The Art of Computational Science

The Kali Code

vol. 3

Integration Algorithms: Exploring the Runge-Kutta Landscape

Piet Hut, Jun Makino and Douglas Heggie

September 14, 2007

Contents

Preface

In this volume, we present a pedagogic overview of the freedom one has in choosing the coefficients in explicit Runge-Kutta integration schemes. We limit ourselves to the simple cases in which there are at most two new evaluations of the right-hand side of the differential equation per time step. We start with first-order differential equations, to explain the general procedures, but then we limit ourselves to the type of second-order differential equation that occurs in classical mechanics, where the forces are dependent only on positions, and independent of the velocities. In that case we derive some of the classical Runge-Kutta-Nyström schemes, and generalize this to include a first evaluation that does not take place at the beginning of a time step. We show how such a generalized approach naturally leads us to the leapfrog-Verlet-Störmer-Delambre scheme, as a particular form of a generalized explicit Runge-Kutta scheme.

0.1 xxx

We thank Kristin Cordwell and xxx for their comments on the manuscript.

Piet Hut, Jun Makino, and Douglas Heggie

Chapter 1

First-Order Differential Equations

1.1 Starting at Square One

Bob: It has been a lot of fun, to derive so many different algorithms and to implement them all in our two-body code.

Alice: Yes, I enjoyed it too, and I must admit, I learned a lot in the process. But I still have the feeling that I'm missing some basic pieces of insight. Do you remember how we struggled, trying to prove that the Abramowicz and Stegun formula was correct, the fourth-order Runge-Kutta-Nyström scheme that had a misprint in it?

Bob: But you figured it out, didn't you?

Alice: Well, yes, after a false start. And it was a bit alarming that at first I didn't even realize that it was a false start. And to be completely honest, even now I'm not a full hundred percent sure that we got things right. Let me put it this way, I feel that I haven't yet gotten a finger-tip feeling for what Runge-Kutta schemes are, and how they really tick.

Bob: There must be several text books that you can look at. Surely they will explain things in more depth than you want to know.

Alice: I did look at some books on numerical methods, but none of them gave me what I really wanted to see. Some of them were just too mathematical in their concern and notation, others didn't provide the type of real detail that I wanted to see, yet others specialized on particular approaches. What I really would like to see is a pedestrian approach, no attempt to design special improvements. While I'm interested in all the extras, from embedded higher-order schemes to using extrapolation methods and symplectic schemes and what

have you, I really would like to first understand the basics better.

Bob: You mean, just the straightforward Runge-Kutta schemes of relatively low order, without any extra bells and whistles?

Alice: Exactly. Here is an idea. If we limit ourselves to performing at most two new force calculations per time step, things can't possibly get too complex. Our Abramowicz and Stegun formula already had three force calculations per time step, and I'm not suggesting that we explore explicitly the whole landscape around that formula, at least not yet.

Bob: So you want to explore a smaller landscape, just to see in front of your eyes how everything works. And while the simplest schemes, like forward Euler and leapfrog, use only one new force calculation per time step, you want to explore the full landscape of two new force calculations per time step. Hmm, I like that. And I'm sure it would be good for our students too, to see such an explicit survey.

Alice: I think so, but really, right now my main motivation is just for myself to see exactly how those classical Runge-Kutta derivations are done, from scratch, without taking anything on faith.

Bob: I like the idea, and I'm game. Where shall we start?

1.2 Keeping it Simple

Alice: One problem for astronomers using books on numerical solutions to differential equations is that most books focus on first-order differential equations. In contrast, we typically work with second-order differential equations, and often ones with special properties. The gravitational equations of motion for the N-body problem, for example, have a force term that is independent of time and velocity.

This suggests to me that we should divide our work into two stages. First we try to figure out how to solve a general first-order differential equation, using up to two force calculations per step. This will reproduce the results from the standard text books, no doubt, but it will give us experience and will allow us to establish a notation and a systematic procedure.

Then, in the second stage, we can cut our teeth on the gravitational N-body system, to see what special methods will work there, and why, and how. The Abramowicz and Stegun formula, for example, is tailored already to second-order differential equations, albeit a general one in which there is still a possible velocity dependence present in the force calculations. We can go one step further, specializing to position dependence only, and just see what spectrum of methods we will find.

Then, with a bit of luck, we will have gained enough experience to be able to look over the horizon, to get an idea what you could do with, say, three new

force calculations per step, which is the landscape within which the Abramowicz and Stegun formula was grown.

Bob: A somewhat ambitious project, but still quite doable, I think. You basically want to take the next step beyond forward Euler and leapfrog, in any possible direction, and see the dimensionality of the space of possible directions.

Alice: Something like that, yes. But let us restrict ourselves, at least at first, to Runge-Kutta methods. This will mean no multi-step methods, such as the original Aarseth scheme. It also means that we won't use higher derivatives, such as the Hermite scheme. In addition, we will exclude the use of implicit methods, which require iteration.

Bob: You could argue that, with two new force calculations per time step, you should allow implicit schemes that have just one new force calcuation per time step.

Alice: You could, even though it is not immediately clear that one iteration will provide you sufficiently rapid convergence. Also, the resulting class of implicit schemes is rather restricted. Perhaps we can look at that later. For now, I really want to be austere and stay to the absolute basics.

Bob: Okay: explicit Runge-Kutta methods using up to two new force calculations per time step, and no evaluations of jerks or anything else.

1.3 Notation

Alice: Let us start by choosing a specific notation. For the simplest form of differential equation, we can write:

$$\frac{dx}{dt} = f(x) \tag{1.1}$$

where we will call the variable x the position and the variable t the time. The solution of this equation is given by $x(t)$. When we solve this equation numerically, we use a finite time step τ. For now, we will analyze the properties of the first time step. We choose $t = 0$ at the beginning of the first time step, and we denote the positions at the beginning and end of the first time step by x_0 and x_1 , respectively:

$$x_0 \equiv x(0) \quad ; \quad x_1 \equiv x(\tau) \tag{1.2}$$

We can introduce the usual notation where a dot over a variable indicates the time derivative and a prime indicates the space derivative:

$$\dot{x} \equiv \frac{dx}{dt} \equiv \frac{d}{dt}x(t) \tag{1.3}$$

$$f' \equiv \frac{df}{dx} \equiv \frac{d}{dx} f(x) \tag{1.4}$$

If we now want to determine the time derivative of the force, we can use the chain rule, differentiating the force first with respect to its argument x, and multiplying the result with the time derivative of x:

$$\dot{f} \equiv \frac{d}{dt} f(x(t)) = \frac{dx}{dt} f' = \dot{x} f' \tag{1.5}$$

For the various derivatives of the position, we can introduce the historical notation in terms of velocity, acceleration, jerk, snap, crackle and pop, respectively:

$$\begin{cases} v \equiv \dot{x} = \frac{d}{dt} x(t) & ; \quad a \equiv \dot{v} = \frac{d^2}{dt^2} x(t) \quad ; \quad j \equiv \dot{a} = \frac{d^3}{dt^3} x(t) \\ s \equiv \dot{j} = \frac{d^4}{dt^4} x(t) & ; \quad c \equiv \dot{s} = \frac{d^5}{dt^5} x(t) \quad ; \quad p \equiv \dot{c} = \frac{d^6}{dt^6} x(t) \end{cases} \tag{1.6}$$

These expressions are especially useful for the type of second-order differential equation encountered in classical mechanics:

$$\ddot{x} \equiv \frac{d^2 x}{dt^2} = f(x) \tag{1.7}$$

which can be written as a system of two first-order differential equations:

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= f(x) \end{cases} \tag{1.8}$$

However, for now we will stick to the first-order differential equation, using the general expression that we started with, but without any explicit time dependence.

1.4 A Matter of Interpretation

Bob: Even though this is just a warming-up exercise, it would be nice to give a physical interpretation to the first-order differential equation that you wrote down:

$$\frac{dx}{dt} = f(x) \tag{1.9}$$

You have been calling $f(x)$ a force, but that doesn't seem right. This equation tells us that the velocity is prescribed, and equal to $f(x)$. A true force would give rise to an acceleration, not a velocity.

Alice: In principle that is correct, but in practice, if we have a lot of resistance, it is the velocity that is proportional to the force. If you move a spoon through molasses, you have to push twice as hard to go twice as fast.

Bob: But even in that case, the initial acceleration must still be proportional to the applied force.

Alice: Yes, but only very briefly. As soon as you pick up a very small amount of speed, friction starts to resist, canceling part of your force. So after the initial transients die out, the velocity settles to a constant value, proportional to the force you use. From than on, in the limit of changes that are slow with respect to the duration of the transients, the acceleration is proportional to the *rate of change* of the force, not to the magnitude of the force.

Bob: I don't like the idea of posing a problem, and then neglecting the interesting part of the solution, namely the transients.

Alice: So for once you are looking for a more clearly abstract model; I thought you would like a quick and dirty physics example!

Bob: Molasses may indeed be too dirty for me. Why don't we stick with considering $f(x)$ as a velocity.

Alice: But the left hand side of the differential equation is a velocity. The right-hand side has to be something else. In Newtonian dynamics we have $f = ma$, which means that the acceleration is proportional to the net force acting on the body. You now want to have a velocity, but you have to specify what it is that is imposing itself on your particle to produce that velocity.

Bob: Well, yeah, hmmm, let's see, that's not so clear.

Alice: Forgive the pun, but why don't we stick to molasses?

Bob: Ah, I got it! Hey, elementary, my dear Watson. If a particle would be rolling down a potential well, without any friction, the total energy would be constant. If we write $\Phi(x)$ for the potential energy per unit mass, and E for the total energy per unit mass, then the velocity can be expressed as:

$$v(t) = \sqrt{2E - \Phi(x(t))} \tag{1.10}$$

Alice: Bravo, that works. Interesting. I hadn't even considered such a possibility, probably because I started out calling $f(x)$ a force from the beginning.

Bob: Okay, so we're talking now about a particle in a potential well.

Alice: You may, but I still prefer to talk about molasses, since in that case we can make a more smooth transition to the case of a second order differential equation.

1.5 Taylor Series

Bob: I still prefer my interpretation. Let's just agree to disagree.

Alice: Fine with me, since, after all, the math will be the same.

Bob: Exactly. Okay, let's get to work. You have struggled with these things a lot more than I have. How do we get started?

Alice: We want to check the quality of any given numerical approximation scheme to the solution of our differential equation. In order to do so, we can compare such a scheme with a Taylor series development of the true solution, around the starting point of our one integration step.

In other words, we can express the position at the end of one time step in the following Taylor series:

$$x_1 = x_0 + v_0\tau + \tfrac{1}{2}a_0\tau^2 + \tfrac{1}{6}j_0\tau^3 + \tfrac{1}{24}s_0\tau^4 + O(\tau^5) \tag{1.11}$$

The velocity at time zero is given directly by the differential equation. The higher derivatives of the position, starting with the acceleration, can be found by differentiating both sides of the differential equation, one or more times. This leads to expressions such as:

$$\begin{aligned}
v_0 &= f(x(0)) = f(x_0) = f_0 \\
a_0 &= \frac{d}{dt}v(t)\bigg|_{t=0} = \frac{d}{dt}f(x(t))\bigg|_{t=0} = \frac{df(x)}{dx}\bigg|_{x=x_0}\frac{dx}{dt}\bigg|_{t=0} = f_0' v_0 = f_0' f_0 \\
j_0 &= \dot{a}_0 = f_0^2 f_0'' + f_0 (f_0')^2 \\
s_0 &= \dot{j}_0 = f_0^3 f_0''' + 4 f_0^2 f_0' f_0'' + (f_0')^3 f_0
\end{aligned} \tag{1.12}$$

The last two lines can be derived in the same way as the second line, by fully writing out the differentiations, using the chain rule.

By the way, here the acceleration comes out nicely as the rate of change of the force applied, as would happen for a spoon moving slowly through molasses.

Bob: That would take a lot of getting used to! For me, the acceleration is just the rate of change of the velocity.

Alice: But isn't that a tautology? After all, the acceleration is by definition the rate of change of the velocity, as a mathematical construction. I thought we were trying to come up with a physical system as an example.

Bob: But a potential well is surely a physical system! And what I thought is that we had agreed to disagree.

Alice: I agree!

1.6 New Force Evaluations

Bob: Me too. Coming back to our task, I like the systematic approach idea, of using up to two new force evaluations per time step. Well, this gives us two choices: either one or two force evaluations.

Alice: Actually, there are four choices. In each case, we can try to recycle a previous force calculation in the next step, or we don't.

Bob: You mean that you use the last force calculation, at the end of a given step, as the first force value that you use for the next step?

Alice: Exactly. And this will put rather strict conditions on the nature of that force calculation.

Bob: It means, of course, that a force calculation needs to take place at the boundary of two steps, otherwise you can't recycle it. But that doesn't seem to be a particularly severe restriction to me.

Alice: In principle, you could even recycle a force that is used in the middle, if you would be willing to used the remembered values of the previous step, you could still recycle. However, that would mean that we would go beyond Runge-Kutta methods, and enter the area of multi-step methods.

Bob: Let's not get into that, at least not know. I'd be happy to first explore the landscape of Runge-Kutta algorithms. Okay, as long as we let our last force calculation occur at the end of a step, we can recycle that calculation for the next step.

Alice: Oh, no, it's not that simple. In a general Runge-Kutta approach, you compute a few forces here and there, and only after doing that, you combine those forces in such a way as to get a combination of them, to give you a value of the new position accurate to high order.

Now the force that you would evaluate at that new position, at the beginning of the next time step, will in general not be the same as the force that you have calculated at the end of the current time step. Even though it was evaluated at the same time, it will in general be evaluated at a slightly different place. The reason is that at the time of evaluation, you didn't yet have in hand the most accurate estimate for the new position.

Bob: Hmm, that's tricky. I hadn't thought about that.

Alice: I hadn't either, until I started playing with some of those schemes in detail. All the more reason to take a really pedestrian approach, and just write everything out, to make sure we're not overlooking something or jumping to conclusions!

To start with, let us *not* try to recycle any forces. Within that category of

attempts, we will first investigate what can happen when we allow just one force evaluation per step, and then we will move on to two force evaluations per step. After that, we'll look at recycling.

Bob: Fair enough!

1.7 One Force Evaluation per Step

Alice: At the start of a time step, the only evaluation of the right-hand side of the differential equation that is possible is the one at $t = 0$:

$$k_1 = f(x_0) \tag{1.13}$$

This leads to the following dimensionally correct expression:

$$x_1 = x_0 + \alpha_1 k_1 \tau \tag{1.14}$$

Combining the last two equations, we have

$$x_1 = x_0 + \alpha_1 f_0 \tau \tag{1.15}$$

We can compare this expression with our Taylor series:

$$x_1 = x_0 + v_0\tau + \tfrac{1}{2}a_0\tau^2 + O(\tau^3) \tag{1.16}$$

Using Eqs. (1.12) we can write this as

$$x_1 = x_0 + f_0\tau + \tfrac{1}{2}f_0 f_0'\tau^2 + O(\tau^3) \tag{1.17}$$

How accurate is our new value x_1 after we take one step? Let us see how well we can match Eq. (1.15) with Eq. (1.17), in successive powers of τ. The constant term x_0 matches trivially, and our first condition arises from the term linear in τ:

$$\alpha_1 f_0 = f_0 \tag{1.18}$$

hence

$$\boxed{\alpha_1 = 1} \tag{1.19}$$

Equation 1.19 looks ok.

We have no free parameter left, so this leads us to the only possible explicit first-order integration scheme

$$\begin{cases} x_1 &= x_0 + k_1\tau \\ k_1 &= f(x_0) \end{cases} \tag{1.20}$$

which is the forward-Euler algorithm. This scheme is only first-order accurate, since it cannot possibly reproduce the $O(\tau^2)$ term in Eq. (1.17): such a match would require $\frac{1}{2} f_0 f_0' = 0$, which is certainly not true for general force prescriptions.

