Stud. Univ. Babeş-Bolyai Math. ??(??), No. ??, 1–4

Solution of nonlinear equations via Padé approximation. A Computer Algebra approach

Radu T. Trîmbiţaş

Abstract. We generate automatically several high order numerical methods for the solution of nonlinear equations using Padé approximation and Maple CAS.

Mathematics Subject Classification (2010): 65H05,65D05,65D99.

Keywords: Nonlinear equation, Padé approximation, Numerical methods, Computer Algebra.

1. Introduction

Consider the nonlinear scalar equation

$$f(x) = 0, \tag{1.1}$$

where $f : D \subseteq \mathbb{R} \to \mathbb{R}$ is a continuous and differentiable as many times as necessary. Let α be a solution of (1.1). Let $\mathcal{R}_{m,p}$ be the set of rational functions with degree of numerator m and degree of denominator p. Suppose f has a formal Taylor series

$$f(z) = c_0 + c_1 z + c_2 z^2 + \cdots .$$

For any pair $(m,p) \in \mathbb{N} \times \mathbb{N}$, $r_{mp} \in \mathcal{R}_{m,p}$ is the type (m,p) Padé approximant to f if their Taylor series at $z = 0$ agree as far as possible:

$$(f - r_{mp})(z) = O(z^{max}) \tag{1.2}$$

We will use three different strategies based on Padé in order to obtain automatically high order method:

- a direct strategy;
- inverse interpolation;
- modified methods.

The features of Maple CAS allow us to generate methods of arbitrary orders. See [4] or [6] for details. The `pade` procedure from the `numapprox` package computes a Padé approximation of degree (m, p) about a given point. The paper [3] and the book [2] contain several interesting examples of using Computer Algebra for the derivation of numerical methods. In the sequel we will consider one-step methods, i.e. methods of the form

$$x_{n+1} = F(x_n), \qquad x_0 \text{ given.}$$

For the sake of brevity we will use the notations $f_n = f(x_n)$ and $f_n^{(k)} = f^{(k)}(x_n)$.

2. The direct approach

The first strategy is to approximate f by its (m, p) Padé approximant $r_{m,p} \in \mathcal{R}_{m,p}$ and to solve the equation $r_{m,p}(x) = 0$. The iteration will have the form

$$x_{n+1} = F(x_n),$$

where $F(x)$ is the root of $r_{m,p}(x) = 0$ as a function of x. In order to avoid the solution of higher order equations we will choose $m = 1$.

For example, for $m = 1$ and $p = 0$, we obtain the Newton's method.

```
> restart;
> with(numapprox):
> F:=pade(f(t),t=x[n],[1,0]):
> G:=collect(solve(%,t),x[n]);
```

$$G := x_n - \frac{f(x_n)}{\mathrm{D}(f)(x_n)}$$

or,

$$x_{n+1} = x_n - \frac{f_n}{f'_n}.$$

For $m = 1$ and $p = 1$, we obtain Halley's method.

```
> F:=pade(f(t),t=x[n],[1,1]):
> G:=collect(solve(%,t),x[n]);
```

$$G := x_n - 2\,\frac{\mathrm{D}(f)(x_n)\,f(x_n)}{2\,(\mathrm{D}(f)(x_n))^2 - \left(D^{(2)}\right)(f)(x_n)\,f(x_n)}$$

or,

$$x_{n+1} = x_n - \frac{2f'_n f_n}{2\,(f'_n)^2 - f''_n f_n.}$$

This formula was obtained using direct Padé approximation in [2].

These are in fact particular cases of Householder-type methods. They could be obtained by considering $(1, p)$ Padé approximation and solving the equation $r_{1p} = 0$. Their order is $p + 2$. If $f \in C^{p+1}(V)$, where V is a neighborhood of α, Househelder showed in [9] that the general form of iteration is

$$x_{n+1} = x_n + (p+1) \left. \frac{\left(\frac{1}{f}\right)^{(p)}}{\left(\frac{1}{f}\right)^{(p+1)}} \right|_{x_n} .$$

The generation of such a method is straightforward with the following one-line Maple code

```
>  Phi:=(x,p)->x+(p+1)*(D@@(p))(1/f)(x)/(D@@(p+1))(1/f)(x):
```

