Note of Numeric Calculus

Moments Regression

Method for approximating a regular function f(x) from its moments

Leonardo Volpi

Dic. 2006

Foxes Team

Contents

Introduction

The moment regression means the objective of finding a function f(x) from the knowledge of its moments. In other words, givens a set of moments we want to search a function f(x), if exists, that best fits the given moments in the specified integration interval.
The numerical solution of this problem is not simple. Using theoretic formulas can easily fail for high complexity or instability (see the moments theorem). In this paper we show a method, quite simple, for finding f(x) as expansion of Legendre polynomials that can be successfully used for fitting smooth regular functions using up to 10 ÷ 20 moments

Moments Definition

Given a function f(x), regular and integrable in the range [a, b], we define[1]

$$m_0 = \int_a^b f(x)\,dx \qquad m_1 = \int_a^b f(x)x\,dx \qquad m_2 = \int_a^b f(x)x^2\,dx \qquad .... \qquad m_n = \int_a^b f(x)x^n dx$$

The quantities [m_0, m_1, m_2...m_n] are called *moments* (or also *raw moments*) of the function f(x)
We note that m_0 represents the mean $\bar{y}$ and m_1 represents the mean $\bar{x}$.
Very diffused are also the *central moments*, defined as

$$\overline{m}_0 = \int_a^b f(x)\,dx \qquad \overline{m}_1 = \int_a^b f(x)(x-\bar{x})\,dx \qquad .... \qquad \overline{m}_n = \int_a^b f(x)(x-\bar{x})^n\,dx$$

There are relations between the raw moments and the central moments so they can be derived from each others. For example, we can derive the raw moments from the central moments using the following iterative algorithm.

$$m_0 = \overline{m}_0\,, \quad m_1 = \overline{m}_1 + \bar{x}\,m_0\,, \quad m_2 = \overline{m}_2 + 2\bar{x}\,m_1 - \bar{x}^2 m_0\,,...$$

and, generally

$$m_n = \overline{m}_n - \sum_{i=1}^{n} \binom{n}{i} (-\bar{x})^i m_{n-i}$$

The above formula computes the moment n^{th} using the correspondent n^{th} central moment and the previous moments m_{n-1}, m_{n-2}, ... m_2, m_1, m_0. Because it always possible to convert the central moments into raw moments, in the following papers we always refer to the raw moments for the sake of simplicity

Legendre Polynomials

We assume to approximate the unknown function f(x) with a polynomial of n^{th} degree

$$f(x) \cong a_0 + a_1 x + ...a_n x^n$$

The unknown coefficients [a_0, a_1, ...a_n] would be determined by n+1 conditions, such as those derived form the n+1 moments. Unfortunately the direct approach leads to a linear system with a Vandermonde matrix that it is hill-conditioned even for low-moderate degree.

[1] The symbols used here are chosen for our convenience. In other documents you may find other symbols.

In order to gain robustness, we use the Legendre orthogonal polynomials expansion
Here are the first Legendre polynomials, from 0 to 6 degree

$$\left[1,\, z\,,\frac{3z^2-1}{2},\frac{5z^3-3z}{2},\frac{35z^4-30z^2+3}{8},\frac{63z^5-70z^3+15z}{8}\right]$$

These polynomials are orthogonal respect to the symmetrical interval [-1, 1]. Thus, the first step is to transform the original integration interval [a, b] into the Legendre interval [-1, 1]. This step is called *normalization*, and it can be always performed with the following linear transformations

$$z=\left(\frac{2}{b-a}\right)x-\left(\frac{a+b}{2}\right) \qquad x=\left(\frac{b-a}{2}\right)z+\left(\frac{a+b}{2}\right)$$

Using the normalized variable z, the Legendre orthogonal polynomials expansion becomes

$$g(z)\cong c_0P_0+c_1P_1(z)+c_2P_2(z)...c_nP_n(z)$$

where: $g(z)=f\left(k\,z+q\right)$ and k = (b - a)/2 and q = (a+b)/2
Of course, the moments m_n of the original function f(x) are different from the moments $\hat{m}_n$ of the new function g(z). They are called ***normalized moments*** respect to the interval [-1, 1]

Normalized Moments

In order to save the information, the original moments must be transformed into normalized moment by the integration variable substitution.

$$m_0=\int_a^b f(x)dx=k\int_{-1}^{1} g(z)dz=k\,\hat{m}_0 \quad \Rightarrow \quad \hat{m}_0=m_0/k$$

$$m_1=\int_a^b f(x)x\,dx= \quad k\int_{-1}^{1} g(z)(kz+q)\,dx= \quad k^2\int_{-1}^{1} g(z)zdz+kq\int_{-1}^{1} g(z)dx= \quad k^2\hat{m}_1+q\,k\,\hat{m}_0$$

$$\Rightarrow \quad \hat{m}_1=m_1/k^2-q/k\,\hat{m}_0$$

For the general n^{th} normalized moment:

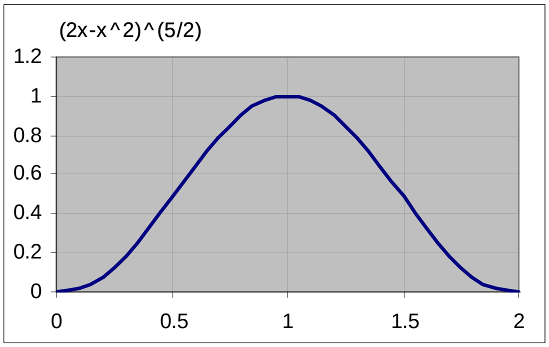
$$\hat{m}_n=\frac{m_n}{k^{n+1}}-\sum_{j=1}^{n}\binom{n}{j}a^{\,j}\hat{m}_{n-j}$$

where $a=q/k=(a+b)/(b-a)$
The above formula computes the normalized moment n^{th} using the correspondent n^{th} moment and the previous normalized moments $\hat{m}_{n-1}$, $\hat{m}_{n-2}$,...$\hat{m}_1$, $\hat{m}_0$.

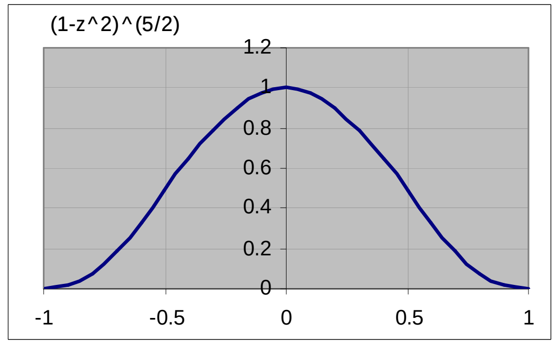
Let's see with practical examples how the normalization works.

Example. Assume to have the following function $f(x)=(2x-x^2)^{5/2}$ in the range $0\le x\le 2$
Performing the substitution x = z+1 we have the following normalized function $g(z)=(1-z^2)^{5/2}$ in the range [- 1, 1]. Note that, in that case the transformation acts a simple translation. The shape remains unchanged but the extreme shifts to the origin. The two functions are shown in the following graphs

Original function f(x)



Normalized function g(z)



Computing analytically the moments of both functions is possible, but not simple. A faster way is the numerical quadrature of the function **integr**[1] in Excel. A simpler arrangement is shown in the following spreadsheet

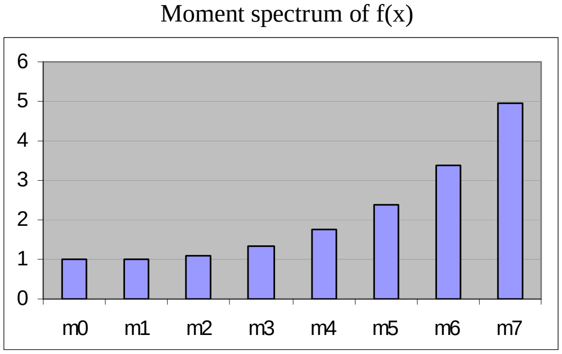
	A	B	C	D	E	F	G	H	I	J
1	**Moment integration**									
2		=integr(B4,C4,D4)								
3		function	a	b	result		function	a	b	result
4	m0	(1-x^2)^(5/2)	-1	1	0.981747704247		(2*x-x^2)^(5/2)	0	2	0.981747704247
5	m1	(1-x^2)^(5/2)*x	-1	1	0		(2*x-x^2)^(5/2)*x	0	2	0.981747704247
6	m2	(1-x^2)^(5/2)*x^2	-1	1	0.122718463031		(2*x-x^2)^(5/2)*x^2	0	2	1.104466167278

The result of the first 8 moments of both functions f(x) and g(z) are shown in the following table.

