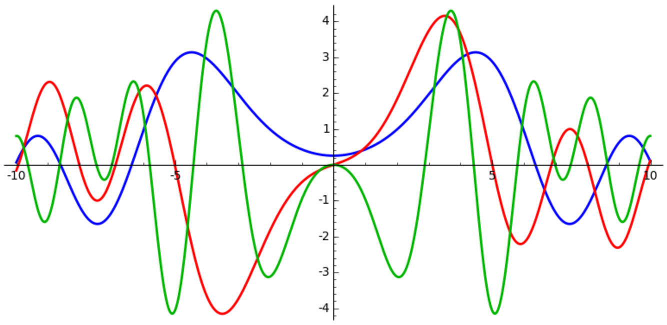
# HP-41 Module: Elliptic Functions and Orthogonal Polynomials




## Overview

This module includes a selection of functions and FOCAL routines mainly focused on the Elliptic Functions field and other related subjects. For the most part the same functions exist in the SandMath Module, but this version is a more portable implementation that suits itself better for Clonix/NoVRAM owners.

The initial section of the module covers the Carlson Integral forms as the method used to calculate the Incomplete Elliptic integral. The Arithmetic-Geometric means (standard and alternate formulations) provide the basis for a faster and much simpler implementation to calculate the Complete Elliptic Integrals.

The Examples section is meant to show a few immediate applications of the Elliptical Integrals, used to calculate Ellipse parameters, Ellipsoid surface areas, oscillation period of a simple pendulum, and mutual inductance between two coaxial coils. This is followed by a section on the Jacobi Elliptic Functions, plus the Theta, Whittaker W and Weierstrass Elliptic functions to round up (no pun intended) the elliptic theme. Note that the last two are not included in the SandMath.

Finally, a relatively large section of the FAT deals with Orthogonal Polynomials. Here too there are a fundamental set of MCODE functions taken from the SandMath and SandMatrix, plus a new group of FOCAL routines written by JM Baillard (a constant reference in this module as well) that calculate the coefficients of the most common orthogonal polynomials. Combined with Polynomial Evaluation, Integral and 1$^{st}$& 2$^{nd}$ Derivatives this set provides a comprehensive section on the subject. Without further ado, see below the list of functions included in the module:

| XROM | Function | Description | Input | Author |
|---|---|---|---|---|
| 30,00 | **-ELLIPTICS** | Section Header | n/a | *n/a* |
| 30,01 | **AGM2** | Arithmetic-Geometric Mean | arguments in X, Y | *Ángel Martin* |
| 30,02 | **AGM2** | Arithmetic-Geometric Mean | arguments in X, Y | *Ángel Martin* |
| 30,03 | ***"CEI"*** | Complete Elliptic Integrals | argument in X | *JM Baillard* |
| 30,04 | **CRF** | Carlson Integral 1st. Kind | arguments in Z, Y, X | *JM Baillard* |
| 30,05 | **CRFZ** | CRF for complex arguments | arguments in Z, Y, X | *JM Baillard* |
| 30,06 | ***"CRG"*** | Carlson integral 2nd. Kind | arguments in Z, Y, X | *JM Baillard* |
| 30,07 | **CRJ** | Carlson Integral 3rd. Kind | arguments in Z, Y, X | *JM Baillard* |
| 30,08 | **CRJZ** | CRJ for complex arguments | arguments in Stack | *JM Baillard* |
| 30,09 | ***"EK"*** | Elliptic Int. 2nd. Order | argument in X | *Ángel Martin* |
| 30,10 | ***"ELI"*** | Incomplete Elliptic Integrals | arguments in X, Y | *JM Baillard* |
| 30,11 | **ELIPE** | Complete Elliptic Int. 2nd. Order | argument in X | *Ángel Martin* |
| 30,12 | **ELIPF** | Incomplete Elliptic Int. 1st. Order | arguments in Y,X | *Ángel Martin* |
| 30,13 | **ELIPK** | Complete Elliptic Int. 1st. Order | argument in X | *Ángel Martin* |
| 30,14 | **GHM** | Geometric-Harmonic Mean | arguments in X, Y | Greg McClure |
| 30,15 | ***"JEF"*** | Jacobi Elliptic Functions | | *JM Baillard* |
| 30,16 | ***"KK"*** | Elliptic Int. 1st. Order | argument in X | *Ángel Martin* |

| | | | | |
|---|---|---|---|---|
| 30,17 | *"LEI1"* | Legendre Integral 1st. Kind | arguments in Y,X | *JM Baillard* |
| 30,18 | *"LEI2"* | Legendre Integral 2nd. Kind | arguments in Y,X | *JM Baillard* |
| 30,19 | *"LEI3"* | Legendre Integral 3rd. Kind | arguments in Z, Y,X | *JM Baillard* |
| 30,20 | **-EXAMPLES** | Section Header | n/a | *n/a* |
| 30,21 | **-/+** | Calculates (y-x)/(y+x) | a,b in X,Y | *Ángel Martin* |
| 30,22 | **ECC** | Eccentricity | a,b in X,Y | *Ángel Martin* |
| 30.23 | **ELP** | Perimeter of Ellipse | a,b in X,Y | *Ángel Martin* |
| 30.24 | *"MIND"* | Mutual Inductance | prompts for data | *Ángel Martin* |
| 30.25 | *"SAE"* | Surface Area of Ellipsoide | a,b,c in Stack | *JM Baillard* |
| 30.26 | *"PEND"* | Pendulum period | prompts for data | *Ángel Martin* |
| 30.27 | **-JACOBIAN** | Section Header | n/a | *Ángel Martin* |
| 30.28 | **ACOSH** | Arc Hyperbolic Sine | argument in X | *Ángel Martin* |
| 30.29 | **AJF** | auxiliary for JEF | n/a | *JM Baillard* |
| 30.30 | **ASINH** | Arc Hyperbolic Sine | argument in X | *Ángel Martin* |
| 30.31 | **ATANH** | Arc Hyperbolic Tangent | argument in X | *JM Baillard* |
| 30.32 | **CBRT** | Cubic Root | argument in X | *Ángel Martin* |
| 30.33 | **COSH** | Hyperbolic Cosine | argument in X | *Ángel Martin* |
| 30.34 | *"P2"* | Quadratic Equation | a,b,c in Stack | *JM Baillard* |
| 30.35 | *"P3"* | Cubic Equation | a,b,c, d in Stack | *JM Baillard* |
| 30.36 | **SINH** | Htyperbolic Sine | argument in X | *Ángel Martin* |
| 30.37 | **TANH** | Hyperbolic Tangent | argument in X | *JM Baillard* |
| 30.38 | **THETA** | Theta Functions | n, q, x in {Z, Y, X} | *Ángel Martin* |
| 30.39 | *"WEF"* | Weierstrass Elliptic Function | | *Ángel Martin* |
| 30.40 | *"WHIW"* | Whittaker "W" function | | *Ángel Martin* |
| 30.41 | **-ORTHOPOL** | Section Header | n/a | *n/a* |
| 30.42 | *"BELL"* | Bell Polynomials | Cnt'l word in Y, argument in X | *JM Baillard* |
| 30.43 | *"BSSL"* | Bessel Polynomials | Cnt'l word in Y, argument in X | *JM Baillard* |
| 30.44 | **CHBT** | Chebyshev T(x) | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.45 | **CHBU** | Chebyshev U(x) | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.46 | *"FIB"* | Fibonacci Polynomials | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.47 | **HMT** | Hermite Polynomials | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.48 | **LAG** | Lagrange Polynomials | a in Z, Cnt'l word in Y, point in X | *Ángel Martin* |
| 30.49 | **LANX** | Generalized Lagrange Polyn | a in Z, Cnt'l word in Y, point in X | *Ángel Martin* |
| 30.50 | **LEG** | Legendre Polynomials | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.51 | *"CBT+"* | Chebyshev T(x) Coefficients | Initial RG in Y, n in X | *JM Baillard* |
| 30.51 | *"CBU+"* | Chebyshev U(x) Coefficients | Initial RG in Y, n in X | *JM Baillard* |
| 30.53 | *"HMT+"* | Hermite Polynomials Coeffs. | Initial RG in Y, n in X | *JM Baillard* |
| 30.54 | *"JCP+"* | Jacobi Polun. Coefficients | Initial RG in Y, n in X | *JM Baillard* |
| 30.55 | *"LANX+"* | Generalized Lagrange Polyn | Initial RG in Y, n in X | *JM Baillard* |
| 30.56 | *"LEG+"* | Legendre Polyn. Coefficients | Initial RG in Y, n in X | *JM Baillard* |
| 30.57 | *"USP+"* | Ultra-Spherical Polyn Coeffs. | Initial RG in Y, n in X | *JM Baillard* |
| 30.58 | **dPL** | 1st. Derivative polynomial | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.59 | **d2PL** | 2nd. Derivative Polynomial | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.60 | **DTC** | Delete Tiny Coeffs | Cnt'l word in X | *Ángel Martin* |
| 30.61 | **ITPL** | Integral of Polynomial | Cnt'l word in Y, argument in X | *Ángel Martin* |
| 30.62 | **PDEG** | Polyn Degree | Cnt'l word in X | *Ángel Martin* |
| 30.63 | **PVAL** | Polyn Evaluation | Cnt'l word in Y, argument in X | *Ángel Martin* |

