Active Richardson's Extrapolation with Euler Forward: 4th order method start-up

Time	EF(h = 0.4)	Time	EF(h = 0.2)	Time	EF(h = 0.1)	Time	EF(h = 0.05)
0	1	0	1	0	1	0	1
0.4	**0.2**	0.2	0.6	0.1	0.8	0.05	0.9
		0.4	**0.399734**	0.2	0.649983	0.1	0.812499
				0.3	0.539854	0.15	0.736241
				0.4	**0.461435**	0.2	0.670089
						0.25	0.613013
						0.3	0.564082
						0.35	0.52245
						0.4	**0.48735**

t	EF(0.4,0.2)	EF(0.2,0.1)	EF(0.1,0.05)	EF(0.4,0.2,0.1)	EF(0.2,0.1,0.05)	EF(h^4)
0.4	0.599468	0.523136	0.513265	0.497692	0.509974186	**0.511729**

We will now apply the Euler Forward and Richardson's extrapolation with the extrapolated initial value at *t* = 0.4 and go up to *t* = 0.8

Active Richardson's Extrapolation with Euler Forward: 4th order method start-up

Time	EF(h = 0.4)	Time	EF(h = 0.2)	Time	EF(h = 0.1)	Time	EF(h = 0.05)
0.4	0.511729	0.4	0.511729	0.4	0.511729	0.4	0.511729
0.8	**0.258113**	0.6	0.384921	0.5	0.448325	0.45	0.480027
		0.8	**0.343881**	0.6	0.406602	0.5	0.453772
				0.7	0.381746	0.55	0.432366
				0.8	**0.369819**	0.6	0.415264
						0.65	0.40197
						0.7	0.392032
						0.75	0.38504
						0.8	**0.380618**

t	EF(0.4,0.2)	EF(0.2,0.1)	EF(0.1,0.05)	EF(0.4,0.2,0.1)	EF(0.2,0.1,0.05)	EF(h^4)
0.8	0.429649	0.395756	0.391417	0.3844589	0.389970418	**0.390758**

We will now apply the Euler Forward and Richardson's extrapolation with the extrapolated initial value at $t = 0.8$ and go up to $t = 1.2$

Active Richardson's Extrapolation with Euler Forward: 4th order method start-up

Time	EF(h = 0.4)	Time	EF(h = 0.2)	Time	EF(h = 0.1)	Time	EF(h = 0.05)
0.8	0.390758	0.8	0.390758	0.8	0.390758	0.8	0.390758
1.2	**0.365094**	1	0.377926	0.9	0.384342	0.85	0.38755
		1.2	**0.39505**	1	0.385806	0.9	0.386359
				1.1	0.392792	0.95	0.386889
				1.2	**0.403354**	1	0.388871
						1.05	0.392058
						1.1	0.396223
						1.15	0.401161
						1.2	**0.406683**

t	EF(0.4,0.2)	EF(0.2,0.1)	EF(0.1,0.05)	EF(0.4,0.2,0.1)	EF(0.2,0.1,0.05)	EF(h^4)
1.2	0.425005	0.411659	0.410012	0.4072102	0.40946293	**0.409785**

With four initial values at *t* = 0, 0.4, 0.8 and 1.2, any 4th order multi-step or BDF methods can start!

4^{th} Order R-K: 4^{th} order method start-up

t	φ_0	φ_1	φ_2	φ_3	y
0					**1**
0.4	-2	-1.00133	-1.4008	-0.48994	**0.51372**
0.8	-0.63802	-0.20759	-0.37976	-0.00627	**0.392453**
1.2	-0.06755	0.083584	0.02313	0.128628	**0.410754**

With four initial values at t = 0, 0.4, 0.8 and 1.2, any 4^{th} order multi-step or BDF methods can start!

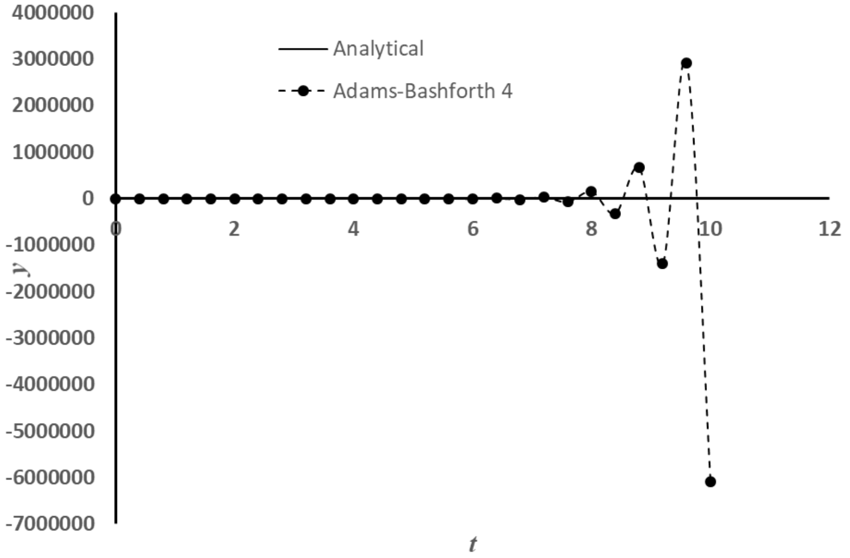
Startup of 4th Order Methods

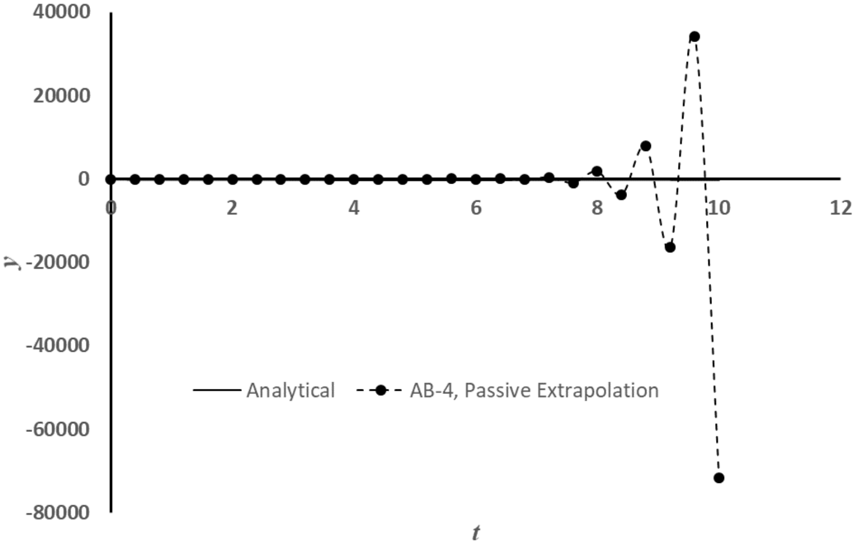
t	*y* (Passive)	*y* (Active)	*y* (R-K)
0	1	1	1
0.4	0.511729	0.511729	0.51372
0.8	0.389799	0.390758	0.392453
1.2	0.409017	0.409785	0.410754

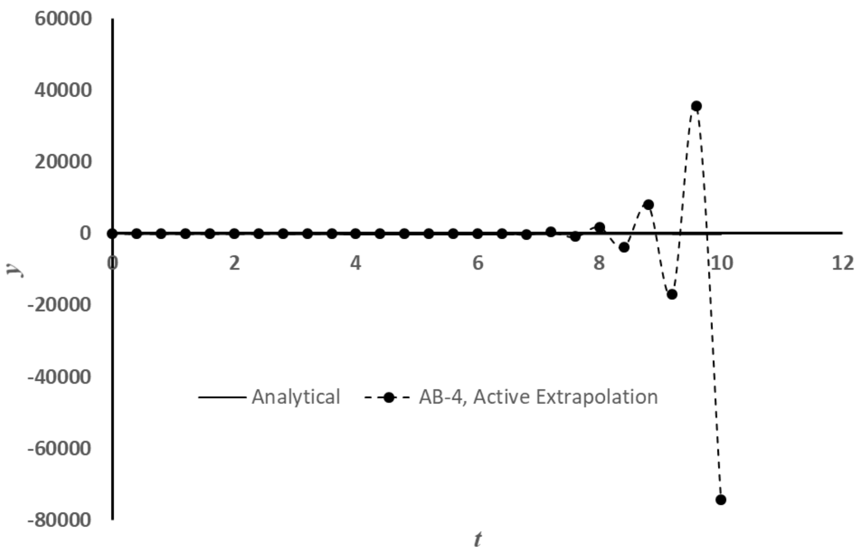
✓ 4th Order Adams-Bashforth (requires all four values)

