
24 FS? 02
25 GTO B
26 13
27 "B"
28 FS? 00
29 "RHO"
30 XROM "?"
31 FS? 03
32 GTO B
33 14
34 "C"
35 FS? 00
36 "BETA"
37 XROM "?"
38 *LBL B
39 5
40 "ST. SIZE"
41 XROM "?"
42 6
43 "# STEPS"
44 XROM "?"
45 "Z^Y^X INI=?"
46 PROMPT
47 STO 02
48 RDN
49 STO 03
50 RDN
51 STO 04
52 RDN
53 CLX
54 STO 01
55 *LBL C
56 RCL 04
57 RCL 03
58 RCL 02
59 RCL 01
60 "X"
61 XROM "="
62 ARCL Y
63 PROMPT
64 "Y"
65 XROM "="
66 ARCL Z
67 PROMPT
68 "Z"
69 XROM "="
70 ARCL T
71 PROMPT
72 "DL/DT"
73 ASTO 00
74 XROM "RK4C"
75 GTO C
76 *LBL "DL/DT"
77 FS? 00
78 dL/dT
79 FS? 01
80 dR/dT
81 FS? 02
82 dT/dT
83 FS? 03
84 dS/dT
85 X<> Z
86 END
```

As was the case for the Pendulum routines the real juicy part are the MCODE routines characterizing the equations of motion for each of the attractors: **dL/dT**, **dR/dT**, **dT/dT**, and **dS/dT**. They're rather long and treacherous, but you're encouraged to look at the Appendix for the MCODE listings if so inclined ;-)

System of three first-order ODE: Runge-Kutta Program

Here's the routine as written by Jean-Marc Baillard for the "Differential Equations" module.
See: http://www.hp41.org/LibView.cfm?Command=Search
and http://hp41programs.yolasite.com/sys-diff-eq.php

User instructions in next page.

```
01  *LBL "RK4C"
02  RCL 06
03  STO 10
04  RCL 05
05  2
06  /
07  STO 11
08  *LBL 01
09  RCL 04
10  RCL 03
11  RCL 02
12  RCL 01
13  XEQ IND 00
14  RCL 11
15  ST+ 01
16  ST* Z
17  ST* T
18  *
19  STO 09
20  RCL 04
21  +
22  X<>Y
23  STO 08
24  RCL 03
25  +
26  R^
27  STO 07
28  RCL 02
29  +
30  RCL 01
31  XEQ IND 00
32  RCL 11
33  ST* Z
34  ST* T
35  *
36  ST+ 09
37  ST+ 09
38  RCL 04
39  +
40  X<>Y
41  ST+ 08
42  ST+ 08
43  RCL 03
44  +
45  R^
46  ST+ 07
47  ST+ 07
48  RCL 02
49  +
50  RCL 01
51  XEQ IND 00
52  RCL 11
53  ST+ 01
54  ST+ X
55  ST* Z
56  ST* T
57  *
58  ST+ 09
59  RCL 04
60  +
61  X<>Y
62  ST+ 08
63  RCL 03
64  +
65  R^
66  ST+ 07
67  RCL 02
68  +
69  RCL 01
70  XEQ IND 00
71  RCL 11
72  ST* Z
73  ST* T
74  *
75  RCL 09
76  +
77  3
78  /
79  ST+ 04
80  X<>Y
81  RCL 08
82  +
83  3
84  /
85  ST+ 03
86  R^
87  RCL 07
88  +
89  3
90  /
91  ST+ 02
92  DSE 10
93  GTO 01
94  RCL 04
95  RCL 03
96  RCL 02
97  RCL 01
98  END
```

© 2022 ÁNGEL MARTIN

User Instructions.

"RK4C" solves the system of <u>three</u> first order ODE defined as:

dy/dx = f(x,y,z,u) ,
dz/dx = g(x,y,z,u) ,
du/dx = h(x,y,z,u)

with the initial values: $y(x_0) = y_0$, $z(x_0) = z_0$, $u(x_0) = u_0$

Data Registers:

(Registers R00 to R06 are to be initialized before executing "RK4C")

- R00: function name
- R01 = x_0
- R02 = y_0
- R03 = z_0
- R04 = u_0
- R05 = h = step size
- R06 = N = number of steps
- R07 to R11: scratch

Flags: none/

Subroutine: a program calculating f(x;y;z;u) in Z-register , g(x;y;z;u) in Y-register and h(x;y;z;u) in X-register assuming x is in X-register , y is in Y-register , z is in Z-register and u is in T-register upon entry.

>>>>> In other words, this subroutine must change the stack from

T = u
Z = z ... Z' = f(x;y;z;u)
Y = y to: Y' = g(x;y;z;u)
X = x ... X' = h(x;y;z;u)

STACK	INPUTS	OUTPUTS
T	/	$u(x_0+N.h)$
Z	/	$z(x_0+N.h)$
Y	/	$y(x_0+N.h)$
X	/	$x_0+N.h$

System of "N" first-order ODE: Runge-Kutta Program

Here's the routine as written by Jean-Marc Baillard for the "Differential Equations" module.
See: http://www.hp41.org/LibView.cfm?Command=Search
and http://hp41programs.yolasite.com/sys-diff-eq.php

User instructions in next page.

```
01  *LBL "RK4N"
02  RCL 03
03  STO 09
04  RCL 01
05  RCL 01
06  RCL 01
07  10.01
08  +
09  STO 04
10  INT
11  +
12  STO 05
13  +
14  STO 06
15  +
16  STO 07
17  RCL 05
18  RCL 06
19   E
20  +
21   E3
22  /
23  +
24  CLRGX
25  11
26  LASTX
27  +
28  RCL 01
29   E6
30  /
31  +
32  STO 08
33  GTO 03
34  *LBL 01
35  XEQ IND 00
36  *LBL 02
37  RCL IND 05
38  RCL 02
39  *
40  2
41  /
42  ST+ IND 06
43  FS? 06
44  ST+ IND 06
45  FC? 07
46  ST+ X
47  RCL IND 07
48  +
49  STO IND 04
50  DSE 07
51  DSE 06
52  DSE 05
53  DSE 04
54  GTO 02
55  RCL 01
56  ST+ 04
57  ST+ 05
58  ST+ 06
59  ST+ 07
60  RTN
61  *LBL 03
62  RCL 08
63  REGMOVE
64  CF 06
65  SF 07
66  XEQ 01
67  SF 06
68  RCL 02
69  2
70  /
71  ST+ 10
72  XEQ 01
73  CF 07
74  XEQ 01
75  CF 06
76  RCL 02
77  2
78  /
79  ST+ 10
80  XEQ 01
81  RCL 08
82  REGSWAP
83  RCL 04
84  RCL 06
85  3
86  SIGN
87  *LBL 04
88  CLX
89  X<> IND Y
90  LASTX
91  /
92  ST+ IND Z
93  DSE Y
94  DSE Z
95  GTO 04
96  DSE 09
97  GTO 03
98  RCL 13
99  RCL 12
100 RCL 11
101 RCL 10
102 END
```

Note that the module could have used RK4N for the system of three ODEs as well, simply using N=3 - but I chose to use the dedicated routine instead counting on simpler requirements (flags and user registers) and potentially faster execution.

User Instructions.

"RK4N" solves the following system of "N" first-order ODEs with the "classical" 4th order Runge-Kutta method.

$$\begin{cases} dy_1/dx = f_1(x,y_1, \dots ,y_n) \\ dy_2/dx = f_2(x,y_1, \dots ,y_n) \\ \dots\dots \\ dy_n/dx = f_n(x,y_1, \dots ,y_n) \end{cases} \quad \text{with the initial values:} \quad y_i(x_0) \quad i = 1 , \dots , n$$

Data Registers:

(Registers R00 thru R03 and R10 thru R10+n are to be initialized before executing "RK4N")

- R00 = subroutine name
- R01 = n = number of equations = number of Functions
- R02 = h = step size
- R03 = N = number of steps
- R04 thru R09: scratch
- R10 = x_0
- R11 = $y_1(x_0)$
- R11+n thru R10+4n: scratch
- R12 = $y_2(x_0)$

  .... *(initial values)*
- R10+n = $y_n(x_0)$

Flags: F06-F07

Subroutine: A program that *calculates and stores* $f_1 (x, y_1 , \dots , y_n) , \dots , f_n (x, y_1 , \dots , y_n)$ in registers R11+n, ... , R10+2n respectively with $x, y_1, \dots , y_n$ in registers R10, R11, ... , R10+n

STACK	INPUTS	OUTPUTS
T	/	$y_3(x_0+N.h)$
Z	/	$y_2(x_0+N.h)$
Y	/	$y_1(x_0+N.h)$
X	/	$x_0+N.h$

IV. N-Body Problem *(by Jean-Marc Baillard)*

This section is a subset of the full implementation done on the "N-Body Problem" Module. It only includes Runge-Kutta and Numerov's methos for an Inertial Frame of Reference case. Refer to said module for a more complete treatment that also includes a Multi-Step method and the Heliocentric coordinates for the resolution on each of the three cases..

Method	**Runge-Kutta**	**Numerov**	**Multi-Step**
Inertial FoR	GM	GM2	~~GM3~~
~~Heliocentric~~	~~PLN~~	~~PLN2~~	~~PLN3~~

Two programs are presented to solve the gravitational n-body problem. Both use rectangular coordinates {x,y,z} reckoned to an inertial frame of reference. In the first one ("GM") the method used is a 4^{th}.-order Runge-Kutta. In the second ("GM2") the method used is Numerov's of order 5.

The problem is treated in the Newtonian theory of gravitation and the relativistic effects are not taken into account.

We have to solve a system of 3n second order differential equations of the type: $d^2y/dt^2 = f(y)$
The time t and the first derivative $y' = dy/dt$ don't appear explicitly in this trajectory problem.
So, we can use a special 4th order Runge-Kutta method, namely:

$$y(t+h) = y(t) + h\,(\,y'(t) + k_1/6 + 2k_2/6\,)$$
$$y'(t+h) = y'(t) + k_1/6 + 2k_2/3 + k_3/6$$

where $k_1 = h.f(y)$; $k_2 = h.f(y+h.y'/2+h.k_1/8)$; $k_3 = h.f(y+h.y'+h.k_2/2)$

Alternatively, we can also use multistep methods *if we know 2 or more initial values* (at t , t-h , ...) *without knowing the speeds*, for instance:

Numerov's method of order 5:

$$y_{m+1} = 2.y_m - y_{m-1} + (h^2/12).(\,f_{m+1} + 10.f_m + f_{m-1}\,)$$

1°) 4th-Order Runge-Kutta Method - *Inertial Frame of Reference*

Let $M_1(\,x_1, y_1,z_1)$; .... ; $M_n(\,x_n, y_n,z_n)$ be n celestial bodies with initial velocities
$V_1(\,x'_1, y'_1,z'_1)$; .... ; $V_n(\,x'_n, y'_n,z'_n)$ at an instant t
and m_1 , ... , m_n the n masses of these bodies.

We want to know their future positions and velocities at an instant t + N.h
where(h = step size ; N = number of steps.

So, we have to solve the system: i , j = 1 , ... , n

$$\begin{cases} d^2x_i /dt^2 = \sum_{j\#i} \ G.m_j (x_j - x_i) / [(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2]^{3/2} \\ d^2y_i /dt^2 = \sum_{j\#i} \ G.m_j (y_j - y_i) / [(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2]^{3/2} \\ d^2z_i /dt^2 = \sum_{j\#i} \ G.m_j (z_j - z_i) / [(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2]^{3/2} \end{cases}$$

where G is the gravitational constant.
It may seem very large but it's not too large for the HP-41: if "GM" is executed directly from extended memory, it can solve the 15-body problem.

Data Registers: The following registers are to be initialized before executing "GM":

R00 = n = number of bodies	R27+n = x_1	R27+4n = x'_1
R24 = G = constant of gravitation	R28+n = y_1	R28+4n = y'_1
R25 = h = step size	R29+n = z_1	R29+4n = z'_1
R26 = N = number of steps		
	....	....
R27 = m_1		
R28 = m_2	R24+4n = x_n	R24+7n = x'_n
....	R25+4n = y_n	R25+7n = y'_n
R26+n = m_n	R26+4n = z_n	R26+7n = z'_n

>>> Thus, the n masses, the 3n position coordinates and the 3n velocity coordinates are to be stored into contiguous registers, starting at R27

>>> Registers R27+16n thru R26+19n must be cleared before the first execution (you can XEQ CLRG before storing numbers). Other registers are used for temporary data storage (control numbers , partial sums , k-numbers of the Runge-Kutta scheme ... etc ...)

N.B.

$G = 6.67259\ 10^{-11}\ m^3\ kg^{-1}\ s^{-2}$ but if the units are Astronomical Unit , Solar mass and day ,
$G = k^2$ where k = 0.01720209895 is the Gaussian gravitational constant.

Program Listing

01 LBL "GM"	42 STO 09	83 GTO 00	124 INT	165 RCL 02	
02 LBL 00	43 XEQ 04	84 LBL 04	125 RCL 13	166 ST+ 05	206 LBL 08
03 RCL 00	44 RCL 25	85 26.026	126 X=Y?	167 RCL 03	207 3
04 26	45 2	86 STO 11	127 GTO 08	168 ST+ 06	208 GTO 07
05 +	46 ST* 09	87 3 E-5	128 XEQ 09	169 2	209 LBL 09
06 .1	47 /	88 RCL 22	129 STO 03	170 ST+ 12	210 RCL IND 13
07 %	48 STO 07	89 +	130 1	171 ST+ 14	211 RCL IND 12
08 +	49 4	90 STO 12	131 ST- 12	172 ST+16	212 -
09 RCL 00	50 ST/ 10	91 RCL 21	132 ST- 13	173 SIGN	213 RCL IND 15
10 3	51 /	92 +	133 ST- 14	174 LBL 07	214 RCL IND 14
11 *	52 RCL 25	93 STO 14	134 ST- 15	175 ST- 13	215 -
12 STO 21	53 *	94 LASTX	135 ST- 16	176 ST- 15	216 RCL 07
13 +	54 STO 08	95 +	136 ST- 17	177 ST- 17	217 *
14 STO 22	55 XEQ 04	96 STO 16	137 XEQ 09	178 DSE 11	218 +
15 RCL 26	56 RCL 25	97 LASTX	138 STO 02	179 GTO 06	219 RCL IND 17
16 STO 23	57 STO 07	98 +	139 1	180 RCL 06	220 RCL IND 16
17 LBL 01	58 CLX	99 STO 18	140 ST- 12	181 XEQ 10	221 -
18 RCL 22	59 STO 09	100 LASTX	141 ST- 13	182 RCL 05	222 RCL 08
19 INT	60 4	101 +	142 ST- 14	183 XEQ 10	223 *
20 1	61 ST* 08	102 STO 19	143 ST- 15	184 RCL 04	224 +
21 +	62 ST* 10	103 LASTX	144 ST- 16	185 XEQ 10	225 RTN
22 STO Y	63 XEQ 04	104 +	145 ST- 17	186 3	226 LBL 10
23 RCL 00	64 4	105 STO 20	146 RCL IND 11	187 ST- 14	227 RCL 24
24 9	65 RCL 21	106 LBL 05	147 XEQ 09	188 ST- 16	228 *
25 *	66 ST* Y	107 RCL 00	148 STO 01	189 DSE 12	229 ENTER^
26 +	67 RCL 22	108 ST+ 11	149 X^2	190 GTO 05	230 STO IND 18
27 RCL 21	68 +	109 RCL 22	150 RCL 02	191 RCL 22	231 RCL 09
28 E3	69 ST+ Y	110 INT	151 X^2	192 RCL 21	232 *
29 /	70 ENTER^	111 STO 13	152 RCL 03	193 ST+ X	233 ST+ IND 19
30 +	71 LBL 03	112 RCL 21	153 X^2	194 1	234 CLX
31 E3	72 CLX	113 +	154 +	195 +	235 RCL 10
32 /	73 X<> IND Z	114 STO 15	155 +	196 .1	236 /
33 +	74 RCL 25	115 LASTX	156 ENTER^	197 %	237 ST+ IND 20
34 REGMOVE	75 *	116 +	157 SQRT	198 +	238 1
35 CLX	76 ST+ IND Y	117 STO 17	158 *	199 +	239 ST- 18
36 STO 07	77 DSE Z	118 CLX	159 /	200 RCL 21	240 ST- 19
37 STO 08	78 DSE Y	119 STO 04	160 ST* 01	201 E6	241 ST- 20
38 RCL 25	79 GTO 03	120 STO 05	161 ST* 02	202 /	242 RTN
39 6	80 DSE 23	121 STO 06	162 ST* 03	203 +	243 END
40 STO 10	81 GTO 01	122 LBL 06	163 RCL 01	204 REGMOVE	
41 /	82 RTN	123 RCL 12	164 ST+ 04	205 RTN	

(375 bytes / SIZE 27+19n)

Example: Let's imagine a system of three stars with masses and initial positions and velocities below:

M_1 (2 ; 0 ;0) ; M_2 (0 ; 4 ; 0) ; M_3 (0 ; 0 ; 1) unit = 1 AU
V_1 (0 ; 0.03 ; 0) V_2 (0 ; 0 ; 0.01) V_3 (-0.02 ; 0 ; 0) unit = 1 AU per day
m_1 = 2 ; m_2 = 1 ; m_3 = 3 unit = 1 Solar mass

Calculate the positions and velocities 10 days later.