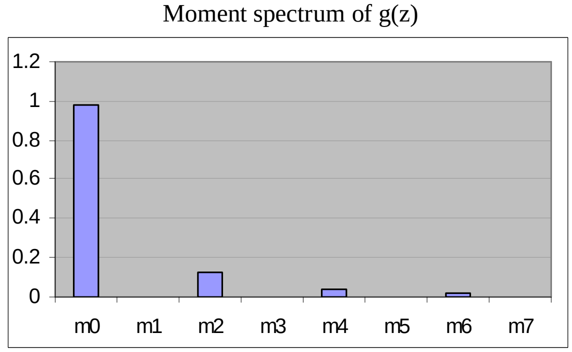
Function f(x)	a	b	result	Function g(z)	a	b	result
(2*x-x^2)^(5/2)	0	2	0.981747704247	(1-x^2)^(5/2)	-1	1	0.981747704247
(2*x-x^2)^(5/2)*x	0	2	0.981747704247	(1-x^2)^(5/2)*x	-1	1	0
(2*x-x^2)^(5/2)*x^2	0	2	1.104466167278	(1-x^2)^(5/2)*x^2	-1	1	0.122718463031
(2*x-x^2)^(5/2)*x^3	0	2	1.349903093339	(1-x^2)^(5/2)*x^3	-1	1	0
(2*x-x^2)^(5/2)*x^4	0	2	1.754874021341	(1-x^2)^(5/2)*x^4	-1	1	0.036815538909
(2*x-x^2)^(5/2)*x^5	0	2	2.393010029102	(1-x^2)^(5/2)*x^5	-1	1	0
(2*x-x^2)^(5/2)*x^6	0	2	3.390097541227	(1-x^2)^(5/2)*x^6	-1	1	0.015339807879
(2*x-x^2)^(5/2)*x^7	0	2	4.954757944871	(1-x^2)^(5/2)*x^7	-1	1	0

The plot of the moments set, similarly with Fourier analysis, is called moments spectrum.

Moment spectrum of f(x)



Moment spectrum of g(z)



The effects of the normalization becomes now more evident. They can be summarized as:

[1] This function belongs to the collection of Xnumbers.xla, add-in for Excel, v5.2, by Foxes Team

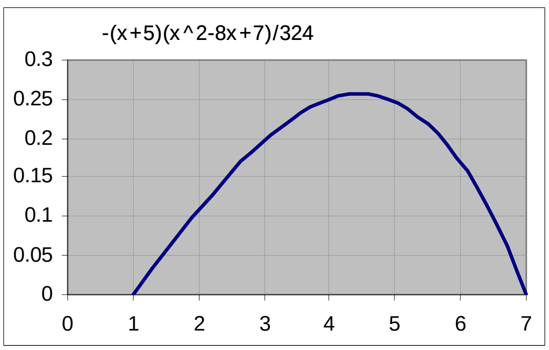
- magnitude reduction of all moments.
- cancellation of odd moments for symmetrical functions
- inhibition of the moments "explosion" effect.

The last effect should be better explained. As we can see, the magnitude of the moments becomes larger as the order increases. For n → ∞ the magnitude blow-up to the infinity. Clearly this is a strong drawback for numerical computing that easily leads to overflow. The normalized spectrum, on the contrary, exhibits an inverse behavior: high order moments have lower values . For n → ∞ the magnitude tend to be negligible.

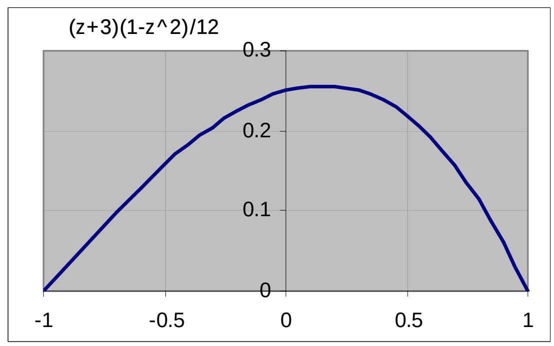
All these advantages contribute greatly to the success of the method.

Example. Assume to have the following function $f(x) = -(x+5)(x^2-8x+7)/32$ in the range $1 \le x \le 7$. Performing the substitution x = 3z + 4 we have the following normalized function $g(z) = (z+3)(1-z^2)/12$ in the range [- 1, 1]. Note that, in that case the transformation acts both a translation and a scaling reduction. The two functions are shown in the following graphs

Original function f(x)

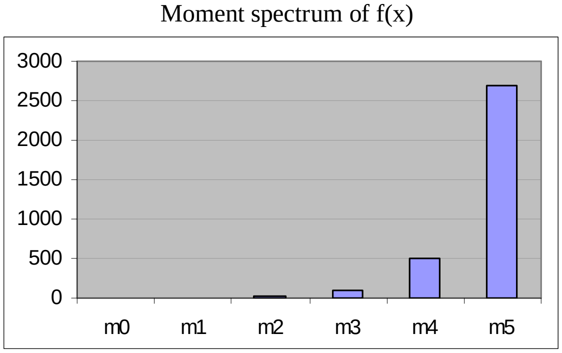


Normalized function g(z)

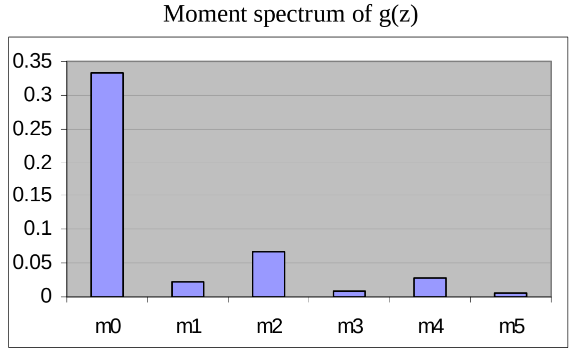


The first 6 raw moments of f(x) are: $[m_0, m_1 ... m_5]$ = [1, 21/5, 97/5, 3359/35, 3495/7, 18887/7]
The first 6 raw moments of g(z) are: $[\hat{m}_0, \hat{m}_1 ... \hat{m}_5]$ = [1/3, 1/45, 1/15, 1/105, 1/35, 1/189]

Moment spectrum of f(x)



Moment spectrum of g(z)



Note that, in this case, the "spectrum explosion" is much more evident and faster of the previous examples. Its is very dangerous from the point of view of numeric calculus. Fortunately, the normalization solves completely this problem.

Legendre Regression Matrix

After the normalization, we have to obtain the coefficients of the Legendre polynomial expansion For the sake of simplicity we suppose to know the first 6 moments $[\hat{m}_0, \hat{m}_1 ... \hat{m}_5]$.

The polynomial to search is thus $c_0 P_0 + c_1 P_1 + c_2 P_2 + c_3 P_3 + c_4 P_4 + c_5 P_5 \cong g(z)$

Substituting the g(z) expansion in the moment integral, we have

$$\hat{m}_n = \int_{-1}^{1} g(z)\, z^n dz \quad \Rightarrow \quad \int_{-1}^{1} z^n \sum_{i=0}^{5} c_i P_i(z)\, dz = \hat{m}_n \qquad \text{for n = 0, 1, 2...5}$$

$$\sum_{i=0}^{5} c_0 \left(\int_{-1}^{1} z^n P_i(z)\, dz \right) = \hat{m}_n \quad \Rightarrow \quad \sum_{i=0}^{5} c_0\, p_{(n+1,\, i+1)} = \hat{m}_n \qquad \text{for n = 0, 1, 2...5}$$

the coefficients $p_{(n+1,\, i+1)}$ defined as $p_{(n+1,\, i+1)} = \int_{-1}^{1} z^n P_i(z)\, dz$ for i = 0..5 and n = 0...5 form a (6x6) matrix **P** independent from the function moments and thus it can be pre-computed once time.

$$\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{2}{3} & 0 & 0 & 0 & 0 \\ \frac{2}{3} & 0 & \frac{4}{15} & 0 & 0 & 0 \\ 0 & \frac{2}{5} & 0 & \frac{4}{35} & 0 & 0 \\ \frac{2}{5} & 0 & \frac{8}{35} & 0 & \frac{16}{315} & 0 \\ 0 & \frac{2}{7} & 0 & \frac{8}{63} & 0 & \frac{16}{693} \end{bmatrix}$$

The matrix **P** = $[p_{(n+1,\, i+1)}]$, thanks to the orthogonal characteristic of the Legendre polynomials $P_i(z)$ is lower triangular with many zeros (sparse matrix). This feature is very important because the linear system

$$\mathbf{P}(c_0, c_1 ... c_5)^T = (\hat{m}_0, \hat{m}_1 ... \hat{m}_5)^T$$

can be solved in a very efficient and fast way by the forward substitution algorithm.
This assures also a good stability and robustness against the round-off errors.

As known, the final solution can be expressed by the following forward recurrent schema

$$c_0 = \frac{\hat{m}_0}{p_{11}}, \ \Rightarrow \ c_1 = \frac{\hat{m}_1 - p_{21} c_0}{p_{22}}, \ \Rightarrow \ c_2 = \frac{\hat{m}_2 - (p_{31} c_0 + p_{32} c_1)}{p_{33}}, \ ...$$

and, in general:

$$c_i = \frac{1}{p_{i+1,i+1}} \left(\hat{m}_i - \sum_{j=0}^{i-1} p_{i+1,\, j+1}\, c_j \right) \qquad \text{for i = 0,1...5}$$

It can be shown that the Legendre regression matrix can be obtained for every (n × n) dimension by the following formula

$$p_{i,j} = \sum_{k=0}^{j-1} \frac{1 - \cos(p\,(k+i))}{(k+i)} b_k^{(j-1)}$$

where $b_k^{(n)}$ are the coefficients of the n^{th} degree Legendre polynomial $P_n(z)$

Regression Examples

Example 1. Assume to have the following 6 normalized moments

$$\hat{m} = \left[\frac{5p}{16},\ 0\ ,\ \frac{5p}{128}\ ,\ 0\ ,\ \frac{3p}{256}, 0 \right]$$