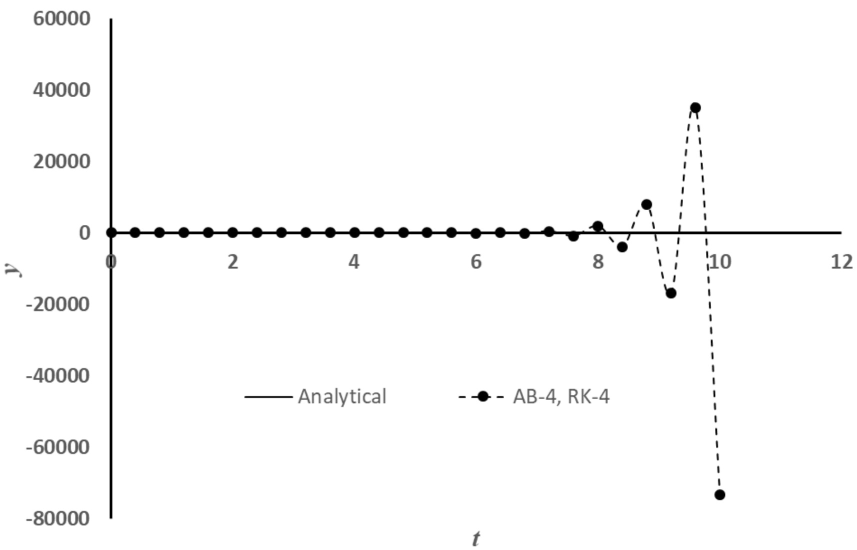
$$y_{n+1} = y_n + h\left[\frac{55}{24}(-2y_n + \sin t_n) - \frac{59}{24}(-2y_{n-1} + \sin t_{n-1}) + \frac{37}{24}(-2y_{n-2} + \sin t_{n-2}) - \frac{3}{8}(-2y_{n-3} + \sin t_{n-3})\right]$$

Startup of 4th Order Adams-Bashforth









Proper start-up does not solve the stability problem!

'

Startup of 4th Order Adams-Moulton

t	y (Passive)	y (Active)	y (R-K)
0	1	1	1
0.4	0.511729	0.511729	0.51372
0.8	0.389799	0.390758	0.392453
1.2	0.409017	0.409785	0.410754

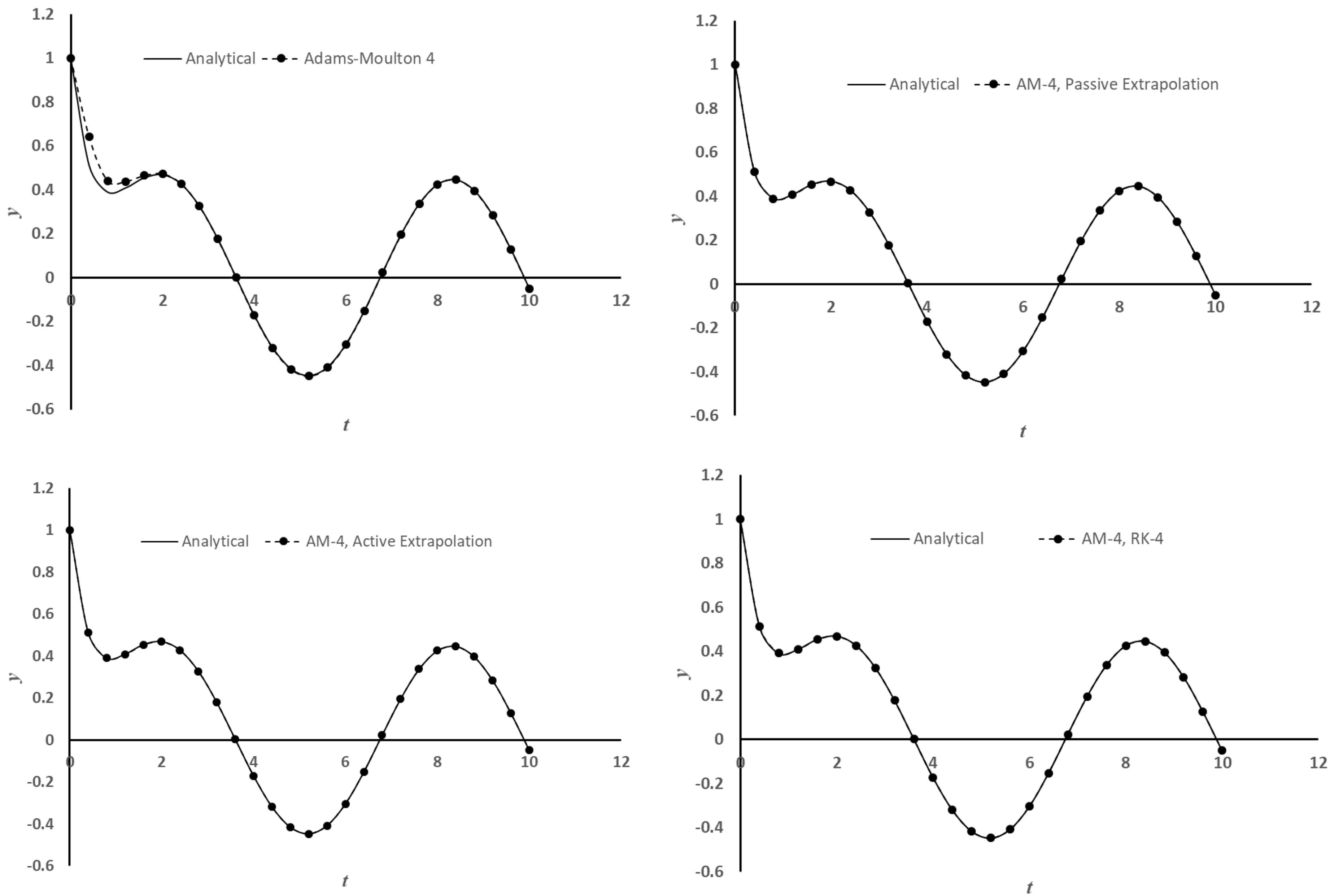
✓ 4th Order Adams-Moulton requires three initial values:

$$y_{n+1} = y_n + h\left[\frac{3}{8}(-2y_{n+1} + \sin t_{n+1}) + \frac{19}{24}(-2y_n + \sin t_n) - \frac{5}{24}(-2y_{n-1} + \sin t_{n-1}) + \frac{1}{24}(-2y_{n-2} + \sin t_{n-2})\right]$$

$$y_{n+1} = \frac{y_n + h\left[\frac{3}{8}\sin t_{n+1} + \frac{19}{24}(-2y_n + \sin t_n) - \frac{5}{24}(-2y_{n-1} + \sin t_{n-1}) + \frac{1}{24}(-2y_{n-2} + \sin t_{n-2})\right]}{\left(1 + \frac{3}{4}h\right)}$$

,

Startup of 4th Order Adams-Moulton



'