SIZE 084 (or greater)
XEQ CLRG (or 75.083 XEQ CLRGX if you have an HP-41 CX)

3 bodies	>>>	3	STO 00
constant of gravitation	>>>	17.20209895 E-3 X^2	STO 24
step size h = 10 days	>>>	10	STO 25
number of steps: 1	>>>	1	STO 26

next we store the 3 masses and the 18 position and velocity coordinates as shown below and XEQ "GM"

>>> the new positions and velocities have replaced the previous ones in registers R30 thru R47 (next to last column) Execution time = 1mn 52s

	Register	input	h = 10 ; N=1	h = 5 ; N=2
m	R27=m1	2	undisturbed	undisturbed
	R28=m2	1		
	R29=m3	3		
pos.	R30=x1	2	1.992077590	1.992077585
	R31=y1	0	0.300333861	0.300333570
	R32=z1	0	0.003673761	0.003673682
	--------	-----	--------------	--------------
	R33=x2	0	0.000661665	0.000661669
	R34=y2	4	3.996080594	3.996080575
	R36=z2	0	0.100603408	0.100603412
	--------	-----	--------------	--------------
	R36=x3	0	-0.194938948	-0.194938946
	R37=y3	0	0.001083895	0.001084095
	R38=z3	1	0.997349690	0.997349741
vel.	R39=x'1	0	-0.001550090	-0.001550083
	R40=y'1	0.03	0.030038159	0.030038158
	R41=z'1	0	0.000706688	0.000706684
	--------	-----	--------------	--------------
	R42=x'2	0	0.000132598	0.000132598
	R43=y'2	0	-0.000790384	-0.000790385
	R44=z'2	0.01	0.010117548	0.010117549
	--------	-----	--------------	--------------
	R45=x'3	-0.02	-0.019010806	-0.019010811
	R46=y'3	0	0.000238022	0.000238023
	R47=z'3	0	-0.000510308	-0.000510306

To estimate the accuracy of the results, we can perform the calculation with a half step size (h = 5 days) and N = 2 instead of 1:
We obtain the results in the last column: errors are smaller than 0.000001 AU
To continue with the same h and N , simply press R/S

2°) Numerov's Method - *Inertial Frame of Reference*

Data Registers:

(Registers R00 and R13 thru R15+7n are to be initialized before executing "GM2")

- R00 = n = number of bodies
- R01 to R10: scratch
- R11-R12: unused
- R13 = G = k^2
- R14 = h = step size
- R15 = N = number of steps

- R16 = m_1 thru: • R15+n = m_n
- R17 = m_2

....
- R15+n = m_n

- R16+n = $x_1^{(0)}$ thru: • R13+4n = $x_n^{(0)}$
- R16+4n = $x_1^{(-1)}$ thru: • R13+7n = $x_n^{(-1)}$
- R16+7n thru R15+19n: scratch

- R17+n = $y_1^{(0)}$ thru: • R14+4n = $y_n^{(0)}$
- R17+4n = $y_1^{(-1)}$ thru: • R14+7n = $y_n^{(-1)}$

- R18+ n = $z_1^{(0)}$ thru: • R15+4n = $z_n^{(0)}$
- R18+4n = $z_1^{(-1)}$ thru: • R15+7n = $z_n^{(-1)}$

where m_i are the n masses, $x_i^{(0)}$ $y_i^{(0)}$ $z_i^{(0)}$ are the positions at an instant t
and $x_i^{(-1)}$ $y_i^{(-1)}$ $z_i^{(-1)}$ are the positions at the instant t-h

Flags: none/
Subroutines: none/

Lines 156-166 are three-byte GTOs

Program Listing.

```
01  LBL "GM2"
02  RCL 00
03  7
04  *
05  15
06  +
07  STO 05
08  LASTX
09  RCL 00
10  4
11  *
12  +
13  E3
14  /
15  +
16  STO 04
17  RCL 00
18  19
19  *
20  15
21  +
22  STO 06
23  LASTX
24  RCL 00
25  +
26  .015
27  +
28  STO 07
29  XEQ 05
30  ST- 05
31  E3
32  /
33  ST- 04
34  XEQ 05
35  RCL 06
36  1
37  +
38  RCL Y
39  E6
40  /
41  +
42  ENTER^
43  INT
44  R^
45  -
46  E3
47  /
48  +
49  REGMOVE
50  LBL 01
51  RCL 15
52  STO 10
53  LBL 02
54  1.004006
55  RCL 00
56  *
57  16.016
58  +
59  REGMOVE
60  RCL 00
61  3
62  *
63  GTO 00
64  LBL 03
65  RCL 00
66  4
67  *
68  15
69  +
70  STO 05
71  LASTX
72  RCL 00
73  +
74  E3
75  /
76  +
77  STO 04
78  RCL 00
79  13
80  *
81  15
82  +
83  STO 06
84  XEQ 05
85  LBL 00
86  STO Y
87  RCL 00
88  15
89  +
90  .1
91  %
92  +
93  +
94  STO 01
95  +
96  STO 02
97  +
98  STO 03
99  +
100  STO 04
101  +
102  STO 05
103  +
104  STO 06
105  CLX
106  LBL 04
107  RCL IND 06
108  RCL IND 05
109  10
110  *
111  +
112  RCL IND 04
113  +
114  RCL 14
115  X^2
116  *
117  12
118  /
119  RCL IND 03
120  -
121  RCL IND 02
122  ST+ X
123  +
124  ENTER^
125  X<> IND 01
126  -
127  ABS
128  +
129  DSE 06
130  DSE 05
131  DSE 04
132  DSE 03
133  DSE 02
134  DSE 01
135  GTO 04
136  X#0?
137  GTO 03
138  RCL 04
139  INT
140  RCL 05
141  INT
142  1
143  ST+ Z
144  +
145  E3
146  /
147  +
148  RCL 00
149  6
150  *
151  E6
152  /
153  +
154  REGMOVE
155  DSE 10
156  GTO 02
157  RCL 01
158  3
159  +
160  RCL IND X
161  DSE Y
162  RCL IND Y
163  DSE Z
164  RCL IND Z
165  RTN
166  GTO 01
167  LBL 05
168  CLX
169  STO 01
170  STO 02
171  STO 03
172  LBL 06
173  RCL 04
174  INT
175  RCL 05
176  X=Y?
177  GTO 08
178  RCL IND 07
179  RCL IND 05
180  RCL IND 04
181  -
182  STO 08
183  X^2
184  DSE 04
185  DSE 05
186  RCL IND 05
187  RCL IND 04
188  -
189  STO 09
190  X^2
191  +
192  DSE 04
193  DSE 05
194  RCL IND 05
195  RCL IND 04
196  -
197  STO T
198  X^2
199  +
200  ENTER^
201  SQRT
202  *
203  /
204  ST* 08
205  ST* 09
206  *
207  ST+ 01
208  RCL 09
209  ST+ 02
210  RCL 08
211  ST+ 03
212  2
213  ST+ 04
214  LBL 07
215  DSE 05
216  DSE 07
217  GTO 06
218  RCL 13
219  ST* 01
220  ST* 02
221  RCL 03
222  *
223  STO IND 06
224  DSE 06
225  RCL 02
226  STO IND 06
227  DSE 06
228  RCL 01
229  STO IND 06
230  DSE 06
231  2
232  ST- 04
233  RCL 00
234  ST+ 07
235  3
236  *
237  ST+ 05
238  DSE 04
239  GTO 05
240  RTN
241  LBL 08
242  2
243  ST- 05
244  GTO 07
245  END
```

(362 bytes / SIZE 19n+16)

STACK	INPUTS	OUTPUTS
Z	/	z_1
Y	/	y_1
X	/	x_1

Example: n = 3 stars / $m_1 = 2$; $m_2 = 1$; $m_3 = 3$ (unit = 1 Solar mass)

We have the following coordinates:

masses	t = 0 positions(0)	t = -5 days positions(-1)	(unit = 1 AU)
2 STO 16	2 STO 19	1.997888568 STO 28	x_1
1 STO 17	0 STO 20	-0.149784693 STO 29	y_1
3 STO 18	0 STO 21	0.001032468 STO 30	z_1
	0 STO 22	0.000165879 STO 31	x_2
	4 STO 23	3.999043454 STO 32	y_2
	0 STO 24	-0.049838219 STO 33	z_2
	0 STO 25	0.101352328 STO 34	x_3
	0 STO 26	0.000175311 STO 35	y_3
	1 STO 27	0.999257761 STO 36	z_3

>Evaluate the coordinates of these 3 stars when t = 10 days

n = 3 STO 00 k^2 = 17.20209895 E-3 X^2 STO 13 h = 5 days STO 14 N = 2 STO 15

XEQ "GM2" >>>> x_1 = 1.992077642 = R19 x_2 = 0.000661670 = R22 x_3 = -0.194938984 = R25
RDN y_1 = 0.300333555 = R20 y_2 = 3.996080573 = R23 y_3 = 0.001084105 = R26
RDN z_1 = 0.003673650 = R21 z_2 = 0.100603410 = R24 z_3 = 0.997349763 = R27

Execution time = 4mn58s
The errors are of the order of 6 10^{-8} AU

The positions at t = 5 days are in registers R28 thru R36
Press R/S or XEQ 01 to continue (after changing N in register R15 if needed)

You can also press XEQ "GM2" but it would needlessly re-calculate values which are already stored in the required registers.

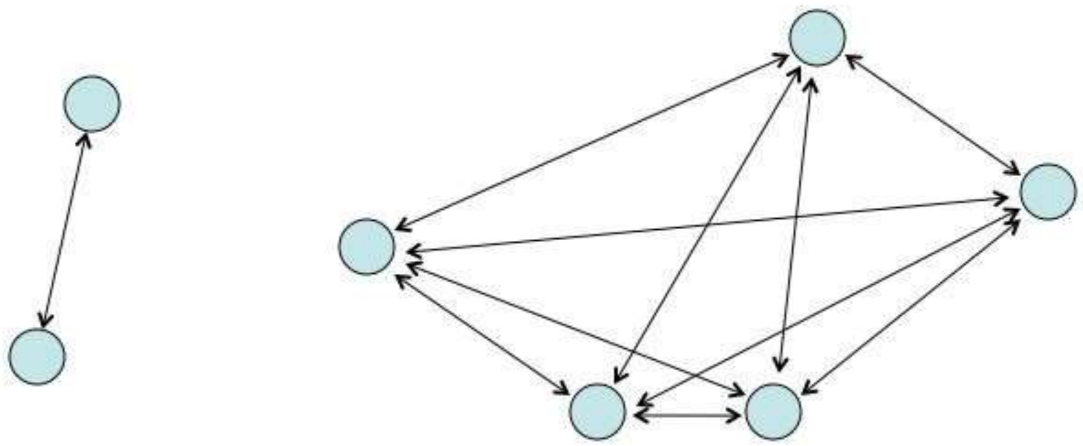
Variant:

The Numerov's formula is applied recursively until 2 successive approximations are equal (line 136).
Therefore, the complicated function (LBL 05) is uselessly calculated at least once.
Alternatively, you can fix the number of iterations: store this value in R11

Replace line 136 by DSE 12
Replace lines 124 to 128 by STO IND 01
Delete line 105
Add RCL 11 STO 12 after line 59

Note:

When h is divided by 2, the errors are approximately divided by $16 = 2^4$



From two bodies to multiple: same concept but radically different consequences.

CODA: - Data Entry driver program.

This module includes a data entry driver to facilitate the process. It automates all the inputs for the system masses and coordinates, thus there's no need to do the error-prone, data register data entry manually.

```
01  *LBL "GM+"
02  0
03  GTO 01
04  *LBL "GM2+"
05  4
06  *LBL 01
07  X<>F
08  "N=?"
09  PROMPT
10  ENTER^
11  ENTER^
12  19
13  *
14  FC? 02
15  27
16  FS? 02
17  16
18  GTO 03
19  *LBL "GM3+"
20  8
21  X<>F
22  "N=?"
23  PROMPT
24  ENTER^
25  ENTER^
26  31
27  *
28  17
29  *LBL 03
30  STO M
31  +
32  SIZE?
33  X<>Y
34  X>Y?
35  PSIZE
36  CLRG
37  RCL Z
38  STO 00
39  RCL M
40  E
41  -
42  STO 05
43  CF 22
44  "G=?"
45  PROMPT
46  FC?C 22
47  <G>
48  STO 14
49  STO 13
50  STO 24
51  "#. STEPS=?"
52  PROMPT
53  STO 16
54  STO 15
55  STO 26
56  "STP. SZE.=?"
57  PROMPT
58  FS? 03
59  STO 15
60  FC? 03
61  STO 14
62  STO 25
63  *LBL A
64  RCL 00
65  E3/E+
66  *LBL 00
67  "M"
68  AINT
69  >"=?"
70  CF 22
71  PROMPT
72  FC?C 22
73  GTO 00
74  RCL Y
75  INT
76  RCL 05
77  +
78  X<>Y
79  STO IND Y
80  RCL Z
81  *LBL 00
82  ISG X
83  GTO 00
84  *LBL B
85  0
86  XEQ 05
87  *LBL C
88  3
89  XEQ 05
90  6
91  FS? 03
92  XEQ 05
93  9
94  FS? 03
95  XEQ 05
96  "GM"
97  FS? 02
98  >"2"
99  FS? 03
100 >"3"
101 7
102 FS? 03
103 13
104 CF 01
105 XROM ">XM"
106 RTN
107 *LBL 05
108 STO 06
109 RCL 00
110 E3/E+
111 *LBL 02
112 "P"
113 RCL 06
114 FS? 03
115 GTO 03
116 X=0?
117 GTO 02
118 "V"
119 FS? 02
120 "P'"
121 GTO 02
122 *LBL 03
123 3
124 /
125 AINT
126 >"-"
127 *LBL 02
128 RDN
129 AINT
130 >"=? X^Y^Z"
131 CF 22
132 PROMPT
133 FC?C 22
134 GTO 02
135 STO N
136 RDN
137 STO M
138 RDN
139 RCL Y
140 INT
141 RCL 06
142 +
143 RCL 00
144 *
145 RCL 05
146 ISG X
147 NOP
148 +
149 X<>Y
150 STO IND Y
151 X<> M
152 ISG Y
153 NOP
154 STO IND Y
155 X<> N
156 ISG Y
157 NOP
158 STO IND Y
159 RCL Z
160 *LBL 02
161 ISG X
162 GTO 02
163 END
```

Note that the driver includes the case for "GM3+" (shaded program lines), available in the full "N-Body Problem" module but not available in this one.

