Gernot Hoffmann

# Simple Integration Algorithms

## Contents

# 1. Introduction

The numerical integration of ordinary differential equations is described in many standard text books [1],[2],[3] .
Technical applications require a guaranteed accuracy. This means, the error from one step to the next is in well defined limits for a stable differential equation.

The situation is different for computer games. Mostly, the programmer is interested in fast and plausible solutions, which are not necessarily correct in the sense of mechanics.
For example, an elastic reflection may have a considerably wrong frequency, but it should not add energy. The reflection should not appear as an unstable oscillation even if the stepsize is not small, compared to the period.

Therefore we may judge about the quality of low level integration algorithms by four criteria:

1) Stability of a programmed oscillator
2) Frequency deviation of the programmed oscillator
3) Speed, expressed by the number of calculations per step
4) Minimal allowed stepsize for the time

The expressions ’False Euler’, ’False Heun’ and ’Semi-False Heun’ are not common.
They are defined by the author, and the algorithms cannot be found in standard textbooks.

Settings in Acrobat Professional and

Edit > Preferences > Page Display > Custom
Resolution 72 dpi and view by zoom 100%

Edit > Preferences > Color Management > Custom
RGB sRGB
CMYK ISO Coated
Gray Black Ink - ISO Coated
derived from ISO Coated in Photoshop

# 2. Nomenclature

| | |
|---|---|
| m | Mass |
| c | Spring constant |
| d | Viscous drag coefficient |
| D | Damping ratio |
| | |
| t | Physical time |
| s | Variable for the Laplace Transformation |
| z | Variable for the z-Transformation |
| j | Imaginary unit |
| | |
| **x** | Input vector |
| **y** | State vector |
| **z** | Auxiliary vector |
| | |
| **A**,**B**,**C**,**D** | Matrices for state space equations |
| | |
| T | Sampling timestep |
| | |
| $\omega_o$ | Frequency of a harmonic oscillator |
| $T_o$ | Period of the harmonic oscillator |
| | |
| $\Omega_o$ | Frequency in the numerical solution |
| | |
| $\Phi = \omega_o T$ | True angle increment |
| $\Psi = \Omega_o T$ | Angle increment in the numerical solution |

# 3. State Space Equations

A system with the input **x**(t) can be described by a set of n first order differential equations for the state variable **y**(t) and m ordinary equations for the auxiliary variable **z**(t) which includes output variables and any other convenient variables. Initial conditions are $\mathbf{y}(0)=\mathbf{y}_o$ .

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, \mathbf{x}, t)$$
$$\mathbf{z} = \mathbf{g}(\mathbf{y}, \mathbf{x}, t)$$

For linear differential equations with constant coefficients the system can be written in matrix form:

$$\dot{\mathbf{y}} = \mathbf{A}\,\mathbf{y} + \mathbf{B}\,\mathbf{x}$$
$$\mathbf{z} = \mathbf{C}\,\mathbf{y} + \mathbf{D}\,\mathbf{x}$$

Auxiliary variables **z**(t) are essential, if the system contains transfer functions $F_{ik}(s)=Y_i(s)/X_k(s)$ in the frequency domain (Laplace Transformation), with *equal* degrees of numerator and denominator.
Because we are talking here about very simple systems, we may use the simplified formulation for only one input x and n state variables. **B** is then a column matrix.

$$\dot{\mathbf{y}} = \mathbf{A}\,\mathbf{y} + \mathbf{B}\,x$$

The most important test example is the oscillator, using the coordinate y, the mass m, the spring constant c, the viscous drag coefficient d and an external force F(t):

$$m\,\ddot{y} = -c\,y - d\,\dot{y} + c\,x + F(t)$$

This can be normalized by the frequency $\omega_o^2 = c/m$, the damping ratio D and the acceleration a(t) due to the force F(t) :

$$\ddot{y} + 2D\omega_o\,\dot{y} + \omega_o^2\,y = \omega_o^2\,x + a$$

D=0 means no damping, $D=\varepsilon$ is very little damping and $D\geq 1$ are the aperiodic cases. D=0.707 is often considered as optimal for filters and control systems (4% overshoot in the step response).

The state space representation is possible in many different ways, e.g. using the velocity $y_1=v$ and the coordinate $y_2 = y$. Any linear combination works as well. This is important for canonical forms.

# State Space Equations . . .

Physical form:

$$\dot{v} = -\omega_o^2 y - 2D\omega_o v + \omega_o^2 x + a$$
$$\dot{y} = v$$

Canonical form:

$$\dot{v} = -\omega_o (y + 2D v - x) + a/\omega_o$$
$$\dot{y} = \omega_o v$$

The canonical form uses a scaled velocity $v := v/\omega_o$ instead of the physical velocity. For $a = 0$ this is better for tests, because the phase diagram v(y) is a circle for the undamped oscillator. The length of the state vector (v,y) is independent of the frequency.
In other words: $\omega_o$ is the same as a scale factor for the time, which can be easily shown by replacing $d/dt$ by a scaled time $d/dt'$ , using $t' = \omega_o t$ .

Later we can apply the damping $2D\omega_o v$ for two totally different purposes:

a) true physical damping, as usual
b) artificial stabilization for otherwise unstable integrations

# 4. z-Transformation

A simple integration by Standard Euler replaces d/dt by d/T, where T is the finite sampling time:

$$\mathbf{dy} = T\,(\mathbf{A\,y} + \mathbf{B\,x})$$
$$\mathbf{y} = \mathbf{y} + \mathbf{dy}$$
$$\mathbf{y} = \mathbf{y} + T(\mathbf{A\,y} + \mathbf{B\,x})$$

It is very important to note, that the new value on the left side is calculated by old values on the right side.
We can write this alternatively by

$$\mathbf{y}_i = \mathbf{y}_{i-1} + T(\mathbf{A\,y}_{i-1} + \mathbf{B\,x}_{i-1}).$$

The shift of one sample distance (-1) is indicated by the operator $z^{-1}$ . This has nothing to do with the previously mentioned auxiliary variable **z**.

$$z^{-1} = e^{-sT}$$

This is the delay operator in the Laplace transformation, but a deeper understanding is here not necessary. Therefore we can write with an appropriate identity matrix **I**:

$$\mathbf{y}_i = (\mathbf{I} + T\mathbf{A})\,\mathbf{y}_i\,z^{-1} + T\,\mathbf{B}\,\mathbf{x}_i\,z^{-1}$$

$$\mathbf{y} = \mathbf{P}\,\mathbf{y}\,z^{-1} + \mathbf{Q}\,\mathbf{x}\,z^{-1}$$

**P** is the transition matrix, which describes the recursion algorithm. For the homogeneous case $\mathbf{x}=\mathbf{0}$ we have merely

$$\mathbf{y} = \mathbf{P}\,\mathbf{y}\,z^{-1}$$

$$\mathbf{I}\,\mathbf{y} = \mathbf{P}\,\mathbf{y}\,z^{-1}$$

$$(\mathbf{I}\,z - \mathbf{P})\,\mathbf{y} = \mathbf{0}$$

Following textbooks about sampled data systems, e.g. [4], this is the basis for the calculation of the eigenvalues $z_1, z_2, \ldots, z_n$ of the transition matrix **P** by the characteristic equation, an n-th order polynomial in z:

$$\det(\mathbf{I}\,z - \mathbf{P}) = 0$$

# z-Transformation . . .

The stability of all solutions of the sample sequence $y_i$ at $t = 0, T, 2T, \ldots iT$ for one of the variables in **y**, starting with initial values $\mathbf{y}_o$, is entirely described by the location of the eigenvalues in the complex plane.
A sample sequence consists of isolated numbers $y_i$. It is convenient to assign to each variable a continuous function g(t) from which the samples could have been taken, but this function is an artefact and not unique.
Then we may define „stability“ by the behaviour of g(t):

Single real eigenvalues $z_1 = a_1$:

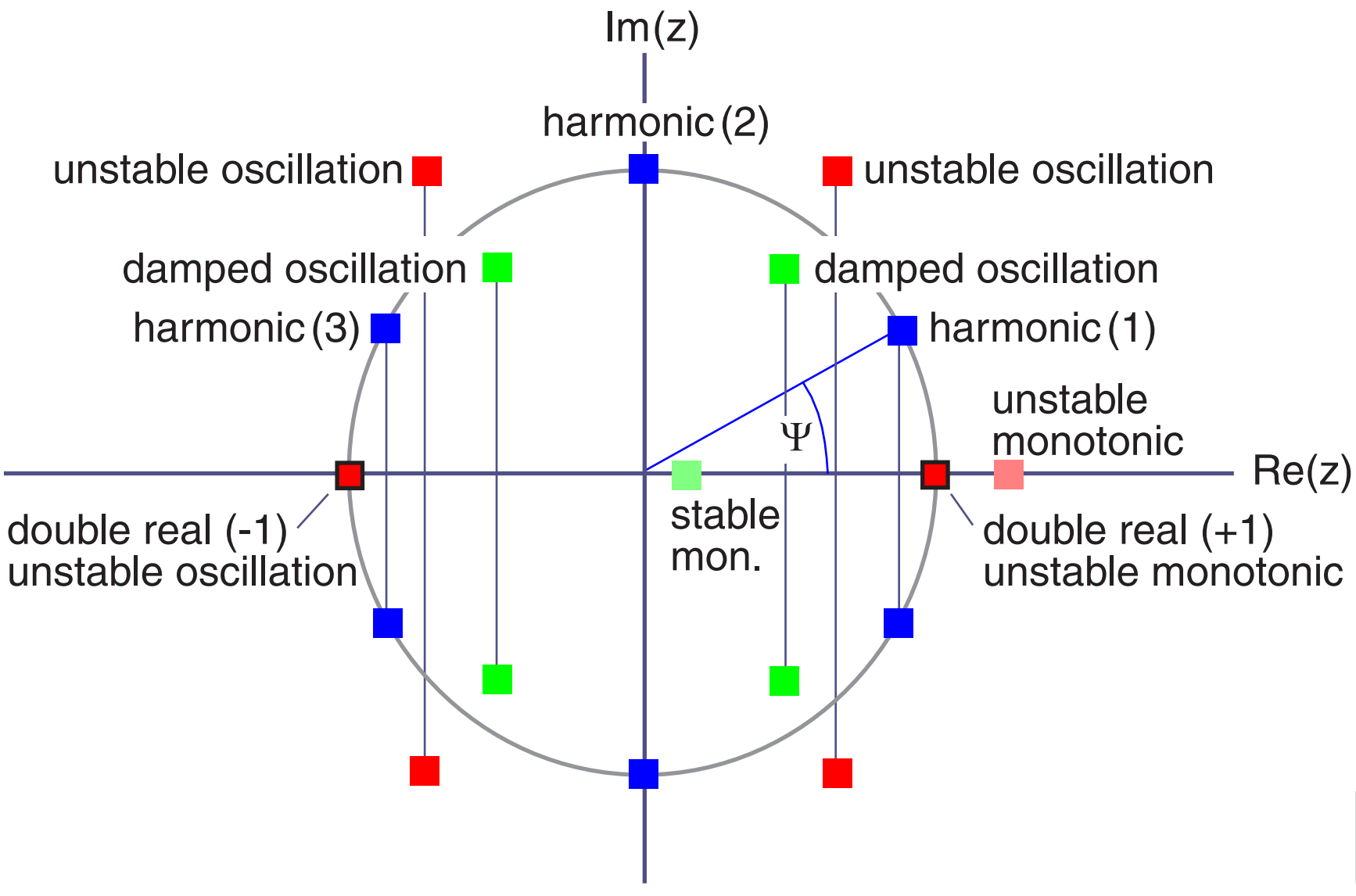
| | |
|---|---|
| $0 < z_1 < 1$ | monotonic stable g(t) |
| $z_1 = 1$ | constant g(t) |
| $1 < z_1$ | monotonic unstable g(t) |

Single pairs of conjugate complex eigenvalues $z_{1/2} = a_1 \pm j\, b_1$ :

g(t) is an oscillation with the effective frequency $\Omega_o$ .

$r_1 = (a_1^2 + b_1^2)^{0.5}$
$\Psi_1 = \arctan(b_1/a_1) = \Omega_o T$ angle increment, $0 \ldots \pi$

| | | | |
|---|---|---|---|
| $r_1$ | = | 1 | harmonic g(t) |
| $r_1$ | < | 1 | damped oscillation g(t) |
| $r_1$ | > | 1 | unstable oscillation g(t) |



# z-Transformation . . .

The harmonic cases 1,2,3 differ by the number of samples per period of the actual function g(t):

1 360 / 30 = 12 samples
2 360/ 90 = 4 samples
3 360/150 = 2.4 samples

Double real eigenvalues:

$z_1 = z_2 = 1$ monotonic unstable g(t)
$z_1 = z_2 = -1$ unstable oscillation g(t)

If $z_{1,}\, z_2$ depend on a parameter and we change the parameter in a way that a single pair of complex eigenvalues with $r_1 = 1$ approaches $z_1 = z_2 = -1$, then the result is not a harmonic oscillation with $\Psi_1 = \pi$ but an instable oscillation. This is somewhat surprising, compared to the behaviour of eigenvalues $s = \sigma + j\omega$ in the Laplace frequency domain.

Major result:
The stability of a sample sequence for a homogeneous test case can be predicted by the eigenvalues of the transition matrix.

We encounter often this characteristic equation:

$$z^2 - 2\,p\,z + q = 0$$

$$z_{1/2} = p \pm (-q + p^2)^{0.5} \qquad \text{real}$$

$$z_{1/2} = p \pm j\,(q - p^2)^{0.5} \qquad \text{conjugate pair}$$

$$r_1^2 = p^2 + (q - p^2) = q$$

Therefore we can test the harmonic case $r_1 = 1$ for conjugate pairs simply by $q = 1$.