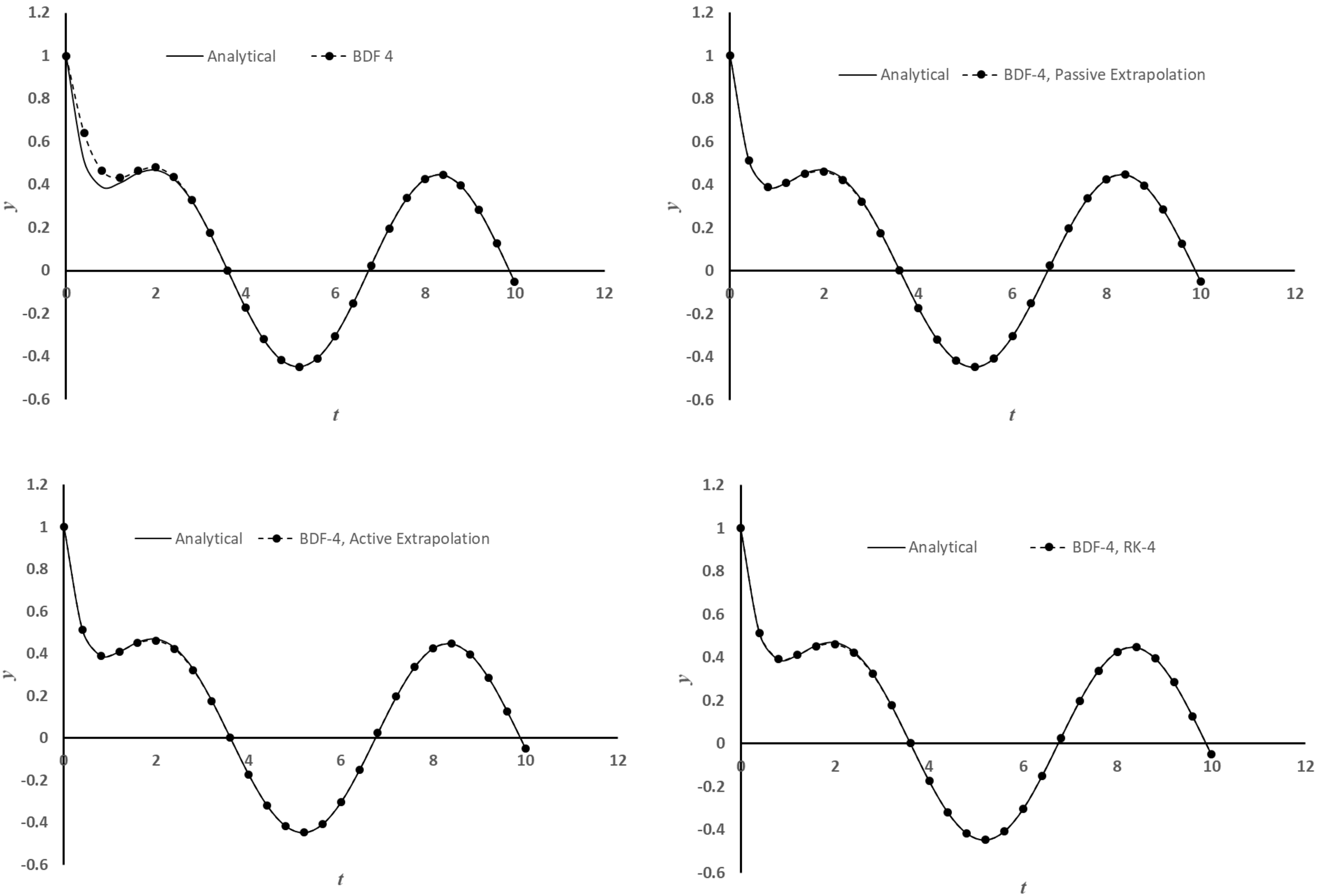
Startup of 4th Order BDF

t	*y* (Passive)	*y* (Active)	*y* (R-K)
0	1	1	1
0.4	0.511729	0.511729	0.51372
0.8	0.389799	0.390758	0.392453
1.2	0.409017	0.409785	0.410754

✓ 4th Order BDF requires all four initial values:

$$y_{n+1} = \frac{4y_n - 3y_{n-1} + \frac{4}{3}y_{n-2} - \frac{1}{4}y_{n-3} + h\sin t_{n+1}}{\left(\frac{25}{12} + 2h\right)}$$

Startup of 4th Order Adams-Moulton



How to start the non-self starting algorithms?

Three commonly used options:

- ✓ Passive Richardson’s Extrapolation
- ✓ Active Richardson’s Extrapolation
  - ✓ With same lower order method
  - ✓ With progressively higher order methods
- ✓ Using the same order Runge-Kutta method!
- ✓ A word of caution:
  - ✓ Active Richardson’s extrapolation may destroy the good stability property of a numerical method (under certain situation). It’s generally not recommended!

Applications: Summary of Concerns

- ✓ Accuracy of the higher order multi-step and BDF methods are affected if the starting values are used from the lower order methods.
  - ✓ How to start these non-self starting algorithms?
- ✓ All implicit methods (multi-step and BDF) may involve solution of non-linear equations (if f contains a non-linear function of the dependent variable y)
  - ✓ Is there a way to avoid this solution of non-linear equations?
- ✓ Numerical oscillations (instability) observed in some methods and not in some!
  - ✓ Is there a way to predict and therefore, choose correct parameters for algorithm so that the numerical oscillations can be avoided?

Combination Methods: Predictor-Corrector

- ✓ How to avoid solution of non-linear equations in implicit methods (multi-step and BDF):
  - ✓ Use a ***Predictor*** method (an explicit method) to compute a value of y_{n+1}
  - ✓ Use the predicted value of y_{n+1} in a ***Corrector*** method (an implicit method) to obtain the value of y_{n+1}
- ✓ All these methods can be applied in two ways:
  - ✓ *Single* application of corrector equation
  - ✓ *Repeated* or *multiple* or *iterative* application of corrector equation until the change in the value of y_{n+1} is below a specified ε.

Predictor-Corrector: Heun's Method

- ✓ Predictor Method: Euler Forward

$$y_{n+1}^{p} = y_n + hf(y_n, t_n)$$

- ✓ Corrector Method: Trapezoidal
  - ✓ Single application:

$$y_{n+1}^{c} = y_n + \frac{h}{2}\left[f\left(y_{n+1}^{p}, t_{n+1}\right) + f(y_n, t_n)\right]$$

  - ✓ Iterative application:

$$y_{n+1}^{c,i+1} = y_n + \frac{h}{2}\left[f\left(y_{n+1}^{c,i}, t_{n+1}\right) + f(y_n, t_n)\right]$$

$$y_{n+1}^{c,0} = y_{n+1}^{p}$$

Let us apply this to our problem!

Heun’s Method: Single corrector application

$$y_{n+1}^{p} = y_n + hf(y_n, t_n) = y_n + h(-2y_n + \sin t_n) = y_n(1 - 2h) + h \sin t_n$$

$$y_{n+1}^{c} = y_n + \frac{h}{2}\left[f\left(y_{n+1}^{p}, t_{n+1}\right) + f(y_n, t_n)\right] =$$
$$= y_n + \frac{h}{2}\left[\left(-2y_{n+1}^{p} + \sin t_{n+1}\right) + (-2y_n + \sin t_n)\right]$$

$$y_0 = y(0) = 1; \quad h = 0.4$$

$$y_1^p = 1(1 - 2 \times 0.4) + 0.4 \sin 0 = 0.2$$
$$y_1^c = y(0.4) = 1 + \frac{0.4}{2}[(-2 \times 0.2 + \sin 0.4) + (-2 \times 1 + \sin 0)] = 0.5979$$

$$y_2^p = 0.5979(1 - 2 \times 0.4) + 0.4 \sin 0.4 = 0.2753$$
$$y_2^c = y(0.8) = 0.5979 + \frac{0.4}{2}[(-2 \times 0.2753 + \sin 0.8) + (-2 \times 0.5979 + \sin 0.4)]$$
$$= 0.4699$$

Heun's Method (Single Corrector) vs. R-K (2nd Order)

- ✓ Predictor

$$y_{n+1}^{p} = y_n + hf(y_n, t_n) = y_n + h\phi_0$$

- ✓ Corrector

$$y_{n+1}^{c} = y_n + \frac{h}{2}\left[f\left(y_{n+1}^{p}, t_{n+1}\right) + f(y_n, t_n)\right] = y_n + \frac{h}{2}[\phi_1 + \phi_0]$$

