WEEF 2013 CARTAGENA COLOMBIA
World Engineering Education Forum
Septiembre 24 a 27
CENTRO DE CONVENCIONES CARTAGENA DE INDIAS
Innovation in research and engineering education: key factors for global competitiveness
Innovación en investigación y educación en ingeniería: factores claves para la competitividad global

'ALIVE' ALGORITHMS FOR NUMERICAL ANALYSIS IN ENGINEERING EDUCATION

Diego Martín Oviedo Salcedo, Claudia Patricia Retamoso Llamas

Universidad Pontificia Bolivariana
Bucaramanga, Colombia

Óscar Javier García Cabrejo

University of Illinois at Urbana
Champaign, IL, USA

Édgar Eduardo Muñoz Díaz

Pontificia Universidad Javeriana
Bogotá, Colombia

Abstract

It has been always the main goal of the numerical analysis algorithms to get a solution for the problem at hand. A solution is nothing but a numerical value: a scalar, a vector, or a matrix. Thus, a solution is all what an engineer need to get his/her job done. However, for an engineering student this is not enough. He or she has to experiment what are each algorithm drawbacks and advantages. Then is necessary to see the insights of each one. Usually the algorithms are recursive formulas. A convenient way to follow closely what an algorithm does is by plotting intermediate steps and their partial results. This explain why while teaching numerical analysis, instructors usually are interested on the visual evolution to show how and why an algorithm might converge or not. However, plotting is a luxury when the instructor has to focus in teaching how to code properly the algorithm. Here a dilemma, to teach the algorithm or to visualize how it works? Well, currently, we are doing the best of both worlds with a single tool.

Applets are popular web-based tools to disseminate information allowing the user to interact by feeding parameters and tweaking some others; however to create them a great deal of knowledge in Java™ o similar programming is required. Coding is not always the main focus of many engineering curricula with the exception of computer sciences. In addition, in many instances these applets are in the form of black boxes. Users feed input, and get results but are not allowed to see the code. Wolfram Mathematica® 9 is the ideal

tool to create applets-like models. Coding within this application is more intuitive and the coder has total control over the every parameter.

Keywords: interactive numerical analysis

Resumen

Los algoritmos aplicados al análisis numérico en ingeniería, tienen por finalidad principal resolver un problema específico. En este caso, nos referimos a una solución como el resultado numérico expresado como un escalar, vector, o matriz. Generalmente, estos resultados numéricos son suficientes en el ejercicio profesional. Sin embargo, para un estudiante de ingeniería esto no es suficiente. El (la) estudiante debe experimentar cuáles son las ventajas y desventajas de cada algoritmo. Así, para el (la) estudiante necesario ver como funciona cada algoritmo en su interior. Es usual que los algoritmos sean fórmulas recursivas. Una manera de hacer seguimiento detallado de un algoritmo consiste en graficar los resultados intermedios y visualizar los resultados parciales. Esto explica por que razón se hace imperativo en el trabajo de instrucción en análisis numérico el uso de gráficas, así se puede observar si el algoritmo converge o no. Pero, esta necesidad de visualización de resultados riñe con el instructor debe esforzarse por codificar adecuadamente un algoritmo. Surge un dilema: se ensena a presentar los resultados parciales o se enseña a codificar? Con la herramienta adecuada se pueden tener las dos cosas simultáneamente.

Existen en la actualidad un sinnúmero de "applets" en la con el propósito de diseminar información los cuales le permiten a un usuario interactuar pro medio de la entrada de ciertos parámetros, o por medio de la modulación y manipulación de otros. Sin embargo, la creación de dichos modelos interactivos requiere un adecuado (de intermedio o avanzado) nivel de conocimiento de lenguajes tales como Java™. Usualmente, la codificación no es el centro de los currículos en Ingeniería, exceptuando la Ingeniería de Computación. Además, muchos de estos "applets" están en forma de cajas negras en donde el usuario introduce parámetros, pero no sabe que ocurre adentro. Wolfram Mathematica® 9 es una herramienta ideal para crear modelos similares a los "applets", con la ventaja que su codificación resulta un poco más intuitiva, donde el creador del modelo tiene control total sobre cada parámetro.