Bob: Of course, forward Euler is the simplest possible scheme. It is what anyone would have guessed, if they had guessed any scheme at all

Alice: That may be true, but I, for one, like to see a *derivation* for any integration scheme, even the simplest and humblest of them all. It is all nice and fine to say that something is intuitively obvious, but I am much happier if you can *prove* that something is not only simple, but actually the simplest, and under certain plausible restrictions, the *only* one of its kind.

Bob: Can't argue about taste. I can see your point, but any good point can be pressed to extremes. Well, as long as you do the calculating, I'll sit back and relax.

1.8 Two Force Evaluations per Step

Alice: Now we can move to more interesting venues, when we allow two force evaluations per step. After a first evaluation of the right-hand side of the differential equation, we can perform a preliminary integration in time, after which we can evaluate the right-hand side again, at a new position:

$$k_1 = f(x_0) \tag{1.21}$$
$$k_2 = f(x_0 + \eta k_1 \tau) \tag{1.22}$$

We can now use a more general expression for the new position, in which we rely on the preliminary information that has been gathered in the two force evoluations. Since each force evaluation has the dimension of a time derivative of the position, we have to multiply each one with a single power of τ. The coefficient for each term is, as yet, arbitrary, so let us parametrize them as follows:

$$x_1 = x_0 + (\alpha_1 k_1 + \alpha_2 k_2)\,\tau \tag{1.23}$$

We can combine these equations, and write them as

$$x_1 = x_0 + \{\alpha_1 f_0 + \alpha_2 f(x_0 + \eta f_0 \tau)\}\,\tau \tag{1.24}$$

Bob: Our strategy is again to compare this expression with a Taylor series defined at the start of the time step, right?

Alice: Yes. And since that Taylor series for x_1 is defined as a series in τ , we must somehow translate the above expressions, too, a series in τ , around $\tau = 0$.

Bob: The main obstacle here is k_2 , which itself involves an expression that depends on τ.

Alice: The solution here is that we can develop k_2 itself in a Taylor series around $\tau = 0$. In general, for any function of , we can write the Taylor series for a position $x + \epsilon$ near x as:

$$f(x + \epsilon) = f(x) + \epsilon f'(x) + \tfrac{1}{2}\epsilon^2 f''(x) + O(\epsilon^3) \tag{1.25}$$

In our particular case, this gives us:

$$k_2 = f(x_0 + \eta f_0 \tau) = f_0 + (\eta f_0) f_0' \tau + \tfrac{1}{2}(\eta f_0)^2 f_0'' \tau^2 + O(\tau^3) \tag{1.26}$$

We thus find for the new position, at the end of our time step:

$$x_1 = x_0 + (\alpha_1 + \alpha_2) f_0 \tau + \alpha_2 \eta f_0 f_0' \tau^2 + \tfrac{1}{2}\alpha_2 \eta^2 f_0^2 f_0'' \tau^3 + O(\tau^4) \tag{1.27}$$

We can now compare this expression with the Taylor series expansion of the true orbit:

$$x_1 = x_0 + v_0 \tau + \tfrac{1}{2} a_0 \tau^2 + \tfrac{1}{6} j_0 \tau^3 + O(\tau^4) \tag{1.28}$$

Using Eq. (1.12), we can write this as

$$x_1 = x_0 + f_0 \tau + \tfrac{1}{2} f_0 f_0' \tau^2 + \tfrac{1}{6}\left\{ f_0^2 f_0'' + f_0 (f_0')^2 \right\} \tau^3 + O(\tau^4) \tag{1.29}$$

To what order can we make Eqs. (1.27) and (1.29) compatible? Starting with terms to first order in τ , we have to insist that

$$(\alpha_1 + \alpha_2) f_0 = f_0 \tag{1.30}$$

which leads to the condition

$$\boxed{\alpha_1 + \alpha_2 = 1} \tag{1.31}$$

To second order in τ , we would like to satisfy:

$$\alpha_2 \eta f_0 f_0' = \tfrac{1}{2} f_0 f_0' \tag{1.32}$$

which can be done through the condition

$$\boxed{\eta = \frac{1}{2\alpha_2}} \tag{1.33}$$

Would it be possible to match Eqs. (1.27) and (1.29) also to third order in τ ? This would require

$$\tfrac{1}{2}\alpha_2\eta^2 f_0^2 f_0'' = \tfrac{1}{6}\left\{f_0^2 f_0'' + f_0(f_0')^2\right\} \tag{1.34}$$

While we can match the first term on the right-hand side, by choosing $\alpha_2\eta^2 = 1/3$, this would require that $f_0(f_0')^2 = 0$, which is not true for general force prescriptions.

1.9 A One-Parameter Family of Algorithms

Bob: So we have to conclude that our scheme is only second-order accurate. That makes sense: with one force evaluation, we got a first-order scheme, and with two force evaluations, we get a second-order scheme. Presumably with p force evaluations, you get a scheme that is accurate to order p.

Alice: I would have guessed so too, but this is not so. Your guess is correct for order 3 and 4, but it turns out that you need 6 force evaluations to build an algorithm that is accurate to order 5!

Bob: That is surprising!

Alice: It is, until you realize that you get more and more equations that you have to satisfy. The number of such conditions grows quite a bit faster than the the number of force calculations. This is not yet obvious in what we have done, but if will become obvious pretty soon. In general, there are a lot of complicated combinatorial surprises in Runge-Kutta derivations.

Bob: Fascinating. But for now, at least, it seems that going to higher order gives us more freedom, rather than less. Unlike the first-order case, we now have an extra parameter to play with.

We started with three free parameters, α_1 , α_2 , and η. Since we only have the two boxed conditions above, we can expect to be left with one degree of freedom in choosing the coefficients in our algorithm.

Alice: Well, let's check. If we define $\alpha \equiv \alpha_2$, we find:

$$\left\{\begin{array}{lcl} \alpha_1 &=& 1-\alpha \\ \alpha_2 &=& \alpha \\ \eta &=& 1/(2\alpha) \end{array}\right. \tag{1.35}$$

We thus obtain the following one-parameter family of algorithms:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + ((1-\alpha)k_1 + \alpha k_2)\,\tau \\ \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \frac{1}{2\alpha}k_1\tau) \end{array}\right. \tag{1.36}$$

Bob: Ah, this is nice. I recognize some of the algorithms that I've been using in the past. One classical choice for a second-order Runge-Kutta is $\alpha = \frac{1}{2}$, leading to:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + \frac{1}{2}\,(k_1 + k_2)\,\tau \\ \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + k_1\tau) \end{array}\right. \tag{1.37}$$

This one goes under the name 'improved Euler scheme.'

Another classical choice is $\alpha = 1$, which gives:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + k_2\tau \\ \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \frac{1}{2}k_1\tau) \end{array}\right. \tag{1.38}$$

This integration scheme is called the 'midpoint scheme.'

Alice: Yes, and now we have given a derivation for why they work. In general, for higher-order algorithms, you have to follow such a derivation to convince yourself that the recipe has the order that is claimed for it. However, in this second-order case, you can still use your intuition to convince yourself that the expressions are okay.

For $\alpha = \frac{1}{2}$, we effectively average the evaluations at the beginning and at the end of the trial step, and you can imagine that this gives you one extra order of accuracy, since you effectively cancel the types of error you would make if you were using a force calculation only at one end of the step.

Similarly, for $\alpha = 1$, we use the evaluation at the end of a smaller trial step that brings us approximately mid-way between the beginning and the end of the step. Then, at that point, we again obtain an estimate for the average of the forces at begin and end of the time step.

Chapter 2

Recycling Force Evaluations

2.1 One Force Evaluation per Step

Alice: So far, we have used up to two force calculations per time step, independently of what has been done in the previous time step. As we discussed before, there are situations in which we can recycle a previous force calculation.

To be specific, taking the result from the previous section, Eq.(1.36)

$$\left\{ \begin{array}{lcl} x_1 & = & x_0 + ((1-\alpha)k_1 + \alpha k_2)\,\tau \\ & & \\ k_1 & = & f(x_0) \\ k_2 & = & f(x_0 + \frac{1}{2\alpha}k_1\tau) \end{array} \right. \tag{2.1}$$

would it be possible to use the force evaluation k_2 of the first step, and to recycle its use, to let it function as the k_1 contribution to the second step?

Bob: Not really, no. At least, I don't think so. The first force calculation for the second step will be evaluated at x_1 , the end point of the first step. However, the last force calculation for the first step was not evaluated at that exact point. Rather it was evaluated at the point that was reached by using only the information given by k_1.

Alice: In general, you must be right. But let's not jump to conclusions; the whole point of our systematic approach is to really make sure that our hunches are correct, by deriving everything to the point of reaching absolute certainty.

Bob: what you call systematic others may call tedious, or worse.

Alice: So be it; I just want to be sure. So, for recycling to work in the strict sense, the position at which k_2 is calculated during the first step should coincide with the position at which k_1 needs to be calculated during the second step. Let us define that first position as $\tilde{x}_1$, which means that $k_2 \equiv f(\tilde{x}_1)$, which implies

$$\tilde{x}_1 = x_0 + \frac{1}{2\alpha} k_1 \tau \tag{2.2}$$

Recycling the last force calculation from the first step, in order to use it for the second step, requires that $\tilde{x}_1 = x_1$:

$$x_0 + \frac{1}{2\alpha} k_1 \tau = x_0 + ((1-\alpha)k_1 + \alpha k_2)\,\tau \tag{2.3}$$

or

$$k_1 = 2\alpha\,((1-\alpha)k_1 + \alpha k_2) \tag{2.4}$$

or

$$2\alpha^2\,(k_1 - k_2) = (2\alpha - 1)k_1 \tag{2.5}$$

Since our algorithm should work for any force $f(x)$, this expression should hold for arbitrary values of k_1 and k_2. If we first look at the dependence on k_2 , we find $2\alpha^2 k_2 = 0$ and therefore $\alpha = 0$. But this then implies that $k_1 = 0$, which is not true in general.

So here is the formal check that your hunch was right!

Bob: This result is not surprising, when we reflect on what it means: if the equality $\tilde{x}_1 = x_1$ would hold exactly, there would be no reason to compute the last force evaluation. For this reason, there should not be any Runge-Kutta scheme that allows strict recycling of a force evaluation, come to think of it.

2.2 What is Good Enough?

Alice: That must be right. The best we can hope for is that $\tilde{x}_1$ is reasonably accurate as a predicted value, good enough, so to speak. Now the question is whether we can find a precise meaning for 'good enough.' What does it mean for $\tilde{x}_1$ not to differ too much from the corrected value x_1 ?

Bob: Lookinging at Eq. (2.5) as a physicist, rather than a mathematician, I would start by noting that $k_1 \approx k_2$, at least in the limit of a small time step. This suggests that the best we can do is to let the right hand side disappear, through the choice $\alpha = \frac{1}{2}$. In that case, the left-hand side will still not be exactly zero, but it will be small.

Alice: Even though you're a physicist, you should at least show that this choice brings $\tilde{x}$ and x close together. Handwaving alone is certainly not good enough!

Bob: Okay, if you insist. For $\alpha = \frac{1}{2}$ we can determine the difference between the two force evaluations as:

$$\begin{aligned} k_1 - k_2 &= f(x_0) - f(x_0 + k_1\tau) \\ &= -f_0' k_1 \tau + O(\tau^2) = -f_0' f_0 \tau + O(\tau^2) \end{aligned} \tag{2.6}$$

This translates into a difference between the two recycle points of:

$$\begin{aligned} \tilde{x}_1 - x_1 &= \{x_0 + k_1\tau\} - \left\{x_0 + \tfrac{1}{2}(k_1 + k_2)\tau\right\} \\ &= \tfrac{1}{2}(k_1 - k_2)\tau \\ &= -\tfrac{1}{2} f_0' f_0 \tau^2 + O(\tau^3) \end{aligned} \tag{2.7}$$

Alice: That strenghtens your argument quite a bit, I'd say. Still, I sometimes like to play the mathematician. While your result is a good one, it is not yet fully clear that it is the optimal one.

Bob: It's clear to me. What else could be better?

Alice: I think I agree, but for future reference, I would like to give a formal derivation. Soon we will get to much more complicated situations, where we can't use intuition anymore, and I would like to see exactly how I can prove that this is the best choice. So bear with me, while I try to minimize the difference between $\tilde{x}_1$ and x_1 directly, starting from the most general form:

$$\begin{aligned} \tilde{x}_1 - x_1 &= \left\{x_0 + \frac{1}{2\alpha} k_1 \tau\right\} - \{x_0 + ((1-\alpha)k_1 + \alpha k_2)\tau\} \\ &= \frac{1}{2\alpha} k_1 \tau + (\alpha - 1) k_1 \tau - \alpha k_2 \tau \\ &= \left(\alpha - 1 + \frac{1}{2\alpha}\right) k_1 \tau - \alpha k_2 \tau \end{aligned} \tag{2.8}$$

As before, we can write $k_1 = f_0$ and use the expansion

$$k_2 = f(x_0 + \frac{1}{2\alpha} k_1 \tau) = f_0 + \frac{1}{2\alpha} f_0 f_0' \tau + O(\tau^2) \tag{2.9}$$

which gives for Eq. (2.8):

$$\begin{aligned} \tilde{x}_1 - x_1 &= \left(\alpha - 1 + \frac{1}{2\alpha}\right) f_0 \tau - \alpha \left(f_0 + \frac{1}{2\alpha} f_0 f_0' \tau\right) \tau + O(\tau^3) \\ &= \left(\frac{1}{2\alpha} - 1\right) f_0 \tau - \tfrac{1}{2} f_0 f_0' \tau^2 + O(\tau^3) \end{aligned} \tag{2.10}$$

In order to let the first order term vanish, we regain our previous results: $\alpha = \frac{1}{2}$ is the best approximation, and the remaining term is second order in τ.

Bob: I told you so! And for good measure, let me give you another physical intuition derivation. At the beginning of the second step, we can only recycle a previous force if that force was performed at the end of the previous step. In first approximation, given the force f_0 at x_0 , we can write $x_1 = x_0 + f_0\tau + O(\tau^2)$. Comparing this with Eq. (2.1), we see immediately that $1/(2\alpha) = 1$, hence $\alpha = \frac{1}{2}$.

Alice: Yes, I fully agree that it is helpful to look at the results from several angles, to get more of a fingertip feeling of what it all means. Still, I wouldn't have been fully happy without a formal derivation. But let's move on.

2.3 Approximate Recycling

Bob: The question is, can we use our buest guess, or in your case, best derivation, for recycling?

Alice: At first sight, the second-order offset in Eqs. (2.7) and (2.10) may seem problematic, since we are aiming at developing a second order algorithm, with third-order errors. However, when we recycle the last force calculation in the next step we will always use it in multiplication with an extra power of τ. This means that the slight offset will cause only third order errors, on the same level of the truncation errors we are making anyway.

