Predicting Classical Motion Directly from the Action Principle III

Ulrich Mutze
Heidelberg Digital Finishing GmbH
73347 Mühlhausen, Germany
ulrich.mutze@de.heidelberg.com

Abstract

For an arbitrary holonomic mechanical system a method for constructing time-discrete trajectories is derived by applying a generalized principle of stationary action to the manifold of those system paths which are parabolic with respect to system of generalized coordinates. The method is applied to the anti-damped harmonic oscillator, and data are reported that suggest that the method accurately represents the growth of the amplitude till numerical overflow. A modified variational derivative of the generalized action integral is shown to agree with the force that the environment of the system exerts to it. This generalizes the characterization of free motion in terms of vanishing variational derivative.

1 Introducing the Motive Force Function

Let us consider a holonomic mechanical system and restrict our interest on a part Ω of the configuration space on which a single system of n generalized coordinates can be introduced. The corresponding coordinate n-tupels form an open, connected subset X_n of $\mathbb{R}^n$ and time is restricted to an open interval I. Then a trajectory of the system is represented as a curve

$$I \longrightarrow X_n\,, \quad t \longmapsto x(t)\,, \tag{1}$$

and the time derivatives of the coordinate n-tupels, the (generalized) velocities, are denoted by v and the second time derivatives by a. The dynamical behavior of the system is characterized by a Lagrangian

$$L : I \times X_n \times \mathbb{R}^n \longrightarrow \mathbb{R}\ , \quad (t,x,v) \longmapsto L(t,x,v)\ . \tag{2}$$

An important potential of generalized coordinates (and the circumstance that the Lagrangian depends only on them and their time derivatives) is to cope with technical systems. It allows to model in an accurate and economical manner constraint motion like that of a pendulum or of a roller coaster cabin. It does not allow, however, to model friction and other forces that are accompanied with power being extracted from or injected into the system. Today, many mechanisms contain motors or other actuators which transmit power under the control of sensor signals and the internal state of micro electronic devices. So it is normal that an internal mass of the system feels forces that are given by a rather arbitrary function of time, position and velocity. [1]

It is the attitude of this article to consider such forces as important as the ones already present in the Lagrangian and to accept no constructs excluding them. Not restricting the nature of the forces also helps to approximate arbitrary non-holonomic constraints by introducing forces that dynamically enforce obedience to these constraints with sufficient precision. Since non-holonomic constraints in technical systems are normally less stiff than holonomic ones this strategy is here much more adequate than for holonomic constraints.

Forces, that are not included in the definition of the Lagrangian are given by their components with respect to the generalized coordinates under consideration (see [1] §33, (7)):

$$F : I \times X_n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n\ , \quad (t,x,v) \longmapsto F(t,x,v)\ . \tag{3}$$

Their operational definition is by a thought experiment: switching F off could be compensated by an external mechanism which delivers the power $-Fv = -F_i v_i$ (sum convention) to the system.

The processes or conditions which exert force F are considered part of the system under discussion. The general Lagrange equations of mechanics (see [1] §34, $(8a)$) determine the trajectories of the system for given initial conditions, i.e. for given values of (t,x,v). Physicists became used to consider this the answer to the basic physical question to be asked in this framework. Here, we consider a slightly more general question which is basic to many technical applications and to our sensorial experience of mechanics: Let us consider a situation in which the

---

[1] and acceleration, ..., a generalization which we shall not take into account.

system is forced by external means (e.g. by a mechanism or by our own hands) to follow a pre-determined trajectory $t \mapsto x(t)$. At each point in time one has to apply a *motive force* M to keep the system 'on track'. In usual physical notation this force is given by

$$M = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} - F = -\frac{\delta L}{\delta x} - F \, , \tag{4}$$

which is a by-product of the reasoning that leads to writing the equation of motion in the form $M = 0$ (see [1] §34). This equation for the motive force implies that for any instant t the quantity M depends only on $x(t), \dot{x}(t), \ddot{x}(t)$ thus giving rise to a function

$$M : I \times X_n \times \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n \, , \quad (t,x,v,a) \longmapsto M(t,x,v,a) \tag{5}$$