We give two examples, for $p = 2$ and $p = 3$. The results were converted to mathematical notation.

```
>  F_2:=x+normal(Phi(x,2)-x);
>  F_3:=x+normal(Phi(x,3)-x);
```

$$F_2 := x - 3 \frac{\left[2f'^2(x) - f''(x)f(x)\right] f(x)}{f'''(x)f^2(x) + 6f'^3(x) - 6f''(x)f'(x)f(x)} \tag{2.1}$$

$$F_3 := x + \frac{4\left[f'''(x)f^2(x) + 6f'^3(x) - 6f''(x)f'(x)f(x)\right] f(x)}{Q(x)}, \tag{2.2}$$

where

$$\begin{aligned} Q(x) = f^{(4)}(x)f^3(x) - 8f'''(x)f'(x)f^2(x) - 24f'^4(x) + \\ 36f''(x)f'^2(x)f(x) - 6f''^2(x)f^2(x) \end{aligned} \tag{2.3}$$

3. Inverse Interpolation

Suppose there exists $g = f^{-1}$ on a neighborhood V of α. The inverse interpolation consists of approximating

$$\alpha = g(0),$$

by the value of an interpolant $\widehat{g}$ for g at 0

$$\alpha = \widehat{g}(0).$$

In this section we will use inverse Padé interpolation. The formula we look for will have the form

$$x_{k+1} = r_{mp}(x_k), \qquad k = 0, 1, ,$$

where r_{mp} is the (m, p) Padé approximant for $g(0)$. For details on inverse interpolation see [1], [5], [7]. The paper [7] uses rational interpolation to derive methods for the solution of scalar nonlinear equations. The Maple procedure `invpade` generates the iteration function based on (m, p)-inverse Padé interpolation.

```
> invPade:=proc(m::nonnegint,p::nonnegint)
> local f,x;
> x+collect(eval(pade((f@@(-1))(y),y=f(x),[m,p]),y=0)-x,
> x,simplify);
> end proc;
```

We give examples for $(m,p) \in \{(1,1),(2,1),(2,2)\}$. The results were edited, in order to fit on page.

Formula for $(1,1)$ is the Halley's formula.

```
> F11:=invPade(1,1);
```

$$F_{11} := x + 2\frac{f'(x)f(x)}{f''(x)f(x) - 2f'^2(x)}$$

Formula for $(2,1)$ was given and studied in [10].

```
> F21:=invPade(2,1);
> convert(%,diff);
```

$$F_{21} := x - \frac{f(x)\left[f(x)f'(x)f'''(x) - \frac{3}{2}f(x)f''^2(x) + 3f'^2(x)f''(x)\right]}{f'(x)\left[f(x)f'(x)f'''(x) - 3f(x)f''^2(x) + 3f'^2(x)f''(x)\right]} \tag{3.1}$$

Note that the formula for $(1,2)$ is different from (2.1) (that is, the direct approach and inverse interpolation generates different formulas for (1,2) pair of degrees). The $(2,2)$-type formula is

$$F_{22} = x + \frac{U}{V}, \tag{3.2}$$

where

$$U = 6ff'\left[f\left(f'\right)^2 f^{(4)} - 6ff'f''f''' + 6f\left(f''\right)^3 + 4f'''\left(f'\right)^3 - 6\left(f''\right)^2\left(f'\right)^2\right](x)$$

and

$$\begin{aligned} V = f^2 & \left(3\left(f'\right)^2 f^{(4)} f'' - 4\left(f'\right)^2\left(f'''\right)^2 - 6f'\left(f''\right)^2 f''' + 9\left(f''\right)^4\right)(x) \\ & - 6f\left(f'\right)^2\left(\left(f'\right)^2 f^{(4)} - 8f'f''f''' + 9\left(f''\right)^3\right)(x) \\ & - 12\left(f'\right)^4\left(2f'f''' - 3\left(f''\right)^2\right)(x). \end{aligned}$$

4. Modified methods

Following the ideas of Sebah and Gourdon [8], we look for an iteration of the form

$$x_{n+1} = x_n + h_n + a_2\frac{h_n^2}{2!} + a_3\frac{h_n^3}{3!} + \cdots, \tag{4.1}$$

where $h_n = -\frac{f(x_n)}{f'(x_n)}$. Under the assumptions that f is sufficiently differentiable and $h_n + a_2\frac{h_n^2}{2!} + a_3\frac{h_n^3}{3!} + \cdots$ is small, we start from Taylor expansion of $f(x_{n+1})$ about x_n, and using the side-relation $f(x_n) + h_n f'(x_n) = 0$, we try to choose a_n's so that to cancel as many terms as possible in the expansion.