# 1 –Elliptic Integrals.

In integral calculus, elliptic integrals originally arose in connection with the problem of giving the arc length of an ellipse. They were first studied by Giulio Fagnano and Leonhard Euler. Modern mathematics defines an "elliptic integral" as any function f which can be expressed in the form

$$f(x) = \int_c^x R\left(t, \sqrt{P(t)}\right) dt,$$

where R is a rational function of its two arguments, P is a polynomial of degree 3 or 4 with no repeated roots, and c is a constant.The most common ones are the incomplete Elliptic Integrals of the first, second and third kinds. The definitions for these functions is as follows:

$$F(\varphi, k) = F(\varphi \,|\, k^2) = F(\sin\varphi; k) = \int_0^\varphi \frac{d\theta}{\sqrt{1 - k^2 \sin^2\theta}}.$$

$$E(\phi, k) = \int_0^\phi \sqrt{1 - k^2 \sin^2(t)}\, dt.$$

$$\Pi(\phi, n, k) = \int_0^\phi \frac{1}{(1 - n\sin^2(t))\sqrt{1 - k^2 \sin^2(t)}}\, dt.$$

Besides the traditional Legendre form, the elliptic integrals may also be expressed in Carlson symmetric form – which has been the basis for this implementation. The Carlson integrals RF and RJ are therefore the basis to calculate the incomplete elliptic integrals if first and second kinds, according to the formulas shown below:

- *Incomplete Elliptic integral of 1st. kind:*

$$F(\phi, k) = \sin\phi R_F\left(\cos^2\phi, 1 - k^2\sin^2\phi, 1\right)$$

- *Incomplete Elliptic integral of 2nd. Kind:*

$E = \sin(\Phi).\ \mathbf{R_F}(\cos^2(\Phi); 1\text{-}k.\sin^2(\Phi); 1) - (k/3)\sin^3(\Phi).\ \mathbf{R_J}(\cos^2(\Phi); 1\text{-}\ k.\sin^2(\Phi); 1)$

- *Incomplete Elliptic Integral of 3rd. kind:*

$P = \sin(\Phi).\ \mathbf{R_F}(\cos^2(\Phi); 1\text{-}k.\sin^2(\Phi); 1) - (k/3)\sin^3(\Phi).\ \mathbf{R_J}(\cos^2(\Phi); 1\text{-}k.\sin^2(\Phi); 1; 1\text{+}n.\sin^2(\Phi))$

Functions CRF and CRJ in the module are written in MCODE, which provides the speed advantage needed in the repeated calculations where these functions have a defining role. There are several functions and programs you can use to calculate these functions, as follows:

| Incomplete Integrals | FOCAL Routine | MCODE Function |
|---|---|---|
| First kind | **“LEI1”** | **ELIPF** |
| Second kind | **“LEI2”** | n/a |
| Third kind | **“LEI3”** | |
| All at once | **“ELI”** | |

Stack input for the first two cases are the amplitude $\Phi$in Y and the argument “m” in degrees in X. – and **LEI3** also expects the characteristic “n” in Z. The result is always returned to X.

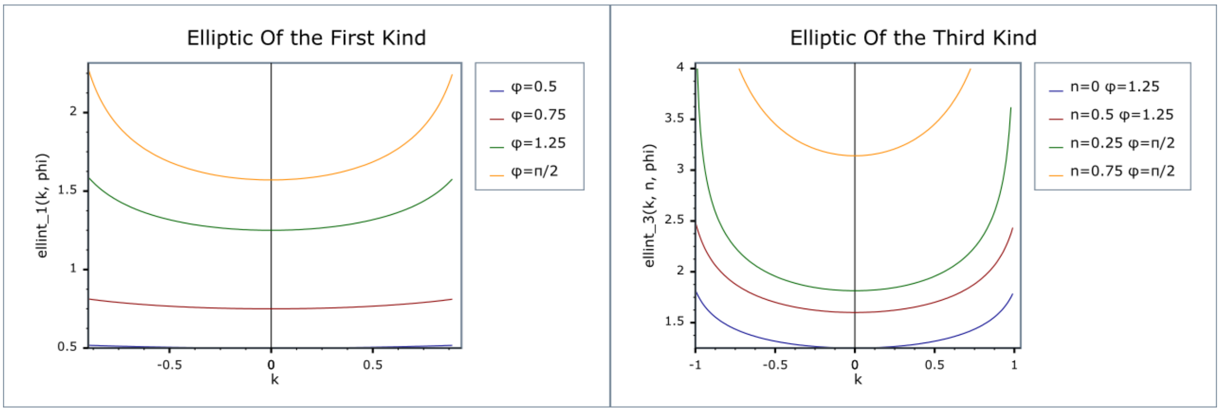
*Examples:* **in DEG mode** (!) calculate F(0.7; 84), E(0.7; 84), and P(0.9; 0.7; 84).-

0.7, ENTER^, 84, XEQ "LEI1" -> F ( 84° | 0.7 ) = 1.884976271
0.7, ENTER^, 84, XEQ "LEI2" -> E ( 84° | 0.7 ) = 1.184070048
0.9, ENTER^, 0.7, ENTER^, 84, XEQ "LEI3"-> P (0.9; 84° | 0.7 ) = 1.336853616

**Note that LEI1 uses data registers {R00 - R03}, and LEI2/3 also use R04.**

Obviously we could have used **ELIPF** for the first case – which has a slightly faster execution and yields the same result. **ELIPF** is implemented as a MCODE function which simply calls **CRF** with the appropriate input parameters. All the heavy lifting is thus performed by **CRF**, which together with **CRJ** do all the hard work in the calculation for the Elliptic Integrals of first, second and third kinds.

The figure below shows the first and third kinds in comparison:




## Complete Forms

Note also that the respective **complete** elliptic integrals are easily obtained by setting the value of the **amplitude**, $\Phi$ (the upper limit of the integrals), to $\pi/2$. Therefore, you could use the same functions to calculate the complete version of the integrals – but that's a slower and generally less accurate approach than using the dedicated functions, based on the Arithmetic-Geometric means.

| Complete Integrals | FOCAL Routine | MCODE Function |
|---|---|---|
| First kind | **"KK"** | **ELIPK** |
| Second kind | **"EK"** | **ELIPE** |
| Third kind | n/a | n/a |
| All at once | **"CEI"** | |

The FOCAL programs "**KK**" and '**EK**" are shown below in case you're interested. As you can see they're little more than a driver for the AGM functions. Note also that the second kind requires calculating the first kind first.

| | | | | |
|---|---|---|---|---|
| 1 | LBL "KK" | | 12 | * |
| 2 | CHS | | 13 | RTN |
| 3 | 1 | | 14 | LBL "EK" |
| 4 | + | | 15 | XROM "KK" |
| 5 | SQRT | | 16 | RCL O |
| 6 | STO O | | 17 | X^2 |
| 7 | 1 | | 18 | 1 |
| 8 | AGM | | 19 | AGM2 |
| 9 | ST+ X | | 20 | + |
| 10 | 1/X | | 21 | END |
| 11 | PI | | | |

*Examples*: calculate the complete forms for the same cases shown above, for amplitude = 90.

