We didn't start the fire

The initial-value problem

$$\frac{dr}{dt} = r^2(1-r), \qquad t > 0, \quad r(0) = r_0, \tag{1}$$

is a simple model for the radius of a spherical flame ball in zero gravity. As $t \to \infty$, the solution tends to 1. In MATLAB, you can create a high-accuracy reference solution by entering, for example:

```
f = @(t,r) r^2*(1-r);
opt = odeset('reltol',1e-13,'abstol',1e-13);
soln = ode113(f,[0 500],0.01,opt);
r_hat = @(t) deval(soln,t);
```

The second line sets an absolute and relative (whichever is weaker) error target for the solution. The third line causes MATLAB to find all the information needed for a numerical solution with $r(0) = 0.01$ over $0 \le t \le 500$ and returns the data in a special structure. The last line creates a callable function that uses the data structure. Thereafter, you can evaluate the reference solution at any time t just by calling `r_hat(t)`, even if the `soln` variable is later changed.

Preparation

Read Section 6.2. On one labeled plot, graph the reference solution described above, $\hat{r}(t)$, for $0 \le t \le 500$ for $r_0 = 0.05$, 0.01, and 0.005.

Goals

Although the Euler method (as implemented by `eulerivp` in the book) is first-order convergent, the behavior of its solutions can be surprisingly complicated, as these experiments will reveal.

Procedure

1. Compute the reference solution for $r_0 = 0.01$ over $0 \le t \le 500$, and plot it.
2. Use `eulerivp` to compute the solution over $0 \le t \le 100$ with step size $h = 1$. Plot the reference solution and the Euler solution together over $[0, 100]$.
3. For $n = 50, 100, 150, \ldots, 10000$, use the `eulerivp` function to solve the IVP with $r_0 = 0.01$ up to time $t = 100$. Make a log-log plot of the error at time $t = 100$ versus n, and verify that the method is first-order convergent.
4. Repeat part 3 for solving at time $t = 500$. The convergence behavior is *very* different from the previous case.
5. In a 4×1 array of subplots, graph the solution $r(t)$ over $0 \le t \le 500$ using `eulerivp` to solve with $n = 100$, 150, 200, and 250. As you can see, the size of the error at a single time does not tell the full story of how the numerical solutions behave.