# 5. Standard Euler

Tests are performed for the homogeneous harmonic oscillator.

$$\dot{v} = -\omega_o y$$
$$\dot{y} = \omega_o v$$

$$dv = -\omega_o T\, y = -\Phi y$$
$$dy = \omega_o T\, v = \Phi v$$

$$v = v + dv = v - \Phi y$$
$$y = y + dy = y + \Phi v$$

$$v = v\, z^{-1} - \Phi y\, z^{-1}$$
$$y = \Phi v\, z^{-1} + y\, z^{-1}$$

Characteristic equation:

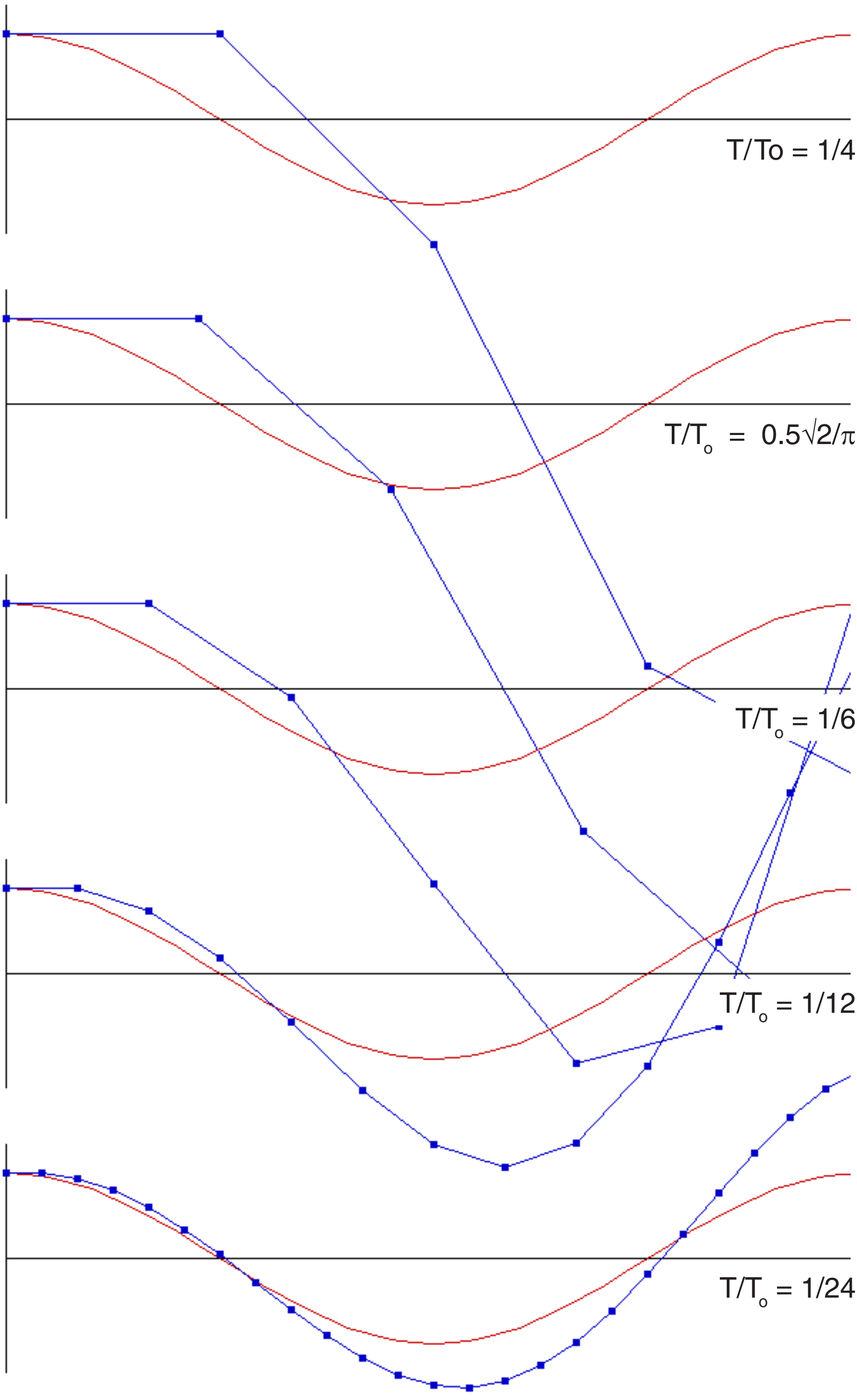
$$z^2 - 2z + (1+\Phi^2) = 0$$

$$z_{1/2} = 1 \pm j\,\Phi$$

$$r_1^{\,2} = 1 + \Phi^2$$

The solution is always an unstable oscillation.

# Standard Euler y(t)



# 6. Damped Euler

Tests are performed for the homogeneous harmonic oscillator, but we add an artificial damping.

$$\begin{aligned} \dot{v} &= -\omega_o (2Dv + y) \\ \dot{y} &= \omega_o v \end{aligned}$$

$$\begin{aligned} dv &= -\omega_o T (2Dv + y) = -\Phi(2Dv + y) \\ dy &= \omega_o T v = \Phi v \end{aligned}$$

$$\begin{aligned} v &= v + dv = v - \Phi (2Dv + y) \\ y &= y + dy = y + \Phi v \end{aligned}$$

$$\begin{aligned} v &= (1 - \Phi 2D)v z^{-1} - \Phi y z^{-1} \\ y &= \Phi v z^{-1} + y z^{-1} \end{aligned}$$

Characteristic equation

$$z^2 - 2(1 - D\Phi) z + (1 - 2D\Phi + \Phi^2) = 0$$

$$z_{1/2} = (1 - D\Phi) \pm j \Phi(1 - D^2)^{0.5}$$

$$r_1^2 = 1 - 2D\Phi + \Phi^2$$

The artificial damping D has only the purpose of stabilizing the solution of an undamped oscillator numerically, by $r_1^2 = 1$:

$$D = 0.5 \Phi$$

Because the sample sequence is now harmonic, we can calculate the true frequency $\Omega_o$ by using $\Psi = \Omega_o T$. The frequency of the sample sequence is always too high.

$$\tan\Psi = \Phi(1 - D^2)^{0.5} / (1 - D\Phi) = \Phi(1 - \Phi^2/4)^{0.5} / (1 - \Phi^2/2)$$

An approximation is calculated by Taylor series (Appendix A):

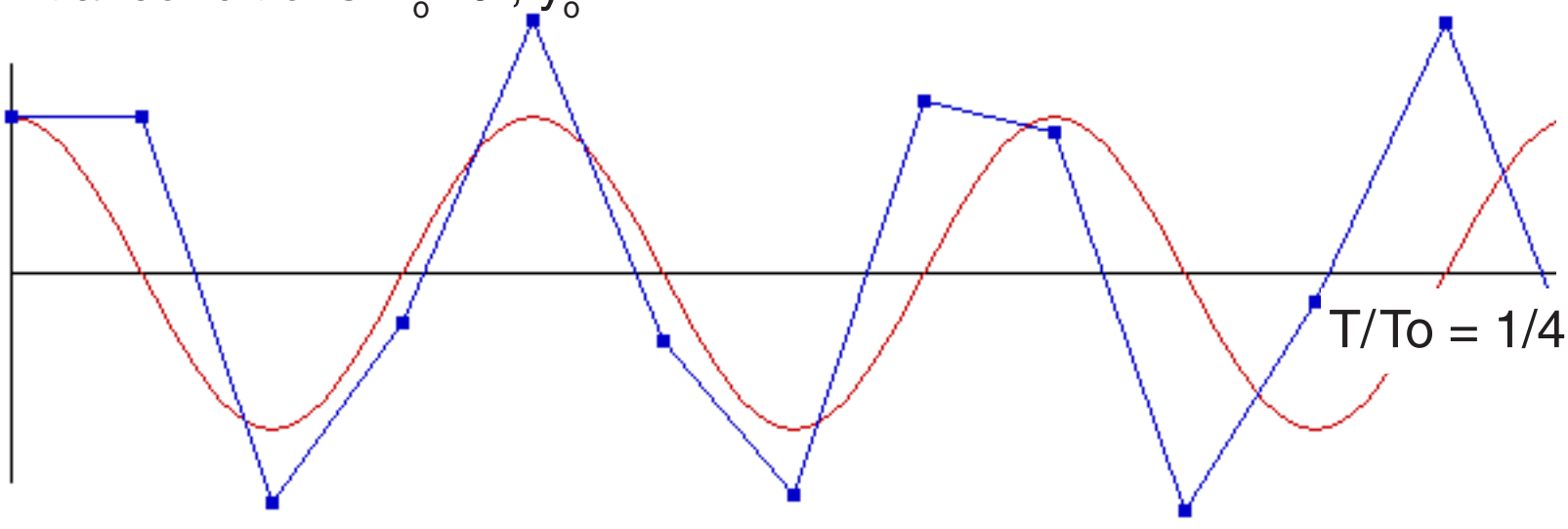
$$\Psi \cong \Phi(1 + \Phi^2/24)$$

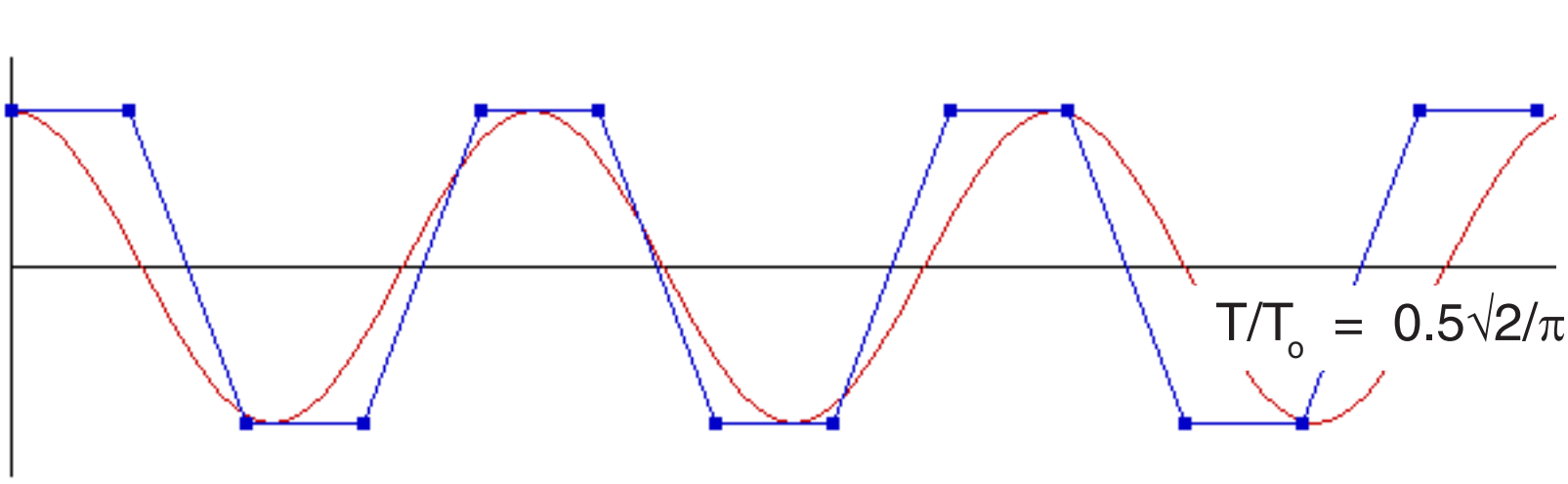
The approximation is valid for $\Phi < 1$ or more than $2\pi$ samples per period.

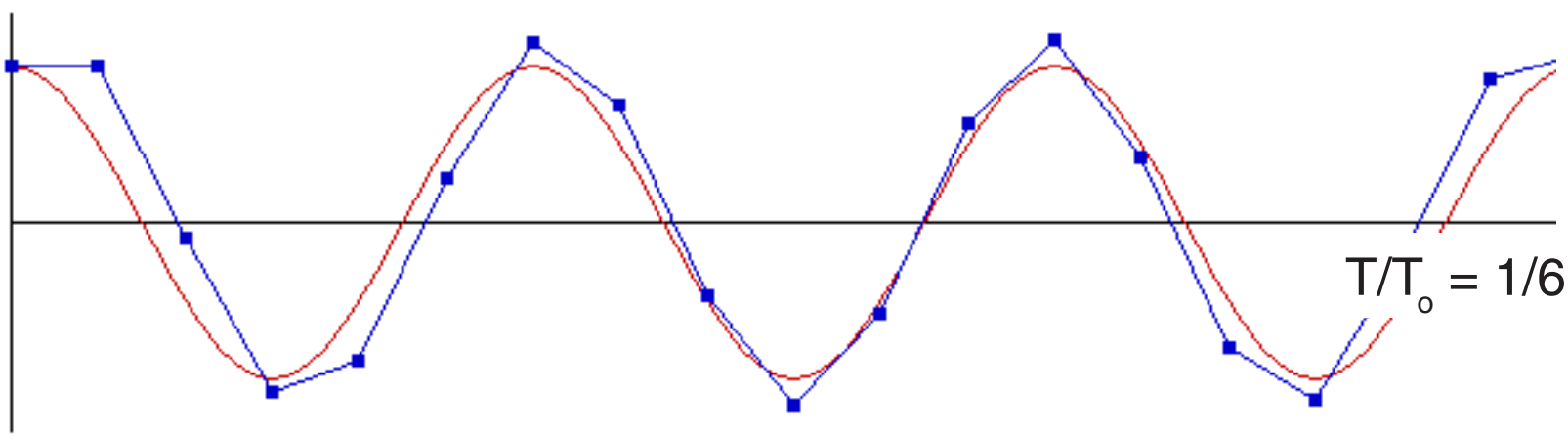
| | | |
|---|---|---|
| $\Phi = 1$ | True $\Psi = 1.047$ | Approximation $\Psi = 1.042$ |
| $\Phi = 1.414$ | True $\Psi = 1.570$ | Approximation $\Psi = 1.532$ |

# Damped Euler y(t)

Initial conditions: $v_o=0$ , $y_o=1$

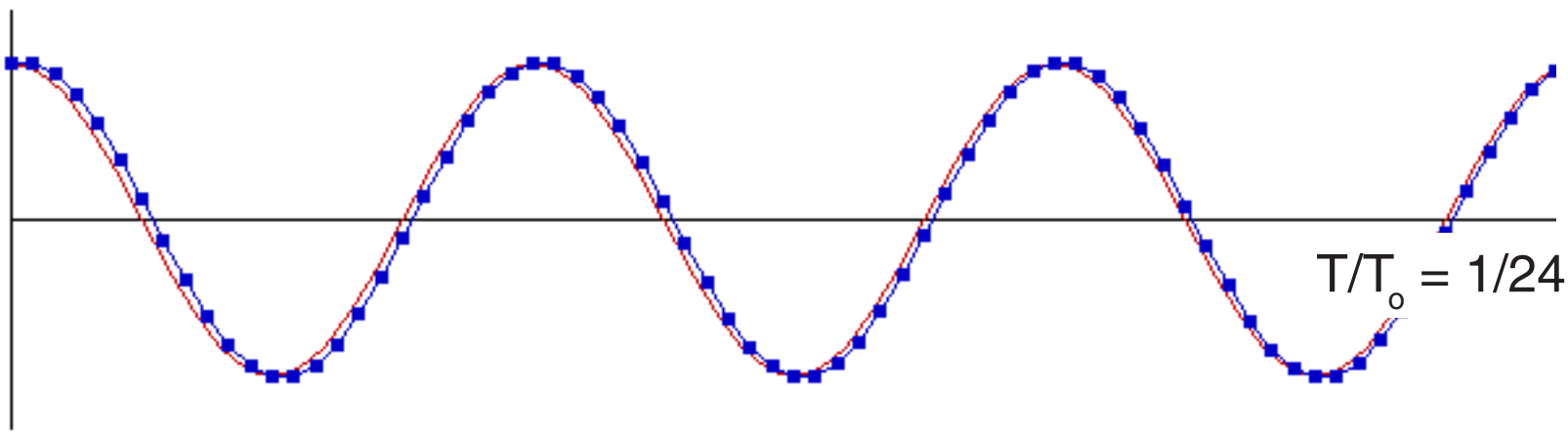








# 7. False Euler

Tests are performed for the homogeneous harmonic oscillator.
The method is called 'false', because the second equation uses the new value v from the first equation.

$$\dot{v} = -\omega_o y$$
$$\dot{y} = \omega_o v$$

$$v = v - \Phi y$$
$$y = y + \Phi v$$

$$v = v z^{-1} - \Phi y z^{-1}$$
$$y = \Phi v z^{o} + y z^{-1} = \Phi v z^{-1} + (1 - \Phi^2) y z^{-1}$$

Characteristic equation:

$$z^2 - 2(1 - \Phi^2/2) z + 1 = 0$$

$$z_{1/2} = 1 - \Phi^2/2 \pm j (1 - (1-\Phi^2/2)^2)^{0.5}$$
$$z_{1/2} = 1 - \Phi^2/2 \pm j \Phi( 1- \Phi^2/4 )^{0.5}$$
$$z_{1/2} = \cos\Psi \pm j \sin\Psi$$

$$r_1^2 = 1$$

Surprisingly, the solution is always a harmonic oscillation.

$$\tan\Psi = \Phi (1 - \Phi^2/4 )^{0.5} / (1 - \Phi^2/2)$$

Approximation for the frequency (Appendix A):

$$\Psi \cong \Phi (1 + \Phi^2/24 )$$

The frequency is always too high.