The solution of the moment regression is found solving this 6×6 linear system

$$\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{2}{3} & 0 & 0 & 0 & 0 \\ \frac{2}{3} & 0 & \frac{4}{15} & 0 & 0 & 0 \\ 0 & \frac{2}{5} & 0 & \frac{4}{35} & 0 & 0 \\ \frac{2}{5} & 0 & \frac{8}{35} & 0 & \frac{16}{315} & 0 \\ 0 & \frac{2}{7} & 0 & \frac{8}{63} & 0 & \frac{16}{693} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \frac{5 \cdot \pi}{16} \\ 0 \\ \frac{5 \cdot \pi}{128} \\ 0 \\ \frac{3 \cdot \pi}{256} \\ 0 \end{bmatrix}$$

Solving, we have the following coefficients

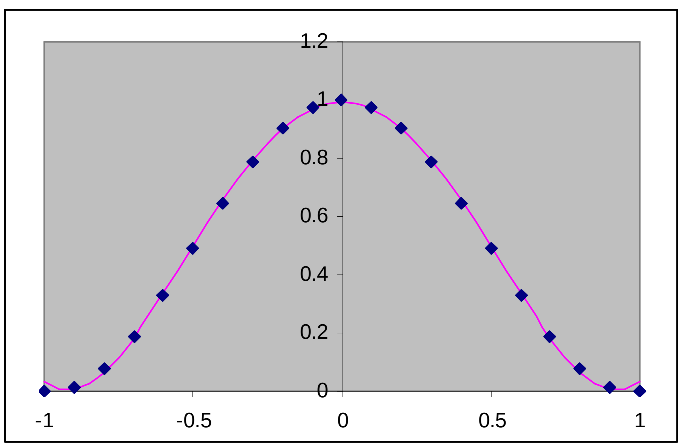
$[\ c_0, c_1, c_2\ , c_3, c_4, c_5\ ] =$

$$\left[\frac{5 \cdot \pi}{32},\ 0,\ -\frac{125 \cdot \pi}{512},\ 0,\ \frac{405 \cdot \pi}{4096},\ 0 \right]$$

The Legendre polynomial expansion will be:

$$g(z) \cong p \left[\frac{5}{32} - \frac{125}{512} \frac{3z^2 - 1}{2} + \frac{405}{4096} \frac{35z^4 - 30z^2 + 3}{8} \right] = \frac{15p\,(945z^4 - 1610z^2 + 689)}{32768}$$

The plot of this polynomial, in the interval [-1, 1] is shown in the following graph (pink line)



In the same graph we have also plotted the points (blue dots) of the exact solution $g(z) = (1 - z^2)^{5/2}$
Note the good general fitting. Consider that we have used only few moments just for exposition simplicity. If we perform the regression using more moments (for example 8 or 12), the matching between calculated line and exact curve will be progressively more tight.

Example 2. Assume to have the following 6 normalized moments

$$\hat{m} = \left[\frac{1}{3}, \frac{1}{45}, \frac{1}{15}, \frac{1}{105}, \frac{1}{35}, \frac{1}{189} \right]$$

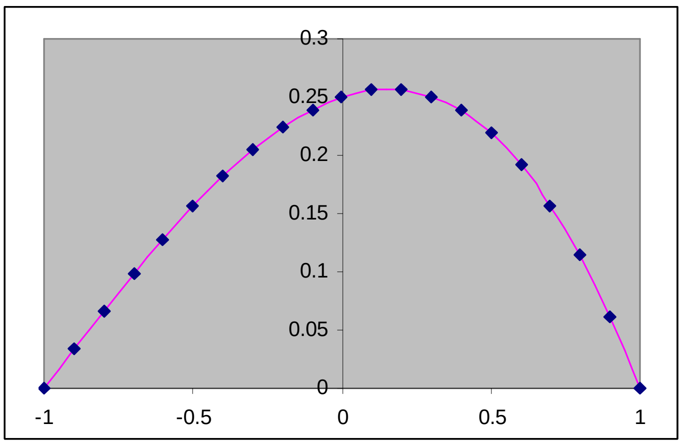
The solution of the moment regression is found solving the 6×6 linear system $\mathbf{P}c = \hat{m}$
The solution is

$$c = \left[\frac{1}{6}, \frac{1}{30}, -\frac{1}{6}, -\frac{1}{30}, 0, 0 \right]$$

The Legendre polynomial expansion will be:

$$g(z) \cong \frac{1}{6} + \frac{1}{30} z - \frac{1}{6} \frac{3z^2 - 1}{2} - \frac{1}{30} \frac{5z^3 - 3z}{2} \quad = \quad \frac{3 + z - 3z^2 - z^3}{12}$$

The plot of this polynomial, in the interval [-1, 1] is shown in the following graph (pink line)



In the same graph we have also plotted the points (blue dots) of the exact solution

$$g(z) = \frac{(z+3)(1-z^2)}{12}$$

The fitting, in that case is exact because the given moments are extracted from a polynomial of degree < 6.

Example 3. Sometime the fitting is not so perfect, of course. In these cases we should use more moments, if possible. Let's see.
Assume that we have a data set of 101 equispaced points [z_i, y_i] in the rage [-1, 1] extracted from the function

$$y = \frac{1}{3z^2 + z + 1}$$

Now we calculate the first 6 moments. The analytic solution is quite cumbersome but we can get a fast numerical solution using Excel and the Xnumbers function **integr**. Solving the system with these moments and, after rearranging, we have finally the following 5th degree polynomial coefficients $g(z) \cong a_0 + a_1 z + a_2 z^2 + a_3 z^3 + a_4 z^4 + a_5 z^5$

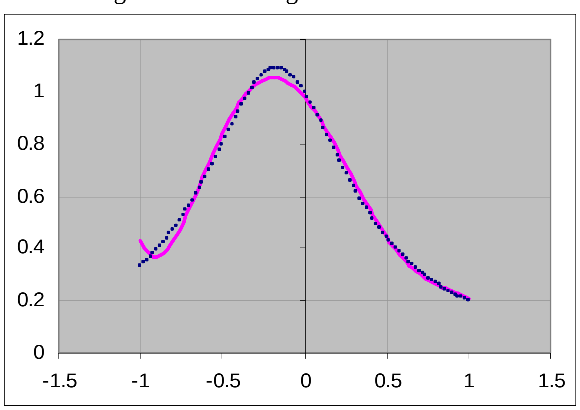
function	a	b	result
1/(3x^2+x+1)	-1	1	1.27444951577901
1/(3x^2+x+1)*x	-1	1	-0.12727064866884
1/(3x^2+x+1)*x^2	-1	1	0.28427371096328
1/(3x^2+x+1)*x^3	-1	1	-0.05233435409815
1/(3x^2+x+1)*x^4	-1	1	0.14490910326718
1/(3x^2+x+1)*x^5	-1	1	-0.03085824972301

	poly coeff.
a_0	0.97814722363
a_1	-0.75084087015
a_2	-1.56460053191
a_3	1.66692563166
a_4	0.90305521589
a_5	-1.02718112621

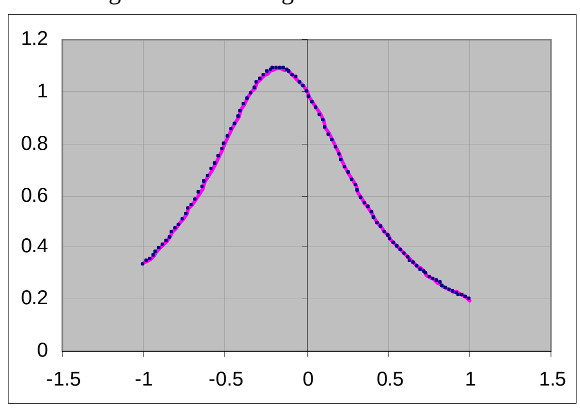
If we plot the 5th degree polynomial obtained by the regression with 6 moments, we note a somewhat poor approximation near the left limit z = -1 and around the peak z = -0.25. This means that, for this kind of functions, the degree of the approximating polynomial should be increased, and this requires, consequently, more moments.

In the following graph we shows two regressions obtained, respectively, with 5th and 9th degree polynomials. The convergence is evident.

Regression of 5th degree with 6 moments



Regression of 9th degree with 10 moments

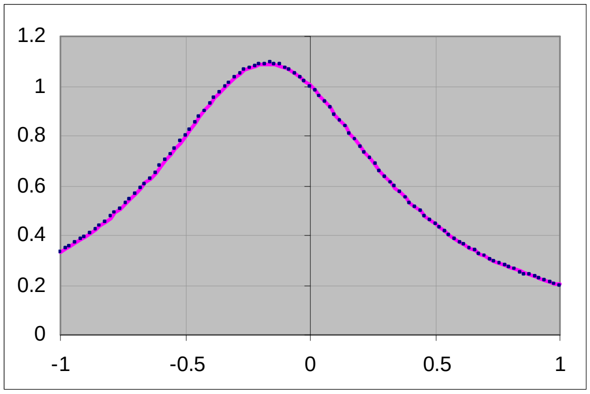


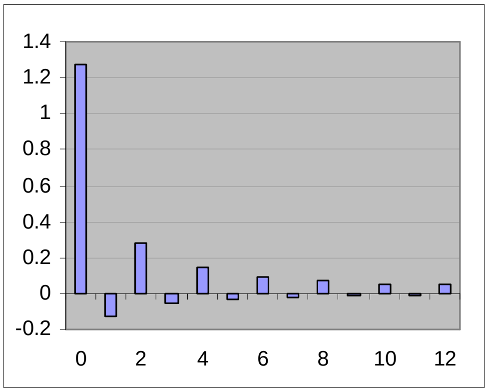
Test Functions

Several functions was used for this regression testing. For each one we have sampled 101 points and we have computed the first 13 moments with the Cavalieri-Simpson rule and then, we have computed the regression curve using a 12th degree polynomial. In a same graph we have compared the original data set (dotted line) with the regression (pink line). The results are shown in the following pages.