- ✓ 2nd Order Runge-Kutta

$$\alpha_0 = 1 = \beta_1, \qquad \omega_0 = \frac{1}{2}, \qquad \omega_1 = \frac{1}{2}$$

$$y_{n+1} = y_n + h\left(\frac{\phi_0}{2} + \frac{\phi_1}{2}\right)$$

$$\phi_0 = f(y_n, t_n) \qquad \phi_1 = f(y_n + h\phi_0, t_n + h)$$

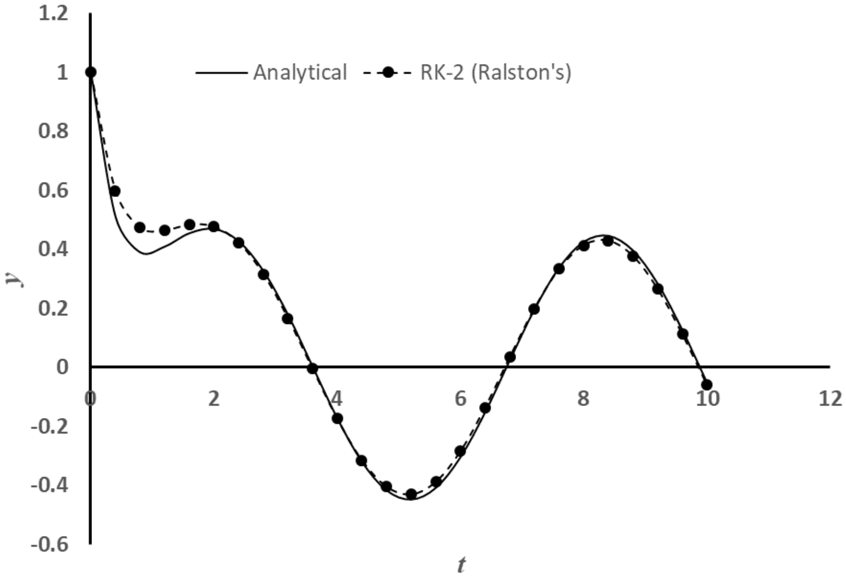
Heun's Method: Iterative corrector application

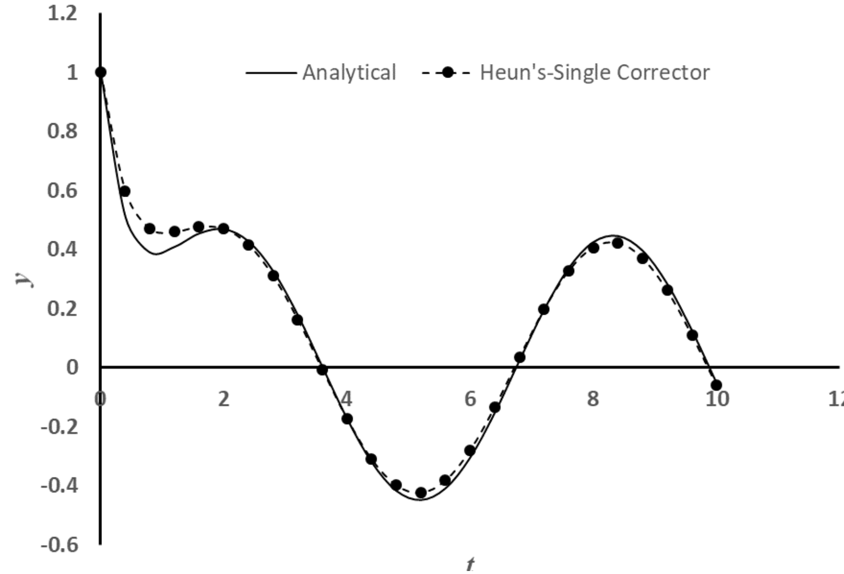
$$y_{n+1}^{p} = y_n(1-2h) + h\sin t_n = y_{n+1}^{c,0}$$
$$y_{n+1}^{c,i+1} = y_n + \frac{h}{2}\left[\left(-2y_{n+1}^{c,i} + \sin t_{n+1}\right) + \left(-2y_n + \sin t_n\right)\right]$$
$$y_0 = y(0) = 1; \quad h = 0.4; \qquad \varepsilon = 0.1\%$$
$$y_1^{p} = 1(1-2\times 0.4) + 0.4\sin 0 = 0.2 = y_1^{c,0}$$
$$y_1^{c,1} = 1 + \frac{0.4}{2}[(-2\times 0.2 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.5979$$
$$y_1^{c,2} = 1 + \frac{0.4}{2}[(-2\times 0.5979 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4699;\ \varepsilon = 26.62\%$$
$$y_1^{c,3} = 1 + \frac{0.4}{2}[(-2\times 0.4699 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.5024;\ \varepsilon = 14.51\%$$
$$y_1^{c,4} = 1 + \frac{0.4}{2}[(-2\times 0.5024 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4769;\ \varepsilon = 5.07\%$$
$$y_1^{c,5} = 1 + \frac{0.4}{2}[(-2\times 0.4769 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4871;\ \varepsilon = 2.14\%$$
$$y_1^{c,6} = 1 + \frac{0.4}{2}[(-2\times 0.4871 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4830;\ \varepsilon = 0.84\%$$
$$y_1^{c,7} = 1 + \frac{0.4}{2}[(-2\times 0.4830 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4847;\ \varepsilon = 0.34\%$$
$$y_1^{c,8} = 1 + \frac{0.4}{2}[(-2\times 0.4847 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4840;\ \varepsilon = 0.13\%$$
$$y_1^{c,9} = 1 + \frac{0.4}{2}[(-2\times 0.4840 + \sin 0.4) + (-2\times 1 + \sin 0)] = 0.4843;\ \varepsilon = 0.05\% < 0.1\%$$

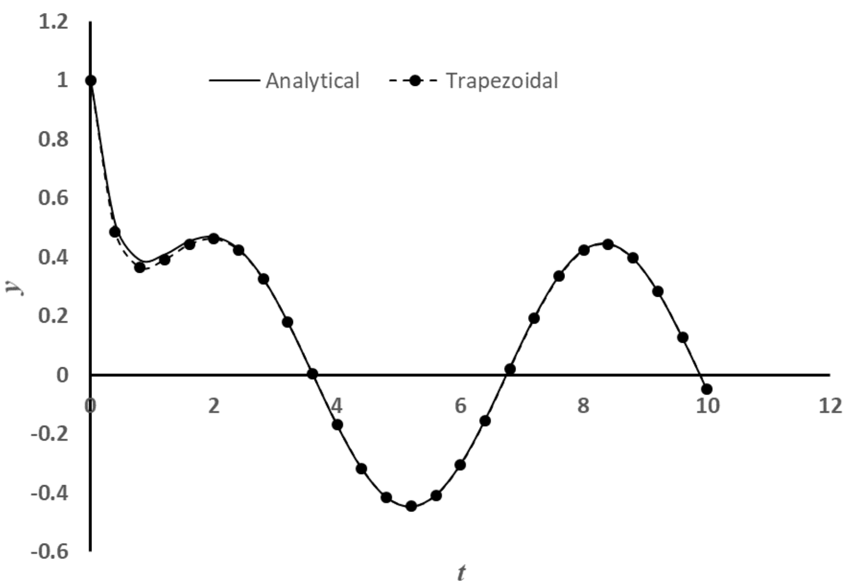
Heun's Method: Iterative corrector application

