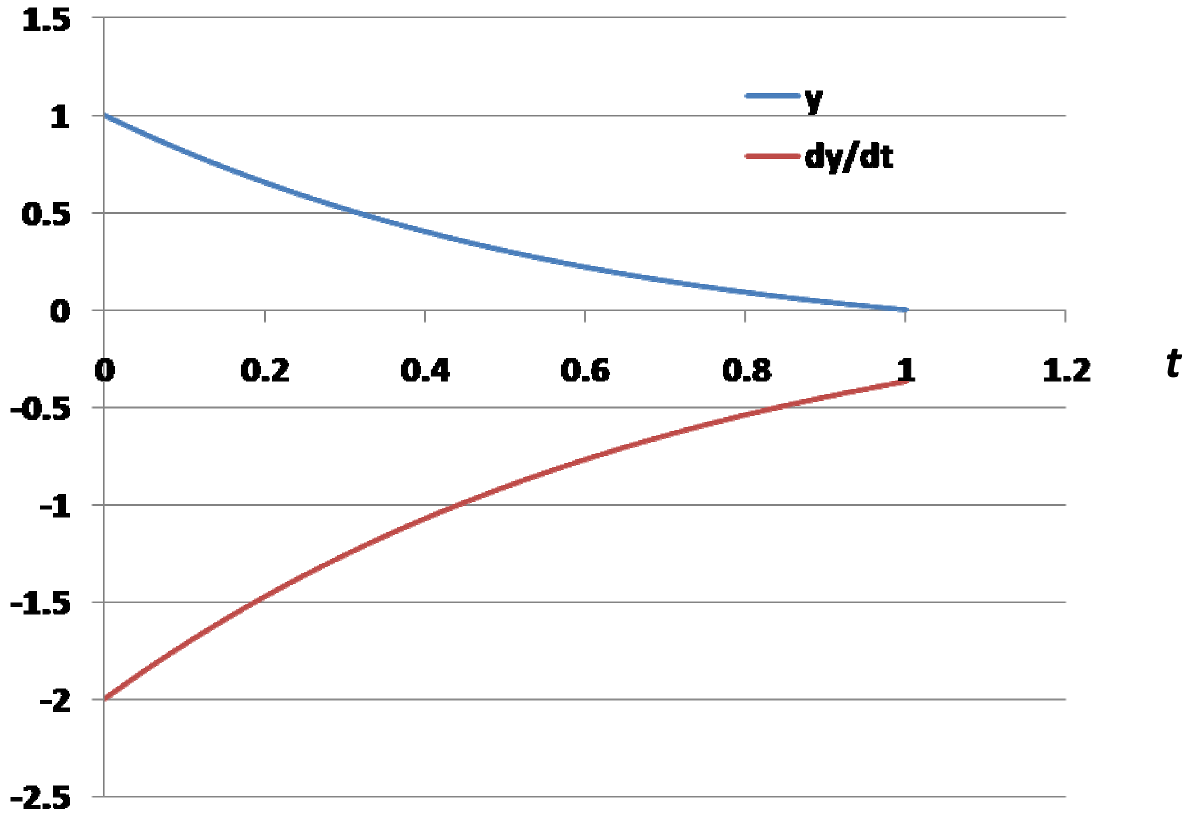
4th order Runge-Kutta method: Example

- Given: $dy/dt = -y - e^{-t}$; $y(0)=1$

➢ Find: y at t=0.2 (using h=0.2) **TV= 0.654985**

➢ Exact Solution: $y = e^{-t}(1-t)$



4^{th} order Runge-Kutta method: Example

- Fourth-order R-K method

$$k_1 = f(t_n, y_n); k_2 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1\right)$$

$$k_3 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2\right); k_4 = f(t_n + h, y_n + hk_3)$$

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

- $t_0=0$, $y_0=1$, $k_1= -1 - e^{-0} = -2$; $k_2= -(1+0.1\text{x}(-2)) - e^{-0.1} = -1.70484$; $k_3= -(1+0.1\text{x}(-1.70484)) - e^{-0.1} = -1.73435$; $k_4= -(1+0.2\text{x}(-1.73435)) - e^{-0.2} = -1.47186$

