

---

# The Shooting Technique for the Solution of Two-Point Boundary Value Problems

Douglas B. Meade\*, Bala S. Haran†, and Ralph E. White‡

---

## Introduction

One of the strengths of Maple is its ability to provide a wide variety of information about differential equations. Explicit, implicit, parametric, series, Laplace transform, numerical, and graphical solutions can all be obtained via the `dsolve` command. Numerical solutions are of particular interest due to the fact that exact solutions do not exist, in closed form, for most engineering and scientific applications. The numerical solution methods available within `dsolve` are applicable only to *initial value problems*. Thus, at first glance, Maple appears to be very limited in its ability to analyze the multitude of two-point boundary value problems that occur frequently in engineering analysis.

A commonly used numerical method for the solution of two-point boundary value problems is the *shooting method*. This well-known technique is an iterative algorithm which attempts to identify appropriate initial conditions for a related initial value problem (IVP) that provides the solution to the original boundary value problem (BVP).

The first objective of this paper is to describe the shooting method and its Maple implementation, `shoot`. Then, `shoot` is used to analyze three common two-point BVPs from chemical engineering: the Blasius solution for laminar boundary-layer flow past a flat plate, the reactivity behavior of porous catalyst particles subject to both internal mass concentration gradients and temperature gradients, and the steady-state flow near an infinite rotating disk.

## A Maple Implementation of the Simple Shooting Method

The basic idea of the shooting method for two-point boundary value problems is to reformulate the problem as a nonlinear parameter estimation problem. The new

problem requires the solution of a related IVP with initial conditions chosen to approximate the boundary conditions at the other endpoint. If these boundary conditions are not satisfied to the desired accuracy, the process is repeated with a new set of initial conditions until the desired accuracy is achieved or an iteration limit is reached.

To be more specific, consider the two-point BVP for a coupled system of  $n$  first-order ODEs

$$\begin{aligned}\frac{dy}{dt} &= f(t, y(t)) \\ y_i(a) &= \alpha_i, \quad i = 1, 2, \dots, m_1 \\ y_{m_1+j}(b) &= \beta_j, \quad j = 1, 2, \dots, m_2.\end{aligned}\tag{1}$$

The vector  $y$  contains the  $n$  unknown functions of the independent variable  $t$ . The unknown functions are ordered so that the first  $m_1$  ( $0 < m_1 < n$ ) components of  $y$  have first-kind boundary conditions at  $t = a$  (*viz.*, (1)<sub>2</sub>). The remaining  $m_2 := n - m_1$  components of the solution have first-kind boundary conditions specified at a second point,  $t = b$ .<sup>1</sup> (Note that if  $m_2 = 0$ , then (1) is an initial value problem.)

The shooting method seeks to identify a vector of parameters  $s \in \mathbb{R}^{m_2}$  so that the solution, denoted by  $y(t; s)$ , to the initial value problem

$$\begin{aligned}\frac{dy}{dt} &= f(t, y(t; s)) \\ y_i(a; s) &= \alpha_i, \quad i = 1, 2, \dots, m_1 \\ y_{m_1+j}(a; s) &= s_j, \quad j = 1, 2, \dots, m_2\end{aligned}\tag{2}$$

agrees with the solution to (1). Note that (2) is simply (1) with the boundary conditions at  $t = b$  replaced with unknown initial conditions at  $t = a$ . To determine the correct initial values, consider the “objective function”  $F$  with components

$$F_j(s) := y_{m_1+j}(b; s) - \beta_j, \quad j = 1, 2, \dots, m_2.$$

Then [1, p. 476], (1) is solvable if and only if there exists  $s \in \mathbb{R}^{m_2}$  so that  $F(s) = 0$ .

The success of this process depends primarily on the iterative procedure used to construct a sequence of parameter vectors that converges to a zero of  $F$ . While

---

<sup>1</sup>The Maple procedure `shoot` supports nonlinear boundary conditions at  $t = b$ . In particular, (1)<sub>3</sub> can be replaced with  $r_j(y(b)) = 0$ ,  $j = 1, 2, \dots, m_2$  where each  $r_j$  is a differentiable function of  $n$  variables.

---

\*Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA; E-mail: [meade@math.sc.edu](mailto:meade@math.sc.edu); WWW URL: <http://www.math.sc.edu/~meade/>

†Department of Chemical Engineering, University of South Carolina, Columbia, SC 29208, USA; E-mail: [bala@sun.che.sc.edu](mailto:bala@sun.che.sc.edu)