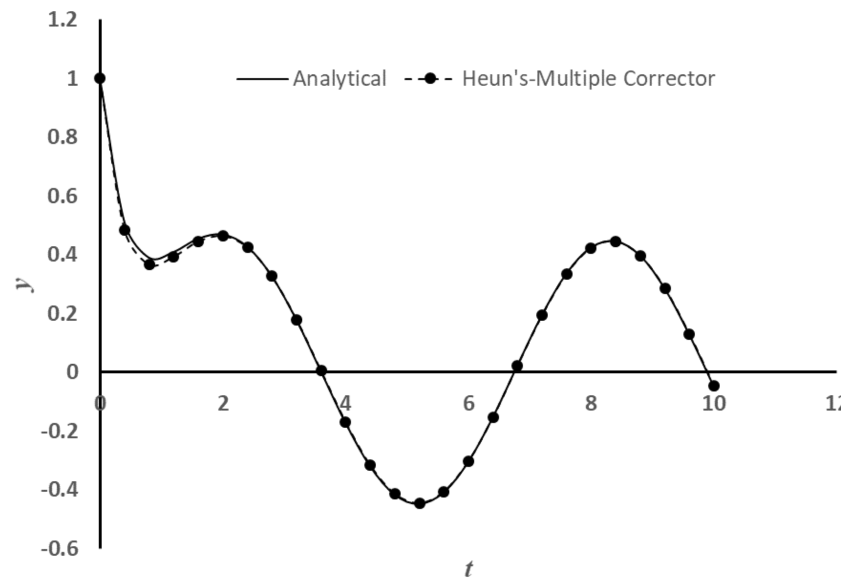
$$y_{n+1}^{p} = y_n(1-2h) + h\sin t_n = y_{n+1}^{c,0}$$
$$y_{n+1}^{c,i+1} = y_n + \frac{h}{2}\left[\left(-2y_{n+1}^{c,i} + \sin t_{n+1}\right) + \left(-2y_n + \sin t_n\right)\right]$$
$$y_0 = y(0) = 1; \qquad y_1 = y(0.4) = 0.4843; \qquad h = 0.4; \qquad \varepsilon = 0.1\%$$
$$y_2^{p} = 0.4843(1-2\times 0.4) + 0.4\sin 0.4 = 0.2526 = y_2^{c,0}$$
$$y_2^{c,1} = 0.4843 + \frac{0.4}{2}[(-2\times 0.2526 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.4108$$
$$y_2^{c,2} = 0.4843 + \frac{0.4}{2}[(-2\times 0.4108 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3475;\ \varepsilon = 15.41\%$$
$$y_2^{c,3} = 0.4843 + \frac{0.4}{2}[(-2\times 0.3475 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3729;\ \varepsilon = 7.28\%$$
$$y_2^{c,4} = 0.4843 + \frac{0.4}{2}[(-2\times 0.3729 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3627;\ \varepsilon = 2.72\%$$
$$y_2^{c,5} = 0.4843 + \frac{0.4}{2}[(-2\times 0.3627 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3668;\ \varepsilon = 1.12\%$$
$$y_2^{c,6} = 0.4843 + \frac{0.4}{2}[(-2\times 0.3668 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3652;\ \varepsilon = 0.44\%$$
$$y_2^{c,7} = 0.4843 + \frac{0.4}{2}[(-2\times 0.3652 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3658;\ \varepsilon = 0.18\%$$
$$y_2^{c,8} = 0.4843 + \frac{0.4}{2}[(-2\times 0.3658 + \sin 0.8) + (-2\times 0.4843 + \sin 0.4)] = 0.3656;\ \varepsilon = 0.07\% < 0.1\%$$

Predictor-Corrector: Heun’s Methods Comparison









Predictor-Corrector: Adams Method

✓ Predictor Method: Adams-Bashforth (4th Order)

$$y_{n+1}^{p} = y_n + h\left[\frac{55}{24}(-2y_n + \sin t_n) - \frac{59}{24}(-2y_{n-1} + \sin t_{n-1}) + \frac{37}{24}(-2y_{n-2} + \sin t_{n-2}) - \frac{3}{8}(-2y_{n-3} + \sin t_{n-3})\right]$$

✓ Corrector Method (Single Application): Adams-Moulton (4th Order)

$$y_{n+1}^{c} = y_n + h\left[\frac{3}{8}\left(-2y_{n+1}^{p} + \sin t_{n+1}\right) + \frac{19}{24}(-2y_n + \sin t_n) - \frac{5}{24}(-2y_{n-1} + \sin t_{n-1}) + \frac{1}{24}(-2y_{n-2} + \sin t_{n-2})\right]$$

Predictor-Corrector: Adams Method

✓ Predictor Method: Adams-Bashforth (4th Order)

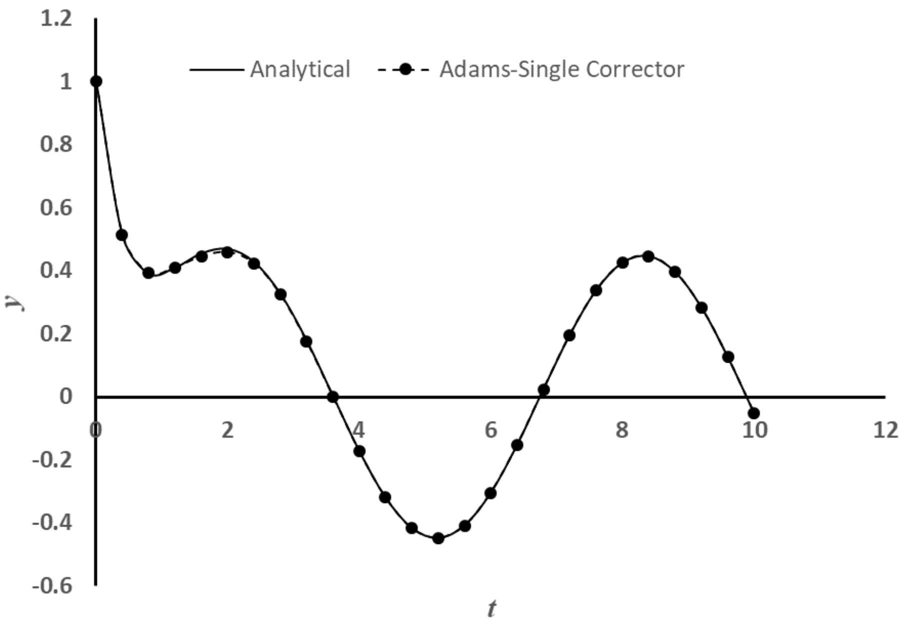
$$y_{n+1}^{p} =$$
$$= y_n$$
$$+ h\left[\frac{55}{24}(-2y_n + \sin t_n) - \frac{59}{24}(-2y_{n-1} + \sin t_{n-1})\right.$$
$$\left. + \frac{37}{24}(-2y_{n-2} + \sin t_{n-2}) - \frac{3}{8}(-2y_{n-3} + \sin t_{n-3})\right]$$

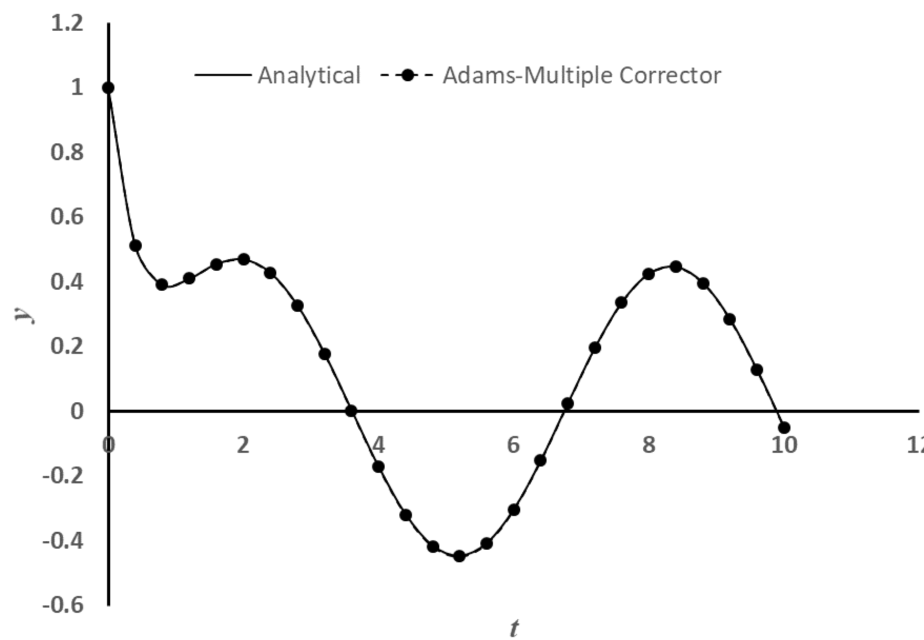
✓ Corrector Method (Iterative Application): Adams-Moulton (4th Order)

$$y_{n+1}^{c,i+1} =$$
$$= y_n$$
$$+ h\left[\frac{3}{8}\left(-2y_{n+1}^{c,i} + \sin t_{n+1}\right) + \frac{19}{24}(-2y_n + \sin t_n)\right.$$
$$\left. - \frac{5}{24}(-2y_{n-1} + \sin t_{n-1}) + \frac{1}{24}(-2y_{n-2} + \sin t_{n-2})\right]$$
$$y_{n+1}^{c,0} = y_{n+1}^{p}$$

Results when we apply this to our problem using 4th order R-K for startup!

