Numbolics: Applied Numerical and Symbolic Computations in Engineering Education

Thomas Schramm

Department of Geomatics, HafenCity University Hamburg

Abstract

Engineers often have a misleading concept in the advantages and disadvantages of numerical or symbolic methods in Mathematics and the use of computational tools. We show some examples and introduce our concept of using computeralgebra and numerical tools as Maple or Matlab to combine both approaches in a synergetic way.

Introduction (Headers)

In short, for most practical purposes, it is not a question of whether to proceed symbolically or numerically; rather, it is a question of how far to proceed symbolically before turning to numerics. Sommese and Wampler (2005, p. 67)

Years ago, when working as "consultant for scientific computing" at a computer center I was sometimes confronted with very strange methodologies for solving mathematical problems. For example, numerical root-finding of third order polynomials, handwritten codes for solving systems of ordinary differential equations with hundreds of equations in Visual Basic, handwritten implementation of finite differences for the solution of partial differential equations in Excel. The strategies typically started with handwritten manipulations of formulae (preconditioning) to find explicit expressions, which are then implemented into a numerical problem-solving environment (PSE, programming language, spreadsheet or even Matlab).

With the advent of Computer-Algebra-Systems (CAS), the situation changed (a bit). Scientists learned that they can even handle high-order equations (or systems) symbolically and got the also misleading idea that it is always better to prefer a symbolical solution, if there is one (compare e.g. Schramm (1998, 2000)).

We show an example adapted from Hermann (2006, p. 31), using Maple, that illuminates the situation:

Compare the three functions

$> P_1 := x \rightarrow x^8 - 8x^7 + 28x^6 - 56x^5 + 70x^4 - 56x^3 + 28x^2 - 8x + 1:$

$> P_2 := x \rightarrow 1 + (-8 + (28 + (-56 + (70 + (-56 + (28 + (-8 + x)x)x)x)x)x)x)x:$

$> P_3 := x \rightarrow (x-1)^8:$

and evaluate them at a location between 0.99 and 1.01, e.g in Maple using 10 significant digits: $P_1(0.998123) = 0., P_2(0.998123) = -1.2\ 10^{-8}, P_3(0.998123) = 1.540686159\ 10^{-22}$

This result is quite puzzling because the three functions are equivalent in a symbolic sense, as it is easy to see in Maple.

$> P_3(x) = expand(P_3(x));$

$$(x-1)^8 = x^8 - 8x^7 + 28x^6 - 56x^5 + 70x^4 - 56x^3 + 28x^2 - 8x + 1$$

and P_2 is just the horner-form of P_1

$> convert(P_1(x), horner);$

$$1 + (-8 + (28 + (-56 + (70 + (-56 + (28 + (-8 + x)x)x)x)x)x)x)x$$

Moreover, the last result (giving a wrong sign) is quite frustrating, since we have been told very often that polynomial expressions must be converted to their horner form if evaluated numerically. In Matlab, the situation is even worse since we have a fixed double precision accuracy. We can reach Matlab from within Maple:

$> with(Matlab) : set\,\mathrm{var}("x", 0.998123) :$

$> evalM(convert(P_1(x), string)) : get\,\mathrm{var}("ans");$

$$-6.21724893790087663\ 10^{-15}$$

$> evalM(convert(P_2(x), string)) : get\,\mathrm{var}("ans");$

$$1.99840144432528176\ 10^{-15}$$

$> evalM(convert(P_3(x), string)) : get\,\mathrm{var}("ans");$

$$1.54068615878524454\ 10^{-22}$$

One should keep in mind that both systems use IEEE-standards to evaluate numeric results.

We can look at the situation using some plots in Maple (note the different scales).