The stability limit results from zero imaginary part of the eigenvalues:

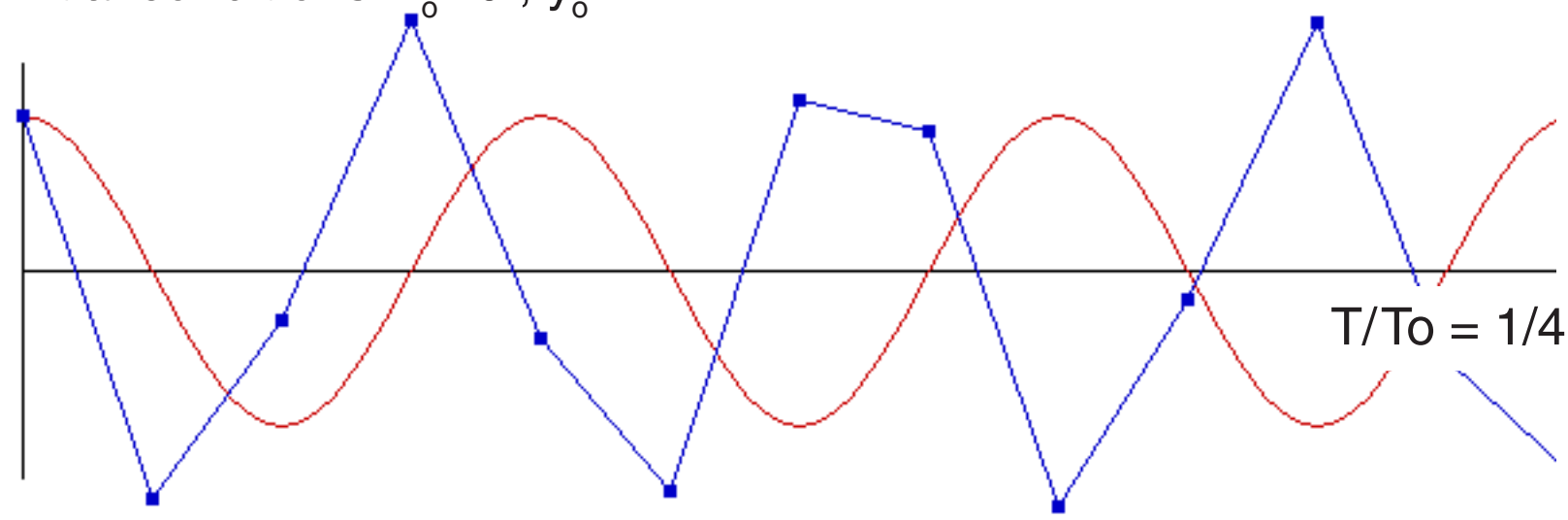
$$\Phi^2 = 4, \quad \omega_o T = 2, \quad 2\pi (T/To) = 2$$

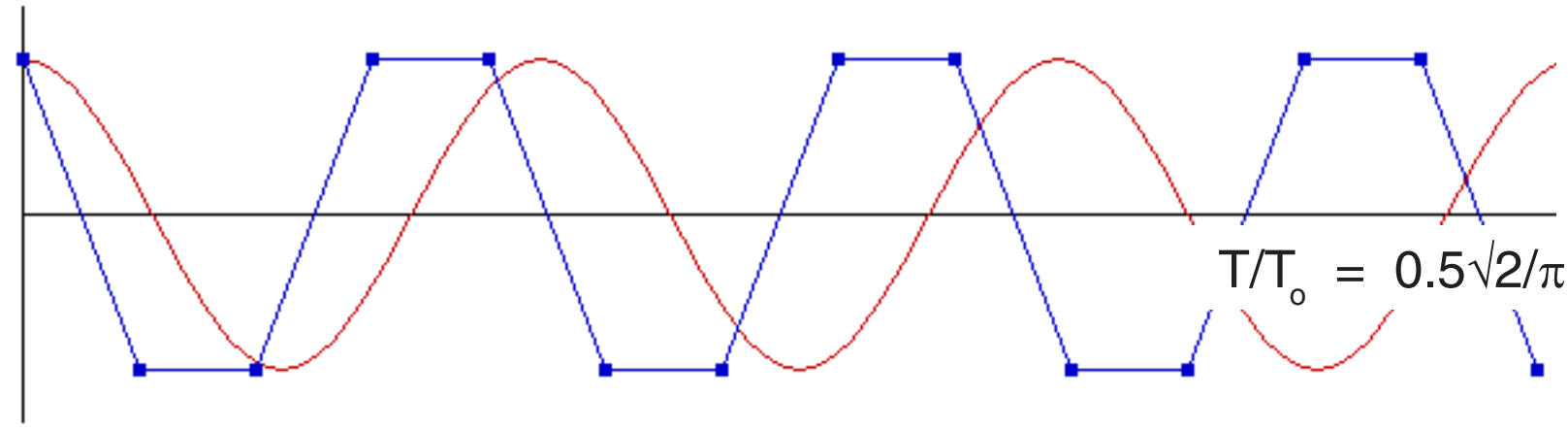
$$T/To = 1/\pi$$

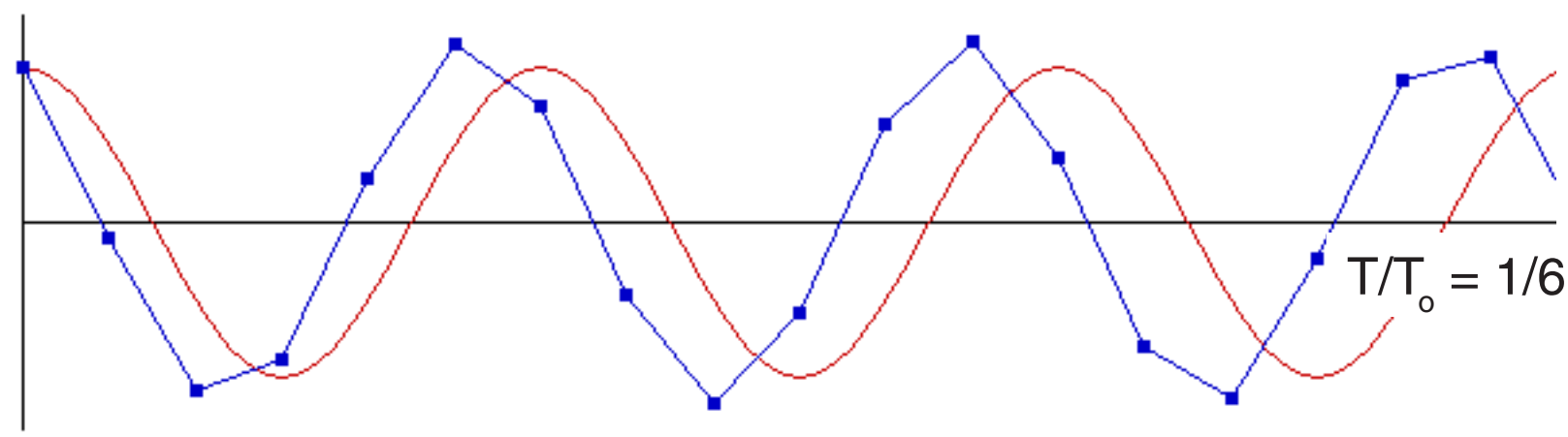
This means: the integration is stable, if we have more than $\pi$ steps per period. This is a very good result, because the theoretical limit is given by the Nyquist criterion: T/To = 1/2.

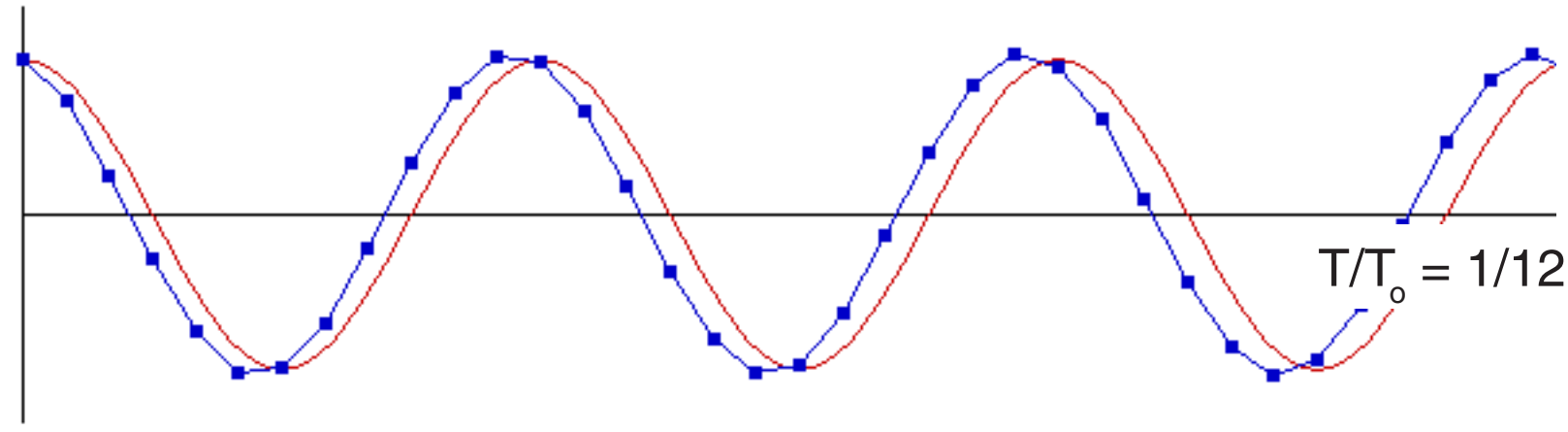
# False Euler y(t)

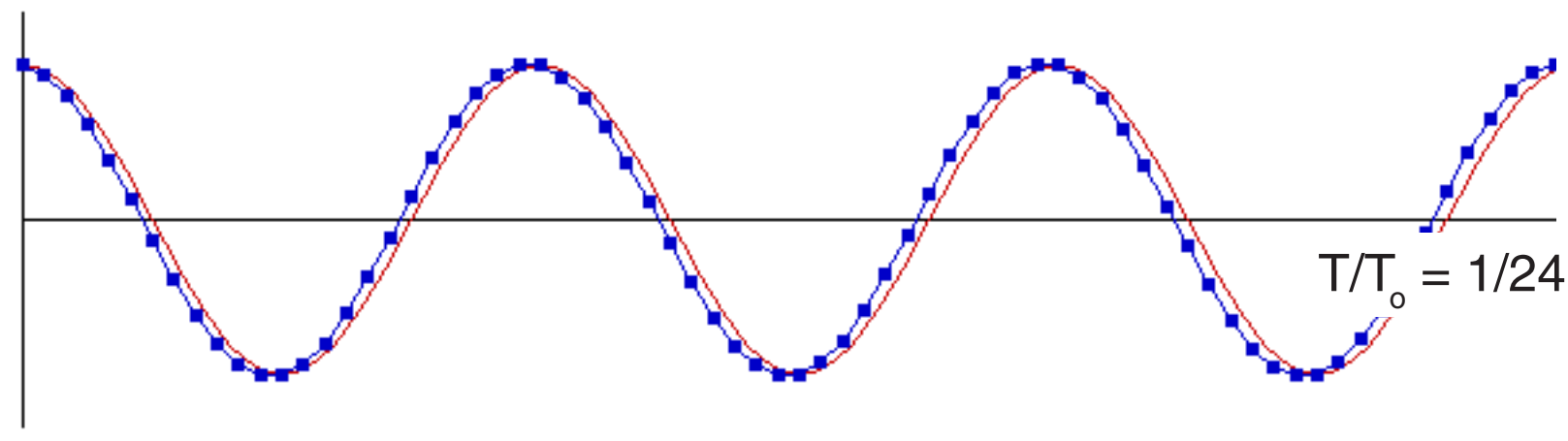
Initial conditions: $v_o=0$ , $y_o=1$



# 8. Verlet Integration

In this Web document [6] the so-called Verlet integration is introduced for the one-dimensional mass point motion:

$$y = a$$

$$y_i = 2y_{i-1} - y_{i-2} + T^2 a_{i-1}$$

The explanation in [6] is not complete, therefore we may prove the algorithm like here. We use the False Euler replacement of v and furtheron a secant differentiation.

$$\dot{v} = a$$
$$\dot{y} = v$$

$$v_i = v_{i-1} + T\, a_{i-1}$$
$$y_i = y_{i-1} + T\, v_i$$

$$v_{i-1} = (y_{i-1} - y_{i-2})/T$$

$$y_i = y_{i-1} + T\,( (y_{i-1} - y_{i-2})/T + T\, a_{i-1} ) = 2y_{i-1} - y_{i-2} + T^2 a_{i-1}$$