0.7, XEQ "ELIPK" -> K(0.7) =2.075363135
0.7, XEQ "ELIPE" ->E (0.7) = 1.241670568

**No data registers are used, and any angular mode can be selected (not relevant here).**
**The MCODE functions will save the initial argument in LastX**

## Auxiliary functions.

The following examples will illustrate the usage of the Carlson Integrals and the AGM functions.Note the inverse order of arguments for the Carlson functions; that **AGM** is a symmetric argument function; and that for **AGM2** the distance between both arguments must be <=1

Calculate$R_F(2;3;4)$,, and $R_G(2;3;4)$.

4 ENTER^, 3 ENTER^, 2 XEQ "CRF" → $R_F(2;3;4) = 0.584082842$
4 ENTER^, 3 ENTER^, 2 XEQ "CRG"→ $R_G(2;3;4) = 1.725503028$

Calculate $R_J(1;2;3;4)$ and $R_J(1;2;4;7)$.

4 ENTER^, 3 ENTER^, 2 ENTER^, 1 XEQ "CRJ" → $R_J(1;2;3;4) = 0.239848100$
7 ENTER^, 4 ENTER^, 2 ENTER^, 1 XEQ "CRJ" → $R_J(1,2,4,7) = 0.147854445$

Calculate the Arithmetic-Geometric Mean for 8 and 23.-

8, ENTER^, 23, XEQ "AGM" -> AGM ( 8, 23 ) = 14.51619896
0.5, ENTER^, 0.9, XEQ "AGM2" -> AGM2(0.5, 0.9) = 0.685370957

For additional information on this subject you should refer to JM Baillard web pages – which also include examples of utilization of the FOCAL programs "CEI" and 'ELI".

http://hp41programs.yolasite.com/ellipticf.php

## 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 $1^{st}$ 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):

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

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 expresion using the **In**complete Elliptic Integral instead, as follows:

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

**ELIPF** { $\pi$/2, [(a-b)/(a+b)]^2 } = $\pi$ (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, compared to the values obtained using **ELIPF**. As you can expect, the execution is substantially faster using the **AGM** approach.

| x | ELIPK(x) | ELIPF ($\pi$/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 |

## Modified Arithmetic-Geometric Mean { AGM2 }

We've seen the relationship between the complete Elliptic integral of first kind (ELIPK) and the AGM largely facilitates the calculation. Would it be possible to calculate the complete Elliptic of 2nd. Kind (ELIPE) using a similar approach, and if so how? As it turns out there is a way – involving the Modified AGM as described below. First we define a sequence of triples as follows:

$$x_{n+1} := \frac{x_n + y_n}{2},\ y_{n+1} := z_n + \sqrt{(x_n - z_n)(y_n - z_n)},$$
$$z_{n+1} := z_n - \sqrt{(x_n - z_n)(y_n - z_n)}.$$

Defining now the modified arithmetic-geometric mean (AGM2) of two positive numbers x and y as the common limit of the descending sequence {Xn} and the ascending sequence {Yn}, with X0 = x and y0=y (and z0=0)

The expressions we're interested in are those linking the Complete Elliptic integrals of first and second kind with the regular AGM and this newly defined AGM2. As it turns out both expressions exist, and are given below:

$$(1) \qquad \int_0^1 \frac{dx}{\sqrt{(1-x^2)(1-y^2x^2)}} = \frac{\pi}{2M(\beta)},$$

$$(2) \qquad \int_0^1 \sqrt{\frac{1-y^2x^2}{1-x^2}}\,dx = \frac{\pi N(\beta^2)}{2M(\beta)},$$

Where M(t) is the regular AGM(1, t) and N(t) the modified AGM2(1, t); and where {$\beta$, $\gamma$} are two positive numbers whose squares sum to one: $\beta$^2 + $\gamma$^2 = 1. In particular the equations hold if (in violation of the assumption, otherwise imposed) $\gamma$^2 = -1 - which implies $\beta$^2 = 2, facilitating the calculation even more.

So there we have it, both complete integrals can be obtained using the AGM and AGM2 functions, an iterative and fast convergent algorithm that can be easily implemented on the SandMath. Once AGM and AGM2 are available it's easy to write **ELIPK** and **ELIPE** – see the method used in the example quick FOCAL program below:

```
01 LBL "KK"
02 CHS
03 1
04 +
05 SQRT
06 STO O
07 1
08 AGM
09 ST+ X
10 1/X
11 PI
12 *
13 RTN

14 LBL "EK"
15 XEQ "KK"
16 RCL 07
17 X^2
18 1
19 AGM2
20 *
21 END
```

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

(*) See Article by Semjon Adlag,http://www.ams.org/notices/201208/rtx120801094p.pdf

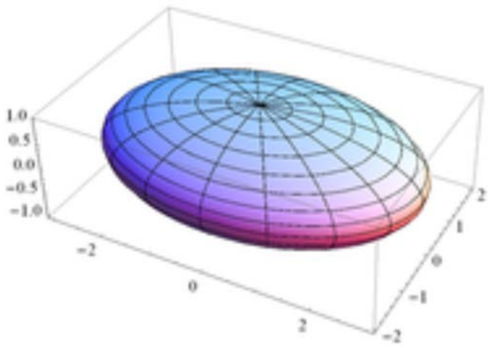
# 2.- Application Examples.

The following two examples should illustrate the applicability of these special functions in the geometry subjects related to ellipses and ellipsoids – and therefore provide some context to their origins and development.

## Example 1.- Surface Area of an Ellipsoid. { SAE }

SAE is a direct application of the Carlson Symmetrical Integral of second kind, **CRG**, used to calculate the surface area of an scalene ellipsoid (i.e. not of revolution):

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1,$$




which formula is:

$$\text{Area} = 4\pi.\mathbf{R}_G(\ a^2b^2\ ,\ a^2c^2\ ,\ b^2c^2\ )$$

with c < b < a

Example: a=2, b=4, c=9 -> A=283.4273843

## Example 2.- Ellipse parameters. { EECC , -/+ }

A related magnitude appearing in formulas related to ellipses is the ratio (a-b)/(a+b), sometimes squared. There's no "proper name" for this parameter (unlike eccentricity) – but regardless the sub-function -/+ (appropriately also without a proper name) in the Auxiliary FAT (the very last one in the catalog) is available to compute it using the values in Y and X registers.

Example: for Y=1 and X=3, -/+ returns –0.5

Using this function we can re-write the **ELIPK** program as follows:

| | | | |
|---|---|---|---|
| 01 | LBL "ELIPK | 10 | * |
| 02 | SQRT | 11 | 1/X |
| 03 | 1 | 12 | PI |
| 04 | X<>Y | 13 | * |
| 05 | **-/+** | 14 | X<>Y |
| 06 | RCL X | 15 | 1 |
| 07 | 1 | 16 | + |
| 08 | **AGM** | 17 | * |
| 09 | 4 | 18 | END |

## Example 3.- Perimeter of the Ellipse. { ELP }

For an ellipse with semi-major axis a and semi-minor axis b and eccentricity e , the complete elliptic integral of the second kind is equal to one quarter of the perimeter C of the ellipse measured in units of the semi-major axis . In other words:

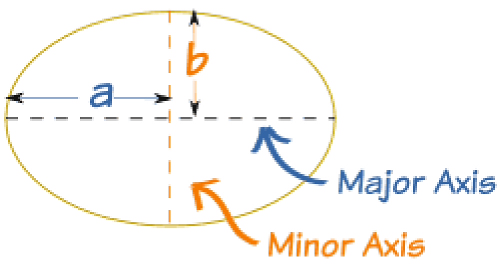
$$c = 4aE(e). \quad \text{, with:} \quad e = \sqrt{1 - b^2/a^2}.$$

or more compactly in terms of the incomplete integral of the second kind E(Φ, k), as:

$$E(k) = E(\tfrac{\pi}{2}, k) = E(1; k).$$

Function **ELP** is available in the auxiliary FAT. It is a FOCAL program like the one listed below, which calculates the perimeter from the semi-axis values input in Y and X stack registers – a sweet and short application of the Elliptic Integrals at work. Note how the (pesky) input conventions are observed: the parameter k needs to be squared!

| | | |
|---|---|---|
| 1 | LBL "ELP" | |
| 2 | EECC | *eccentricity* |
| 3 | LASTX | *semi-axis a* |
| 4 | X<>Y | |
| 5 | X^2 | *e^2* |
| 6 | ELIPE | *complete elliptic 2nd. Kind* |
| 7 | * | *a*E(e^2)* |
| 8 | 4 | |
| 9 | * | *4a*E(e^2)* |
| 10 | END | *done!* |




Where we have also put **EECC** to work as a nice shortcut for the calculations, and one of the nice things it does is making sure the larger semi-axis is used as denominator, regardless of its location in the stack (either X- or Y- register).