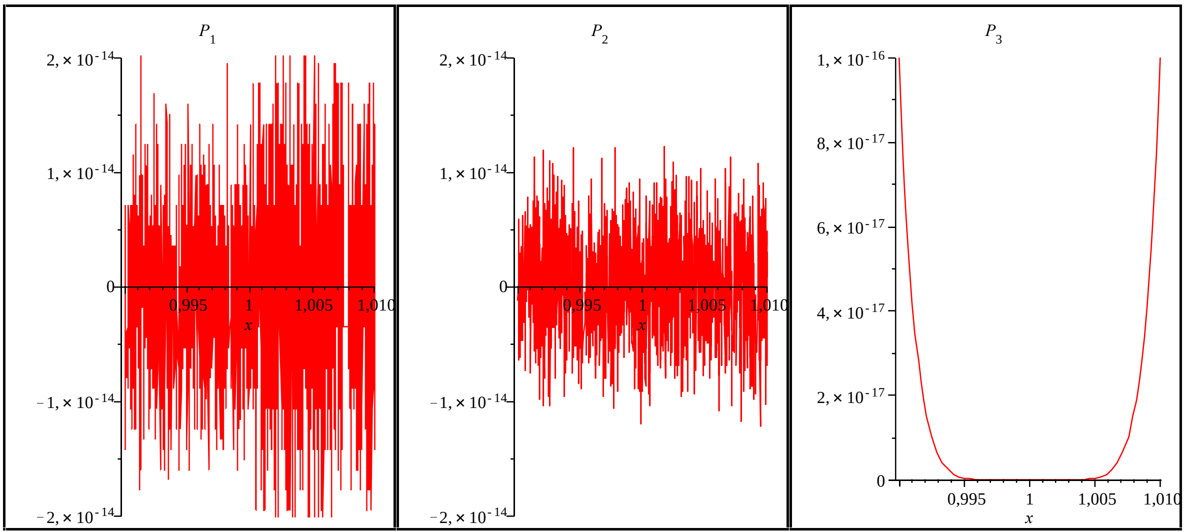
$> with(plots):$

$> display(Array([$

$plot(P_1(x),\ x = .99\ ..\ 1.01,\ \text{-0.2e-13}\ ..\ \text{0.2e-13},\ title = typeset(P_1)),$

$plot(P_2(\mathrm{x}),\ \mathrm{x} = .99\ ..\ 1.01,\ \text{-0.2e-13}\ ..\ \text{0.2e-13},\ title = typeset(P_2)),$

$plot(\mathrm{P}_3(x),\ x = .99\ ..\ 1.01,\ title = typeset(P_3))]));$



To analyze the situation we should try to understand the rules of machine computing, roundoff errors, catastrophic cancellations etc. (compare e.g. Hermann (2006)). However, using a CAS, we can do some experiments, e.g. we can change the accuracy and look when the result becomes stable. For example, the Maple command evalf(ex, n) evaluates the expression ex with n significant digits. Here a list of values for P_2 with ten to thirty significant digits.

> $seq(evalf(P_2(0.998123), i), i = 10 .. 30);$

$-1.2\ 10^{-8}$, $-1\ 10^{-10}$, $-1.4\ 10^{-10}$, $-4\ 10^{-12}$, $5\ 10^{-14}$, $5.9\ 10^{-14}$, $1.39\ 10^{-14}$, $5.1\ 10^{-16}$, $1.3\ 10^{-17}$, $3.9\ 10^{-18}$, $1.1\ 10^{-19}$, $1.29\ 10^{-19}$, $-4.\ 10^{-21}$, $5.3\ 10^{-22}$, $1.41\ 10^{-22}$, $1.539\ 10^{-22}$, $1.5430\ 10^{-22}$, $1.54060\ 10^{-22}$, $1.540673\ 10^{-22}$, $1.5406871\ 10^{-22}$, $1.54068593\ 10^{-22}$

What can we learn from this?

- It is not always a good idea to trust the numerical approximations of complicated symbolic expressions.
- Investigate the different forms of symbolic expressions.
- CAS can be used to analyse the effects of floating point arithmetic.

Numbolics

In the last section, we tried to explain that numerics or symbolics alone could lead to problems if one is note able to change the viewing direction. However, it occurs, that the preconditioning of formulae is too complex to give a canonical algorithm to be tackled by a numerical PSE. Here, only a combined approach of numerical and symbolic methods would apply. We show an example for the construction of averaging splines.

Averaging Splines

Splines are typically used for interpolation purposes. Datapoints are connected using (piecewise) cubic polynomials with connection conditions of identical slope and curvature in the datapoints. If the datapoints contain errors, it is senseless to run the splines exactly through the datapoints, i.e. we need an approximation scheme to find the connection points of the splines at the nodes of the datapoints. In the literature (Engeln-Müllges & Uhlig (1996, pp 287-297)), one finds such *fitting splines* with weighted datapoints introducing relations between artificial weights, datapoints and third derivatives of the spline function to have enough conditions to compute the connection points. A nice property of this approach is that one finds a regression line for small weights and an interpolations curve for large weights.