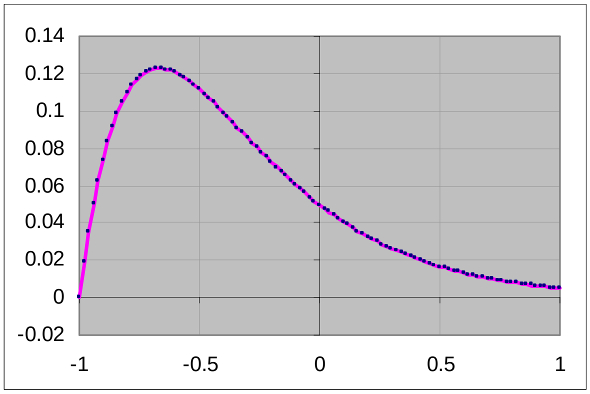
.

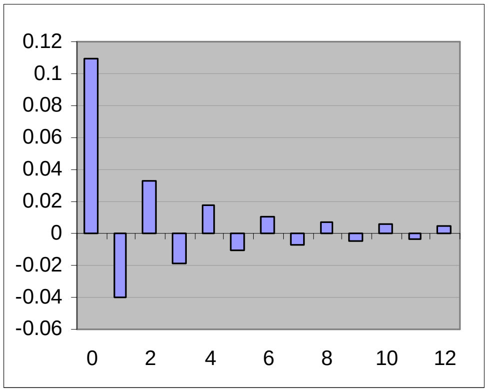
$y = 1/(3z^2 + z + 1)$



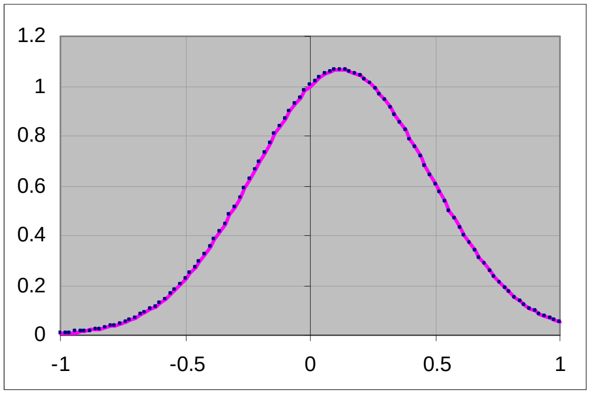


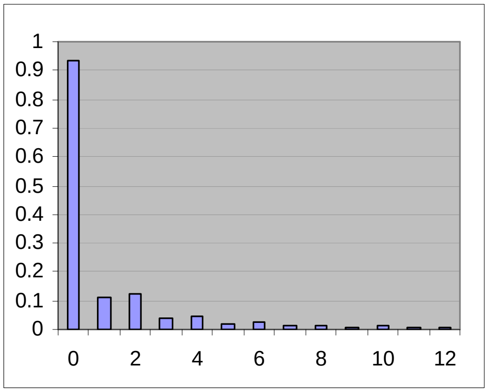
$y = (z+1)e^{-3(z+1)}$



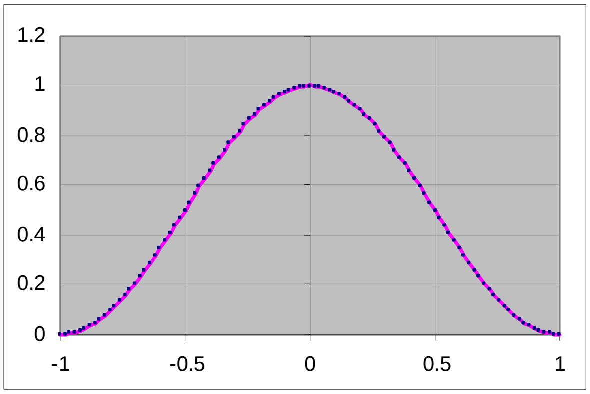


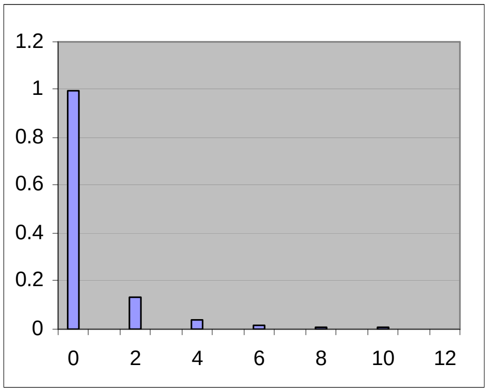
$y = e^{-4x^2 + x}$



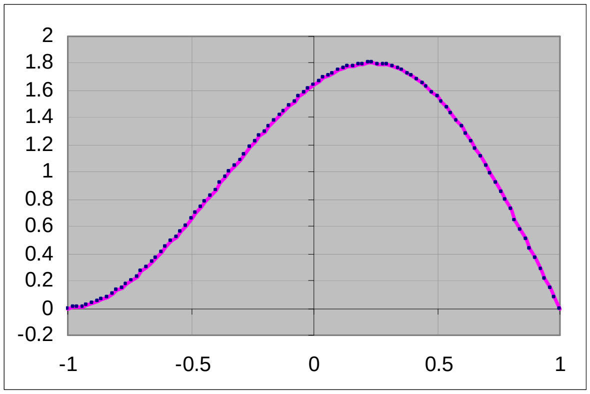


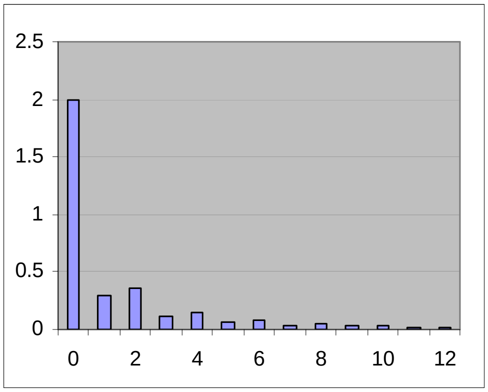
$y = \cos^2(p / 2 \cdot z)$



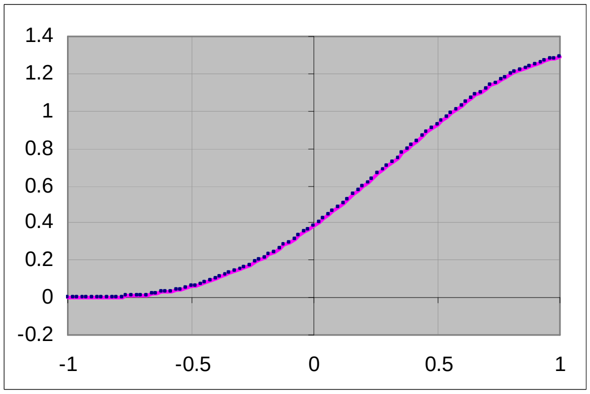


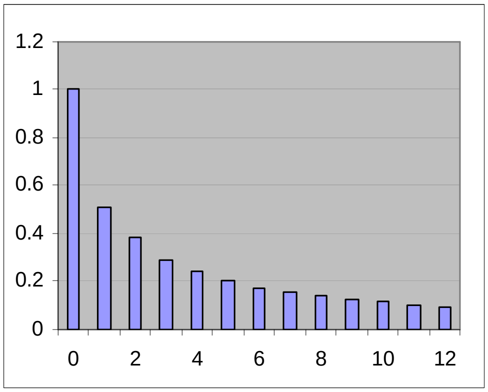
$y = 7(z-1)(z+1)^2(z^3+3z^2+3z-15)/64$



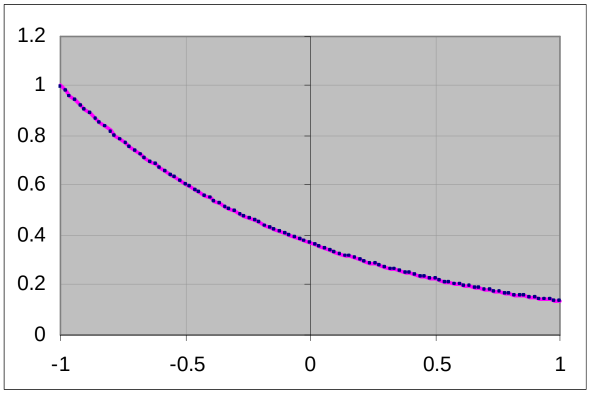


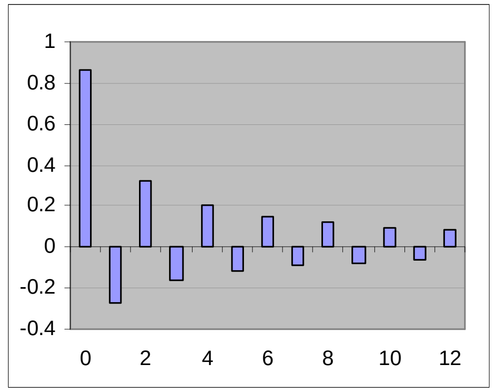
$y = \sin(z+1) - \sin(2z+2)/2$



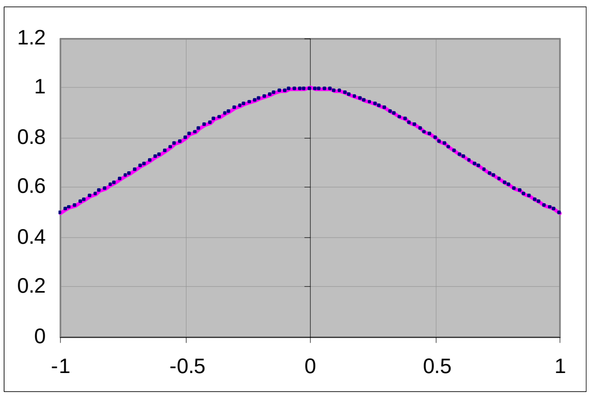


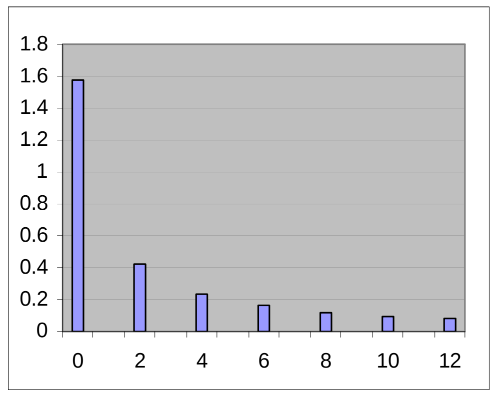
$y = e^{-(z+1)}$



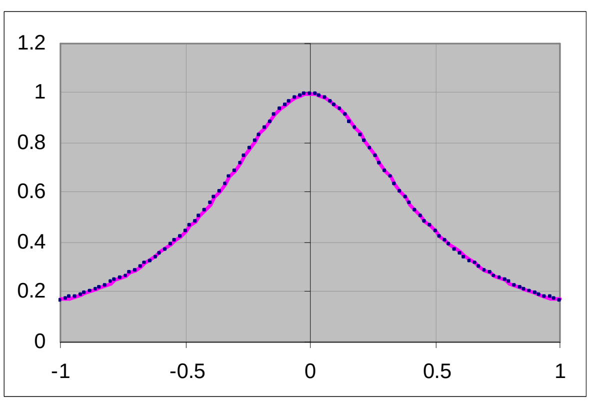


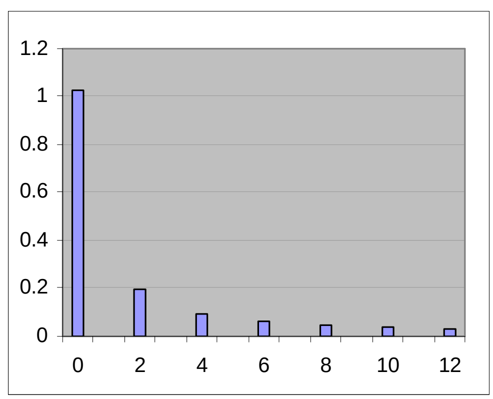
$y = 1/(z^2 + 1)$



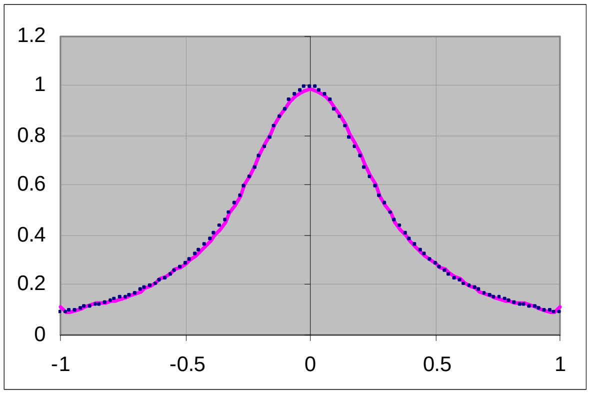


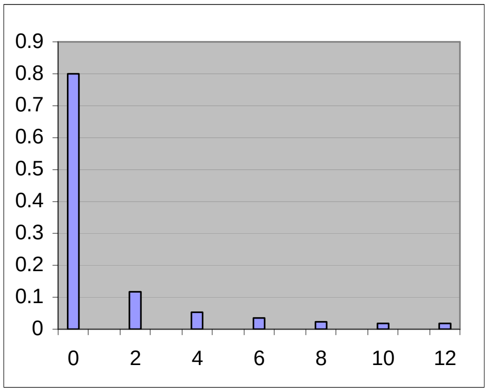
$y = 1/(5z^2 + 1)$