Now we apply this to the harmonic oscillator. The acceleration is produced by the spring:

$$a_{i-1} = -\omega_o{}^2\, y_{i-1}$$

$$y_i = 2y_{i-1} - y_{i-2} + T^2(-\omega_o^2 y_{i-1})$$

$$y_i - 2(1-\Phi^2/2)\, y_{i-1} + y_{i-2} = 0$$

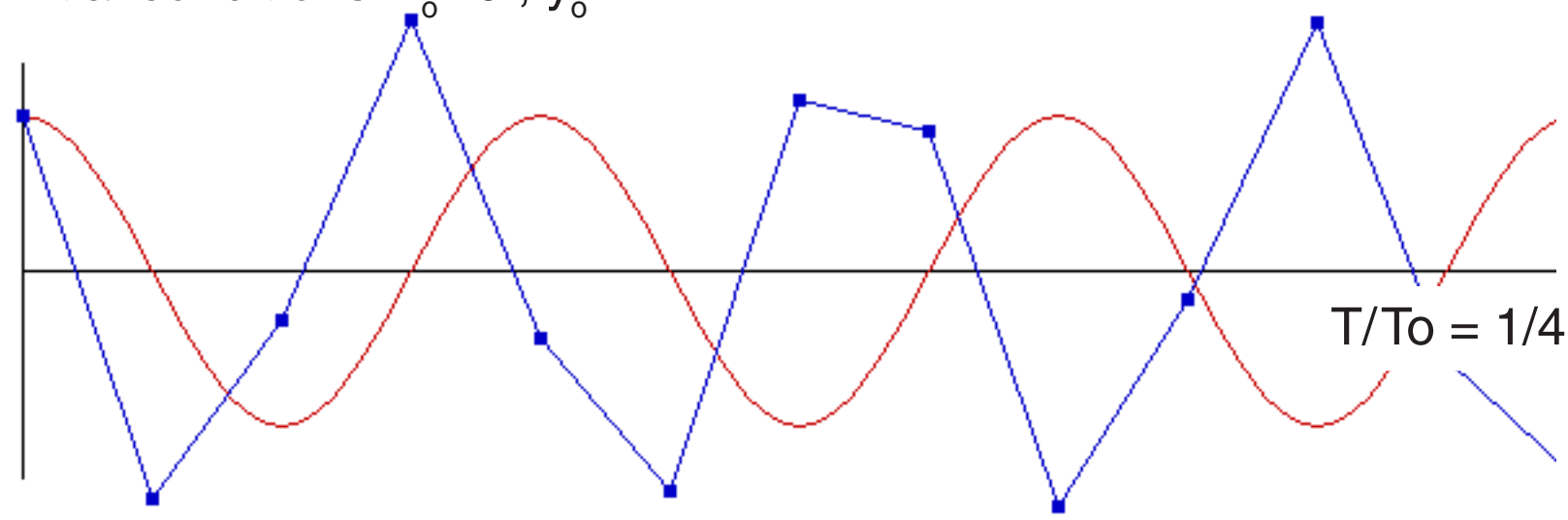
$$z^2 - 2(1 - \Phi^2/2)\, z + 1 = 0$$

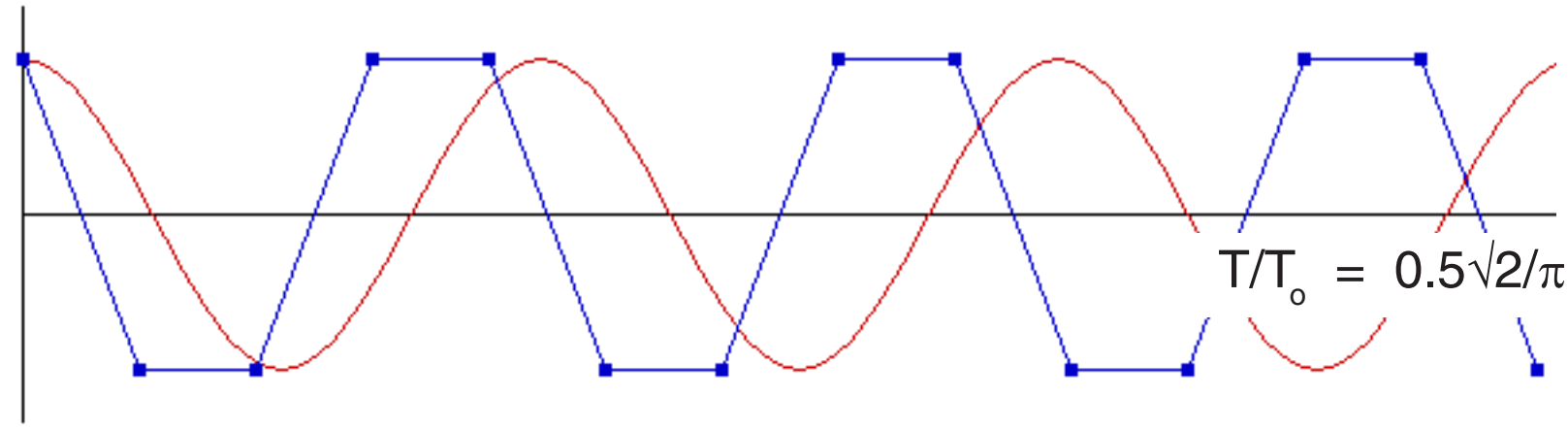
This is the characteristic equation of False Euler – Verlet is the same.

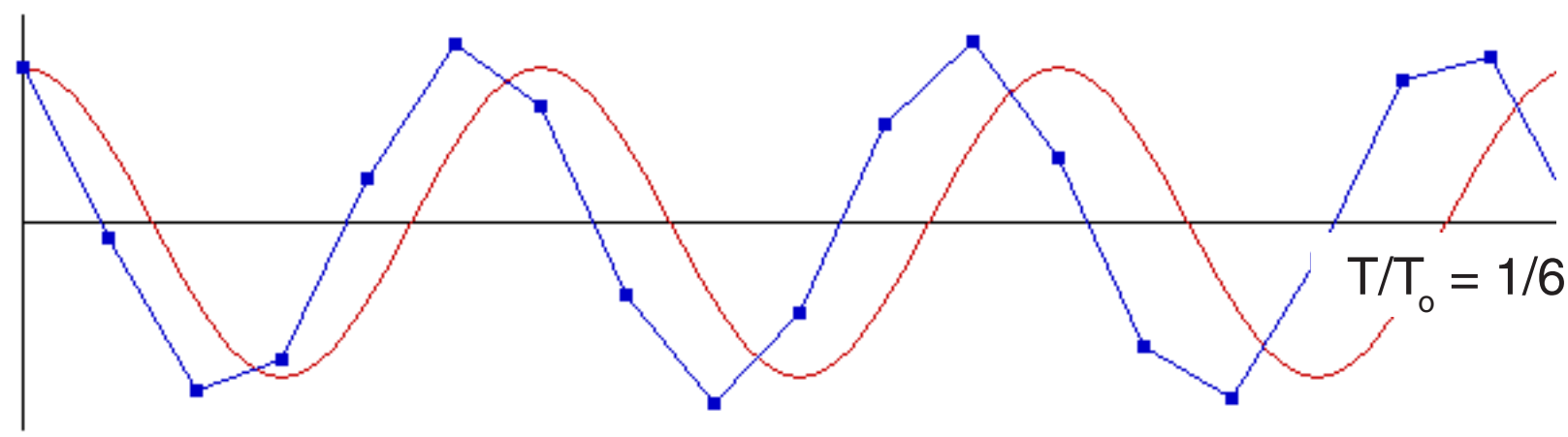
Verlet Integration does not use the state variable v explicitely. This causes big problems for the introduction of the initial condition $v(0) = v_o$ and also for the introduction of physical forces which depend on the velocity, like viscous drag or Coulomb friction. Therefore it is much better to use False Euler. Verlet is a bad formulation of a good algorithm.
The speed depends on the programming. Essentially, both methods need two multiplications and two additions per step.

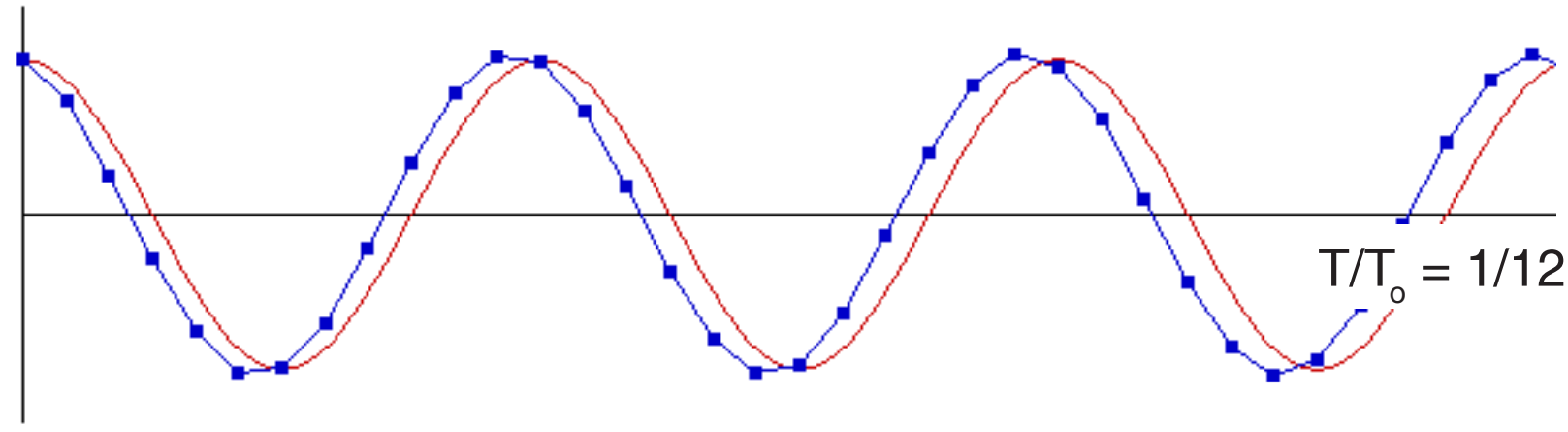
# Verlet Integration y(t)

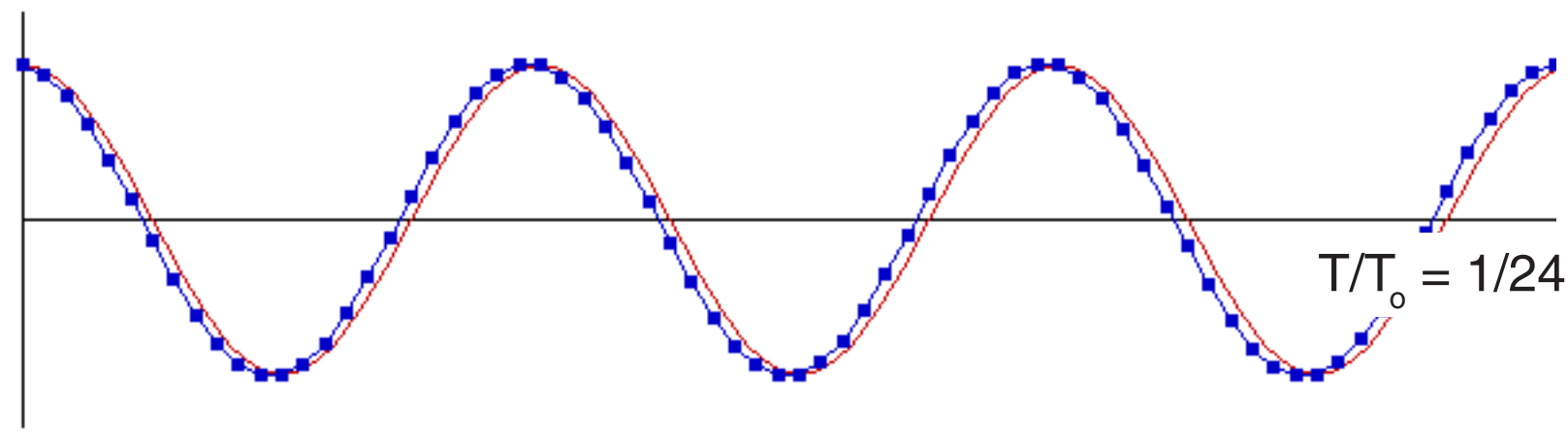
Initial conditions: $v_o=0$ , $y_o=1$



# 9. Standard Heun

General formulation for Standard Heun (Predictor Corrector), here for linear systems:

$$\dot{\mathbf{y}} = \mathbf{A}\,\mathbf{y} + \mathbf{B}\,\mathbf{x}$$

$$\begin{aligned}
\mathbf{dy}^{(1)} &= T\,(\mathbf{A}\,\mathbf{y} + \mathbf{B}\,\mathbf{x}) \\
\mathbf{y}^{(1)} &= \mathbf{y} + \mathbf{dy}^{(1)} \\
\mathbf{dy}^{(2)} &= T\,(\mathbf{A}\,\mathbf{y}^{(1)} + \mathbf{B}\,\mathbf{x}) \\
\mathbf{dy} &= 0.5\,(\mathbf{dy}^{(1)} + \mathbf{dy}^{(2)}) \\
\mathbf{y} &= \mathbf{y} + \mathbf{dy}
\end{aligned}$$

Tests are performed for the homogeneous harmonic oscillator.

$$\begin{aligned}
\dot{v} &= -\omega_o\, y \\
\dot{y} &= \omega_o\, v
\end{aligned}$$

$$\begin{aligned}
dv^{(1)} &= -\Phi\, y \\
dy^{(1)} &= \Phi\, v \\
v^{(1)} &= v - \Phi\, y \\
y^{(1)} &= y + \Phi\, v \\
dv^{(2)} &= -\Phi\,(y + \Phi\, v) \\
dy^{(2)} &= \Phi\,(v - \Phi\, y)
\end{aligned}$$

$$\begin{aligned}
v &= (1 - \Phi^2/2)\, v\, z^{-1} - \Phi\, y z^{-1} \\
y &= \Phi\, v\, z^{-1} + (1 - \Phi^2/2)\, y z^{-1}
\end{aligned}$$

Characteristic equation:

$$z^2 - 2(1 - \Phi^2/2)\, z + (1 + \Phi^4/4) = 0$$

$$z_{1/2} = (1 - \Phi^2/2) \pm j\,\Phi$$

$$r_1^{\,2} = 1 + \Phi^4/4$$

The solution is always an unstable oscillation. The instability is weak for sufficiently small timesteps T. The approximation for $\Psi$ is derived in Appendix A.

$$\begin{aligned}
\tan\Psi &= \Phi\,/\,(1 - \Phi^2/2) \\
\Psi &\cong \Phi \cdot (1 + \Phi^2/6)
\end{aligned}$$

The frequency is always too high.