This driver routine makes use of a couple of auxiliary functions, also listed below.

1.- X-Mem Storage

```
01 *LBL ">XM"
02 STO 07
03 RCL 00
04 *
05 E
06 +
07 FC? 01
08 GTO 01
09 RCL 00
10 +
11 E
12 +
13 *LBL 01
14 SF 25
15 PURFL
16 CF 25
17 CRFLD
18 .
19 SEEKPTA
20 RCL 00
21 SAVEX
22 RCL 07
23 *
24 RCL 05
25 +
26 FS? 01
27 RCL 00
28 FS? 01
29 +
30 E3
31 /
32 RCL 05
33 FC? 01
34 ISG X
35 *LBL 10
36 +
37 SAVERX
38 END
```

2.- Gravitational constant:

Header	A1EB	*0BE*	**">"**	***Gravitational Constant***
Header	A1EC	*007*	**"G"**	*in N.m^2/kg*
Header	A1ED	*03C*	**"<"**	*Ángel Martin*
<G>	**A1EE**	**18C**	**?FSET 11**	
	A1EF	*3B5*	*?C XQ*	*Stack lift*
	A1F0	*051*	*->14ED*	*[R_SUB]*
	A1F1	04E	C=0 ALL	
	A1F2	35C	PT= 12	
	A1F3	2A0	SETDEC	
	A1F4	190	LD@PT- 6	
	A1F5	190	LD@PT- 6	
	A1F6	1D0	LD@PT- 7	***6,67428 E-11***
	A1F7	110	LD@PT- 4	*in N.m^2/kg*
	A1F8	090	LD@PT- 2	
	A1F9	210	LD@PT- 8	
	A1FA	130	LDI S&X	
	A1FB	089	CON:	
	A1FC	276	C=C-1 XS	
	A1FD	*0E8*	*WRIT 3(X)*	
	A1FE	3E0	RTN	

3.- ALPHA Integer:

Header	AFB4	094	"T"	
Header	AFB5	00E	"N"	**ALPHA Integer X**
Header	AFB6	009	"I"	
Header	AFB7	001	"A"	Frits Ferwerda
AINT	**AFB8**	**0F8**	**READ 3(X)**	
	AFB9	070	N=C ALL	
	AFBA	361	?NC XQ	
	AFBB	050	->14D8	[CHK$S]
	AFBC	1DD	?NC XQ	
	AFBD	044	->1177	[INT]
	AFBE	0EE	C<>B ALL	
	AFBF	3B8	READ 14(d)	
	AFC0	070	N=C ALL	
	AFC1	05C	PT= 4	
	AFC2	010	LD@R- 0	
	AFC3	210	LD@R- 8	
	AFC4	15C	PT= 6	
	AFC5	010	LD@PT- 0	
	AFC6	3A8	WRIT 14(d)	
	AFC7	0A1	?NC XQ	
	AFC8	018	->0628	[AFORMT]
	AFC9	0B0	C=N ALL	
	AFCA	3A8	WRIT 14(d)	
	AFCB	**3E0**	**RTN**	

4.- Pointer builder:

Header	ADB6	0AB	"+"	
Header	ADB7	005	"E"	**1+X/1000**
Header	ADB8	02F	"/"	
Header	ADB9	033	"3"	
Header	ADBA	005	"E"	Ángel Martin
E3/E+	**ADBB**	**108**	**SETF 8**	
	ADBC	**0F8**	**READ 3(X)**	
	ADBD	361	?NC XQ	(includes SETDEC)
	ADBE	050	->14D8	[CHK_NO_S]
	ADBF	266	C=C-1 S&X	
	ADC0	266	C=C-1 S&X	divide by 1,000
	ADC1	266	C=C-1 S&X	
	ADC2	10E	A=C ALL	holds sign and S&X
	ADC3	02E	B=0 ALL	clears it
	ADC4	0FA	B<>C M	holds 13-digit mant
	ADC5	001	?NC XQ	13-bit form
	ADC6	060	->1800	[ADDONE]
	ADC7	331	?NC GO	Overflow, DropST, FillXL & Exit
	ADC8	002	->00CC	[NFRX]

Appendix 1. MCODE routines for pendulum equations.

1. Double Pendulum Lagrangian.

$$L \equiv T - V$$

$$= \tfrac{1}{2}(m_1 + m_2) l_1^2 \dot{\theta}_1^2 + \tfrac{1}{2} m_2 l_2^2 \dot{\theta}_2^2 + m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2)$$

$$+ (m_1 + m_2) g l_1 \cos\theta_1 + m_2 g l_2 \cos\theta_2.$$

DPLAGR	*Header*	A42B	*087*	***"G"***	
DPLAGR	*Header*	A42C	*001*	***"A"***	*w1 - T*
DPLAGR	*Header*	A42D	*00C*	***"L"***	*w2 - Z*
DPLAGR	*Header*	A42E	*010*	***"P"***	*θ1 - Y*
DPLAGR	*Header*	A42F	*002*	***"B"***	*θ2 - X*
DPLAGR	*Header*	A430	*004*	***"D"***	*Ángel Martin*
DPLAGR	**DBPLAG**	**A431**	*349*	*PORT DEP:*	*Puts L2 in N*
DPLAGR		A432	*08C*	*XQ*	*leaves Chip0 selected*
DPLAGR		A433	*33A*	*->A33A*	*[N=L2] - returns in DEC*
DPLAGR		A434	0B0	C=N ALL	*N is used by [COS1]*
DPLAGR		A435	***128***	***WRIT 4(L)***	*L2*
DPLAGR		A436	0F8	READ 3(X)	*θ2*
DPLAGR		A437	070	N=C ALL	*argument for SIN/COS*
DPLAGR		A438	3C4	ST=0	*skips [TRGSET]*
DPLAGR		A439	048	SETF 4	*result in RAD*
DPLAGR		A43A	*22D*	*?NC XQ*	*cosθ2*
DPLAGR		A43B	*048*	*->128B*	*[COS1]*
DPLAGR		A43C	138	READ 4(L)	*L2*
DPLAGR		A43D	*13D*	*?NC XQ*	*L2.cosθ2*
DPLAGR		A43E	*060*	*->184F*	*[MP1_10]*
DPLAGR		A43F	260	SETHEX	
DPLAGR		A440	*349*	*PORT DEP:*	*multiplies {A,B} by g*
DPLAGR		A441	*08C*	*XQ*	*leaves result in {A,B}*
DPLAGR		A442	*35C*	*->A35C*	*[G*AB] - returns in DEC*
DPLAGR		A443	***128***	***WRIT 4(L)***	*g.L2.cos θ2*
DPLAGR		A444	0F8	READ 3(X)	*θ2*
DPLAGR		A445	2BE	C=-C-1 MS	*Sign change*
DPLAGR		A446	10E	A=C ALL	
DPLAGR		A447	0B8	READ 2(Y)	*θ1*
DPLAGR		A448	*01D*	*?NC XQ*	*θ1-θ2*
DPLAGR		A449	*060*	*->1807*	*[AD2_10]*
DPLAGR		A44A	070	N=C ALL	*argument for SIN/COS*
DPLAGR		A44B	3C4	ST=0	*skips [TRGSET]*
DPLAGR		A44C	048	SETF 4	*result in RAD*
DPLAGR		A44D	*22D*	*?NC XQ*	*cos(θ1-θ2)*
DPLAGR		A44E	*048*	*->128B*	*[COS1] - uses [STSCR]*
DPLAGR		A44F	11E	A=C MS	*bug or what??*
DPLAGR		A450	078	READ 1(Z)	w2
DPLAGR		A451	*13D*	*?NC XQ*	*w2.cos(θ1-θ2)*
DPLAGR		A452	*060*	*->184F*	*[MP1_10]*
DPLAGR		A453	046	C=0 S&X	
DPLAGR		A454	270	RAMSLCT	*select Chip0*
DPLAGR		A455	038	READATA	*w1*
DPLAGR		A456	*13D*	*?NC XQ*	*w1.w2.cos(θ1-θ2)*

DPLAGR	A457	060	->184F	[MP1_10]
DPLAGR	A458	260	SETHEX	
DPLAGR	A459	349	PORT DEP:	Does [STSCR] and Puts L2 in N
DPLAGR	A45A	08C	XQ	leaves Chip0 selected
DPLAGR	A45B	338	->A338	[N=L2] - returns in DEC
DPLAGR	A45C	260	SETHEX	
DPLAGR	A45D	130	LDI S&X	
DPLAGR	A45E	**01B**	**R27**	pointer to **L1**
DPLAGR	A45F	29D	?NC XQ	Get Register Value
DPLAGR	A460	100	->40A7	[READRG#]
DPLAGR	A461	2A0	SETDEC	
DPLAGR	A462	10E	A=C ALL	
DPLAGR	A463	0B0	C=N ALL	L2
DPLAGR	A464	135	?NC XQ	L1.L2
DPLAGR	A465	060	->184D	[MP2_10]
DPLAGR	A466	0C9	?NC XQ	w1.w2.cos(θ1−θ2)
DPLAGR	A467	064	->1932	[RCSCR*] - selects Chip0
DPLAGR	A468	149	?NC XQ	L1.L2.w1.w2.cos(θ1−θ2)
DPLAGR	A469	060	->1852	[MP2_13]
DPLAGR	A46A	**138**	**READ 4(L)**	g.L2.cos θ2
DPLAGR	A46B	025	?NC XQ	g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)
DPLAGR	A46C	060	->1809	[AD1_10]
DPLAGR	A46D	260	SETHEX	
DPLAGR	A46E	349	PORT DEP:	Does [STSCR] and Puts L2 in N
DPLAGR	A46F	08C	XQ	leaves Chip0 selected
DPLAGR	A470	338	->A338	[N=L2] - returns in DEC
DPLAGR	A471	078	READ 1(Z)	w2
DPLAGR	A472	10E	A=C ALL	
DPLAGR	A473	0B0	C=N ALL	L2
DPLAGR	A474	135	?NC XQ	L2.w2
DPLAGR	A475	060	->184D	[MP2_10]
DPLAGR	A476	13D	?NC XQ	(L2.w2)^2
DPLAGR	A477	060	->184F	[MP1_10]
DPLAGR	A478	3D9	?NC XQ	{A,B} = {A,B} /2
DPLAGR	A479	13C	->4FF6	[DIVBY2]
DPLAGR	A47A	0C9	?NC XQ	g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)
DPLAGR	A47B	064	->1932	[RCSCR*] - selects Chip0
DPLAGR	A47C	031	?NC XQ	(1/2).L2^2.w2^2 + g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)
DPLAGR	A47D	060	->180C	[AD2_13]
DPLAGR	A47E	070	N=C ALL	(1/2).L2^2.w2^2 + g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)
DPLAGR	A47F	260	SETHEX	
DPLAGR	A480	130	LDI S&X	
DPLAGR	A481	**01E**	**R30**	pointer to **m2**
DPLAGR	A482	29D	?NC XQ	Get Register Value
DPLAGR	A483	100	->40A7	[READRG#]
DPLAGR	A484	2A0	SETDEC	
DPLAGR	A485	10E	A=C ALL	m2
DPLAGR	A486	046	C=0 S&X	
DPLAGR	A487	270	RAMSLCT	select Chip0
DPLAGR	A488	0B0	C=N ALL	(1/2).L2^2.w2^2 + g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)
DPLAGR	A489	135	?NC XQ	m2.[(1/2).L2^2.w2^2 + g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)]
DPLAGR	A48A	060	->184D	[MP2_10]

DPLAGR	A48B	**228**	**WRIT 8(P)**	m2.[(1/2).L2^2.w2^2 + g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)]
DPLAGR	A48C	260	SETHEX	
DPLAGR	A48D	349	PORT DEP:	Puts L1 in N
DPLAGR	A48E	08C	XQ	leaves Chip0 selected
DPLAGR	A48F	335	->A335	[N=L1] - returns in DEC
DPLAGR	A490	0B0	C=N ALL	N is used by [COS1]
DPLAGR	A491	**128**	**WRIT 4(L)**	L1
DPLAGR	A492	0B8	READ 2(Y)	θ1
DPLAGR	A493	070	N=C ALL	argument for SIN/COS
DPLAGR	A494	3C4	ST=0	skips [TRGSET]
DPLAGR	A495	048	SETF 4	result in RAD
DPLAGR	A496	22D	?NC XQ	cosθ1
DPLAGR	A497	048	->128B	[COS1]
DPLAGR	A498	11E	A=C MS	bug or what??
