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

Douglas B. Meade
Department of Mathematics
University of South Carolina
Columbia, SC 29208, USA;
E-mail: meade@math.sc.edu
WWW URL: http://www.math.sc.edu/~meade/

Bala S. Haran
Department of Chemical Engineering
University of South Carolina
Columbia, SC 29208, USA
E-mail: bala@sun.che.sc.edu

Ralph E. White
Department of Chemical Engineering
University of South Carolina
Columbia, SC 29208, USA
E-mail: rew@sun.che.sc.edu

>

README: upgrade to Maple 6 from Maple V Release 3

>

This worksheet, and shoot, were originally created in Maple V Release 3. The update to Release 5, and Maple 6, has required a complete reimplementation of shoot. The worksheet is also significantly modified in some places. However, the content and functionality are as close as possible to that of the original.

Please let me know if you have any problems or questions. I also would like to hear about any successes that you have using shoot.

Thank you!

Douglas B. Meade
meade@math.sc.edu

>

Introduction

>

One of the strengths of Maple is its ability to provide a wide variety of information about solutions to 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

$$\begin{aligned} \frac{\partial}{\partial t}\mathrm{y}(t) &= \mathrm{f}(t, \mathrm{y}(t)) \\ y_i(a) &= \alpha_i\,, \quad i = 1\,..\,m_1 \\ y_{m_1+j}(b) &= \beta_j\,, \quad j = 1\,..\,m_2 \end{aligned} \qquad (1)$$

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 $t = a$. The remaining $m_2 := n - m_1$ components of the solution have first-kind boundary conditions specified at a second point, $t = b$. The Maple procedure shoot supports nonlinear boundary conditions at $t = b$. In particular, $(1)_3$ can be replaced with $r_j(\mathrm{y}(b)) = 0,\ j = 1\,..\,m_2$, where each r_j is a differentiable function of n variables. The examples discussed in this paper all use $r_j(\mathrm{y}(b)) := y_{m_1+j}(b) - \beta_j$. (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 R^{m_2} so that the solution, denoted by $\mathrm{y}(t, s)$, to the initial value problem

$$\begin{aligned} \frac{\partial}{\partial t}y &= \mathrm{f}(t, \mathrm{y}(t, s)) \\ y_i(a, s) &= \alpha_i\,, \qquad i = 1\,..\,m_1 \\ y_{m_1+j}(a, s) &= s_j\,, \qquad j = 1\,..\,m_2 \end{aligned}$$

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

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

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

>

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 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 R^{m_2}, define a sequence of initial conditions $\{s^k\}$ by

$$s^{(k+1)} := s^k - \mathrm{JF}(s^k)^{(-1)}\,\mathrm{F}(s^k)$$

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

$$\frac{\partial}{\partial t}\left(\frac{\partial}{\partial s_j} y_i(t)\right) = \left(\frac{\partial}{\partial y_i} f\right)\left(\frac{\partial}{\partial s_j} y_i\right), \quad i = 1 .. n,\ j = 1 .. m_2$$

with corresponding initial conditions

$$\frac{\partial}{\partial s_j} y_i = 0 \ , \qquad \text{for all } i = 1 .. m_1, j = 1 .. m_2$$

$$\frac{\partial}{\partial s_j} y_{m_1+j} = \delta_{ij}, \qquad \text{for all } i, j = 1 .. m_2$$

>

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 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 and Broyden'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 ν 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 };
```

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

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: $\mathrm{u}(x, 0)=\mathrm{v}(x, 0)=0$, and the free stream-merge condition $\lim_{y \to \infty} \mathrm{u}(x, y) = u_0$, 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);
```

$$simsubs := \eta(x, y) = \frac{1}{2} y \sqrt{2} \sqrt{\frac{u_0}{\nu x}}$$

The corresponding nondimensional stream function for the flow is:

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

$$stream := \psi(x, y) = \sqrt{2} \sqrt{\nu x u_0} \, \mathrm{f}(\eta(x, y))$$

Thus, the velocities can be expressed as:

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

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

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

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

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));
```

$$ODE := \left\{-\frac{1}{2}\frac{u_0^{\,2}\left((\mathrm{D}^{(2)})(f)\left(\frac{1}{2}y\sqrt{2}\sqrt{\frac{u_0}{\nu\,x}}\right)\mathrm{f}\left(\frac{1}{2}y\sqrt{2}\sqrt{\frac{u_0}{\nu\,x}}\right)\nu\sqrt{\frac{u_0}{\nu\,x}}\,x + \sqrt{\nu\,x\,u_0}\,(\mathrm{D}^{(3)})(f)\left(\frac{1}{2}y\sqrt{2}\sqrt{\frac{u_0}{\nu\,x}}\right)\right)}{\nu\,x^2\sqrt{\frac{u_0}{\nu\,x}}} = 0,\ 0 = 0\right\}$$

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 η. 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);
```

$$ODE := \left\{-\frac{1}{2}\frac{u_0^{\,2}\left((\mathrm{D}^{(2)})(f)(\eta)\,\mathrm{f}(\eta) + (\mathrm{D}^{(3)})(f)(\eta)\right)}{x} = 0,\ 0 = 0\right\}$$

It is now easy to identify the Blasius equation

$$\left(\frac{\partial^3}{\partial\eta^3}\mathrm{f}(\eta)\right) + \mathrm{f}(\eta)\left(\frac{\partial^2}{\partial\eta^2}\mathrm{f}(\eta)\right) = 0, \qquad \eta>0.$$

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

\>

The "boundary condition" at $\eta = \infty$ presents a problem. However, a simple asymptotic analysis shows that solutions at η=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

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

where β is the control parameter. The objective is to find β 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:

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

where, for example, $f_\beta(t, \beta) := \frac{\partial}{\partial \beta} f(t, \beta)$. Now, assume that an approximate solution to (4) and (5) has been computed with $s = s^k$, i.e., $\beta = \beta^k$. Then, assuming $\left| g(10, \beta^k) - 1 \right|$ 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:

```
> with(Shoot):
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the assigned name shoot now has a global binding
```

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:

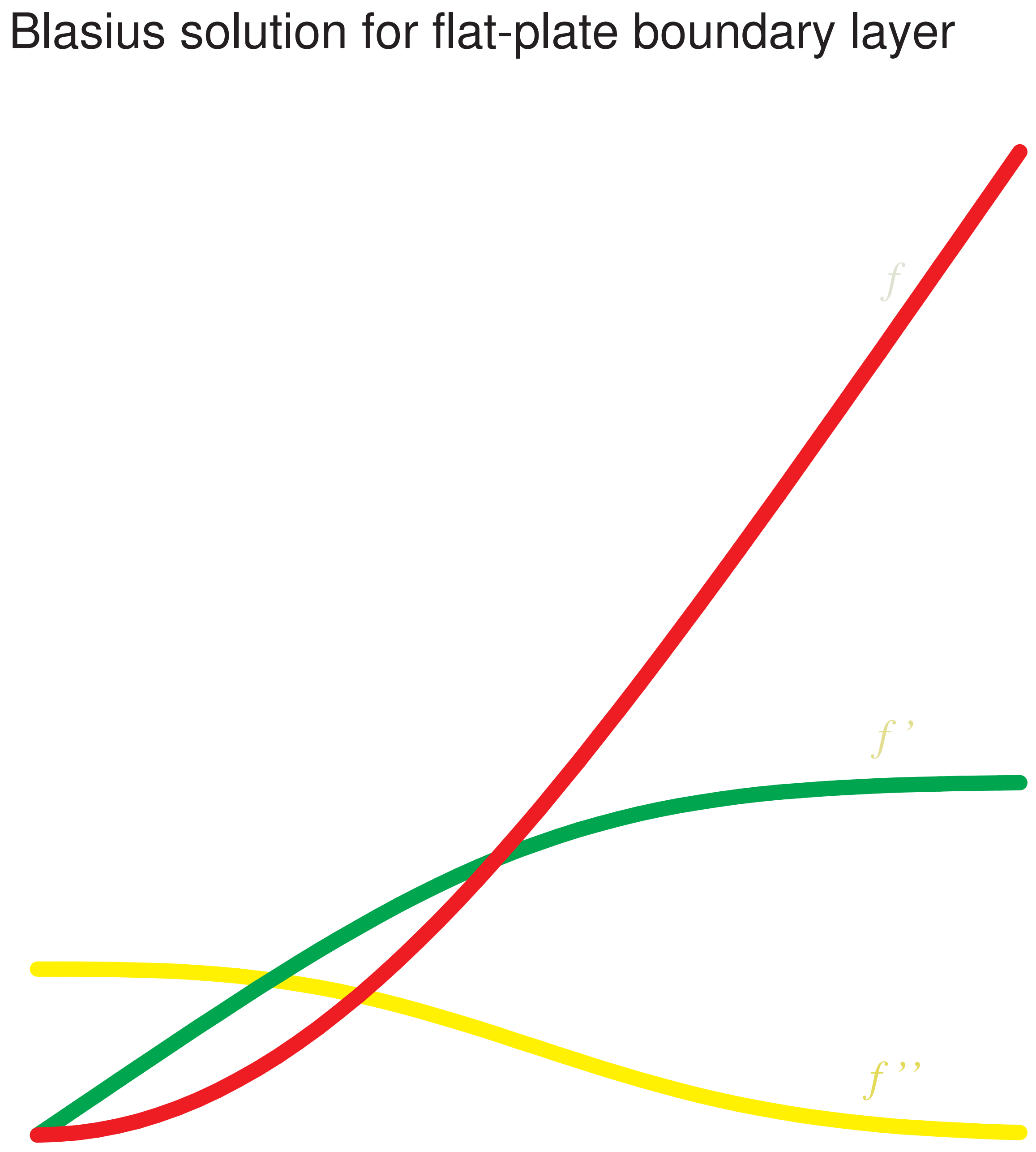
```
> IC:={ f(0)=0, g(0)=0, h(0)=beta }:
> BC:=g(10)=1:
```