- $y_{0.2}$ = 1+0.2/6(-2-2x1.70484 - 2x1.73435 -1.47186) = 0.654992

Error Analysis

- The "local truncation error (LTE)" is the error over one interval (t_n, t_{n+1})
- The "global truncation error (GTE)" is the error over the entire time period (t_0, t_{n+1})
- ***E.g., Euler Forward***

$$y_{n+1} = y_n + hf\left(t_n, y_n\right)$$

- ***Compare with Taylor's series***

$$y_{n+1} = y_n + hf\left(t_n, y_n\right) + \frac{h^2}{2} f'\left(t_n, y_n\right) + \ldots$$

- ***Local Error is O(h^2)***
- ***Global?***

Global Truncation Error

- We start with t_0, where y_0 is known exactly
- $y_1 = y_0 + f(t_0,y_0)$
- **Error *(LTE & GTE)*** $\dfrac{h^2}{2} y''(\zeta_0); \zeta_0 \in (t_0, t_1)$

- For the next step $y_2 = y_1 + hf(t_1, y_1)$
- But, y_1 is not exact. Let ^ indicate exact values
- ***Error:***

$$\hat{y}_2 - (y_1 + hf(t_1, y_1)) = \hat{y}_1 + h\hat{f}_1 + \frac{h^2}{2} y''(\zeta_1) - y_1 - hf_1$$

$$= \frac{h^2}{2} y''(\zeta_0) + \frac{h^2}{2} y''(\zeta_1) + H.O.T.$$

Global Truncation Error

- LTE in the second step is $\dfrac{h^2}{2} y''(\zeta_1); \zeta_1 \in (t_1, t_2)$
- ***GTE is*** $\dfrac{h^2}{2} y''(\zeta_0) + \dfrac{h^2}{2} y''(\zeta_1)$
- Proceeding similarly, GTE up to t_{n+1}:

$$= \frac{h^2}{2} \sum_{i=0}^{n} y''(\zeta_i); \zeta_i \in (t_i, t_{i+1})$$

- Using mean value theorem, GTE is O(h):

$$GTE = \frac{(n+1)h^2}{2} y''(\bar{\zeta}); \bar{\zeta} \in (t_0, t_{n+1})$$

$$= \frac{(t_{n+1} - t_0)h^2}{2h} y''(\bar{\zeta}) = \frac{(t_{n+1} - t_0)h}{2} y''(\bar{\zeta})$$

Global Truncation Error

- A numerical scheme for solving the ODE is called a k^{th} order scheme, if the GTE is $O(h^k)$
- The LTE is $O(h^{k+1})$
- For example, 2^{nd} order R-K method have LTE $O(h^3)$ and GTE $O(h^2)$
- Let us take the same example and solve by the Mid-point method (2^{nd} order R-K) and the 4^{th}-order R-K method.
- $dy/dt = -y - e^{-t}$; $y(0)=1$
- Use $h=0.1$, 0.2, and 0.3 and solve up to $t=0.6$

					RK2	
t	TV		yi	k1	k2	yi+1
0.0	1.000000		1.000000	-2.000000	-1.851229	0.814877
0.1	0.814354		0.814877	-1.719714	-1.589599	0.655917
0.2	0.654985		0.655917	-1.474648	-1.360986	0.519819
0.3	0.518573		0.519819	-1.260637	-1.161475	0.403671
0.4	0.402192		0.403671	-1.073991	-0.987600	0.304911
0.5	0.303265		0.304911	-0.911442	-0.836289	0.221282
0.6	0.219525		0.221282	-0.770094	-0.704823	0.150800

						RK4		
t	TV		yi	k1	k2	k3	k4	yi+1
0.0	1.000000		1.000000	-2.000000	-1.851229	-1.858668	-1.718971	0.814354
0.1	0.814354		0.814354	-1.719191	-1.589102	-1.595607	-1.473524	0.654985
0.2	0.654985		0.654985	-1.473716	-1.360100	-1.365781	-1.259225	0.518573
0.3	0.518573		0.518573	-1.259392	-1.160292	-1.165247	-1.072369	0.402193
0.4	0.402192		0.402193	-1.072513	-0.986195	-0.990511	-0.909672	0.303266
0.5	0.303265		0.303266	-0.909797	-0.834726	-0.838480	-0.768230	0.219525
0.6	0.219525		0.219525	-0.768337	-0.703154	-0.706413	-0.645469	0.148976