Note as well that no data registers are used with this scheme.

*Example:* calculate the perimeter for a=3 and b=2

3, ENTER^, 2, **ΣF$** "ELP" -> 15.86543959

**Example 4.- Period of a Simple Pendulum.**

The differential equation which represents the motion of a simple pendulum is:

$$\frac{d^2\theta}{dt^2} + \frac{g}{\ell}\sin\theta = 0$$

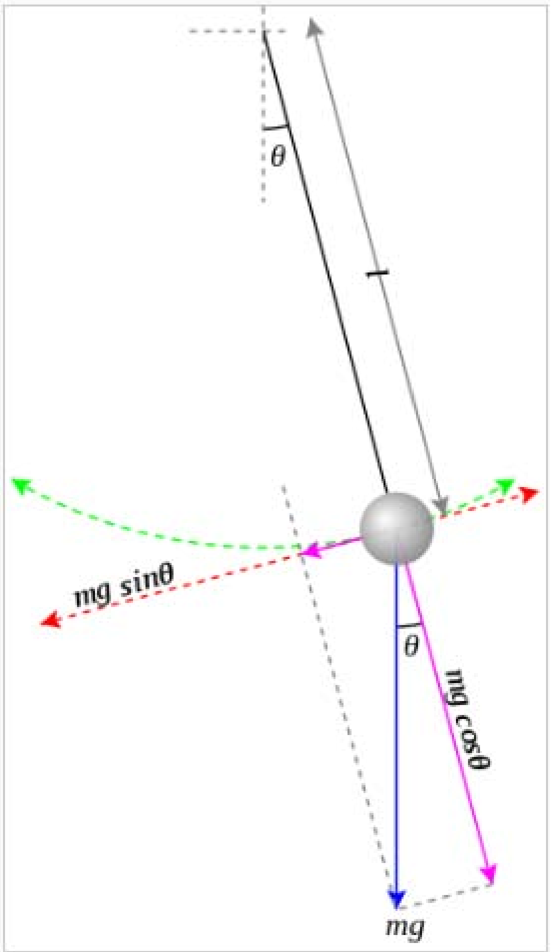
where $g$ is acceleration due to gravity, $\ell$ is the length of the pendulum, and $\theta$ is the angular displacement.

For amplitudes beyond the small angle approximation, one can compute the exact period by first inverting the equation for the angular velocity obtained from the energy method (Eq. 2),

$$\frac{dt}{d\theta} = \sqrt{\frac{\ell}{2g}}\frac{1}{\sqrt{\cos\theta - \cos\theta_0}}$$

which after integration and substitution leads to an expression in function of the complete elliptic integral of the first kind:

$$T = 4\sqrt{\frac{\ell}{g}}\,K\left(\sin\left(\frac{\theta_0}{2}\right)\right)$$




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 |

## Example 5.- Mutual inductance of two coaxial circular coils.

```
LBL "MIND"
"R1=?"
PROMPT
STO 06
"R2=?"
PROMPT
STO 07
LBL 00
"d=?"
PROMPT
LBL C
STO 05
RCL 07
RCL 06
*
4
*
RCL 06
RCL 07
+
X^2
RCL 05
X^2
+
/
STO 05
ELIPK  (ΣFL# 43)
STO 08
RCL 05
ELIPE(ΣFL# 41)
STO 09
E
RCL 05
2
/
-
RCL 08
*
RCL 09
-
PI
*
8 E-7
*
RCL 06
RCL 07
*
RCL 05
/
SQRT
*
"MI="
ARCL X
PROMPT
GTO 00
END
```

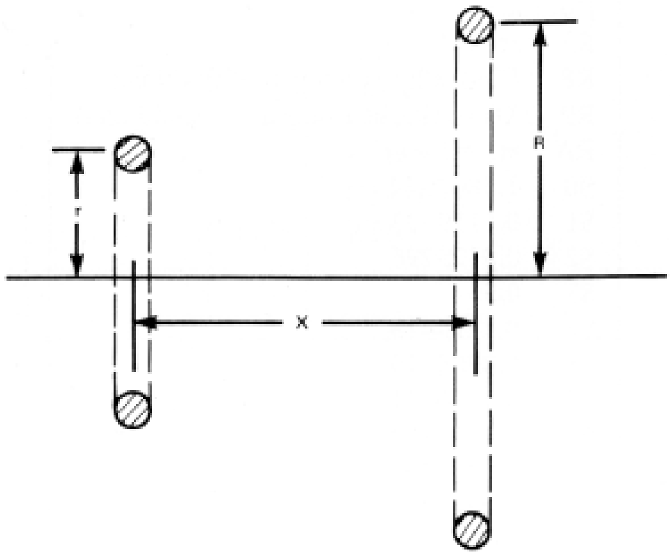
This example shows a practical utilization of functions **ELIPK** and **ELIPE** to calculate the mutual inductance between two coaxial circular coils or radius r1 and r2, separated a distance “d”. The example is taken from page# 83 of the NASA SP-42 document, “Space Resources and Space settlements”.

Note the conventions used in the definition, especially for the “k” parameter – not squared!

Test cases: with r1=0.2, r2=0.25

1. d= 0.1 ->MI=2,48787E-7
2. d= 0.2 ->MI=1,23957E-7

These results are in henries.




$M$= mutual inductance of coil pair (henries)

$$= \frac{8\pi \times 10^{-7}\sqrt{rR}}{k}\left[\left(1 - \frac{k^2}{2}\right)K - E\right]$$

where

$$k^2 = m = \frac{4rR}{[(R+r)^2 + x^2]}$$

Complete elliptic integrals of the first and second kind are

$$E(m) = \int_0^{\pi/2} \sqrt{1 - m\sin^2\theta}\, d\theta$$

$$K(m) = \int_0^{\pi/2} d\theta/\sqrt{1 - m\sin^2\theta}$$

# 3.- Jacobi Elliptic functions.

In mathematics, the Jacobi elliptic functions are a set of basic elliptic functions, and auxiliary theta functions, that are of historical importance. Many of their features show up in important structures and have direct relevance to some applications (e.g. the equation of a pendulum). They also have useful analogies to the functions of trigonometry, as indicated by the matching notation sn for sin. They were introduced by Carl Gustav Jakob Jacobi (1829).

## Definition as inverses of elliptic integrals

There is a simpler, but completely equivalent definition, giving the elliptic functions as inverses of the incomplete elliptic integral of the first kind. Let

$$u = \int_0^{\phi} \frac{d\theta}{\sqrt{1 - m\sin^2\theta}}.$$

Then the elliptic functions sn(u,m), cn(u,m), and dn(u,m) are given by:

$\mathrm{sn}\,(u,m) = \sin(\Phi)$, $\mathrm{cn}\,(u,m) = \cos(\Phi)$, and $\mathrm{dn}\; u = \sqrt{1 - m\sin^2\phi}$

Here, the angle Φ is called the amplitude. On occasion, dn(u) = Δ(u) is called the delta amplitude. In the above, the value m is a free parameter, usually taken to be real, 0 ≤ m ≤ 1, and so the elliptic functions can be thought of as being given by two variables, the amplitude Φ and the parameter m.

The elliptic functions can be given in a variety of notations, which can make the subject unnecessarily confusing. Elliptic functions are functions of two variables. The first variable might be given in terms of the **amplitude** φ, or more commonly, in terms of *u* given below. The second variable might be given in terms of the **parameter** *m*, or as the elliptic modulus *k*, where $k^2 = m$, or in terms of the modular angle α, where $m = \sin^2 \alpha$.