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

```
> S:=shoot( ODE, IC, BC, FNS, beta=0,
>           abserr=Float(5,-7), output=listprocedure, method=taylorseries
  ):
```

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`,``],
>    title=`Blasius solution for flat-plate `||
>          `boundary layer` );
Warning, the name changecoords has been redefined
```



With increasing distance from the leading edge of the plate in the downstream direction the thickness δ 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 $f'(\eta) = .99$:

```
> fp0:=subs( S, g(eta) ):
> fp := proc(x)
>   if not type(evalf(x),`numeric`) then
>     'procname'(x);
>   else
>     fp0(x);
>   end if;
```

```
> end proc:
> eta[`99%`]:=fsolve( fp(eta)=0.99, eta=3..4 );
```

$$\eta_{99\%} := 3.471886875$$

Note that the computation of $\eta_{99\%}$ has changed significantly for Maple 6. The solution returned by shoot is obtained with method=taylorseries and the function used in fsolve must be written to avoid evaluation in cases when the argument is not numeric. But, in the end, the result is equivalent to the one reported in the original paper.

```
>
```

The corresponding boundary layer thickness is:

```
> delta[`99%`]:=subs(eta=eta[`99%`],
>                  combine(rhs(op(simsubs2))) );
```

$$\delta_{99\%} := 3.471886875\,\sqrt{2}\,\sqrt{\frac{\nu\,x}{u_0}}$$

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

```
>
```

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

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

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

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 ) );
```

$$\tau_w := .2347999942\,\frac{\mu\,\sqrt{\nu\,x\,u_0}\,u_0\,\sqrt{2}}{\nu\,x}$$

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-> B which occurs inside the pellet. Mass balance and conservation of energy on the spherical pellet give:

$$\frac{\mathrm{D}_e}{r^2}\,\frac{\partial}{\partial r}\,r^2\left(\frac{\partial}{\partial r}\,c\right) - \mathrm{R}(c, T) = 0 \qquad (6)$$

$$\frac{k_e}{r^2}\,\frac{\partial}{\partial r}\,r^2\left(\frac{\partial}{\partial r}\,T\right) + (-\Delta\,H_R)\,\mathrm{R}(c, T) = 0$$

where c is the concentration of A, D_e is the effective diffusivity, $\mathrm{R}(c, T)$ 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:

$$\frac{\partial}{\partial r} c = 0, \quad \frac{\partial}{\partial r} T = 0, \quad \text{at } r = 0. \tag{7}$$