Predictor-Corrector: Adams Methods Comparison





Combination Methods: Predictor-Corrector

- ✓ Predictor-Corrector methods only avoids the solution of non-linear equation. They still require appropriate startup to produce proper results!
- ✓ Predictor-Corrector method have the same stability properties as the corrector method
  - ✓ Adams-Bashforth (4^{th} order) was unstable for this problem with h = 0.4 but the predictor-corrector method produced correct results because of the influence of the corrector method Adams-Moulton (4^{th} order) which was stable!
- ✓ Iterative application of corrector produces more accurate results but at the expense of higher FLOPs.

Applications: Summary of Concerns

- ✓ Accuracy of the higher order multi-step and BDF methods are affected if the starting values are used from the lower order methods.
  - ✓ How to start these non-self starting algorithms?
- ✓ All implicit methods (multi-step and BDF) may involve solution of non-linear equations (if f contains a non-linear function of the dependent variable y)
  - ✓ Is there a way to avoid this solution of non-linear equations?
- ✓ Numerical oscillations (instability) observed in some methods and not in some!
  - ✓ Is there a way to predict and therefore, choose correct parameters for algorithm so that the numerical oscillations can be avoided?

Let's do *Convergence Analysis*!

ESO 208A: Computational Methods in Engineering

Ordinary Differential Equation: Consistency, Stability, Convergence

Saumyen Guha

Department of Civil Engineering
IIT Kanpur

Numerical Methods for IVPs: Convergence

Lax Equivalence Theorem:

The **numerical solution** by a finite difference method ***converges*** to the **true solution,** if and only if, the method is both ***consistent*** and ***stable***.

Consistency:

A finite difference method for an IVP is *consistent* if the global truncation error (GTE) is such that for any $\varepsilon > 0$, there exists a $h(\varepsilon) > 0$ for which $|GTE| < \varepsilon$. This also means,

$$\lim_{h \to 0} GTE = 0$$

A *consistent* numerical method is said to have an *order of accuracy* of p, if p is the largest positive integer such that $|GTE| \leq Kh^p$ for $0 < h \leq H$, where K and h are constants.

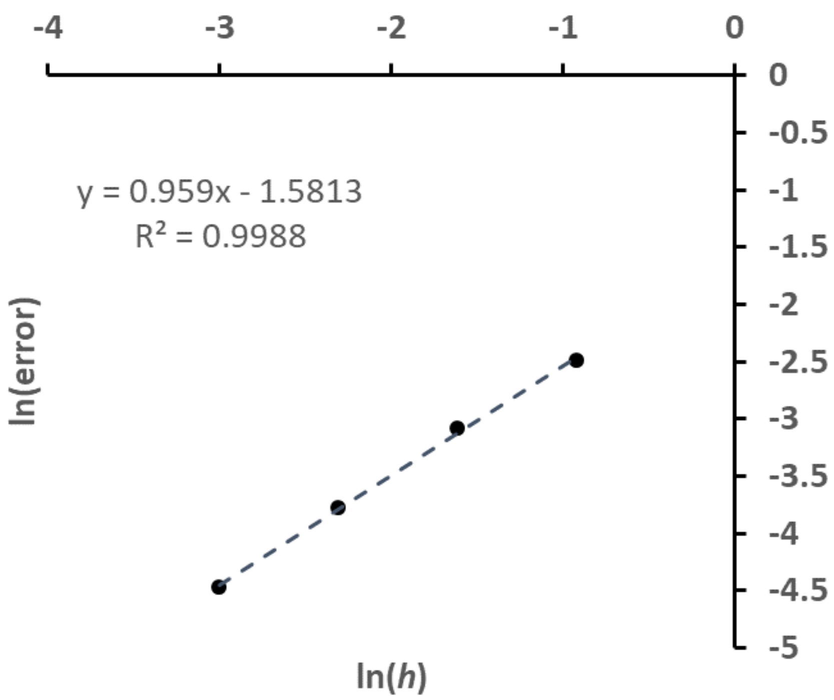
For all the methods derived so far, $p \geq 1$, therefore, consistent!

Numerical Methods for IVPs: Consistency

Consistency:

This also means, $\ln(|GTE|) = \ln(K) + p\ln(h)$

One can estimate the order of the method numerically.



This is the same result of the Euler Forward method used for startup using different values of h. The absolute values of true errors were computed at $h = 1.2$

Numerical Methods for IVPs: Stability

Stability:

If numerical solutions for an IVP is obtained using a method with two sets of initial conditions $\{y_0, y_1, y_2, \cdots y_n\}$ and $\{z_0, z_1, z_2, \cdots z_n\}$, the method is *stable* if there exists a constant K such that the following equality holds as $h \to 0$,

$$|y_{n+\mathrm{m}} - z_{n+\mathrm{m}}| \leq K \max\{|y_0 - z_0|, |y_1 - z_1|, \cdots |y_n - z_n|\} \ \forall m$$

In true mathematical terminology, the method will be called *zero stable*.

Consistency and *zero stability* are necessary conditions for *convergence*

Numerical Methods for IVPs: Stability

- ✓ *Zero stability* defines stability in asymptotic limit $h \rightarrow 0$
- ✓ For application with $h > 0$, let us define an *amplification factor* (σ)

$$\left|\frac{y_{n+1}}{y_n}\right| = \sigma$$

- ✓ σ can be real or complex
- ✓ Condition of stability is satisfied if $|\sigma| < 1$
- ✓ We need a ***model problem*** to assess and compare stability of various methods!

Stability: Model Problem

Mathematical Problem:

$$\frac{dy}{dt} = f(y,t) \qquad y(t_0) = y_0 \quad t \geq 0$$

- ✓ Since the solution starts from the fixed initial point (y_0, t_0) and expands to it's neighbourhood, let us explore the influence of function $f(y, t)$ in this region:

$$\frac{dy}{dt} = f(y_0, t_0) + (y - y_0)\left.\frac{\partial f}{\partial y}\right|_{(y_0,t_0)} + (t - t_0)\left.\frac{\partial f}{\partial t}\right|_{(y_0,t_0)} + \frac{(y-y_0)^2}{2}\left.\frac{\partial^2 f}{\partial y^2}\right|_{(y_0,t_0)}$$

$$+ (y - y_0)(t - t_0)\left.\frac{\partial^2 f}{\partial y \partial t}\right|_{(y_0,t_0)} + \frac{(t-t_0)^2}{2}\left.\frac{\partial^2 f}{\partial y^2}\right|_{(y_0,t_0)} + \cdots$$

- ✓ Collecting only the linear terms:

$$\frac{dy}{dt} = \alpha + \beta t + \lambda y + \cdots \qquad \Longrightarrow \qquad \frac{dy}{dt} = \lambda y + \gamma(t)$$