which according to (4) has the representation

$$\begin{aligned} M_i(t,x,v,a) = {} & L_{v_i v_k}(t,x,v) a_k + L_{v_i x_k}(t,x,v) v_k \\ & + L_{v_i t}(t,x,v) - L_{x_i}(t,x,v) - F_i(t,x,v) \, , \end{aligned} \tag{6}$$

where the indexes to L denote partial derivatives with respect to the argument positions that normally are associated with variable names t, x, v. For later convenience we introduce a notation for the expression arising from solving the equation $M = 0$ for a:

$$\begin{aligned} a &= (L_{vv}(t,x,v))^{-1}(F(t,x,v) + L_x(t,x,v) - L_{vt}(t,x,v) - L_{vx}(t,x,v)v) \\ &=: A(t,x,v) \, . \end{aligned} \tag{7}$$

Here, the inverse power denotes for $n > 1$ the inversion of a matrix which very conveniently can be defined to mean the always existing pseudo-inverse. For systems of point particles in Cartesian coordinates the matrix L_{vv} is diagonal so that matrix inversion becomes inversion of numbers. If F, L_x, L_{vt} depend on v trivially, and L_{vx} vanishes, A depends on v trivially. This then gives the time stepping algorithm in section 3 a particularly simple form.

Notice that function M can be studied experimentally also for arguments that never would arise from a free trajectory starting from any initial condition that can be realized experimentally. Further it should be noted that knowing this expression is more than knowing the equation of motion: the latter can be multiplied by any non-zero factor and will remain a equation of motion, but it will loose the capability to predict reaction forces of the system to the out-side world.

2 Motive Force and the Action along Parabolic Paths

Let us now investigate how the motive force function M is related to the action integral. Since we include forces which do not result from a Lagrangian the action integral has to be extended. This extension is suggested by known formulations of the Hamilton's principle for non-conservative forces([2] p. 42-44) and was proposed in a restricted context in [5]. It deals with paths that are short enough to justify a representation in terms of quadratic polynomials and it was motivated by the desire to improve the Euler rule for a finite time step trajectory both in accuracy and stability while maintaining the simple physics-based logic of this rule.

With a parabolic path segment of time extension 2τ

$$[t, t+2\tau] \longrightarrow X_n\,, \quad t+h \longmapsto x+vh+a\frac{h^2}{2}\,, \tag{8}$$

we associate the *generalized action integral*

$$\begin{aligned} S(t,x,v,a;\tau) :=& \int_0^{2\tau} L(t+h, x+vh+a\frac{h^2}{2}, v+ah)\, dh \\ &+ \int_0^{2\tau} F(t+h, x+vh+a\frac{h^2}{2}, v+ah)\, a\frac{h}{2}(h-2\tau)\, dh \end{aligned} \tag{9}$$

on the natural domain of this expression:

$$\begin{aligned} Y := \{&(t,x,v,a,\tau) \in I \times X_n \times \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \mid \\ &t+2\tau \in I,\ \bigwedge\nolimits_{h\in[0,2\tau[} x+vh+a\frac{h^2}{2} \in X_n\}\,. \end{aligned} \tag{10}$$

In addition to the parametrization of a path segment in terms of (x,v,a), we introduce one in terms of three configurations (x_1, x_2, x_3) for three equidistant points in time:

$$\begin{aligned} x_1 &:= x \\ x_2 &:= x+v\tau+a\frac{\tau^2}{2} \\ x_3 &:= x+v(2\tau)+a\frac{(2\tau)^2}{2}\,. \end{aligned} \tag{11}$$

In terms of these quantities we have

$$\begin{aligned} x &= x_1 \\ v &= \frac{-3x_1+4x_2-x_3}{2\tau} \\ a &= \frac{x_1-2x_2+x_3}{\tau^2} \,. \end{aligned} \tag{12}$$