***Palabras clave**: análisis numérico interactivo*

1. Introduction

Numerical analysis for engineers is one of the key courses in most of the undergraduate engineering curricula. It is an important transition course allowing the future engineers realize at least these three issues: 1. Math foundations from previous courses are truly useful to solve "real life" problems; thus math finally makes sense for them, 2. It helps to understand how engineers should breakdown mathematical concepts in pieces in order for these to be read and "understood" by computers; and 3. Computers are really necessary to scale methods up, if complexity is incorporated in analysis.

All the authors in addition to have been involved in numerical analysis for civil engineering students teaching, we recall the type of abstractions required to make those algorithms working. On one hand the mathematical notion and, on the other the coding process to feed the machine to get results. The current work summarizes how the teaching experience of numerical analysis has evolved and adapted to the current

technologies for over 10 years at Civil Engineering Department in two Colombian Colleges: the Universidad Pontificia Bolivariana in Bucaramanga and the Pontificia Universidad Javeriana in Bogota. Several examples are illustrated in this document using different applications in its own highlights: *MS-Excel®*, a combination of *MATLAB®* and *BEAMER®* to display dynamics, and *Mathematica®* 9 to solve, visualize and interact with any algorithm in numerical analysis.

2. Methods and materials

This work documents two of the most useful resources in numerical analysis: Newton-Raphson method and Taylor series expansion. The first allows demonstrating how we can gain flexibility by using different software. With MS-Excel® is a widely used tool; its power stems from the friendly environment where the user computes what he or she wants by using functions and arithmetic operations combining cells. Pointing and clicking may execute some of the work; but MS-Excel® may be "programmed" by using its macros or by connecting it with Visual Basic for Applications (VBA). In this work, just the basic functionality has been used. Then the combine use of MATLAB® and BEAMER on the Newton-Raphson method is presented. While the workforce in solving and processing the steps on such an algorithm is achieved by using MATLAB® the outcome is displayed by using the LaTeX-based class BEAMER. This is a tool to animate by using frames where the user may be superimpose different layers giving the idea of an animation with in a PDF document. A great deal of coding, both in MATLAB® and LaTeX, has to be done to get interesting results. Finally, with Mathematica® the Newton-Raphson method is presented as a recursive formula by using a few lines of code; then, a more elaborated code allows to get real interactivity to explore this algorithm inside-out.

The second concept, the results of the Taylor series expansion are presented as the MATLAB® and BEAMER combination to show dynamics. Then, a dynamic model based on Mathematica allows showing full interactivity. In addition, the classical solution of the Taylor series is presented as the result of taking full advantage of symbolic computation provided by Mathematica®.

3. Newton-Raphson (N-R) algorithm

This is one of the most widely used approaches to get roots of any function. It requires an initial value, the function and its derivative evaluation on the initial value to get an approximated zero. This becomes the new value to be fed on the recursive formula until the method achieves the maximum iteration number or the desired tolerance.

3.1. N-R algorithm using MS-Excel®

Figure 1 shows the basic notions of the N-R in the table within this MS-Excel caption. Here the N-R is applied to compute an optimal value in the u-shape beam design. A second order polynomial is solved; meaning that its derivative is a linear function.