Stability: Model Problem

$$\frac{dy}{dt} = \lambda y + \gamma(t)$$

- ✓ Any approximate solution $\tilde{y}$ computed using a numerical method satisfies the approximate equation

$$\frac{d\tilde{y}}{dt} = \lambda \tilde{y} + \gamma(t)$$

- ✓ The error (ε) is defined as: $\varepsilon = y - \tilde{y}$. We may write:

$$\frac{d\varepsilon}{dt} = \lambda \varepsilon$$

- ✓ The function of the independent variable $\gamma(t)$ has no influence on the progression of error in the equation. Therefore the model problem for *linear stability analysis* is:

$$\frac{dy}{dt} = \lambda y; \quad y(0) = y_0 \quad \Longrightarrow \quad y = y_0 e^{\lambda t}$$

- ✓ We may evaluate the analytical solution in a discrete grid $\{t_0, t_1, t_2, \cdots t_n\}$ with a time step of h: $y_n = y_0 e^{\lambda n h}$

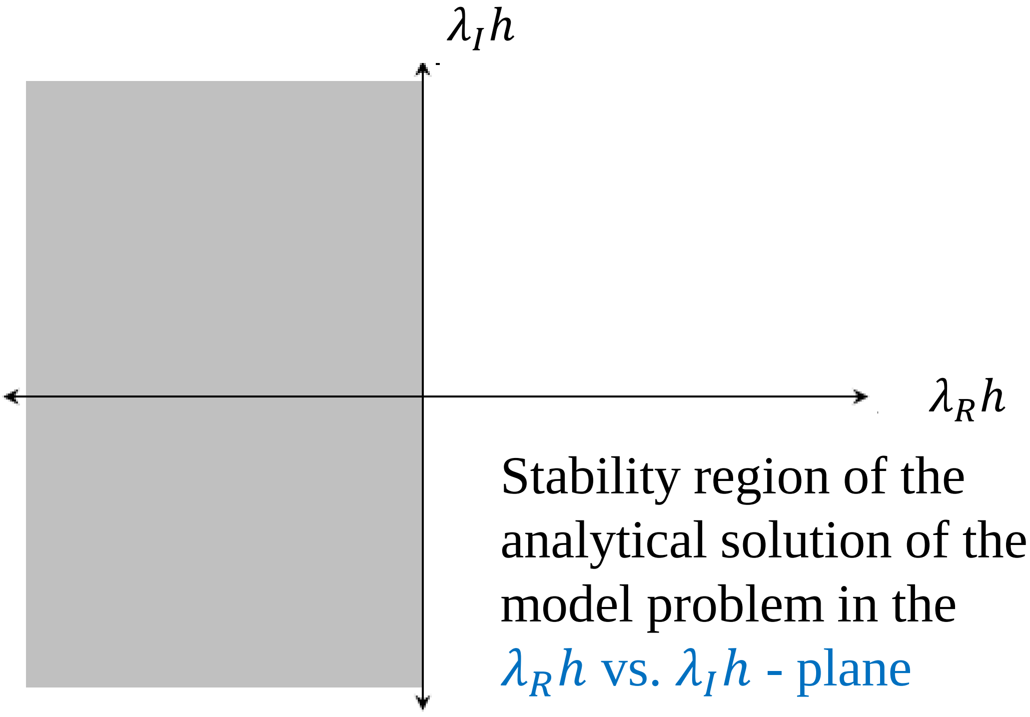
Stability: Model Problem

- ✓ For general applicability to all types of functions or problems, we let λ to be complex:

$$\lambda = \lambda_R + i\lambda_I \qquad \Longrightarrow \qquad y_n = y_0 e^{\lambda n h} = y_0 e^{(\lambda_R h)n} e^{i(\lambda_I h)n}$$

- ✓ The analytical solution grows unbounded, i.e., $|\sigma| > 1$ if $\lambda_R h > 0$ irrespective of the value of $\lambda_I h$

- ✓ We will now derive the stability regions of the numerical methods in the $\lambda_R h$ vs. $\lambda_I h$ - plane and compare that with the stability region of the analytical solution.



Stability region of the analytical solution of the model problem in the $\lambda_R h$ vs. $\lambda_I h$ - plane

Stability: Multi-Step Methods (explicit) Example

- ✓ Recall: The stability region (in the $\lambda_R h$ vs. $\lambda_I h$ - plane) of a numerical method is an open set C that contains the collection of those λh for which $|\sigma| < 1$. Boundary of the region is often characterized by $|\sigma| = 1$.
- ✓ Since, σ is a function of λh, σ is complex: $\sigma = \Lambda e^{i\theta}$

$$\left|e^{i\theta}\right| = 1 \quad \Longrightarrow \quad |\sigma| < 1 \quad \Longrightarrow \quad |\Lambda| < 1$$

- ✓ **Euler Forward:** applying the method to the model problem,

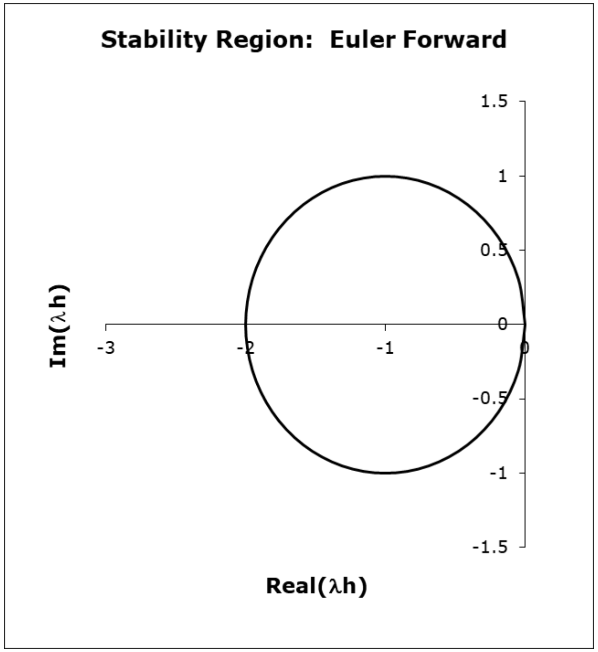
$$y_{n+1} = y_n + \lambda h y_n = (1 + \lambda_R h + i\lambda_I h) y_n = \sigma y_n \quad \Longrightarrow \quad y_n = \sigma^n y_0$$

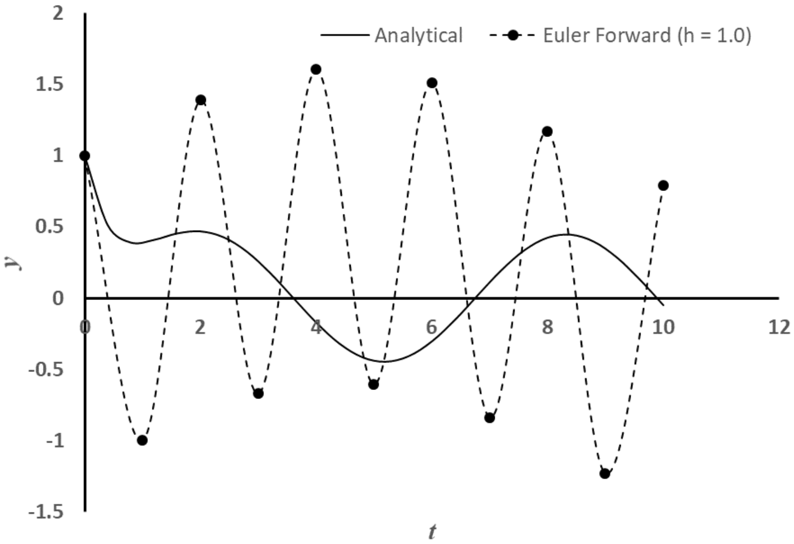
$$|\sigma| = |(1 + \lambda_R h) + i\lambda_I h| < 1 \quad \Longrightarrow \quad (1 + \lambda_R h)^2 + (\lambda_I h)^2 < 1$$