‡Department of Chemical Engineering, University of South Carolina, Columbia, SC 29208, USA; E-mail: [rew@sun.che.sc.edu](mailto:rew@sun.che.sc.edu)

any numerical root-finding algorithm could be employed for this step, one step of the Newton-Raphson method is most commonly used. That is, given an initial guess  $s^0 \in \mathbb{R}^{m_2}$ , define a sequence of initial conditions  $\{s^k\}$  by

$$s^{k+1} := s^k - (\nabla F(s^k))^{-1} F(s^k)$$

for all  $k \geq 0$ . To implement this, note that the vector  $F(s^k)$  is directly available from the solution of (2), but the Jacobian matrix  $\nabla F(s^k)$  requires the values of  $\frac{\partial y_{m_1+i}}{\partial s_j}(b; s^k)$  for all  $i, j = 1, 2, \dots, m_2$ . These values can be obtained by solving the  $n$  IVPs in (2) together with the  $nm_2$  sensitivity equations ([2, p. 226], [3, pp. 54–58]):

$$\frac{d}{dt} \left( \frac{\partial y_i}{\partial s_j} \right) = \frac{\partial f}{\partial y_i} \frac{\partial y_i}{\partial s_j}, \quad i = 1, 2, \dots, n, \quad j = 1, \dots, m_2$$

with corresponding initial conditions

$$\begin{aligned} \frac{\partial y_i}{\partial s_j} &= 0, & \text{for all } i = 1, \dots, m_1, \quad j = 1, 2, \dots, m_2 \\ \frac{\partial y_{m_1+i}}{\partial s_j} &= \delta_{ij}, & \text{for all } i, j = 1, \dots, m_2 \end{aligned}$$

The above algorithm is known as the *simple*, or *single*, *shooting method*. While this method is effective for many problems, three specific problems should be mentioned [4]. Assume (1) has a unique solution. There is no guarantee that the initial value problem (2) has a solution on the interval  $[a, b]$  for all  $s \in \mathbb{R}^{m_2}$ . Even if (2) does have a solution on  $[a, b]$ , the problem may be stiff. In such a case the solution at  $t = b$  may be so inaccurate as to make the results of the Newton-Raphson step meaningless. When the accuracy of the solution at  $t = b$  is known with sufficient accuracy, the local convergence of the Newton-Raphson step may prevent the iterations from converging to a solution of the original boundary value problem (1). This difficulty can be addressed by replacing the Newton-Raphson step with another iterative solver with improved convergence properties (*e.g.*, modified Newton's method). The *multiple*, or *parallel*, *shooting method* addresses the other difficulties. As these problems are generally more pronounced as  $b - a$  increases, it seems appropriate to consider partitioning the interval into  $N$  subintervals. Then, using compatibility conditions between the subintervals, a well-posed IVP is obtained on each subinterval.

The combination of Maple's facilities for symbolic manipulation and numerical solution of IVPs provide an excellent environment for the translation of the simple shooting method into the Maple programming language. The syntax for this procedure, called `shoot`, closely follows the syntax of `dsolve`. In particular, the data structures returned by `shoot` are identical to the ones returned by `dsolve/numeric`. Thus, all the techniques and tools for manipulating Maple's numerical solutions

of differential equations, *e.g.* `plots[odeplot]`, can be used to interpret the results from `shoot`. (The source code for `shoot`, including on-line help documentation, is available upon request from the first author.)

Other numerical methods for the solution of two-point boundary value problems have also been developed on other platforms. One of the most well-known is the FORTRAN subroutine `COLSYS` (available from Netlib). The numerical method used in this routine is collocation of B-splines at Gaussian points [5]. Finite difference methods can be easily implemented using MATLAB's `fsolve` command.

The three examples discussed in this paper provide samples of how simple shooting, in particular `shoot`, can be applied to the analysis of boundary value problems that are encountered in chemical engineering.

## Laminar boundary-layer flow past a flat plate

Consider a fluid stream with velocity  $u_0$  and kinematic viscosity  $\nu$  in which a thin plate is inserted parallel with the fluid flow. Determining the velocity of the fluid in the region close to the plate is the *Blasius problem* [6, p. 233]. Assuming the flow is steady, incompressible, and Newtonian, the plate is infinitely wide, and neglecting buoyancy, the equations of motion and continuity are:

```
> restart;
> alias( U=u(x,y), V=v(x,y) );
> PDE:={ U*diff(U,x)+V*diff(U,y)-nu*diff(U,y$2)=0,
>         diff(U,x)+diff(V,y)=0 };
```

$$\begin{aligned} PDE := & \left\{ U \left( \frac{\partial}{\partial x} U \right) + V \left( \frac{\partial}{\partial y} U \right) \right. \\ & \left. - \nu \left( \frac{\partial^2}{\partial y^2} U \right) = 0, \right. \\ & \left. \left( \frac{\partial}{\partial x} U \right) + \left( \frac{\partial}{\partial y} V \right) = 0 \right\} \end{aligned}$$

where  $u$  and  $v$  are the  $x$ - and  $y$ -components of the fluid velocity. The boundary conditions consist of the “no-slip” conditions on the boundary of the plate:  $u(x, 0) = v(x, 0) = 0$ , and the free stream-merge condition  $\lim_{y \rightarrow \infty} u(x, y) = u_0$ , for all  $x > 0$ .