Then the generalized action integral gives rise to a 'three-point function':

$$\tilde{S}(t,x_1,x_2,x_3;\tau) := S(t,x,v,a;\tau) \,, \tag{13}$$

where x,v, and a are given by (12). Hamilton's principle is concerned with variations of the path for which the endpoints are held fixed. In the present context such a variation corresponds simply to a change of the variable x_2 and the variational derivative of the action integral becomes proportional to the normal derivative with respect to this variable. A factor of proportionality is built into the following definition in order to absorb a factor which otherwise would appear in the following theorem. Here, we use the 3-point parameterization only as a heuristic interplay and return to the original parameters:

$$\begin{aligned} \frac{3}{4\tau}\frac{\partial\tilde{S}}{\partial x_2}(t,x_1,x_2,x_3;\tau) &= \frac{3}{2\tau^2}\left(S_v(t,x,v,a;\tau)-\frac{1}{\tau}S_a(t,x,v,a;\tau)\right) \\ &=: \hat{S}(t,x,v,a;\tau) \,. \end{aligned} \tag{14}$$

The expression $\hat{S}$ shall be called the *midpoint derivative* of S. The symbol ˆ tries to suggest a shift of the central one of three points, and its position indicates a derivative in analogy to Newton's dot notation. This position does not interfere with the coordinate index, which the symbol carries in a more verbose notation. Now all expressions are in place to formulate the main result:

Theorem 1 *Let L and F be regular enough that M as defined in (5), (6) is C^2, and let function $\hat{S}$ be defined on Y of (10) by equations (9) and (14). Then*

$$\hat{S}(t,x,v,a;\tau) = -M(t+\tau,x+v\tau,v+a\tau,a)+O(\tau^2) \,. \tag{15}$$

Verification of this statement is straigtforward in principle since both sides can be expanded in powers of τ by Taylor's formula. For the left-hand side it is convenient to perform the midpoint derivative under the integral first:

$$\frac{4\tau}{3}\hat{S}_i(t,x,v,a;\tau) =$$
$$\int_0^{2\tau} dh\frac{h}{2\tau}(2-\frac{h}{\tau})(2F_i + a_k\frac{\partial F_k}{\partial x_i} - 2\frac{\partial L}{\partial x_i})(t+h, x+vh+a\frac{h^2}{2}, v+ah) +$$
$$\int_0^{2\tau} \frac{dh}{\tau}(1-\frac{h}{\tau})(a_k\frac{\partial F_k}{\partial v_i} - 2\frac{\partial L}{\partial v_i})(t+h, x+vh+a\frac{h^2}{2}, v+ah) \tag{16}$$

and then expand the integrand in powers of h and finally get a series in τ by integration. The number of terms created during this expansion is so large that I switched to using computer programs for symbolic computations. I built the expressions for the one-dimensional case by *Mathematica* which is perfectly straightforward since all derivatives, integrals, and power expansions can be formulated in terms of built-in functions. It is rather obvious how the terms resulting from the 1-dimensional expansion may be interpreted as a shorthand-notation for the terms resulting from the expansion in any dimension. To bring such an argument in a rigorous form, one had to work through some technicalities. In order to avoid this I implemented the rules that create the expansion coefficients in the language of the powerful program *FORM* by J.Vermaseren and got a result that agreed with the *Mathematica* result when applying the shorthand interpretation mentioned above. The *FORM* program and result listing is given here as an Appendix. This is not only the most efficient way to convince oneself of the correctness of the theorem but also is a convenient tool for testing variations of it. So, one easily finds that replacing $M(t+\tau, x+v\tau, v+a\tau, a)$ by $M(t+\tau, x+v\tau+a\frac{\tau^2}{2}, v+a\tau, a)$ does *not* increase the order of the approximation, [2] and that a replacement by $M(t+\tau, x+v\tau, v, a)$ lowers it.

The precise structure of the expressions in the theorem was found by trial and error from special results that were obtained for Lagrangians of special type, some also by using broken linear paths instead of parabolic ones.

