Math 542 Spring 2015 — Assignment #3. Due 3/6/2015.

Explicit Runge-Kutta Methods; code abstraction

The goal of this assignment is to write a general purpose RK routine; here described in a matlab context (but feel free to re-interpret into and program in any language environment).

```
[t_out,y_out,e_out] = rk( ode_RHS, y_0, t_range, c, A, b1, b2 )
```

Input Parameters	
`ode_RHS`	The function handle to the right-hand-side f of the ODE $y' = f(t, y)$.
`y_0`	The initial value for the ODE, *i.e.* $y(T_{\text{start}})$. Note that this may be a vector.
`t_range`	For now, the vector $T_{\text{start}}{:}h{:}T_{\text{stop}}$, where h is the step-size.
`c`	The vector $\vec{c}$ in the Butcher Array that specifies the RK-method.
`A`	The matrix A in the Butcher Array that specifies the RK-method.
`b1`	The vector $\vec{b_1}$ in the Butcher Array that specifies the primary (stepping) RK-method.
`b2`	[Optional parameter] The vector $\vec{b_2}$ in the Butcher Array that specifies the secondary (error-estimating) RK-method.
Output Parameters	
`t_out`	The times where the solution was computed. For now, this is simply `t_range`.
`y_out`	The solution computed at the times `t_out`.
`e_out`	The step-error estimated at the times `t_out`. Computed only when `b2` is supplied.

Note, "[Optional parameter]" — May or may not be supplied; your code should handle *both* cases.

In the end you should be able to do something like this:

```
>> c = [0 1/4 3/8 12/13 1 1/2];
>> A = [0 0 0 0 0 0; 1/4 0 0 0 0 0; 3/32 9/32 0 0 0 0];
>> A = [A; 1932/2197 -7200/2197 7296/2197 0 0 0];
>> A = [A; 439/216 -8 3680/513 -845/4104 0 0];
>> A = [A; -8/27 2 -3544/2565 1859/4104 -11/40 0];
>> b1 = [ 25/216 0 1408/2565 2197/4104 -1/5 0];
>> b2 = [ 16/135 0 6656/12825 28561/56430 -9/50 2/55];
>> f = @(t,y) (y + 2*t - 1);
>> [tv001,yv001,ev001] = rk( f, 1, 0:(1/1):1, c, A, b1, b2 );
>> [tv002,yv002,ev002] = rk( f, 1, 0:(1/2):1, c, A, b1, b2 );
>> [tv004,yv004,ev004] = rk( f, 1, 0:(1/4):1, c, A, b1, b2 );
>> [tv008,yv008,ev008] = rk( f, 1, 0:(1/8):1, c, A, b1, b2 );
>> [tv016,yv016,ev016] = rk( f, 1, 0:(1/16):1, c, A, b1, b2 );
>> [tv032,yv032,ev032] = rk( f, 1, 0:(1/32):1, c, A, b1, b2 );
>> [tv064,yv064,ev064] = rk( f, 1, 0:(1/64):1, c, A, b1, b2 );
>> [tv128,yv128,ev128] = rk( f, 1, 0:(1/128):1, c, A, b1, b2 );
```

Questions?

1. Where is the step size h?! — You probably want to use something like `h_current_step = t_range(k) - t_range(k-1)` so that your code does not break if/when user passes in a non-uniform `t_range`, *e.g.* `t_range = [0 0.1 0.2 0.4 0.46 0.47 0.5 0.7 0.8 0.9 0.99 0.999 0.9999 1]`