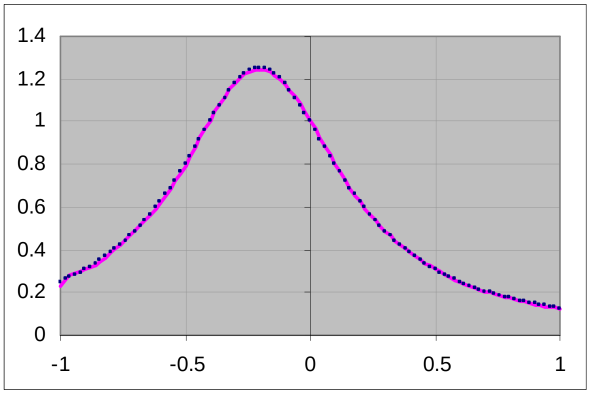


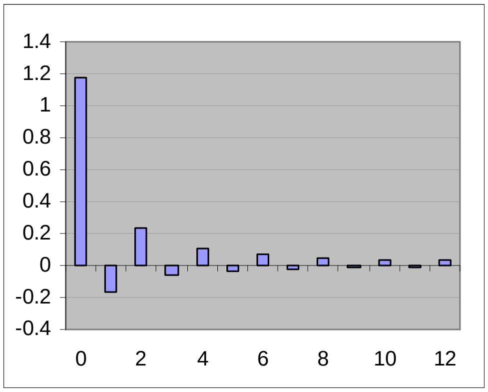
$y = 1/(10z^2 + 1)$



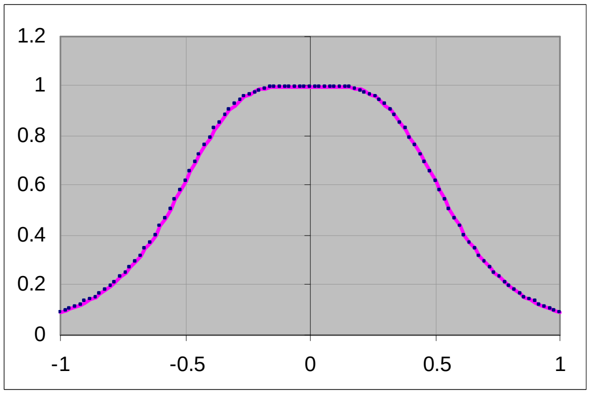


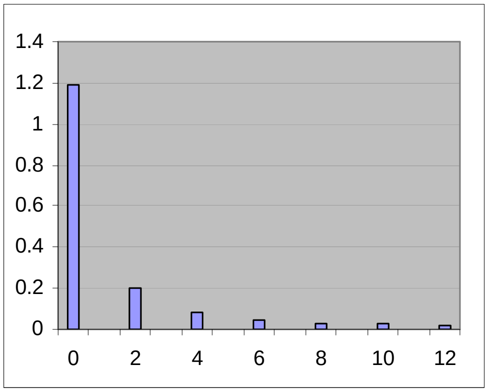
$y = 1/(5z^2 + 2z + 1)$



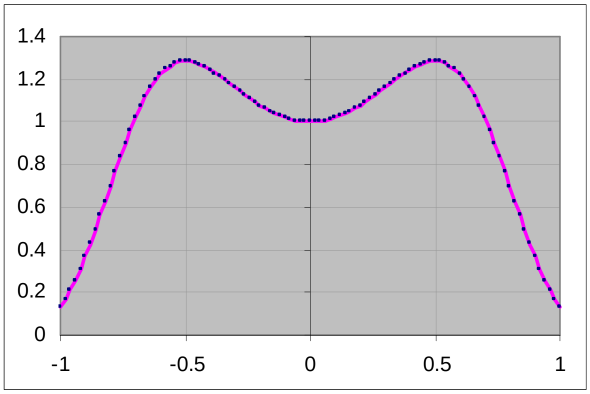


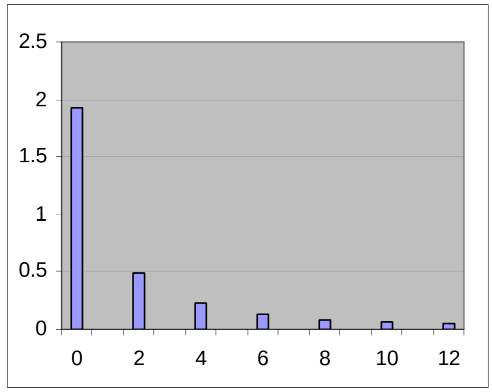
$y = 1/(10z^4 + 1)$



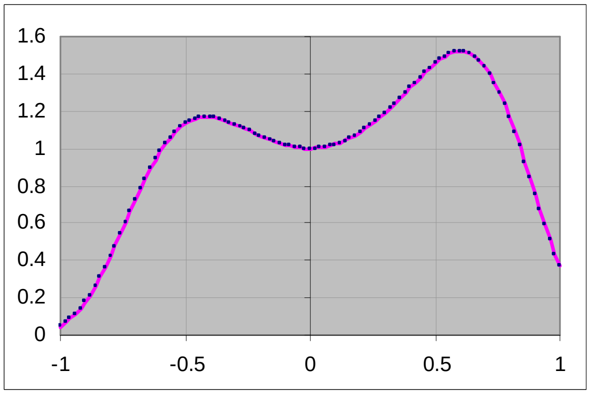


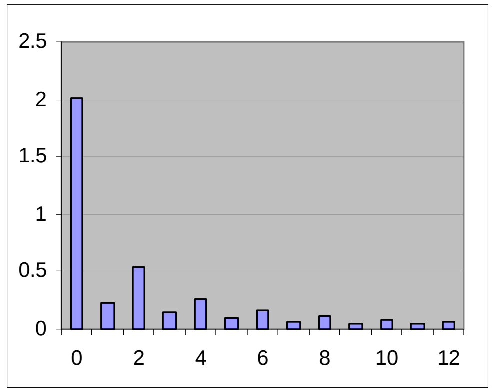
$y = e^{2z^2 - 4z^4}$



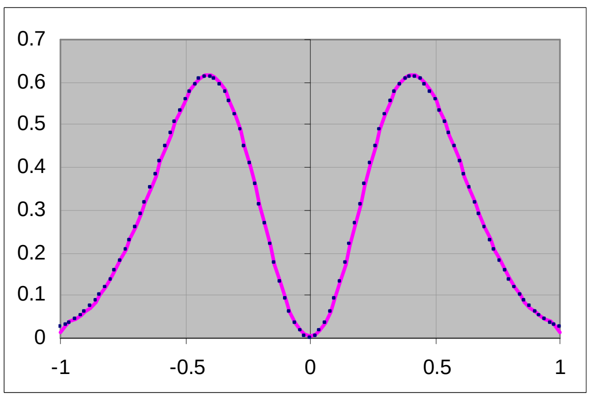


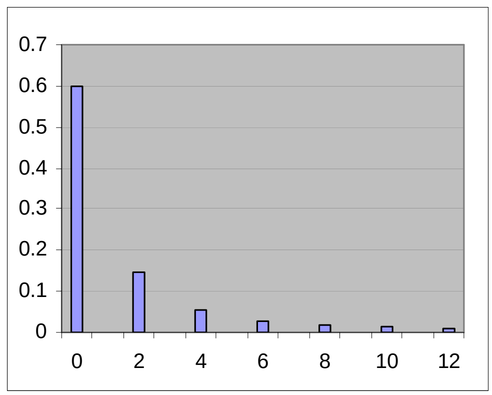
$y = e^{2z^2 + z^3 - 4z^4}$



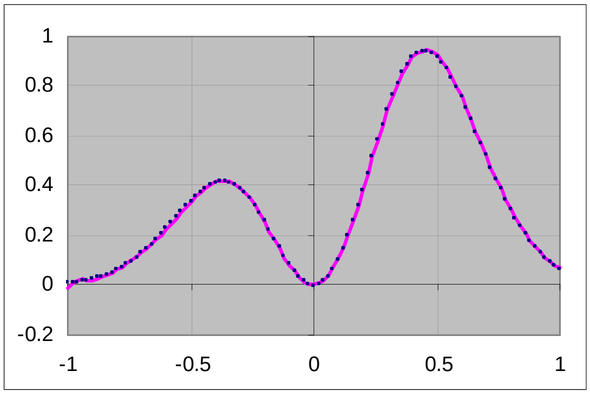


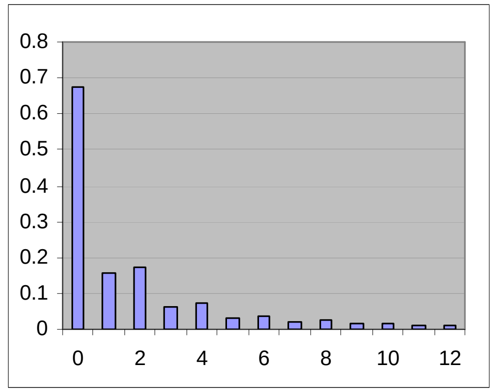
$y = 10z^2 e^{-6z^2}$



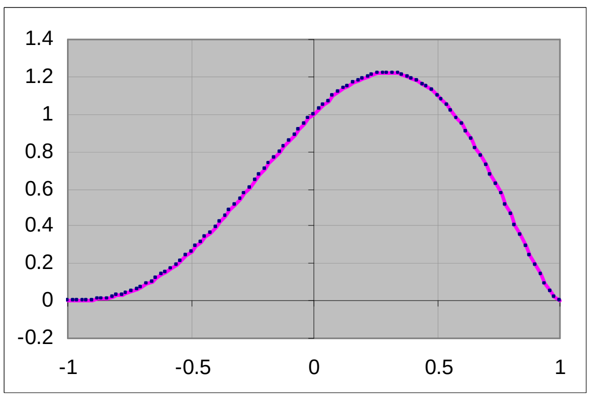


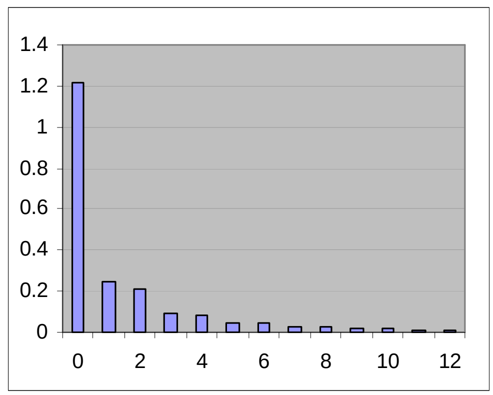
$y = 10z^2 e^{-6z^2 + z}$



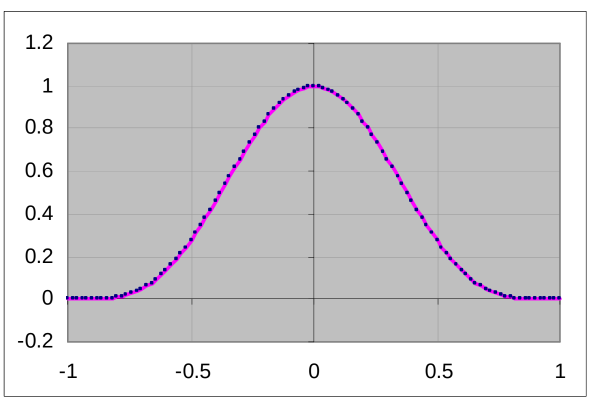


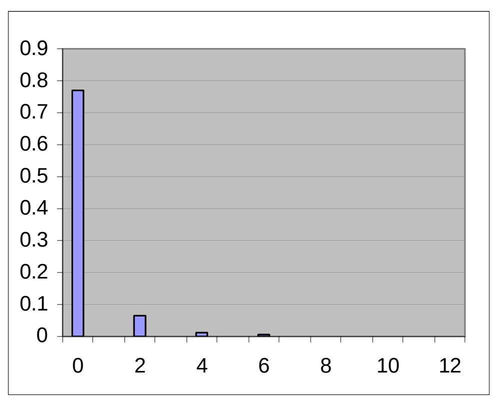
$y = (1 - z)^{3/2}(1 + z)^{14/5}$



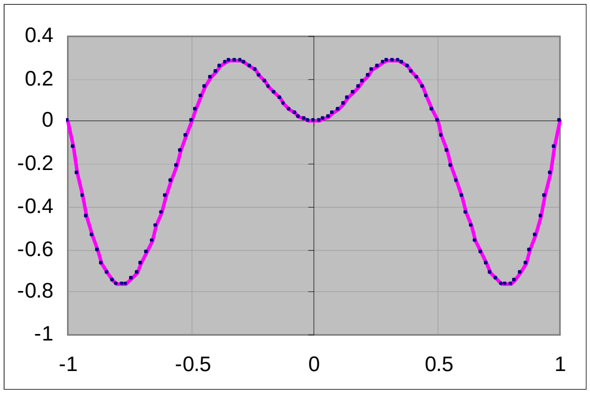


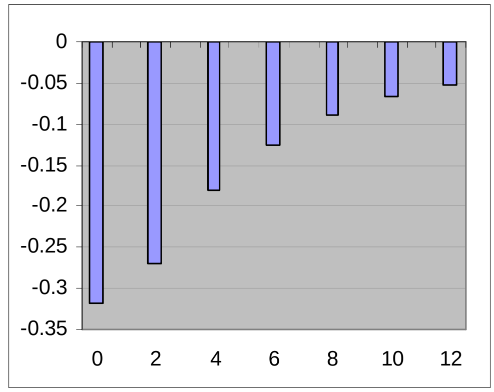
$y = (1 + z^2)^{9/2}$