DPLAGR	A499	138	READ 4(L)	L1
DPLAGR	A49A	13D	?NC XQ	L1.cosθ1
DPLAGR	A49B	060	->184F	[MP1_10]
DPLAGR	A49C	260	SETHEX	
DPLAGR	A49D	349	PORT DEP:	multiplies {A,B} by g
DPLAGR	A49E	08C	XQ	leaves result in {A,B}
DPLAGR	A49F	35C	->A35C	[G*AB] - returns in DEC
DPLAGR	A4A0	**128**	**WRIT 4(L)**	g.L1.cos θ1
DPLAGR	A4A1	260	SETHEX	
DPLAGR	A4A2	349	PORT DEP:	Puts L1 in N
DPLAGR	A4A3	08C	XQ	leaves Chip0 selected
DPLAGR	A4A4	335	->A335	[N=L1] - returns in DEC
DPLAGR	A4A5	038	READATA	w1
DPLAGR	A4A6	10E	A=C ALL	
DPLAGR	A4A7	0B0	C=N ALL	L2
DPLAGR	A4A8	135	?NC XQ	w1.l1
DPLAGR	A4A9	060	->184D	[MP2_10]
DPLAGR	A4AA	13D	?NC XQ	(w1.L1)^2
DPLAGR	A4AB	060	->184F	[MP1_10]
DPLAGR	A4AC	3D9	?NC XQ	{A,B} = {A,B} /2
DPLAGR	A4AD	13C	->4FF6	[DIVBY2]
DPLAGR	A4AE	**138**	**READ 4(L)**	g.L1.cos θ1
DPLAGR	A4AF	025	?NC XQ	g.L1.cos(θ1) + (1/2).L1^2.w1^2
DPLAGR	A4B0	060	->1809	[AD1_10]
DPLAGR	A4B1	260	SETHEX	
DPLAGR	A4B2	349	PORT DEP:	does [STSRC] and recall {m2, m1}
DPLAGR	A4B3	08C	XQ	puts m2 in N, m1 in A and C
DPLAGR	A4B4	343	->A343	[M2NM1A] - returns in DEC
DPLAGR	A4B5	0B0	C=N ALL	m2
DPLAGR	A4B6	01D	?NC XQ	m1+m2
DPLAGR	A4B7	060	->1807	[AD2_10]
DPLAGR	A4B8	0C9	?NC XQ	g.L1.cos(θ1) + (1/2).L1^2.w1^2
DPLAGR	A4B9	064	->1932	[RCSCR*] - selects Chip0
DPLAGR	A4BA	149	?NC XQ	(m1+m2).(g.L1.cosθ1 + (1/2).L1^2.w1^2)
DPLAGR	A4BB	060	->1852	[MP2_13]
DPLAGR	A4BC	**238**	**READ 8(P)**	m2.[(1/2).L2^2.w2^2 + g.L2.cos(θ2) + L1.L2.w1.w2.cos(θ1−θ2)]
DPLAGR	A4BD	025	?NC XQ	
DPLAGR	A4BE	060	->1809	[AD1_10]
DPLAGR	A4BF	331	?NC GO	Overflow, DropST, FillXL & Exit
DPLAGR	A4C0	002	->00CC	[NFRX]

2. *Elastic Pendulum Lagrangian.*

$$L[x,\dot{x},\theta,\dot{\theta}] = \frac{1}{2}m(\dot{x}^2 + (l_0 + x)^2\dot{\theta}^2) - \frac{1}{2}kx^2 + gm(l_0 + x)\cos\theta$$

ELPLAG	*Header*	A3D1	*087*	*"G"*	
ELPLAG	*Header*	A3D2	*001*	*"A"*	***T - w***
ELPLAG	*Header*	A3D3	*00C*	*"L"*	***Z - V***
ELPLAG	*Header*	A3D4	*010*	*"P"*	***Y - θ***
ELPLAG	*Header*	A3D5	*00C*	*"L"*	***X - x***
ELPLAG	*Header*	A3D6	*005*	*"E"*	*Ángel Martin*
ELPLAG	**ELPLAG**	**A3D7**	*179*	*?NC XQ*	*Check Data in {X,Y}*
ELPLAG		A3D8	*100*	*->405E*	*[CHKST4] - sets DEC*
		A3D9	0B8	READ 2(Y)	*θ*
R27 - Lo		A3DA	070	N=C ALL	*argument for SIN/COS*
R28 - k		A3DB	3C4	ST=0	*skips [TRGSET]*
R29 - m		A3DC	048	SETF 4	*result in RAD*
		A3DD	*22D*	*?NC XQ*	*cos θ*
ELPLAG		A3DE	*048*	*->128B*	*[COS1]*
ELPLAG		A3DF	11E	A=C MS	*bug or what??*
ELPLAG		A3E0	***375***	*PORT DEP:*	*multiplies {A,B} by g*
ELPLAG		A3E1	*03C*	*XQ*	*leaves result in {A,B}*
ELPLAG		A3E2	*35C*	*->A35C*	*[G*AB] - returns in DEC*
ELPLAG		A3E3	***375***	*PORT DEP:*	*Does [STSCR] and Puts Lo in N*
ELPLAG		A3E4	*03C*	*XQ*	*leaves Chip0 selected*
ELPLAG		A3E5	*333*	*->A333*	*[N=L1+] - returns in DEC*
ELPLAG		A3E6	0F8	READ 3(X)	x
ELPLAG		A3E7	10E	A=C ALL	
ELPLAG		A3E8	0B0	C=N ALL	*Lo*
ELPLAG		A3E9	*01D*	*?NC XQ*	*Lo+x*
ELPLAG		A3EA	*060*	*->1807*	*[AD2_10]*
ELPLAG		A3EB	***128***	***WRIT 4(L)***	*Lo+x*
ELPLAG		A3EC	*0D1*	*?NC XQ*	*g.cos θ*
ELPLAG		A3ED	*064*	*->1934*	*[RCSCR]*
ELPLAG		A3EE	*149*	*?NC XQ*	*(Lo+x).g.cos θ*
ELPLAG		A3EF	*060*	*->1852*	*[MP2_13]*
ELPLAG		A3F0	*025*	*?NC XQ*	*doubles it*
ELPLAG		A3F1	*060*	*->1809*	*[AD1_10]*
ELPLAG		A3F2	*089*	*?NC XQ*	*2.(Lo+x).g.cos θ*
ELPLAG		A3F3	*064*	*->1922*	*[STSCR]*
ELPLAG		A3F4	046	C=0 S&X	
ELPLAG		A3F5	270	RAMSLCT	*select Chip0*
ELPLAG		A3F6	038	READATA	*w*
ELPLAG		A3F7	10E	A=C ALL	
ELPLAG		A3F8	138	READ 4(L)	*(Lo+x)*
ELPLAG		A3F9	*135*	*?NC XQ*	*w.(Lo+x)*
ELPLAG		A3FA	*060*	*->184D*	*[MP2_10]*
ELPLAG		A3FB	*13D*	*?NC XQ*	*w^2.(Lo+x)^2*
ELPLAG		A3FC	*060*	*->184F*	*[MP1_10]*
ELPLAG		A3FD	*0D1*	*?NC XQ*	*2.(Lo+x).g.cos θ*
ELPLAG		A3FE	*064*	*->1934*	*[RCSCR]*
ELPLAG		A3FF	*031*	*?NC XQ*	*w^2.(Lo+x)^2 + 2.(Lo+x).g.cos θ*
ELPLAG		A400	*060*	*->180C*	*[AD2_13]*
ELPLAG		A401	*089*	*?NC XQ*	*w^2.(Lo+x)^2 + 2.(Lo+x).g.cos θ*

50	ELPLAG	A402	064	->1922	[STSCR]
51	ELPLAG	A403	078	READ 1(Z)	v
52	ELPLAG	A404	10E	A=C ALL	
53	ELPLAG	A405	135	?NC XQ	v^2
54	ELPLAG	A406	060	->184D	[MP2_10]
55	ELPLAG	A407	0D1	?NC XQ	w^2.(Lo+x)^2 + 2.(Lo+x).g.cos θ
56	ELPLAG	A408	064	->1934	[RCSCR]
57	ELPLAG	A409	031	?NC XQ	v^2 + w^2.(Lo+x)^2 + 2.(Lo+x).g.cos θ
58	ELPLAG	A40A	060	->180C	[AD2_13]
59	ELPLAG	A40B	070	N=C ALL	
60	ELPLAG	A40C	260	SETHEX	
61	ELPLAG	A40D	**349**	PORT DEP:	puts m1 in A and C
62	ELPLAG	A40E	08C	XQ	leaves **R29** selected
63	ELPLAG	A40F	34A	->A34A	[A=M1*] - returns in DEC
64	ELPLAG	A410	0B0	C=N ALL	
65	ELPLAG	A411	135	?NC XQ	(m/2).(v^2+(Lo+x)^2.w^2)+2.g(Lo+x).cos θ)
66	ELPLAG	A412	060	->184D	[MP2_10]
67	ELPLAG	A413	081	?NC XQ	(m/2).(v^2+(Lo+x)^2.w^2)+2.g(Lo+x).cos θ)
68	ELPLAG	A414	064	->1920	[STSCR*] - selects Chip0
69	ELPLAG	A415	260	SETHEX	
70	ELPLAG	A416	**349**	PORT DEP:	Puts k in N
71	ELPLAG	A417	08C	XQ	leaves Chip0 selected
72	ELPLAG	A418	33A	->A33A	[N=L2] - returns in DEC
73	ELPLAG	A419	0F8	READ 3(X)	
74	ELPLAG	A41A	10E	A=C ALL	x
75	ELPLAG	A41B	135	?NC XQ	x^2
76	ELPLAG	A41C	060	->184D	[MP2_10]
77	ELPLAG	A41D	0B0	C=N ALL	k
78	ELPLAG	A41E	2BE	C=-C-1 MS	Sign change
79	ELPLAG	A41F	000	NOP	let carry settle
80	ELPLAG	A420	13D	?NC XQ	-k.x^2 /2
81	ELPLAG	A421	060	->184F	[MP1_10]
82	ELPLAG	A422	0D1	?NC XQ	(m/2).(v^2+(Lo+x)^2.w^2)+2.g(Lo+x).cos θ)
83	ELPLAG	A423	064	->1934	[RCSCR]
84	ELPLAG	A424	031	?NC XQ	2,dw/dt
85	ELPLAG	A425	060	->180C	[AD2_13]
86	ELPLAG	A426	3D9	?NC XQ	{A,B} = {A,B} /2
87	ELPLAG	A427	13C	->4FF6	[DIVBY2]
88	ELPLAG	A428	331	?NC GO	Overflow, DropST, FillXL & Exit
89	ELPLAG	A429	002	->00CC	[NFRX]

3. Elastic Pendulum Equations of Movement.

dE/dT	*Header*	*A365*	*094*	**"T"**	***T - w***
dE/dT	*Header*	*A366*	*044*	**"d"**	***Z - V***
dE/dT	*Header*	*A367*	*02F*	**"/"**	***Y - θ***
dE/dT	*Header*	*A368*	*005*	**"E"**	***X - x***
dE/dT	*Header*	*A369*	*044*	**"d"**	*Ángel Martin*
dE/dT	**dE/dT**	**A36A**	*1A5*	*?NC XQ*	*Check Data in {X,Y}*
dE/dT		A36B	*100*	*->4069*	*[CHKST2] - sets DEC*
dE/dT		A36C	0B8	READ 2(Y)	*θ*
dE/dT		A36D	070	N=C ALL	*argument for SIN/COS*
dE/dT		A36E	***375***	*PORT DEP:*	*computes g.sin(N)*
		A36F	*03C*	*XQ*	*leaves result in {A,B}*
R27 - Lo		A370	*353*	*->A353*	*[GSINN] - returns in DEC*
R28 - k		A371	*089*	*?NC XQ*	*g.sin θ*
R29 - m		A372	*064*	*->1922*	*[STSCR]*
		A373	078	READ 1(Z)	v
dE/dT		A374	10E	A=C ALL	
dE/dT		A375	*01D*	*?NC XQ*	*2.v*
dE/dT		A376	*060*	*->1807*	*[AD2_10]*
dE/dT		A377	046	C=0 S&X	
dE/dT		A378	270	RAMSLCT	*select Chip0*
dE/dT		A379	038	READATA	w
dE/dT		A37A	*13D*	*?NC XQ*	*2.v.w*
dE/dT		A37B	*060*	*->184F*	*[MP1_10]*
dE/dT		A37C	*0D1*	*?NC XQ*	*g.sin θ*
dE/dT		A37D	*064*	*->1934*	*[RCSCR]*
dE/dT		A37E	*031*	*?NC XQ*	*2.w.v + g.sin θ*
dE/dT		A37F	*060*	*->180C*	*[AD2_13]*
dE/dT		A380	2BE	C=-C-1 MS	*Sign change*
dE/dT		A381	11E	A=C MS	*ditto in 13-digit form*
dE/dT		A382	***375***	*PORT DEP:*	*Does [STSCR], Puts R27 in N*
dE/dT		A383	*03C*	*XQ*	*leaves Chip0 selected*
dE/dT		A384	*333*	*->A333*	*[N=L1+] - returns in DEC*
dE/dT		A385	0B0	C=N ALL	
dE/dT		A386	10E	A=C ALL	*Lo*
dE/dT		A387	0F8	READ 3(X)	x
dE/dT		A388	*01D*	*?NC XQ*	*Lo+x*
dE/dT		A389	*060*	*->1807*	*[AD2_10]*
dE/dT		A38A	***128***	***WRIT 4(L)***	*Lo+x*
dE/dT		A38B	*0D1*	*?NC XQ*	*-2.w.v - g.sin θ*
dE/dT		A38C	*064*	*->1934*	*[RCSCR]*
dE/dT		A38D	*24D*	*?NC XQ*	*{M,C} / {A,B}*
dE/dT		A38E	*060*	*->1893*	*[X/Y13]*
dE/dT		A38F	**070**	**N=C ALL**	***w'***
dE/dT		A390	260	SETHEX	
dE/dT		A391	130	LDI S&X	
dE/dT		A392	***011***	***R17***	*index to R17*
dE/dT		A393	*29D*	*?NC XQ*	*Get Register Value*
dE/dT		A394	*100*	*->40A7*	*[READRG#]*
dE/dT		A395	*031*	*?NC XQ*	*uses value in N*
dE/dT		A396	*124*	*->490C*	*[WRTSEL]-1 (Selects Chip0)*
dE/dT	***start v'***	A397	2A0	SETDEC	
dE/dT		A398	0B8	READ 2(Y)	*θ*
dE/dT		A399	070	N=C ALL	*argument for SIN/COS*

dE/dT	A400	3C4	ST=0	skips [TRGSET]
dE/dT	A39B	048	SETF 4	result in RAD
dE/dT	A39C	22D	?NC XQ	sin z
dE/dT	A39D	048	->128B	[COS1]
dE/dT	A39E	11E	A=C MS	bug or what??