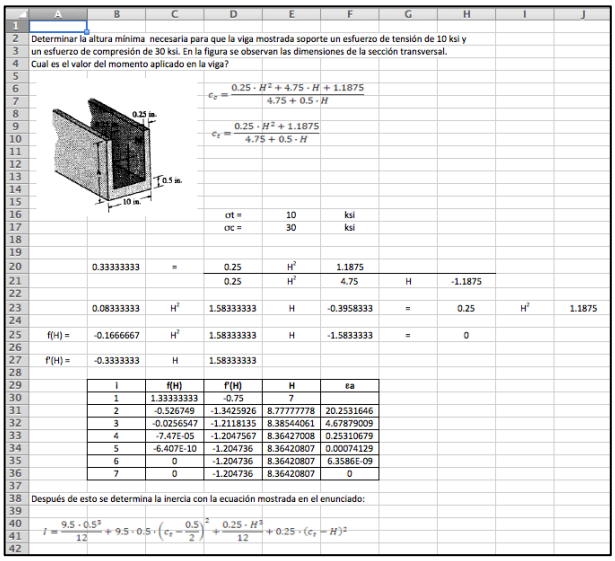


Figure 1. N-R method in MS-Excel, an application to a Civil Engr. problem

3.2. MATLAB® and BEAMER on N-R

Figure 2 displays a sequence, from left to right then bottom in exact order, of N-R steps. Notice the control buttons below each panel. Basically this are to move forward or backward.

