Illustrative example

Consider a system of ODEs[8]

$$f''' - (f')^2 + ff'' + 2\lambda g + \beta[2ff'f'' - f^2 f'''] = 0, \tag{1}$$

$$g'' - f'g + fg' - 2\lambda f' + \beta[2ff'g' - f^2 g''] = 0, \tag{2}$$

subject to

$$f'(0) = 1, f(0) = 0, g(0) = 0, \tag{3}$$

$$f'(\infty) = 0, g(\infty) = 0, \tag{4}$$

where λ is the rotation parameter, β is the viscoelastic parameter, and the prime indicates the differentiation with respect to η. This system models two-dimensional flow of an upper convected Maxwell fluid in a rotating frame. It has been solved by the HAM in Ref. 8.

To solve this problem by `BVPh 2.0`, we have to input the differential equations, boundary conditions, initial guesses and convergence-control parameters. The differential equations (1) and (2) can be coded as follows

```
TypeEQ = 1;
NumEQ = 2;
f[1,z_,{f_,g_},Lambda_]:=
    D[f,{z,3}]-D[f,z]^2+f*D[f,{z,2}]+2*la*g+
    beta*(2*f*D[f, z]*D[f,{z,2}]-f^2*D[f,{z,3}]);
f[2,z_,{f_,g_},Lambda_]:=
    D[g,{z,2}]-D[f,z]*g+f*D[g,z]-2*la*D[f,z]+
    beta*(2*f*D[f, z]*D[g, z]-f^2*D[g,{z, 2}]);
```

Here `TypeEQ` controls the type of governing equations: `TypeEQ=1` corresponds to a system of ODEs without an unknown to be determined, `TypeEQ=2` corresponds to a system of ODEs with an unknown, `Lambda`, to be determined. Since all the parameters in the problem will be given, we set `TypeEQ` to 1. Note that we use the delayed assignment `SetDelayed(:=)` in Mathematica to define these ODEs to avoid the evaluation when the assignment is made.

The boundary conditions (3) and (4) are defined in a semi-infinite interval, from 0 to $+\infty$. They are coded as

```
NumBC = 5;
BC[1,z_,{f_,g_}]:=(D[f, z]-1)/.z->0;
BC[2,z_,{f_,g_}]:=f/.z->0;
BC[3,z_,{f_,g_}]:=g/.z->0;
BC[4,z_,{f_,g_}]:=D[f,z]/.z->infinity;
BC[5,z_,{f_,g_}]:=g/.z->infinity;
```

Here `NumBC` is the number of boundary conditions of the problem. For this problem, we have 5 boundary conditions, so `NumBC` is set to 5. The symbol `infinity` is introduced in our package to denote ∞. When an expression of the boundary conditions contains `infinity`, the limit of the expression is computed as `z` approaches ∞. The delayed assignment (`:=`) is also used to avoid the evaluation when the assignment is made—the same reason as defining the differential equations.

For a multi-layer problem, the differential equations in the system are not necessarily in the same interval (see example 4 in Section 4). Hence, we have to give each equation its solution interval. To measure the accuracy of the approximate solutions, we have to compute the squared residual over the corresponding solution interval. In practice, when the differential equation is defined in a semi-infinite interval, we simply truncate the infinite interval to a finite interval to compute the squared residual, or it will take a lot of computation time. For this problem, the solution interval for each equation and the integral interval for squared residual are defined as

```
    zL[1] = 0;
    zR[1] = infinity;
    zL[2] = 0;
    zR[2] = infinity;
    zRintegral[1] = 10;
    zRintegral[2] = 10;
```

Here `zL[k]` (or `zR[k]`) is the left (or right) end point of the solution interval for the k-th equation `f[k,z,{f,g},Lambda]`. And `zLintegral[k]` (or `zRintegral[k]`) is the left (or right) end point of the integral interval to compute the squared residual for the k-th equation. If the value of `zL[k]` (or `zR[k]`) is a finite number, `zLintegral[k]` (or `zRintegral[k]`) is set to the same value automatically. However, if any of them contains the symbol `infinity`, we have to set the corresponding end point of the integral interval to a finite value. That's why we write explicitly `zRintegral[1]=10` and `zRintegral[2]=10`. For this problem, the squared residual is integrated over the range [0, 10] for both equations.