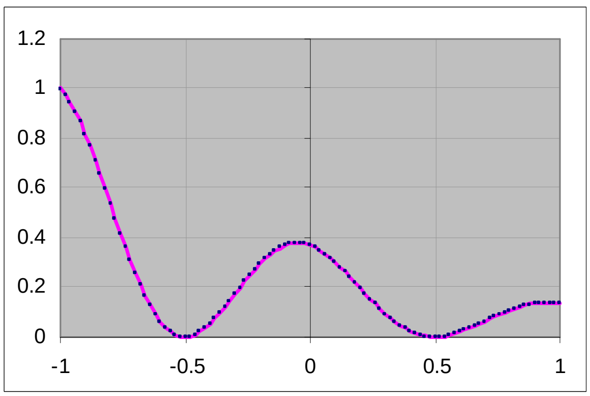


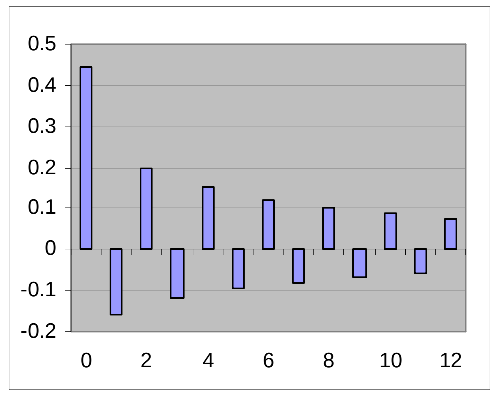
$y = z \cdot \sin(2p \cdot z)$



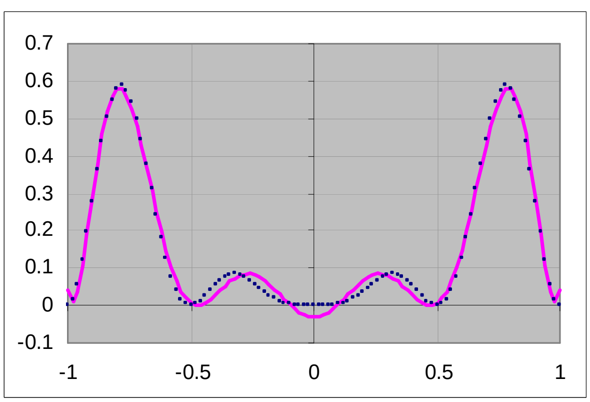


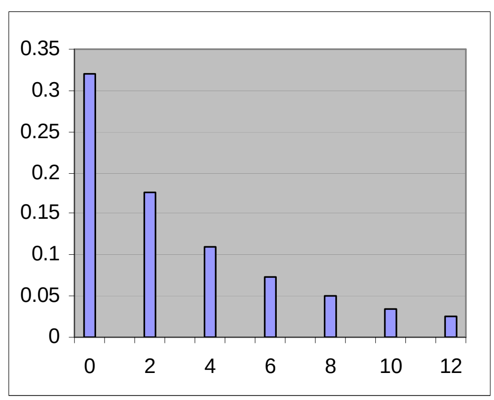
$y = e^{-(z+1)} \cos^2(p \cdot z)$



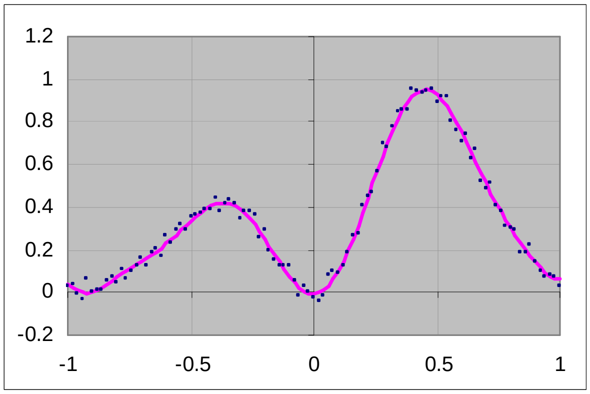


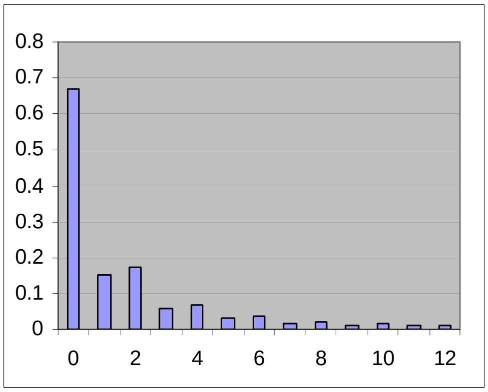
$y = [z \cdot \sin(2p \cdot z)]^2$



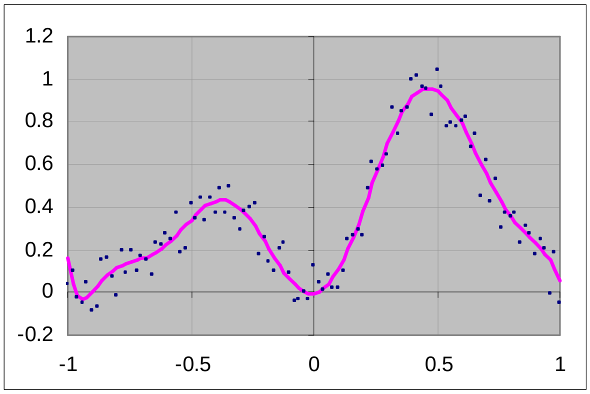


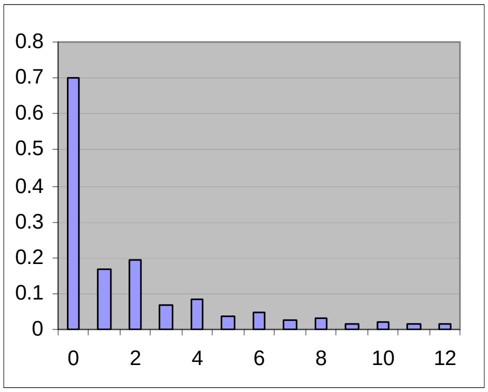
$$y = 10z^2 e^{-6z^2 + z} + 0.1 \cdot \text{rnd}$$



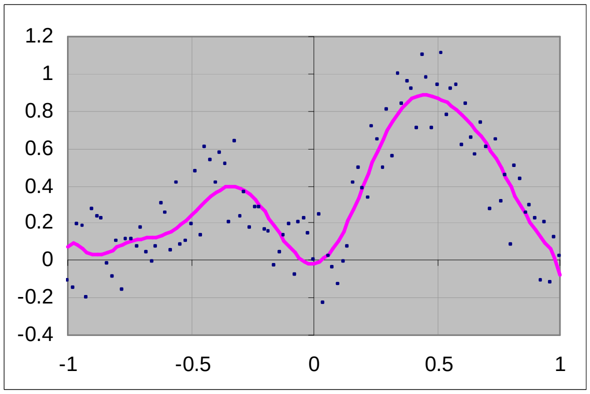


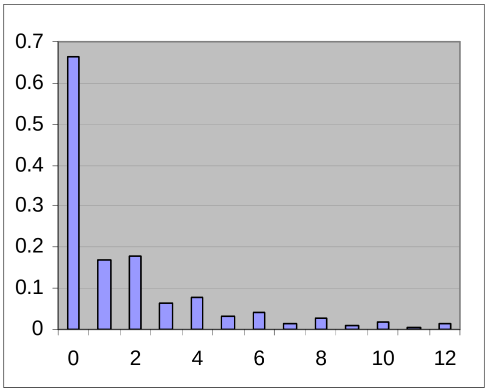
$$y = 10z^2 e^{-6z^2 + z} + 0.25 \cdot \text{rnd}$$





$$y = 10z^2 e^{-6z^2 + z} + 0.5 \cdot \text{rnd}$$





Moments Theorem.

Under suitable conditions it is possible to obtain a density function from its moments m_0, m_1, m_2....
From theory we know that the Laplace transform of a density f(x) is

$$L[f(t)] = \int_0^\infty f(t)e^{-st}dt$$

Substituting the exponential with its Taylor series we have