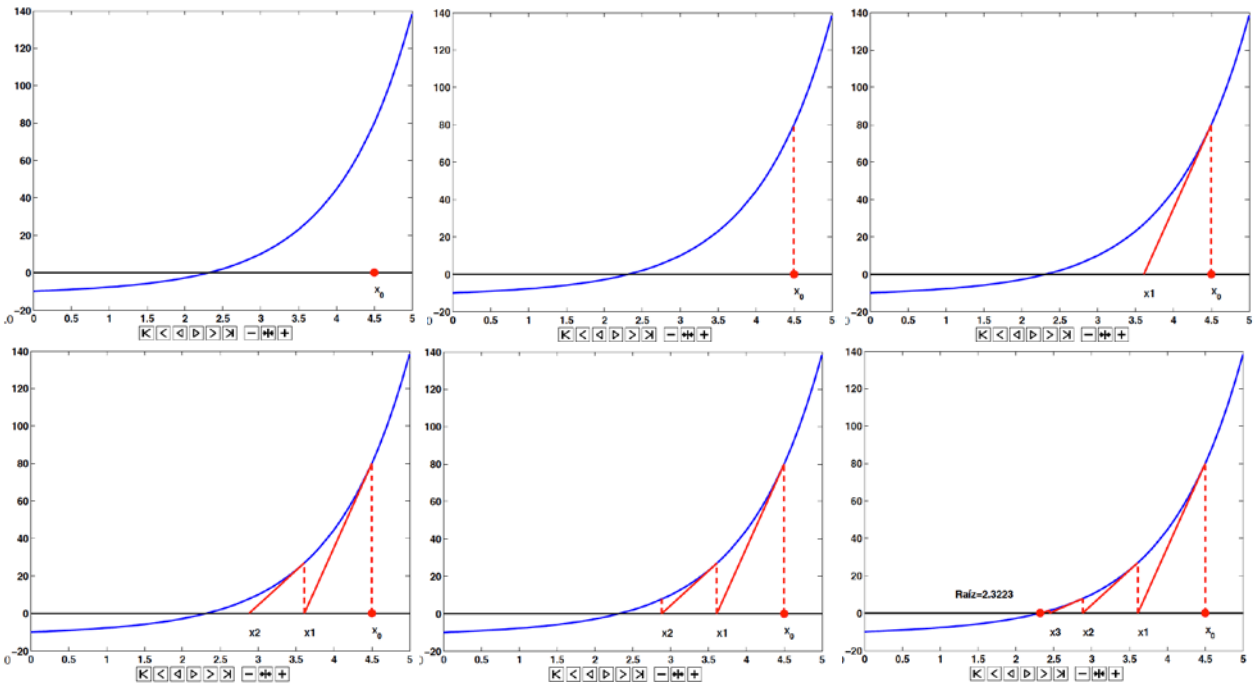


Figure 2. Rendering of the N-R algorithm with BEAMER solve with MATLAB®

3.3. Interactive Newton-Raphson model with Mathematica®

Figure 3 depicts the code that creates the interactive model on Figures 4-7. If the comments were removed from the code, it would be shorter.

```
polynomial[x_] := x^3 - 2 x + 2        (* a particular polynomial for N-R method *)

sine[x_] := Sin[x]                     (* the sine function *)

newton[f_, x0_, a_, b_, n_] := Module[{list = NestList[# - f[#]/f'[#] &, x0, n]},  (* here the core of the N-R method *)

  Column[{TraditionalForm@Text@Style[Row[{HoldForm[{x_k}_{k=0}^n], " = ", list}], 14],
    Plot[f[x], {x, a, b}, PlotRange → {-2.5, 3.5}, AxesLabel → {Style[x, 16], Style[y, 16]}, PlotStyle → Thickness[.006],
     Epilog → {{Red, Thickness[.002], Arrowheads[.03], Arrow[Most[Flatten[Map[{{#, 0}, {#, f[#]}} &, list], 1]]]},
       {PointSize[0.015], Point[{x0, 0}]}}, ImageSize → {600, 325}]}, Center]]
 (* this lines displays the result and the plot that tracks the root finding process *)
```

```
Manipulate[newton[d, N[x0], -2 π, 2 π, n], (* here we use the newton function we have created above *)
 {{d, sine, "function"}, {polynomial → TraditionalForm[x^3 - 2 x + 2 ], sine → TraditionalForm[Sine[x]]}, ControlType → PopupMenu},
 (* this controls the popup menu to select the function we want to play with *)
 {{n, 1, "n"}, {0, 1, 2, 3, 4, 5, 6}, ControlType → SetterBar}, (* this controls the number of iterations *)
 {{x0, .6, Subscript["x", "0"]}, 0.01, 6.11, Appearance → "Labeled" }, SaveDefinitions → True]
(* this controls the initial point *)
```

Figure 3. This is the caption of the entire commented code to create an interactive model of N-R

Figure 4-6 shows the main interactive features in such a model: dropdown menu to select the function, buttons to change the number of iterations and a slider to choose the N-R initial value. These results are replicated with the use of pure programming sequences in Figures 7 and 8. Notice how handy the "NestList[ ]" is. It is in charge of the recursive calculations.