# Standard Heun y(t)

Initial conditions: $v_o=0$ , $y_o=1$

# 10. False Heun

We may try to apply the same strategy as in False Euler – use the already calculated v in the equation for y:
Tests are performed for the homogeneous harmonic oscillator.

$$\dot{v} = -\omega_o\, y$$
$$\dot{y} = \omega_o\, v$$

$$dv^{(1)} = -\Phi\, y$$
$$v^{(1)} = v - \Phi\, y$$
$$dy^{(1)} = \Phi\, v^{(1)} = \Phi\, v - \Phi^2\, y$$
$$y^{(1)} = y + \Phi\, v - \Phi^2\, y$$
$$dv^{(2)} = -\Phi\,(y + \Phi v - \Phi^2\, y)$$
$$v^{(2)} = v - \Phi\, y - \Phi^2\, v + \Phi^3\, y$$
$$dy^{(2)} = \Phi\,(v - \Phi^2\, v - \Phi\, y + \Phi^3\, y)$$

$$v = v + 0.5(-2\Phi\, y + \Phi^3\, y - \Phi^2\, v)$$
$$y = y + 0.5(2\Phi\, v - \Phi^3\, v - \Phi^2\, y + \Phi^4\, y)$$
$$v = v - \Phi\, y - 0.5\,\Phi^2\, v + 0.5\,\Phi^3\, y$$
$$y = y + \Phi\, v - \Phi^2\, y - 0.5\,\Phi^3\, v + 0.5\,\Phi^4\, y$$

$$v = (1 - \Phi^2/2)\, v\, z^{-1} - \Phi\,(1 - \Phi^2/2)\, y\, z^{-1}$$
$$y = \Phi\,(1 - \Phi^2/2)\, v\, z^{-1} + (1 - \Phi^2/2)\, y\, z^{-1}$$

Characteristic equation:

$$z^2 - 2\,(1 - 3/4\,\Phi^2 + \Phi^4/4)\, z + (1 - \Phi^2/2) = 0$$

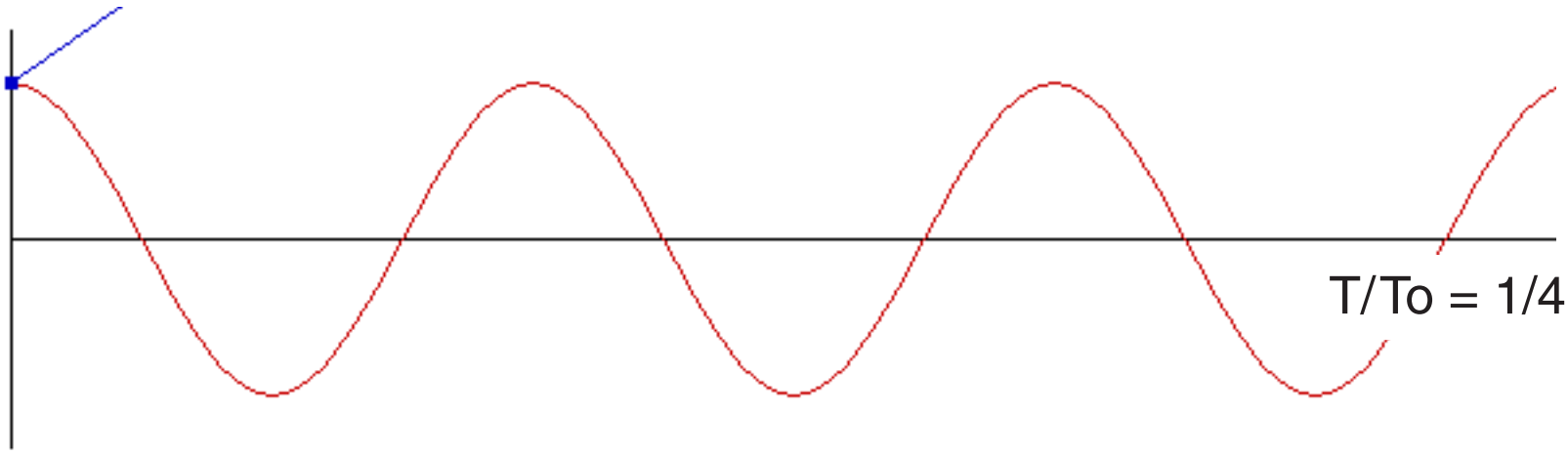
$$r_1^2 = 1 - \Phi^2/2$$

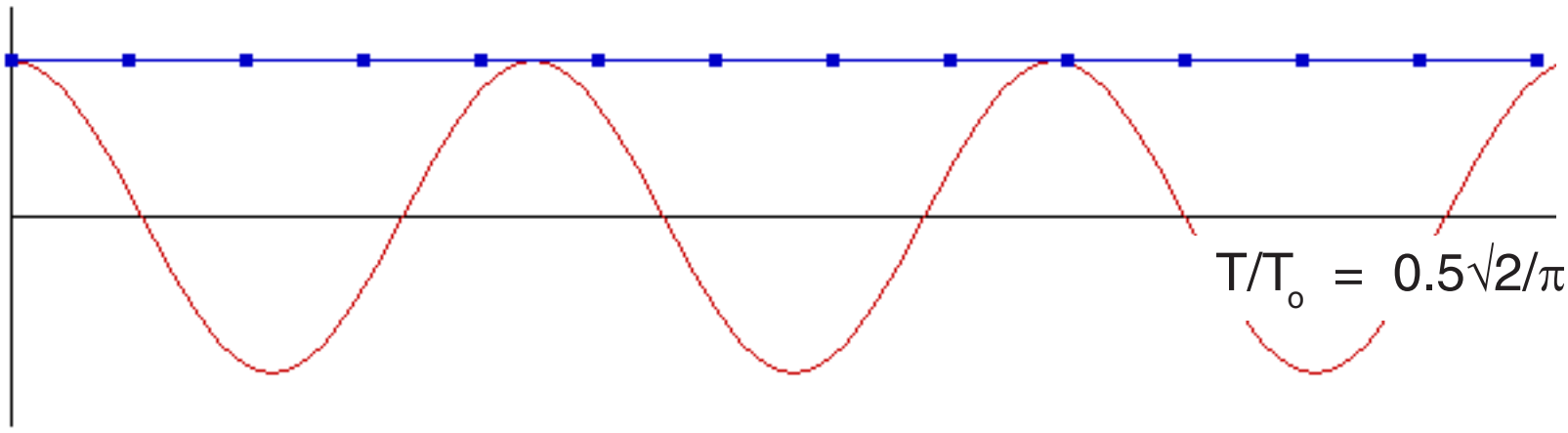
The solution is either monotonic unstable or a damped oscillation.

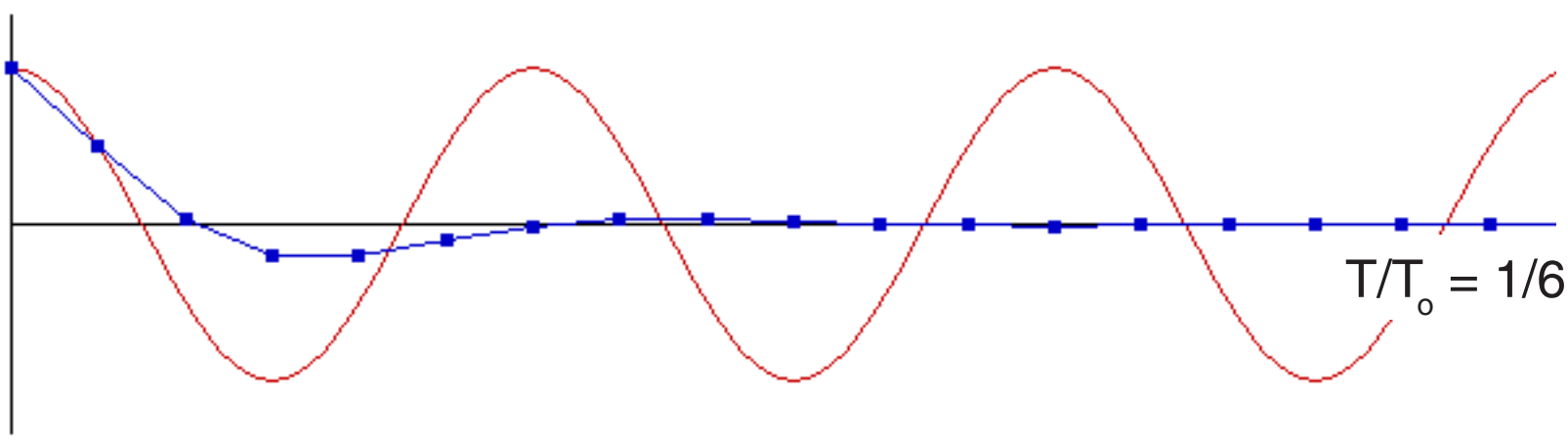
False Heun is not recommended.

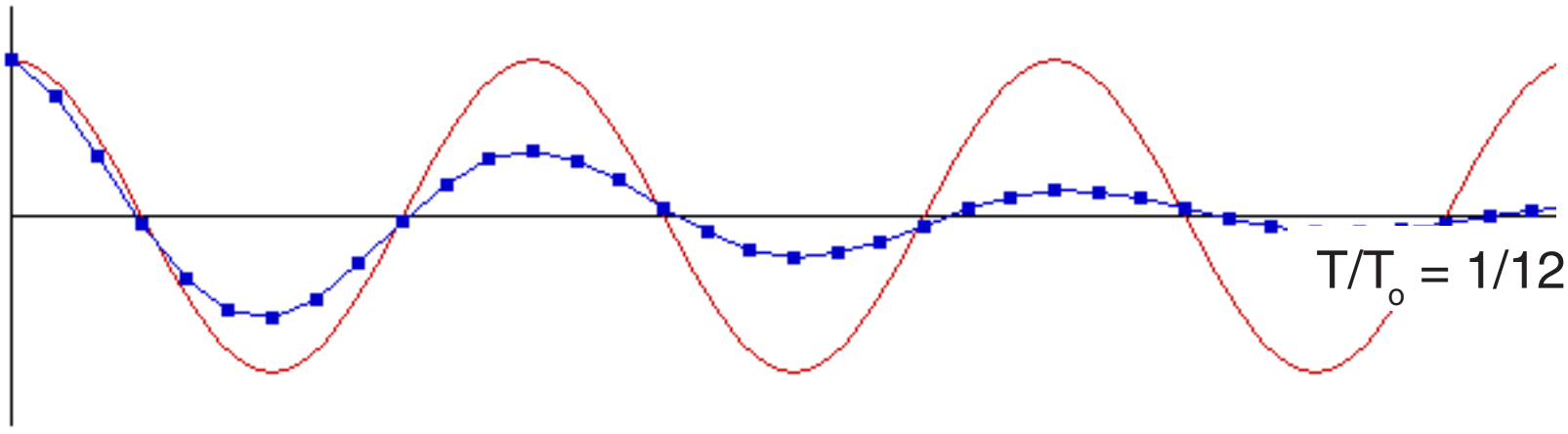
# False Heun y(t)

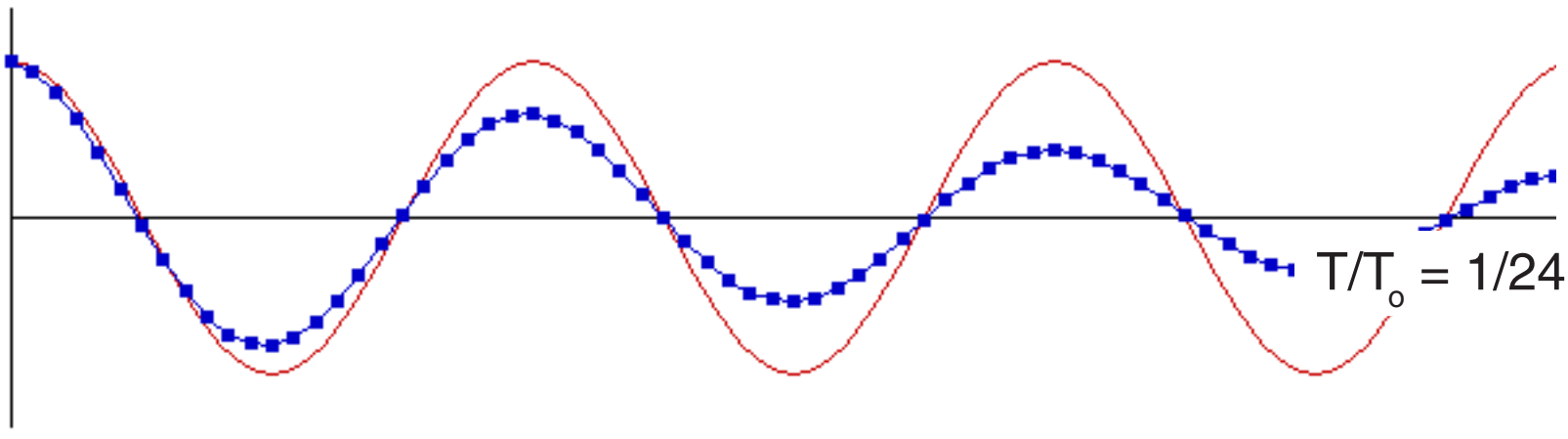
Initial conditions: $v_o=0$ , $y_o=1$



# 11. Semi-False Heun

We may try to apply the same strategy as in False Euler, but now we use the already calculated v only in the second step:
Tests are performed for the homogeneous harmonic oscillator.

$$\dot{v} = -\omega_o\, y$$
$$\dot{y} = \omega_o\, v$$

$$dv^{(1)} = -\Phi\, y$$
$$v^{(1)} = v - \Phi\, y$$
$$dy^{(1)} = \Phi\, v$$
$$y^{(1)} = y + \Phi\, v$$
$$dv^{(2)} = -\Phi\,(y + \Phi\, v)$$
$$v^{(2)} = v - \Phi\, y - \Phi^2\, v$$
$$dy^{(2)} = \Phi\, v^{(2)} = \Phi\,(v - \Phi\, y - \Phi^2\, v)$$

$$v = v + 0.5(-2\Phi\, y - \Phi^2\, v)$$
$$y = y + 0.5(2\Phi\, v - \Phi^2\, y - \Phi^3\, v)$$
$$v = v - \Phi\, y - 0.5\,\Phi^2\, v$$
$$y = y + \Phi\, v - 0.5\,\Phi^2\, y - 0.5\,\Phi^3\, v$$

$$v = (1 - \Phi^2/2)\, v\, z^{-1} - \Phi\, y\, z^{-1}$$
$$y = \Phi(1 - \Phi^2/2)\, v\, z^{-1} + (1 - \Phi^2/2)\, y\, z^{-1}$$