To show this explicitly, let us extend our notation, using $k_{i,j}$ to denote k_j for the step starting at x_i , and let us use tildes to indicate the approximate solution that we obtain when we recycle the previous force evaluation. Here are the expressions for the first step:

$$\left\{\begin{array}{lcl} x_1 & = & x_0 + \frac{1}{2}\left(k_{0,1} + k_{0,2}\right)\tau \\ k_{0,1} & = & f(x_0) \\ k_{0,2} & = & f(x_0 + k_{0,1}\tau) \end{array}\right. \tag{2.11}$$

Here is the correct second step without recycling:

$$\left\{\begin{array}{lcl} x_2 & = & x_1 + \frac{1}{2}\left(k_{1,1} + k_{1,2}\right)\tau \\ k_{1,1} & = & f(x_1) \\ k_{1,2} & = & f(x_1 + k_{1,1}\tau) \end{array}\right. \tag{2.12}$$

And here is the approximate second step when we use recycling:

$$\left\{\begin{array}{lcl} \tilde{x}_2 & = & x_1 + \frac{1}{2}\left(\tilde{k}_{1,1} + \tilde{k}_{1,2}\right)\tau \\ \tilde{k}_{1,1} & = & k_{0,2} \\ \tilde{k}_{1,2} & = & f(x_1 + \tilde{k}_{1,1}\tau) \end{array}\right. \tag{2.13}$$

More generally, we can express step number i without recycling as:

$$\left\{ \begin{array}{lcl} x_{i+1} & = & x_i + \frac{1}{2}\left(k_{i,1} + k_{i,2}\right)\tau \\ k_{i,1} & = & f(x_i) \\ k_{i,2} & = & f(x_i + k_{i,1}\tau) \end{array} \right. \tag{2.14}$$

and with recycling as:

$$\left\{ \begin{array}{lcl} \tilde{x}_{i+1} & = & x_i + \frac{1}{2}\left(\tilde{k}_{i-1,2} + \tilde{k}_{i,2}\right)\tau \\ \tilde{k}_{i,2} & = & f(x_i + \tilde{k}_{i-1,2}\tau) \end{array} \right. \tag{2.15}$$

At each step, the difference between $\tilde{x}_{i+1}$ and x_{i+1} is of third order in τ , as we can illustrate by evaluating down the differences in position at the end of the second step:

$$\tilde{x}_2 - x_2 = \frac{1}{2}\left(\tilde{k}_{1,1} - k_{1,1}\right)\tau + \left(\tilde{k}_{1,2} - k_{1,2})\right)\tau \tag{2.16}$$

Using Eq. (2.10), we can expand the first term on the right hand side as follows

$$\begin{array}{rcl} \tilde{k}_{1,1} - k_{1,1} & = & f(\tilde{x}_1) - f(x_1) = (\tilde{x}_1 - x_1)f'(x_1) + O(\tau^3) \\ & = & -\frac{1}{2}f_0 f_0' f'(x_0 + \frac{1}{2}(k_{0,1} + k_{0,2})\tau)\tau^2 + O(\tau^3) \\ & = & -\frac{1}{2}f_0(f_0')^2\tau^2 + O(\tau^3) \end{array} \tag{2.17}$$

This result can in turn be used to expand the second term on the right-hand side of Eq.(2.16):

$$\begin{array}{rcl} \tilde{k}_{1,2} - k_{1,2} & = & f(x_1 + \tilde{k}_{1,1}\tau) - f(x_1 + k_{1,1}\tau) \\ & = & \left(\tilde{k}_{1,1} - k_{1,1}\right) f'(x_1)\tau + O(\tau^4) \\ & = & -\frac{1}{2}f_0(f_0')^2 f'(x_0 + \frac{1}{2}(k_{0,1} + k_{0,2})\tau)\tau^3 + O(\tau^4) \\ & = & -\frac{1}{2}f_0(f_0')^3\tau^3 + O(\tau^4) \end{array} \tag{2.18}$$

This means that in Eq. (2.16) the first term on the right-hand side dominates, and we find:

$$\tilde{x}_2 - x_2 = -\frac{1}{2}f_0(f_0')^2\tau^3 + O(\tau^4) \tag{2.19}$$

This is the promised result: recycling the force calculation at the end of one step introduces an extra error in the next step which is third order in τ. Since our basic algorithm is only second-order accurate in τ per step, the only effect is to change the magnitude of the leading error term, without affecting the second-order nature of the algorithm.

2.4 Summary

Bob: Great! So there is a place for recycling, after all. And the scheme we have found, for $\alpha = 1/2$ is just one of the classic second-order Runge-Kutta schemes, the one we already wrote down in Eq. (1.37). I had no idea that that algorithm could be used in a recycling fashion.

Alice: I didn't either. Normally, it is presented in the text books as a scheme where you simply have to evaluate the force two times in every step.

Bob: Most likely, the accuracy will be less per time step. However, if force evaluation is the most expensive part of the calculation, as it certainly is for the N-body problem, switching to recycling allows us to take a step size that is two times smaller, for the same number of force calculations.

Alice: That probably means that it depends on the particular application whether recycling is a good idea or not. Making the step size two times smaller means that the error per step will become eight times smaller, and the error for a fixed time interval four times smaller, at least approximately. If the extra error introduced by recycling makes the calculation error more than four times larger, it is not a good idea.

Bob: At least we have an extra tool in our toolbox. I like gathering extra algorithms! It would be fun to see under which circumstances we get a better result.

Alice: But not right now. I prefer to continue first our systematic investigation with paper and pencil, before we start coding things up again.

Bob: Fine.

Alice: Let us summarize what we have learned so far.

- If we use only one force calculation per step, without recycling any force evaluation results, we have to settle for a first-order scheme, the forward-Euler algorithm, Eq. (1.20).

- If we use two force calculations per step, without recycling any force evaluation results, we find a one-parameter family of second-order scheme, the classical second-order Runge-Kutta algorithms, given in Eq. (1.36).

- If we use only one new force calculation per step, but in addition we recycle the last force calculation from the previous step, we have the best of both worlds: we obtain a second-order scheme for the same price in terms of number of force evaluations as the first-order scheme. This clever scheme is given in Eq. (2.15), as the recycled version of what otherwise be Eq. (2.14), which is the same as Eq. (1.36) for $\alpha = \frac{1}{2}$, also given above as Eq. (1.37).

2.5 Two Force Evaluations per Step

Bob: You would think that we can now add a third force calculation per step, while recycling the last one. This would mean to new force calculations and one recycled one per step. And just as we found a second-order scheme when using one old and one new force, I seems pretty clear that we can now find a third-order scheme, using one old and two new forces.

Alice: I agree that that seems likely, but there is no guarantee. Remember that you can obtain a fourth-order scheme with four forces, but that a fifth-order scheme requires six forces. These combinatoric questions cannot be derived by analogy; I'm afraid we just will have to do the hard work of deriving them.

Our first task is to write the form of a general Runge-Kutta scheme with three force calculations per time step. Once we have this form, we can insist on the extra condition that the position of the final force calculation coincides with the position at the beginning of the next time step, at least to within second order in τ.

The general three-stage Runge-Kutta scheme looks like this:

$$\begin{aligned} k_1 &= f(x_0) \\ k_2 &= f(x_0 + \eta_{21} k_1 \tau) \\ k_3 &= f(x_0 + \eta_{31} k_1 \tau + \eta_{32} k_2 \tau) \\ \\ x_1 &= x_0 + (\alpha_1 k_1 + \alpha_2 k_2 + \alpha_3 k_3)\, \tau \end{aligned} \tag{2.20}$$

Our analysis proceeds as before, but with more complex terms. Instead of Eq. (1.24), we now have

$$\begin{aligned} x_1 = x_0 + \{&\alpha_1 f_0 + \alpha_2 f(x_0 + \eta_{21} f_0 \tau) + \\ &\alpha_3 f(x_0 + \eta_{31} f_0 \tau + \eta_{32} f(x_0 + \eta_{21} f_0 \tau)\tau)\}\, \tau \end{aligned} \tag{2.21}$$

Instead of Eq. (1.26) we have

$$\begin{aligned} k_2 &= f(x_0 + \eta_{21} f_0 \tau) \;= \\ &\quad f_0 + \eta_{21} f_0 f_0' \tau + \tfrac{1}{2} \eta_{21}^2 f_0^2 f_0'' \tau^2 + O(\tau^3) \end{aligned} \tag{2.22}$$

The expression for the next force evaluation can be derived similarly:

$$k_3 = f_0 + (\eta_{31} k_1 + \eta_{32} k_2)\, f_0' \tau +$$

$$\begin{aligned} & \tfrac{1}{2}\left(\eta_{31}k_1+\eta_{32}k_2\right)^2 f_0''\tau^2+O(\tau^3) \\ = \;& f_0+\left(\eta_{31}+\eta_{32}\right)f_0f_0'\tau+\eta_{32}\eta_{21}f_0\left(f_0'\right)^2\tau^2+ \\ & \tfrac{1}{2}\left(\eta_{31}+\eta_{32}\right)^2 f_0^2f_0''\tau^2+O(\tau^3) \end{aligned} \tag{2.23}$$

We thus find for the new position, at the end of our time step, as the generalization of Eq. (1.27)

$$\begin{aligned} x_1 \;=\; & x_0+(\alpha_1+\alpha_2+\alpha_3)f_0\tau+ \\ & \alpha_2\eta_{21}f_0f_0'\tau^2+\tfrac{1}{2}\alpha_2\eta_{21}^2f_0^2f_0''\tau^3+ \\ & \alpha_3\left(\eta_{31}+\eta_{32}\right)f_0f_0'\tau^2+\alpha_3\eta_{32}\eta_{21}f_0\left(f_0'\right)^2\tau^3+ \\ & \tfrac{1}{2}\alpha_3\left(\eta_{31}+\eta_{32}\right)^2f_0^2f_0''\tau^3+\;O(\tau^4) \\ = \;& x_0+(\alpha_1+\alpha_2+\alpha_3)f_0\tau+ \\ & \left(\alpha_2\eta_{21}+\alpha_3\left(\eta_{31}+\eta_{32}\right)\right)f_0f_0'\tau^2+ \\ & \tfrac{1}{2}\left(\alpha_2\eta_{21}^2+\alpha_3\left(\eta_{31}+\eta_{32}\right)^2\right)f_0^2f_0''\tau^3+ \\ & \alpha_3\eta_{32}\eta_{21}f_0\left(f_0'\right)^2\tau^3+\;O(\tau^4) \end{aligned} \tag{2.24}$$

As we did in Eqs. (1.28) and (1.29), we have to equate this expression term for term with the corresponding expressions in the Taylor series expansion

$$x_1=x_0+v_0\tau+\tfrac{1}{2}a_0\tau^2+\tfrac{1}{6}j_0\tau^3+O(\tau^4) \tag{2.25}$$

Using Eqs. (1.12), we can now write this as

$$x_1=x_0+f_0\tau+\tfrac{1}{2}f_0f_0'\tau^2+\tfrac{1}{6}\left\{f_0^2f_0''+f_0(f_0')^2\right\}\tau^3+O(\tau^4) \tag{2.26}$$

Equating the coefficients for the various terms in Eqs. (2.24) and (2.26), we find for the first order in τ the relation

$$\boxed{\alpha_1+\alpha_2+\alpha_3=1} \tag{2.27}$$

For the second order terms in τ we find

$$\boxed{\alpha_2\eta_{21}+\alpha_3\left(\eta_{31}+\eta_{32}\right)=\tfrac{1}{2}} \tag{2.28}$$

For the third order terms in τ involving the second derivative of the force, we find

$$\boxed{\alpha_2\eta_{21}^2+\alpha_3\left(\eta_{31}+\eta_{32}\right)^2=\tfrac{1}{3}} \tag{2.29}$$

while for the third order terms in τ involving the square of the first derivative of the force, we find

$$\boxed{\alpha_3\eta_{32}\eta_{21} = \tfrac{1}{6}} \tag{2.30}$$

2.6 Two Examples

Bob: This is all nice and fine, but I'd like to see some concrete examples. Since we have four equations for six unknown variables, we expect to have a two-parameter freedom of choice. Let's use that freedom, and write down a few examples, to get a feeling for the type of algorithms we have at our hands.

Alice: A natural choice would be to require that the second force evaluation takes place in the middle of the time step ($\eta_{21} = \frac{1}{2}$) , while the third force evaluation takes place at the end of the step ($\eta_{31} + \eta_{32} = 1$). With these two extra conditions, barring unforeseen complications, we can expect to find a unique solution.

Let's check that. By substituting our two conditions into the four boxed equations we found above, we get:

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 &= 1 \\ \frac{1}{2}\alpha_2 + \alpha_3 &= \frac{1}{2} \\ \frac{1}{4}\alpha_2 + \alpha_3 &= \frac{1}{3} \\ \alpha_3\eta_{32} = \frac{1}{3} \end{cases} \tag{2.31}$$

The second and third equations above can be solved readily to find $\alpha_2 = 2/3$ and $\alpha_3 = 1/6$, after which the first equation yields $\alpha_1 = 1/6$. The last equation then gives $\eta_{32} = 2$ which implies $\eta_{31} = -1$. We thus arrive at the following third-order scheme:

$$\begin{cases} x_1 &= x_0 + \frac{1}{6}\left(k_1 + 4k_2 + k_3\right)\tau \\ \\ k_1 &= f(x_0) \\ k_2 &= f(x_0 + \frac{1}{2}k_1\tau) \\ k_3 &= f(x_0 - k_1\tau + 2k_2\tau) \end{cases} \tag{2.32}$$

Bob: Good! This is indeed one of the classical third-order Runge-Kutta algorithms.

Alice: Another natural choice is to spread the force calculations evenly over the interval, at times 0 , $\tau/3$, and $2\tau/3$, before starting the calculations for the new step at time τ.

Bob: Such a scheme obviously cannot be used for our current purposes. You need the third force calculation at the very end of the step, otherwise there is nothing to recycle.

Alice: That is true, but you asked for example algorithms, and I expect this to lead to another well-known scheme, so let us derive it here on the side. If nothing else, it can function as a check on our calculations. We require that $\eta_{21} = \frac{1}{3}$ and $\eta_{31} + \eta_{32} = \frac{2}{3}$. Plugging this into the four conditions we have found before leads to:

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 &= 1 \\ \frac{1}{3}\alpha_2 + \frac{2}{3}\alpha_3 &= \frac{1}{2} \\ \frac{1}{9}\alpha_2 + \frac{4}{9}\alpha_3 &= \frac{1}{3} \\ \alpha_3\eta_{32} = \frac{1}{2} \end{cases} \tag{2.33}$$

The second and third equations imply $\alpha_2 = 0$ and $\alpha_3 = 3/4$, and whith the first equation we find $\alpha_1 = 1/4$. The last equation yields $\eta_{32} = 2/3$ which then determines $\eta_{31} = 0$. We thus arrive at:

$$\begin{cases} x_1 &= x_0 + \frac{1}{4}\left(k_1 + 3k_3\right)\tau \\ k_1 &= f(x_0) \\ k_2 &= f(x_0 + \frac{1}{3}k_1\tau) \\ k_3 &= f(x_0 + \frac{2}{3}k_2\tau) \end{cases} \tag{2.34}$$

Bob: Right you are: an alternative classical third-order Runge-Kutta scheme. I agree, it is good to know that we can reproduce this.

2.7 Recycle Conditions

Alice: It is time to return to our original objective, to find a third-order scheme that uses three force calculations per time step, two of which are computed anew, while the third one is being recycled from its use in the previous step. With two free parameters, we seem to have a good chance to find such a scheme.

As in Eqs. (2.8) and (2.10), we have to calculate the difference between the position $\tilde{x}_1$ at which the last force calculation of the previous step took place and the actual position x_1 at the end of that step. In Eq. (2.10) we only needed to let the term linear in τ vanish, in order to obtain a consistent second-order scheme. In the present case, for a third-order scheme, we need to let both the linear and quadratic terms in τ vanish. Using Eq. (2.22), we have:

$$\begin{aligned}\tilde{x}_1 &= x_0 + \eta_{31}k_1\tau + \eta_{32}k_2\tau \\ &= x_0 + (\eta_{31} + \eta_{32})\, f_0\tau + \\ &\quad \eta_{32}\eta_{21}f_0f_0'\tau^2 + O(\tau^3)\end{aligned} \tag{2.35}$$

Comparing this with Eq. (2.24), we have to the same order in τ:

$$\begin{aligned}x_1 &= x_0 + (\alpha_1 + \alpha_2 + \alpha_3)f_0\tau + \\ &\quad (\alpha_2\eta_{21} + \alpha_3\,(\eta_{31} + \eta_{32}))\, f_0f_0'\tau^2 + O(\tau^3)\end{aligned} \tag{2.36}$$

Requiring the coefficients of τ and τ^2 to match in the last two equations gives us two extra conditions:

$$\boxed{\alpha_1 + \alpha_2 + \alpha_3 = \eta_{31} + \eta_{32}} \tag{2.37}$$

and

$$\boxed{\alpha_2\eta_{21} + \alpha_3\,(\eta_{31} + \eta_{32}) = \eta_{32}\eta_{21}} \tag{2.38}$$

Bob: You see, I guessed right! The six boxed equations here will allow us to determine the six variables $\{\alpha_1, \alpha_2, \alpha_3, \eta_{21}, \eta_{31}, \eta_{32}\}$.

