parameters $\tilde{E}$ and a_0. It saves the results of the integrations as `insoln` and `outsoln`, respectively.

In this case, we already had a good idea [4] of what the energy eigenvalue $\tilde{E}$ is for the $1s$ ground state. After some fiddling, we found an initial choice of parameters

$$\tilde{E} = 0.82, \quad a_0 = 0.2, \quad a_1 = 1000.0 \tag{25}$$

for which the resulting outwards and inwards integrations yielded the curves for $g(x)$ and $f(x)$ shown in Fig. 1. (As mentioned above, the asymptotic normalization a_1 is at this point arbitrary, to be fixed later by the normalization condition.)

It is useful at this point to define a function, `calcmatchgaps`, that uses `insoln` and `outsoln` to calculate and print out the values of $g(x)$ and $f(x)$ at the match point x_{match}, along with their slopes and their gaps scaled as in Eq. (18). These numbers provide guidance as to how to proceed when developing the Mathematica notebook.

Next, we need the four partial derivatives in the two-by-two matrix M of Eq. (22). In pseudocode, the subroutine for calculating $\partial\,\Delta g/\partial\tilde{E}$ is

```
subroutine dggapbydE(tildeE,a0)
   delE = delta*tildeE
   outsoln1 = outward integration with parameters (tildeE + delE,a0)
              from epsilon to xmatch
   g_out1 = g(x) at x_match from outsoln1
   insoln1 = inward integration with parameters (tildeE + delE,a0)
             from xmax to xmatch
   g_in1 = g(x) at x_match from insoln1
   ggap1 = 2.0*(g_out1 - g_in1)/(g_out1 + g_in1)
   dggapdE = (ggap1 - ggap)/delE
   return dggapdE
   end subroutine
```

Similar subroutines are also implemented for the three other partial derivatives. We also found it useful, once these subroutines were in place, to define another subroutine, `gapsandpartials`, which calls `shootinandout` for the present values of the parameters $\tilde{E}$ and a_0, followed by calls of `calcmatchgaps` and the four subroutines for the partial derivatives.

At this point we are ready to solve the two-by-two linear system for improved values of the parameters $\tilde{E}$ and a_0. Manipulating arrays is a bit tricky in a programming language such as Fortran or C, but in Mathematica one can simply define the matrix M in Eq. (22) as a list of lists and the column vector C_{old} is a simple list. Mathematica also provides a