## Formulae and Methodology.

The implementation is based on the Gauss transformation, with the formulas used being:

With $m' = 1-m$, let $\mu = [(1-\mathrm{sqrt}(m')/(1+\mathrm{sqrt}(m')]^2$ and $v = u/(1+\mathrm{sqrt}(\mu))$, we have:

$$\mathrm{sn}(u|m) = [(1 + \mathrm{sqrt}(\mu))\,\mathrm{sn}(v|\mu)] / [1 + \mathrm{sqrt}(\mu)\,\mathrm{sn}^2(v|\mu)]$$
$$\mathrm{cn}(u|m) = [\mathrm{cn}(v|\mu)\,\mathrm{dn}(v|\mu)] / [1 + \mathrm{sqrt}(\mu)\,\mathrm{sn}^2(v|\mu)]$$
$$\mathrm{dn}(u|m) = [1 - \mathrm{sqrt}(\mu)\,\mathrm{sn}^2(v|\mu)] / [1 + \mathrm{sqrt}(\mu)\,\mathrm{sn}^2(v|\mu)]$$

These formulas are applied recursively until μ is small enough to use.

The program calculates the three functions simultaneously, returning the result in the stack registers X [sn], Y [cn], and Z [dn]. The input parameters are the amplitude m, and the argument u – expected in Y and X respectively before calling **JEF**.

Two functions are included in the module, JEF and AJF. The main program is **JEF**, which can be used to calculate the results for any value of the amplitude m (*). **AJF** is a MCODE funtion used to speed up the calculations, applicable when the amplitude lies between 0 and 1. You could use **AJF** directly in this case, since **JEF** does nothing but calling it in that circumstance.

*(*) If m < -9999999999 the program can give wrong results.*

*Example 1*- Evaluate sn ( 0.7 | 0.3 ) cn ( 0.7 | 0.3 ) dn ( 0.7 | 0.3 )

| | | | |
|---|---|---|---|
| 0.3, ENTER^, 0.7, **XEQ** "JEF" | | -> | sn ( 0.7 \| 0.3 ) = 0.632304776 |
| | RDN | -> | cn ( 0.7 \| 0.3 ) = 0.774719736 |
| | RDN | -> | dn ( 0.7 \| 0.3 ) = 0.938113640 |

*Example 2* - Likewise for x=0.7 and amplitudes { 1, 2, -3 }

| | | |
|---|---|---|
| sn ( 0.7 \| 1 ) = 0.604367777 | sn ( 0.7 \| 2 ) = 0.564297007 | sn ( 0.7 \| -3 ) = 0.759113421 |
| cn ( 0.7 \| 1 ) = 0.796705460 | cn ( 0.7 \| 2 ) = 0.825571855 | cn ( 0.7 \| -3 ) = 0.650958382 |
| dn ( 0.7 \| 1 ) = 0.796705460 | dn ( 0.7 \| 2 ) = 0.602609138 | dn ( 0.7 \| -3 ) =1.651895746 |

*Example 3*.- Let's verify the inverse relationship between the Jacobi Elliptic functions and the Elliptic Integral – for a given elliptic modulus (k) that will remain constant in both cases. The expression to verify can be written as:

$$\Phi = \text{asin} ( \text{sn} [k ; F ( k | \Phi) ]$$

Let's use the values $\Phi$ = 84 and k =0.7 - We start by obtaining the value of F:

0.7, ENTER^, 84, **XEQ** "ELIPF" -> F ( 84° | 0.7 ) = 1.884976271

Then we use this intermediate result (and the initial parameter) as input for JEF as follows:

0.7, X<>Y, **XEQ** "JEF" -> sn (0.7 | F(84° | 0.7) = 0.994521895

And finally get the arc sine of the sn value to recover the original amplitude:

ASIN => 84.00000002

Which matches the initial value with an accuracy of E-8.

## Final remarks on the Jacobi Elliptic functions.

Note the interesting role of the parameter m as it moves from 0 to 1. The condition m=0 causes the functions to become the same as the trigonometric sin and cos, whereas in the other extreme for m=1 they become the hyperbolic tanh and sech. In more proper terms, these functions are doubly periodic generalizations of the trigonometric functions satisfying:

sn ( v | 0 ) = sin v ; cn ( v | 0 ) = cos v ; and dn ( v | 0 ) = 1
sn ( v | 1 ) = tanh v ; cn ( v | 0 ) = sech v ; and dn ( v | 1 ) = sech v

The figures in next page represent three intermediate stages; observe the tendency as the elliptic modulus k varies towards both ends of the range. Quite a remarkable behavior showing how the interrelationships amongst seemingly unrelated topics appear.

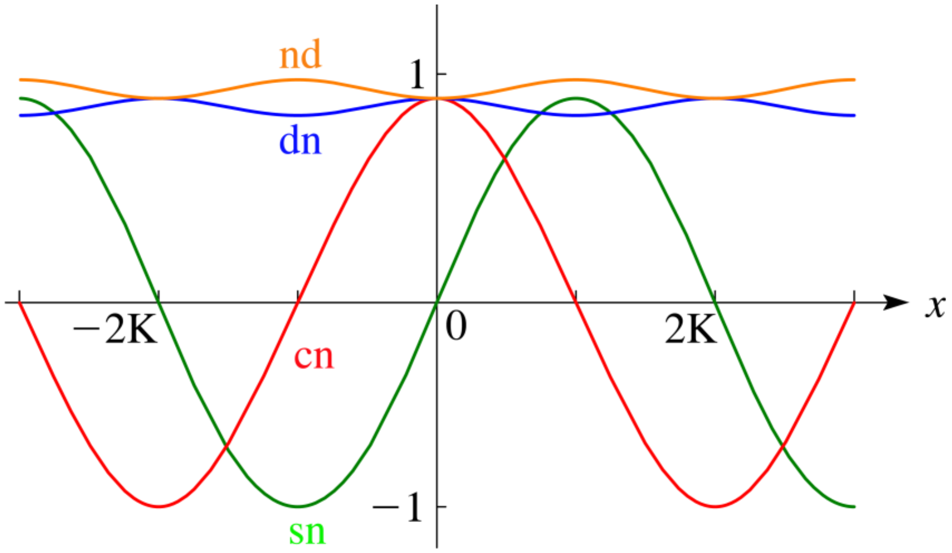


Figure 22.3.1: $k = 0.4$, $-3K \le x \le 3K$, $K = 1.6399$ ....

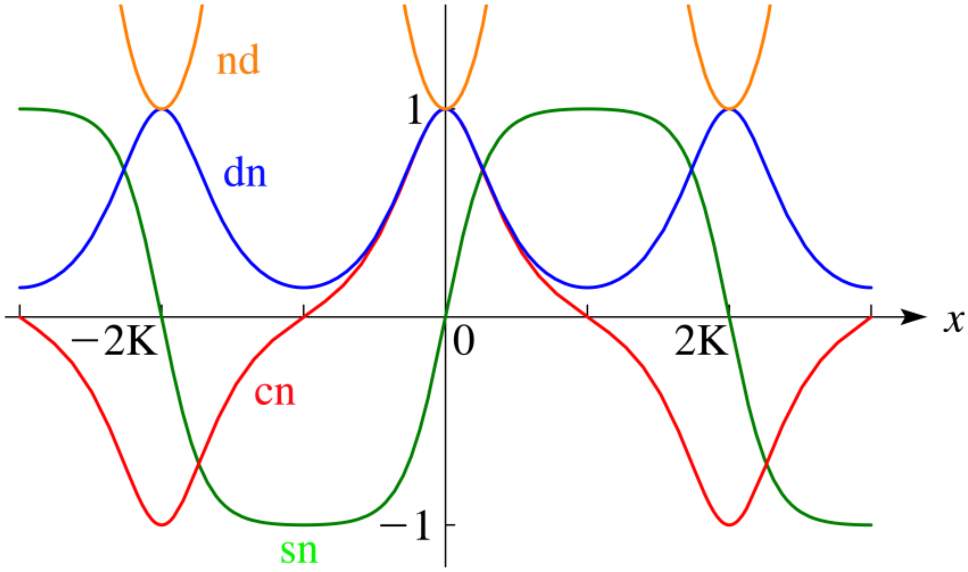


Figure 22.3.3: $k = 0.99$, $-3K \le x \le 3K$, $K = 3.3566$ ....