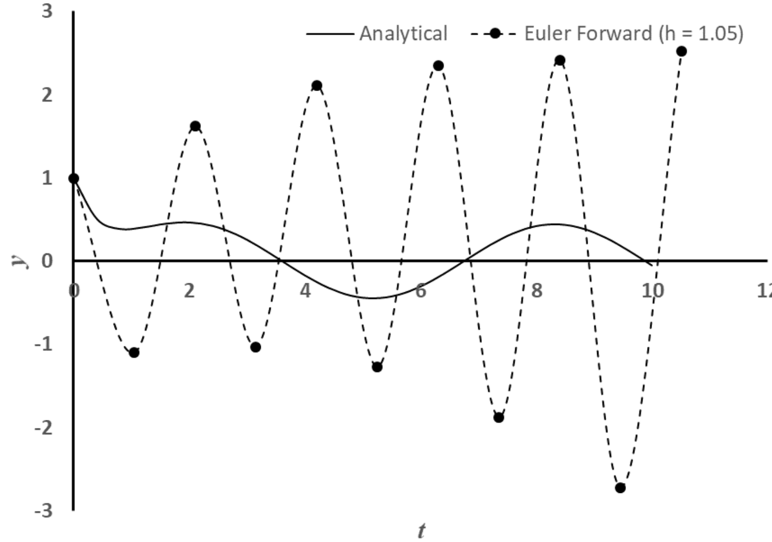
- ✓ For real λ: $\lambda_I = 0$; $\lambda_R = -\lambda$ (necessary for stability)

$$0 < h < \frac{2}{\lambda}$$

Stability: Multi-Step Methods (explicit) Example







Consider our problem:

$$\frac{dy}{dt} = -2y + \sin t$$

For stability with Euler forward, h < 1

Stability: Multi-Step Methods (explicit) Example

For higher order methods, let's consider 3rd order Adams-Bashforth:

- ✓ **Adams-Bashforth (3rd Order):** applying the method to the model problem,

$$y_{n+1} = y_n + h\left(\frac{23}{12}\lambda y_n - \frac{4}{3}\lambda y_{n-1} + \frac{5}{12}\lambda y_{n-2}\right)$$

$$\lambda h = \lambda_R h + i\lambda_I h = \frac{\sigma^3 - \sigma^2}{\left(\frac{23}{12}\sigma^2 - \frac{4}{3}\sigma + \frac{5}{12}\right)}$$

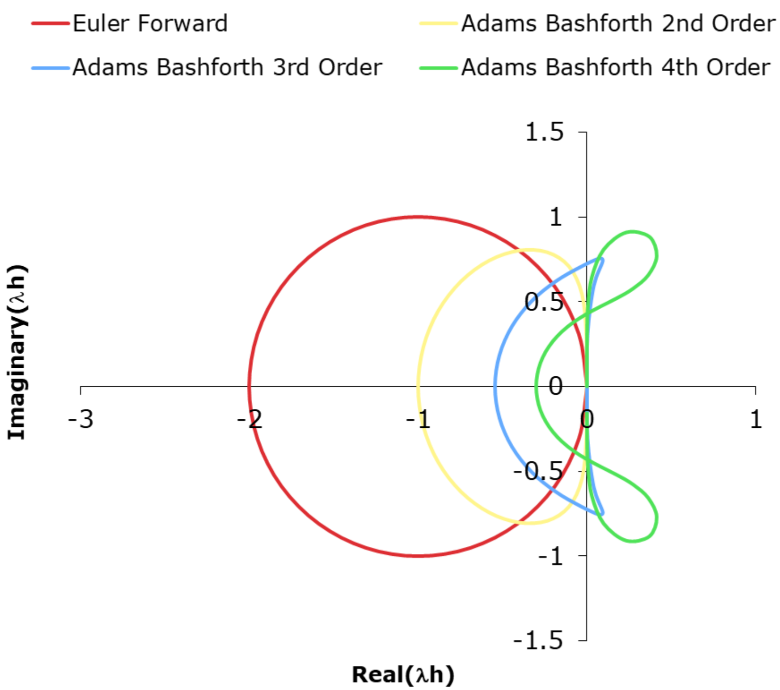
- ✓ In order to plot the region, recall that the boundary of the region is characterized by $|\sigma| = 1$ or $\sigma = e^{i\theta}$

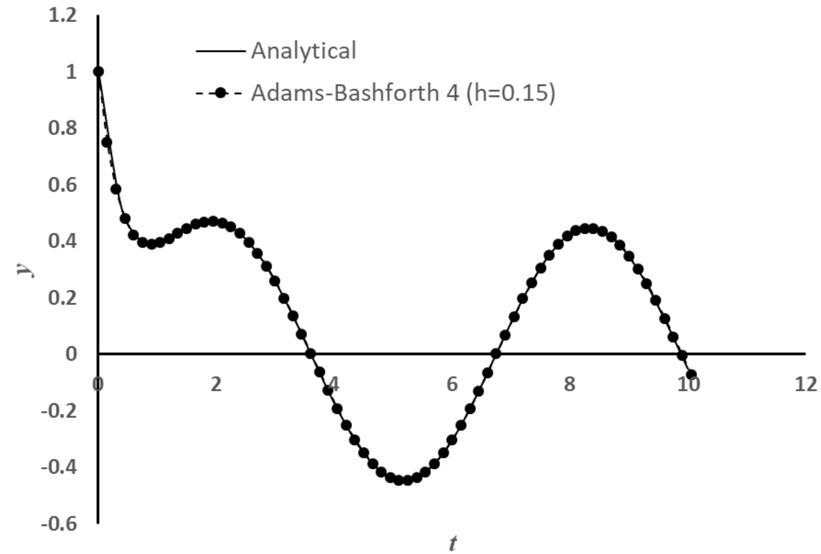
$$\lambda_R h + i\lambda_I h = \frac{(\cos 3\theta - \cos 2\theta) + i(\sin 3\theta - \sin 2\theta)}{\left(\frac{23}{12}\cos 2\theta - \frac{4}{3}\cos\theta + \frac{5}{12}\right) + i\left(\frac{23}{12}\sin 2\theta - \frac{4}{3}\sin\theta\right)}$$

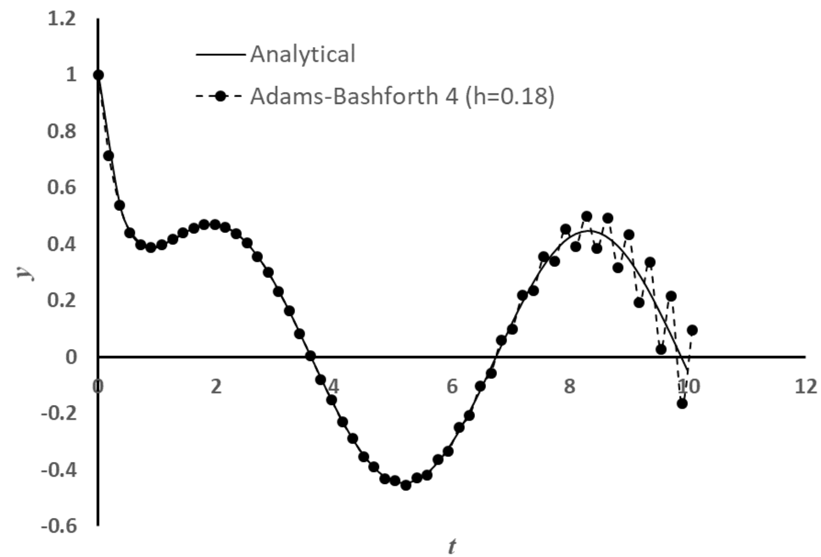
- ✓ One can now easily compute $\lambda_R h$ and $\lambda_I h$ for $\theta \in (0, 2\pi)$

Let's compare the stability regions of all the explicit multi-step methods!

Stability: Multi-Step Methods (explicit) Example







All explicit methods are ***conditionally stable***.

Consider our problem:

$$\frac{dy}{dt} = -2y + \sin t$$

For stability with Adams-Bashforth (4^{th} Order), $h < 0.3/2 = 0.15$