A similarity transformation can be used to reduce this parabolic system of PDEs to a single ODE. This can be done by choosing the dimensionless similarity variable to be:

```
> simsubs:=eta(x,y)=y*sqrt(u[0]/nu/x/2);
simsubs := eta(x,y) = 1/2 y sqrt(2) sqrt(u0/nu x)
```

The corresponding nondimensional stream function for the flow is:

```
> stream:=psi(x,y)=sqrt(2*nu*x*u[0])*f(eta(x,y));
```

$$\begin{aligned} stream &:= \\ \psi(x,y) &= \sqrt{2} \sqrt{\nu x u_0} f(\eta(x,y)) \end{aligned}$$

Thus, the velocities can be expressed as:

```
> Usubs:=U=diff(rhs(stream),y);
```

$$\begin{aligned} Usubs &:= U = \sqrt{2} \sqrt{\nu x u_0} \\ D(f)(\eta(x,y)) &\left( \frac{\partial}{\partial y} \eta(x,y) \right) \end{aligned}$$

```
> Vsubs:=V=-diff(rhs(stream),x);
```

$$\begin{aligned} Vsubs &:= V = -\frac{1}{2} \frac{\sqrt{2} f(\eta(x,y)) \nu u_0}{\sqrt{\nu x u_0}} \\ &\sqrt{2} \sqrt{\nu x u_0} D(f)(\eta(x,y)) \\ &\left( \frac{\partial}{\partial x} \eta(x,y) \right) \end{aligned}$$

Substituting the stream function representations of the velocities into the PDEs is tedious to complete by hand. Fortunately, this is exactly one of Maple's strengths:

```
> ODE:=simplify(subs(Usubs,Vsubs,simsubs,PDE));
```

$$\begin{aligned} ODE &:= \left\{ 0 = 0, -\frac{1}{2} u_0^2 \left( \right. \right. \\ &D^{(2)}(f) \left( \frac{1}{2} y \sqrt{2} \sqrt{\frac{u_0}{\nu x}} \right) \\ &f \left( \frac{1}{2} y \sqrt{2} \sqrt{\frac{u_0}{\nu x}} \right) \nu x \sqrt{\frac{u_0}{\nu x}} \\ &+ \sqrt{\nu x u_0} D^{(3)}(f) \left( \frac{1}{2} y \sqrt{2} \sqrt{\frac{u_0}{\nu x}} \right) \\ &\left. \left. \right) / \left( \nu x^2 \sqrt{\frac{u_0}{\nu x}} \right) = 0 \right\} \end{aligned}$$

The trivial fulfillment of the continuity equation is evident in this result. However, the first equation is not so readily identified. To simplify this further, note that each argument to  $f$ , and its derivatives, is simply  $\eta$ . To force this simplification,

```
> simsubs2:=solve(subs(eta(x,y)=eta,simsubs),{y});
```

$$simsubs2 := \left\{ y = \frac{\eta \sqrt{2}}{\sqrt{\frac{u_0}{\nu x}}} \right\}$$

```
> ODE:=simplify(subs(simsubs2,ODE),symbolic);
```

$$\begin{aligned} ODE &:= \left\{ -\frac{1}{2} u_0^2 \right. \\ &(D^{(2)}(f)(\eta) f(\eta) + D^{(3)}(f)(\eta)) \\ &\left. / x = 0, 0 = 0 \right\} \end{aligned}$$

It is now easy to identify the Blasius equation

$$f'''(\eta) + f(\eta) f''(\eta) = 0, \quad \eta > 0. \quad (3)$$

The conversion of the boundary conditions can be done by inspection. The resulting conditions are  $f(0) = f'(0) = 0$  and  $\lim_{\eta \rightarrow \infty} f'(\eta) = 1$ .

The "boundary condition" at  $\eta = \infty$  presents a problem. However, a simple asymptotic analysis shows that solutions at  $\eta = 10$  are safely in the far-field. The problem now takes the form of a third-order two-point boundary value problem for  $f$  on  $[0, 10]$ . The reformulation as a first-order system is straightforward; let  $g := f'$  and  $h := f''$ , then

$$\begin{aligned} f' &= g, & f(0) &= 0 \\ g' &= h, & g(0) &= 0 \\ h' &= -fh, & h(0) &= \beta \end{aligned} \quad (4)$$

where  $\beta$  is the control parameter. The objective is to find  $\beta$  so that the solution to (4) satisfies the boundary condition  $g(10) = 1$ . Since there is only one boundary condition at  $\eta = 10$  we have  $m_2 = 1$  and  $s = \beta$ . The sensitivity equations that are obtained from (4), together with their accompanying initial values, are:

$$\begin{aligned} f'_\beta &= g_\beta, & f_\beta(0) &= 0 \\ g'_\beta &= h_\beta, & g_\beta(0) &= 0 \\ h'_\beta &= -f_\beta h_\beta - f h_\beta, & h_\beta(0) &= 1 \end{aligned} \quad (5)$$

where, for example,  $f_\beta(t; \beta) := \frac{\partial f}{\partial \beta}(t; \beta)$ . Now, assume that an approximate solution to (4) and (5) has been computed with  $s = s^k$ , i.e.  $\beta = \beta^k$ . Then, assuming  $|g(10; \beta^k) - 1|$  is not sufficiently small, the next iteration will be made with

$$\beta^{k+1} := \beta^k - \frac{g(10; \beta^k) - 1}{g_\beta(10; \beta^k)}.$$

While it is important to understand how `shoot` obtains its approximations, it is also a tremendous advantage to use Maple to both compute automatically, and symbolically, the sensitivity equations and iteratively solve the combined system of IVPs and take one step of the Newton-Raphson method until the boundary conditions are satisfied within a prescribed tolerance.