dE/dT	A39F	**375**	PORT DEP:	multiplies {A,B} by g
dE/dT	A3A0	03C	XQ	leaves result in {A,B}
dE/dT	A3A1	35C	->A35C	[G*AB] - returns in DEC
dE/dT	A3A2	089	?NC XQ	g.cos θ
dE/dT	A3A3	064	->1922	[STSCR]
dE/dT	A3A4	046	C=0 S&X	
dE/dT	A3A5	270	RAMSLCT	select Chip0
dE/dT	A3A6	038	READATA	w
dE/dT	A3A7	10E	A=C ALL	
dE/dT	A3A8	135	?NC XQ	w^2
dE/dT	A3A9	060	->184D	[MP2_10]
dE/dT	A3AA	138	READ 4(L)	Lo+x
dE/dT	A3AB	13D	?NC XQ	(Lo+x).w^2
dE/dT	A3AC	060	->184F	[MP1_10]
dE/dT	A3AD	0D1	?NC XQ	g.cos θ
dE/dT	A3AE	064	->1934	[RCSCR]
dE/dT	A3AF	031	?NC XQ	(Lo+x).w^2 + g.cos θ
dE/dT	A3B0	060	->180C	[AD2_13]
dE/dT	A3B1	089	?NC XQ	(Lo+x).w^2 + g.cos θ
dE/dT	A3B2	064	->1922	[STSCR]
dE/dT	A3B3	260	SETHEX	
dE/dT	A3B4	130	LDI S&X	
dE/dT	A3B5	**01C**	**R28**	pointer to **k**
dE/dT	A3B6	29D	?NC XQ	Get Register Value
dE/dT	A3B7	100	->40A7	[READRG#]
dE/dT	A3B8	2A0	SETDEC	
dE/dT	A3B9	070	N=C ALL	
dE/dT	A3BA	**375**	PORT DEP:	puts m1 in A and C
dE/dT	A3BB	03C	XQ	leaves **R29** selected
dE/dT	A3BC	34A	->A34A	[A=M1*] - returns in DEC
dE/dT	A3BD	0B0	C=N ALL	k
dE/dT	A3BE	0AE	A<>C ALL	k to A, m to C
dE/dT	A3BF	261	?NC XQ	k/m
dE/dT	A3C0	060	->1898	[DV2_10]
dE/dT	A3C1	046	C=0 S&X	
dE/dT	A3C2	270	RAMSLCT	select Chip0
dE/dT	A3C3	0F8	READ 3(X)	x
dE/dT	A3C4	13D	?NC XQ	k.x/m
dE/dT	A3C5	060	->184F	[MP1_10]
dE/dT	A3C6	2BE	C=-C-1 MS	Sign change
dE/dT	A3C7	11E	A=C MS	ditto in 13-digit form
dE/dT	A3C8	0D1	?NC XQ	(Lo+x).w^2 + g.cos θ
dE/dT	A3C9	064	->1934	[RCSCR]
dE/dT	A3CA	031	?NC XQ	-k.x/m + (Lo+x).w^2 + g.cos θ
dE/dT	A3CB	060	->180C	[AD2_13]
dE/dT	A3CC	**070**	**N=C ALL**	**v'**
dE/dT	A3CD	**365**	PORT DEP:	stores N in R18
dE/dT	A3CE	03C	**GO**	and terminates
dE/dT	A3CF	32B	->A32B	[NSTO18]

© 2022 ÁNGEL MARTIN

3. Magnetic Pendulum Equations of Movement

$$x'' + Rx' - \sum_i \frac{x_i - x}{((x_i - x)^2 + (y_i - y)^2 + d^2)^{3/2}} + Cx = 0,$$

$$y'' + Ry' - \sum_i \frac{y_i - y}{((x_i - x)^2 + (y_i - y)^2 + d^2)^{3/2}} + Cy = 0.$$

dM/dT	*Header*	A109	*094*	**"T"**	*T - Vx*
dM/dT	*Header*	A10A	*044*	**"d"**	*Z - Vy*
dM/dT	*Header*	A10B	*02F*	**"/"**	*Y - x*
dM/dT	*Header*	A10C	*00D*	**"M"**	*X - y*
dM/dT	*Header*	A10D	*044*	**"d"**	*Ángel Martin*
dM/dT	**dM/dT**	**A10E**	*1A5*	*?NC XQ*	*Check Data in {X,Y}*
dM/dT		A10F	*100*	*->4069*	*[CHKST2] - sets DEC*
		A110	*345*	*?NC XQ*	*reset initial sums*
R27	**C**	A111	*040*	*->10D1*	*[CLA]*
R28	**d**	A112	260	SETHEX	
R29	**R**	A113	130	LDI S&X	
R30, R31	**x1, y1**	A114	***01E***	***R30***	*X1*
R32, R33	**x2, y2**	A115	*29D*	*?NC XQ*	*Get Register Value*
R34, R35	**x3, y3**	A116	*100*	*->40A7*	*[READRG#]*
		A117	070	N=C ALL	*x1 to N*
dM/dT		A118	046	C=0 S&X	
dM/dT		A119	270	RAMSLCT	*select Chip0*
dM/dT	**x1, y1**	A11A	130	LDI S&X	
dM/dT		A11B	***01F***	***R31***	*Y1*
dM/dT		A11C	*29D*	*?NC XQ*	*Get Register Value*
dM/dT		A11D	*100*	*->40A7*	*[READRG#]*
dM/dT		A11E	2A0	SETDEC	
dM/dT		A11F	10E	A=C ALL	*y1 to A*
dM/dT		A120	***375***	*PORT DEP:*	*puts new Xi sum in O(7)*
dM/dT		A121	*03C*	*XQ*	*puts new Yi sum in P(8)*
dM/dT		A122	*18E*	*->A18E*	*[XYTERM] - returns in DEC*
dM/dT	Σ2	A123	260	SETHEX	
dM/dT		A124	130	LDI S&X	
dM/dT		A125	***020***	***R32***	*X2*
dM/dT		A126	*29D*	*?NC XQ*	*Get Register Value*
dM/dT		A127	*100*	*->40A7*	*[READRG#]*
dM/dT		A128	070	N=C ALL	
dM/dT		A129	046	C=0 S&X	
dM/dT		A12A	270	RAMSLCT	*select Chip0*
dM/dT	**x2, y2**	A12B	130	LDI S&X	
dM/dT		A12C	***021***	***R33***	*Y2*
dM/dT		A12D	*29D*	*?NC XQ*	*Get Register Value*
dM/dT		A12E	*100*	*->40A7*	*[READRG#]*
dM/dT		A12F	2A0	SETDEC	
dM/dT		A130	10E	A=C ALL	
dM/dT		A131	***375***	*PORT DEP:*	*puts new Xi sum in O(7)*
dM/dT		A132	*03C*	*XQ*	*puts new Yi sum in P(8)*
dM/dT		A133	*18E*	*->A18E*	*[XYTERM] - returns in DEC*
dM/dT	Σ3	A134	260	SETHEX	
dM/dT		A135	130	LDI S&X	
dM/dT		A136	***022***	***R34***	*X3*

dM/dT		A137	29D	?NC XQ	Get Register Value
dM/dT		A138	100	->40A7	[READRG#]
dM/dT		A139	070	N=C ALL	
dM/dT		A13A	046	C=0 S&X	
dM/dT		A13B	270	RAMSLCT	select Chip0
dM/dT	**x3, y3**	A13C	130	LDI S&X	
dM/dT		A13D	***023***	***R35***	*Y3*
dM/dT		A13E	29D	?NC XQ	Get Register Value
dM/dT		A13F	100	->40A7	[READRG#]
dM/dT		A140	2A0	SETDEC	
dM/dT		A141	10E	A=C ALL	
dM/dT		A142	**375**	PORT DEP:	puts new Xi sum in O(7)
dM/dT		A143	03C	XQ	puts new Yi sum in P(8)
dM/dT		A144	18E	->A18E	[XYTERM] - returns in DEC
dM/dT	*NEXT*	A145	**375**	PORT DEP:	Puts **R27** (i.e. "C") in N
dM/dT		A146	03C	XQ	leaves Chip0 selected
dM/dT		A147	335	->A335	[N=L1] - returns in DEC
dM/dT		A148	**375**	PORT DEP:	puts **R29** (i.e. "R") in A and C
dM/dT		A149	03C	XQ	leaves **R29** selected
dM/dT		A14A	34C	->A34C	[A=M1] - returns in DEC
dM/dT		A14B	046	C=0 S&X	
dM/dT		A14C	270	RAMSLCT	select Chip0
dM/dT		A14D	038	READATA	Vx
dM/dT		A14E	***1E8***	***WRIT 7(O)***	
dM/dT		A14F	0B8	READ 2(Y)	x
dM/dT		A150	***128***	***WRIT 4(L)***	
dM/dT		A151	0AE	A<>C ALL	***R***
dM/dT		A152	***1A8***	***WRIT 6(N)***	
dM/dT		A153	0B0	C=N ALL	***C***
dM/dT		A154	***168***	***WRIT 5(M)***	
dM/dT		A155	**375**	PORT DEP:	
dM/dT		A156	03C	XQ	
dM/dT		A157	173	->A173	[XFINAL] - returns in DEC
dM/dT		A158	**070**	**N=C ALL**	***Vx'***
dM/dT		A159	260	SETHEX	
dM/dT		A15A	130	LDI S&X	
dM/dT		A15B	***011***	***R17***	*index to R17*
dM/dT		A15C	29D	?NC XQ	Get Register Value
dM/dT		A15D	100	->40A7	[READRG#]
dM/dT		A15E	031	?NC XQ	uses value in N
dM/dT		A15F	124	->490C	[WRTSEL]-1 (Selects Chip0)
dM/dT		A160	078	READ 1(Z)	Vy
dM/dT		A161	***1E8***	***WRIT 7(O)***	
dM/dT		A162	0B8	READ 2(Y)	y
dM/dT		A163	***128***	***WRIT 4(L)***	
dM/dT		A164	**375**	PORT DEP:	
dM/dT		A165	03C	XQ	
dM/dT		A166	171	->A171	[YFINAL] - returns in DEC
dM/dT		A167	**070**	**N=C ALL**	***Vy'***
dM/dT		A168	260	SETHEX	
dM/dT		A169	130	LDI S&X	
dM/dT		A16A	***012***	***R18***	*index to R18*
dM/dT		A16B	29D	?NC XQ	Get Register Value
dM/dT		A16C	100	->40A7	[READRG#]
dM/dT		A16D	031	?NC XQ	uses value in N

102	dM/dT		A16E	124	->490C	[WRTSEL]-1 (Selects Chip0)
103	dM/dT		A16F	3C1	?NC GO	Normal Function Return
104	dM/dT		A170	002	->00F0	[NFRPU]
105	dM/dT	**YFINAL**	**A171**	**088**	**SETF 5**	**dVy/dt**
106	dM/dT		A172	013	JNC +02	
107	dM/dT	**XFINAL**	**A173**	**084**	**CLRF 5**	**dVx/dt**
108	dM/dT	BOTH	A174	2A0	SETDEC	
109	dM/dT		A175	**138**	**READ 4(L)**	x
110	dM/dT		A176	10E	A=C ALL	
111	dM/dT		A177	**178**	**READ 5(M)**	C
112	dM/dT		A178	135	?NC XQ	C.x
113	dM/dT		A179	060	->184D	[MP2_10]
114	dM/dT		A17A	089	?NC XQ	C.x
115	dM/dT		A17B	064	->1922	[STSCR]
116	dM/dT		A17C	**1F8**	**READ 7(O)**	Vx
117	dM/dT		A17D	10E	A=C ALL	
118	dM/dT		A17E	**1B8**	**READ 6(N)**	R
119	dM/dT		A17F	135	?NC XQ	R.Vx
120	dM/dT		A180	060	->184D	[MP2_10]
121	dM/dT		A181	0D1	?NC XQ	C.x
122	dM/dT		A182	064	->1934	[RCSCR]
123	dM/dT		A183	031	?NC XQ	C.x + R.Vx
124	dM/dT		A184	060	->180C	[AD2_13]
125	dM/dT		A185	2BE	C=-C-1 MS	Sign change
126	dM/dT		A186	11E	A=C MS	ditto in 13-digit form
127	dM/dT		A187	08C	?FSET 5	is it for Y?
128	dM/dT		A188	**01F**	**JC +03**	yest, skip over
129	dM/dT		A189	**1F8**	**READ 7(O)**	FINAL x-sum
130	dM/dT		A18A	013	JNC + 02	
131	dM/dT		A18B	**238**	**READ 8(P)**	FINAL y-sum
132	dM/dT		A18C	025	?NC GO	dVx/dt
133	dM/dT		A18D	062	->1809	[AD1_10]
134	dM/dT	**XYTERM**	**A18E**	**2A0**	**SETDEC**	
135	dM/dT		A18F	046	C=0 S&X	
136	dM/dT		A190	270	RAMSLCT	select Chip0
137			A191	0AE	A<>C ALL	
138	*expects xi in N, Yi in A*		A192	**1A8**	**WRIT 6(N)**	Yi
139			A193	0B0	C=N ALL	
140	dM/dT		A194	**168**	**WRIT 5(M)**	Xi
141	dM/dT		A195	0B8	READ 2(Y)	x
142	dM/dT		A196	2BE	C=-C-1 MS	-x
143	dM/dT		A197	10E	A=C ALL	
144	dM/dT		A198	**178**	**READ 5(M)**	Xi
145	dM/dT		A199	01D	?NC XQ	Xi-X
146	dM/dT		A19A	060	->1807	[AD2_10]
147	dM/dT		A19B	**168**	**WRIT 5(M)**	Xi-X
148	dM/dT		A19C	13D	?NC XQ	(Xi-X)^2
149	dM/dT		A19D	060	->184F	[MP1_10]
150	dM/dT		A19E	081	?NC XQ	(Xi-X)^2
151	dM/dT		A19F	064	->1920	[STSCR*] - selects Chip0
152	dM/dT		A1A0	0F8	READ 3(X)	y
153	dM/dT		A1A1	2BE	C=-C-1 MS	-y
154	dM/dT		A1A2	10E	A=C ALL	
155	dM/dT		A1A3	**1B8**	**READ 6(N)**	Yi

156	dM/dT	A1A4	01D	?NC XQ	Yi-y
157	dM/dT	A1A5	060	->1807	[AD2_10]
158	dM/dT	A1A6	**1A8**	**WRIT 6(N)**	Yi-Y
159	dM/dT	A1A7	13D	?NC XQ	(Yi-y)^2
160	dM/dT	A1A8	060	->184F	[MP1_10]
161	dM/dT	A1A9	0C9	?NC XQ	(Xi-x)^2
162	dM/dT	A1AA	064	->1932	[RCSCR*] - selects Chip0
163	dM/dT	A1AB	031	?NC XQ	(Xi-x)^2 + (Yi-y)^2
164	dM/dT	A1AC	060	->180C	[AD2_13]
165	dM/dT	A1AD	089	?NC XQ	(Xi-x)^2 + (Yi-y)^2
166	dM/dT	A1AE	064	->1922	[STSCR]
167	dM/dT	A1AF	260	SETHEX	
168	dM/dT	A1B0	130	LDI S&X	
169	dM/dT	A1B1	**01C**	**R28**	d
170	dM/dT	A1B2	29D	?NC XQ	Get Register Value
171	dM/dT	A1B3	100	->40A7	[READRG#]
172	dM/dT	A1B4	2A0	SETDEC	
173	dM/dT	A1B5	10E	A=C ALL	
174	dM/dT	A1B6	135	?NC XQ	d^2
175	dM/dT	A1B7	060	->184D	[MP2_10]
176	dM/dT	A1B8	0C9	?NC XQ	(Xi-x)^2 + (Yi-y)^2
177	dM/dT	A1B9	064	->1932	[RCSCR*] - selects Chip0
178	dM/dT	A1BA	031	?NC XQ	d^2 + (Xi-X)^2 + (Yi-Y)^2
179	dM/dT	A1BB	060	->180C	[AD2_13]
180	dM/dT	A1BC	070	N=C ALL	
181	dM/dT	A1BD	13D	?NC XQ	squared
182	dM/dT	A1BE	060	->184F	[MP1_10]
183	dM/dT	A1BF	0B0	C=N ALL	
184	dM/dT	A1C0	13D	?NC XQ	cubed
185	dM/dT	A1C1	060	->184F	[MP1_10]
186	dM/dT	A1C2	305	?NC XQ	sqrt(d^2 + (Xi-X)^2 + (Yi-Y)^2)
187	dM/dT	A1C3	060	->18C1	[SQR13]
188	dM/dT	A1C4	239	?NC XQ	1/result
189	dM/dT	A1C5	060	->188E	[ONE_BY_X13]
190	dM/dT	A1C6	**128**	**WRIT 4(L)**	1/result
191	dM/dT	A1C7	**178**	**READ 5(M)**	(Xi-x)
192	dM/dT	A1C8	13D	?NC XQ	(Xi-x)/denom
193	dM/dT	A1C9	060	->184F	[MP1_10]
194	dM/dT	A1CA	**1F8**	**READ 7(O)**	previous x-sum
195	dM/dT	A1CB	025	?NC XQ	add term
196	dM/dT	A1CC	060	->1809	[AD1_10]
197	dM/dT	A1CD	**1E8**	**WRIT 7(O)**	updated x-sum
198	dM/dT	A1CE	**138**	**READ 4(L)**	1/result
199	dM/dT	A1CF	10E	A=C ALL	
200	dM/dT	A1D0	**1B8**	**READ 6(N)**	(Yi-y)
201	dM/dT	A1D1	135	?NC XQ	(Yi-y)/denom
202	dM/dT	A1D2	060	->184D	[MP2_10]
203	dM/dT	A1D3	**238**	**READ 8(P)**	previous y-sum
204	dM/dT	A1D4	01D	?NC XQ	add term
205	dM/dT	A1D5	060	->1807	[AD2_10]
206	dM/dT	A1D6	**228**	**WRIT 8(P)**	updated Y-sum
207	dM/dT	A1D7	3E0	RTN	

4. Double Pendulum Equations of Movement

This one is by far the most complicated of all. Its saving grace is perhaps to be the one with subroutines used by all others, but even so it's a far cry from the equivalent FOCAL routine!