Characteristic equation:

$$z^2 - 2(1 - \Phi^2/2)\, z + (1 - \Phi^4/4) = 0$$

$$z_{1/2} = (1 - \Phi^2/2) \pm j\,\Phi\,(1 - \Phi^2/2)^{0.5}$$

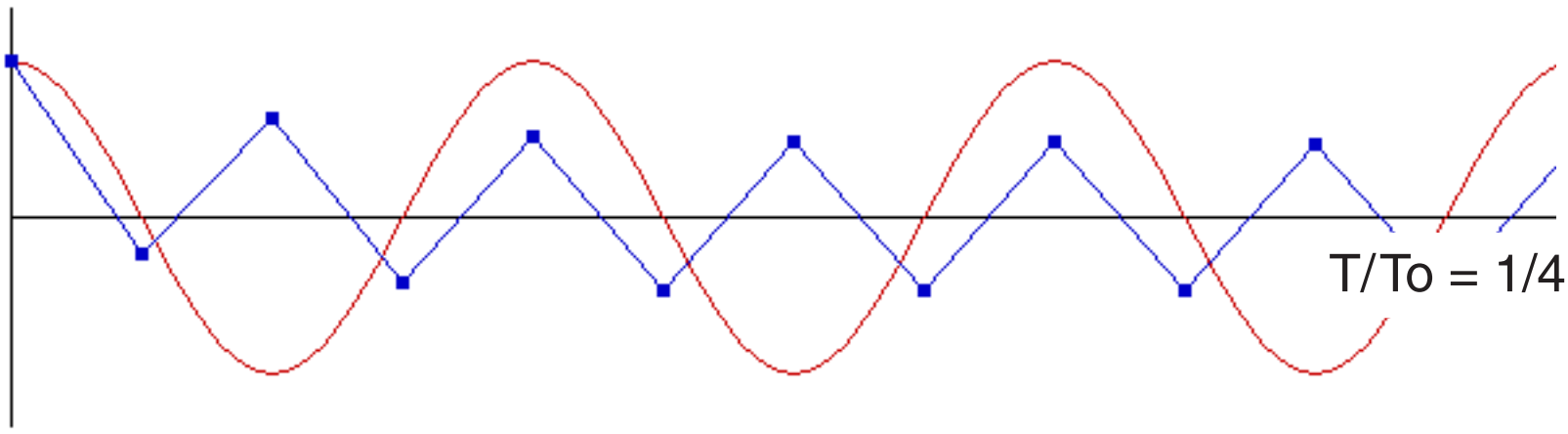
$$r_1^2 = 1 - \Phi^4/4$$

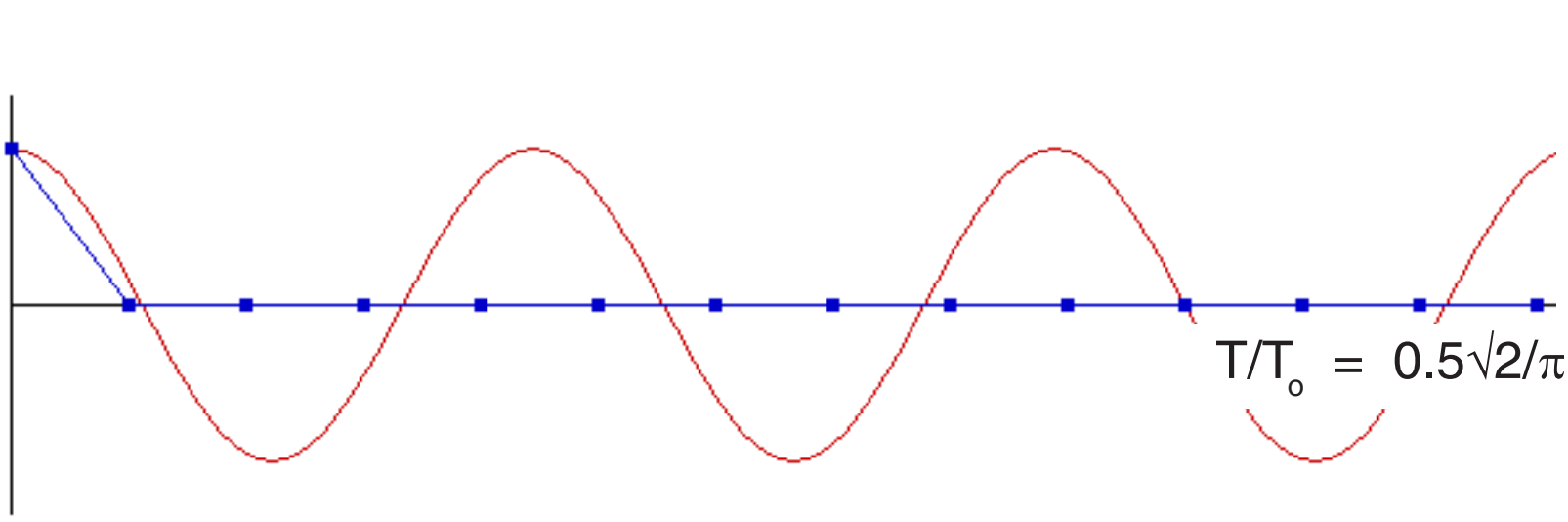
The solution is always stable and for suffiently small timesteps T a damped oscillation, as accurate as Standard Heun (which is unstable).

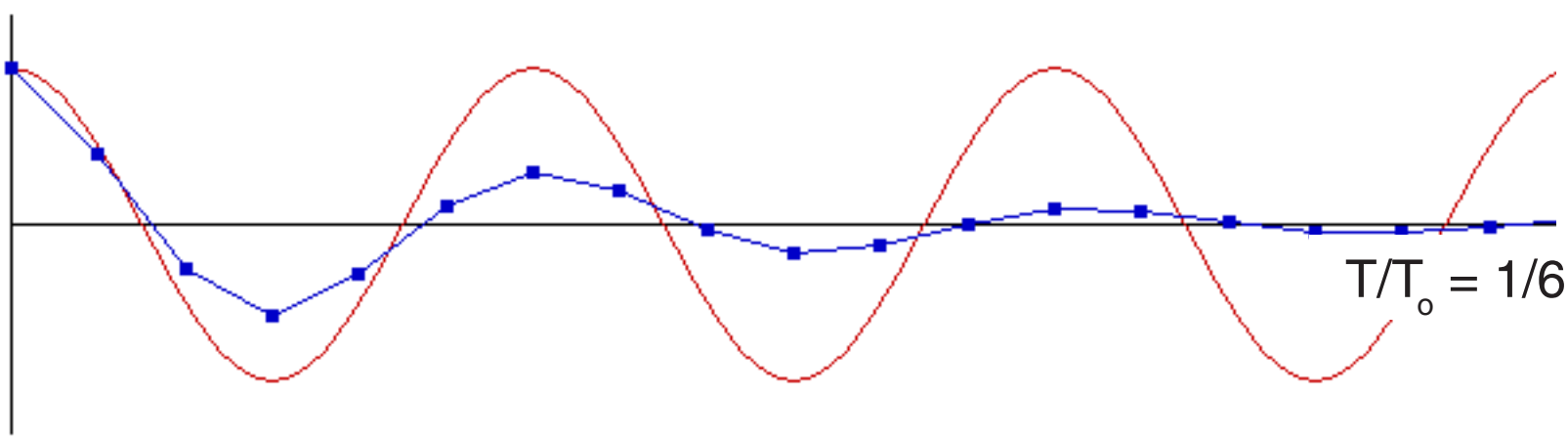
Because of the odd programming structure and the limited scope of applications, Semi-False Heun cannot be recommended. It is merely an academic example.

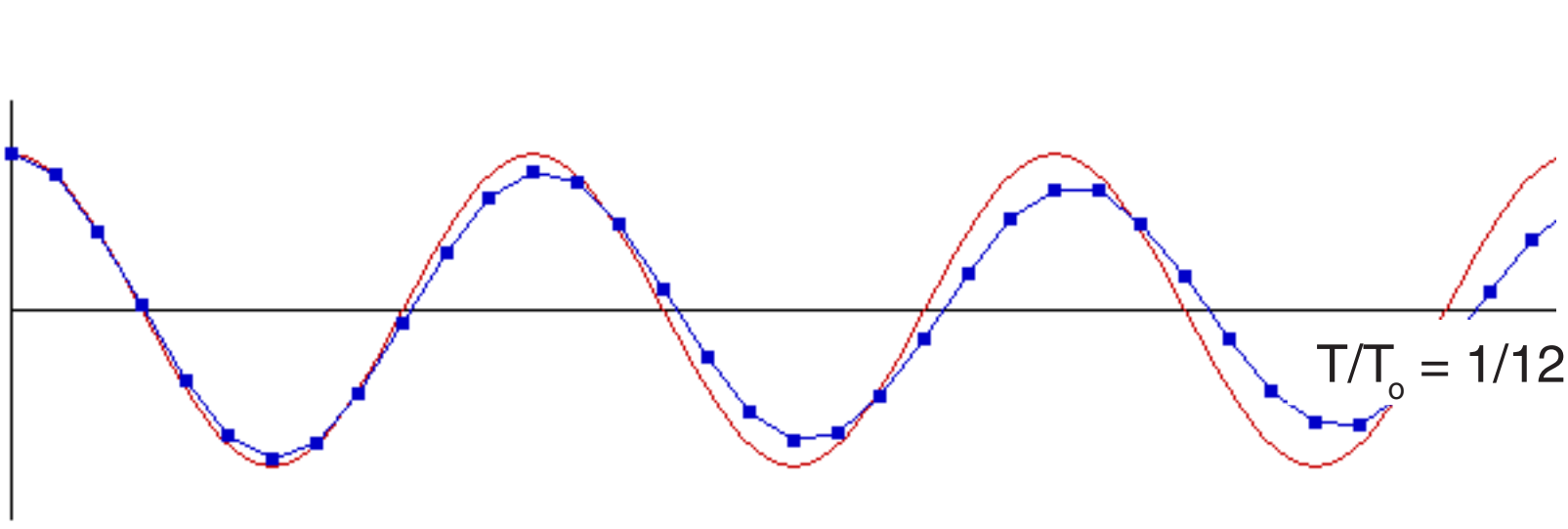
# Semi-False Heun y(t)

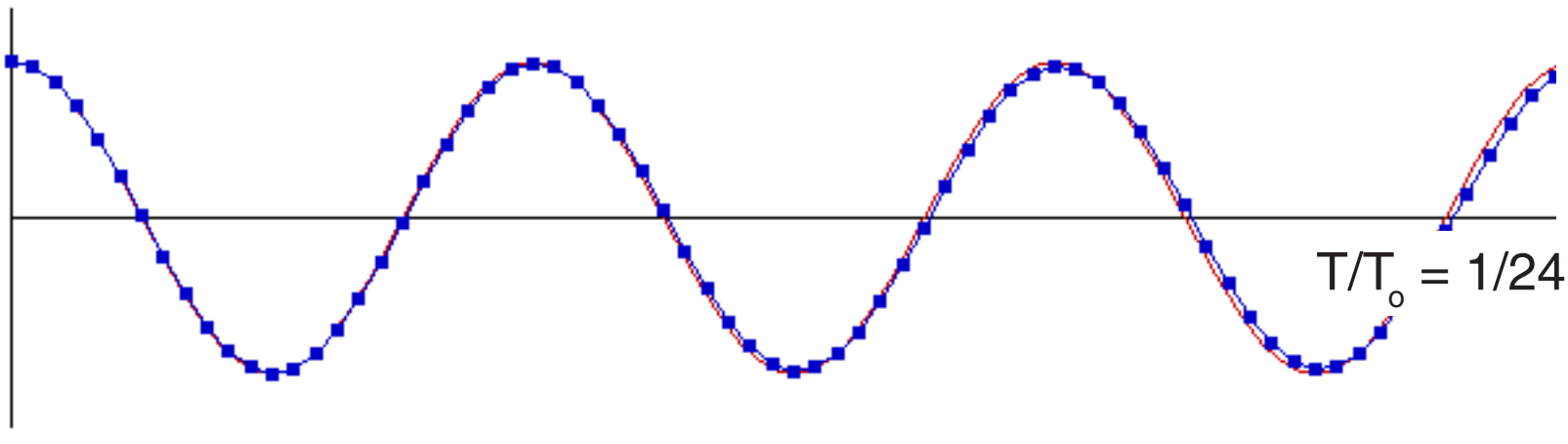
Initial conditions: $v_o=0$ , $y_o=1$



# 12. Conclusions

For computer games and non-professional flight or car simulators it boils down to this:

Use False Euler for fast calculations for the specified simple systems, like particle motion, mass and spring systems or celestial motions with low accuracy but guaranteed stability.

Use Standard Heun for general systems with medium accuracy and medium speed, if a suffiently small timestep can be guaranteed.

All other systems should be integrated by professional methods like higher order Runge-Kutta with stepsize control (RK-Merson, RK-Fehlberg).

Some applications require much more sophisticated strategies. For example the motion of a satellite (without drag) cannot be calculated by straight-forward high quality integration.The radius of the trajectory and the phase angle will drift away. In cases like this, additional constraints have to be used as control terms. Tests have shown, that motions under the influence of gravitational forces can be integrated by introducing the constant total energy as a control term [7] .
The author had simulated three-body systems with chaotic but bounded solutions, introducing the constant total energy, Appendix D.
The same strategy is necessary if e.g. the motion of a space pendulum is simulated in x,y,z coordinates instead of two generalized coordinates.
The length of the rod is a priori constant, and the numerical deviation is the control term [7].
Even the norm of a quaternion can be stabilized, if deviations are expected. Simply resetting the quaternion norm to one may practically work, but control is probably better.

More about programming highly accurate oscillators and digital filters can be found here [8], [9].

This report has been initiated by the presentation of the Verlet Integration in [6 ]. Now we see, that False Euler is a better formulation.

The mathematical kernels were evaluted on a hotel balcony in Rethymnon, Crete, in July 2002.

Thanks to Prof. Dr. Norbert Stuckenberg, Wolfenbüttel, for the discussion about the meaning of z-eigenvalues.

# 13. Appendix A

Approximations by Taylor series

Accurate solution for Damped Euler and False Euler:

$$\tan\Psi = \Phi(1 - D^2)^{0.5} / (1 - D\Phi) = \Phi(1 - \Phi^2/4)^{0.5} / (1 - \Phi^2/2)$$

Approximation by Taylor series:

$$\Psi + \Psi^3/3 \cong \Phi(1 - \Phi^2/8)\cdot(1 + \Phi^2/2) = +\Phi(1+ 3/8\Phi^2 + ...)$$

$$\Psi(1+ \Psi^2/3) \cong \Phi(1 + 3/8\Phi^2)$$

$$\Psi \cong \Phi(1 + 3/8\Phi^2) / (1+ \Psi^2/3) \cong \Phi(1 + 3/8\Phi^2)\cdot(1- \Psi^2/3)$$

Now we replace on the right side $\Psi^2/3$ by $\Phi^2/3$ :

Approximation:

$$\Psi \cong \Phi(1 + \Phi^2/24)$$

Accurate solution for Standard Heun:

$$\tan\Psi = \Phi / (1 - \Phi^2/2)$$

$$\Psi(1+ \Psi^2/3) \cong \Phi(1 + \Phi^2/2)$$

$$\Psi \cong \Phi(1 + \Phi^2/2)\cdot(1 - \Phi^2/3)$$

Approximation:

$$\Psi \cong \Phi(1+\Phi^2/6)$$

# 14. Appendix B

```
Procedure Func(sel,dir: Integer; wo,dT,Tmax: Double);
{ Numerical Integration; Direct or Transition Matrix }
Var t0,y0,v0,t1,y1,v1,y2           : Double;
    dv1,dy1,dv2,dy2,y11,v11,v12    : Double;
    woT,woT2,woT4                  : Double;
    i,k                            : Integer;
Begin
k:=Round(Tmax/0.025);
    { Cosine }
    t0:=0; y0:=1;
    MapP(t0,y0,p0,q0);
    For i:=1 to k Do
    Begin
     t1:=t0+0.025;
     y1:=coc(t1*wo);
     MapP(t1,y1,p1,q1);
     MakeSline(p0,q0,p1,q1,0,100);
     t0:=t1; y0:=y1; p0:=p1; q0:=q1;
    End;
woT :=wo*dT;
woT2:=Sqr(woT);
woT4:=Sqr(woT2);
k:=Round(Tmax/dT);
Case sel Of
```

```
1:  Begin
    WrTxtWXY(3,blac,1,1,'Standard Euler');
    t0:=0; v0:=0; y0:=1;
    MapP(t0,y0,p0,q0); Mark (p0,q0,120,100);
    For i:=1 to k Do
    Begin
     t1:= t0+dT;
    If dir=+1 Then
     Begin { Direct Integration }
       v1:=v0-woT*y0;
       y1:=y0+woT*v0;
     End Else
     Begin { Transition matrix }
       v1:= v0-woT*y0;
       y1:= woT*v0+y0;
     End;
    MapP(t1,y1,p1,q1);
    MakeSline(p0,q0,p1,q1,120,100); Mark (p1,q1,120,100);
    t0:=t1; v0:=v1; y0:=y1; p0:=p1; q0:=q1;
    End;
   End;
```