Alice: Not so fast. Don't count your chickens before they are hatched!

Bob: I haven't heard that expression in a long time. Well, hatching shouldn't be too difficult.

Alice: Gathering all six equations, we get:

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 &= 1 \\ \alpha_2\eta_{21} + \alpha_3 &= \frac{1}{2} \\ \alpha_2\eta_{21}^2 + \alpha_3 &= \frac{1}{3} \\ \alpha_3 &= \frac{1}{3} \\ \eta_{31} + \eta_{32} &= 1 \\ \eta_{32}\eta_{21} &= \frac{1}{2} \end{cases} \tag{2.39}$$

where we have already simplified the expressions somewhat by substituting, for example, one earlier relation $\alpha_1 + \alpha_2 + \alpha_3 = 1$ and another one in simplified form as $\alpha_2\eta_{21} + \alpha_3 = \frac{1}{2}$ into the last two boxed equations.

Bob: So far, so good.

Alice: Or so it seems. Look, when we substitute the fourth relation into the second and third one, we obtain:

$$\begin{aligned} \alpha_2\eta_{21} &= \tfrac{1}{6} \\ \alpha_2\eta_{21}^2 &= 0 \end{aligned} \tag{2.40}$$

There is no way that we can satisfy these two equations simultaneously!

The last line implies that either $\alpha_2 = 0$ or $\eta_{21} = 0$. Either case would imply $\alpha_2\eta_{21} = 0$, in contradiction with the requirement that $\alpha_2\eta_{21} = 1/6$.

Bob: I guess hatching was unsuccessful. That's a disappointment!

Alice: We have to conclude, somewhat surprisingly, that there just is no third-order recycling scheme. Whether we use two new force calculations per time step, or whether we recycle an additional force calculation from the previous time step, in both cases we wind up with a second-order algorithm.

Bob: That's a pity.

2.8 Remaining Freedom

Alice: However, not all is lost: our scheme is still second-order, and has more freedom than our non-recycling scheme. Specifically, let us gather the set of conditions necessary to guarantee at least second-order behavior for our recycling method. These are, from Eqs. ((2.27), ((2.28), and ((2.37):

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 = 1 \\ \alpha_2\eta_{21} + \alpha_3\left(\eta_{31} + \eta_{32}\right) = \frac{1}{2} \\ \alpha_1 + \alpha_2 + \alpha_3 = \eta_{31} + \eta_{32} \end{cases} \tag{2.41}$$

which simplifies to

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 = \eta_{31} + \eta_{32} = 1 \\ \alpha_2\eta_{21} + \alpha_3 = \frac{1}{2} \end{cases} \tag{2.42}$$

These are three equations for six unknown variables. If we introduce $\alpha \equiv \alpha_2$, $\eta \equiv \eta_{21}$, and $\zeta \equiv \eta_{32}$, we get the following parametrized solutions:

$$\left\{\begin{array}{lcl} \alpha_1 &=& \frac{1}{2} - \alpha + \alpha\eta \\ \alpha_2 &=& \alpha \\ \alpha_3 &=& \frac{1}{2} - \alpha\eta \\ \eta_{21} &=& \eta \\ \eta_{31} &=& 1 - \zeta \\ \eta_{32} &=& \zeta \end{array}\right. \tag{2.43}$$

This gives:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + \frac{1}{2}\Big((1-2\alpha+2\alpha\eta)k_1 + 2\alpha k_2 + (1-2\alpha\eta)k_3\Big)\tau \\ \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \eta k_1 \tau) \\ k_3 &=& f(x_0 + (1-\zeta)k_1\tau + \zeta k_2 \tau) \end{array}\right. \tag{2.44}$$

2.9 Summary

Bob: I'm not sure whether we've gained anything, by getting extra free parameters. I had hoped for a third-order scheme.

Alice: We haven't gained anything, but neither have we lost anything. Later, when we will apply these various algorithms, we can check to see whether any of the new parameters allow choices that give us more accurate results.

We can compare Eq. (2.44) with the non-recycling schemes, where we also perform two force calculations per step, and for which we obtained a second-order scheme as well. We found there, as Eq.(1.36):

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + ((1-\alpha)k_1 + \alpha k_2)\,\tau \\ \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \frac{1}{2\alpha}k_1\tau) \end{array}\right. \tag{2.45}$$

Bob: Ah yes, that is interesting. Let us see whether we can obtain Eq. (2.45) from Eq. (2.44). In that case, we'd better not use the third force calculation k_3 in the calculation of the new position. This means:

$$(1 - 2\alpha\eta) = 0 \tag{2.46}$$

or

$$2\alpha\eta = 1 \tag{2.47}$$

Plugging this back into the first line of Eq. (2.44), we get for the new position:

$$x_1 = x_0 + ((1-\alpha)k_1 + \alpha k_2)\,\tau \tag{2.48}$$

just as in Eq. (2.45. Since $\eta = 1/(2\alpha)$, the expression for the second force becomes:

$$k_2 = f(x_0 + \frac{1}{2\alpha}k_1\tau) \tag{2.49}$$

We conclude that, for the choice $\eta = 1/(2\alpha)$, Eq. (2.44) becomes Eq. (2.45). It all hangs together! The third force calculation in Eq. (2.44) effectively drops out, for this choice of parameters.

Alice: It is also instructive to compare this scheme with the second-order scheme we found based on one new force calculation and one recycled force calculation:

$$\begin{cases} x_1 &= x_0 + \frac{1}{2}\,(k_1 + k_2)\,\tau \\ \\ k_1 &= f(x_0) \\ k_2 &= f(x_0 + k_1\tau) \end{cases} \tag{2.50}$$

Bob: which in fact is exactly the previous set, Eq. (2.45), with the further restriction that $\alpha = \frac{1}{2}$.

Alice: Let us sum up. We conclude that we have found three different ways of constructing a second-order Runge-Kutta method:

1. *Without* recycling, we have Eq.(2.45), with *two* new force calculations per time step, and *one* free parameter;
2. *With* recycling, we have Eq.(2.50), with *one* new force calculation per time step, and *no* free parameters;
3. *With* recycling, we have Eq.(2.44), with *two* new force calculations per time step, and *three* free parameters.

Bob: Well done! Now it's time to leave this first-order differential equation behind us. I think we've learned enough, and I would prefer to go to the more realistic case of a second-order differential equation.

Chapter 3

Second-Order Differential Equations

3.1 Formulating the Problem

Alice: Now that we know how to solve a first-order differential equation, we can extend our methods immediately to treat the case of a general second-order differential equation

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}\right) \tag{3.1}$$

Here the force per unit mass $f(x)$ exerted on a particle depends explicitly on both the position and the velocity of that particle. An example of such a force is the motion of a mass point under the influence of friction. Indeed, our physical interpretation of the case of a first-order differential equation followed from the above form in the limit of infinitely strong friction, where the velocity-dependent term dominated completely. Another example would be the force that is felt by an electron moving in an electromagnetic field (where x would have be interpreted as a three-dimensional vector, a point we will come back to later in this chapter).

Bob: Wouldn't it be simpler to restrict ourselves immediately to the type of equations we are dealing with in stellar dynamics, without any velocity dependence in the forces?

Alice: I would prefer to hold off just a bit, because the simplest way to treat second-order equations is by following the same recipe as we did before. In that case, velocity dependence does not pose any problems.

The second-order equation above can be rewritten as a system of two first-order

differential equations:

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= f(x,v) \end{cases} \tag{3.2}$$

In principle we can look at this as a single first-order differential equation for a two-component vector.

Bob: Ah, that is a nice short-cut. Let's do that, and then we should be able to use all the results from the previous chapters immediately!

3.2 Vector Notation

Alice: We'll think have to put in some thought, since not all scalar equations generalize in an obvious way to the vectorial case, but I agree that it would probably be a good guide line.

Okay, to introduce vector notation, let us define:

$$\vec{s} \equiv \begin{pmatrix} x \\ v \end{pmatrix} \tag{3.3}$$

and

$$\vec{g} \equiv \begin{pmatrix} v \\ f(x,v) \end{pmatrix} \tag{3.4}$$

We can then write our second-order differential equation as a single first-order equation in terms of vectors:

$$\frac{d}{dt}\vec{s} = \vec{g} \tag{3.5}$$

or simply:

$$\dot{\vec{s}} = \vec{g} \tag{3.6}$$

When written out, this leads to the two equations implicit in:

$$\begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v \\ f(x,v) \end{pmatrix} \tag{3.7}$$

Bob: In our case of interest, classical mechanics, we will drop the velocity dependence, and study instead the simpler equation:

$$\frac{d^2x}{dt^2} = f(x) \tag{3.8}$$

or, equivalently

$$\begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v \\ f(x) \end{pmatrix} \tag{3.9}$$

Alice: As we will see in the next chapter, we can exploit the simple form of these two equations to get an extra order of accuracy, seemingly for free, using what is called a partitioned Runge-Kutta algorithm. However, in order to put that clever trick in a clear context, in the current section we will be less clever. Rather, we will simply apply the same treatment that we have developed in the previous chapters.

Bob: Fine, but let's not linger too long! And please, let us drop the velocity dependence in the forces. Life is complicated enough as it is.

3.3 One Force Evaluation per Step

Alice: Okay, okay, we'll work with $f(x)$ then. Now, everything we have done so far carries over directly to the case of two coupled differential equations. To show this, let us repeat the same derivation, but now in vector form. At the start of a time step, we evaluate the right-hand side of the differential equation at $t = 0$:

$$\vec{k}_1 = \vec{g}_0 = \begin{pmatrix} v_0 \\ f_0 \end{pmatrix} \tag{3.10}$$

This leads to the following dimensionally correct expression:

$$\vec{s}_1 = \vec{s}_0 + \alpha_1 \vec{k}_1 \tau \tag{3.11}$$

Combining the last two equations, we have

$$\vec{s}_1 = \vec{s}_0 + \alpha_1 \vec{g}_0 \tau \tag{3.12}$$

We can compare this expression with the Taylor series:

$$\vec{s}_1 = \vec{s}_0 + \dot{\vec{s}}_0 \tau + \tfrac{1}{2} \ddot{\vec{s}}_0 \tau^2 + O(\tau^3) \tag{3.13}$$

where we use the shorthand notations

$$\dot{\vec{s}}_0 = \begin{pmatrix} v_0 \\ a_0 \end{pmatrix} \tag{3.14}$$

and

$$\ddot{\vec{s}}_0 = \begin{pmatrix} a_0 \\ j_0 \end{pmatrix} \tag{3.15}$$

to avoid introducing yet another set of new symbols for the various derivatives of $\vec{s}$ at time 0.

In order to compute

$$\ddot{\vec{s}}_0 = \frac{d}{dt}\vec{g}_0 \tag{3.16}$$

we will need to determine the time derivative of $\vec{g}$.

3.4 Not So Fast

Bob: presumably that is simply

$$\frac{d}{dt}\vec{g}_0 = \vec{g}_0\vec{g}'_0 \tag{3.17}$$

in analogy to what we did in the first chapter, where we used $\frac{d}{dt}f(x_0) = f_0 f'_0$.

Alice: Not so fast! You have not specified what you mean with that notation. The left-hand side is a vector, while the right-hand side suggests the product of two vectors. What does it mean?

Bob: Hmm. I hadn't thought about that. Good question.

Alice: If it would be an inner product, the left hand side should be a scalar. If, however, it is a tensor product, the left hand side should be a tensor. In neither case does it produce a vector.

Bob: Again, good point. Wel, in case of doubt, write it out! What does it look like in components?

Alice: Let us check. The most intuitive approach would be to start with a small variation in $\vec{g}$, which can be expressed as

$$d\vec{g} = d\begin{pmatrix} v \\ f(x) \end{pmatrix} = \begin{pmatrix} dv \\ f'(x)dx \end{pmatrix} \tag{3.18}$$

This implies:

$$\frac{d}{dt}\vec{g} = \begin{pmatrix} f \\ vf' \end{pmatrix} \tag{3.19}$$

Bob: Good! So we do get a vector, after all.

Alice: Yes. And even if you would have allowed me to retain a velocity dependence in the force, we would still have wound up with a vector, but in this case we would have:

$$d\vec{g} = d\begin{pmatrix} v \\ f(x) \end{pmatrix} = \begin{pmatrix} dv \\ f_x(x)dx + f_v(x)dv \end{pmatrix} \tag{3.20}$$

where now $f_x = f' = \partial f/\partial x$ and $f_v = \partial f/\partial v$. This would give us

$$\frac{d}{dt}\vec{g} = \begin{pmatrix} f \\ vf_x + ff_v \end{pmatrix} \tag{3.21}$$

Bob: Nice to know, but no thanks, let's stick with position-dependent forces only.

Alice: If you insist. At least the notation above will point the way for further generalizations, whenever we want to go that route.

Bob: Thanks!

3.5 Forward Euler in Vector Form

Alice: Let us introduce the symbol $\vec{h}_0$ for the right-hand side of Eq. (3.19), at time 0:

$$\vec{h}_0 \equiv \begin{pmatrix} f_0 \\ v_0 f_0' \end{pmatrix} \tag{3.22}$$

We can then write Eq. (3.13) as

$$\vec{s}_1 = \vec{s}_0 + \vec{g}_0\tau + \tfrac{1}{2}\vec{h}_0\tau^2 + O(\tau^3) \tag{3.23}$$

We demand that Eqs. (3.12) and Eq. (3.23) should be equal. The constant term $\vec{s}_0$ matches trivially, and the first condition arises from the term linear in τ:

$$\alpha_1\vec{g}_0 = \vec{g}_0 \tag{3.24}$$

hence

$$\boxed{\alpha_1 = 1} \tag{3.25}$$

We have no free parameter left, so this leads us to the only possible explicit first-order integration scheme

$$\begin{cases} \vec{s}_1 &= \vec{s}_0 + \vec{k}_1\tau \\ \\ \vec{k}_1 &= \vec{g}(\vec{s}_0) \end{cases} \tag{3.26}$$

which is the forward-Euler algorithm, now in vector form. If we write this out in components, we get:

$$\begin{pmatrix} x_1 \\ v_1 \end{pmatrix} = \begin{pmatrix} x_0 + \alpha_1 v_0 \tau \\ v_0 + \alpha_1 f_0 \tau \end{pmatrix} \tag{3.27}$$

Let us define

$$\vec{k}_1 \equiv \begin{pmatrix} v_0 \\ k_1 \end{pmatrix} \tag{3.28}$$

where we will use the same symbol k for the vector and the last componenent, again to avoid introducing yet more new letters. With this notation, we can write Eq. (3.24) in a more traditional form as:

$$\begin{cases} x_1 &= x_0 + v_0\tau \\ v_1 &= v_0 + k_1\tau \\ \\ k_1 &= f(x_0) \end{cases} \tag{3.29}$$

We thus recover the forward-Euler algorithm. As before, this scheme is only first-order accurate, since it cannot possibly reproduce the $O(\tau^2)$ term in Eq. (3.23).

Bob: Quite a bit of work to regain Forward Euler!

3.6 Two Force Evaluations per Step

Alice: Yes, and we might have guessed the result, but when we move on to using two force evaluations per step, things will undoubtedly get messier.