Prior to preparing this problem for solution by `shoot`, it is necessary to load the definition of this procedure:

```
> read shoot;
```

Now, define the dependent variables for the first-order system:

```
> FNS:={ f(eta), g(eta), h(eta) };
```

and the differential equations which they satisfy:

```
> ODE:={ diff(f(eta),eta)=g(eta),
>         diff(g(eta),eta)=h(eta),
>         diff(h(eta),eta)=-f(eta)*h(eta) };
```

The initial condition for  $f''$  is unknown, and the boundary condition at  $\eta = 10$  is the control for our iterations:

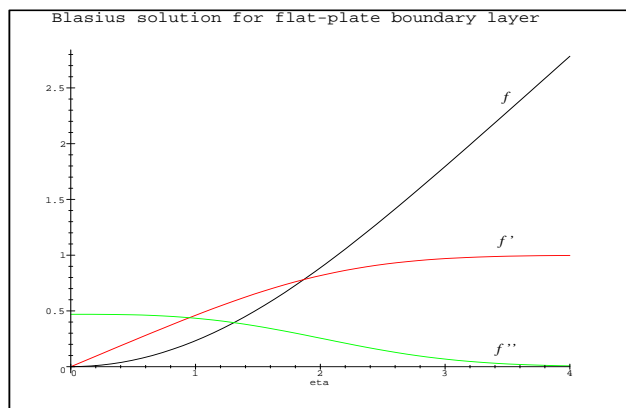
```
> IC:={ f(0)=0, g(0)=0, h(0)=beta };
> BC:=g(10)=1;
```

We take the first shot with  $\beta = 0$ , and iterate until the boundary condition is satisfied to six decimal places:

```
> S:=shoot( ODE union IC, FNS, BC,beta=0,
>          bcerr=Float(5,-7) );
```

This solution can be analyzed using any of the standard Maple tools for manipulating the numerical solution of a differential equation. A common place to begin is a plot of the solution, including labels to distinguish the different curves:

```
> with(plots):
> C:=odeplot( S, [ [eta,f(eta)],
>                 [eta,g(eta)], [eta,h(eta)] ],0..4):
> subs( [g='f''', h='f'''],
>       subs( [seq( op(0,lhs(F))=rhs(F(3.5)),
>                 F=subsop(1=NULL,S) ) ] ) );
> L:=textplot(
>   subs( eta=3.5,
>         map(F->[eta,rhs(F)+1/10,lhs(F)],"")),
>   font=[TIMES,ITALIC,18], align=ABOVE ):
> display( {C,L}, labels=['eta','f'''],
>   title='Blasius solution for flat-plate
>         boundary layer' );
```



With increasing distance from the leading edge of the plate in the downstream direction the thickness  $\delta$  of the retarded boundary layer increases continuously as increasing quantities of fluid become affected. In the boundary layer the velocity of the fluid increases from

zero at the wall (no slip) to its full value which corresponds to external frictionless flow. The velocity reaches 99% of its bulk value when  $f'(\eta) = 0.99$ :

```
> fp:=subs( S, g(eta) );
> eta['99%']:=fsolve( 'fp(eta)'=0.99, 'eta' );
> eta99% := 3.471887111
```

The corresponding boundary layer thickness is:

```
> delta['99%']:=subs(eta=eta['99%'],
>                   combine(rhs(op(simsubs2)))));
> delta99% := 3.471887111 * (sqrt(2) * sqrt(nu * x * u0)) / u0
```

As is now evident, the thickness of the boundary layer decreases with decreasing viscosity.

The shear stress,  $\tau = \mu \frac{\partial u}{\partial y}$ , is:

```
> tau:=mu*simplify(
>   diff(subs(simsubs,rhs(Usubs)),y));
```

$$\tau := \frac{1}{2} \mu \sqrt{\nu x u_0} D^{(2)}(f) \left( \frac{1}{2} y \sqrt{2} \sqrt{\frac{u_0}{\nu x}} \right) u_0 \sqrt{2} / (\nu x)$$

Note that a large velocity gradient across the flow creates considerable shear stress in the boundary layer. The wall shear stress is:

```
> tau[w]:=subs( (D@@2)(f)(0)=subs(S,h(eta))(0),
>               subs( y=0, tau ) );
```

$$\tau_w := .2347998643 \frac{\mu \sqrt{\nu x u_0} u_0 \sqrt{2}}{\nu x}$$

The analysis can be continued to obtain an understanding of parameters such as the displacement thickness and drag coefficient.

## Porous Catalyst

Prediction of the diffusion and reaction in a porous catalyst pellet is another important problem in chemical engineering. The reaction under consideration is  $A \rightarrow B$  which occurs inside the pellet. Mass balance and conservation of energy on the spherical pellet give:

$$\begin{aligned} \frac{D_e}{r^2} \frac{d}{dr} \left( r^2 \frac{dc}{dr} \right) - R(c, T) &= 0 \\ \frac{k_e}{r^2} \frac{d}{dr} \left( r^2 \frac{dT}{dr} \right) + (-\Delta H_R) R(c, T) &= 0 \end{aligned} \quad (6)$$

where  $c$  is the concentration of  $A$ ,  $D_e$  is the effective diffusivity,  $R(c, T)$  the reaction rate expression as a function of concentration  $c$  and temperature  $T$ ,  $k_e$  the effective thermal conductivity and  $-\Delta H_R$  the heat of the reaction. Due to radial symmetry about the center of the pellet:

$$\frac{dc}{dr} = 0, \quad \frac{dT}{dr} = 0 \quad \text{at } r = 0. \quad (7)$$

On the surface of the pellet the quantities are at their bulk values:

$$c = c_0, \quad T = T_0, \quad \text{at } r = R. \quad (8)$$

For a simple first-order irreversible reaction

$$R(c, T) = k_0 c \exp\left(-\frac{Q}{RT}\right);$$

the relationship between the reactant concentration and temperature at any point in the catalyst is given by [7]:

$$\Delta T = T - T_0 = \frac{-\Delta H_R D_e}{k_e} (c_0 - c).$$

Introducing dimensionless variables

$$\gamma := \frac{Q}{RT_0}, \quad \beta := \frac{c_0(-\Delta H_R)D_e}{k_e T_0}, \quad y := \frac{c}{c_0},$$

the reaction rate can be expressed as

$$R(y, T) = k_0 c_0 y \exp\left(\frac{\gamma\beta(1-y)}{1+\beta(1-y)}\right).$$

Substituting these into (6) and the boundary conditions (7) and (8) leads to the two-point BVP ([7])

$$\begin{aligned} \frac{d^2 y}{dx^2} + \frac{2}{x} \frac{dy}{dx} &= \phi^2 y \exp\left(\frac{\gamma\beta(1-y)}{1+\beta(1-y)}\right), \\ \frac{dy}{dx}(0) &= 0, \\ y(1) &= 1 \end{aligned}$$

where  $\phi := R\sqrt{\frac{k_0}{D_e}}$  and  $x := \frac{r}{R}$ .

When expressed as a first-order system, this problem is easily solved by `shoot` for specific values of  $\gamma$ ,  $\beta$ , and  $\phi$ . For example:

```
> restart;
> read shoot;
> ODE:={ diff(y(x),x) = z(x),
>         diff(z(x),x) = -2/x*z(x)
>         + phi^2*y(x)
>         *exp(gamma*beta*(1-y(x))
>         /(1+beta*(1-y(x))) ) };
> FNS:={ y(x), z(x) };
> IC:={ y(0)=alpha, z(0)=0 };
> BC:={ y(1)=1:
> COEF:=[ gamma=30, beta=2/5, phi=3/10 ]:
> S1:=shoot( subs(COEF,ODE) union IC, FNS, BC,
>             alpha=1, value=array([0,1]) );
```

```
SI :=
[[ x y(x) z(x) ]]
[
[0, .9829180602000001, 0]
[1., 1.000000000280384,
.03231087397210381 ]]
```

The effectiveness factor,  $\eta$ , for the reaction is the ratio

of the average reaction rate with diffusion to the average reaction rate when the reaction rate is evaluated at the bulkstream (or boundary) values ([3, pp. 58–62], [8, p. 83]). Here is one way in which this quantity can be computed from the results returned by `shoot`:

```
> eta:=3/phi^2*D(c)(1) = subs(COEF,3/phi^2*dcd1);
eta := 3 * D(c)(1) / phi^2 = 100 / 3 * dcd1
> dcd1:=S1[2,1][2,3];
dcd1 := .03231087397210381
> 'eta'=rhs(eta);
eta = 1.077029132
```

As is well-known [8, pp. 87–88], for a given value of  $\phi$ , the effectiveness factor can be multiple-valued. The other values can be obtained by starting the shooting method with different values of the control parameter:

```
> Sh:=shoot( subs(COEF,ODE) union IC, FNS, BC,
>             alpha=1/2, value=array([0,1]) );
> dcd1:=Sh[2,1][2,3];
> 'eta'=rhs(eta);
```

```
Sh :=
[[ x y(x) z(x) ]]
[
[0, .1442192834000000, 0]
[1., .99999999503557,
.3250550528505473 ]]
```

```
eta = 10.83516843
```

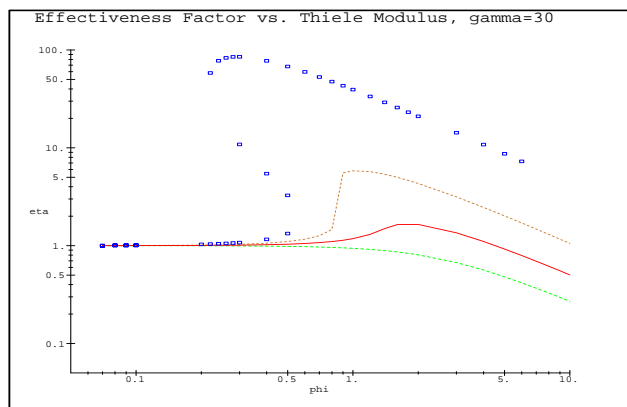
```
> S0:=shoot( subs(COEF,ODE) union IC, FNS, BC,
>             alpha=0, value=array([0,1]) );
> dcd1:=S0[2,1][2,3];
> 'eta'=rhs(eta);
```

```
S0 :=
[[ x y(x) z(x) ]]
[
[0, .1096674904000000 10^-5, 0]
[1., .999999998529494,
2.554522460710149 ]]
```

```
eta = 85.15074869
```

The effectiveness factor has been computed for a wide range of values for  $\gamma$ ,  $\beta$ , and  $\phi$ . A sample of the results, for  $\gamma = 30$ , are displayed in the following plot. The four datasets displayed in this plot correspond to  $\beta = 0.4$  (box – top curve),  $\beta = 0.2$  (dotted line – second curve from top),  $\beta = 0.1$  (solid line – second curve from bottom), and  $\beta = 0$  (dashed line – bottom curve). The

multiple values for the exothermic catalyst ( $\beta = 0.4$ ) were obtained using different initial guesses for the control parameter ( $\alpha = 0$ ,  $\alpha = 0.5$ , and  $\alpha = 1$ ).