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

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

For a simple first-order irreversible reaction

$$\mathrm{R}(c, T) = k_0\, c\, \mathbf{e}^{\left(-\frac{Q}{R\,T}\right)};$$

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

$$\Delta\, T = T - T_0 = \frac{-\Delta\, H_R\, \mathrm{D}_e}{k_e} (c_0 - c)\ .$$

>

Introducing dimensionless variables

$$\gamma := \frac{Q}{R\, T_0}, \quad \beta := \frac{c_0\,(-\Delta\, H_R)\, \mathrm{D}_e}{k_e\, T_0}, \quad y := \frac{c}{c_0},$$

the reaction rate can be expressed as

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

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

$$\frac{\partial^2}{\partial x^2} y + \frac{2}{x} \frac{\partial}{\partial x} y = \phi^2\, y\, \mathbf{e}^{\left(\frac{\gamma\,\beta\,(1-y)}{1+\beta\,(1-y)}\right)},$$

$$\frac{\partial}{\partial x} y = 0 \quad \text{at} \quad x = 0$$

$$\mathrm{y}(1) = 1$$

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

>

When expressed as a first-order system, this problem is easily solved by shoot for specific values of γ, β, and ϕ. For example:

```
> restart;
> with(Shoot):
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the assigned name shoot now has a global binding
```

Note that there is a singularity at the initial point, $x = 0$. We appear to have been fortunate that Maple V, Releases 3 and 4, handled this potential problem without any adverse effects. In Release 5 and Maple 6, Maple detects this problem and requires that the user indicate how to handle this problem and not count on our good fortune. In this case, the "solution" to this problem is to move the initial point to a point $x = x_0$ with x_0>0.

```
> 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(x0)=alpha, z(x0)=0 }:
> BC:=y(1)=1:
> COEF:=[ gamma=30, beta=2/5, phi=3/10 ]:
> x0 := 0.0001:
> 'IC' = IC;
> S1:=shoot( subs(COEF,ODE), IC, BC, FNS,
>            alpha=1, value=array([0,x0,1]) );
```

$$IC = \{ z(.0001) = 0, y(.0001) = \alpha \}$$

$$S1 := \left[\begin{array}{c} [x, y(x), z(x)] \\ \left[\begin{array}{ccc} 0 & \mathrm{Float}(1, \infty) & \mathrm{Float}(-1, \infty) \\ .0001 & .9829180602 & 0. \\ 1 & 1.000000004 & .03231087063 \end{array} \right] \end{array} \right]$$

This answer agrees with the one reported in the original paper to eight significant digits.

The effectiveness factor, η, 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*dcdr1);
```

$$\eta := 3 \frac{\mathrm{D}(c)(1)}{\phi^2} = \frac{100}{3} dcdr1$$

```
> dcdr1:=S1[2,1][3,3];
```

$$dcdr1 := .03231087063$$

```
> 'eta'=rhs(eta);
```

$$\eta = 1.077029021$$

As is well-known [8, pp. 87-88], for a given value of ϕ, 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), IC, BC, FNS,
>            alpha=1/2, value=array([0,x0,1]) );
> dcdr1:=Sh[2,1][3,3]:
> 'eta'=rhs(eta);
```

$$Sh := \left[\begin{array}{c} [x, y(x), z(x)] \\ \left[\begin{array}{ccc} 0 & \mathrm{Float}(1, \infty) & \mathrm{Float}(-1, \infty) \\ .0001 & .1442194186 & 0. \\ 1 & 1.000000005 & .3250550494 \end{array} \right] \end{array} \right]$$

$$\eta = 10.83516831$$

```
> S0:=shoot( subs(COEF,ODE), IC, BC, FNS,
>            alpha=0, value=array([0,x0,1]) );
> dcdr1:=S0[2,1][3,3]:
> 'eta'=rhs(eta);
```

$$S0 := \left[\begin{array}{c}[x, \mathrm{y}(x), \mathrm{z}(x)] \\ \left[\begin{array}{ccc} 0 & \mathrm{Float}(1, \infty) & \mathrm{Float}(-1, \infty) \\ .0001 & .1096677497\ 10^{-5} & 0. \\ 1 & .9999844799 & 2.554567407 \end{array}\right]\end{array}\right]$$

$$\eta = 85.15224690$$

The effectiveness factor has been computed for a wide range of values for γ, β, and ϕ. A sample of the results, for $\gamma = 30$, are displayed in the following plot. The four datasets displayed in this plot correspond to $\beta = .4$ (box -- top curve), $\beta = .2$ (dotted line -- second curve from top), $\beta = .1$ (solid line -- second curve from bottom), and $\beta = 0$ (dashed line -- bottom curve). The multiple values for the exothermic catalyst ($\beta = .4$) were obtained using different initial guesses for the control parameter ($\alpha = 0$, $\alpha = .5$, and $\alpha = 1$).