$$e^{-st} = 1 - st + \frac{(st)^2}{2!} - \frac{(st)^3}{3!} ... = \sum_{k=0}^{\infty}(-1)^k \frac{(st)^k}{k!}$$

$$L[f(t)] = \int_0^\infty f(t)\left(1 - st + \frac{(st)^2}{2!} - \frac{(st)^3}{3!} ...\right)dt =$$

If is valid the integration splitting term to term, we can transform into

$$= \int_0^\infty f(t)dt - s\int_0^\infty f(t)tdt + \frac{s^2}{2!}\int_0^\infty f(t)t^2dt - ... \ (-1)^n \frac{s^n}{n!}\int_0^\infty f(t)t^n dt + ...$$

Observing that $\int_0^\infty f(t)t^n dt = m_n$ is the moment of n^th^ order, we obtain the following series

$$L[f(t)] = \sum_{n=0}^{\infty}(-1)^n m_n \frac{s^n}{n!} \qquad (1)$$

We see that the Laplace transform of a density f(t) can be expressed as a series of its moments M_n .Observe that, in this case, the moments are similar to the derivatives coefficients of the Taylor expansion.
The formula (1) can be used for many scopes: demonstration, simplification, convolution, etc. But not for the scope of plotting f(t) because the variable s is in general complex $s = x + iy$.