## Infinite Rotating Disk

Consider the steady fluid flow generated when the infinite plane  $z = 0$ , immersed in a Newtonian viscous fluid, rotates about the axis  $r = 0$  with a constant angular velocity  $\omega$ . The viscous drag of the rotating surface creates a swirling flow toward the disk. The motion is characterized in terms of the pressure,  $p$ , and the three components of the velocity,  $v^r$ ,  $v^\theta$ ,  $v^z$ , in cylindrical coordinates. The radial symmetry of this problem eliminates  $\theta$  as an independent variable. Thus, with  $\rho$  and  $\nu$  denoting the density and kinematic viscosity of the fluid and writing partial derivatives as subscripts, the equations of continuity and conservation of momentum reduce to [6]:

$$\begin{aligned}
 & \frac{1}{r}(rv^r)_r + (v^z)_z \\
 & = 0 \\
 & v^r(v^r)_r + v_z(v^r)_z - \frac{1}{r}(v^\theta)^2 \\
 & = -\frac{1}{\rho}p_r + \nu \left( (v^r)_{rr} + \frac{1}{r}(v^r)_r + (v^r)_{zz} - \frac{v^r}{r^2} \right) \\
 & v^r(v^\theta)_r + v^z(v^\theta)_z + \frac{1}{r}v^r v^\theta \\
 & = \nu \left( (v^\theta)_{rr} + \frac{1}{r}(v^\theta)_r + (v^\theta)_{zz} - \frac{v^\theta}{r^2} \right) \\
 & v^r(v^z)_r + v^z(v^z)_z \\
 & = -\frac{1}{\rho}p_z + \nu \left( (v^z)_{rr} + \frac{1}{r}(v^z)_r + (v^z)_{zz} \right).
 \end{aligned} \tag{9}$$

The boundary conditions are chosen (for all  $r > 0$ ) to enforce no slippage at the interface with the disk:

$$v^r(r, 0) = v^z(r, 0) = 0,$$

$$\begin{aligned}
 v^\theta(r, 0) & = r\omega, \\
 p(r, 0) & = 0
 \end{aligned} \tag{10}$$

and no non-axial viscous effect in the far-field:

$$\begin{aligned}
 \lim_{z \rightarrow \infty} v^r(r, z) & = 0 \\
 \lim_{z \rightarrow \infty} v^\theta(r, z) & = 0 \\
 \lim_{z \rightarrow \infty} (v^z)_z(r, z) & = 0.
 \end{aligned} \tag{11}$$

Similarity solutions to this equation were found by Kármán [9], who noted that each of  $\frac{v^r}{r}$ ,  $\frac{v^\theta}{r}$ ,  $\frac{v^z}{r}$ , and  $p$  depends only on the distance from the disk,  $z$ . The system of PDEs reduces to a system of ODEs with the introduction of  $z^* = z\sqrt{\frac{\omega}{\nu}}$  as the dimensionless independent variable for the dimensionless functions  $F$ ,  $G$ ,  $H$ , and  $P$  defined according to

$$\begin{aligned}
 v^r & = r\omega F(z^*), \\
 v^\theta & = r\omega G(z^*), \\
 v^z & = \sqrt{\omega\nu}H(z^*), \\
 p & = \rho\omega\nu P(z^*).
 \end{aligned} \tag{12}$$

The substitution of (12) into (9), (10), and (11) can be completed in a manner analogous to that used in the Blasius problem. These steps are omitted here in the interest of space; the resulting BVP is:

$$\begin{aligned}
 H' & = -2F \\
 F'' & = F^2 - G^2 + F'H \\
 G'' & = 2FG + HG' \\
 P' & = -HH' + H'' \tag{13} \\
 F(0) = H(0) = P(0) & = 0 \\
 G(0) & = 1 \\
 \lim_{z^* \rightarrow \infty} F(z^*) = \lim_{z^* \rightarrow \infty} G(z^*) & = 0.
 \end{aligned}$$

Note that (13) can be solved as a two-point BVP for  $F$ ,  $G$ ,  $H$ . Once this solution is known, the pressure equation can be integrated to yield  $P = -\frac{1}{2}H^2 + H'$ . In addition, note that any physically realistic solution must have  $F, G, P > 0$  and  $H < 0$  for all  $z^*$  [6, p. 164].