dW/dT	Header	A217	094	"T"	w1 - T
dW/dT	Header	A218	044	"d"	w2 - Z
dW/dT	Header	A219	02F	"/"	θ1 - Y
dW/dT	Header	A21A	017	"W"	θ2 - X
dW/dT	Header	A21B	044	"d"	Ángel Martin
dW/dT	**dW/dT**	**A21C**	1A5	?NC XQ	Check Data in {X,Y}
		A21D	100	->4069	[CHKST2] - sets DEC
L1 - R27		A21E	10E	A=C ALL	θ2
L2 - R28		A21F	01D	?NC XQ	2θ2
m1 - R29		A220	060	->1807	[AD2_10]
m2 - R30		A221	2BE	C=-C-1 MS	Sign change
		A222	11E	A=C MS	ditto in 13-digit form
dW/dT		A223	089	?NC XQ	–2.θ2
dW/dT		A224	064	->1922	[STSCR]
dW/dT		A225	0B8	READ 2(Y)	θ1
dW/dT		A226	10E	A=C ALL	
dW/dT		A227	01D	?NC XQ	2θ1
dW/dT		A228	060	->1807	[AD2_10]
dW/dT		A229	0D1	?NC XQ	–2.θ2
dW/dT		A22A	064	->1934	[RCSCR]
dW/dT		A22B	031	?NC XQ	2θ1-2θ2
dW/dT		A22C	060	->180C	[AD2_13]
dW/dT		A22D	070	N=C ALL	-2.θ2
dW/dT		A22E	3C4	ST=0	skips [TRGSET]
dW/dT		A22F	048	SETF 4	result in RAD
dW/dT		A230	22D	?NC XQ	Cos(2.θ1-2.θ2) - skipping [TRGSET]
dW/dT		A231	048	->128B	[COS1]
dW/dT		A232	2BE	C=-C-1 MS	Sign change
dW/dT		A233	11E	A=C MS	ditto in 13-digit form
dW/dT		A234	001	?NC XQ	1-cos(2θ1-2θ2)
dW/dT		A235	060	->1800	[ADDONE]
dW/dT		A236	070	N=C ALL	1-cos(2θ1-2θ2)
dW/dT		A237	260	SETHEX	
dW/dT		A238	130	LDI S&X	
dW/dT		A239	**01E**	**R30**	pointer to **m2**
dW/dT		A23A	29D	?NC XQ	Get Register Value
dW/dT		A23B	100	->40A7	[READRG#]
dW/dT		A23C	2A0	SETDEC	
dW/dT		A23D	10E	A=C ALL	
dW/dT		A23E	0B0	C=N ALL	1-cos(2θ1-2θ2)
dW/dT		A23F	135	?NC XQ	m2.(1-cos(2θ1-2θ2))
dW/dT		A240	060	->184D	[MP2_10]
dW/dT		A241	081	?NC XQ	m2.(1-cos(2θ1-2θ2))
dW/dT		A242	064	->1920	[STSCR*] - selects Chip0
dW/dT		A243	**375**	PORT DEP:	puts m1 in A and C
dW/dT		A244	03C	XQ	leaves **R29** selected
dW/dT		A245	34C	->A34C	[A=M1] - returns in DEC
dW/dT		A246	01D	?NC XQ	2.m1

dW/dT		A247	060	->1807	[AD2_10]
dW/dT		A248	0C9	?NC XQ	m2.(1-cos(2θ1-2θ2))
dW/dT		A249	064	->1932	[RCSCR*] - selects Chip0
dW/dT		A24A	031	?NC XQ	2m1+m2,(1-cos(2θ1-2θ2))
dW/dT		A24B	060	->180C	[AD2_13]
dW/dT		A24C	**168**	**WRIT 5(M)**	common denominator
dW/dT	**start w1'**	A24D	0F8	READ 3(X)	θ2
dW/dT		A24E	2BE	C=-C-1 MS	Sign change
dW/dT		A24F	10E	A=C ALL	–θ2
dW/dT		A250	0B8	READ 2(Y)	θ1
dW/dT		A251	01D	?NC XQ	θ1-θ2
dW/dT		A252	060	->1807	[AD2_10]
dW/dT		A253	**128**	**WRIT 4(L)**	θ1-θ2 for later…
dW/dT		A254	070	N=C ALL	argument for SIN/COS
dW/dT		A255	3C4	ST=0	skips [TRGSET]
dW/dT		A256	048	SETF 4	result in RAD
dW/dT		A257	22D	?NC XQ	cos(θ1-θ2) - skipping [TRGSET]
dW/dT		A258	048	->128B	[COS1]
dW/dT		A259	**1E8**	**WRIT 7(O)**	cos(θ1-θ2) for later…
dW/dT		A25A	11E	A=C MS	bug or what??
dW/dT		A25B	**375**	PORT DEP:	Does [STSCR] and Puts L1 in N
dW/dT		A25C	03C	XQ	leaves ChIp0 selected
dW/dT		A25D	333	->A333	[N=L1+] - returns in DEC
dW/dT		A25E	038	READATA	w1
dW/dT		A25F	10E	A=C ALL	
dW/dT		A260	135	?NC XQ	w1^2
dW/dT		A261	060	->184D	[MP2_10]
dW/dT		A262	0B0	C=N ALL	L1
dW/dT		A263	13D	?NC XQ	L1.w1^2
dW/dT		A264	060	->184F	[MP1_10]
dW/dT		A265	0C9	?NC XQ	cos(θ1-θ2) - to {C,M}
dW/dT		A266	064	->1932	[RCSCR*] - selects Chip0
dW/dT		A267	149	?NC XQ	L1.w1^2.cos(θ1-θ2)
dW/dT		A268	060	->1852	[MP2_13]
dW/dT		A269	**375**	PORT DEP:	Does [STSCR] and Puts L2 in N
dW/dT		A26A	03C	XQ	leaves ChIp0 selected
dW/dT		A26B	338	->A338	[N=L2] - returns in DEC
dW/dT		A26C	078	READ 1(Z)	w2
dW/dT		A26D	10E	A=C ALL	
dW/dT		A26E	135	?NC XQ	w2^2
dW/dT		A26F	060	->184D	[MP2_10]
dW/dT		A270	0B0	C=N ALL	L2
dW/dT		A271	13D	?NC XQ	L2.w2^2
dW/dT		A272	060	->184F	[MP1_10]
dW/dT		A273	0C9	?NC XQ	L1.w1^2.cos(θ1-θ2)
dW/dT		A274	064	->1932	[RCSCR*] - selects Chip0
dW/dT		A275	031	?NC XQ	L2.w2^2 + L1.w1^2.cos(θ1-θ2)
dW/dT		A276	060	->180C	[AD2_13]
dW/dT		A277	**228**	**WRIT 8(P)**	sin/cos use scratch regs!?
dW/dT		A278	000	NOP	
dW/dT		A279	**138**	**READ 4(L)**	θ1-θ2
dW/dT		A27A	2EE	?C#0 ALL	is it zero?
dW/dT		A27B	033	JNC +06	yes, avoid "-0" issues
dW/dT		A27C	070	N=C ALL	argument for SIN/COS
dW/dT		A27D	3C4	ST=0	skips [TRGSET]

dW/dT		A27E	048	SETF 4	result in RAD
dW/dT		A27F	229	?NC XQ	sin(θ1-θ2)
dW/dT		A280	048	->128A	[SIN1]
dW/dT	ISZERO	A281	10E	A=C ALL	
dW/dT		A282	01D	?NC XQ	2.sin(θ1-θ2)
dW/dT		A283	060	->1807	[AD2_10]
dW/dT		A284	**1A8**	**WRIT 6(N)**	2.sin(θ1-θ2) saved for later…
dW/dT		A285	**238**	**READ 8(P)**	L2.w2^2 + L1.w1^2.cos(θ1-θ2)
dW/dT		A286	000	NOP	
dW/dT		A287	13D	?NC XQ	2.sin(θ2-θ1).[L2.w2^2 + L1…
dW/dT		A288	060	->184F	[MP1_10]
dW/dT		A289	**228**	**WRIT 8(P)**	2.sin(θ2-θ1).[L2.w2^2 + L1…
dW/dT		A28A	000	NOP	
dW/dT		A28B	0F8	READ 3(X)	θ2
dW/dT		A28C	10E	A=C ALL	
dW/dT		A28D	01D	?NC XQ	2.θ2
dW/dT		A28E	060	->1807	[AD2_10]
dW/dT		A28F	2BE	C=-C-1 MS	Sign change
dW/dT		A290	11E	A=C MS	ditto in 13-digit form
dW/dT		A291	0B8	READ 2(Y)	θ1
dW/dT		A292	025	?NC XQ	θ1-2.θ2
dW/dT		A293	060	->1809	[AD1_10]
dW/dT		A294	070	N=C ALL	argument for SIN/COS
dW/dT		A295	**375**	PORT DEP:	computes g.sin(N)
dW/dT		A296	03C	XQ	leaves result in {A,B}
dW/dT		A297	353	->A353	[GSINN] - returns in DEC
dW/dT		A298	**238**	**READ 8(P)**	2.sin(θ2-θ1).[L2.w2^2 + L1…
dW/dT		A299	000	NOP	
dW/dT		A29A	025	?NC XQ	g.sin(θ1-2.θ2) + 2.sin(θ2-θ1)…
dW/dT		A29B	060	->1809	[AD1_10]
dW/dT		A29C	089	?NC XQ	g.sin(θ1-2θ2) + 2.sin(θ2-θ1)…
dW/dT		A29D	064	->1922	[STSCR]
dW/dT		A29E	260	SETHEX	
dW/dT		A29F	130	LDI S&X	
dW/dT		A2A0	**01E**	**R30**	pointer to **m2**
dW/dT		A2A1	29D	?NC XQ	Get Register Value
dW/dT		A2A2	100	->40A7	[READRG#]
dW/dT		A2A3	2A0	SETDEC	
dW/dT		A2A4	070	N=C ALL	
dW/dT		A2A5	046	C=0 S&X	
dW/dT		A2A6	270	RAMSLCT	select Chip0
dW/dT		A2A7	0A9	?NC XQ	g.sin(θ1-2θ2)+2.sin(θ2-θ1).(L2..
dW/dT		A2A8	064	->192A	[EXSCR]
dW/dT		A2A9	0B0	C=N ALL	m2
dW/dT		A2AA	13D	?NC XQ	g.m2.sin(θ1-2θ2)+2.m2.sin…
dW/dT		A2AB	060	->184F	[MP1_10]
dW/dT		A2AC	**128**	**WRIT 4(L)**	g.m2.sin(θ1-2θ2)+2.m2.sin…
dW/dT		A2AD	0B8	READ 2(Y)	θ1
dW/dT		A2AE	070	N=C ALL	argument for SIN/COS
dW/dT		A2AF	**375**	PORT DEP:	computes g.sin(N)
dW/dT		A2B0	03C	XQ	leaves result in {A,B}
dW/dT		A2B1	353	->A353	[GSINN] - returns in DEC
dW/dT		A2B2	**375**	PORT DEP:	does [STSRC] and recall {m2, m1}
dW/dT		A2B3	03C	XQ	puts m2 in N, m1 in A and C
dW/dT		A2B4	343	->A343	[M2NM1A] - returns in DEC

dW/dT		A2B5	01D	?NC XQ	2.m1
dW/dT		A2B6	060	->1807	[AD2_10]
dW/dT		A2B7	0B0	C=N ALL	m2
dW/dT		A2B8	025	?NC XQ	2.m1+m2
dW/dT		A2B9	060	->1809	[AD1_10]
dW/dT		A2BA	0C9	?NC XQ	g.sinθ1
dW/dT		A2BB	064	->1932	[RCSCR*] - selects Chip0
dW/dT		A2BC	149	?NC XQ	(m2+2m1).g.sinθ1
dW/dT		A2BD	060	->1852	[MP2_13]
dW/dT		A2BE	**138**	**READ 4(L)**	g.m2.sin(θ1-2θ2)+2.m2.sin...
dW/dT		A2BF	025	?NC XQ	(m2+2m1).g.sinθ1 + g.m2.sin...
dW/dT		A2C0	060	->1809	[AD1_10]
dW/dT		A2C1	2BE	C=-C-1 MS	Sign change
dW/dT		A2C2	11E	A=C MS	ditto in 13-digit form
dW/dT		A2C3	**178**	**READ 5(M)**	common denominator
dW/dT		A2C4	269	?NC XQ	
dW/dT		A2C5	060	->189A	[DV1_10]
dW/dT		A2C6	**375**	PORT DEP:	Does [STSCR} ad Puts L1 in N
dW/dT		A2C7	03C	XQ	leaves Chip0 selected
dW/dT		A2C8	333	->A333	[N=L1+] - returns in DEC
dW/dT		A2C9	0A9	?NC XQ	{A,B} <-> {Q,+}
dW/dT		A2CA	064	->192A	[EXSCR]
dW/dT		A2CB	0B0	C=N ALL	L1
dW/dT		A2CC	269	?NC XQ	w1' at last!
dW/dT		A2CD	060	->189A	[DV1_10]
dW/dT		A2CE	**070**	**N=C ALL**	**w1'**
dW/dT		A2CF	260	SETHEX	
dW/dT		A2D0	130	LDI S&X	
dW/dT		A2D1	**011**	**R17**	index to R17
dW/dT		A2D2	29D	?NC XQ	Get Register Value
dW/dT		A2D3	100	->40A7	[READRG#]
dW/dT		A2D4	031	?NC XQ	uses value in N
dW/dT		A2D5	124	->490C	[WRTSEL]-1 (Selects Chip0)
dW/dT	**start w2'**	A2D6	130	LDI S&X	
dW/dT		A2D7	**01E**	**R30**	pointer to **m2**
dW/dT		A2D8	29D	?NC XQ	Get Register Value
dW/dT		A2D9	100	->40A7	[READRG#]
dW/dT		A2DA	2A0	SETDEC	
dW/dT		A2DB	070	N=C ALL	m2 in N
dW/dT		A2DC	046	C=0 S&X	
dW/dT		A2DD	270	RAMSLCT	select Chip0
dW/dT		A2DE	078	READ 1(Z)	w2
dW/dT		A2DF	10E	A=C ALL	w2
dW/dT		A2E0	135	?NC XQ	w2^2
dW/dT		A2E1	060	->184D	[MP2_10]
dW/dT		A2E2	0B0	C=N ALL	m2
dW/dT		A2E3	13D	?NC XQ	m2.w2^2
dW/dT		A2E4	060	->184F	[MP1_10]
dW/dT		A2E5	070	N=C ALL	m2.w2^2
dW/dT		A2E6	260	SETHEX	
dW/dT		A2E7	130	LDI S&X	
dW/dT		A2E8	**01C**	**R28**	pointer to **L2**
dW/dT		A2E9	29D	?NC XQ	Get Register Value
dW/dT		A2EA	100	->40A7	[READRG#]
dW/dT		A2EB	2A0	SETDEC	

dW/dT	A2EC	10E	A=C ALL	*L2*
dW/dT	A2ED	046	C=0 S&X	
dW/dT	A2EE	270	RAMSLCT	*select Chip0*
dW/dT	A2EF	0B0	C=N ALL	*m2.w2^2*
dW/dT	A2F0	*135*	*?NC XQ*	*L2.m2.w2^2*
dW/dT	A2F1	*060*	*->184D*	*[MP2_10]*
dW/dT	A2F2	***1F8***	***READ 7(O)***	*cos(θ1-θ2)*
dW/dT	A2F3	*13D*	*?NC XQ*	*L2.m2.w2^2.cos(θ1-θ2)*
dW/dT	A2F4	*060*	*->184F*	*[MP1_10]*
dW/dT	A2F5	***1E8***	***WRIT 7(O)***	*L2.m2.w2^2.cos(θ1-θ2)*
dW/dT	A2F6	0B8	READ 2(Y)	*θ1*
dW/dT	A2F7	070	N=C ALL	*argument for SIN/COS*
dW/dT	A2F8	3C4	ST=0	*skips [TRGSET]*
dW/dT	A2F9	048	SETF 4	*result in RAD*
dW/dT	A2FA	*22D*	*?NC XQ*	*cosθ1 - skipping [TRGSET]*
dW/dT	A2FB	*048*	*->128B*	*[COS1]*
dW/dT	A2FC	11E	A=C MS	*bug or what??*
dW/dT	A2FD	***375***	*PORT DEP:*	*multiplies {A,B} by g*
dW/dT	A2FE	*03C*	*XQ*	*leaves result in {A,B}*
dW/dT	A2FF	*35C*	*->A35C*	*[G*AB] - returns in DEC*
dW/dT	A300	***375***	*PORT DEP:*	*Does [STSCR] and Puts L1 in N*
dW/dT	A301	*03C*	*XQ*	*leaves Chip0 selected*
dW/dT	A302	*333*	*->A333*	*[N=L1+] - returns in DEC*
dW/dT	A303	038	READATA	*w1*
dW/dT	A304	10E	A=C ALL	*w1*
dW/dT	A305	*135*	*?NC XQ*	*w1^2*
dW/dT	A306	*060*	*->184D*	*[MP2_10]*
dW/dT	A307	0B0	C=N ALL	*L1*
dW/dT	A308	*13D*	*?NC XQ*	*L1.w1^2*
dW/dT	A309	*060*	*->184F*	*[MP1_10]*
dW/dT	A30A	*0C9*	*?NC XQ*	*g.cosθ1*
dW/dT	A30B	*064*	*->1932*	*[RCSCR*] - selects Chip0*
dW/dT	A30C	*031*	*?NC XQ*	*g.cosθ1 + L1.w1^2*
dW/dT	A30D	*060*	*->180C*	*[AD2_13]*
dW/dT	A30E	***375***	*PORT DEP:*	*does [STSRC] and recall {m2, m1}*
dW/dT	A30F	*03C*	*XQ*	*puts m2 in N, m1 in A and C*
dW/dT	A310	*343*	*->A343*	*[M2NM1A] - returns in DEC*
dW/dT	A311	0B0	C=N ALL	*m2*
dW/dT	A312	*01D*	*?NC XQ*	*m1+m2*
dW/dT	A313	*060*	*->1807*	*[AD2_10]*
dW/dT	A314	*0C9*	*?NC XQ*	*g.cosθ1 + L1.w1^2*
dW/dT	A315	*064*	*->1932*	*[RCSCR*] - selects Chip0*
dW/dT	A316	*149*	*?NC XQ*	*(m1+m2).(g.cosθ1 + L1.w1^2)*
dW/dT	A317	*060*	*->1852*	*[MP2_13]*
dW/dT	A318	***1F8***	***READ 7(O)***	*L2.m2.w2^2.cos(θ1-θ2)*
dW/dT	A319	*025*	*?NC XQ*	*(m1+m2).(g.cosθ1 + L1.w1^2) + …*
dW/dT	A31A	*060*	*->1809*	*[AD1_10]*
dW/dT	A31B	***1B8***	***READ 6(N)***	*2,sin(θ1-θ2)*
dW/dT	A31C	*13D*	*?NC XQ*	*numerator*
dW/dT	A31D	*060*	*->184F*	*[MP1_10]*
dW/dT	A31E	***178***	***READ 5(M)***	*common denominator*
dW/dT	A31F	*269*	*?NC XQ*	
dW/dT	A320	*060*	*->189A*	*[DV1_10]*
dW/dT	A321	***375***	*PORT DEP:*	*Does [STSCR] and Puts L2 in N*
dW/dT	A322	*03C*	*XQ*	*leaves Chip0 selected*

dW/dT		A323	338	->A338	[N=L2+] - returns in DEC
dW/dT		A324	0A9	?NC XQ	{A,B} <-> {Q,+}
dW/dT		A325	064	->192A	[EXSCR]
dW/dT		A326	0B0	C=N ALL	L2
dW/dT		A327	269	?NC XQ	w2' at last!