There is also another simplified form of the moment theorem. Substituting the variable $s = iw$ we have

$$\Phi(w) = F[f(t)] = \sum_{n=0}^{\infty} m_n \frac{(jw)^n}{n!} = \left(1 - m_2\frac{w^2}{2!} + m_4\frac{w^4}{4!} ...\right) + j\left(m_1 w - m_3\frac{w^3}{3!} + m_5\frac{w^5}{5!} ...\right)$$

The function $\Phi(w)$ is sometime called "characteristic function" and it is, in general, complex depending from the real variable ω. But for obtaining the plot of f(t) we have to take the Fourier anti-transform of $\Phi(w)$ that, in general is not a trivial task

References

"*Handbook of Mathematical Functions*", M. Abramowitz and I. Stegun, Dover, 1964

"Numerical Recipes in FORTRAN 77- The Art of Scientific Computing - 1986-1992 by Cambridge University Press. Programs Copyright (C) 1986-1992 by Numerical Recipes Software

"*How to tabulate the Legendre's polynomials coefficients*" Leonardo Volpi, Foxes Team (http://digilander.libero.it/foxes/Documents)

Appendix- Routines for Moments Regression in VB6

All the computation and graphs appearing in this document have been made with the aid of the file "Moments regression test1.xls" [1], an Excel workbook containing the following macro-routines.

```
'-----------------------------------------------------------------------------------------
' MOMENTS REGRESSION ROUTINES
' 9.12.2006 by LV, Foxes Team
'
' These routines compute the array of data regression [xi, yi] from the
' moments of a regular function f(x) using the orthogonal polynomials method
' Moments of f(x) are definite as
' M(m)= Integral( f(x)*x^m, a<x<b )   with m=0,1,2...22
'
'-----------------------------------------------------------------------------------------
Option Explicit
'-----------------------------------------------------------------------------------------
'this routine performs the moments regression using the Legendre orthogonal polynomials
'9.12.2006 by LV, Foxes Team
'polynomial order < 23. Max moments = 23
'Integral( f(x)*x^m, a<x<b ) = M(m)  with m=0,1,2...22
'where f(x)= c0+c1*x+c2*x^2+...cm*x^m
'input Mom = vector of moments [M(0),M(1),M(2)...M(m)]
'input xmax, xmin = integration interval [xmax xmin]
'output Rcf = vector of regression polynomial coefficients [c0, c1, c2...cm]
'-----------------------------------------------------------------------------------------
Sub Moment_Regression(Mom, xmin, xmax, Rcf)
Dim i&, j&, k&, m&, n&, degree&, kn(), kd, cf#(), a#()
Dim x#(), t#, h#, t0#, t1#, Momn()
'initialize
degree = UBound(Mom) 'max legendre poly degree
m = degree + 1
ReDim a(1 To m, 1 To m)
ReDim cf(degree, degree)
ReDim Rcf(degree)
ReDim x(1 To m)
ReDim Momn(degree)
'build the legendre coefficient matrix
cf(0, 0) = 1
For n = 1 To degree
    Poly_Legendre_Builder n, kn, kd
For j = 0 To n
    'takes the i-th degree of legendre polynomial
    cf(n, j) = kn(j) / kd
Next j
Next n
'build the system solving matrix
For i = 1 To m
For j = 1 To i
    If (i + j) Mod 2 = 0 Then
        For k = 0 To j - 1
            If (k + i) Mod 2 <> 0 Then
                a(i, j) = a(i, j) + cf(j - 1, k) * 2 / (k + i)
            End If
        Next k
    End If
Next j
Next i
'moments normalization
If xmin <> -1 Or xmax <> 1 Then
    Call Moments_Normalize(xmin, xmax, Mom, Momn)
Else
    For i = 0 To degree: Momn(i) = Mom(i): Next i
End If
'solve the triangular system with forward substitution
```

[1] Disclaimer: This code is provided as a courtesy to others that may find it useful.
I have checked these functions reasonably well but we make no claims about their accuracy.
As with any computer software, check to make sure the answers you get make sense and are accurate.
That will ensure that you understand their use. There's no warranty that's free of bugs.
If you find any errors, please notify us via email. leovlp@libero.it
You are free to use it but with your own responsibility.
The author shall take no responsibility for any trouble associated with the use of this software.

```
For i = 1 To m
    For k = 1 To i - 1
        x(i) = x(i) - a(i, k) * x(k)
    Next k
    x(i) = (x(i) + Momn(i - 1)) / a(i, i)
Next i
'build the final regression polynomial
For i = 0 To degree
For k = 0 To degree
    Rcf(i) = Rcf(i) + x(k + 1) * cf(k, i)
Next k
Next i
End Sub

'Horner-Ruffini algorithm for polynomial computing
Private Function polyeval(coeff, x)
Dim i&, y#
For i = UBound(coeff) To 0 Step -1
    y = y * x + coeff(i)
Next i
polyeval = y
End Function

'-----------------------------------------------------------------------------------------
'this routine tabulates the moments regression using the Legendre polynomials
'9.12.2006 by LV, Foxes Team
' yi = p(xi) , xi=xmin+(xmax-xmin)/(pmax-1)*i  i=0,1, pmax-1
'input xmin, xmax = sampling interval [xmin, xmax]
'input pmax = max number of samples
'input coef = regression polynomial coefficients
'output Dxy = array of data sampled [xi, yi]
'-----------------------------------------------------------------------------------------
Sub Moment_Regression_Sampling(xmin, xmax, pmax, Rcoef, Dxy)
Dim h#, t0#, t1#, t#, i&
ReDim Dxy(1 To pmax, 1 To 2)
h = 2 / (pmax - 1)
t0 = (xmax + xmin) / 2
t1 = (xmax - xmin) / 2
For i = 1 To pmax
    t = (i - 1) * h - 1
    Dxy(i, 1) = t1 * t + t0
    Dxy(i, 2) = polyeval(Rcoef, t)
Next i
End Sub

'-----------------------------------------------------------------------------------------
'this routine extract the moments from a given dataset xy
'using the Cavalieri-Simpson quadrature formula
'9.12.2006 by LV, Foxes Team
'Integral( f(x)*x^m, a<x<b ) = M(m)  with m=0,1,2...22
'input Dataxy = array of data points [xi, yi]
'input m = number of moment to extract
'output Mom = vector of moments [M(0),M(1),M(2)...M(m)]
'-----------------------------------------------------------------------------------------
Sub Moments_Extractor(Dataxy, m, Mom)
Dim i&, j&, pmax&, xmin, xmax, row0&, h#, f#(), p#, s#, tiny#
tiny = 2 * 10 ^ -15
pmax = UBound(Dataxy)
xmin = Dataxy(1, 1)
xmax = Dataxy(pmax, 1)
If pmax Mod 2 = 0 Then
'    MsgBox "points must be odd", vbInformation
    Exit Sub
End If
'integration step
h = (xmax - xmin) / (pmax - 1)
ReDim f(1 To pmax)
For i = 1 To pmax: f(i) = Dataxy(i, 2): Next i
ReDim Mom(m - 1)
Do
    'compute the j-th moments
    s = 0: p = 0
    For i = 2 To pmax - 1 Step 2: s = s + f(i): Next i
    For i = 3 To pmax - 1 Step 2: p = p + f(i): Next i
    Mom(j) = h / 3 * (f(1) + 4 * s + 2 * p + f(pmax))
    If Abs(Mom(j)) < tiny Then Mom(j) = 0
    j = j + 1
    If j = m Then Exit Do
```

```
    For i = 1 To pmax: f(i) = f(i) * Dataxy(i, 1): Next i
Loop
End Sub

'---------------------------------------------------------------------------
' This routine converts the moments from a given interval [a, b] to [-1, 1]
' 9.12.2006 by LV, Foxes Team
' Mom =  raw moments in the interval [a, b]
' MomN = normalized moments in the interval [-1, 1] (normalized)
'---------------------------------------------------------------------------
Sub Moments_Normalize(a, b, Mom, Momn)
Dim i&, j&, m&, r#, c#, tiny#
tiny = 2 * 10 ^ -15
m = UBound(Mom)
ReDim Momn(m)
r = (a + b) / (b - a)
For i = 0 To m
    Momn(i) = Mom(i) * (2 / (b - a)) ^ (i + 1)
    For j = 1 To i
        c = Application.WorksheetFunction.Combin(i, j)
        Momn(i) = Momn(i) - c * r ^ j * Momn(i - j)
    Next j
Next i
'mopup
For i = 0 To m
    If Abs(Momn(i)) < tiny * (1 + Abs(Mom(i))) Then Momn(i) = 0
Next i
End Sub

'---------------routines extracted from XNUMBERS 5.2 ------------------------------

Private Sub Poly_Legendre_Builder(degree As Long, coef(), kd)
'compute the coefficients of the Legendre polynomial
' degree < 24  in standard 32-bit precision
Dim Nmax&, n&, i&, L() As Double, Ld() As Double
Dim a, b, c
Nmax = degree
ReDim L(Nmax, 2), Ld(2)
L(0, 0) = 1
L(1, 1) = 1
Ld(0) = 1
Ld(1) = 1
n = 2
Do Until n > Nmax
    'iterate
    a = Ld(0) * (2 * n - 1)
    b = (n - 1) * Ld(1)
    L(0, 2) = -b * L(0, 0)
    For i = 1 To n
        L(i, 2) = a * L(i - 1, 1) - b * L(i, 0)
    Next i
    'compute the GCD
    c = n * Ld(1) * Ld(0)
    Ld(2) = c
    For i = 1 To n
        c = mcd_2(c, L(i, 2))
    Next i
    'reduce terms
    Ld(2) = Ld(2) / c
    For i = 0 To n
        L(i, 2) = L(i, 2) / c
    Next i
    'shift
    For i = 0 To n
        L(i, 0) = L(i, 1)
        L(i, 1) = L(i, 2)
        L(i, 2) = 0
    Next i
    Ld(0) = Ld(1)
    Ld(1) = Ld(2)
    n = n + 1
Loop

ReDim coef(Nmax)
For i = 0 To Nmax
    coef(i) = L(i, 1)
Next i
kd = Ld(1)
```

```
End Sub

Private Function mcd_2(a, b)
'Find the MCD between two integer numbers
'by the Euclid method
Dim y#, x#, r#
y = Abs(a): x = Abs(b)
Do Until x = 0
    r = y - x * Int(y / x)
    y = x: x = r
Loop
mcd_2 = y
End Function
```