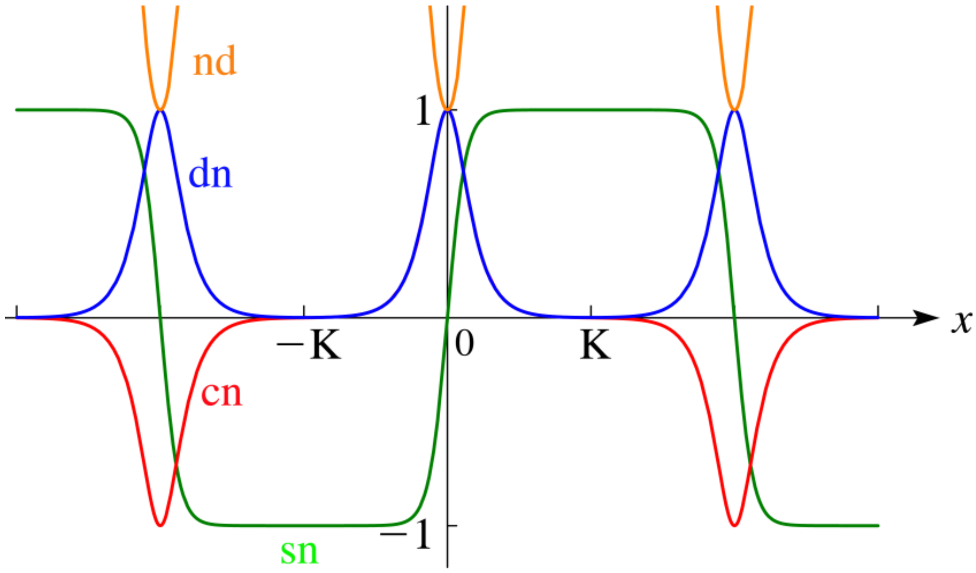


Figure 22.3.4: $k = 0.999999$, $-3K \le x \le 3K$, $K = 7.9474$ ....

## (Jacobian) Theta Functions. {THETA}

There are several closely related functions called Jacobi theta functions, and many different and incompatible systems of notation for them. One Jacobi theta function (named after Carl Gustav Jacob Jacobi) is a function defined for two complex variables z and τ, where z can be any complex number and τ is confined to the upper half-plane, which means it has positive imaginary part. It is given by the formula:

$$\vartheta(z;\tau) = \sum_{n=-\infty}^{\infty} \exp(\pi i n^2 \tau + 2\pi i n z) = 1 + 2\sum_{n=1}^{\infty} \left(e^{\pi i \tau}\right)^{n^2} \cos(2\pi n z)$$

The SandMath uses the following definitions as per JM Baillard, with: $q = e^{-\pi K'/K}$ $(0 <= q < 1)$

$$\text{Theta1}(x;q) = 2.q^{1/4}\Sigma_{k>=0}\ (-1)^k\, q^{k(k+1)} \sin(2k+1)x$$
$$\text{Theta2}(x;q) = 2.q^{1/4}\Sigma_{k>=0}\ q^{k(k+1)} \cos(2k+1)x$$
$$\text{Theta3}(x;q) = 1 + 2\,\Sigma_{k>=1}\ q^{k*k} \cos 2kx$$
$$\text{Theta4}(x;q) = 1 + 2\,\Sigma_{k>=1}\ (-1)^k\, q^{k*k} \cos 2kx$$

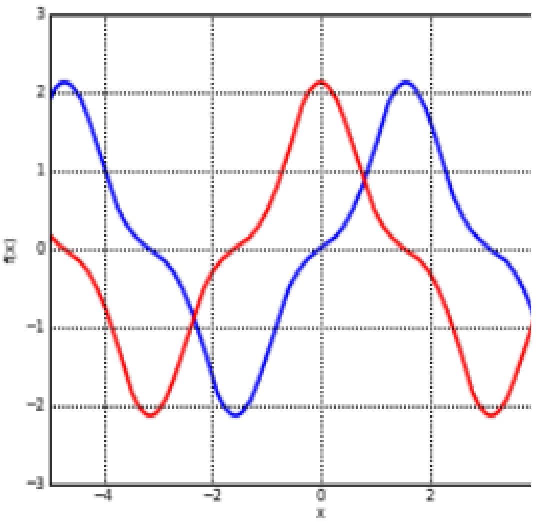
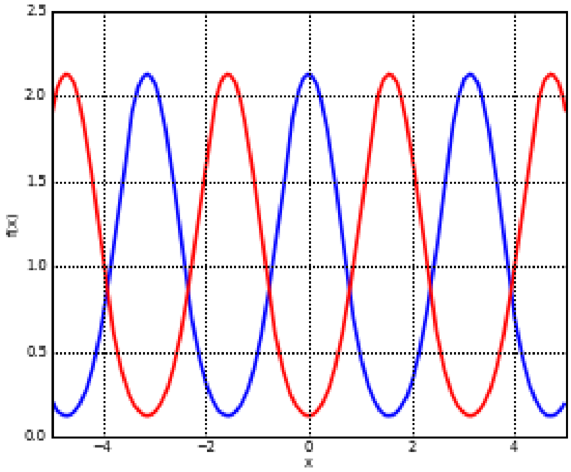
Use the function "**THETA**" to calculate any of these, with the function index in Z, and the two arguments (q, x) in Y and X. The result is returned in X.

| Stack | Input | Output |
|---|---|---|
| T | n# | n# |
| Y | q | q |
| X | x | Theta(n,q,x) |

*Example:* Compute Theta1(x;q) , Theta2(x;q) , Theta3(x;q) , Theta4(x;q) for x = 2 ; q = 0.3

1, ENTER^, 0.3, ENTER^, 2, XEQ "THETA" -> 1.382545289
2, ENTER^, 0.3, ENTER^, 2 XEQ "THETA" -> -0.488962527
3, ENTER^, 0.3, ENTER^, 2, XEQ "THETA" -> 0.605489938
4, ENTER^, 0.3, ENTER^, 2, XEQ "THETA" -> 1.389795845

The picture below shows the Theta functions 1-2 (on the left) and 3-4 (right) for a range of x between [-5,5] and a second argument y kept constant. Note the similar shapes between cn with T1,T2, as well as sn with T3,T4

## Whittaker Functions. { $\boxed{\text{WHIM}}$ , $\boxed{\text{WHIW}}$ } - <**Requires SandMath**>

In mathematics, a Whittaker function is a special solution of Whittaker's equation, a modified form of the confluent hypergeometric equation introduced by Whittaker (1904) to make the formulas involving the solutions more symmetric.

Whittaker's equation is

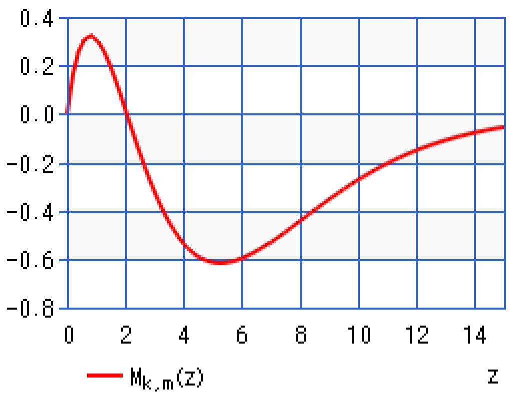
$$\frac{d^2w}{dz^2} + \left(-\frac{1}{4} + \frac{\kappa}{z} + \frac{1/4 - \mu^2}{z^2}\right) w = 0$$

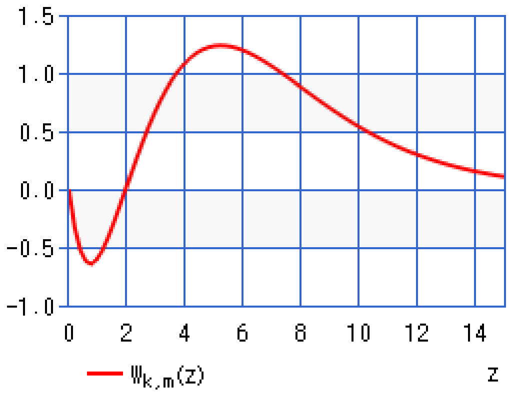
It has a regular singular point at 0 and an irregular singular point at ∞. Two solutions are given by the Whittaker functions $M_{\kappa,\mu}(z)$, $W_{\kappa,\mu}(z)$, defined in terms of Kummer's confluent hypergeometric functions M and U by

$$M_{\kappa,\mu}(z) = \exp(-z/2)\, z^{\mu+\frac{1}{2}} M\left(\mu - \kappa + \frac{1}{2}, 1 + 2\mu; z\right)$$

$$W_{\kappa,\mu}(z) = \exp(-z/2)\, z^{\mu+\frac{1}{2}} U\left(\mu - \kappa + \frac{1}{2}, 1 + 2\mu; z\right).$$

