absorption.m

```
function f = absorption( x )
%f(1) = left-hand side of Eq. (1) written in the form LHS = 0
%f(2) = left-hand side of Eq. (2) written in the form LHS = 0
%x(1) = n_1^l
%x(2) = n_2^l

%Eq. (1)
f(1) = ( 232 - x(1)) / (232 - x(1) + 8881 - x(2) + 5557) ...
       - 1440 / 1.2 / 760 * x(1) / (x(1) + x(2));

%Eq. (2)
f(2) = (8881 - x(2)) / (232 - x(1) + 8881 - x(2) + 5557) ...
       - 0.0419 / 1.2     * x(2) / (x(1) + x(2));

end
```

solve.m

```
fun = @absorption;
x0 = [0,8881];
x = fsolve(fun,x0)
```

solve_alt.m

```
options = optimoptions('fmincon');
fun = @absorption;
x0 = [0,8881];
x = fsolve(fun,x0,options)
```

Session

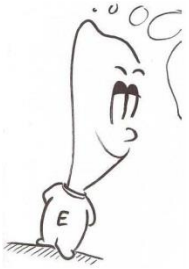
```
>> solve

Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

x =

   1.0e+03 *

    0.1130    8.6811
```

```
>> solve_alt
Warning: You have passed FMINCON options to FSOLVE. FSOLVE will use the
common options and ignore the
FMINCON options that do not apply.
To avoid this warning, convert the FMINCON options using OPTIMOPTIONS.
> In SolverOptions>SolverOptions.convertForSolver at 452
  In prepareOptionsForSolver at 25
  In fsolve at 140
  In solve_alt at 4

Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

x =

   1.0e+03 *

    0.1128    8.6798

>>
```

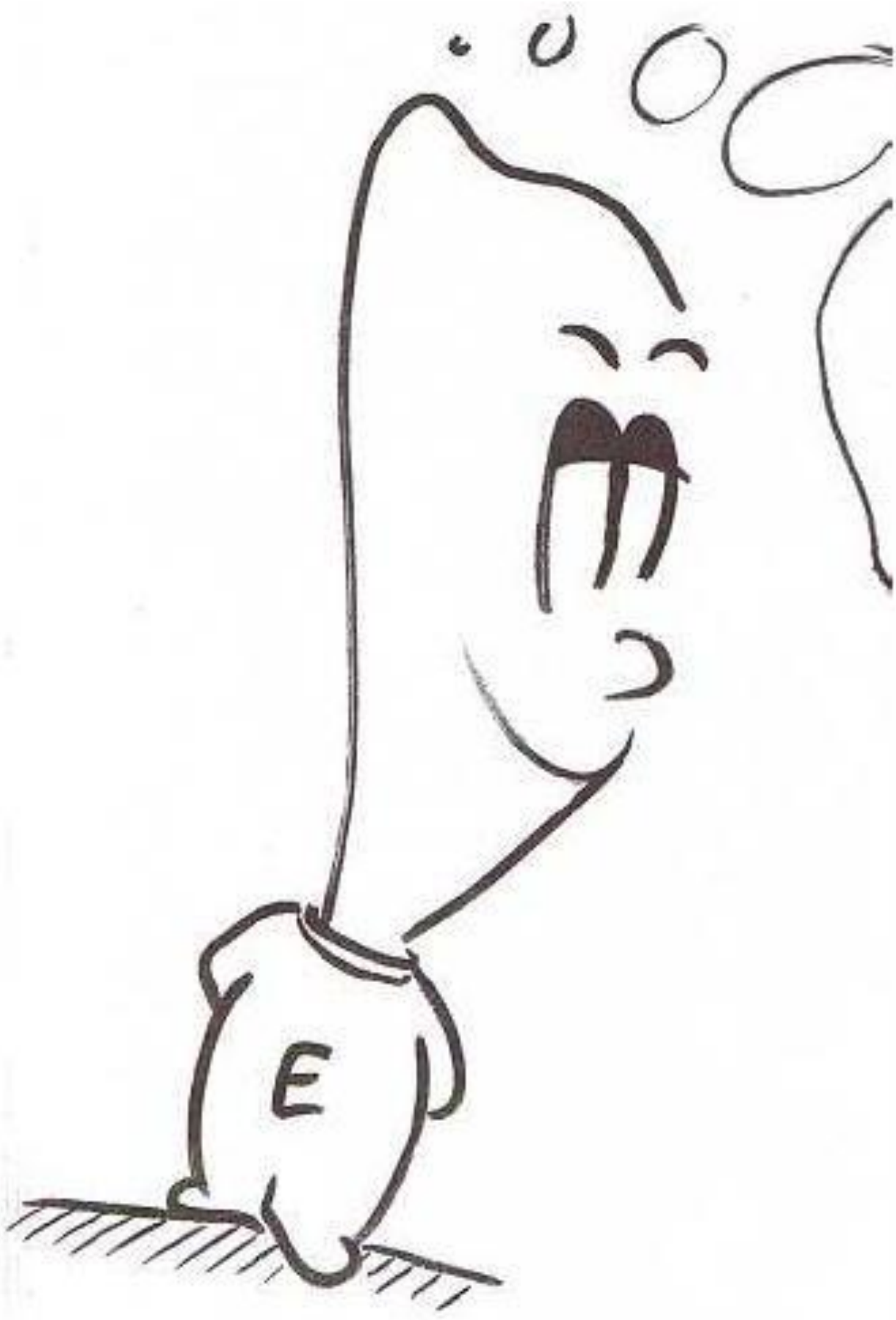
Comments

Matlab makes things way easy, man! How *can* life be so good?!

Second solution is more accurate.



How *very* nice!


E