The auxiliary linear operators for this problem are chosen as $\mathcal{L}_1 = \frac{\partial^3}{\partial\eta^3} - \frac{\partial}{\partial\eta}$, $\mathcal{L}_2 = \frac{\partial^2}{\partial\eta^2} - 1$, which are coded as

```
    L[1,u_] := D[u,{z,3}]-D[u,z];
    L[2,u_] := D[u,{z,2}]-u;
```

Here `L[k,u]` is the auxiliary linear operator corresponding to the k-th equation. Note that i) η is the independent variable in the differential equations (1) and (2), while z is the universal independent variable in the package `BVPh 2.0`; ii) The delayed assignment `SetDelayed(:=)` is used to define the operator; iii) u is a formal parameter.

For this problem, the initial guesses are $f_0 = 1 - e^{-z}$ and $g_0 = ze^{-z}$. They are coded as

```
U[1,0] = 1-Exp[-z];
U[2,0] = alpha*z*Exp[-z];
```

Here `alpha` is an introduced convergence-control parameter that will be determined later. `U[k,0]` is the initial guess of the k-th equation. Note that `U[k,0]` and `u[k,0]` are usually the same in the package `BVPh 2.0`.

We want to solve this problem when the physical parameters $\beta = 1/5$ and $\lambda = 1/10$. These two parameters are coded as

```
beta = 1/5;
la = 1/10;
```

So far, we have defined all the input of this problem properly, except the convergence-control parameter `c0[k]` and `alpha`. Usually, the optimal values of the convergence-control parameters are obtained by minimizing the squared residual error. For this problem, we get the approximate optimal values of `c0[1]`, `c0[2]` and `alpha` by minimizing the squared residual error of the 3rd-order approximation as

```
GetOptiVar[3, {}, {c0[1],c0[2],alpha}];
```

The first parameter of `GetOptiVar` denotes which order approximation is used. Here 3 means the 3rd-order approximation is used. The second parameter denotes a list of constraints used in the optimization. When the second parameter of `GetOptiVar` is an empty list, it means the squared residual is minimized without any constraint. Here we add no constraints to minimize the squared residual. The third parameter is a list of the variables to be optimized. Here we want to optimize `c0[1]`, `c0[2]` and `alpha`. After some computation, it gives the optimized convergence-control parameters `c0[1]=-1.26906`, `c0[2]=-1.19418` and `alpha=-0.0657063`.

Now we can use

```
BVPh[1,10]
```

to get the 10th-order approximation. If we are not satisfied with the accuracy of the 10th-order approximation, we can use `BVPh[11,20]`, instead of `BVPh[1,20]`, to get 20th-order approximation or higher order approximation.

The kth-order approximation of the ith differential equation is stored in `U[i,k]`. We can use

```
Plot[{U[1,20], U[2,20]}, {z,0,10},
AxesLabel->{"\[Eta]", ""},
PlotStyle->{{Thin, Red},{Dashed, Blue}}]
```

to plot the 20th-order approximate solution, which is shown in Fig. 1.

The accuracy of the kth-order approximation is measured by the squared residual. We can use

```
ListLogPlot[Table[{2*i,ErrTotal[2*i]},{i,1,10}],
Joined->True,Mesh->All,
PlotRange->{{2,20},{10^(-15),10^(-5)}},
AxesLabel->{"m", "error"}];
```

to plot the curve of the total error versus the order of approximation, which is shown in Fig. 2. Note that `ErrTotal[k]` stores the total error of the system when the kth-order approximation is used, while `Err[k]` is a list that stores the error for each ODE in the system when the kth-order approximation is used.