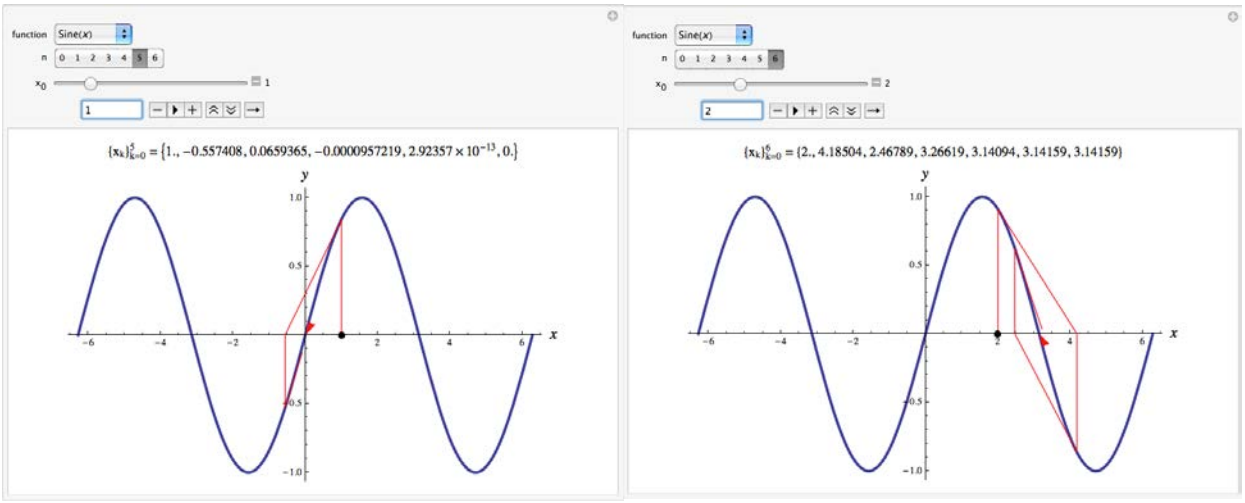


Figure 4. Influence of initial point in root finding by means of N-R

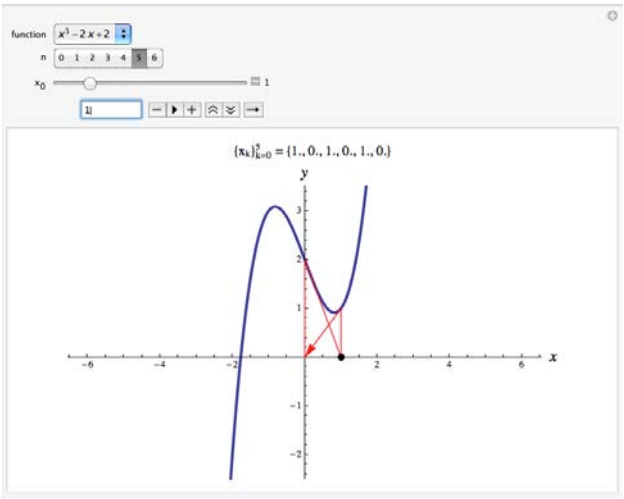


Figure 5. An infinity loop: any root is found

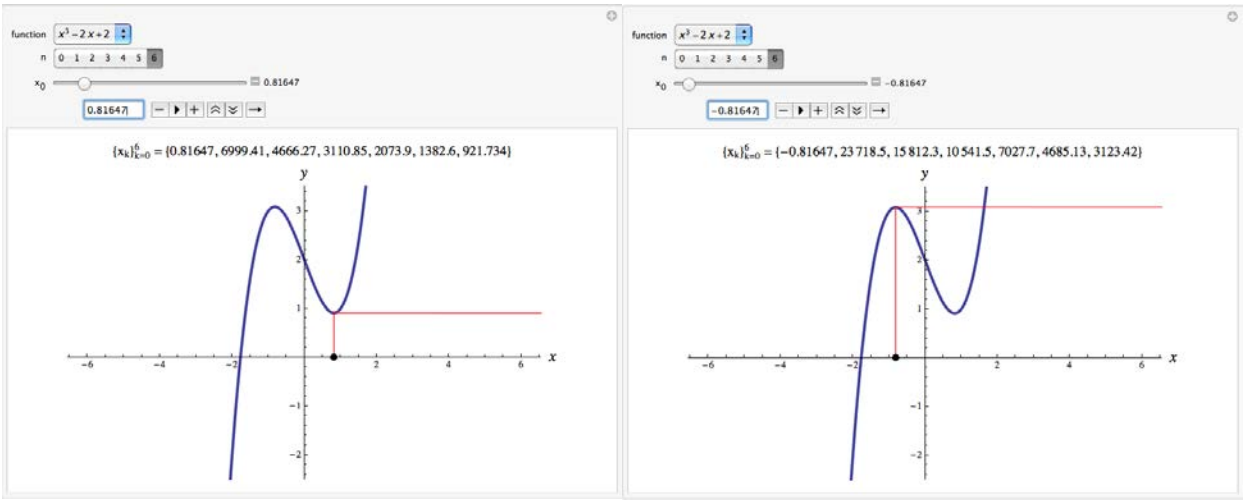


Figure 6. When the derivative is equal to zero: non-convergence in N-R

```
In[1]:= f[x_] := Sin[x]

In[3]:= NestList[# - f[#]/f'[#] &, 1., 6]

Out[3]= {1., -0.557408, 0.0659365, -0.0000957219, 2.92357×10^-13, 0., 0.}

In[4]:= NestList[# - f[#]/f'[#] &, 2., 6]

Out[4]= {2., 4.18504, 2.46789, 3.26619, 3.14094, 3.14159, 3.14159}
```

Figure 7. N-R code in Mathematica® to show the influence of the initial point

```
In[5]:= g[y_] := y^3 - 2 y + 2

In[6]:= NestList[# - g[#]/g'[#] &, 1., 6]

Out[6]= {1., 0., 1., 0., 1., 0., 1.}

In[7]:= NestList[# - g[#]/g'[#] &, Sqrt[2/3], 6]

        Power::infy : Infinite expression 1/0 encountered. >>

        Infinity::indet : Indeterminate expression 2 + ComplexInfinity + ComplexInfinity encountered. >>

Out[7]= {Sqrt[2/3], ComplexInfinity, Indeterminate, Indeterminate, Indeterminate, Indeterminate, Indeterminate}

        NestList[# - g[#]/g'[#] &, -0.816497, 6]

Out[11]= {0.816497, -443 899., -295 933., -197 288., -131 526., -87 683.7, -58 455.8}
```