The graphics below show both functions for the particular case k=2 and m=0.5




DATA REGISTERS: R00 thru R02:
Flags: none.

| Stack | Input | Output |
|---|---|---|
| Z | K | / |
| Y | μ | / |
| X | x | **W**(k,m,x) |

*Examples:*

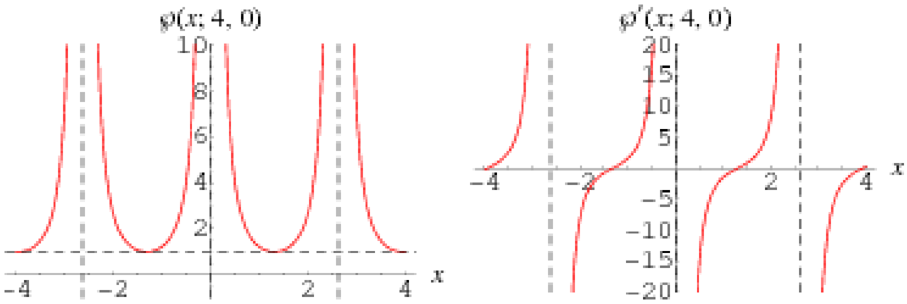
2, SQRT, 3, SQRT , PI, **ΣF$** "WHIM" ->**M**( sqrt(2), sqrt(3), $\pi$) = 5.612426206

2, SQRT, 3, SQRT, PI, XEQ "WHIMW" ->**W**( sqrt(2), sqrt(3), $\pi$) =2.177593412

## (Jacobian) Weierstrass Elliptic Functions. $\wp$

In mathematics, Weierstrass's elliptic functions are elliptic functions that take a particularly simple form; they are named for Karl Weierstrass. This class of functions are also referred to as P-functions and generally written using the symbol $\wp$ ., with variables $\wp$(x, g2, g3)

Relation to Jacobi elliptic functions.For numerical work, it is often convenient to calculate the Weierstrass elliptic function in terms of Jacobi's elliptic functions. The basic relations are described in JM Bailard's web pages, depending on the roots of the polynomial p(x) = $4x^3 - g_2.x - g_3$ - where g2, g3 are the function's "elliptic invariants".




The above plots show the Weierstrass elliptic function P(x ;g2,g3) and its derivative P'(x;g2,g3) for elliptic invariants g2=4 and g3=0 along the real axis.

The program uses data registers R00 – R07, as well as several auxiliary functions as resources. The results include both the function value and its first derivative in the stack – plus the half-periods in R09 & R10.

| STACK | INPUTS | OUTPUTS |
|---|---|---|
| Z | $g_3$ | / |
| Y | $g_2$ | **P'**$(x;g_2;g_3)$ |
| X | x | **P**$(x;g_2;g_3)$ |

Example1: Calculate $\wp$(x;g2;g3) &$\wp$'(x;g2;g3) for x=2, g2=4, g3=1

1, ENTER^, 4, ENTER^, 2, XEQ "WEF" -> P(2;4;1) = 4.950267724
X<>Y ->P'(2;4;1) = 21.55057197

We have R09 = 1.225694692 & R10 = 1.496729323 ( Ω&Ω' because F01 is clear )
Therefore the primitive half-periods are: 1.225694692 & 1.496729323 i

Example2: Calculate $\wp$(x;g2;g3) &$\wp$'(x;g2;g3) for x=1, g2=2, g3=3

3, ENTER^, 2, ENTER^, 1 XEQ "WEF" -> P(1;2;3) = 1.214433709
X<>Y -> P'(1;2;3) = -1.317406193

We have R09 = 1.197220889 & R10 = 2.350281226 ( Ω2&Ω'2 because F01 is set )
Whence: Ω = 0.598610445 - 1.175140613 i &Ω' = 0.598610445 + 1.175140613 i

# 4.- Orthogonal Polynomials.

The last section in the module includes a comprehensive function set to calculate orthogonal polynomials. Some of the functions are written in MCODE, and therefore feature a speed and accuracy advantage over equivalent user code routines.

All these routines use a similar convention for the data entry parameters: the order in the Y-register and the evaluation point in the X-register. The generalized Laguerre polynomials require a third parameter, which is to be entered in the Z-register. Upon completion, the result is left in the X register, and for the MCODE functions the original evaluation point is saved in the LastX register as well. No data registers are used.

*Legendre Polynomials*

$n.P_n(x) = (2n-1).x.P_{n-1}(x) - (n-1).P_{n-2}(x)$ ; $P_0(x) = 1$ ; $P_1(x) = x$

Examples: Calculate $P_7(4.9)$

7, ENTER^, 4.9, XEQ"LEG" -> $P_7(4.9)$ =1,698,444.018

*Laguerre Polynomials.*

$n! . L_n(x) = (2n-1-x).L_{n-1}(x) - (n-1)^2.L_{n-2}(x)$ ; $L_0(x) = 1$ ; $L_1(x) = 1 - x$, and:
$L_0^{(a)}(x) = 1$ ; $L_1^{(a)}(x) = a+1-x$ ; $n.L_n^{(a)}(x) = (2.n+a-1-x).L_{n-1}^{(a)}(x) - (n+a-1).L_{n-2}^{(a)}(x)$

Examples: Calculate $L_7(3.14)$ and $L_7^{(1.4)}(Pi)$

7, ENTER, 3.14, XEQ "LAG" -> L(7, 3.14) = -0.978658720
1.4, ENTER^, 7, PI, XEQ "LANX" -> L(1.4, 7, 3.14)= 1.688893513

*Hermite Polynomials.*

$H_n(x) = 2x.H_{n-1}(x) - 2(n-1).H_{n-2}(x)$ ; $H_0(x) = 1$ ; $H_1(x) = 2x$

Examples: Calculate $H_7(3.14)$

7, ENTER^, 3.14, XEQ "HMT" -> 73,726.24325

*Chebyshev Polynomials of the first and second kind*

$T_n(x) = 2x.T_{n-1}(x) - T_{n-2}(x)$ ; $T_0(x) = 1$ ; $T_1(x) = x$ - first kind
$U_n(x) = 2x.U_{n-1}(x) - U_{n-2}(x)$ ; $U_0(x) = 1$ ; $U_1(x) = 2x$ - second kind.

Examples: Compute $T_7(0.314)$ and $U_7(0.314)$

7, ENTER^, .314, XEQ "CHBT" ->$T_7(0.314)$ = -0.582815680
7, ENTER^, 0.314, XEQ "CHBU" ->$U_7(0.314)$ = -0.786900700

*Fibonnaci's Polynomials.*

$F_0(x) = 0 \quad F_1(x) = 1$ and $F_n(x) = x\, F_{n-1}(x) + F_{n-2}(x)$ if $n > 1$

Example: Compute $F_8(1/\pi)$

8, PI, 1/X,. XEQ "FIB" -> $F_8(1/\pi) = 1.615692565$

*Bell Polynomials. Recurrent expression:*

$$B_{n,k} = \sum_{i=1}^{n-k+1} \binom{n-1}{i-1} x_i B_{n-i,k-1}$$

$$B_{0,0} = 1; B_{n,0} = 0 \text{ for } n \geq 1; B_{0,k} = 0 \text{ for } k \geq 1.$$