A simple consequence of (15) is

$$\hat{S}(t,x,v,a;\tau) = -M(t,x,v,a) + O(\tau)\,, \tag{17}$$

---

[2] this would give rise to the implicit midpoint rule instead of the explicit one in the course of the following development

This shows that the the general Lagrange equations of mechanics, given by $M = 0$ (see (4)), follow from a *local action principle* saying that the midpoint derivative of the generalized action integral vanishes on a trajectory. This suggests to take

$$\hat{S}(t,x,v,a;\tau) = 0 \tag{18}$$

as a rule for defining a in (8) to approximate the actual system trajectory with initial conditions (t,x,v) over a span 2τ. Since in most cases of practical interest, the function $\hat{S}$ can't be calculated exactly, it is natural to use the approximation provided by the theorem and to determine a from the equation

$$M(t+\tau, x+v\tau, v+a\tau, a) = 0\,. \tag{19}$$

Notice that Euler's rule just uses equation $M(t,x,v,a) = 0$ instead. This is the only rational rule in a situation where no information is available on the time span over which a will be used to predict the trajectory. If this span 2τ is known, a much better rule can be given in the form of (19). The next section discusses the computational solution of the initial value problem in terms of 2τ time steps in more detail. Section 4 will give an example where (19) works better than (18).

3 Time stepping algorithm

Given (t,x,v,τ) we determine the acceleration a from (19) and put for the next state

$$\begin{aligned} \underline{t} &:= t + 2\tau \\ \underline{v} &:= v + a(2\tau) \\ \underline{x} &:= x + v(2\tau) + a\frac{(2\tau)^2}{2} \end{aligned} \tag{20}$$

or in a more economical form

$$\begin{aligned} \underline{t} &:= t + 2\tau \\ \hat{v} &:= v + a(2\tau) \\ \underline{x} &:= x + (v + \hat{v})\tau \\ \underline{v} &:= \hat{v} \end{aligned} \tag{21}$$

The main task of the time stepping algorithm thus is to solve (19) for a. Using the function A from (7), this can be written as

$$a = A(t+\tau, x+v\tau, v+a\tau)\ . \tag{22}$$

If A depends on v trivially, we get a simply as

$$a = A(t+\tau, x+v\tau, v)\ , \tag{23}$$

and if it depends linearly : $A(t,x,v) = A_0(t,x) + A_1(t,x)v$, we get

$$a = (1 - \tau A_1(t+\tau, x+v\tau))^{-1}(A_0(t+\tau, x+v\tau) + A_1(t+\tau, x+v\tau)v)\ . \tag{24}$$

Else, the equation is implicit in a harmless manner and can be solved by a few iterations of

$$\begin{aligned} a_0 &:= 0 \\ a_{n+1} &:= A(t+\tau, x+v\tau, v+a_n\tau)\ . \end{aligned} \tag{25}$$

This algorithm is a modification of what is known as the *explicit midpoint rule*, see [3], [4]. It is interesting to observe that (23) can easily be applied to quantum dynamics and to hyperbolic partial differential equations. Let us spell out here the quantum version for a time-dependent Hamiltonian H_t:

$$\begin{aligned} a &:= -\frac{1}{\hbar^2} H^2_{t+\tau}(\psi_t + \tau\dot{\psi}_t)\ , \\ \psi_{t+2\tau} &:= \psi_t + 2\tau\dot{\psi}_t + 2\tau^2 a\ , \\ \dot{\psi}_{t+2\tau} &:= \dot{\psi}_t + 2\tau a\ , \end{aligned} \tag{26}$$

where the initialization of the velocities is done by $\dot{\psi}_0 := \frac{1}{i\hbar} H_0 \psi_0$.

3.1 Informal aside

With (26) it is a matter of half an hour to set up a one-dimensional computer model of a single quantum particle under the influence of arbitrary potentials. In all the many experiments done in this way, I found no exception to the following remarkable property: the norm and the energy expectation value (since the potentials did not depend on time) of the wave function deviate from strict constancy during phases of very violent motion (where the spatial and temporal variations are clearly too large for the discretization length and time of the model) but come back to the original values with nearly the accuracy of the computer's number representation in all phases of calmer motion. No trend for changing the norm or the energy in the long run was observed. Notice that this is more than unitarity and energy conservation. A method enjoying the property of unitarity would conserve the disturbed value of the norm and would not restore it. I detected comparable stability properties whenever I applied the method characterized by equations (21) and (23) to physical systems, ranging from point particles, to rigid bodies, waves, and quantum particles. So, there seems to exists *the* method to start with, when the dynamics of physical systems is to be modeled on a computer. The implementation is nearly as simple as that of the Euler rule and whenever the discretization is chosen with a minimum of insight one will see the system evolve without the explosive auto-acceleration effects that the Euler rule will show after a few integration steps. The simplicity of the algorithm guarantees fast execution. Only if high accuracy it to be achieved with a small number of steps and in systems of sufficiently smooth forces, more specialized methods may offer substantial advantages. It is not clear whether it makes sense to ask for a reason behind this behavior that I always experienced as robustness, reliability, or - my highest ranking - naturalness. Here is my attempt: Since all fundamental equations of motion are at most of second order, locally parabolic trajectories are more basic than any trajectories of higher order. They are 'the stuff with which the internal logic of dynamics has to deal'. Equation (23) formulates a the most symmetric and only reasonable rule to determine a value for a (the only degree of freedom of a locally quadratic path with given initial position and velocity) that is intended to be constantly valid over a tiny but finite time span. I imagine that Leonhard Euler would have switched soon to that symmetrical mode of updating the acceleration in the center of the time step if he would have been interested in computing a finite piece of a trajectory. For doing it by thought experiment, the symmetrical mode offers no advantage since the number of acceptable time steps is unlimited then.