As before, after a first evaluation of the right-hand side of the differential equation, we can perform a preliminary integration in time, after which we can evaluate the right-hand side again, at a new position:

$$\vec{k}_1 = \vec{g}(\vec{s}_0) \tag{3.30}$$
$$\vec{k}_2 = \vec{g}(\vec{s}_0 + \eta\vec{k}_1\tau) \tag{3.31}$$

We can now use a more general expression for the new position, in which we rely on the preliminary information that has been gathered in the two force evoluations. Since each force evaluation has the dimension of a time derivative of the position, we have to multiply each one with a single power of τ. The coefficient for each term is, as yet, arbitrary, so let us parametrize them as follows:

$$\vec{s}_1 = \vec{s}_0 + \left(\alpha_1\vec{k}_1 + \alpha_2\vec{k}_2\right)\tau \tag{3.32}$$

We can combine these equations, and write them as

$$\vec{s}_1 = \vec{s}_0 + \left\{\alpha_1\vec{g}_0 + \alpha_2\vec{g}(\vec{s}_0 + \eta\vec{g}_0\tau)\right\}\tau \tag{3.33}$$

We want to determine

$$\vec{k}_2 = \vec{g}(\vec{s}_0 + d\vec{s}) \tag{3.34}$$

where

$$d\vec{s} = \eta\vec{g}_0\tau \tag{3.35}$$

We have already seen that

$$d\vec{g} = d\begin{pmatrix} v \\ f(x) \end{pmatrix} = \begin{pmatrix} dv \\ f'(x)dx \end{pmatrix} \tag{3.36}$$

and we can also write

$$d\vec{s} = \begin{pmatrix} dx \\ dv \end{pmatrix} = \eta\begin{pmatrix} v_0 \\ f_0 \end{pmatrix}\tau \tag{3.37}$$

The two results that are encoded here, $dx = \eta v_0\tau$ and $dv = \eta f_0\tau$, can now be plugged back into the definition of $d\vec{g}$

$$d\vec{g} = \begin{pmatrix} dv \\ f'(x)dx \end{pmatrix} = \eta\begin{pmatrix} f_0 \\ v_0 f_0' \end{pmatrix}\tau \tag{3.38}$$

We finally get

$$\vec{k}_2 = \vec{g}(\vec{s}_0 + d\vec{s}) = \vec{g}(\vec{s}_0) + d\vec{g} = \vec{k}_1 + \eta\vec{h}_0\tau \tag{3.39}$$

where we use again the notation

$$\vec{h}_0 \equiv \begin{pmatrix} f_0 \\ v_0 f_0' \end{pmatrix} \tag{3.40}$$

Bob: A useful function, obviously.

3.7 Putting Everything Together

Alice: To sum up: developing $\vec{k}_2$ in a Taylor series around $\tau = 0$ gives:

$$\vec{k}_2 = \vec{g}(\vec{s}_0 + \eta\vec{g}_0\tau) = \vec{g}_0 + \eta\vec{h}_0\tau + O(\tau^2) \tag{3.41}$$

Using this result in Eq. (3.33), we find for the new position, at the end of our time step:

$$\vec{s}_1 = \vec{s}_0 + (\alpha_1 + \alpha_2)\vec{g}_0\tau + \alpha_2\eta\vec{h}_0\tau^2 + O(\tau^3) \tag{3.42}$$

We can now compare this expression with the Taylor series expansion of the true orbit, as we did in the previous section:

$$\vec{s}_1 = \vec{s}_0 + \dot{\vec{s}}_0\tau + \tfrac{1}{2}\ddot{\vec{s}}_0\tau^2 + O(\tau^3) \tag{3.43}$$

We can write this, as we saw in Eq. (3.23), in terms of $\vec{h}_0$ as follows:

$$\vec{s}_1 = \vec{s}_0 + \vec{g}_0\tau + \tfrac{1}{2}\vec{h}_0\tau^2 + O(\tau^3) \tag{3.44}$$

Starting with terms to first order in τ in Eqs. (3.42) and (3.44), we have to insist that

$$(\alpha_1 + \alpha_2)\vec{g}_0 = \vec{g}_0 \tag{3.45}$$

which leads to the condition

$$\boxed{\alpha_1 + \alpha_2 = 1} \tag{3.46}$$

To second order in τ , we would like to satisfy:

$$\alpha_2\eta\vec{h}_0 = \tfrac{1}{2}\vec{h}_0 \tag{3.47}$$

which can be done through the condition

$$\boxed{\eta = \frac{1}{2\alpha_2}} \tag{3.48}$$

We conclude that when we allow two force evaluations, we again are left with one free parameter α.

Bob: So, now we're done, and we can move on!

3.8 Summary

Alice: Yes, but let us put all our results on the table first.

To summarize, we can write:

$$\begin{cases} \vec{s}_1 &= \vec{s}_0 + \left((1-\alpha)\vec{k}_1 + \alpha\vec{k}_2\right)\tau \\ \\ \vec{k}_1 &= \vec{g}(\vec{s}_0) \\ \vec{k}_2 &= \vec{g}(\vec{s}_0 + \frac{1}{2\alpha}\vec{k}_1\tau) \end{cases} \tag{3.49}$$

Notice that this is exactly the vector generalization of the expressions we found in the case of a first-order differential equation:

$$\begin{cases} x_1 &= x_0 + ((1-\alpha)k_1 + \alpha k_2)\,\tau \\ \\ k_1 &= f(x_0) \\ k_2 &= f(x_0 + \frac{1}{2\alpha}k_1\tau) \end{cases} \tag{3.50}$$

Bob: I'd like to see our vector expressions written out in components. That way it will be easier to make contact with the previous chapters.

Alice: My pleasure! Starting with

$$\vec{k}_1 = \vec{g}(\vec{s}_0) = \vec{g}\left(\begin{pmatrix} x_0 \\ v_0 \end{pmatrix}\right) = \begin{pmatrix} v_0 \\ f_0 \end{pmatrix} \tag{3.51}$$

we find

$$\vec{k}_2 = \vec{g}\left(\begin{pmatrix} x_0 \\ v_0 \end{pmatrix} + \frac{1}{2\alpha}\begin{pmatrix} v_0 \\ f_0 \end{pmatrix}\tau\right) \tag{3.52}$$

$$= \vec{g}\left(\begin{pmatrix} x_0 + \frac{1}{2\alpha}v_0\tau \\ v_0 + \frac{1}{2\alpha}f_0\tau \end{pmatrix}\right) = \begin{pmatrix} v_0 + \frac{1}{2\alpha}f_0\tau \\ f(x_0 + \frac{1}{2\alpha}v_0\tau) \end{pmatrix} \tag{3.53}$$

which leads to the expression for $\vec{s}_1$:

$$\begin{pmatrix} x_1 \\ v_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ v_0 \end{pmatrix} + \begin{pmatrix} (1-\alpha)v_0\tau + \alpha(v_0 + \frac{1}{2\alpha}f_0\tau)\tau \\ (1-\alpha)f_0\tau + \alpha f(v_0 + \frac{1}{2\alpha}f_0\tau)\tau \end{pmatrix} \tag{3.54}$$

$$= \begin{pmatrix} x_0 + v_0\tau + \frac{1}{2}f_0\tau^2 \\ v_0 + (1-\alpha)f_0\tau + \alpha f(x_0 + \frac{1}{2\alpha}v_0\tau)\tau \end{pmatrix} \tag{3.55}$$

We can write this in components as

$$\begin{cases} x_1 = x_0 + v_0\tau + \frac{1}{2}f_0\tau^2 \\ v_1 = v_0 + ((1-\alpha)k_1 + \alpha k_2)\tau \\ \\ k_1 = f(x_0) \\ k_2 = f(x_0 + \frac{1}{2\alpha}v_0\tau) \end{cases} \tag{3.56}$$

3.9 Two Examples

Bob: I find this a lot more understandable that the vector notation. And for practical application, let's look at a couple special case. For $\alpha = 1$ we find

$$\begin{cases} x_1 = x_0 + v_0\tau + \frac{1}{2}k_1\tau^2 \\ v_1 = v_0 + k_2\tau \\ \\ k_1 = f(x_0) \\ k_2 = f(x_0 + \frac{1}{2}v_0\tau) \end{cases} \tag{3.57}$$

and for $\alpha = \frac{1}{2}$ we find

$$\begin{cases} x_1 = x_0 + v_0\tau + \frac{1}{2}k_1\tau^2 \\ v_1 = v_0 + \frac{1}{2}(k_1 + k_2)\tau \\ \\ k_1 = f(x_0) \\ k_2 = f(x_0 + v_0\tau) \end{cases} \tag{3.58}$$

Alice: Ah, that is interesting! This is almost our leapfrog scheme.

Bob: How so?

Alice: The expression for x_1 is exactly the same as what we have used when we implemented the leapfrog.

Bob: And so is the expression for v_1. The whole idea of leapfrogging is to advance the velocity with an acceleration that is the exact average of the force calculation at the beginning and at the end of the step.

Alice: Ah, the word *exact* is important here! While it is true that k_1 is the force evaluation at the beginning of the step, the *exact* force calculation at the end of the step would be $f(x_1) = x_0 + v_0\tau + \frac{1}{2}k_1\tau^2$. However, in our scheme above, $k_2 = x_0 + v_0\tau$, and the last term is missing.

Bob: Tricky! So we are dealing with an almost-leapfrog scheme, where the last force calculation is based on a predicted value for the new position, instead of the corrected value.

Alice: Yes, that's a good way of putting it. And all this can serve as an invitation to go beyond the straightforward generalizations of the Runge-Kutta schemes for first-order differential equations. It is time to look at more imaginative schemes, that treat position and velocity in different ways!

Chapter 4

Partitioned Runge-Kutta Algorithms

Bob: You promised a better approach to solving second-order differential equations, using Runge-Kutta schemes. What did you call such an algorithm again?

Alice: It is called a partitioned Runge-Kutta algorithm. The idea is to combine the force calculations in different ways for the position and for the velocity. The word 'partitioned' here means that separate the treatment of x from the treatment of v. We already saw an example at the end of our previous discussion, where we had found a scheme that was almost, but not quite, a leapfrog scheme. If we would have tinkered with that scheme, we could have turned it into a leapfrog, but it would then no longer be a vector generalization of a Runge-Kutta scheme.

Bob: So you're saying that we have a lot more freedom, when we allow separate ways to update position and velocity, after first calculation a number of force evaluations.

Alice: Exactly. And we have already done this, for our fourth-order integrator, the one we plucked from Abramowitz and Stegun.

Bob: Does this mean that we can write our good old leapfrog as a partitioned Runge-Kutta scheme? That would be interesting! I have always thought about Runge-Kutta methods and the leapfrog scheme as two completely different animals, pardon the pun. Do you think that the leapfrog can be view as a type of Runge-Kutta algorithm?

Alice: I'm not sure. One reason to do this systematic landscape exploration is to find the answers to that type of question! And I'm sure we'll find out soon. By exploring all possible schemes with up to two new force calculations per step, we're bound to encounter the leapfrog, if indeed it is a citizen of the Runge-Kutta world.

So let us return to our special second-order differential equation

$$\frac{d^2x}{dt^2} = f(x). \tag{4.1}$$

Let us first gather some useful expressions, starting with the two first-order equations

$$\begin{cases} \dfrac{dx}{dt} &= v \\ \dfrac{dv}{dt} &= f(x) \end{cases} \tag{4.2}$$

As before, we expand the position and the velocity of the orbit in Taylor series:

$$x_1 = x_0 + v_0\tau + \tfrac{1}{2}a_0\tau^2 + \tfrac{1}{6}j_0\tau^3 + \tfrac{1}{24}s_0\tau^4 + \tfrac{1}{120}c_0\tau^5 + \tfrac{1}{720}p_0\tau^6 + O(\tau^7) \tag{4.3}$$

$$v_1 = v_0 + a_0\tau + \tfrac{1}{2}j_0\tau^2 + \tfrac{1}{6}s_0\tau^3 + \tfrac{1}{24}c_0\tau^4 + \tfrac{1}{120}p_0\tau^5 + O(\tau^6) \tag{4.4}$$

and when we differentiate the set of differential equations several times, we obtain the following equations:

$$\begin{aligned} a_0 &= f(x(0)) = f(x_0) = f_0 \\ j_0 &= \left.\frac{d}{dt}a(t)\right|_{t=0} = \left.\frac{d}{dt}f(x(t))\right|_{t=0} = \left.\frac{df(x)}{dx}\right|_{x=x_0}\left.\frac{dx}{dt}\right|_{t=0} = f_0'v_0 \\ s_0 &= \dot{j}_0 = f_0''v_0^2 + f_0'f_0 \\ c_0 &= \dot{s}_0 = f_0'''v_0^3 + 3f_0''f_0v_0 + (f_0')^2v_0 \\ p_0 &= \dot{c}_0 = f_0''''v_0^4 + 6f_0'''f_0v_0^2 + f_0''\left\{3f_0^2 + 5f_0'v_0^2\right\} + (f_0')^2f_0 \end{aligned} \tag{4.5}$$

The last three lines can be derived in the same way as the second line, by fully writing out the differentiations, using the chain rule. This derivation is completely analogous to what we did for our first-order differential equation.

4.1 One Force Evaluation per Step

Bob: I guess we will forget about force recycling, at least for now.

Alice: Yes. To keep things simple, let us look at a single integration step. But we have another choice to make.

In the case of a first-order differential equation, at the start of our integration we can only evaluate the right-hand side at time zero, at the beginning of the integration time step. If we simply follow that example, we start with:

$$k_1 = f(x_0) \tag{4.6}$$

This leads to the following dimensionally correct expressions:

$$\begin{aligned} x_1 &= x_0 + \alpha_0 v_0 \tau + \alpha_1 k_1 \tau^2 && (4.7)\\ v_1 &= v_0 + \beta_1 k_1 \tau && (4.8) \end{aligned}$$

We can write the expression for the position, substituting k_1, as

$$x_1 = x_0 + \alpha_0 v_0 \tau + \alpha_1 f_0 \tau^2 \tag{4.9}$$

We have to compare this with the Taylor series

$$x_1 = x_0 + v_0 \tau + \tfrac{1}{2} a_0 \tau^2 + \tfrac{1}{6} j_0 \tau^3 + O(\tau^4) \tag{4.10}$$

Using Eqs. (XXXXX) we can write this as

$$x_1 = x_0 + v_0 \tau + \tfrac{1}{2} f_0 \tau^2 + \tfrac{1}{6} v_0 f_0' \tau^3 + O(\tau^4) \tag{4.11}$$

Comparing Eqs. (4.9) and (4.11), we find:

$$\boxed{\alpha_0 = 1} \tag{4.12}$$

and

$$\boxed{\alpha_1 = \tfrac{1}{2}} \tag{4.13}$$

We cannot satisfy the third order term in τ, so as far as the expression for the position is concerned, we can make our scheme second-order accurate.

Looking now at the velocity, we have

$$v_1 = v_0 + \beta_1 f_0 \tau \tag{4.14}$$

We have to compare this with the Taylor series

$$v_1 = v_0 + a_0\tau + \tfrac{1}{2}j_0\tau^2 + \tfrac{1}{6}s_0\tau^3 + O(\tau^4) \tag{4.15}$$

Using Eqs. (XXXXX) we can write this as

$$v_1 = v_0 + f_0\tau + \tfrac{1}{2}v_0 f_0'\tau^2 + O(\tau^4) \tag{4.16}$$

Comparing Eqs. (4.14) and (4.16), we find from the terms that are first order in τ:

$$\boxed{\beta_1 = 1} \tag{4.17}$$

Going to second order in τ would require that

$$j_0 = f_0' v_0 = 0 \tag{4.18}$$

which cannot be true for general f and v_0. Even though we can construct a second-order algorithm for the position, we can only find a first-order algorithm for the velocity.

Bob: And I presume that there is no point in using a higher-order algorithm for the position than for the velocity, since the overall order of the integration scheme must be the lowest order of that of the components. Hmmm. Is that so?

Alice: Yes, that is correct. For the very first step, it is possible in this case to find a new position that is second-order accurate. But as soon as we take the second step, we use the velocity that we arrived at in the first step, which is only first-order accurate. The same is true for each subsequent step: we always use the velocity value from the previous step.

Bob: But each time we multiply the velocity with τ. So even though the velocity is first-order, the product of the velocity with τ must be second-order correct, leading to an error term that is third-order in τ.