The Maple procedure `modPade` below returns the coefficients (a_k) and the modified method (4.1) truncated to a given number of terms.

```
> modPade:=proc(nmax::nonnegint)
> local k, inc,dT, dT2, sol, a, ec, so, it, n ;
> inc:=h+add(a[k]*h^k/k!,k=2..max(nmax+1,3));
> dT:=convert(taylor(f(x[n]+t),t=0,nmax+1),polynom);
> dT:=simplify(subs(t=inc,dT),[f(x[n])+h*D(f)(x[n])=0]):
> dT2:=collect(dT,h,simplify):
> for k from 2 to nmax+1 do
> ec[k]:=coeff(dT2,h,k);
> end;
> so:=solve([seq(ec[k],k=2..nmax+1)],[seq(a[k],k=2..nmax+1)]);
> assign(so);
> it:=x[n]+eval(subs(h=-f(x[n])/D(f)(x[n]),factor(inc)));
> return a,it;
> end proc:
```

`modPade` computes for a_k, $k=2,\ldots,6$, the following values

$$a_2 = -\frac{f_n''}{f_n'}$$

$$a_3 = \frac{3\left(f_n''\right)^2 - f_n''' f_n'}{\left(f_n'\right)^2}$$

$$a_4 = -\frac{f_n^{(4)}\left(f_n'\right)^2 - 10 f_n''' f_n'' f_n' + 15\left(f_n''\right)^3}{\left(f_n'\right)^3}$$

$$a_5 = \frac{105\left(f_n''\right)^4 - 105 f_n'''\left(f_n''\right)^2 f_n' + 15 f_n^{(4)} f_n''\left(f_n'\right)^2 + 10\left(f_n'\right)^2\left(f_n'''\right)^2 - f_n^{(5)}\left(f_n'\right)^3}{\left(f^{(4)}\right)^4}$$

$$a_6 = -\frac{7}{\left(f_n'\right)^5}\Big(135\left(f_n''\right)^5 - 180 f_n'''\left(f_n''\right)^3 f_n' + 30 f_n^{(4)}\left(f_n''\right)^2\left(f_n'\right)^2 + 40 f_n''\left(f_n'''\right)^2\left(f_n'\right)^2$$
$$-3 f_n^{(5)} f_n''\left(f_n'\right)^3 - 5 f_n''' f_n^{(4)}\left(f_n'\right)^3\Big)$$

For $n_{\max}=4$, `modPade` gives the fourth-order formula

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} - \frac{f''(x_n) f^2(x_n)}{2\left(f'(x_n)\right)^3} + \frac{\left(f'''(x_n) f'(x_n) - 3\left(f''(x_n)\right)^2\right) f^3(x_n)}{6\left(f'(x_n)\right)^5} \tag{4.2}$$

For $n_{\max}=5$, `modPade` gives the fifth-order formula

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} - \frac{f''(x_n) f^2(x_n)}{2\left(f'(x_n)\right)^3} + \frac{\left(f'''(x_n) f'(x_n) - 3\left(f''(x_n)\right)^2\right) f^3(x_n)}{6\left(f'(x_n)\right)^5}$$
$$- \frac{\left(f^{(4)}(x_n)\left(f'(x_n)\right)^2 - 10 f'''(x_n) f''(x_n) f'(x_n) + 15\left(f''(x_n)\right)^3\right) f^4(x_n)}{24\left(f'(x_n)\right)^7} \tag{4.3}$$

Remark 4.1. These methods are the same as Chebyshev methods and could be generated using inverse Taylor interpolation (see [1, 7]).

5. Numerical examples

We wish to compare the different iterations on the solution of the equation

$$xe^x + x^2 - 6 = 0. \tag{5.1}$$

First, we compute the solution using `fsolve` function with `Digits` set to 400.

```
> Digits:=400:
> eq:=x*exp(x)+x^2-6:
> root1:=fsolve(eq,x);
```

$$\begin{aligned} \mathit{root1} :=&1.257169468081542443224161713705996802920131265042900 76\backslash\\ &142355162009975113083056615579120160569103718598288101\backslash\\ &1405588031134339216304359398109887530866 36\ldots \end{aligned}$$

Then, for each method we execute a small number of iteration steps and count the number of correct digits and compute the absolute error as the modulus of the difference between `root1` and the computed approximation.

- Padé $(1,2)$, order 4 (formula (2.1))

$x_1 = 1.26(257\ldots)$ 2 digits

$x_2 = 1.2571694681(095\ldots)$ 10 digits

$x_3 = 1.257169468081542443224161713705996802920 1312(853\ldots)$ 43 digits

$x_4 = 1.2571694680815424432241617137059968029201312 6504(\ldots)$ 176 digits

- inverse Padé $(2,1)$, order 4 (formula (3.1))

$x_1 = 1.2(727\ldots)$ 1 digits

$x_2 = 1.2571694(737\ldots)$ 8 digits

$x_3 = 1.257169468081542443224161713705996 9(004\ldots)$ 34 digits

$x_4 = 1.25716946808154244322416171370599680292013126 50(\ldots)$ 137 digits

- modified method, order 4 (formula (4.2))