Errors in R-K methods

t	RK2	RK4		t	RK2	RK4		t	RK2	RK4
0.0	0.00000000	0.00000000000		0.0	0.00000000	0.00000000		0	0.00000000	0.00000000
0.1	-0.00052338	-0.00000023413		0.2	-0.00404791	-0.00000732		0.3	-0.01321485	-0.00005439
0.2	-0.00093252	-0.00000041628		0.4	-0.00642561	-0.00001157		0.6	-0.01870549	-0.00007630
0.3	-0.00124582	-0.00000055494		0.6	-0.00764207	-0.00001369				
0.4	-0.00147906	-0.00000065736								
0.5	-0.00164579	-0.00000072977								
0.6	-0.00175758	-0.00000077747								

Stability Analysis

- Stability: The numerical solution should be bounded if the exact solution is bounded
- Different from "error," a stable solution could have large errors
- A numerical scheme may be stable for all values of time-step (unconditionally stable) or only for time-step less than a threshold (conditionally stable)
- Also, a numerical sheme with the same time-step may be stable for some ODE's and unstable for some other

Linear Stability Analysis

- We perform only a Linear Stability analysis
- Expand the function *f* (i.e., dy/dt) in a Taylor's series and ignore the higher order terms

$$f(t,y) = f(t_0, y_0) + (t - t_0)\left.\frac{\partial f}{\partial t}\right|_{(t_0,y_0)} + (y - y_0)\left.\frac{\partial f}{\partial y}\right|_{(t_0,y_0)} + \ldots$$

which may be written as $\frac{d\hat{y}}{dt} = c_0 + c_1 t + \lambda \hat{y}$

- Similarly, the approximate solution is written as

$$\frac{dy}{dt} = c_0 + c_1 t + \lambda y$$

Linear Stability Analysis

- The error is then obtained from

$$\frac{d(\hat{y}-y)}{dt}=\lambda(\hat{y}-y)$$

- Which implies that the growth of error follows the first-order “model problem”

$$\frac{dy}{dt}=\lambda y$$

- Therefore, for a linear stability analysis, we follow the differential equation given above

Linear Stability Analysis

- Mostly we deal with real values of λ, but consider here a more general case:

$$\frac{dy}{dt} = (\lambda_r + i\lambda_i)y$$

- The exact solution of the problem, with the initial condition $y=y_0$ at $t=t_0$, is

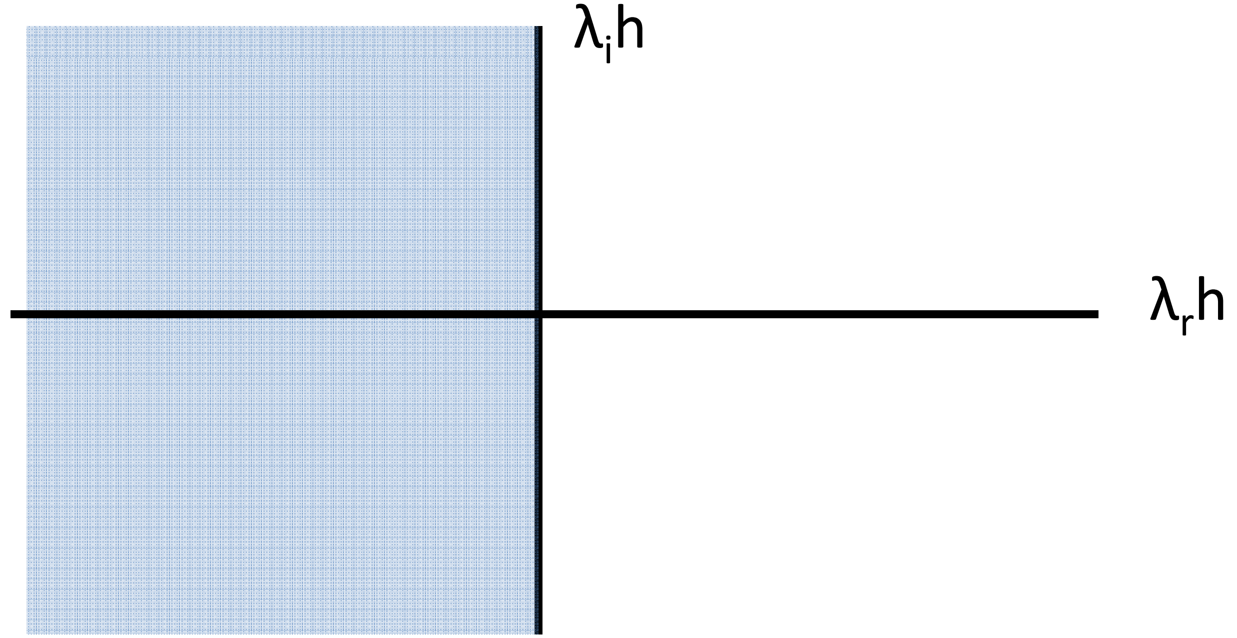
$$y = y_0 e^{(\lambda_r + i\lambda_i)t} = y_0 e^{n(\lambda_r h + i\lambda_i h)}$$