dW/dT		A328	060	->189A	[DV1_10]
dW/dT		A329	**070**	**N=C ALL**	***w2' at last!***
dW/dT		A32A	260	SETHEX	
dW/dT	**NSTO18**	**A32B**	130	LDI S&X	
dW/dT		A32C	***012***	***R18***	index to R18
dW/dT		A32D	29D	?NC XQ	Get Register Value
dW/dT		A32E	100	->40A7	[READRG#]
dW/dT		A32F	031	?NC XQ	uses value in N
dW/dT		A330	124	->490C	[WRTSEL]-1 (Selects Chip0)
dW/dT		A331	3C1	?NC GO	Normal Function Return
dW/dT		A332	002	->00F0	[NFRPU]
dW/dT	**N=L1+**	**A333**	089	?NC XQ	g.cos$\theta 1$
dW/dT		A334	064	->1922	[STSCR]
dW/dT	**N=L1**	**A335**	130	LDI S&X	
dW/dT		A336	***01B***	***R27***	pointer to ***L1***
dW/dT		A337	02B	JNC +05	[BOTH]
dW/dT	**N=L2+**	**A338**	089	?NC XQ	
dW/dT		A339	064	->1922	[STSCR]
dW/dT	**N=L2**	**A33A**	130	LDI S&X	
dW/dT		A33B	***01C***	***R28***	pointer to ***L2***
dW/dT	BOTH	A33C	29D	?NC XQ	Get Register Value
dW/dT		A33D	100	->40A7	[READRG#]
dW/dT		A33E	2A0	SETDEC	
dW/dT		A33F	070	N=C ALL	L2
dW/dT		A340	046	C=0 S&X	
dW/dT		A341	270	RAMSLCT	select Chip0
dW/dT		A342	3E0	RTN	
dW/dT	**M2NM1A**	**A343**	081	?NC XQ	g.cos$\theta 1$ + L1.w1^2
dW/dT		A344	064	->1920	[STSCR*] - selects Chip0
dW/dT		A345	130	LDI S&X	
dW/dT		A346	***01E***	***R30***	pointer to ***m2***
dW/dT		A347	29D	?NC XQ	Get Register Value
dW/dT		A348	100	->40A7	[READRG#]
dW/dT		A349	070	N=C ALL	m2 in N
dW/dT	**A=M1***	**A34A**	**046**	**C=0 S&X**	
dW/dT		A34B	270	RAMSLCT	
dW/dT	**A=M1**	**A34C**	130	LDI S&X	
dW/dT		A34D	***01D***	***R29***	pointer to ***m1***
dW/dT		A34E	29D	?NC XQ	Get Register Value
dW/dT		A34F	100	->40A7	[READRG#]
dW/dT		A350	2A0	SETDEC	
dW/dT	ZEROED	A351	10E	A=C ALL	
dW/dT		A352	3E0	RTN	
dW/dT	**GSINN**	**A353**	**2A0**	**SETDEC**	
dW/dT		A354	0B0	C=N ALL	
dW/dT		A355	2EE	?C#0 ALL	is it zero?
dW/dT		A356	***3DB***	***JNC -05***	yes, avoid "-0" issues
dW/dT		A357	3C4	ST=0	skips [TRGSET]
dW/dT		A358	048	SETF 4	result in RAD
dW/dT		A359	229	?NC XQ	sin$\theta 1$ - skipping [TRGSET]

324	dW/dT		A35A	*048*	*->128A*	*[SIN1]*
325	dW/dT		A35B	11E	A=C MS	*bug or what??*
326	dW/dT	**G*AB**	**A35C**	**2A0**	**SETDEC**	
327	dW/dT		A35D	04E	C=0 ALL	
328	dW/dT		A35E	35C	PT= 12	
329	dW/dT		A35F	250	LD@PT- 9	*g = 9.81 m/s^2*
330	dW/dT		A360	210	LD@PT- 8	
331	dW/dT		A361	050	LD@PT- 1	
332	dW/dT		A362	*13D*	*?NC GO*	*g.sinθ1 <or> g.cosθ1*
333	dW/dT		A363	*062*	*->184F*	*[MP1_10]*

$$\omega_1' = \frac{-g\,(2\,m_1 + m_2)\sin\theta_1 - m_2\,g\sin(\theta_1 - 2\,\theta_2) - 2\sin(\theta_1 - \theta_2)\,m_2\,(\omega_2{}^2\,L_2 + \omega_1{}^2\,L_1\cos(\theta_1 - \theta_2))}{L_1\,(2\,m_1 + m_2 - m_2\cos(2\,\theta_1 - 2\,\theta_2))}$$

$$\omega_2' = \frac{2\sin(\theta_1 - \theta_2)\,(\omega_1{}^2\,L_1\,(m_1 + m_2) + g(m_1 + m_2)\cos\theta_1 + \omega_2{}^2\,L_2\,m_2\cos(\theta_1 - \theta_2))}{L_2\,(2\,m_1 + m_2 - m_2\cos(2\,\theta_1 - 2\,\theta_2))}$$

The differential equations of movement for the pendula cases are of second-order, therefore we need to add two additional variables to reduce them to first-order ODEs, and thus we're ok to use the Runge-Kutta methodology. This trick doubes the number of ODE's, even if the two added ones are extremely simple:

dθ1/dt =w1

dθ2/dt = w2

and thus removing the second derivative of the angles from the system:

$d^2\theta 1/dt^2$ => dw1/dt

$d^2\theta 2/dt^2$ => dw2/dt

Appendix 2. MCODE routines for Attractor equations.

1. Lorenz System of first-order ODEs

dL/dT	*Header*	AD31	*094*	*"T"*	***T - zo***
dL/dT	*Header*	AD32	*064*	*"d"*	***Z - yo***
dL/dT	*Header*	AD33	*02F*	*"/"*	***Y - x0***
dL/dT	*Header*	AD34	*00C*	*"L"*	***X - t0***
dL/dT	*Header*	AD35	*064*	*"d"*	*Ángel Martin*
dL/dT	**dL/dT**	**AD36**	*3A5*	*?NC XQ*	*align vars w/ stack letters*
dL/dT		AD37	*050*	*->14E9*	*[RDNSUB]*
		AD38	*199*	*?NC XQ*	*Checks XYZ - sets DEC mode*
R14 - β		AD39	*100*	*->4066*	*[CHKST3]*
R13 - ρ		AD3A	0F8	READ 3(X)	*x*
R12 - σ		AD3B	10E	A=C ALL	
		AD3C	0B8	READ 2(Y)	*y*
dL/dT		AD3D	*135*	*?NC XQ*	*xy*
dL/dT		AD3E	*060*	*->184D*	*[MP2_10]*
dL/dT		AD3F	*089*	*?NC XQ*	*xy*
dL/dT		AD40	*064*	*->1922*	*[STSCR]*
dL/dT		AD41	260	SETHEX	
dL/dT		AD42	130	LDI S&X	
dL/dT		AD43	***00E***	***R14***	*β*
dL/dT		AD44	*29D*	*?NC XQ*	*Get Register Value*
dL/dT		AD45	*100*	*->40A7*	*[READRG#]*
dL/dT		AD46	2A0	SETDEC	
dL/dT		AD47	2BE	C=-C-1 MS	*$-\beta$*
dL/dT		AD48	10E	A=C ALL	*put it in A for math*
dL/dT		AD49	046	C=0 S&X	
dL/dT		AD4A	270	RAMSLCT	*select Chip0*
dL/dT		AD4B	078	READ 1(Z)	*z*
dL/dT		AD4C	*135*	*?NC XQ*	*$-\beta$.z*
dL/dT		AD4D	*060*	*->184D*	*[MP2_10]*
dL/dT		AD4E	*0D1*	*?NC XQ*	*xy*
dL/dT		AD4F	*064*	*->1934*	*[RCSCR]*
dL/dT		AD50	*031*	*?NC XQ*	*xy-βz*
dL/dT		AD51	*060*	*->180C*	*[AD2_13]*
dL/dT		AD52	070	N=C ALL	*dz/dt*
dL/dT		AD53	260	SETHEX	
dL/dT		AD54	130	LDI S&X	
dL/dT		AD55	***00D***	***R13***	*ρ*
dL/dT		AD56	*29D*	*?NC XQ*	*Get Register Value*
dL/dT		AD57	*100*	*->40A7*	*[READRG#]*
dL/dT		AD58	2A0	SETDEC	
dL/dT		AD59	10E	A=C ALL	*ρ*
dL/dT		AD5A	046	C=0 S&X	
dL/dT		AD5B	270	RAMSLCT	*select Chip0*
dL/dT		AD5C	078	READ 1(Z)	*z*
dL/dT		AD5D	2BE	C=-C-1 MS	*-z*
dL/dT		AD5E	000	NOP	*let carry settle*
dL/dT		AD5F	*01D*	*?NC XQ*	*(ρ-z)*
dL/dT		AD60	*060*	*->1807*	*[AD2_10]*
dL/dT		AD61	0F8	READ 3(X)	*x*
dL/dT		AD62	*13D*	*?NC XQ*	*x.(ρ-z)*

51	dL/dT	AD63	*060*	*->184F*	*[MP1_10]*
52	dL/dT	AD64	0B8	READ 2(Y)	
53	dL/dT	AD65	2BE	C=-C-1 MS	*-y*
54	dL/dT	AD66	000	NOP	*let carry settle*
55	dL/dT	AD67	*025*	*?NC XQ*	*x.(ρ-z)-y*
56	dL/dT	AD68	*060*	*->1809*	*[AD1_10]*
57	dL/dT	AD69	0F0	C<>N ALL	*dz/dt*
58	dL/dT	AD6A	***068***	***WRIT 1(Z)***	***dz/dt in Z***
59	dL/dT	AD6B	0F8	READ 3(X)	
60	dL/dT	AD6C	2BE	C=-C-1 MS	*-x*
61	dL/dT	AD6D	10E	A=C ALL	
62	dL/dT	AD6E	0B8	READ 2(Y)	*y*
63	dL/dT	AD6F	*01D*	*?NC XQ*	*y-x*
64	dL/dT	AD70	*060*	*->1807*	*[AD2_10]*
65	dL/dT	AD71	0F0	C<>N ALL	*dy/dt*
66	dL/dT	AD72	***0A8***	***WRIT 2(Y)***	***dy/dt in Y***
67	dL/dT	AD73	260	SETHEX	
68	dL/dT	AD74	130	LDI S&X	
69	dL/dT	AD75	***00C***	***R12***	*σ*
70	dL/dT	AD76	*29D*	*?NC XQ*	*Get Register Value*
71	dL/dT	AD77	*100*	*->40A7*	*[READRG#]*
72	dL/dT	AD78	2A0	SETDEC	
73	dL/dT	AD79	10E	A=C ALL	*σ*
74	dL/dT	AD7A	046	C=0 S&X	
75	dL/dT	AD7B	270	RAMSLCT	*select Chip0*
76	dL/dT	AD7C	0B0	C=N ALL	*y-x*
77	dL/dT	AD7D	*135*	*?NC XQ*	*σ.(y-x)*
78	dL/dT	AD7E	*060*	*->184D*	*[MP2_10]*
79	dL/dT	AD7F	***0E8***	***WRIT 3(X)***	***dx/dt n X***
80	dL/dT	AD80	3E0	RTN	*as per the manual…*

$$\begin{cases} \dfrac{dx}{dt} = \sigma(-x+y) \\ \dfrac{dy}{dt} = -xz + \rho x - y \\ \dfrac{dz}{dt} = xy - \beta z \end{cases}$$

2. Rössler system of first-order ODEs

dR/dT	*Header*	AC0D	*094*	*"T"*	***T - zo***
dR/dT	*Header*	AC0E	*064*	*"d"*	***Z - yo***
dR/dT	*Header*	AC0F	*02F*	*"/"*	***Y - x0***
dR/dT	*Header*	AC10	*012*	*"R"*	***X - t0***
dR/dT	*Header*	AC11	*064*	*"d"*	*Ángel Martin*
dR/dT	**dR/dT**	**AC12**	*3A5*	*?NC XQ*	*align vars w/ stack letters*
dR/dT		AC13	*050*	*->14E9*	*[RDNSUB]*
dR/dT		AC14	*199*	*?NC XQ*	*Checks XYZ - sets DEC mode*
dR/dT		AC15	*100*	*->4066*	*[CHKST3]*
dR/dT		AC16	260	SETHEX	
dR/dT		AC17	130	LDI S&X	
dR/dT		AC18	***00C***	***R12***	*A*
dR/dT		AC19	*29D*	*?NC XQ*	*Get Register Value*
dR/dT		AC1A	*100*	*->40A7*	*[READRG#]*
dR/dT		AC1B	2A0	SETDEC	
dR/dT		AC1C	10E	A=C ALL	*A*
dR/dT		AC1D	046	C=0 S&X	
dR/dT		AC1E	270	RAMSLCT	*select Chip0*
dR/dT		AC1F	0B8	READ 2(Y)	
dR/dT		AC20	*135*	*?NC XQ*	*A.y*
dR/dT		AC21	*060*	*->184D*	*[MP2_10]*
dR/dT		AC22	0F8	READ 3(X)	
dR/dT		AC23	*025*	*?NC XQ*	*x+A.y*
dR/dT		AC24	*060*	*->1809*	*[AD1_10]*
dR/dT		AC25	0F0	C<>N ALL	*park it in N*
dR/dT		AC26	0B8	READ 2(Y)	
dR/dT		AC27	10E	A=C ALL	
dR/dT		AC28	078	READ 1(Z)	
dR/dT		AC29	*01D*	*?NC XQ*	*y+z*
dR/dT		AC2A	*060*	*->1807*	*[AD2_10]*
dR/dT		AC2B	2BE	C=-C-1 MS	*dx/dt*
dR/dT		AC2C	0F0	C<>N ALL	*park it in N*
dR/dT		AC2D	***0A8***	***WRIT 2(Y)***	***dy/dt***
dR/dT		AC2E	260	SETHEX	
dR/dT		AC2F	130	LDI S&X	
dR/dT		AC30	***00E***	***R14***	*C*
dR/dT		AC31	*29D*	*?NC XQ*	*Get Register Value*
dR/dT		AC32	*100*	*->40A7*	*[READRG#]*
dR/dT		AC33	2A0	SETDEC	
dR/dT		AC34	2BE	C=-C-1 MS	
dR/dT		AC35	10E	A=C ALL	
dR/dT		AC36	046	C=0 S&X	
dR/dT		AC37	270	RAMSLCT	*select Chip0*
dR/dT		AC38	0F8	READ 3(X)	
dR/dT		AC39	*01D*	*?NC XQ*	*x-C*
dR/dT		AC3A	*060*	*->1807*	*[AD2_10]*
dR/dT		AC3B	078	READ 1(Z)	
