Convergence rates
Computing, reporting, and plotting

MATH1090: Directed Study in Differential Equations

http://people.sc.fsu.edu/~jburkardt/classes/math1090_2020/convergence/convergence.pdf

---

We hope that our ODE process converges if we try hard enough!

Convergence

We need a way to organize the investigation of convergence of ODE solutions, as well as the difficulties of accurately solving a stiff ODE.

1 Computing errors and error rates

In the last discussion, we focused on an example we called the "stiff equation". We expect that, for any reasonable ODE solver, our approximate solution will converge to the exact solution, if we use enough steps n, or equivalently, use a small enough stepsize $h = \frac{\text{Tfinal}-\text{Tinitial}}{n}$.

In many cases, we can predict in advance the relationship between e, the norm of the error, and h, the stepsize. If the relationship is $e \propto h^k$, we say that our ODE solver is "order k accurate". We have already seen that both the explicit and backward Euler methods are order 1 methods, and that the Runge-Kutta-4 method is order 4.

It is important to know how to verify claims like this. To do so, we have to collect data (run the ODE solver on the same problem for several stepsizes), and then display the data, either by computing ratios for a table, or plotting $\log 10(h)$ versus $\log 10(e)$ and estimating the slope of the resulting curve.

```
1 for n = [ 20, 40, 80, 160, 320, 640 ]
2   [ t, y1 ] = stiff_euler ( n );
3   y2 = stiff_exact ( t );
4   e = rms ( y1 - y2 );
5   disp ( [ n, h, e ] )
6 end
```

Listing 1: Collecting convergence data.

n	h	$\|\|\text{error}\|\|_2$
20	0.0500	973.6328
40	0.0250	0.0841
80	0.0125	0.0279
160	0.0063	0.0122
320	0.0031	0.0058
640	0.0016	0.0028

Table 1: Convergence results for the stiff equation.

One way to study the convergence rate is to print the successive ratios of h and e. For instance, if we repeatedly double n (halving h) then our h ratio will be 2, and our e ratio will tend to 2 for a method of order 1, or 4 for a method of order 2, or in general 2^k for a method of order k.

```
for n = [ 20, 40, 80, 160, 320, 640 ]
  h = 1.0 / n;
  [ t, y1 ] = stiff_euler ( n );
  y2 = stiff_exact ( t );
  e = rms ( y1 - y2 );
  if ( 20 < n )
    disp ( [ hold/h, eold/e ] )
  end
  hold = h;
  eold = e;
end
```

Listing 2: Tabulating step and error ratios.

To make a table from this data, we might use the following LaTeXcommands:

```
\begin{table}
  \begin{center}
  \begin{tabular}{rll}
    n  &  h  &  $||\text{error}||_2$ \\
    \hline
   20   &     0.0500 &    973.6328\\
   40   &     0.0250 &      0.0841\\
   80   &     0.0125 &      0.0279\\
  160   &     0.0063 &      0.0122\\
  320   &     0.0031 &      0.0058\\
  640   &     0.0016 &      0.0028\\
  \end{tabular}
  \caption{Convergence results for the stiff equation.}
  \end{center}
\end{table}
```

which will produce the table, although it might not show up in the exact spot where you issued the commands.

For technical reports, it is more usual to plot $\log 10(h)$ versus $\log 10(e)$. We can estimate the slope of the resulting line, which gives us a measure of the convergence rate.

```
hvec = [];
evec = [];
for n = [ 20, 40, 80, 160, 320, 640 ]
  h = 1.0 / n;
  hvec = [ hvec, h ];
```

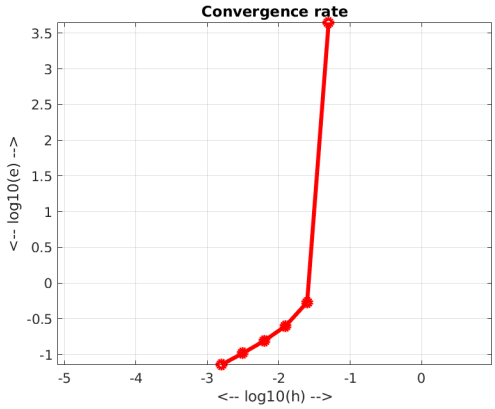
```
  [ t, y1 ] = stiff_euler ( n );
  y2 = stiff_exact ( t );
  e = rms ( y1 - y2 );
  evec = [ evec, e ];
end

plot ( log10(hvec), log10(evec), 'r-o', 'linewidth', 3 );
axis ( 'equal' )
grid ( 'on' );
xlabel ( '<-- log10(h) -->' );
ylabel ( '<-- log10(e) -->' );
title ( 'Convergence rate' );
filename = 'stiff_convergence.png';
print ( '-dpng', filename );
```

Listing 3: Plotting log10(h) versus log10(e).

and we can display the plot using LaTeXcommands like this:

```
\begin{center}
  \includegraphics[scale=0.5]{stiff_convergence.png}
  \vskip 0.1in
  {\it{Log/Log convergence plot for the stiff equation.}}
\end{center}
```

although, again, the plot will not necessarily show up exactly where your commands are.



Log/Log convergence plot for the stiff equation.

If we estimate the slope of the line in the formula $\log 10(e) = \text{intercept} + \text{slope} * \log 10(h)$, we can get an estimate for the slope of about 0.7, meaning that $e \approx C * h^{0.7}$ so that we are getting pretty poor convergence for this problem.