4 Application to the damped harmonic oscillator

Let us consider the one-dimensional harmonic oscillator with a Lagrangian and force written as

$$L(t,x,v) = \frac{m}{2}v^2 - \frac{k}{2}x^2 \,, \quad F(t,x,v) = -bv \,, \tag{27}$$

from which we get (see (6),(7))

$$M(t,x,v,a) = ma + bv + kx \,, \quad A(t,x,v) = -\frac{1}{m}(kx + bv) \,. \tag{28}$$

Here, (9) and (14) are easily calculated exactly, resulting in an expression which is linear in a. So, (18) also can be solved exactly:

$$a = -\frac{bv + k(x + v\tau)}{m + \tau(b + \frac{3}{5}k\tau)}\,, \tag{29}$$

The same is true for (19) which according to (24) gives an even simpler result:

$$a = -\frac{bv + k(x + v\tau)}{m + \tau b}\,. \tag{30}$$

Running (21) with these expressions may be compared to the known exact solution of equation

$$m\ddot{x} + b\dot{x} + kx = 0\,. \tag{31}$$

Most interesting is the somewhat artificial situation $b < 0$, in which there is no friction but a driving force which increases with speed. Then the amplitude of the exact solution grows exponentially. When multiplied by $e^{\frac{b}{2m}t}$, it becomes an exact harmonic oscillation. It is convenient to multiply the numerical solutions with the same factor and to observe the slow change in amplitude which then indicates a deviation from the exact solution. The following numbers are representative for the many numerical experiments which I ran. The number of steps for one oscillation period was 32 (as in the anharmonic oscillator example in [5]). The value of b was chosen such that the amplitude grows by a factor of 10^{32} in 1000 oscillations (by 7.6 % during one period). Then the Euler method lets the reduced amplitude grow by a factor of 2 shortly after the second oscillation period is completed. The Runge Kutta method of second order gives the same excess in period 153, method (29) does this in period 1642 and (30) gives a deviation of 1% in period 10068. Here, the amplitude of the exact amplitude is too large for being represented by a 64bit number and it is only due to some increased internal accuracy that the program did not stop on overflow some 300 steps before.

The method seems to create no trend in the amplitude error. So it is much more stable than the method (29) resulting from the exact solution of a variation problem. Thus, contrary to what I argued in [5], also the property of an integrator formula to represent an exact solution of the local form of the action principle has to be ruled out as a guide to understand this marvelous stability of the explicit midpoint method.

5 Acknowledgments

I'm grateful to J.Vermaseren for having made his great program FORM publicly available, and to my colleges Thomas Dera, Helmut Domes, and Eric Stelter for many suggestions on algorithms and computation.