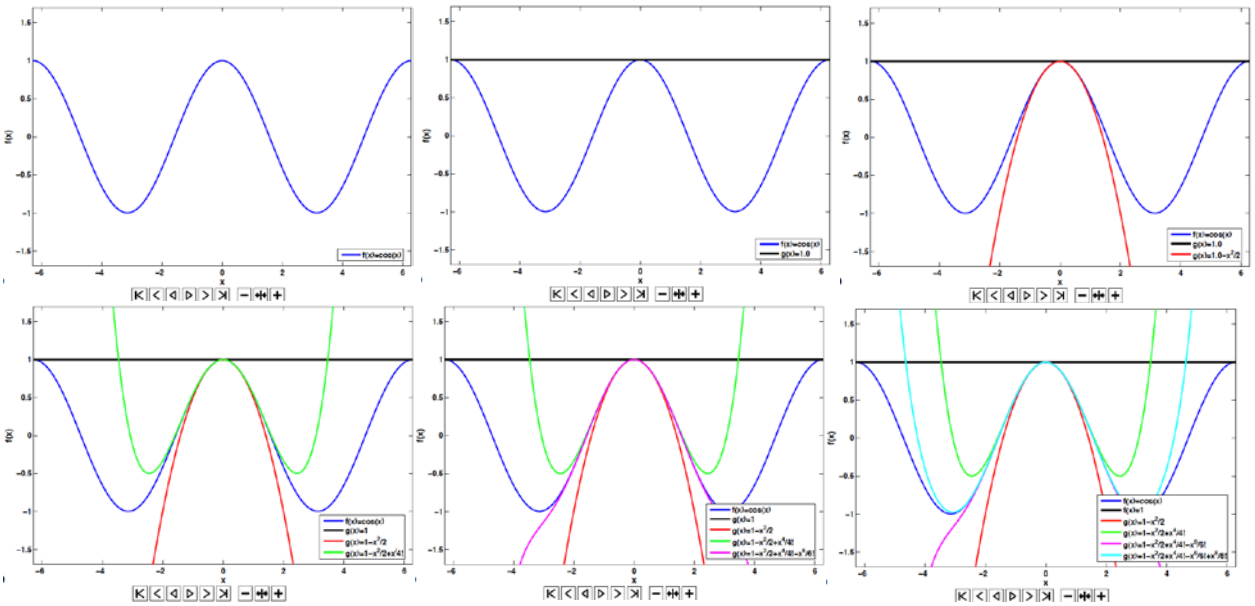
Figure 8. N-R code to show a never-ending loop, indetermination and a slow convergence case

4. Taylor series expansion

This is the main topic when approximations are explained in the light of mathematical concepts.

4.1. Animated Taylor expansion on the cosine function around a = 0

The six captions below, from left to right then bottom and exact sequence, show up to the fifth expansion on the cosine function



d. e. f.

Figure 9. Rendering of the Taylor expansion in BEAMER solved by MATLAB®

4.2. Interactive model on Taylor series

The captions below in Figure 10 are produced with the demonstration by (Brown et al., 2013). Notice the interactive model is plenty of controls allowing the user to explore and learn.

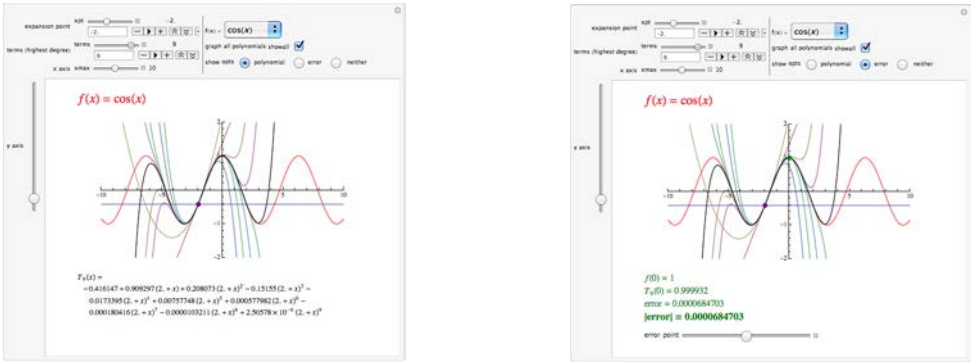


Figure 10. Rendering of the Taylor expansion as an interactive model using Mathematica®