One can cast the conditions into a solvable linear system for the general case of a piecewise spline function for n datapoints and solve it using numerical PSEs.

However, there are some disadvantages:

- For n datapoints, we need $n-1$ splines. For dense datasets, these splines will mostly be nearly straight lines. A numerical overcast.
- The statistical properties of these splines are not very obvious.

Therefore, we use a different approach. We simply compute an avaraging spline for an artificial set of nodes in the datafield in the least squares sense. The algorithm goes as follows:

For a given set of n datapoints $x_i, y_i, i=1..n$

1. Choose a set of intermediate nodes $X_j, j=1..m, \; m<n, \; x_1 \le X_1, X_m \le x_n$
2. Construct a **symbolic** spline using the artificial nodes X_j and their **symbolic** function values Y_j.
3. Insert the datapoints into the symbolic spline function. The result is an overdetermined linear system of n equations in m unknowns Y_j.
4. Solve this system using the standard least squares method.
5. Replace the unknowns in the symbolic spline by their least squares approximation. The result is an averaging spline in the sense of least squares.

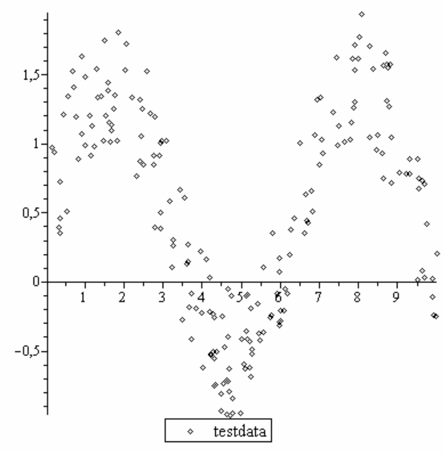
A working example shows how to implement this method in Maple:

Load and display a set of testdata

$> restart : with(plots) :$

$> TD := readdata(\text{"testdata.txt"}, 2, float):$

$> p_1 := pointplot(TD, legend = \text{"testdata"}); display(p_1);$



$> n := nops(TD);$

$$n := 200$$

Extract the x- and y-values

$x := TD[1..n,1]: \; y := TD[1..n,2]:$

Step 1: We decide to construct an averaging spline for ten equidistant nodes

$> m := 10: \quad X := \left[x_1 + \frac{(x_n - x_1)}{m-1} \cdot j \$ j = 0..m-1 \right];$

X:=[0.131, 1.225333333, 2.319666666, 3.413999999, 4.508333332, 5.602666665, 6.696999998, 7.791333331, 8.885666664, 9.979999997]

Definition of the symbolic Y-values

$> Ylist := [Y_j \$ j = 1 .. m)];$

$$Ylist := [Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7, Y_8, Y_9, Y_{10}]$$

Step 2: Construction of the symbolic spline function S where we use ξ as free variable

> $S := unapply(CurveFitting[Spline](X, Ylist, \xi), \xi)$:

Step 3: Construction of the overdetermined linear system as set of equations

> $sys := [S(x_i) = y_i \$ i = 1 .. n]$:

and in matrix form

> $LinearAlgebra[GenerateMatrix](sys, Ylist)$;

$$A, b := \begin{bmatrix} \text{200 x 10 Matrix} \\ \text{Data Type : anything} \\ \text{Storage : rectangular} \\ \text{Order : Fortran\_order} \end{bmatrix} \begin{bmatrix} \text{1..200 } \mathit{Vector}_{column} \\ \text{Data Type : anything} \\ \text{Storage : rectangular} \\ \text{Order : Fortran\_Order} \end{bmatrix}$$

Step 4/5: with solution

> $Y := LinearAlgebra[LeastSquares](A, b)$;

$$Y := \begin{bmatrix} 0.713861487313643184 \\ 1.27752906020948488 \\ 1.19573843117451338 \\ 0.279316935006456156 \\ -0.559365367832887928 \\ -0.245539768515272788 \\ 0.665558850251661726 \\ 1.43528572834436163 \\ 1.12751943464574711 \\ -0.0629727008745847306 \end{bmatrix}$$