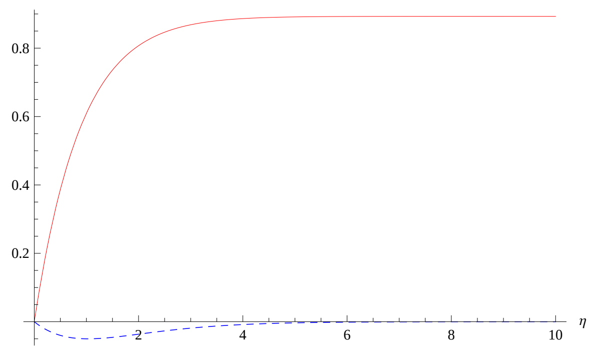

Fig. 1: The curve of $f(z)$ (solid) and $g(z)$ (dashed) for the illustrative example when $\beta = 1/5$, $\lambda = 1/10$.

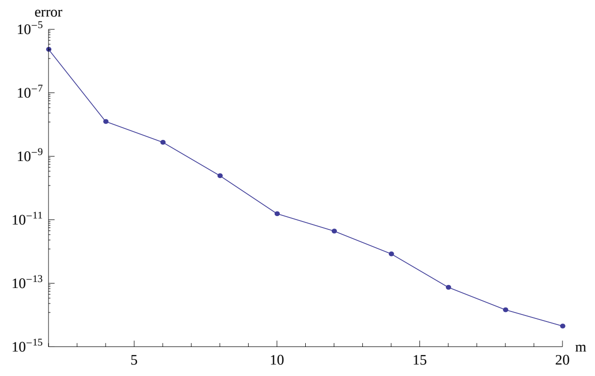

Fig. 2: Total error vs. order of approximation for the illustrative example when $\beta = 1/5$, $\lambda = 1/10$.

References

1. S. J. Liao, *Proposed homotopy analysis techniques for the solution of nonlinear problem.* Ph.D. thesis, Shanghai Jiao Tong University (1992).

2. S. J. Liao, A uniformly valid analytic solution of two-dimensional viscous flow over a semi-infinite flat plate, *J. Fluid Mech..* **385**, 101–128 (1999).

3. S. J. Liao, On the analytic solution of magnetohydrodynamic flows of non-newtonian fluids over a stretching sheet, *J. Fluid Mech..* **488**, 189–212 (2003).

4. S. J. Liao, Series solutions of unsteady boundary-layer flows over a stretching flat plate, *Stud. Appl. Math..* **117**(3), 239–263 (2006).

5. S. J. Liao, *Beyond Perturbation—Introductioin to the Homotopy Analysis Method.* Chapman & Hall/CRC Press, Boca Raton (2003).

6. S. J. Liao, *Homotopy Analysis Method in Nonlinear differential equations.* Springer-Verlag Press, New York (2011).

7. S. J. Liao, Notes on the homotopy analysis method: Some definitions and theorems, *Commun. Nonlinear Sci. Numer. Simulat..* **14**, 983–997 (2009).

8. M. Sajid, Z. Iqbal, T. Hayat and S. Obaidat, Series solution for rotating flow of an upper convected Maxwell fluid over a strtching sheet, *Commun. Theor. Phys..* **56**(4), 740–744 (2011).

9. T. Hayat, M. Nawa and A. A. Hendi, Heat transfer analysis on axisymmetric MHD flow of a micropolar fluid between the radially stretching sheets, *J. Mech..* **27**(4), 607–617 (2011).

10. H. Xu, T. Fan and I. Pop, Analysis of mixed convection flow of a nanofluid in a vertical channel with the Buongiorno mathematical model, *Int. Commun. Heat Mass.* **44**, 15–22 (2013).

11. J. C. Umavathi, I. C. Liu and J. Prathap Kumar. Magnetohydrodynamic Poiseuille-Couette flow and heat transfer in an inclined channel, *J. Mech..* **26**(4), 525–532 (2010).

12. J. P. Boyd, An analytical and numerical study of the two-dimensional Bratu equation, *J. Sci. Comput..* **1**(2), 183–206 (1986).

13. J. Jacobsen and K. Schmitt, The Liouville-Bratu-Gelfand problem for radial operators, *J. Differ. Equations.* **184**, 283–298 (2002).

14. J. S. McGough, Numerical continuation and the Gelfand problem, *Appl. Math. Comput..* **89**(1-3), 225–239 (1998).

15. Y. L. Zhao, Z. L. Lin and S. J. Liao, An iterative analytical approach for nonlinear boundary value problems in a semi-infinite domain, *Comput. Phys. Commun..* Online.