$x_1 = 1.3(106\ldots)$ 1 digits

$x_2 = 1.25717(411\ldots)$ 5 digits

$x_3 = 1.257169468081542443224(458\ldots)$ 21 digits

$x_4 = 1.2571694680815424432241617137059968029201312 6504(\ldots)$ 86 digits

- Padé $(1,3)$, order 5 (formulas (2.2) and (2.3))

$x_1 = 1.257(703\ldots)$ 3 digits

$x_2 = 1.257169468081542443(624\ldots)$ 18 digits

$x_3 = 1.25716946808154244322416171370599680292013126 5(\ldots)$ 94 digits

$x_4 = 1.2571694680815424432241617137059968029201312 6504(\ldots)$ 472 digits

Note that this method was tested for `Digits` *set to* 500.

- inverse Padé $(2,2)$, order 5 (formula (3.2))

$x_1 = 1.26(\ldots)$ 2 digits

$x_2 = 1.2571694680815(682\ldots)$ 13 digits

$x_3 = 1.257169468081542443224161713705996802920131265(\ldots)$ 69 digits

$x_4 = 1.257169468081542443224161713705996802920131265 04(\ldots)$ 348 digits

- modified method, order 5 (formula (4.3))

$x_1 = 1.(2846\ldots)$ 1 digits

$x_2 = 1.257169(479\ldots)$ 7 digits

$x_3 = 1.2571694680815424432241617137059968 02920(249\ldots)$ 39 digits

$x_4 = 1.257169468081542443224161713705996802920131265 0(\ldots)$ 199 digits

Tables 1 and 2 give the error after each iteration for 4th order and for 5th order methods, respectively.

Iteration	Padé $(1,2)$	Inverse Padé $(2,1)$	Modified order 4
1	$5.4033e-03$	$1.5528e-02$	$5.3445e-02$
2	$2.7982e-11$	$5.6144e-09$	$4.6404e-06$
3	$2.0247e-44$	$9.7495e-35$	$2.9607e-22$
4	$5.5508e-177$	$8.8659e-138$	$4.9061e-87$

Table 1. Errors for each iteration, 4th order methods

Iteration	Padé $(1,3)$	Inverse Padé $(2,2)$	Modified order 5
1	$5.3370e-04$	$3.7722e-03$	$2.7441e-02$
2	$4.0001e-19$	$2.5751e-14$	$1.0904e-08$
3	$9.4690e-95$	$3.8318e-70$	$1.1775e-40$
4	$7.0386e-473$	$2.7954e-349$	$1.7284e-200$

Table 2. Errors for each iteration, 5th order methods

6. Conclusions

.

All methods presented computes a large number of correct digits in a small number of iterations. Direct Padé and inverse Padé methods are superior to modified methods. Direct Padé methods, (in fact, Householder methods) have a better accuracy than methods based on inverse Padé interpolation of the same total degree, at least for equation (5.1). The approach presented in this paper could be useful in the context of symbolic computation, when a large number of digits is required and to automatically generate numerical methods for the solution of nonlinear equations.

References

[1] Agratini, O., Blaga, P., Chiorean, I., Coman, Gh., Stancu, D.D., Trîmbiţaş, R.T., *Numerical Analysis and Approximation Theory* (vol. III), Cluj University Press, Cluj-Napoca, 2002 (in Romanian).

[2] Gander, W., Gander, M.J., Kwok, F., *Scientific Computing. An Introduction Using Maple and MATLAB*, Springer, 2014.

[3] Gander, W., Gruntz, D., Derivation of numerical methods using Computer Algebra, *SIAM Rev.*, 41(1999), no. 3, 577-593.

[4] Garvan, F., *The Maple Book*, 1st Edition, Chapman & Hall/CRC, 2001.

[5] Gautschi, W., Numerical Analysis, Second Edition, Springer Science+Business Media, 2012.

[6] Heck, A., *Introduction to Maple*, Third Edition, Springer-Verlag, New York, 2003.

[7] Păvăloiu, I., Equations Solution through Interpolation, Dacia Publishers, 1981 (in Romanian).

[8] Sebah, P., Gourdon, X., Newton's method and high order iterations, `numbers.computation.free.fr/Constants/constants.html`

[9] Householder, A. S., *The Numerical Treatment of a Single Nonlinear Equation*, McGraw-Hill, New York, (1970)

[10] Trîmbiţaş, R., An application of inverse Padé interpolation, *Stud. Univ. Babeş-Bolyai Math.* **64**(2019), No. 2, 291–296

Radu T. Trîmbiţaş
"Babeş-Bolyai" University,
Faculty of Mathematics and Computer Sciences
1, Kogălniceanu Street,
400084 Cluj-Napoca,
Romania
e-mail: `tradu@math.ubbcluj.ro`