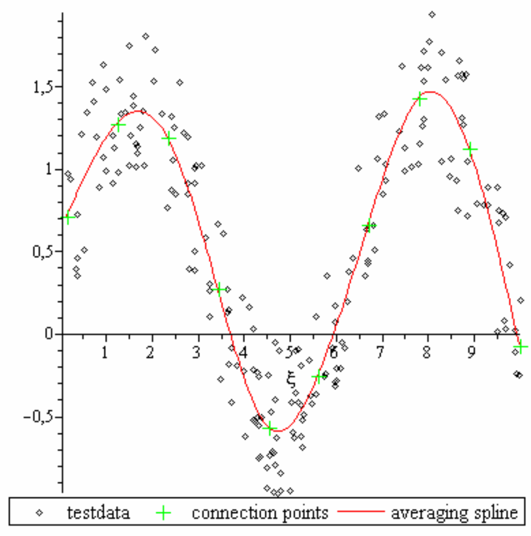
The spline function now reads

> $S(\xi);$

$$\begin{cases} 0.6326190102+0.6171158460\xi+0.0349893475\xi^2-0.0890314189\xi^3 & \xi < 1.225333333 \\ 0.5559490056+0.8048280417\xi-0.1182034120\xi^2-0.047357655\xi^3 & \xi < 2.319666666 \\ -1.709147077+3.734252494\xi-1.381067811\xi^2+0.1341144541\xi^3 & \xi < 3.413999999 \\ -4.612589258+6.285607675\xi-2.128389181\xi^2+0.2070808056\xi^3 & \xi < 4.508333332 \\ 27.40550992-15.02033633\xi+2.597513364\xi^2-0.1423389761\xi^3 & \xi < 5.602666665 \\ 13.17013909-7.397874561\xi+1.237007335\xi^2-0.06139501919\xi^3 & \xi < 6.696999998 \\ 47.50310900-22.77773265\xi+3.533536826\xi^2-0.1757014004\xi^3 & \xi < 7.791333331 \\ -59.19824616+18.30689971\xi-1.739582735\xi^2+0.04989625543\xi^3 & \xi < 8.885666664 \\ -111.7013145+36.03311108\xi-3.734504667\xi^2+0.1247329552\xi^3 & \text{otherwise} \end{cases}$$

and can be used for plotting

> $p_2 := plot(S(\xi), \xi = x_1 .. x_n, legend = \text{"averaging spline"});$
$p_3 := pointplot(X, convert(Y, list), color = green, symbolsize = 20,$
$symbol = cross, legend = \text{"connection points"});$
$display(p_1, p_2, p_3);$



In the last example, we used the symbolical features of a CAS to construct the system of equations that was solved using the numerical features. We used the artificial word "**numbolics**" to state that both features are important. Most CAS can do that und have (against the common belief) a similar numerical performance as dedicated numerical systems. However, it is a matter of taste, which systems are to be used. If you like e.g. matlab better, one can include the symbolic toolbox (which is a full CAS) to do the same task.

Conclusions

We showed some examples about the interaction of numerics and symbolics in mathematical problem solving. Engineering students should be aware of these implications. They should not know only about the theoretical background but also about the software systems to tackle their mathematical problems. They should be able to use both numerical and symbolic strategies to combine them to powerful methods.

References (Headers)

Engeln-Müllges, G. and Uhlig, F. (2006) *Numerical Algorithms with Fortran*, Springer.

Hermann, M. (2006) *Numerische Mathematik*. Oldenbourg Wissenschaftsverlag GmbH, München.

Schramm, T. (1998) Computer Algebra Systems in Engineering Education. *Global J. of Eng. Educ.*, Vol. 2, No.2: 187-194.

Schramm, T. (2000) "Mathematical Problem Solving Environments for education and research." In *Proc. 2nd Global Congress on Engineering Education*, UICEE, Wismar, Germany, 2-7 July.

Sommese, A.J., Wampler II, C.W. (2005) *The Numerical Solution of Systems of Polynomials*, World Scientific Publishing Co. Pte. Ltd.