- Since we want to compare the analytical and numerical solutions, we write t=nh, although the analytical solution has no relation with h

Linear Stability Analysis

$$y = y_0 e^{n(\lambda_r h + i\lambda_i h)}$$

- Clearly, the analytical solution is bounded for all negative λ_r
- We show this through a stability region



Linear Stability Analysis

- We now look at the stability region of various numerical methods.
- Start with the Euler Forward

$$y_{n+1} = y_n + hf\left(t_n, y_n\right) = y_n\left(1 + \lambda_r h + i\lambda_i h\right)$$

- Define an amplification factor, σ, as the ratio of y at two consecutive time steps (y_{n+1}/y_n)

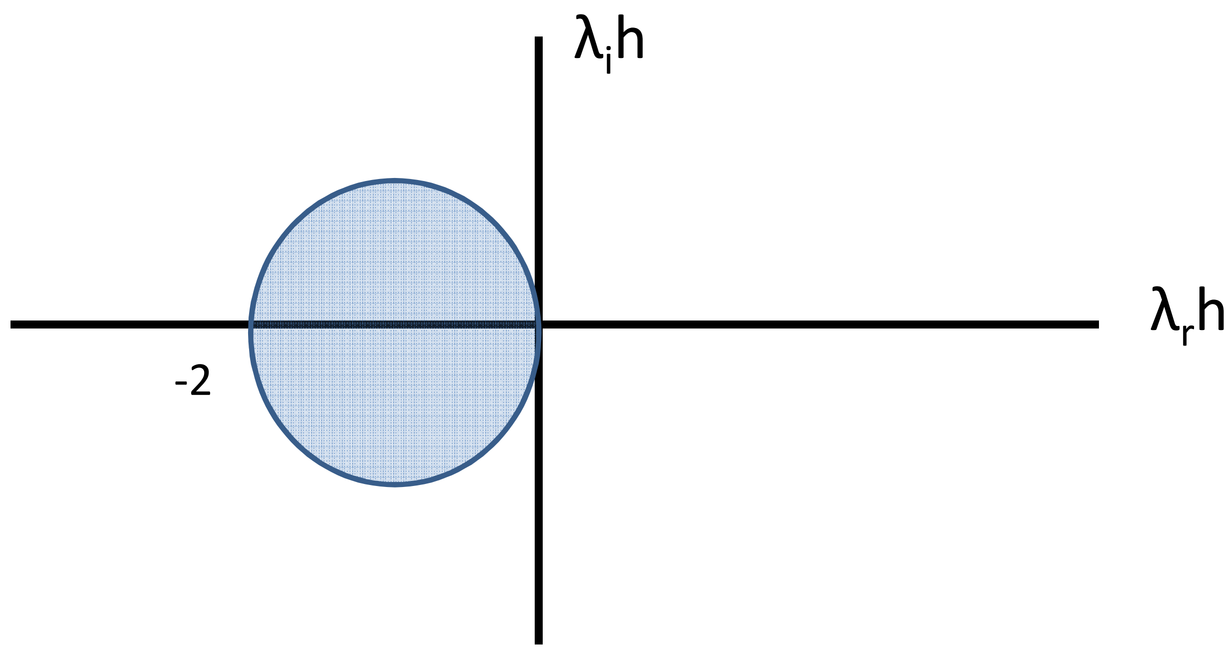
$$\sigma = 1 + \lambda_r h + i\lambda_i h$$

- For solution to be bounded, |σ| must be ≤1
- The stability region is, therefore, given by

$$\left(1 + \lambda_r h\right)^2 + \lambda_i^2 h^2 \leq 1$$

Linear Stability Analysis: Euler Forward

- The stability region is shown below: a circle of radius 1, centered at (-1,0)
- For real negative values of λ, the condition is $|\lambda h| \leq 2$