2 The Flame ODE

The flame ODE is a simple model of combustion. We suppose that the air has been sprayed with a flammable substance. At time $t = 0$, a small ball of initial radius $r(0) = \delta$ ignites into flame. The amount of burning material inside the ball is proportional to r^3. In order for the flame to continue burning, it must "breathe" oxygen from the surrounding atmosphere. The amount of oxygen it can breathe is proportional to the surface area, which is proportional to r^2. The ball of flame can expand as long as the surface area exceeds

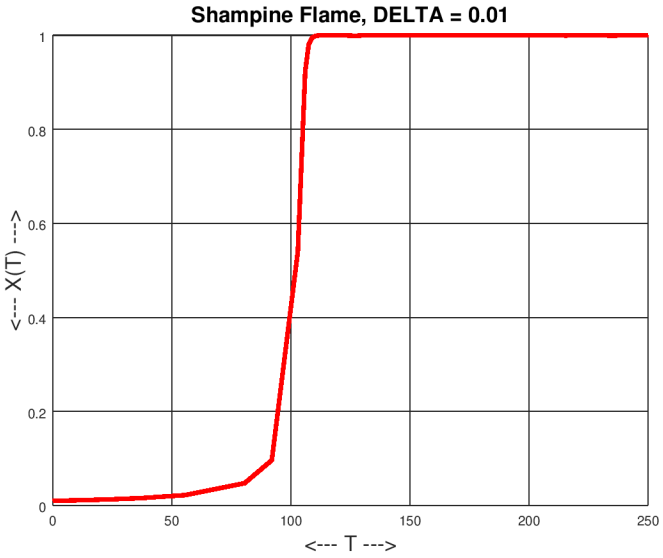
the volume. This results in the following ODE for the radius:

$$\frac{dr}{dt} = r^2 - r^3$$
$$r(0) = \delta$$

We can see that the right hand side has two roots, at $r = 0$ and $r = 1$. If the initial condition is positive, then the solution should always tend towards $r = 1$. Negative initial conditions are not physically meaningful, but they should result in solutions that tend to $r = 0$.

There is an exact solution for this problem, which is expressed in terms of the Lambert W function as follows:

$$a = \frac{\delta - 1}{\delta}$$
$$r(t) = \frac{1}{W(ae^{a-t} + 1)}$$



A computed solution to the flame problem for $\delta = 0.01$

As the plot suggests, the solution is likely to involve a sudden sharp rise in the solution value, corresponding physically to a kind of explosion. We will be interested in seeing how well various ODE methods can accurately approximate this behavior.

You might assume that, if a solver has difficulty with this equation, it is just at portion of the curve with the sharp rise. In fact, problems are even more likely to occur when the computed solution is "cruising" around near the final portion of the curve, where $r(t) \approx 1$. This is because any solution below 1 will tend to move sharply upwards, and any solution above 1 will move sharply downwards. And thus, some ODE solvers will zigzag above and below the correct curve; this, in turn, will "confuse" the solver, making it cut back severely in stepsize. Even though the correct, exact solution is very smooth, the neighborhood of this solution is very choppy, and an explicit ODE solver will get lost in these cross currents.

For more information on the flame problem, and its relevance to stiff ODE solvers, see:

1. Guy Rouleau, *Guy on Simulink: Why do we need stiff ODE solvers?*, https://blogs.mathworks.com/simulink/2012/07/03/why-do-we-need-stiff-ode-solvers/?s_tid=srchtitle, posted 03 July 2012;

2. Cleve Moler, *Cleve's Corner: Ordinary Differential Equations, Stiffness*, https://blogs.mathworks.com/cleve/2014/06/09/ordinary-differential-equations-stiffness/?s_tid=srchtitle, posted 09 June 2014.

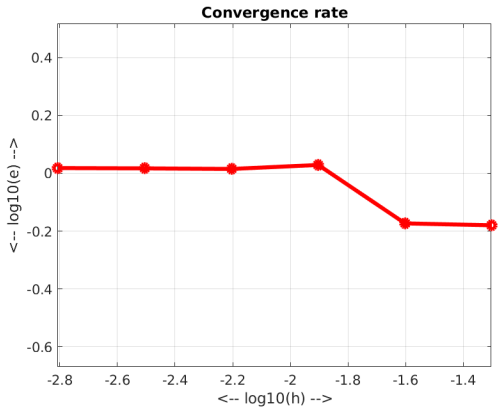
3 Convergence study for the flame ODE

Just as we did with the stiff equation, we can do a convergence study for the flame ODE. We have two functions available. Notice that, for the exact function, we need the additional input of δ, the radius of the ball at the initial time, which wewill take to be $\delta = 0.01$.

- `drdt = flame_deriv ( t, r )`: evaluates the right hand side of the ODE;
- `r = flame_exact ( t, delta )`: returns the exact solution at time t;

Notice that `flame_exact()` evaluates the Lambert W function using the built-in MATLAB function `value = lambertw(x)`.

The flame problem is actually difficult to solve with good precision. We might try using a backward Euler solver for a first attempt. Unfortunately, we end up getting a "convergence" plot that tells us that we are **not** converging:



Log/Log convergence plot for the flame equation with backward Euler.

This plot is a warning that our errors are not decreasing as we decrease h! We will have to wait til next time for a discussion of what is going wrong, and what we can do about it.

4 Report: Convergence of rk4 for the flame problem

We know that the rk4 ODE solver has higher accuracy (of order 4, actually) than the Euler methods. Can we hope that using such a method will give us a reasonable convergence plot for the flame problem?

Using a starting value of $\delta = 0.01$, and running from $0 \leq t \leq 250$, try to do a convergence study for the flame problem with rk4, similar to what I showed for the backward Euler method. Create a convergence table and a convergence plot.

Bring your completed table to our next meeting, at 2:00pm, Thursday, 20 February, room Thackery 624.