4.3. Symbolic solution and seeking for help

The documentation and the help system in Mathematica® is really informative. It is loaded with "alive' examples allowing the user to take advantage of the software. At the bottom of Figure 11, the canonical form of the Taylor series can be seen

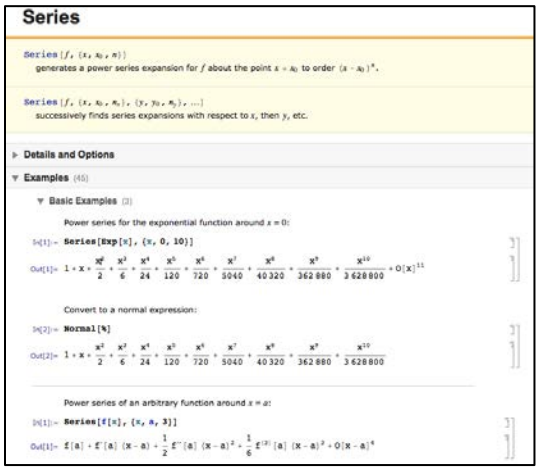


Figure 11. Help page of the "Series[ ]" built-in Mathematica® function

5. Discussion

There is no a unique way to teach numerical analysis regarding the visual support provided to the fundamentals. A variety of software applications may be used to convey the main ideas in numerical analysis. Dynamic and interactive visuals combined are powerful tools in debriefing the imposed burdens of the mathematical abstractions and the coding translation from the natural language to the computational one.

Technologies are continuously evolving. Every time they allow doing things "more easily". However, this is not the only benefit. At the same time, they allow to address more complex problems. An application is not only valuable due to its computational capabilities, but also to its ability to convey concepts in a neat manner. Most of the examples presented in this document have been designed to show how important the visual rendering is.

6. References

Books

- García-Cabrejo, O. J., Oviedo-Salcedo, D. M., and Muñoz-Díaz E. E., (2011) Apuntes de Clase de Análisis Numérico para Ingenieros Civiles, published by authors. ISBN: in process

Electronic sources

- Wolfram Demonstrations Project, checked on May 7, 2013 at demonstrations.wolfram.com

Software

- MS-Excel, Microsoft Corporation
- MATLAB®, The Mathworks inc.
- Beamer class of LaTeX
- Mathematica®, Wolfram Research Inc.

About the authors

- **Diego Martín Oviedo Salcedo**, Civil Engineer – UIS – Bucaramanga, Masters in Civil Engineering (Geothecncs) – Universidad de Los Andes – Bogotá, PhD in Civil Engineering (Hydrogeology) – University of Illinois at Urbana – Champaign, Certified Mathematica® Trainer. diego.oviedo@upb.edu.co
- **Claudia Patricia Retamoso Llamas**, Civil Engineer – UIS – Bucaramanga, Masters in Civil Engineering (Structural Engineering) – Universidad de Los Andes – Bogotá. claudia.retamoso@upb.edu.co
- **Óscar Javier García Cabrejo**, Geologist – UNAL – Bogotá, Magister en Ingeniería Civil (Recorsos Hídricos) – Pontificia Universidad Javeriana – Bogotá, PhD Candidate in Civil Engineering (Hydrogeology) – University of Illinois at Urbana – Champaign. garcia30@illinois.edu
- **Édgar Eduardo Muñoz Díaz**, Civil Engineer – Universidad de Lasalle – Bogotá, Masters in Civil Engineering (Structural Engineering) – Universidad de Los Andes – Bogotá edgar.munoz@javeriana.edu.co