dR/dT		AC3C	*13D*	*?NC XQ*	*z.(x-C)*
dR/dT		AC3D	*060*	*->184F*	*[MP1_10]*
dR/dT		AC3E	0F0	C<>N ALL	
dR/dT		AC3F	***0E8***	***WRIT 3(X)***	***dx/dt***
dR/dT		AC40	260	SETHEX	

53	dR/dT	AC41	130	LDI S&X	
54	dR/dT	AC42	***00D***	***R13***	*B*
55	dR/dT	AC43	*29D*	*?NC XQ*	*Get Register Value*
56	dR/dT	AC44	*100*	*->40A7*	*[READRG#]*
57	dR/dT	AC45	2A0	SETDEC	
58	dR/dT	AC46	10E	A=C ALL	
59	dR/dT	AC47	046	C=0 S&X	
60	dR/dT	AC48	270	RAMSLCT	*select Chip0*
61	dR/dT	AC49	0B0	C=N ALL	
62	dR/dT	AC4A	*01D*	*?NC XQ*	*B+z.(x-C)*
63	dR/dT	AC4B	*060*	*->1807*	*[AD2_10]*
64	dR/dT	AC4C	***068***	***WRIT 1(Z)***	***dz/dt***
65	dR/dT	AC4D	3E0	RTN	

Röesler equations:

$$\begin{cases} \frac{dx}{dt} = -(y+z) \\ \frac{dy}{dt} = x + ay \\ \frac{dz}{dt} = b + z(x-c) \end{cases}$$

3. Thomas system of first-order ODEs

dT/dT	*Header*	AC77	*094*	*"T"*	***T - zo***
dT/dT	*Header*	AC78	*064*	*"d"*	***Z - yo***
dT/dT	*Header*	AC79	*02F*	*"/"*	***Y - x0***
dT/dT	*Header*	AC7A	*014*	*"T"*	***X - t0***
dT/dT	*Header*	AC7B	*064*	*"d"*	*Ángel Martin*
dT/dT	**dT/dT**	**AC7C**	*3A5*	*?NC XQ*	*align vars w/ stack letters*
dT/dT		AC7D	*050*	*->14E9*	*[RDNSUB]*
dT/dT		AC7E	*199*	*?NC XQ*	*Checks XYZ - sets DEC mode*
dT/dT		AC7F	*100*	*->4066*	*[CHKST3]*
dT/dT		AC80	078	READ 1(Z)	*z*
dT/dT		AC81	070	N=C ALL	
dT/dT		AC82	***375***	*PORT DEP:*	*puts sin(z) in scratch*
dT/dT		AC83	*03C*	*XQ*	*returns in DEC mode*
dT/dT		AC84	*0C4*	*->ACC4*	*[SINN]*
dT/dT		AC85	260	SETHEX	
dT/dT		AC86	130	LDI S&X	
dT/dT		AC87	***00C***	***R12***	*A*
dT/dT		AC88	*29D*	*?NC XQ*	*Get Register Value*
dT/dT		AC89	*100*	*->40A7*	*[READRG#]*
dT/dT		AC8A	2A0	SETDEC	
dT/dT		AC8B	2BE	C=-C-1 MS	
dT/dT		AC8C	10E	A=C ALL	*-b*
dT/dT		AC8D	046	C=0 S&X	
dT/dT		AC8E	270	RAMSLCT	*select Chip0*
dT/dT		AC8F	0AE	A<>C ALL	
dT/dT		AC90	10E	A=C ALL	
dT/dT		AC91	***028***	***WRIT 0(T)***	*saves -b in T(0)*
dT/dT		AC92	0B8	READ 2(Y)	*y*
dT/dT		AC93	*135*	*?NC XQ*	*-b.y*
dT/dT		AC94	*060*	*->184D*	*[MP2_10]*
dT/dT		AC95	*0D1*	*?NC XQ*	*sin z*
dT/dT		AC96	*064*	*->1934*	*[RCSCR]*
dT/dT		AC97	*031*	*?NC XQ*	*sin z - b.y*
dT/dT		AC98	*060*	*->180C*	*[AD2_13]*
dT/dT		AC99	***128***	***WRIT 4(L)***	*sin z - b.y*
dT/dT		AC9A	0B8	READ 2(Y)	*y*
dT/dT		AC9B	070	N=C ALL	
dT/dT		AC9C	***375***	*PORT DEP:*	*puts sin(y) in scratch*
dT/dT		AC9D	*03C*	*XQ*	*returns in DEC mode*
dT/dT		AC9E	*0C4*	*->ACC4*	*[SINN]*
dT/dT		AC9F	0F8	READ 3(X)	*x*
dT/dT		ACA0	10E	A=C ALL	
dT/dT		ACA1	046	C=0 S&X	
dT/dT		ACA2	270	RAMSLCT	*select Chip0*
dT/dT		ACA3	038	READATA	*-b*
dT/dT		ACA4	*135*	*?NC XQ*	*-b.x*
dT/dT		ACA5	*060*	*->184D*	*[MP2_10]*
dT/dT		ACA6	*0D1*	*?NC XQ*	*sin y*
dT/dT		ACA7	*064*	*->1934*	*[RCSCR]*
dT/dT		ACA8	*031*	*?NC XQ*	*sin y - b.x*
dT/dT		ACA9	*060*	*->180C*	*[AD2_13]*
dT/dT		ACAA	0AE	A<>C ALL	
dT/dT		ACAB	138	READ 4(L)	*sin z - b.y*

54	dT/dT		ACAC	***0A8***	***WRIT 2(Y)***	*now Y is free to use.*
55	dT/dT		ACAD	0AE	A<>C ALL	*sin y - b.x*
56	dT/dT		ACAE	128	WRIT 4(L)	*dx/dt*
57	dT/dT		ACAF	0F8	READ 3(X)	x
58	dT/dT		ACB0	070	N=C ALL	
59	dT/dT		ACB1	***375***	*PORT DEP:*	*puts sin(x) in scratch*
60	dT/dT		ACB2	*03C*	*XQ*	*returns in DEC mode*
61	dT/dT		ACB3	*0C4*	*->ACC4*	*[SINN]*
62	dT/dT		ACB4	078	READ 1(Z)	*z*
63	dT/dT		ACB5	10E	A=C ALL	
64	dT/dT		ACB6	046	C=0 S&X	
65	dT/dT		ACB7	270	RAMSLCT	*select Chip0*
66	dT/dT		ACB8	038	READATA	*-b*
67	dT/dT		ACB9	*135*	*?NC XQ*	*-b.z*
68	dT/dT		ACBA	*060*	*->184D*	*[MP2_10]*
69	dT/dT		ACBB	*0D1*	*?NC XQ*	*sin X*
70	dT/dT		ACBC	*064*	*->1934*	*[RCSCR]*
71	dT/dT		ACBD	*031*	*?NC XQ*	*sin x - b.z*
72	dT/dT		ACBE	*060*	*->180C*	*[AD2_13]*
73	dT/dT		ACBF	***068***	***WRIT 1(Z)***	***dz/dt***
74	dT/dT		ACC0	138	READ 4(L)	
75	dT/dT		ACC1	***0E8***	***WRIT 3(X)***	***dx/dt***
76	dT/dT		ACC2	*3C1*	*?NC GO*	*finish off…*
77	dT/dT		ACC3	*002*	*->00F0*	*[NFRPU]*
78	dT/dT	**SINN**	**ACC4**	**2A0**	**SETDEC**	
79	dT/dT		ACC5	0B0	C=N ALL	
80	dT/dT		ACC6	2EE	?C#0 ALL	*is it zero?*
81	dT/dT		ACC7	***01F***	***JC +03***	*[DOSIN]*
82	dT/dT		ACC8	10E	A=C ALL	
83	dT/dT		ACC9	*033*	*JNC +06*	*yes, avoid "-0" issues*
84	dT/dT	DOSIN	ACCA	070	N=C ALL	*argument for SIN/COS*
85	dT/dT		ACCB	3C4	ST=0	*skips [TRGSET]*
86	dT/dT		ACCC	048	SETF 4	*result in RAD*
87	dT/dT		ACCD	*229*	*?NC XQ*	*sin z*
88	dT/dT		ACCE	*048*	*->128A*	*[SIN1]*
89	dT/dT	EXIT	ACCF	*089*	*?NC GO*	
90	dT/dT		ACD0	*066*	*->1922*	*[STSCR]*

$$\begin{cases} \dfrac{dx}{dt} = \sin y - bx \\ \dfrac{dy}{dt} = \sin z - by \\ \dfrac{dz}{dt} = \sin x - bz \end{cases}$$

© 2022 ÁNGEL MARTIN

4. Sprott system of first-order ODEs

1	dS/dT	*Header*	ACD2	*094*	*"T"*	***T - zo***
2	dS/dT	*Header*	ACD3	*064*	*"d"*	***Z - yo***
3	dS/dT	*Header*	ACD4	*02F*	*"/"*	***Y - x0***
4	dS/dT	*Header*	ACD5	*013*	*"S"*	***X - t0***
5	dS/dT	*Header*	ACD6	*064*	*"d"*	*Ángel Martin*
6	dS/dT	**dS/dT**	**ACD7**	*3A5*	*?NC XQ*	*align vars w/ stack letters*
7	dS/dT		ACD8	*050*	*->14E9*	*[RDNSUB]*
8	dS/dT		ACD9	*199*	*?NC XQ*	*Checks XYZ - sets DEC mode*
9	dS/dT		ACDA	*100*	*->4066*	*[CHKST3]*
10	dS/dT		ACDB	260	SETHEX	
11	dS/dT		ACDC	130	LDI S&X	
12	dS/dT		ACDD	***00C***	***R12***	*A*
13	dS/dT		ACDE	*29D*	*?NC XQ*	*Get Register Value*
14	dS/dT		ACDF	*100*	*->40A7*	*[READRG#]*
15	dS/dT		ACE0	2A0	SETDEC	
16	dS/dT		ACE1	10E	A=C ALL	
17	dS/dT		ACE2	046	C=0 S&X	
18	dS/dT		ACE3	270	RAMSLCT	*select Chip0*
19	dS/dT		ACE4	0F8	READ 3(X)	
20	dS/dT		ACE5	*135*	*?NC XQ*	*A.x*
21	dS/dT		ACE6	*060*	*->184D*	*[MP2_10]*
22	dS/dT		ACE7	*001*	*?NC XQ*	*1+A/x*
23	dS/dT		ACE8	*060*	*->1800*	*[ADDONE]*
24	dS/dT		ACE9	0B8	READ 2(Y)	
25	dS/dT		ACEA	*13D*	*?NC XQ*	*y.(1+A.x)*
26	dS/dT		ACEB	*060*	*->184F*	*[MP1_10]*
27	dS/dT		ACEC	*089*	*?NC XQ*	*y.(1+A.x)*
28	dS/dT		ACED	*064*	*->1922*	*[STSCR]*
29	dS/dT		ACEE	0F8	READ 3(X)	
30	dS/dT		ACEF	10E	A=C ALL	
31	dS/dT		ACF0	078	READ 1(Z)	
32	dS/dT		ACF1	*135*	*?NC XQ*	*x.z*
33	dS/dT		ACF2	*060*	*->184D*	*[MP2_10]*
34	dS/dT		ACF3	*0D1*	*?NC XQ*	*y.(1+A.x)*
35	dS/dT		ACF4	*064*	*->1934*	*[RCSCR]*
36	dS/dT		ACF5	*031*	*?NC XQ*	*xz+y.(1+A.x)*
37	dS/dT		ACF6	*060*	*->180C*	*[AD2_13]*
38	dS/dT		ACF7	070	N=C ALL	*dx/dt*
39	dS/dT		ACF8	0F8	READ 3(X)	
40	dS/dT		ACF9	2BE	C=-C-1 MS	
41	dS/dT		ACFA	10E	A=C ALL	
42	dS/dT		ACFB	*1E1*	*?NC XQ*	*Increment C*
43	dS/dT		ACFC	*100*	*->4078*	*[INCC10]*
44	dS/dT		ACFD	0F8	READ 3(X)	
45	dS/dT		ACFE	*13D*	*?NC XQ*	*x.(1-x)*
46	dS/dT		ACFF	*060*	*->184F*	*[MP1_10]*
47	dS/dT		AD00	*089*	*?NC XQ*	*x.(1-x)*
48	dS/dT		AD01	*064*	*->1922*	*[STSCR]*
49	dS/dT		AD02	0B8	READ 2(Y)	
50	dS/dT		AD03	10E	A=C ALL	
51	dS/dT		AD04	*135*	*?NC XQ*	*y^2*
52	dS/dT		AD05	*060*	*->184D*	*[MP2_10]*
53	dS/dT		AD06	2BE	C=-C-1 MS	*sign change*

© 2022 ÁNGEL MARTIN

dS/dT	AD07	11E	A=C MS	*ditto in 13-digit form*
dS/dT	AD08	*0D1*	*?NC XQ*	*x.(1-x)*
dS/dT	AD09	*064*	*->1934*	*[RCSCR]*
dS/dT	AD0A	*031*	*?NC XQ*	*x.(1-x) + y^2*
dS/dT	AD0B	*060*	*->180C*	*[AD2_13]*
dS/dT	AD0C	128	WRIT 4(L)	*dz/dt*
dS/dT	AD0D	260	SETHEX	
dS/dT	AD0E	130	LDI S&X	
dS/dT	AD0F	***00D***	***R13***	*B*
dS/dT	AD10	*29D*	*?NC XQ*	*Get Register Value*
dS/dT	AD11	*100*	*->40A7*	*[READRG#]*
dS/dT	AD12	2A0	SETDEC	
dS/dT	AD13	2BE	C=-C-1 MS	*-B*
dS/dT	AD14	10E	A=C ALL	
dS/dT	AD15	046	C=0 S&X	
dS/dT	AD16	270	RAMSLCT	*select Chip0*
dS/dT	AD17	0F8	READ 3(X)	
dS/dT	AD18	*135*	*?NC XQ*	*B.x*
dS/dT	AD19	*060*	*->184D*	*[MP2_10]*
dS/dT	AD1A	0F8	READ 3(X)	
dS/dT	AD1B	*13D*	*?NC XQ*	*B.x^2*
dS/dT	AD1C	*060*	*->184F*	*[MP1_10]*
dS/dT	AD1D	*001*	*?NC XQ*	*1-B.x^2*
dS/dT	AD1E	*060*	*->1800*	*[ADDONE]*
dS/dT	AD1F	*089*	*?NC XQ*	*1-B.x^2*
dS/dT	AD20	*064*	*->1922*	*[STSCR]*
dS/dT	AD21	078	READ 1(Z)	
dS/dT	AD22	10E	A=C ALL	
dS/dT	AD23	0B8	READ 2(Y)	
dS/dT	AD24	*135*	*?NC XQ*	*y.z*
dS/dT	AD25	*060*	*->184D*	*[MP2_10]*
dS/dT	AD26	*0D1*	*?NC XQ*	*1 - B.x^2*
dS/dT	AD27	*064*	*->1934*	*[RCSCR]*
dS/dT	AD28	*031*	*?NC XQ*	*1 - B.x^2 + y.z*
dS/dT	AD29	*060*	*->180C*	*[AD2_13]*
dS/dT	AD2A	0A8	WRIT 2(Y)	***dy/dt***
dS/dT	AD2B	138	READ 4(L)	
dS/dT	AD2C	***068***	***WRIT 1(Z)***	***dz/dt***
dS/dT	AD2D	0B0	C=N ALL	
dS/dT	AD2E	***0E8***	***WRIT 3(X)***	***dx/dt***
dS/dT	AD2F	3E0	RTN	

$$\begin{cases} \dfrac{dx}{dt} = y + axy + xz \\ \dfrac{dy}{dt} = 1 - bx^2 + yz \\ \dfrac{dz}{dt} = x - x^2 - y^2 \end{cases}$$