Linear Stability Analysis: Euler Backward

- Similarly, for Euler Backward

$$y_{n+1} = y_n + hf\left(t_{n+1}, y_{n+1}\right) \Rightarrow y_{n+1} = y_n / \left(1 - \lambda_r h - i\lambda_i h\right)$$

$$\sigma = 1/(1 - \lambda_r h - i\lambda_i h)$$

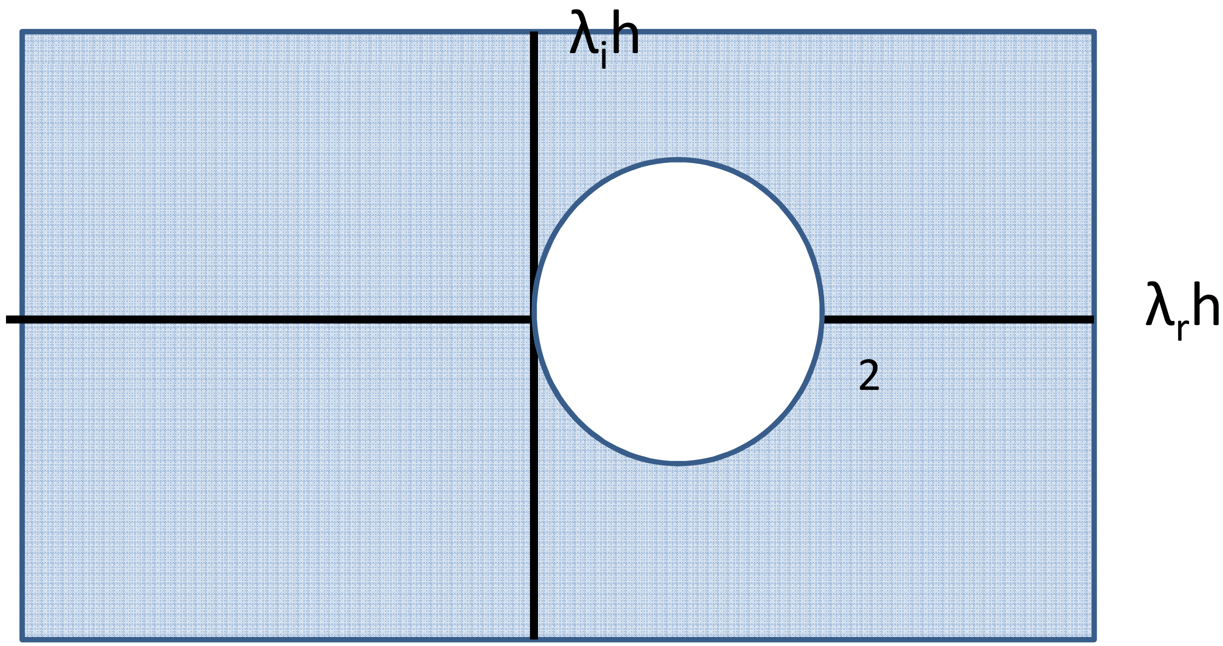
- For solution to be bounded, |σ| must be ≤1
- The stability region is, therefore, given by

$$\left(\lambda_r h - 1\right)^2 + \lambda_i^2 h^2 \geq 1$$

- Which is the area outside the circle of radius 1, centered at (1,0)

Linear Stability Analysis: Euler Backward

- The stability region is shown below: outside a circle of radius 1, centered at (-1,0)
- For real negative values of λ, the method is unconditionally stable



Linear Stability Analysis: Trapezoidal method

- For Trapezoidal method

$$y_{n+1} = y_n + h\frac{f(t_n, y_n) + f(t_{n+1}, y_{n+1})}{2} \Rightarrow y_{n+1} = y_n \frac{1 + \lambda_r h/2 + i\lambda_i h/2}{1 - \lambda_r h/2 - i\lambda_i h/2}$$

- The stability region is, therefore, given by

$$\left|\frac{1 + \lambda_r h/2 + i\lambda_i h/2}{1 - \lambda_r h/2 - i\lambda_i h/2}\right| \le 1$$

- Which implies $\lambda_r h \le 0$
- Same as that for the exact solution.
- Unconditionally stable, does not give bounded solution when the exact is not bounded!