References

[1] Arnold Sommerfeld: Vorlesungen über Theoretische Physik, Mechanik Verlag Harri Deutsch, 1977

[2] Herbert Goldstein: Klassische Mechanik, Akademische Verlagsgesellschaft,1963

[3] A.M. Stuart and A.R. Humphries: Dynamical Systems and Numerical Analysis, Cambridge University Press, 1996

[4] J.M. Sanz-Serna and M.P. Calvo: Numerical Hamiltonian Problems, Chapman & Hall, 1994

[5] U. Mutze: Predicting Classical Motion Directly from the Action Principle II. Mathematical Physics Preprint Archive 99-271

http://www.ma.utexas.edu/mp_arc/index-99.html, July 16

6 Appendix

```
FORM by J.Vermaseren,version 3.0(Jan 28 2001) Run at: Tue May  1 13:36:54 2001
    *****************************************************************
    * Ulrich Mutze 2001-4-23
    * Symbolic computation using the program FORM (by J.A.M. Vermaseren)
    * for proving Theorem 1 of
    * 'Predicting Classical Motion Directly from the Action Principle III'
    *******************************************************************

    #define N "16"
    * number of generalized coordinates. Is not treated on a symbolic level.
    * Actually one can vary this N and will always get the same result.
    #define p "3"
    * p=3 sufficient for getting all lowest order terms in expressions diff right

    Dimension 'N';
```

```
Symbols j,h,dt,L;
Functions Dt,Dxa,Dxv,Dva,H;
Vectors dx,dv,x,v,a,F;
Index i,k;
Off statistics;

Local dG0=Dt+Dva+Dxv;
* 0 refers to law h |--> (t+h, x+v*h,v+a*h) =: phi0(h)

Local dG1=Dt+Dva+Dxv+H*Dxa;
* 1 refers to law h |--> (t+h, x+v*h+a*h*h/2,v+a*h) =: phi1(h)

Local G0=1+sum_(j,1,‘p’,invfac_(j)*(h*dG0)^j);
Local G1=1+sum_(j,1,‘p’,invfac_(j)*(h*dG1)^j);

* after all of the following commutations are done, we have up to p-th
* order in h for each function f:RxRnxRn-->R :
* f(phi0(h))=(G0f)(h)
* f(phi1(h))=(G1f)(h)

repeat;
id Dt*Dxv=Dxv*Dt;
id Dt*Dxa=Dxa*Dt;
id Dt*Dva=Dva*Dt;
id Dt*H=1+H*Dt;
* Dt is derivation with respect to h, meaning of all derivation operators
* becomes clear with definitions (*) later
endrepeat;

id H=0;
* Taylor’s expansion takes all derivations for h=0

repeat;
id Dxa*Dxv=Dxv*Dxa+Dva;
id Dva*Dxv=Dxv*Dva;
endrepeat;

* now the definitions (*) which were refered to earlier:
id Dt=dt;
id Dxv=dx(k)*v(k);
id Dxa=dx(k)*a(k);
id Dva=dv(k)*a(k);

.sort
* .sort finishes a module, so that we may enter into a new
* fixed order cycle of
* 1. Declarations: starting with keywords Symbol(s), Function(s), ...