Alice: Yes, that is formally correct. However, \dots

We have to conclude that our approach only leads to a first-order correct algorithm, which is of course the forward Euler algorithm:

$$\begin{cases} x_1 &= x_0 + v_0\tau \\ v_1 &= v_0 + k_1\tau \\ \\ k_1 &= f(x_0) \end{cases} \tag{4.19}$$

%\subsubsubsection{A Delayed Force Evaluation}

{\bf 4.1.1.2. A Delayed Force Evaluation}

Given the special form of our second-order differential equation, it is not necessary to start with a force evaluation at time zero. The first equation in the set

$$\begin{cases} \dfrac{dx}{dt} &= v \\ \dfrac{dv}{dt} &= f(x) \end{cases} \tag{4.20}$$

allows us to make a first-order prediction of the position, as:

$$x(t) = x_0 + v_0 t + O(t^2) \tag{4.21}$$

which in turn allows us to postpone the first force evaluation to this non-zero time:

$$\dot{v}(t) = f(t) = f(x_0 + v_0 t + O(t^2)) = f(x_0 + v_0 t) + O(t^2). \tag{4.22}$$

Note that this trick is not possible for a general force term that would depend on velocity as well. In that case, the last equation would read

$$\dot{v}(t) = f(x(t), v(t)) = f(x_0 + v_0 t, v_0 + a_0 t) = f(x_0 + v_0 t, v_0 + f_0 t) \tag{4.23}$$

which would mean that we need an initial force evaluation f_0 at time zero, before we can perform a subsequent force evaluation at time t.

4.2 xxx

Let us exploit this extra freedom, for our special differential equation, by repeating our previous analysis for a delayed force evaluation. Our first force evaluation can now take place at time $t = \eta_1 \tau$ where η_1 is a free parameter. Using the linear extrapolation of the position, as sketched above, we obtain:

$$k_1 = f(x_0 + \eta_1 v_0 \tau) \tag{4.24}$$

This leads to the following dimensionally correct expressions:

$$x_1 = x_0 + v_0 \tau + \alpha_1 k_1 \tau^2 \tag{4.25}$$

$$v_1 = v_0 + \beta_1 k_1 \tau \tag{4.26}$$

If we expand k_1 to first order in τ , we obtain:

$$k_1 = f_0 + \eta_1 f_0' v_0 \tau \tag{4.27}$$

Eq. (4.25) can thus be written as:

$$\begin{cases} x_1 &= x_0 + v_0\tau + \alpha_1 f_0 \tau^2 + \alpha_1\eta_1 f_0' v_0 \tau^3 + O(\tau^4) \\ v_1 &= v_0 + \beta_1 f_0 \tau + \beta_1\eta_1 f_0' v_0 \tau^2 + O(\tau^3) \end{cases} \tag{4.28}$$

Comparing this with

$$x_1 = x_0 + v_0\tau + \tfrac{1}{2}a_0\tau^2 + \tfrac{1}{6}j_0\tau^3 + O(\tau^4) \tag{4.29}$$

EXPAND

let us first consider the $O(\tau^2)$ terms, which leads to the requirement:

$$a_0 = f_0 = 2\alpha_1 f_0 \tag{4.30}$$

which leads to

$$\alpha_1 = \tfrac{1}{2} \tag{4.31}$$

Similarly, we can use the Taylor series for v:

$$v_1 = v_0 + a_0\tau + \tfrac{1}{2}j_0\tau^2 + \tfrac{1}{6}s_0\tau^3 + O(\tau^4) \tag{4.32}$$

EXPAND

Considering the $O(\tau)$ term, we have:

$$a_0 = f_0 = \beta_1 f_0 \tag{4.33}$$

which leads to

$$\beta_1 = 1 \tag{4.34}$$

Considering the $O(\tau^2)$ terms, we have:

$$j_0 = f_0' v_0 = 2\beta_1\eta_1 f_0' v_0 \tag{4.35}$$

This implies:

$$\eta_1 = \tfrac{1}{2} \tag{4.36}$$

In this way, all three free parameters are fixed, by requiring the algorithm to be second-order in $O(\tau)$ in both position and velocity.

Could it be that we are in luck, and that this fixed solution can give us expressions for x and v that are also third-order correct in $O(\tau)$? Let us start with the position equation. This would require:

$$j_0 = f_0' v_0 = 6\alpha_1\eta_1 f_0' v_0 \tag{4.37}$$

Alas, since we are forced to use $\alpha_1 = \eta_1 = \frac{1}{2}$, the coefficient on the right-hand side is 3/2 while the one on the left-hand side is 1. We have no freedom left, so this equation has no solutions for a general function f and a general initial velocity v_0.

%\subsubsubsection{Verlet-St\"ormer-Delambre Scheme}

{\bf 4.1.1.3. Verlet-St\"ormer-Delambre Scheme}

Using one evaluation of the right-hand side of the differential equation, we have thus arrived at the following second-order integration scheme:

$$\begin{cases} x_1 &= x_0 + v_0\tau + \frac{1}{2}k_1\tau^2 \\ v_1 &= v_0 + k_1\tau \\ \\ k_1 &= f(x_0 + \frac{1}{2}v_0\tau) \end{cases} \tag{4.38}$$

Even though the equation for the velocity looks first-order, it is actually second-order accurate, through the clever choice of time at which the right-hand side of the differential equation is evaluated, namely in between the times at which v_0 and v_1 are determined.

In fact, this scheme is nothing else than the good old leapfrog algorithm, also known as the Verlet-St\"ormer-Delambre scheme, as we will show now.

Define

$$x_{1/2} \equiv x_0 + \tfrac{1}{2}v_0\tau \tag{4.39}$$

and

$$v_{1/2} \equiv v_0 + \tfrac{1}{2}f_{1/2}\tau \tag{4.40}$$

where

$$f_{1/2} \equiv f(x_{1/2}) = f(x_0 + \tfrac{1}{2}v_0\tau) \tag{4.41}$$

Our new scheme can then be written as

$$\left\{ \begin{array}{lcl} x_1 & = & x_0 + v_{1/2}\tau \\ v_1 & = & v_0 + f_{1/2}\tau \end{array} \right. \tag{4.42}$$

Alternatively, we can express the first of these equations in terms of

$$\begin{array}{rcl} x_{3/2} & = & x_1 + \frac{1}{2}v_1\tau \\ & = & \left(x_0 + v_0\tau + \frac{1}{2}f_{1/2}\tau^2\right) + \frac{1}{2}\left(v_0 + f_{1/2}\tau\right)\tau \\ & = & x_{1/2} + \left(v_0 + f_{1/2}\tau\right)\tau \\ & = & x_{1/2} + v_1\tau \end{array} \tag{4.43}$$

We have thus derived at expressions that show the leapfrog nature of the algorithm most clearly:

$$\left\{ \begin{array}{lcl} x_{3/2} & = & x_{1/2} + v_1\tau \\ v_1 & = & v_0 + f_{1/2}\tau \end{array} \right. \tag{4.44}$$

This representation shows clearly that the equations are fully time symmetric. This fact was rather hidden in the original formulation, eqs. (4.38). However, if we explicitly take a step forward and then another step backward, using eqs. (4.38), we can recover the time symmetry inherent in these equations, as follows. Let us denote the resulting position and velocity by $\{x_{-0}, v_{-0}\}$, which are obtained from $\{x_1, v_1\}$ by taking a time step with size $-\tau$. Our task is to show that $\{x_{-0}, v_{-0}\}$ actually coincide with $\{x_0, v_0\}$, not only to second order, as would be guaranteed in any second order scheme, but in fact to all orders in τ.

$$\begin{array}{rcl} x_{-0} & = & x_1 - v_1\tau + \frac{1}{2}f(x_1 - \frac{1}{2}v_1\tau)\tau^2 \\ & = & \left\{x_0 + v_0\tau + \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2\right\} - \left\{v_0 + f(x_0 + \frac{1}{2}v_0\tau)\tau\right\}\tau + \\ & & \frac{1}{2}f(\{x_0 + v_0\tau + \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2\} - \frac{1}{2}\{v_0 + f(x_0 + \frac{1}{2}v_0\tau)\tau\}\tau)\tau^2 \\ & = & x_0 - \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2 + \\ & & \frac{1}{2}f(x_0 + v_0\tau + \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2 - \frac{1}{2}v_0\tau - \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2)\tau^2 \\ & = & x_0 - \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2 + \\ & & \frac{1}{2}f(x_0 + \frac{1}{2}v_0\tau)\tau^2 \\ & = & x_0 \end{array} \tag{4.45}$$

$$\begin{array}{rcl} v_{-0} & = & v_1 - f(x_1 - \frac{1}{2}v_1\tau)\tau \\ & = & \left\{v_0 + f(x_0 + \frac{1}{2}v_0\tau)\tau\right\} \end{array}$$

$$
\begin{aligned}
& -f(\{x_0 + v_0\tau + \tfrac{1}{2}f(x_0 + \tfrac{1}{2}v_0\tau)\tau^2\} - \tfrac{1}{2}\{v_0 + f(x_0 + \tfrac{1}{2}v_0\tau)\tau\}\tau \\
= \quad & v_0
\end{aligned} \tag{4.46}
$$

Finally, we can also write

$$
\begin{aligned}
x_{3/2} &= x_{1/2} + v_1\tau \\
&= x_{1/2} + v_0\tau + f_{1/2}\tau^2 \\
&= x_{1/2} + v_{1/2}\tau + \tfrac{1}{2}f_{1/2}\tau^2
\end{aligned} \tag{4.47}
$$

and

$$
\begin{aligned}
v_{3/2} &= v_1 + \tfrac{1}{2}f_{3/2}\tau \\
&= v_0 + f_{1/2}\tau + \tfrac{1}{2}f_{3/2}\tau \\
&= v_{1/2} + \tfrac{1}{2}\left(f_{1/2} + f_{3/2}\right)\tau
\end{aligned} \tag{4.48}
$$

We thus find

$$
\left\{
\begin{array}{lcl}
x_{3/2} &=& x_{1/2} + v_{1/2}\tau + \frac{1}{2}f_{1/2}\tau^2 \\
v_{3/2} &=& v_{1/2} + \frac{1}{2}\left(f_{1/2} + f_{3/2}\right)\tau
\end{array}
\right. \tag{4.49}
$$

We can now shift our zero point in time by half a time step, to arrive at the more convenient notation:

$$
\left\{
\begin{array}{lcl}
x_1 &=& x_0 + v_0\tau + \frac{1}{2}f_0\tau^2 \\
v_1 &=& v_0 + \frac{1}{2}(f_0 + f_1)\tau
\end{array}
\right. \tag{4.50}
$$

Comparing this with our starting point, eqs. (4.38):

$$
\left\{
\begin{array}{lcl}
x_1 &=& x_0 + v_0\tau + \frac{1}{2}k_1\tau^2 \\
v_1 &=& v_0 + k_1\tau \\
\\
k_1 &=& f(x_0 + \frac{1}{2}v_0\tau)
\end{array}
\right. \tag{4.51}
$$

we have arrived at the alternative formulation:

$$
\left\{
\begin{array}{lcl}
x_1 &=& x_0 + v_0\tau + \frac{1}{2}k_1\tau^2 \\
v_1 &=& v_0 + (k_1 + k_2)\tau \\
\\
k_1 &=& f(x_0) \\
k_2 &=& f(x_0 + v_0 + \frac{1}{2}k_1\tau^2)
\end{array}
\right. \tag{4.52}
$$

The second formulation seems rather different, in that it requires two force calculations. Note, however, that the position at which the second force calculation takes place is *exactly* the same position at which the first force calculation for the next step will take place. Therefore, the second force calculation of each step can be recycled as the first force calculation of the next step. Effectively, we thus use only one force calculation per step. This trick is known in the literature as FSAL, short for First-Same-As-Last. We will come back to this point below.

%\subsubsubsection{An Historical Note}

{\bf 4.1.1.4. An Historical Note}

Almost everywhere in the literature, Runge-Kutta methods are assumed to start with $k_1 = f_0$: letting the first evaluation of the right-hand side of the differential equation take place at the very beginning of the step. This is necessary in the general case, but not for the special case of a second-order differential equation where there is no velocity dependence in the force term. The only place we have found so far in the literature, which mentions the possibility of starting with the force evaluation already at a later time is a paragraph in Nyström (1925), the original paper introducing what is now known as the Runge-Kutta-Nyström algorithms.

In his section 2, p. 7, near the bottom, he remarks that, to be consistent, we should allow the freedom to write a general expression of the type we have done above in Eq. (4.24). He then adds that he decided against considering this extra freedom, for two reasons, both pragmatic, the first related to speed of execution of the algorithms, the second related to speed of derivation of the expressions fot the algorithms. Here are his arguments.

First of all, we often know already the force evaluation at the beginning of the step, from the last stage of the calculation of the previous step (at least approximately; and using even earlier force calculations, we can further improve the accuracy, without having to perform new force evaluations). Secondly, he adds, starting from such a general expression has led him to such unwieldy expressions that he was more or less forced to put $\eta_1 = 0$ in his equivalent to our Eq. (4.24).

Of course, current availability of algebraic manipulation programs have now invalidated his second argument. Curiously, all text books seem to propagate the simplifying assumption $\eta_1 = 0$ without questioning what the basis for this assumption may have been.

4.3 Two Force Evaluations per Step

%\subsubsubsection{General Form}

{\bf 4.1.2.1. General Form}

If we allow two evaluations of the right-hand side of the differential equation, we can work with the following general expression that is dimensionally correct

$$\begin{aligned} k_1 &= f(x_0 + \eta_{11} v_0 \tau) && (4.53)\\ k_2 &= f(x_0 + \eta_{21} v_0 \tau + \eta_{22} k_1 \tau^2) && (4.54) \end{aligned}$$

which leads to the following expressions for position and velocity steps:

$$\begin{aligned} x_1 &= x_0 + v_0\tau + (\alpha_1 k_1 + \alpha_2 k_2)\,\tau^2 && (4.55)\\ v_1 &= v_0 + (\beta_1 k_1 + \beta_2 k_2)\,\tau && (4.56) \end{aligned}$$

Substituting the k_i values, these equations expand into

$$\left\{ \begin{array}{rcl} x_1 &=& x_0 + v_0\tau + \alpha_1 f(x_0 + \eta_{11} v_0 \tau)\tau^2 + \\ & & \alpha_2 f(x_0 + \eta_{21} v_0 \tau + \eta_{22} f(x_0 + \eta_{11} v_0 \tau)\tau^2)\tau^2 \\ v1 &=& v_0 + \beta_1 f(x_0 + \eta_{11} v_0 \tau)\tau + \\ & & \beta_2 f(x_0 + \eta_{21} v_0 \tau + \eta_{22} f(x_0 + \eta_{11} v_0 \tau)\tau^2)\tau \end{array} \right. \qquad (4.57)$$

So far, everything is still the exact prescription, given in the original algorithmic scheme. If we now expand the expressions in powers of τ , we get:

$$\left\{ \begin{array}{rcl} x_1 &=& x_0 + v_0\tau + (\alpha_1 + \alpha_2) f_0 \tau^2 + \\ & & (\alpha_1\eta_{11} + \alpha_2\eta_{21}) f_0' v_0 \tau^3 + O(\tau^4) \\ v1 &=& v_0 + (\beta_1 + \beta_2) f_0 \tau + (\beta_1\eta_{11} + \beta_2\eta_{21}) f_0' v_0 \tau^2 + \\ & & (\tfrac{1}{2}(\beta_1\eta_{11}^2 + \beta_2\eta_{21}^2) f_0'' v_0^2 + \beta_2\eta_{22} f_0 f_0')\tau^3 + O(\tau^4) \end{array} \right. \qquad (4.58)$$

This should be equal to the Taylor series:

$$\begin{aligned} x_1 &= x_0 + v_0\tau + \tfrac{1}{2}a_0\tau^2 + \tfrac{1}{6}j_0\tau^3 + \tfrac{1}{24}s_0\tau^4 + O(\tau^5) && (4.59)\\ v_1 &= v_0 + a_0\tau + \tfrac{1}{2}j_0\tau^2 + \tfrac{1}{6}s_0\tau^3 + \tfrac{1}{24}c_0\tau^4 + O(\tau^5) && (4.60) \end{aligned}$$