Assuming the boundary condition at  $z^* = \infty$  can be applied at a finite distance from the disk, `shoot` can be used to obtain an approximate solution to this system. First, introduce the first derivatives of  $F$  and  $G$  as new dependent variables:

```

> restart;
> read shoot;
> with(plots):
> FNS:={ F(Z), G(Z), H(Z), Fp(Z), Gp(Z) };
```

and reformulate the system as a first-order system of ODEs:

```

> ODE:={diff(H(Z),Z) = -2*F(Z),
>        diff(F(Z),Z) = Fp(Z),
>        diff(Fp(Z),Z) = -G(Z)^2+F(Z)^2+Fp(Z)*H(Z),
>        diff(G(Z),Z) = Gp(Z),
>        diff(Gp(Z),Z) = 2*F(Z)*G(Z)+H(Z)*Gp(Z)}:
    
```

with initial conditions:

```

> IC:={ F(0)=0, G(0)=1, H(0)=0,
>        Fp(0)=alpha, Gp(0)=beta }:
    
```

and boundary conditions (again,  $z^* = 10$  can be shown to be in the far-field):

```

> BC:={ F(10)=0, G(10)=0 }:
    
```

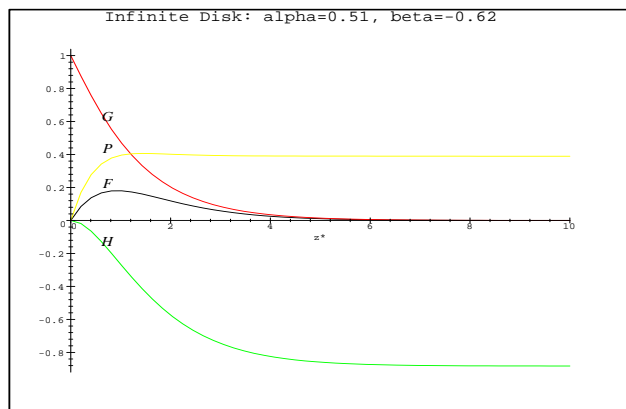
Note that, as in the first example, the boundary condition at infinity has been moved to a finite position. The value  $z^* = 10$  is chosen, as before, on the basis that this is already in the far field. (In fact, truncating the computations at  $z^* = 7$  yields essentially the same solution.) Note also that since there are two boundary conditions at the second boundary point, there are two parameters to be determined in the shooting method (*i.e.*,  $m_2 = 2$ ). This can be handled by `shoot` without modification:

```

> infolevel[shoot]:=1:
> S:=shoot( ODE union IC, FNS, BC,
>           [alpha=0.51, beta=-0.62] ):
shoot:  alpha = .51, beta = -.62
shoot:  alpha = .5100033193, beta = -.6161487475
shoot:  alpha = .5102117355, beta = -.6159070527
shoot:  alpha = .5102135910, beta = -.6159096496
    
```

```

> P:=Z->-H(Z)^2/2-2*F(Z):
> C:=odeplot(S,[ [Z,F(Z)], [Z,G(Z)],
>                [Z,H(Z)], [Z,-P(Z)] ], 0..10 ):
> subs( [seq( op(0,lhs(f))=rhs(f(3/4))),
>        f=select(has,S,{F,G,H}) ] ):
> [op("),P=subs(" , H^2+2*F)]:
> L:=textplot(
>   subs(z=3/4, map(f->[z,rhs(f)+1/45,lhs(f)],")),
>   font=[TIMES,ITALIC,18], align=ABOVE ):
> display( {C,L}, labels=[ 'z*', ' ' ],
>          title='Infinite Disk:
>          . 'alpha=0.51, beta=-0.62' );
    
```



This plot confirms that the solution satisfies the original boundary conditions at  $z^* = \infty$ . (In fact, the solution obtained when the computational domain is trun-

cated at  $x^* = 7$ .) Compare the solution in the preceding plot with the solution obtained when the starting values of the control parameters are  $\alpha = 0.50$  and  $\beta = -0.61$  – each just 0.01 less than the first attempt:

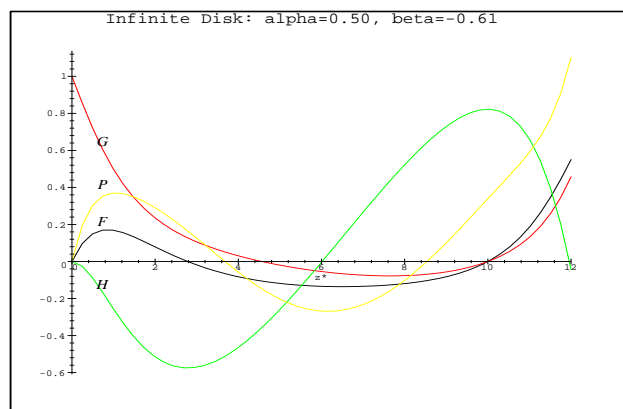
```

> S2:=shoot( ODE union IC, FNS, BC,
>            [alpha=0.5, beta=-0.61] ):
shoot:  alpha = .5, beta = -.61
shoot:  alpha = .5181506133, beta = -.5834922887
shoot:  alpha = .5178653168, beta = -.5900697906
shoot:  alpha = .5019889119, beta = -.5940268791
shoot:  alpha = .5040798081, beta = -.5927957583
shoot:  alpha = .5042719718, beta = -.5927086067
shoot:  alpha = .5042737786, beta = -.5927079110
    
```