* 2. Specifications: e.g.statistics Off ...
* 3. Definitions: starting with keywords Local, ...
```

```
 * 4. Executable Statements: starting with keywords id ...
 * 5. Output control: such as Print and Bracket

 Symbols tau,dh,n;
 Local G0tau=dh*G0;
 * without this marking technique also the h in expressions G1 would be
 * changed to tau despite the encapsulation of the id statement in a module:
 id dh*h^n?=tau^n;
 .sort

 Symbols dy;
 *doing the midpoint derivative under the integral sign of the L term
 * the expression is calculated by hand
 Local deri1=-tau^-2*(h*(h-2*tau)*dx(i)+2*(h-tau)*dv(i))*L;
 Local deri2=
 tau^-2*(2*h*(2*tau-h)*a(k)*dx(i)*F(k)+2*(h-tau)*a(k)*dv(i)*F(k)-2*F(i));
 Local Si1=dy*deri1*G1;
 Local Si2=dy*(h/2)*(h-2*tau)*deri2*G1;
 * doing the integral
 id dy*h^n?=(2*tau)^(n+1)/(n+1);
 .sort

 * expressions for the motive force from midpoint derivative of the
 * action integral; 1 refers to the contribution of the Lagrangian,
 * 2 to the additional forces:
 Local MiAction=-3/(4*tau)*(Si1+Si2);

 * expressions for the motive force from Lagrange's differential expression.
 Local MiTau=((dv(i)*(dv(k)*a(k)+dx(k)*(v(k)+tau*a(k))+dt)-dx(i))*L-F(i))*G0tau;

 * Comparing the expressions:
 Local diff=MiAction-MiTau;

 Bracket tau;
 Print diff;
 * development should start with tau^2, notice that constant terms (that are
 * claimed not to exist) would appear at the end of the development

 .end;
```

```
diff =
    + tau^2 * (  - 1/10*dx(i)*dt^2*L - 1/5*dx(i)*dx.v*dv.a*L - 1/5*dx(i)*
      dx.v*dt*L - 1/10*dx(i)*dx.v^2*L - 3/5*dx(i)*dx.a*L - 1/5*dx(i)*dv.a*
      dt*L - 1/10*dx(i)*dv.a^2*L + 4/5*dx(i)*a.F + 1/10*dv(i)*dt^3*L + 4/5*
      dv(i)*dx.v*dx.a*L + 3/5*dv(i)*dx.v*dv.a*dt*L + 3/10*dv(i)*dx.v*dv.a^2
      *L + 1/5*dv(i)*dx.v*a.F + 3/10*dv(i)*dx.v*dt^2*L + 3/10*dv(i)*dx.v^2*
      dv.a*L + 3/10*dv(i)*dx.v^2*dt*L + 1/10*dv(i)*dx.v^3*L + 4/5*dv(i)*
      dx.a*dv.a*L + 4/5*dv(i)*dx.a*dt*L + 1/5*dv(i)*dv.a*a.F + 3/10*dv(i)*
      dv.a*dt^2*L + 3/5*dv(i)*dv.a*L + 3/10*dv(i)*dv.a^2*dt*L + 1/10*dv(i)*
```

```
  dv.a^3*L + 1/5*dv(i)*a.F*dt - 1/10*F(i)*dt^2 - 1/5*F(i)*dx.v*dv.a - 1/
  5*F(i)*dx.v*dt - 1/10*F(i)*dx.v^2 - 3/5*F(i)*dx.a - 1/5*F(i)*dv.a*dt
   - 1/10*F(i)*dv.a^2 )

+ tau^3 * (  - 1/10*dx(i)*dt^3*L - 4/5*dx(i)*dx.v*dx.a*L - 3/5*dx(i)*
  dx.v*dv.a*dt*L - 3/10*dx(i)*dx.v*dv.a^2*L + 4/5*dx(i)*dx.v*a.F - 3/10
  *dx(i)*dx.v*dt^2*L - 3/10*dx(i)*dx.v^2*dv.a*L - 3/10*dx(i)*dx.v^2*dt*