EXPAND

This leads to the following conditions:

$$\begin{cases} \alpha_1 + \alpha_2 = \frac{1}{2} \\ \alpha_1\eta_{11} + \alpha_2\eta_{21} = \dfrac{1}{6} \\ \beta_1 + \beta_2 = 1 \\ \beta_1\eta_{11} + \beta_2\eta_{21} = \frac{1}{2} \\ \beta_1\eta_{11}^2 + \beta_2\eta_{21}^2 = \dfrac{1}{3} \\ \beta_2\eta_{22} = \dfrac{1}{6} \end{cases} \tag{4.61}$$

These can be solved in terms of η_{11} , as follows:

$$\begin{cases} \eta_{21} = \dfrac{3\eta_{11} - 2}{3(2\eta_{11} - 1)} \\ \eta_{22} = \dfrac{2(3\eta_{11}^2 - 3\eta_{11} + 1)}{9(4\eta_{11}^2 - 4\eta_{11} + 1)} \\ \alpha_1 = \dfrac{-\eta_{11} + 1}{4(3\eta_{11}^2 - 3\eta_{11} + 1)} \\ \alpha_2 = \dfrac{6\eta_{11}^2 - 5\eta_{11} + 1}{4(3\eta_{11}^2 - 3\eta_{11} + 1)} \\ \beta_1 = \dfrac{1}{4(3\eta_{11}^2 - 3\eta_{11} + 1)} \\ \beta_2 = \dfrac{3(4\eta_{11}^2 - 4\eta_{11} + 1)}{4(3\eta_{11}^2 - 3\eta_{11} + 1)} \end{cases} \tag{4.62}$$

%\subsubsubsection{Examples}

{\bf 4.1.2.2. Examples}

If we take the standard assumption $\eta_{11} = 0$, we get:

$$\left\{\eta_{11} = 0 \ ; \ \eta_{21} = \tfrac{2}{3} \ ; \ \eta_{22} = \tfrac{2}{9} \ ; \ \alpha_1 = \tfrac{1}{4} \ ; \ \alpha_2 = \tfrac{1}{4} \ ; \ \beta_1 = \tfrac{1}{4} \ ; \ \beta_2 = \tfrac{3}{4}\right\} \tag{4.63}$$

This produces exactly what Nyström (1925) gives this as his simplest algorithm:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + v_0\tau + \frac{1}{4}\left(k_1 + k_2\right)\tau^2 \\ v_1 &=& v_0 + \frac{1}{4}\left(k_1 + 3k_2\right)\tau \\ & & \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \frac{2}{3}v_0\tau + \frac{2}{9}k_1\tau^2) \end{array}\right. \tag{4.64}$$

Henrici (1962) also lists this algorithm, refering back to Nyström (1925). However, Henrici's expressions contain a typo: he lists the last coefficient as $\frac{1}{3}k_1\tau^2$. Nyström does list the term correctly, as $\frac{2}{9}k_1\tau^2$.

If we try the other obvious choice $\eta_{11} = \frac{1}{2}$, we find that some of the coefficients diverge: $\eta_{21} = \eta_{21} = \infty$. With two force evaluations, it seems not to be possible to let the first one start right in the middle.

There is a natural choices that leads to relatively simple expressions for the coefficients: $\eta_{11} = \frac{1}{3}$. Here the first force evaluation takes place after one third of the duration of the time step. In this case we get:

$$\left\{\eta_{11} = \tfrac{1}{3}\ ;\ \eta_{21} = 1\ ;\ \eta_{22} = \tfrac{2}{3}\ ;\ \alpha_1 = \tfrac{1}{2}\ ;\ \alpha_2 = 0\ ;\ \beta_1 = \tfrac{3}{4}\ ;\ \beta_2 = \tfrac{1}{4}\right\} \tag{4.65}$$

This leads to the following equations:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + v_0\tau + \frac{1}{2}k_1\tau^2 \\ v_1 &=& v_0 + \frac{1}{4}\left(3k_1 + k_2\right)\tau \\ & & \\ k_1 &=& f(x_0 + \frac{1}{3}v_0\tau) \\ k_2 &=& f(x_0 + v_0\tau + \frac{2}{3}k_1\tau^2) \end{array}\right. \tag{4.66}$$

A complementary choise is $\eta_{11} = \frac{2}{3}$, for which the first force evaluation takes place after two third of the duration of the time step. In this case we get:

$$\left\{\eta_{11} = \tfrac{2}{3}\ ;\ \eta_{21} = 0\ ;\ \eta_{22} = \tfrac{2}{3}\ ;\ \alpha_1 = \tfrac{1}{4}\ ;\ \alpha_2 = \tfrac{1}{4}\ ;\ \beta_1 = \tfrac{3}{4}\ ;\ \beta_2 = \tfrac{1}{4}\right\} \tag{4.67}$$

This leads to the following equations:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + v_0\tau + \frac{1}{4}(k_1 + k_2)\tau^2 \\ v_1 &=& v_0 + \frac{1}{4}\left(3k_1 + k_2\right)\tau \\ & & \\ k_1 &=& f(x_0 + \frac{2}{3}v_0\tau) \\ k_2 &=& f(x_0 + \frac{2}{3}k_1\tau^2) \end{array}\right. \tag{4.68}$$

There seem to be no other sets of simple coefficients. We might be tempted to try, say, $\eta_{11} = \frac{1}{4}$, but in that case we get the much more complicated looking set:

$$\left\{\eta_{11} = \tfrac{1}{4}\ ;\ \eta_{21} = \tfrac{5}{6}\ ;\ \eta_{22} = \tfrac{7}{18}\ ;\ \alpha_1 = \tfrac{3}{7}\ ;\ \alpha_2 = \tfrac{1}{14}\ ;\ \beta_1 = \tfrac{4}{7}\ ;\ \beta_2 = \tfrac{3}{7}\right\} \quad (4.69)$$

This leads to the following equations:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + v_0\tau + \frac{1}{14}\left(6k_1 + k_2\right)\tau^2 \\ v_1 &=& v_0 + \frac{1}{7}\left(4k_1 + 3k_2\right)\tau \\ & & \\ k_1 &=& f(x_0 + \frac{1}{4}v_0\tau) \\ k_2 &=& f(x_0 + \frac{5}{6}v_0\tau + \frac{7}{18}k_1\tau^2) \end{array}\right. \quad (4.70)$$

xxx

Chapter 5

Recycling Force Evaluations

5.1 One Force Evaluation per Step

%\subsubsubsection{General Form} {\bf 4.2.1.1. General Form}

$$\begin{aligned} k_1 &= f(x_0) && (5.1)\\ k_2 &= f(x_0 + v_0\tau + \eta k_1 \tau^2) && (5.2) \end{aligned}$$

$$\begin{aligned} x_1 &= x_0 + v_0\tau + (\alpha_1 k_1 + \alpha_2 k_2)\,\tau^2 && (5.3)\\ v_1 &= v_0 + (\beta_1 k_1 + \beta_2 k_2)\,\tau && (5.4) \end{aligned}$$

3rd order not possible: in Eq. (4.63) we see that starting the first force calculation at time zero implies that the coefficient for v_0 in k_2 should be 2/3 , and not 1 as we insist upon above. **[EXPAND THIS]**

This means that we only have to expand up to powers in τ^2.

with

$$k_2 = f_0 + v_0 f_0' \tau + O(\tau^2) \tag{5.5}$$

we get

$$\begin{aligned} x_1 &= x_0 + v_0\tau + (\alpha_1 + \alpha_2)\, f_0 \tau^2 + O(\tau^3)\\ v_1 &= v_0 + (\beta_1 + \beta_2)\, f_0 \tau + \beta_2 v_0 f_0' \tau^2 + O(\tau^3) && (5.6) \end{aligned}$$

This has to be equal to the Taylor series expansions:

$$\begin{aligned} x_1 &= x_0 + v_0\tau + \tfrac{1}{2}f_0\tau^2 + O(\tau^3) \\ v_1 &= v_0 + f_0\tau + \tfrac{1}{2}v_0 f_0'\tau^2 + O(\tau^3) \end{aligned} \tag{5.7}$$

This implies:

$$\boxed{\alpha_1 + \alpha_2 = \tfrac{1}{2}} \tag{5.8}$$

and

$$\boxed{\beta_1 + \beta_2 = 1} \tag{5.9}$$

and

$$\boxed{\beta_2 = \tfrac{1}{2}} \tag{5.10}$$

From the last two, we get $\beta_1 = \frac{1}{2}$. Two parameter freedom, with $\alpha \equiv \alpha_2$:

$$\begin{cases} x_1 &= x_0 + v_0\tau + \frac{1}{2}\left((1-\alpha)k_1 + \alpha k_2\right)\tau^2 \\ v_1 &= v_0 + \frac{1}{2}\left(k_1 + k_2\right)\tau \\ & \\ k_1 &= f(x_0) \\ k_2 &= f(x_0 + v_0\tau + \eta k_1\tau^2) \end{cases} \tag{5.11}$$

%\subsubsubsection{Second Order Recycle Conditions}

{\bf 4.2.1.2. Second Order Recycle Conditions}

Now insist that $\tilde{x}_1 - x_1 = O(\tau^2)$:

$$\begin{aligned} \tilde{x}_1 &= x_0 + v_0\tau + \eta k_1\tau^2 \\ x_1 &= x_0 + v_0\tau + \tfrac{1}{2}\left((1-\alpha)k_1 + \alpha k_2\right)\tau^2 \end{aligned} \tag{5.12}$$

Already okay. So we are left with a two-parameter freedom.

For $\alpha = 0$ and $\eta = \frac{1}{2}$, simplest choice: leapfrog.

[check for which values time symmetry; presumably only for $\alpha = 0$ and $\eta = \frac{1}{2}$]

5.2 Two Force Evaluations per Step

%\subsubsubsection{General Form} {\bf 4.2.2.1. General Form}

$$\begin{aligned} k_1 &= f(x_0) \\ k_2 &= f(x_0 + \eta_{21} v_0 \tau + \eta_{22} k_1 \tau^2) \\ k_3 &= f(x_0 + v_0 \tau + \eta_{32} k_1 \tau^2 + \eta_{33} k_2 \tau^2) \end{aligned} \tag{5.13}$$

$$\begin{aligned} x_1 &= x_0 + v_0 \tau + (\alpha_1 k_1 + \alpha_2 k_2 + \alpha_3 k_3)\, \tau^2 \\ v_1 &= v_0 + (\beta_1 k_1 + \beta_2 k_2 + \beta_3 k_3)\, \tau \end{aligned} \tag{5.14}$$

Let us expand up to powers in τ^3.

$$k_2 = f_0 + \eta_{21} v_0 f_0' \tau + \left\{ \eta_{22} f_0 f_0' + \tfrac{1}{2} \eta_{21}^2 v_0^2 f_0'' \right\} \tau^2 + O(\tau^3) \tag{5.15}$$

$$k_3 = f_0 + v_0 f_0' \tau + \left\{ (\eta_{32} + \eta_{33})\, f_0 f_0' + \tfrac{1}{2} v_0^2 f_0'' \right\} \tau^2 + O(\tau^3) \tag{5.16}$$

we get

$$\begin{aligned} x_1 &= x_0 + v_0 \tau + (\alpha_1 + \alpha_2 + \alpha_3)\, f_0 \tau^2 \\ &\quad + (\alpha_2 \eta_{21} + \alpha_3)\, v_0 f_0' \tau^3 + O(\tau^4) \end{aligned} \tag{5.17}$$

This has to be equal to the Taylor series expansions:

$$x_1 = x_0 + v_0 \tau + \tfrac{1}{2} f_0 \tau^2 + \tfrac{1}{6} v_0 f_0' \tau^3 + O(\tau^4) \tag{5.18}$$

which implies

$$\boxed{\alpha_1 + \alpha_2 + \alpha_3 = \tfrac{1}{2}} \tag{5.19}$$

$$\boxed{\alpha_2 \eta_{21} + \alpha_3 = \tfrac{1}{6}} \tag{5.20}$$

$$\begin{aligned} v_1 &= v_0 + (\beta_1 + \beta_2 + \beta_3)\, f_0 \tau + (\beta_2 \eta_{21} + \beta_3)\, v_0 f_0' \tau^2 \\ &\quad + \left\{ \Big(\beta_2 \eta_{22} + \beta_3 (\eta_{32} + \eta_{33}) \Big) f_0 f_0' + \tfrac{1}{2} \left(\beta_2 \eta_{21}^2 + \beta_3 \right) v_0^2 f_0'' \right\} \tau^3 \\ &\quad + O(\tau^4) \end{aligned} \tag{5.21}$$

This has to be equal to the Taylor series expansions:

$$v_1 = v_0 + f_0\tau + \tfrac{1}{2}v_0 f_0'\tau^2 + \tfrac{1}{6}\left\{f_0 f_0' + v_0^2 f_0''\right\}\tau^3 + O(\tau^4) \tag{5.22}$$

which implies

$$\boxed{\beta_1 + \beta_2 + \beta_3 = 1} \tag{5.23}$$

$$\boxed{\beta_2\eta_{21} + \beta_3 = \tfrac{1}{2}} \tag{5.24}$$

$$\boxed{\beta_2\eta_{21}^2 + \beta_3 = \tfrac{1}{3}} \tag{5.25}$$

$$\boxed{\beta_2\eta_{22} + \beta_3(\eta_{32} + \eta_{33}) = \tfrac{1}{6}} \tag{5.26}$$

%\subsubsubsection{Third Order Recycle Conditions}

{\bf 4.2.2.2. Third Order Recycle Conditions}

Now insist that $\tilde{x}_1 - x_1 = O(\tau^3)$:

$$\begin{aligned} \tilde{x}_1 &= x_0 + v_0\tau + (\eta_{32}k_1 + \eta_{33}k_2)\,\tau^2 + O(\tau^3) \\ x_1 &= x_0 + v_0\tau + \tfrac{1}{2}f_0\tau^2 \end{aligned} \tag{5.27}$$

$$\boxed{\eta_{32} + \eta_{33} = \tfrac{1}{2}} \tag{5.28}$$

seven equations for ten variables.

From the last two:

$$\beta_2\eta_{22} + \tfrac{1}{2}\beta_3 = \tfrac{1}{6} \tag{5.29}$$

Combining that with the equation above the two we just used:

$$\beta_2\left(\tfrac{1}{2}\eta_{21}^2 - \eta_{22}\right) = 0 \tag{5.30}$$

Two possibilities: $\beta_2 = 0$ or $\eta_{22} = \frac{1}{2}\eta_{21}$. In the first case, $\beta_3 = \frac{1}{2}$ and $\beta_3 = \frac{1}{3}$ from the next to last equation and the one above that. Contradiction. Hence:

$$\eta_{22} = \tfrac{1}{2}\eta_{21} \tag{5.31}$$

Introduce $\alpha \equiv \alpha_2$ and $\eta \equiv \eta_{21}$, and use $\zeta \equiv \eta_{33}$ as the third parameter. Then

$$\left\{\begin{array}{lcl} \alpha_1 &=& \frac{1}{3} + \alpha(\eta - 1) \\ \alpha_2 &=& \alpha \\ \alpha_3 &=& \frac{1}{6} - \alpha\eta \\ \beta_1 &=& \dfrac{3\eta - 1}{6\eta} \\ \beta_2 &=& \dfrac{1}{6\eta(1-\eta)} \\ \beta_3 &=& \dfrac{2 - 3\eta}{6(1-\eta)} \\ \eta_{21} &=& \eta \\ \eta_{22} &=& \frac{1}{2}\eta^2 \\ \eta_{32} &=& \frac{1}{2} - \zeta \\ \eta_{33} &=& \zeta \end{array}\right. \tag{5.32}$$

So in general:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + v_0\tau + \left((\frac{1}{3} + \alpha(\eta - 1))k_1 + \alpha k_2 + (\frac{1}{6} - \alpha\eta)k_3\right)\tau^2 \\ v_1 &=& v_0 + \left(\frac{3\eta-1}{6\eta}k_1 + \frac{1}{6\eta(1-\eta)}k_2 + \frac{2-3\eta}{6(1-\eta)}k_3\right)\tau \\ & & \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \eta v_0\tau + \frac{1}{2}\eta^2 k_1\tau^2) \\ k_3 &=& f(x_0 + v_0\tau + ((\frac{1}{2} - \zeta)k_1 + \zeta k_2)\tau^2) \end{array}\right. \tag{5.33}$$

Example: for $\alpha = \zeta = 0$, $\eta = \frac{1}{2}$:

$$\left\{\begin{array}{lcl} x_1 &=& x_0 + v_0\tau + \frac{1}{6}\left(2k_1 + k_3\right)\tau^2 \\ v_1 &=& v_0 + \frac{1}{6}\left(k_1 + 4k_2 + k_3\right)\tau \\ & & \\ k_1 &=& f(x_0) \\ k_2 &=& f(x_0 + \frac{1}{2}v_0\tau + \frac{1}{8}k_1\tau^2) \\ k_3 &=& f(x_0 + v_0\tau + \frac{1}{2}k_1\tau^2) \end{array}\right. \tag{5.34}$$

Like Simpon's rule for velocity integration.