Example: Compute $B_8(1/\pi)$

8, PI, 1/X,. XEQ "FIB" -> $F_8(1/\pi) = 3.4051766$ E10

*Bessel Polynomials. Recurrent expression:*

$$y_n(x) = (2n-1)x\, y_{n-1}(x) + y_{n-2}(x)$$

$$y_0(x) = 1\,;\; y_1(x) = x + 1$$

Particular Values:

$$\begin{aligned}
y_0(x) &= 1 \\
y_1(x) &= x + 1 \\
y_2(x) &= 3x^2 + 3x + 1 \\
y_3(x) &= 15x^3 + 15x^2 + 6x + 1 \\
y_4(x) &= 105x^4 + 105x^3 + 45x^2 + 10x + 1 \\
y_5(x) &= 945x^5 + 945x^4 + 420x^3 + 105x^2 + 15x + 1
\end{aligned}$$

Example: Compute $y_5(1/\pi)$

5, PI, 1/X,. XEQ "*BSSL*" -> $F_5(1/\pi) = 42.74840691$

We can verify the last result using the y5(x) polynomial above and function **PVAL**. Which returns a value of $F_5(1/\pi) = 42.74840688$

## Coefficients of Orthogonal Polynomials

Besides obtaining their values using recurrent expressions, an interesting subject is the calculation of the coefficients of the orthogonal polynomials. A considerably large program written by JM Baillard accomplishes this goal for the most frequently used cases. You're encouraged to see JM's webpage at:

http://hp41programs.yolasite.com/orthopoly.php

The program requires at least two inputs: the index of first register for coefficient storage in Y, and the order of the polynomial in X. Because the program itself uses the first 11 data registers, the first register available for coefficients must be 11 or greater. When a third parameter is required it is expected to be in the Z register.

Note that as additional bonus the program returns the coeffs. for the polynomial of degree n-1 as well as the degree "n". Upon completion the X-register has the control word that defines the polynomial in data registers, and the Y-register has the control word for the polynomial of previous order (n-1) as well.

Once the coefficients are calculated and stored in data registers, you can use the evaluation functions to obtain their values, derivatives and primitives. The control word follows the same convention for all programs.

*Example 1*. Find the Legendre polynomial of order n=6

11, ENTER^, 6, XEQ "**LEG+**" -> 11,007 (and X<>Y: 18,023)

Listing those registers we see: R11 = 231/16; R12 = 0; R13 = -315/16
R14 = 0; R15= 106/16; R16 = 0;
R17 = -5/16

Therefore: $L_6(x) = ( 231\ x^6 - 315\ x^4 + 105\ x^2 - 5 ) / 16$

Example 2-. Find the Generalized Laguerre Polynomial $L_n^{(a)}(x)$ with $a = 3 , n = 6$

3, ENTER^, 11, ENTER^, 6, XEQ "**LANX+**" -> 11,007 (and X<>Y: 18,023)

{R11 - R17} give: $L_6^{(3)}(x) = x^6 / 720 - 3\ x^5 / 40 + 3\ x^4 / 2 - 14\ x^3 + 63\ x^2 - 126\ x + 84$
and {R18 - R23}: $L_5^{(3)}(x) = - x^5 / 120 + x^4 / 3 - 14\ x^3 / 3 + 28\ x^2 - 70\ x + 56$

Example 3.- Chebyshev's Polynomials. Find $T_6(x)$ and $U_6(x)$

11, ENTER, 6, XEQ "**CBT+**" -> 11,007 (and X<>Y: 18,023)

{R11 - R17} give: $T_6(x) = 32\ x^6 - 48\ x^4 + 18\ x^2 – 1$
and {R18 - R23}: $T_5(x) = 16\ x^5 - 20\ x^3 + 5\ x$

11, ENTER, 6, XEQ "**CBU+**" -> 11,007 (and X<>Y: 18,023)

{R11 - R17} give: $U_6(x) = 64\ x^6 - 80\ x^4 + 24\ x^2 - 1$
and {R18 - R23}: $U_5(x) = 32\ x^5 - 32\ x^3 + 6\ x$

*Example 4*.- Ultraspherical polynomials. If a = 3 & n = 6 , Cn(a)(x) = ?

3, ENTER^, 11, 6, XEQ "**USP+**" -> 11,007 (and X<>Y: 18,023)

{R11 - R17} give: $C_6^{(3)}(x) = 1792\ x^6 - 1680\ x^4 + 360\ x^2 - 10$
and {R18 - R23}: $C_5^{(3)}(x) = 672\ x^5 - 480\ x^3 + 60\ x$

*Example 5.*- Hermite Polynomials. Find Hermite polynomial of order n = 6

11, ENTER^, 6, XEQ "**HMT+**" -> 11,007 (and X<>Y: 18,023)

{R11 - R17} give: $H_6(x) = 64\ x^6 - 480\ x^4 + 720\ x^2 - 120$
and {R18 - R23}: $H_5(x) = 32\ x^5 - 160\ x^3 + 120\ x$

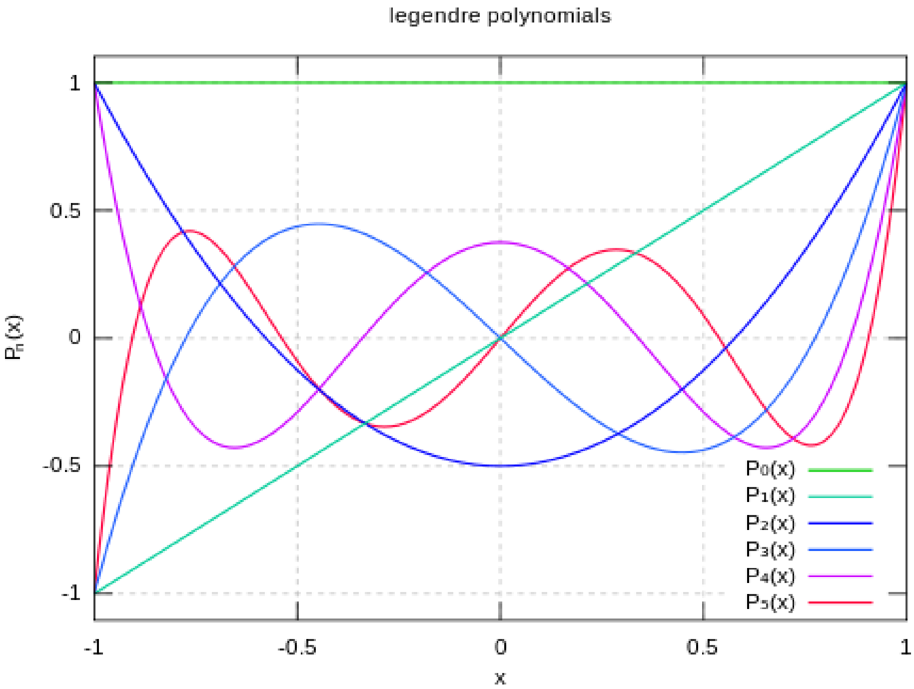
*Example 6*. Jacobi Polynomials. Find Pn(a;b)(x) with a = 3 , b = 4 , n = 6

3, ENTER^, 4, ENTER^, 11, ENTER^, 8, XEQ "JCP+" -> 11,007 (and X<>Y: 18,023)

{R11 - R17} give:
$P_6^{(3;4)}(x) = 423.9375\ x^6 - 133.875\ x^5 - 334.6875\ x^4 + 78.75\ x^3 + 59.0625\ x^2 - 7.875\ x - 1.3125$

and {R18 - R23}:
$P_5^{(3;4)}(x) = 193.375\ x^5 - 56.875\ x^4 - 113.75\ x^3 + 22.75\ x^2 + 11.375\ x - 0.875$

Obviously when the coefficients are not integers, the HP-41 may give approximate values - not the fractions directly. Though mathematically equivalent, evaluating these polynomials would often produce p(x) with a lower precision because of cancellation of leading digits, especially for large n-values: the signs of the coefficients alternate.



## Polynomial Primitive and Derivatives

Lastly, a few other functions deal with the calculation of derivatives and primitive (that vanishes for x=0) for any polynomial – also written in MCODE and using the 13-digit O/S routines for intermediate calculations.

For example, evaluate the $H_6(x)$ polynomial and its derivatives & primitive at the point x=1.

11,017, 1, ENTER^, XEQ "**PVAL**" -> 184.0000000
RDN, 1, XEQ "**dPL**" -> -96.00000000
RDN, 1, XEQ "**dPL2**" -> -2,400.000000
RDN, 1, XEQ "**ITPL**" -> 33.14285712