# Appendix B …

```
2:  Begin
    WrTxtWXY(3,blac,1,1,'Damped Euler');
    t0:=0; v0:=0; y0:=1;
    D:=0.5*woT;
    MapP(t0,y0,p0,q0);
    Mark(p0,q0,120,100);
    For i:=1 to k Do
    Begin
    t1:= t0+dT;
    If dir=+1 Then
    Begin { Direct Integration }
        v1:= v0-woT*(y0+2*D*v0);
        y1:= y0+woT*v0;
     End Else
    Begin { Transition matrix }
        v1:= (1-woT2)*v0-woT*y0;
        y1:= woT*v0+y0;
    End;
     MapP(t1,y1,p1,q1);
     MakeSline(p0,q0,p1,q1,120,100); Mark(p1,q1,120,100);
     t0:=t1; v0:=v1; y0:=y1; p0:=p1; q0:=q1;
    End;
    End;
```

```
3:  Begin
    WrTxtWXY(3,blac,1,1,'False Euler');
    t0:=0; v0:=0; y0:=1;
    MapP(t0,y0,p0,q0); Mark(p0,q0,120,100);
    For i:=1 to k Do
    Begin
    t1:= t0+dT;
    If dir=+1 Then
    Begin { Direct Integration }
        v1:= v0-woT*y0;
        y1:= y0+woT*v1;
    End Else
    Begin { Transition matrix }
        v1:= v0-woT*y0;
        y1:= woT*v0+(1-woT2)*y0;
     End;
    MapP(t1,y1,p1,q1);
     MakeSline(p0,q0,p1,q1,120,100); Mark(p1,q1,120,100);
     t0:=t1; v0:=v1; y0:=y1; p0:=p1; q0:=q1;
    End;
    End;
```

# Appendix B …

```
4:  Begin
    WrTxtWXY(3,blac,1,1,'Verlet');
    t0:=0; y0:=1; y2:=1;  { v0=0 }
    MapP(t0,y0,p0,q0); Mark (p0,q0,120,100);
    For i:=1 to k Do
    Begin
    t1:= t0+dT;
    y1:= 2*y0-y2+Sqr(dT)*(-wo*wo*y0);
    MapP(t1,y1,p1,q1);
    MakeSline(p0,q0,p1,q1,120,100); Mark(p1,q1,120,100);
    t0:=t1; y2:=y0; y0:=y1; p0:=p1; q0:=q1;
    End;
    End;
```

```
5:  Begin
    WrTxtWXY(3,blac,1,1,'Standard Heun');
    t0:=0; v0:=0; y0:=1;
    MapP(t0,y0,p0,q0); Mark(p0,q0,120,100);
    For i:=1 to k Do
    Begin
    t1 := t0+dT;
    If dir=+1 Then
    Begin { Direct Integration }
        dv1:=-woT*y0;
        v11:= v0+dv1;
        dy1:= woT*v0;
        y11:= y0+dy1;
        dv2:=-woT*y11;
        dy2:= woT*v11;
        v1 := v0+0.5*(dv1+dv2);
        y1 := y0+0.5*(dy1+dy2);
    End Else
    Begin { Transition matrix }
        v1 :=(1-0.5*woT2)*v0- woT*y0;
        y1 := woT*v0+(1-0.5*woT2)*y0;
    End;
    MapP(t1,y1,p1,q1);
    MakeSline(p0,q0,p1,q1,120,100); Mark(p1,q1,120,100);
        t0:=t1; v0:=v1; y0:=y1; p0:=p1; q0:=q1;
    End;
    End;
```

# Appendix B …

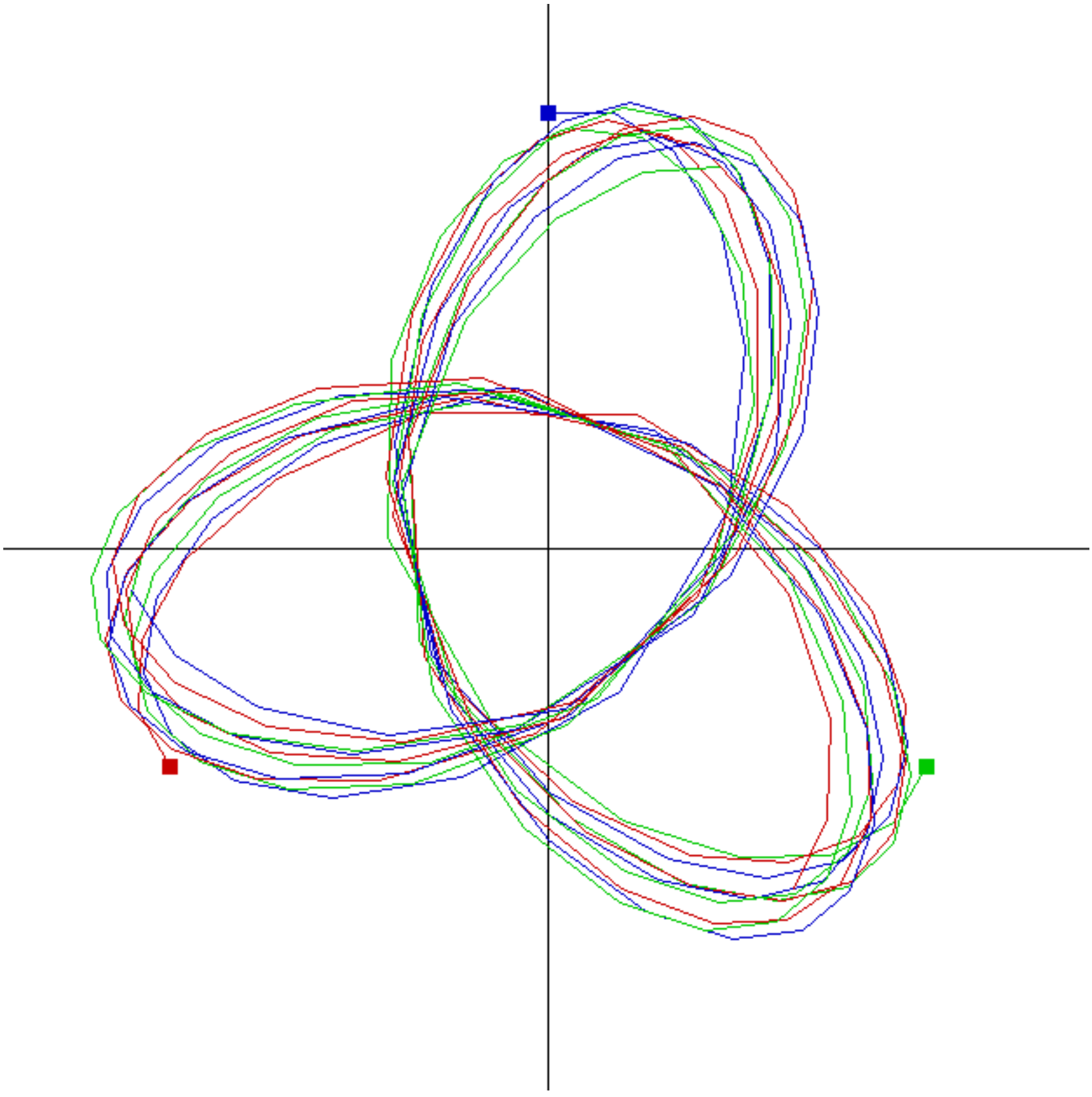
```
6:  Begin
    WrTxtWXY(3,blac,1,1,'False Heun');
    t0:=0; v0:=0; y0:=1;
    MapP(t0,y0,p0,q0); Mark(p0,q0,120,100);
    For i:=1 to k Do
    Begin
    t1 := t0+dT;
    If dir=+1 Then
    Begin { Direct Integration }
        dv1:=-woT*y0;
        v11:= v0+dv1;
        dy1:= woT*v11;
        y11:= y0+dy1;
        dv2:=-woT*y11;
        v12:= v0+dv2;
        dy2:= woT*v12;
        v1 := v0+0.5*(dv1+dv2);
        y1 := y0+0.5*(dy1+dy2);
    End Else
    Begin { Transition matrix }
        v1 :=(1-0.5*woT2)*v0-woT*(1-0.5*woT2)*y0;
        y1 := woT*(1-0.5*woT2)*v0+(1-woT2+0.5*woT4)*y0;
    End;
    MapP(t1,y1,p1,q1);
    MakeSline(p0,q0,p1,q1,120,100); Mark(p1,q1,120,100);
    t0:=t1; v0:=v1; y0:=y1; p0:=p1; q0:=q1;
    End;
    End;
```

# Appendix B …

```
7:  Begin
    WrTxtWXY(3,blac,1,1,'Semi-False Heun');
    x0:=0; v0:=0; y0:=1;
    MapP(x0,y0,p0,q0);
    Mark(p0,q0,120,100);
    For i:=1 to k Do
   Begin
    x1 := x0+dT;
    If dir=+1 Then
    Begin { Direct Integration }
        dv1:=-woT*y0;
        v11:= v0+dv1;
        dy1:= woT*v0;
        y11:= y0+dy1;
        dv2:=-woT*y11;
        v12:= v0+dv2;
        dy2:= woT*v12;
        v1 := v0+0.5*(dv1+dv2);
        y1 := y0+0.5*(dy1+dy2);
    End Else
    Begin { Transition matrix }
        v1:=(1-0.5*woT2)*v0-woT*y0;
        y1:= woT*(1-0.5*woT2)*v0+(1-0.5*woT2)*y0;
    End;
        MapP(x1,y1,p1,q1);
    MakeSline(p0,q0,p1,q1,120,100); Mark(p1,q1,120,100);
    x0:=x1; v0:=v1; y0:=y1; p0:=p1; q0:=q1;
    End;
    End;
End; { case}
End;
```

# 15. Appendix C

Three ’celestial’ bodies under the influence of gravitational forces in the xy plane. The initial conditions are symmetric. The true trajectory is the averaged clover leaf. Solution by False Euler.
The image shows the first 100 steps. The solution is not accurate but stable ad infinitum.



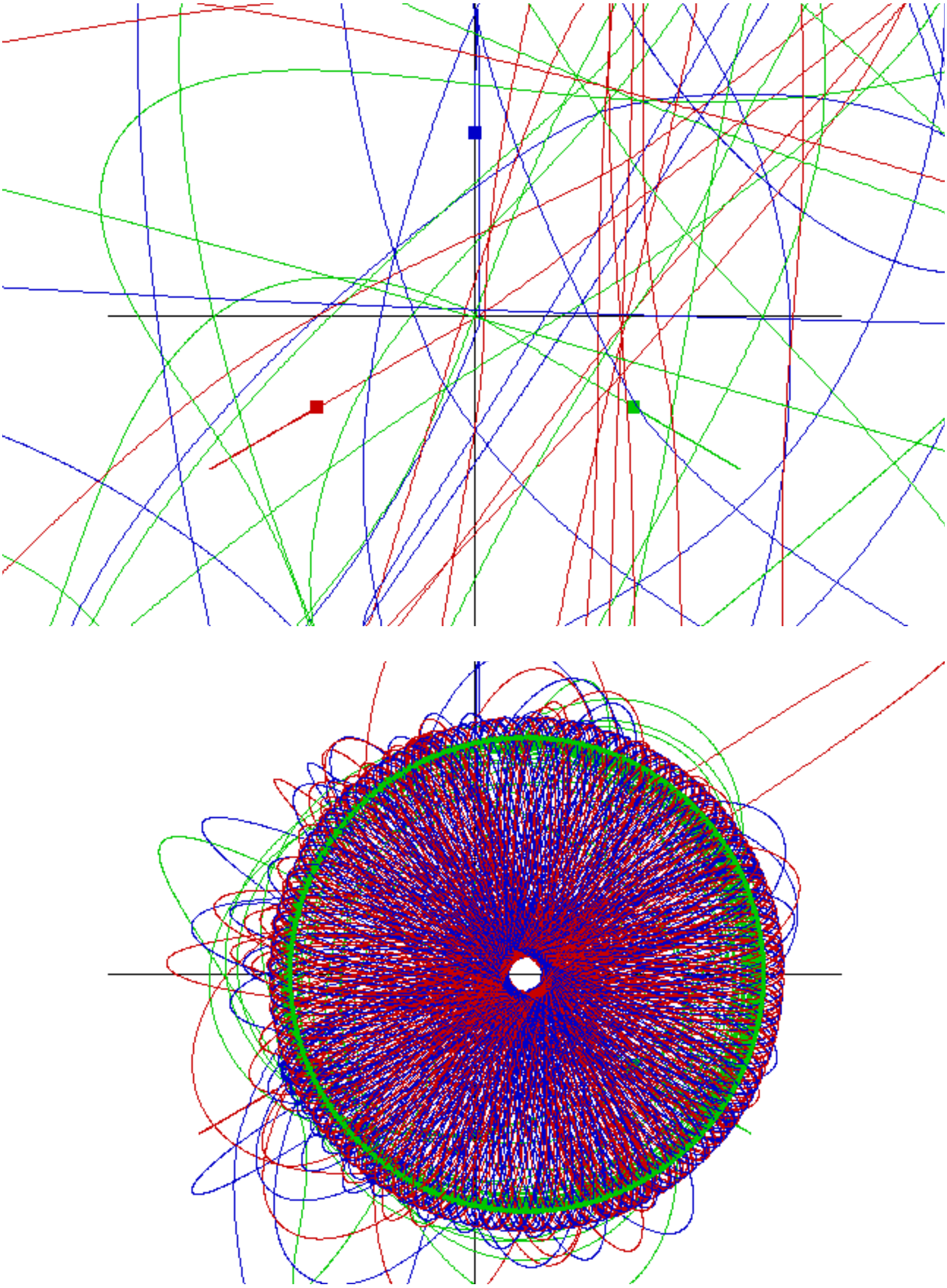
```
Procedure Func;
{ Three Bodies }
Var x1o,y1o,x2o,y2o,x3o,y3o : Double;
    v1o,w1o,v2o,w2o,v3o,w3o : Double;
    x1 ,y1 ,x2 ,y2 ,x3 ,y3  : Double;
    v1 ,w1 ,v2 ,w2 ,v3, w3  : Double;
    p1o,q1o,p2o,q2o,p3o,q3o : Integer;
    p1 ,q1 ,p2 ,q2 ,p3 ,q3  : Integer;
    r12,r13,r23,c3,dT       : Double;
    si,co                   : Single;
    k                       : LongInt;
```

# Appendix C …

```
Begin
SicCoc(30*wrad,si,co);
dT:=0.3;
{ False Euler }
    x1o:=-co; y1o:=-si; v1o:=-si; w1o:= co;
    x2o:= co; y2o:=-si; v2o:=-si; w2o:=-co;
    x3o:= 0;  y3o:= 1;  v3o:=1; w3o:=0;
    MapP(x1o,y1o,p1o,q1o); Mark(p1o,q1o,  0,100);
    MapP(x2o,y2o,p2o,q2o); Mark(p2o,q2o, 60,100);
    MapP(x3o,y3o,p3o,q3o); Mark(p3o,q3o,120,100);
    For k:=1 to 100 Do
    Begin
        r12:=Sqr(x2o-x1o)+Sqr(y2o-y1o);
        r13:=Sqr(x3o-x1o)+Sqr(y3o-y1o);
        r23:=Sqr(x3o-x2o)+Sqr(y3o-y2o);
        v1:=v1o+dT*((x2-x1)/r12+(x3-x1)/r13);
        v2:=v2o+dT*((x3-x2)/r23+(x1-x2)/r12);
        v3:=v3o+dT*((x1-x3)/r13+(x2-x3)/r23);
        w1:=w1o+dT*((y2-y1)/r12+(y3-y1)/r13);
        w2:=w2o+dT*((y1-y2)/r12+(y3-y2)/r23);
        w3:=w3o+dT*((y1-y3)/r13+(y2-y3)/r23);
        x1:=x1o+dT*v1;
        y1:=y1o+dT*w1;
        x2:=x2o+dT*v2;
        y2:=y2o+dT*w2;
        x3:=x3o+dT*v3;
        y3:=y3o+dT*w3;
        MapP(x1,y1,p1,q1);
        MapP(x2,y2,p2,q2);
        MapP(x3,y3,p3,q3);
        MakeSline(p1o,q1o,p1,q1,  0,100);
        MakeSline(p2o,q2o,p2,q2, 60,100);
        MakeSline(p3o,q3o,p3,q3,120,100);
        x1o:=x1; y1o:=y1;
        x2o:=x2; y2o:=y2;
        x3o:=x3; y3o:=y3;
        v1o:=v1; w1o:=w1;
        v2o:=v2; w2o:=w2;
        v3o:=v3; w3o:=w3;
        p1o:=p1; q1o:=q1;
        p2o:=p2; q2o:=q2;
        p3o:=p3; q3o:=q3;
    End;
End;
```

# 16. Appendix D

Three ’celestial’ bodies under the influence of gravitational forces in the xy plane. The initial conditions are slightly non symmetric. 100.000 steps.
Upper image: Standard Euler, unstable.
Lower image: Standard Euler with energy control, stable.
For symmetric initial conditions we get a circle.