The different values of  $\alpha$  and  $\beta$  suggest that the shooting method has returned a different solution. Is this also a solution to the problem?

```

> C:=odeplot(S2,[ [Z,F(Z)], [Z,G(Z)],
>                 [Z,H(Z)], [Z,-P(Z)] ], 0..12 ):
> subs( [seq( op(0,lhs(f))=rhs(f(3/4))),
>        f=select(has,S2,{F,G,H}) ] ):
> [op("),P=subs(" , H^2+2*F)]:
> L:=textplot(
>   subs(z=3/4, map(f->[z,rhs(f)+1/45,lhs(f)],")),
>   font=[TIMES,ITALIC,18], align=ABOVE ):
> display( {C,L}, labels=[ 'z*', ' ' ],
>          title='Infinite Disk:
>          . 'alpha=0.50, beta=-0.61' );
    
```



Note that this solution does not satisfy the aforementioned sign constraints for  $F$ ,  $G$ ,  $H$ , and  $P$  for a physical solution to (13). Furthermore, while the boundary conditions at  $z^* = 10$  are satisfied, it is extremely unlikely that this solution will satisfy the boundary conditions at  $z^* = \infty$ . This is a spurious solution [6, p. 167]. Automating the detection of spurious solutions is extremely difficult. In some instances, this difficulty can be avoided by the use of one of the other solution techniques for boundary value problems mentioned previously.

## Conclusion

In this brief article we have introduced `shoot`, a Maple implementation of the simple shooting method for the

numerical solution of two-point boundary value problems, and illustrated the application of `shoot` to assist with the analysis of three classic problems from chemical engineering. These examples demonstrate some of the potential that is available with Maple's combination of symbolic, numeric, and graphic computation.

## Acknowledgment

The authors would like to thank the referees for their useful comments. In addition, we would like to recognize our colleague, John Weidner, for his proofreading and suggestions.

This project is a result of the interdisciplinary interactions supported by the Center for Electrochemical Engineering (URL: <http://www-cee.che.sc.edu/>), which has been funded by the State of South Carolina. In addition, the first author was partially supported by the National Science Foundation under grant DMS-9404488.

## References

- [1] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis* Springer-Verlag, 1980.
- [2] Yonathan Bard, *Nonlinear Parameter Estimation*, Academic Press, 1974.
- [3] Mark E. Davis, *Numerical Methods and Modeling for Chemical Engineers*, John Wiley & Sons, Inc., 1984.
- [4] Uri M. Ascher, Robert M. M. Mattheij, and Robert D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, SIAM, 1995.
- [5] U. Ascher, J. Christiansen and R. D. Russell, *COLSYS - a collocation code for boundary value problems*, Proc. Conf. for Codes for BVPs in ODEs, Houston, Texas, 1978.
- [6] Frank M. White, *Viscous Fluid Flow*, 1st ed., McGraw-Hill, Inc., 1974.
- [7] P. B. Wiesz and J. S. Hicks, *The behaviour of porous catalyst particles in view of internal mass and heat diffusion effects*, *Chemical Engineering Science*, **17**, pp. 265–275, 1962.
- [8] Bruce A. Finlayson, *Nonlinear Analysis in Chemical Engineering*, McGraw-Hill, 1980.
- [9] T. von Kármán, "Über laminare und turbulente Reibung," *Z. Angew. Math. Mech.*, vol. 1, pp. 233–252, 1921.

## Biographies

### Douglas B. Meade

Douglas B. Meade received a Ph.D. in Mathematics from Carnegie Mellon University in 1989, then spent two years as a Research Assistant Professor in the Center of Applied Mathematics at Purdue University. Since 1991 he has been an Assistant Professor in the Department of Mathematics at the University of South Carolina. Research interests are presently directed towards numerical methods for PDEs, in particular, wave propagation on unbounded domains and nonlinear reaction–diffusion equations. Maple is used extensively in this research. Dr. Meade has also incorporated student usage of Maple into a variety of undergraduate and graduate courses. These efforts have been generously supported by the Office of the Provost, including funding (joint with Professor White) from the Lilly Endowment Teaching Fellows Program.

### Bala S. Haran

Bala S. Haran received a Bachelors of Technology in Electrochemical Engineering from Central Electrochemical Research Institute, India in 1989. Since 1990 he has been a doctoral student in the Department of Chemical Engineering at the University of South Carolina. Research interests lie in the field of Electrochemical Engineering, in particular mathematical modeling of electro-kinetic processes, batteries and corrosion.

### Ralph E. White

Ralph E. White received his B.S. in Engineering from the University of South Carolina in 1971 and his M.S. and Ph.D. degrees from the University of California at Berkeley in 1973 and 1977, respectively. Dr. White joined the faculty at Texas A&M University (TAMU) in 1977 and was promoted to Associate Professor and Professor in 1981 and 1985, respectively. He served at TAMU as the Associate Head of the Department of Chemical Engineering from 1990-1992. Dr. White joined the faculty at the University of South Carolina in January 1993, and is now serving as the Chairman of the Department of Chemical Engineering and as a Westinghouse Distinguished Scientist. In addition, he is Director of the state authorized Center for Electrochemical Engineering in the Department of Chemical Engineering. Dr. White has also been active in research, having published over 130 refereed journal articles.