  L - 1/10*dx(i)*dx.v^3*L - 4/5*dx(i)*dx.a*dv.a*L - 4/5*dx(i)*dx.a*dt*L
   + 4/5*dx(i)*dv.a*a.F - 3/10*dx(i)*dv.a*dt^2*L - 4/15*dx(i)*dv.a*L -
  3/10*dx(i)*dv.a^2*dt*L - 1/10*dx(i)*dv.a^3*L + 4/5*dx(i)*a.F*dt - 1/6
  *dv(i)*dt^4*L - dv(i)*dx.v*dx.a*dv.a*L - dv(i)*dx.v*dx.a*dt*L + 2/5*
  dv(i)*dx.v*dv.a*a.F - 2*dv(i)*dx.v*dv.a*dt^2*L - 2*dv(i)*dx.v*dv.a^2*
  dt*L - 2/3*dv(i)*dx.v*dv.a^3*L + 2/5*dv(i)*dx.v*a.F*dt - 2/3*dv(i)*
  dx.v*dt^3*L - 1/2*dv(i)*dx.v^2*dx.a*L - 2*dv(i)*dx.v^2*dv.a*dt*L -
  dv(i)*dx.v^2*dv.a^2*L + 1/5*dv(i)*dx.v^2*a.F - dv(i)*dx.v^2*dt^2*L -
  2/3*dv(i)*dx.v^3*dv.a*L - 2/3*dv(i)*dx.v^3*dt*L - 1/6*dv(i)*dx.v^4*L
   - dv(i)*dx.a*dv.a*dt*L - 1/2*dv(i)*dx.a*dv.a^2*L + 1/5*dv(i)*dx.a*
  a.F - 1/2*dv(i)*dx.a*dt^2*L + 2/5*dv(i)*dv.a*a.F*dt - 2/3*dv(i)*dv.a*
  dt^3*L + 1/5*dv(i)*dv.a^2*a.F - dv(i)*dv.a^2*dt^2*L - 2/3*dv(i)*
  dv.a^3*dt*L - 1/6*dv(i)*dv.a^4*L + 1/5*dv(i)*a.F*dt^2 - 1/10*F(i)*
  dt^3 - 4/5*F(i)*dx.v*dx.a - 3/5*F(i)*dx.v*dv.a*dt - 3/10*F(i)*dx.v*
  dv.a^2 - 3/10*F(i)*dx.v*dt^2 - 3/10*F(i)*dx.v^2*dv.a - 3/10*F(i)*
  dx.v^2*dt - 1/10*F(i)*dx.v^3 - 4/5*F(i)*dx.a*dv.a - 4/5*F(i)*dx.a*dt
   - 4/15*F(i)*dv.a - 3/10*F(i)*dv.a*dt^2 - 3/10*F(i)*dv.a^2*dt - 1/10*
  F(i)*dv.a^3 )

+ tau^4 * ( 32/35*dx(i)*dx.v*dv.a*a.F + 32/35*dx(i)*dx.v*a.F*dt + 16/35
  *dx(i)*dx.v^2*a.F + 16/35*dx(i)*dx.a*a.F + 32/35*dx(i)*dv.a*a.F*dt +
  16/35*dx(i)*dv.a^2*a.F + 16/35*dx(i)*a.F*dt^2 - dv(i)*dx.v*dx.a*dv.a*
  dt*L - 1/2*dv(i)*dx.v*dx.a*dv.a^2*L + 12/35*dv(i)*dx.v*dx.a*a.F - 1/2
  *dv(i)*dx.v*dx.a*dt^2*L + 24/35*dv(i)*dx.v*dv.a*a.F*dt + 12/35*dv(i)*
  dx.v*dv.a^2*a.F + 12/35*dv(i)*dx.v*a.F*dt^2 - 1/2*dv(i)*dx.v^2*dx.a*
  dv.a*L - 1/2*dv(i)*dx.v^2*dx.a*dt*L + 12/35*dv(i)*dx.v^2*dv.a*a.F +
  12/35*dv(i)*dx.v^2*a.F*dt - 1/6*dv(i)*dx.v^3*dx.a*L + 4/35*dv(i)*
  dx.v^3*a.F + 12/35*dv(i)*dx.a*dv.a*a.F - 1/2*dv(i)*dx.a*dv.a*dt^2*L
   - 1/2*dv(i)*dx.a*dv.a^2*dt*L - 1/6*dv(i)*dx.a*dv.a^3*L + 12/35*dv(i)
  *dx.a*a.F*dt - 1/6*dv(i)*dx.a*dt^3*L + 4/35*dv(i)*dv.a*a.F + 12/35*
  dv(i)*dv.a*a.F*dt^2 + 12/35*dv(i)*dv.a^2*a.F*dt + 4/35*dv(i)*dv.a^3*
  a.F + 4/35*dv(i)*a.F*dt^3 )

+ tau^5 * ( 4/7*dx(i)*dx.v*dx.a*a.F + 8/7*dx(i)*dx.v*dv.a*a.F*dt + 4/7*
  dx(i)*dx.v*dv.a^2*a.F + 4/7*dx(i)*dx.v*a.F*dt^2 + 4/7*dx(i)*dx.v^2*
  dv.a*a.F + 4/7*dx(i)*dx.v^2*a.F*dt + 4/21*dx(i)*dx.v^3*a.F + 4/7*
  dx(i)*dx.a*dv.a*a.F + 4/7*dx(i)*dx.a*a.F*dt + 4/21*dx(i)*dv.a*a.F + 4/
  7*dx(i)*dv.a*a.F*dt^2 + 4/7*dx(i)*dv.a^2*a.F*dt + 4/21*dx(i)*dv.a^3*
  a.F + 4/21*dx(i)*a.F*dt^3 );
```