# Appendix D ...

```
Procedure Func;
{ Three Bodies }
Var x1o,y1o,x2o,y2o,x3o,y3o : Double;
    v1o,w1o,v2o,w2o,v3o,w3o : Double;
    x1 ,y1 ,x2 ,y2 ,x3 ,y3  : Double;
    v1 ,w1 ,v2 ,w2 ,v3, w3  : Double;
    p1o,q1o,p2o,q2o,p3o,q3o : Integer;
    p1 ,q1 ,p2 ,q2 ,p3 ,q3  : Integer;
    r12,r13,r23,c3,dT,Cf    : Double;
    dEn,Eno,eps             : Double;
    si,co                   : Single;
    k                       : LongInt;
Begin
SicCoc(30*wrad,si,co);
dT:=0.01;
Cf:=0.3; { experimental control factor  }
{ Standard Euler with Energy control }
    eps:=0.01;
    x1o:=-co; y1o:=-si; v1o:=-co; w1o:=-si;
    x2o:= co; y2o:=-si; v2o:= co; w2o:=-si;
    x3o:=  0; y3o:=  1; v3o:= eps;w3o:=  1;
    MapP(x1o,y1o,p1o,q1o); Mark(p1o,q1o,  0,100);
    MapP(x2o,y2o,p2o,q2o); Mark(p2o,q2o, 60,100);
    MapP(x3o,y3o,p3o,q3o); Mark(p3o,q3o,120,100);
    { Initial total energy }
    r12:=Sqr(x2o-x1o)+Sqr(y2o-y1o);
    r13:=Sqr(x3o-x1o)+Sqr(y3o-y1o);
    r23:=Sqr(x3o-x2o)+Sqr(y3o-y2o);
    Eno:=(sqr(v1o)+sqr(w1o)+sqr(v2o)+sqr(w2o)+sqr(v3o)+sqr(w3o))/2
         -Sqrt(1/r12)-Sqrt(1/r23)-Sqrt(1/r13);
    ASM FNINIT END;
    For k:=1 to 100000 Do
    Begin
       r12:=Sqr(x2o-x1o)+Sqr(y2o-y1o);
       r13:=Sqr(x3o-x1o)+Sqr(y3o-y1o);
       r23:=Sqr(x3o-x2o)+Sqr(y3o-y2o);
{ Actual energy difference }
dEn:=(sqr(v1o)+sqr(w1o)+sqr(v2o)+sqr(w2o)+sqr(v3o)+sqr(w3o))/2
     -Sqrt(1/r12)-Sqrt(1/r23)-Sqrt(1/r13)-Eno;
ASM FNINIT END;
{ Control by partial derivatives dE/dv1 etc. }
    v1:=v1o+dT*( (x2-x1)/r12+(x3-x1)/r13 - Cf*dEn*v1o );
    v2:=v2o+dT*( (x3-x2)/r23+(x1-x2)/r12 - Cf*dEn*v2o );
    v3:=v3o+dT*( (x1-x3)/r13+(x2-x3)/r23 - Cf*dEn*v3o );
    w1:=w1o+dT*( (y2-y1)/r12+(y3-y1)/r13 - Cf*dEn*w1o );
    w2:=w2o+dT*( (y3-y2)/r23+(y1-y2)/r12 - Cf*dEn*w2o );
    w3:=w3o+dT*( (y1-y3)/r13+(y2-y3)/r23 - Cf*dEn*w3o );
x1:=x1o+dT*v1o;
y1:=y1o+dT*w1o;
```

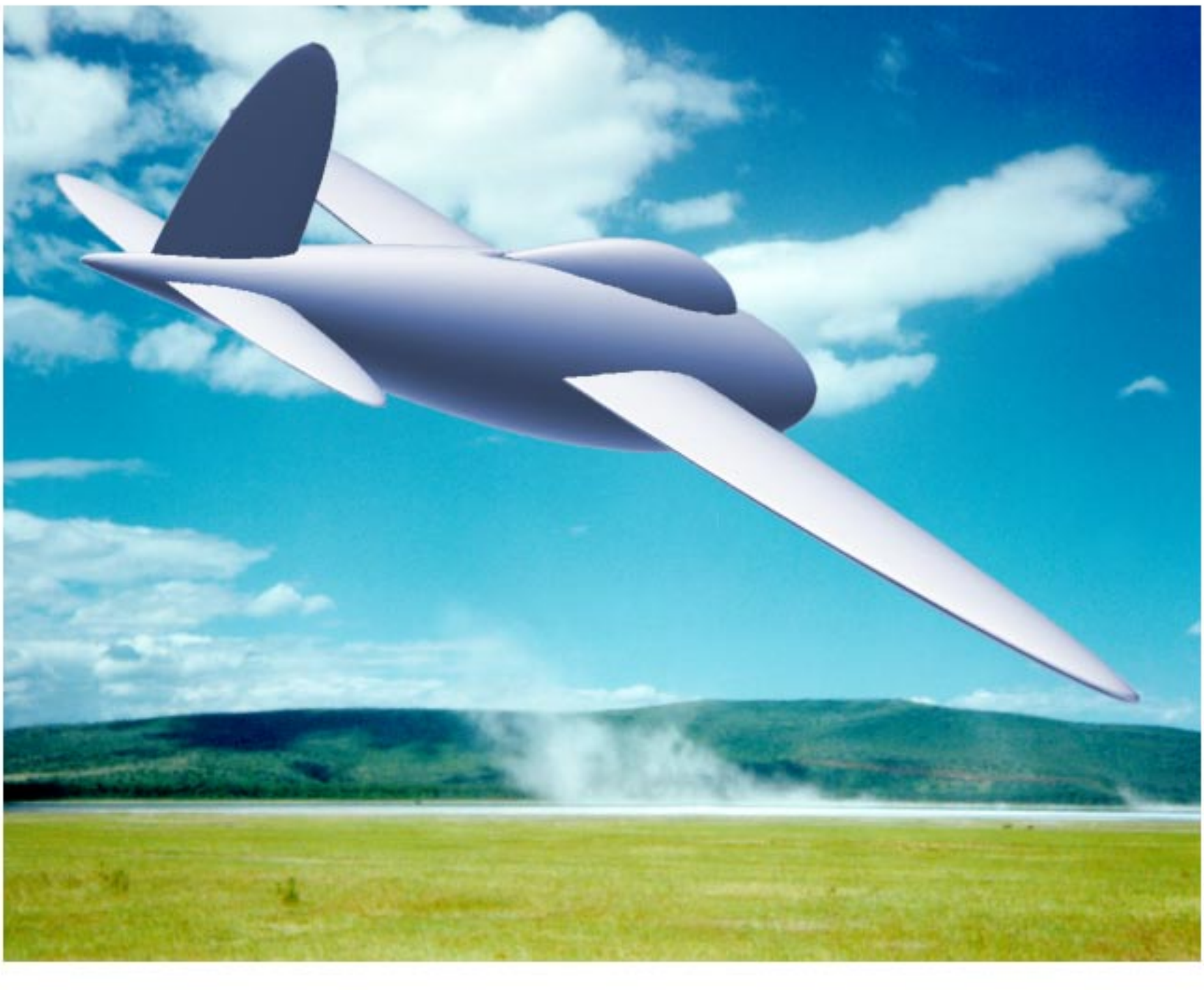
# Appendix D …

```
    x2:=x2o+dT*v2o;
    y2:=y2o+dT*w2o;
    x3:=x3o+dT*v3o;
    y3:=y3o+dT*w3o;
    MapP(x1,y1,p1,q1);
    MapP(x2,y2,p2,q2);
    MapP(x3,y3,p3,q3);
    MakeSline(p1o,q1o,p1,q1,  0,100);
    MakeSline(p2o,q2o,p2,q2, 60,100);
    MakeSline(p3o,q3o,p3,q3,120,100);
    x1o:=x1; y1o:=y1;
    x2o:=x2; y2o:=y2;
    x3o:=x3; y3o:=y3;
    v1o:=v1; w1o:=w1;
    v2o:=v2; w2o:=w2;
    v3o:=v3; w3o:=w3;
    p1o:=p1; q1o:=q1;
    p2o:=p2; q2o:=q2;
    p3o:=p3; q3o:=q3;
    End;
End;
```

# 17. Appendix E

Simulation of the rotation of an aircraft. The example shows the simple application of False Euler for a rigid body rotation.

Coordinate system in principal axes directions.
Simplified roll angle and pitch angle controller.
Yaw rate by sine of roll angle.



# Appendix E …

```
Procedure Integ;
{   Rotational differential equations for an aircraft
    Integration by False Euler
    Simplified attitude controller
    Command input in degrees for rollangle and pitchangle }
Var    sPhi,cPhi,sThe,cThe,sPsi,cPsi,icThe: Double;
Const  Jx=0.6; Jy=1; Jz=1.5;
           dampP=2;   contPhi=0.06;
           dampQ=3;   contThe=0.10;
           dampR=1;   contPsi=1;
           eps =1E-5;
           wrad=pi/180;
           dT  =0.1;
           dT1 =dT/wrad;
Begin
{ Fast Sines and Cosines }
siccoc(wrad*Phi,sPhi,cPhi);
siccoc(wrad*The,sThe,cThe);
siccoc(wrad*Psi,sPsi,cPsi);
{   Rolldamper, Pitchdamper, Yawdamper
    Rollcontroller, Pitchcontroller, Yawrate by Rollangle
    Phic = Command Rollangle
    Thec = Command Pitchangle
    Txc,Tyc,Tzz = Command Torques }
Tx:=-dampP*wx+Txc;
Ty:=-dampQ*wy+Tyc;
Tz:=-dampR*wz-contPsi*sPhi*cThe +Tzc;
If COn=1 Then
Begin
While Phi> 180 Do Phi:=Phi-360;
While Phi<-180 Do Phi:=Phi+360;
While The> 180 Do The:=The-360;
While The<-180 Do The:=The+360;
Tx:=Tx-contPhi*(Phi-Phic);
Ty:=Ty-contThe*(The-Thec);
End;
{ Body fixed coordinate system in principal axes direction
{ Integration of angular velocities }
wx:=wxo + dT*(wyo*wzo*(Jy-Jz)+Tx)/Jx;
wy:=wyo + dT*(wzo*wxo*(Jz-Jx)+Ty)/Jy;
wz:=wzo + dT*(wxo*wyo*(Jx-Jy)+Tz)/Jz;
wxo:=wx; wyo:=wy; wzo:=wz;
{ Trap gimbal lock singularity }
If Abs(cThe)<eps Then
    Begin
    If cThe>0 Then cThe:=eps Else cThe:=-eps;
    End;
icThe:=1/cThe;
```

# Appendix E …

```
{ Integration of Euler angle derivatives }
Phi:=Phi    +   dT1*(wx+wy*sPhi*sThe*icThe+wz*cPhi*sThe*icThe);
The:=The    +   dT1*(   wy*cPhi           -wz*sPhi           );
Psi:=Psi    +   dT1*(   wy*sPhi*icThe     +wz*cPhi*icThe     );
WrNumWin(1,grel,1,'Phi', Phi);
WrNumWin(1,grel,2,'The',-The);
WrNumWin(1,grel,3,'Psi',-Psi);
{ Rotation matrix }
    o11:=+cpsi*cthe;
    o12:=-spsi*cphi+cpsi*sthe*sphi;
    o13:=+spsi*sphi+cpsi*sthe*cphi;
    o21:=+spsi*cthe;
    o22:=+cpsi*cphi+spsi*sthe*sphi;
    o23:=-cpsi*sphi+spsi*sthe*cphi;
    o31:=-sthe;
    o32:=+cthe*sphi;
    o33:=+cthe*cphi;
End;
```

# 18. References


[ 1 ] Schrüfer, E.
Signalverarbeitung
Carl Hanser Verlag, München, Wien, 1990

[ 2 ] Schwarz, H.R.
Numerische Mathematik
B.G.Teubner, Stuttgart, 1993

[ 3 ] Mathews, J.H. and Fink, K.D.
Numerical Methods
Prentice-Hall, Upper Saddle River, NJ07458, 1999

[ 4 ] Isermann, R.
Digitale Regelungssysteme, Band 1
Springer Verlag, Berlin, ..., Tokyo, 1987

[ 5 ] Stearns, S.
Digitale Verarbeitung analoger Signale
R.Oldenbourg Verlag, München, Wien, 1988

[ 6 ] http://ns.ioi.dk/homepages/tj/publications/gdc2001.htm

[ 7 ] Baumgarte, J.
Stabilization of Constraints and Integrals of Motion in Dynamical Systems
Computer Meth. in Appl. Mechanics and Engineering 1(1972)
16 pages

[ 8 ] Hoffmann, G.
Zeitsynchrone sprungfreie Bahngeneratoren
ZAMM, Z. angew. Math. Mech. 74 (1994) S.69-71
http://docs-hoffmann.de/oscil09082002.pdf

[ 9 ] Hoffmann, G.
Musikgesteuerte Marionette
Automatisierungstechnik 42 (8,1994) S.356-365
http://docs-hoffmann.de/mario01.pdf