>

Plot not feasible to include here. Please refer to the original journal article, or contact the first author for a copy of this plot.

>

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 ω. 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^θ, v^z, in cylindrical coordinates. The radial symmetry of this problem eliminates θ as a independent variable. Thus, with ρ and ν 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]:

$$\frac{1}{r}(r\, v^r)_r + (v^z)_z = 0,$$

$$v^r\,(v^r)_r + v_z (v^r)_z - \frac{1}{r}(v^\theta)^2 = -\frac{1}{\rho}p_r + \nu\left((v^r)_{rr} + \frac{1}{r}(v^r)_r + (v^r)_{zz} - \frac{v^r}{r^2}\right) \quad (9)$$

$$v^r\,(v^\theta)_r + v^z\,(v^\theta)_z + \frac{1}{r}v^r\,v^\theta = \nu\left((v^\theta)_{rr} + \frac{1}{r}(v^\theta)_r + (v^\theta)_{zz} - \frac{v^\theta}{r^2}\right)$$

$$v^r\,(v^z)_r + v^z\,(v^z)_z = -\frac{1}{\rho}p_z + \nu\left((v^z)_{rr} + \frac{1}{r}(v^z)_r + (v^z)_{zz}\right)$$

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

$$(v^r)(r, 0) = (v^z)(r, 0) = 0,$$
$$(v^\theta)(r, 0) = r\,\omega, \quad (10)$$
$$\mathrm{p}(r, 0) = 0$$

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

$$\lim_{z \to \infty} (v^r)(r, z) = 0,$$
$$\lim_{z \to \infty} (v^\theta)(r, z) = 0, \quad (11)$$

$$\lim_{z \to \infty} (v^z)_z(r, z) = 0.$$

>

Similarity solutions to this equation were found by Karman [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

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

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:

$$\begin{aligned} H' &= -2F, \\ F'' &= F^2 - G^2 + F'H, \\ G'' &= 2FG + HG' \\ P' &= -HH' + H'', \end{aligned} \qquad (13)$$

$$\mathrm{F}(0) = \mathrm{H}(0) = \mathrm{P}(0) = 0,$$

$$\mathrm{G}(0) = 1,$$

$$\lim_{z^* \to \infty} \mathrm{F}(z^*) = \lim_{z^* \to \infty} \mathrm{G}(z^*) = 0.$$

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* = ∞ 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;
> with(Shoot):
> with(plots):
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the assigned name shoot now has a global binding

Warning, the name changecoords has been redefined

> 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:

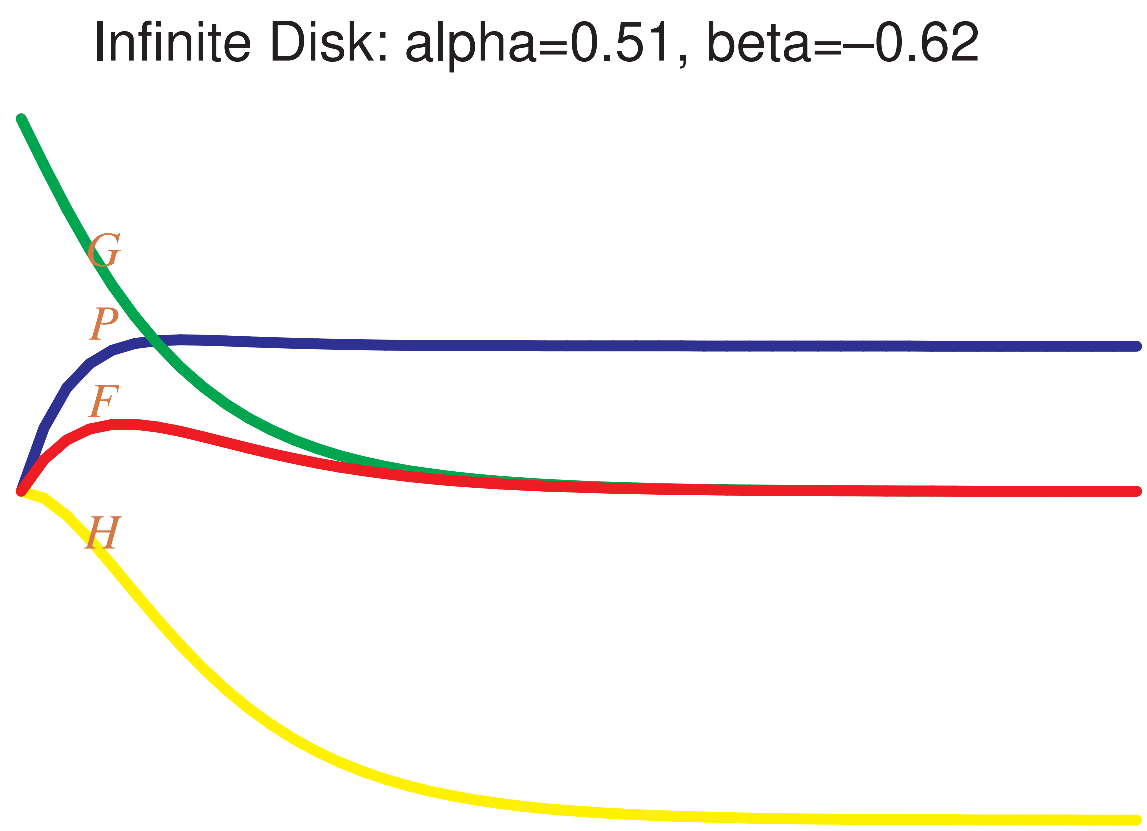
```
> IC:={ F(0)=0, G(0)=1, H(0)=0,
>        Fp(0)=alpha, Gp(0)=beta }:
```

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, IC, BC, FNS,
>           [alpha=0.51, beta=-0.62] ):
newton:   Step #    1
newton:   Parameter values :    alpha = .51, beta = -.62
newton:   Step #    2
newton:   Parameter values :    alpha = .5100033193, beta = -.6161487475
newton:   Step #    3
newton:   Parameter values :    alpha = .5102117353, beta = -.6159070528
newton:   Step #    4
newton:   Parameter values :    alpha = .5102135912, beta = -.6159096495
>
> 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 ):
> seq( op(0,lhs(f))=rhs(f), f=select(has,S(3/4),{F,G,H}) ):
> [%,P=subs(%, H^2+2*F)]:
> L:=textplot( [ seq([3/4,rhs(f)+1/45,lhs(f)],f=%) ],
>              font=[TIMES,ITALIC,18], align=ABOVE ):
> display( [C,L], labels=[`z*`,``],
>          title=`Infinite Disk: `
>                ||`alpha=0.51, beta=-0.62` );
```



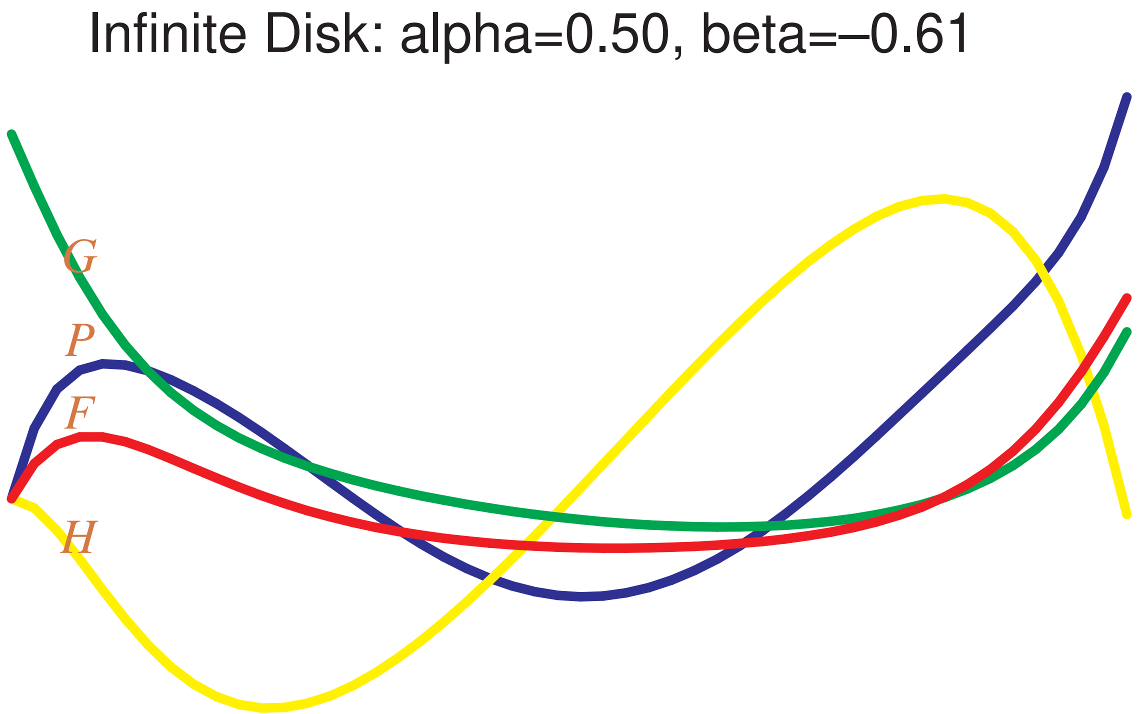
This plot confirms that the solution satisfies the original boundary conditions at $z^*=\infty$. (In fact, the solution obtained when the computational domain is truncated at $z^*=7$.) Compare the solution in the preceding plot with the solution obtained when the starting values of the control parameters are $\alpha = .50$ and $\beta = -.61$ -- each just 0.01 less than the first attempt:

```
> S2:=shoot( ODE, IC, BC, FNS,
>            [alpha=0.5, beta=-0.61] ):
newton:   Step #     1
newton:   Parameter values :    alpha = .5, beta = -.61
newton:   Step #     2
newton:   Parameter values :    alpha = .5181506110, beta = -.5834922896
newton:   Step #     3
newton:   Parameter values :    alpha = .5178653205, beta = -.5900697917
newton:   Step #     4
newton:   Parameter values :    alpha = .5019889184, beta = -.5940268798
newton:   Step #     5
newton:   Parameter values :    alpha = .5040798104, beta = -.5927957600
newton:   Step #     6
newton:   Parameter values :    alpha = .5042719728, beta = -.5927086094
newton:   Step #     7
newton:   Parameter values :    alpha = .5042737798, beta = -.5927079136
```

The different values of α and β 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 ):
> seq( op(0,lhs(f))=rhs(f), f=select(has,S2(3/4),{F,G,H}) ):
> [%,P=subs(%, H^2+2*F)]:
> L:=textplot( [ seq([3/4,rhs(f)+1/45,lhs(f)],f=%) ],
>              font=[TIMES,ITALIC,18], align=ABOVE ):
```

```
> display( {C,L}, labels=[`z*`,``],
>          title=`Infinite Disk: `
>                ||`alpha=0.50, beta=-0.61` );
               #read `/k1/faculty/meade/www/maple/worksheets/shoot`;
```



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.che.sc.edu/centers/CEE/, 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

[SB] J. Stoer and R. Bulirsch, Introduction to Numerical Analysis, Springer-Verlag, 1980.
[B] Yonathan Bard, Nonlinear Parameter Estimation, Academic Press, 1974.
[D] Mark E. Davis, Numerical Methods and Modeling for Chemical Engineers, John Wiley & Sons, Inc., 1984.
[AMR] Uri M. Ascher, Robert M. M. Mattheij, and Robert D. Russell, Numerical Solution of Boundary Value Problems for Ordinary Differential Equations}, SIAM, 1995.
[ACR] 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.
[W] Frank M. White, Viscous Fluid Flow, 1st ed., McGraw--Hill, Inc., 1974.
[WH] 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.
[F] Bruce A. Finlayson, Nonlinear Analysis in Chemical Engineering, McGraw-Hill, 1980.
[K] T. von Karman, ‘Uber laminare und turbulente Reibung, Z. Angew. Math. Mech., 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.

>