Another example: for $\alpha = \zeta = 0$, $\eta = \frac{1}{3}$:

$$\left\{ \begin{array}{lcl} x_1 & = & x_0 + v_0\tau + \frac{1}{6}\left(2k_1 + k_3\right)\tau^2 \\ v_1 & = & v_0 + \frac{1}{4}\left(3k_2 + k_3\right)\tau \\ & & \\ k_1 & = & f(x_0) \\ k_2 & = & f(x_0 + \frac{1}{3}v_0\tau + \frac{1}{18}k_1\tau^2) \\ k_3 & = & f(x_0 + v_0\tau + \frac{1}{2}k_1\tau^2) \end{array} \right. \tag{5.35}$$

%\subsubsubsection{Exact Recycling}

{\bf 4.2.2.3. Exact Recycling}

So far, only pseudo-FSAL, or better: pseudo Runge Kutta!

Can we make it really FSAL Runge Kutta, to all orders?

$$\frac{1}{2} - \zeta = \frac{1}{3} - \alpha(\eta - 1) \tag{5.36}$$

$$\zeta = \alpha \tag{5.37}$$

$$\frac{1}{6} - \alpha\eta = 0 \tag{5.38}$$

With the first two:

$$\frac{1}{6} + \alpha\eta = 2\alpha \tag{5.39}$$

combining this with the third:

$$\alpha = \frac{1}{6} \tag{5.40}$$

leading to

$$\zeta = \frac{1}{6} \tag{5.41}$$

but alas:

$$\eta = 1 \tag{5.42}$$

So this doesn't work: the coefficients for the k_2 and k_3 terms in the expression for v_1 in Eq. (5.33) blow up.

%\subsubsubsection{A Search for a Fourth Order Scheme}

{\bf 4.2.2.4. A Search for a Fourth Order Scheme}

Given that we have three free parameters left in our construction of a recycling scheme that is third order correct, it is tempting to search for a fourth-order scheme, based on only two new force calculations per time step.

Repeating the previous analysis to one order higher in τ , we get

$$\begin{aligned} k_2 &= f_0 + \eta_{21} v_0 f_0' \tau + \left\{ \eta_{22} f_0 f_0' + \tfrac{1}{2} \eta_{21}^2 v_0^2 f_0'' \right\} \tau^2 \\ &\quad + \left\{ \eta_{21} \eta_{22} v_0 f_0 f_0'' + \tfrac{1}{6} \eta_{21}^3 v_0^3 f_0''' \right\} \tau^3 + O(\tau^4) \end{aligned} \tag{5.43}$$

$$\begin{aligned} k_3 &= f_0 + v_0 f_0' \tau + \left\{ (\eta_{32} + \eta_{33}) f_0 f_0' + \tfrac{1}{2} v_0^2 f_0'' \right\} \tau^2 \\ &\quad + \left\{ \eta_{21} \eta_{33} v_0 \left(f_0' \right)^2 + (\eta_{32} + \eta_{33}) v_0 f_0 f_0'' + \tfrac{1}{6} v_0^3 f_0''' \right\} \tau^3 \\ &\quad + O(\tau^4) \end{aligned} \tag{5.44}$$

we get

$$\begin{aligned} x_1 &= x_0 + v_0 \tau + (\alpha_1 + \alpha_2 + \alpha_3) f_0 \tau^2 + (\alpha_2 \eta_{21} + \alpha_3) v_0 f_0' \tau^3 \\ &\quad + (\alpha_2 \eta_{22} + \alpha_3 (\eta_{32} + \eta_{33})) f_0 f_0' \tau^4 + \tfrac{1}{2} \left(\alpha_2 \eta_{21}^2 + \alpha_3 \right) v_0^2 f_0'' \tau^4 \\ &\quad + O(\tau^5) \end{aligned} \tag{5.45}$$

This has to be equal to

$$x_1 = x_0 + v_0 \tau + \tfrac{1}{2} f_0 \tau^2 + \tfrac{1}{6} v_0 f_0' \tau^3 + \tfrac{1}{24} f_0 f_0' \tau^4 + \tfrac{1}{24} v_0^2 f_0'' \tau^4 + O(\tau^5) \tag{5.46}$$

In addition to the previous conditions, we get the following two additional requirements:

$$\boxed{\alpha_2 \eta_{22} + \alpha_3 (\eta_{32} + \eta_{33}) = \tfrac{1}{24}} \tag{5.47}$$

$$\boxed{\alpha_2 \eta_{21}^2 + \alpha_3 = \tfrac{1}{12}} \tag{5.48}$$

Now:

$$\begin{aligned} v_1 &= v_0 + (\beta_1 + \beta_2 + \beta_3) f_0 \tau + (\beta_2 \eta_{21} + \beta_3) v_0 f_0' \tau^2 \\ &\quad + \left\{ \Big(\beta_2 \eta_{22} + \beta_3 (\eta_{32} + \eta_{33}) \Big) f_0 f_0' + \tfrac{1}{2} \left(\beta_2 \eta_{21}^2 + \beta_3 \right) v_0^2 f_0'' \right\} \tau^3 \\ &\quad + \Big\{ \beta_3 \eta_{21} \eta_{33} v_0 \left(f_0' \right)^2 + \Big(\beta_2 \eta_{21} \eta_{22} + \beta_3 (\eta_{32} + \eta_{33}) \Big) v_0 f_0 f_0'' \\ &\qquad + \tfrac{1}{6} \left(\beta_2 \eta_{21}^3 + \beta_3 \right) v_0^3 f_0''' \Big\} \tau^4 \\ &\quad + O(\tau^5) \end{aligned} \tag{5.49}$$

This has to be equal to

$$\begin{aligned} v_1 &= v_0 + f_0\tau + \tfrac{1}{2}v_0 f_0'\tau^2 + \tfrac{1}{6}\left\{f_0 f_0' + v_0^2 f_0''\right\}\tau^3 \\ &\quad + \tfrac{1}{24}\left\{v_0\left(f_0'\right)^2 + 3v_0 f_0 f_0'' + v_0^3 f_0'''\right\}\tau^4 + O(\tau^5) \end{aligned} \tag{5.50}$$

We get the additional equations:

$$\boxed{\beta_2\eta_{21}^3 + \beta_3 = \tfrac{1}{4}} \tag{5.51}$$

$$\boxed{\beta_2\eta_{21}\eta_{22} + \beta_3(\eta_{32} + \eta_{33}) = \tfrac{1}{8}} \tag{5.52}$$

$$\boxed{\beta_3\eta_{21}\eta_{33} = \tfrac{1}{24}} \tag{5.53}$$

We have thus eleven conditions gathered so far for the ten unknown parameters $\{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \eta_{21}, \eta_{22}, \eta_{32}, \eta_{33}\}$. *A priori* we would expect to find no solutions in such an overdetermined system. However, let's see how far we get when we try. Let us list the conditions here together:

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 = \frac{1}{2} \\ \alpha_2\eta_{21} + \alpha_3 = \frac{1}{} \\ \alpha_2\eta_{21}^2 + \alpha_3 = \frac{1}{12} \\ \alpha_2\eta_{22} + \alpha_3\left(\eta_{32} + \eta_{33}\right) = \frac{1}{24} \\ \beta_1 + \beta_2 + \beta_3 = 1 \\ \beta_2\eta_{21} + \beta_3 = \frac{1}{2} \\ \beta_2\eta_{21}^2 + \beta_3 = \frac{1}{3} \\ \beta_2\eta_{21}^3 + \beta_3 = \frac{1}{4} \\ \beta_2\eta_{22} + \beta_3(\eta_{32} + \eta_{33}) = \frac{1}{6} \\ \beta_2\eta_{21}\eta_{22} + \beta_3(\eta_{32} + \eta_{33}) = \frac{1}{8} \\ \beta_3\eta_{21}\eta_{33} = \frac{1}{24} \end{cases} \tag{5.54}$$

Subtracting the 6th and 7th equation, we find

$$\beta_2 \eta_{21}(1-\eta_{21}) = \tfrac{1}{6} \tag{5.55}$$

and subtracting the 7th and 8th equation, we find

$$\beta_2 \eta_{21}^2(1-\eta_{21}) = \tfrac{1}{12} \tag{5.56}$$

Together, these two expressions imply

$$\eta_{21}(1-\eta_{21}) = 2\eta_{21}^2(1-\eta_{21}) \tag{5.57}$$

There are three solutions: $\eta_{21} = 0$, $\eta_{21} = 1$, and $\eta_{21} = \frac{1}{2}$. The first two solutions can be discarded, because they would imply that the left-hand side of the 6th, 7th, and 8th equations above would all have the same value, contradicting the fact that their right-hand sides have different values. We thus find

$$\eta_{21} = \tfrac{1}{2} \tag{5.58}$$

With this result, we can use the remaining information in the 6th, 7th, and 8th equations above to determine the other two values:

$$\beta_2 = \tfrac{2}{3} \tag{5.59}$$

$$\beta_3 = \tfrac{1}{6} \tag{5.60}$$

The 5th equation gives us

$$\beta_1 = \tfrac{1}{6} \tag{5.61}$$

Subtracting the 9th and 10th equation, we find

$$\beta_2 \eta_{22}(1-\eta_{21}) = \tfrac{1}{24} \tag{5.62}$$

and plugging in the values we have found so far gives us

$$\eta_{22} = \tfrac{1}{8} \tag{5.63}$$

The 9th and 10th equation then give us:

$$\eta_{32} + \eta_{33} = \tfrac{1}{2} \tag{5.64}$$

Since the 11th equation gives us

$$\eta_{33} = \tfrac{1}{2} \tag{5.65}$$

we conclude that

$$\eta_{32} = 0 \tag{5.66}$$

We can now write the 2nd and 3rd equations as

$$\tfrac{1}{2}\alpha_2 + \alpha_3 = \tfrac{1}{6} \tag{5.67}$$

$$\tfrac{1}{4}\alpha_2 + \alpha_3 = \tfrac{1}{12} \tag{5.68}$$

Subtraction those expressions gives us

$$\tfrac{1}{4}\alpha_2 = \tfrac{1}{12} \tag{5.69}$$

or

$$\alpha_2 = \tfrac{1}{3} \tag{5.70}$$

and plugging this back in the expressions above gives

$$\alpha_3 = 0 \tag{5.71}$$

The 1st equation then gives

$$\alpha_1 = \tfrac{1}{6} \tag{5.72}$$

Remarkably, we have been able to solve the eleven equations for the ten unknowns and found a consistent solution! To summarize:

$$\begin{cases} \alpha_1 = \frac{1}{6} \quad ; \quad \alpha_2 = \frac{1}{3} \quad ; \quad \alpha_3 = 0 \\ \beta_1 = \frac{1}{6} \quad ; \quad \beta_2 = \frac{2}{3} \quad ; \quad \beta_3 = \frac{1}{6} \\ \eta_{21} = \frac{1}{2} \quad ; \quad \eta_{22} = \frac{1}{8} \quad ; \quad \eta_{32} = 0 \quad ; \quad \eta_{33} = \frac{1}{2} \end{cases} \tag{5.73}$$

Not only that, it turns out that we get an additional bonus: these solutions solve the previous relation for demanding the $\tilde{x}_1 = x_1 + O(\tau^3)$, which was

$$\eta_{32} + \eta_{33} = \tfrac{1}{2} \tag{5.74}$$

We have thus found a consistent set of solutions for ten variables satisfying twelve equations. Could we be really lucky? Could it be that in fact $\tilde{x}_1 = x_1 + O(\tau^4)$? If that were true, our fourth-order scheme would allow us to recycle the last force calculation, and we would really have obtained a fourth-order scheme with an effective costs of only two new force calculations per step. This does sound too good to be true, but let's just check.

$$\begin{aligned}
\tilde{x}_1 - x_1 &= \left\{(\eta_{32}k_1 + \eta_{33}k_2) - (\alpha_1 k_1 + \alpha_2 k_2 + \alpha_3 k_3)\right\}\tau^2 \\
&= \left\{\tfrac{1}{2}k_2 - \left(\tfrac{1}{6}k_1 + \tfrac{1}{3}k_2\right)\right\}\tau^2 \\
&= \tfrac{1}{6}(k_2 - k_1)\tau^2 \\
&= \tfrac{1}{12}v_0 f_0' \tau^3 + O(\tau^4)
\end{aligned} \tag{5.75}$$

It was too good to be true!

We thus have:

$$\left\{\begin{array}{lcl}
x_1 &=& x_0 + v_0\tau + \tfrac{1}{6}\left(k_1 + 2k_2\right)\tau^2 \\
v_1 &=& v_0 + \tfrac{1}{6}\left(k_1 + 4k_2 + k_3\right)\tau \\
 & & \\
k_1 &=& f(x_0) \\
k_2 &=& f(x_0 + \tfrac{1}{2}v_0\tau + \tfrac{1}{8}k_1\tau^2) \\
k_3 &=& f(x_0 + v_0\tau + \tfrac{1}{2}k_2\tau^2)
\end{array}\right. \tag{5.76}$$

Looking at the equations this way, we can in fact see directly that a fourth-order scheme doesn't work ('directly' once you have become sufficiently familiar with all these expressions). For the scheme to be fourth order, the position where the last force calculation is computed should agree to third order with the new position at the end of the time step. However, the latter has a k_1 term which the former lacks, and since the difference $k_1 - k_2$ is of first order in τ , there is a real third-order difference between the two positions, hence between the forces computed in these two positions. The upshot is that this will introduce a fourth-order error in the velocity in the next step, when we recycle the last force calculation. Our scheme is thus only third-order accurate when we recycle, even though it is fourth-order accurate if we decide to compute all three new forces anew at each step.

By the way, as a fourth-order scheme, it is listed in Abramowitz and Stegun's welknown Handbook of Mathematical Functions as eq. 25.5.22, but with a typo: the error in the position is listed as being $O(\tau^4)$, while it really should be $O(\tau^5)$; in addition no error is listed for the velocity. As we have seen, for the velocity, too, the error is $O(\tau^5)$.

nil nil nil nil nil

Chapter 6

Literature References

Abramowitz, M. and Stegun, I.A., 1965, Handbook of Mathematical Functions [Dover], p. xxx.

Delambre, J., 1792, Mem. Acad. Turin, 5, 143, 1790-1793

Henrici, P., 1962, Discrete Variable Methods in Ordinary Differential Equations [Wiley], pp. 172-173

Nyström, E.J., 1925, Acta Soc. Sci. Fenn., 50, No.13, 1-55

Toxvaerd, S., 1993, J. Chem. Phys. 99, 2277 (1993),