Integration Schemes

Keerthana P. G.

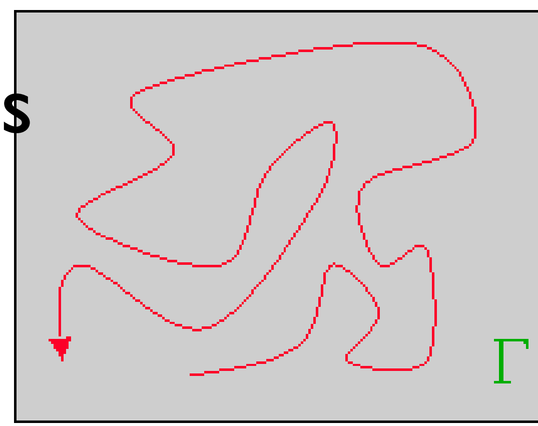
Revisiting some Familiar Concepts

- State of a System: The different configurations in which a system can exist
- Specifying the State of a system: State of a mechanical system is specified by the positions and momenta of all particles of the system

- Phase space: a space in which all possible states of a system are represented, with each possible state of the system corresponding to one unique point in the phase space.

$$\Gamma = (p^N, r^N)$$

•Every state is a point in the phase space and the dynamics of a system can be visualised as its movement through the phase space.



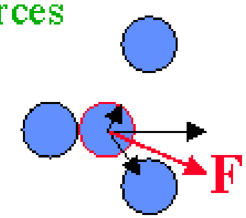
Integrators

- Algorithms used to model the time evolution of a system given its initial state using the equations of motion.

$$\boxed{\begin{aligned}\frac{d\mathbf{r}_j}{dt} &= \frac{\mathbf{p}_j}{m}\\ \frac{d\mathbf{p}_j}{dt} &= \mathbf{F}_j\end{aligned}}$$

$$\left.\begin{aligned}\mathbf{r} &= (r_x, r_y)\\ \mathbf{p} &= (p_x, p_y)\end{aligned}\right\}\ \text{2-dimensional space (for example)}$$

$$\mathbf{F}_j = \sum_{\substack{i=1\\ i\neq j}}^{N} \mathbf{F}_{ij} \quad \text{pairwise additive forces}$$



Characteristics of a good Integrator

- Conserves energy well and is time-reversible
- Takes up little memory
- Allows a long time step δt
- Only one calculation of forces per time step(*)

Important

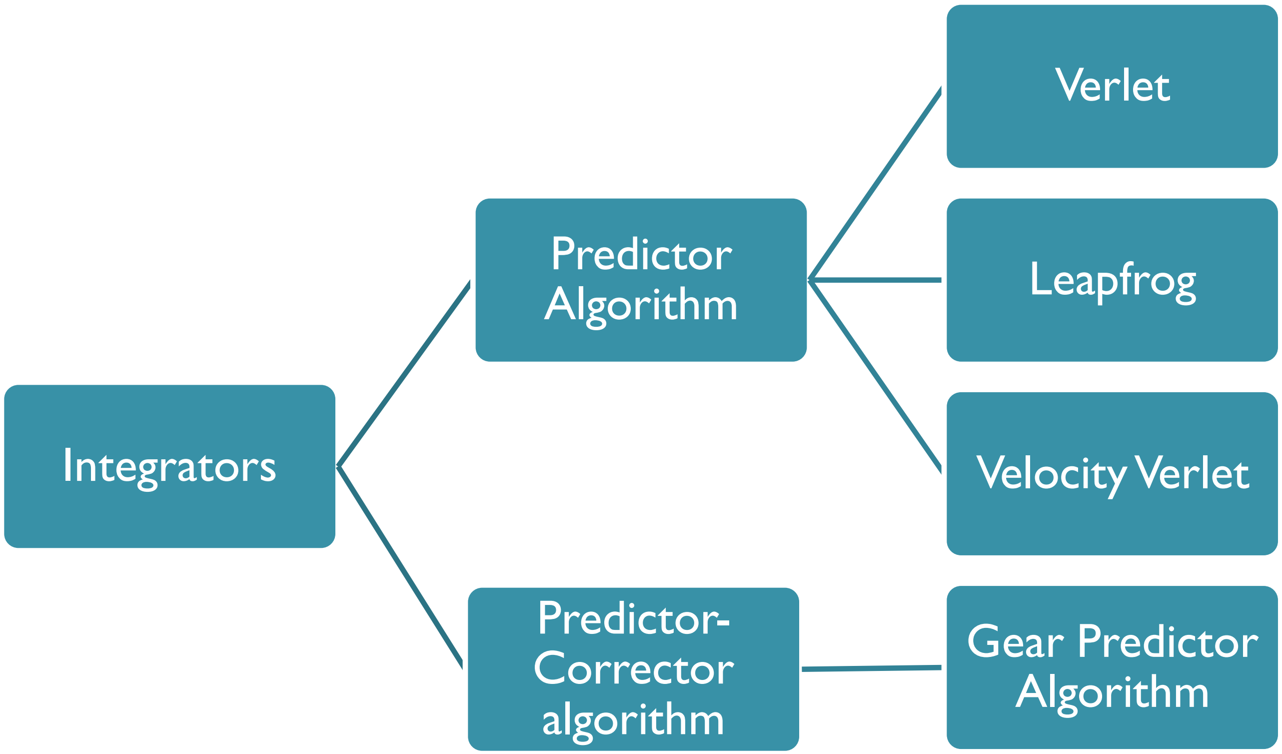
- Fast
- Easy to implement

Not so important

- Reproduces the correct path

Despite the quality of the integrator, there are always numerical errors (round-off etc.) which will lead to deviations that grow in time.

Integrators



Verlet Algorithm

$$r_{n+1} = r_n + v_n \Delta t + \frac{1}{2}\left(\frac{F_n}{m}\right)\Delta t^2 + O\left(\Delta t^3\right)..$$

$$r_{n-1} = r_n - v_n \Delta t + \frac{1}{2}\left(\frac{F_n}{m}\right)\Delta t^2 - O\left(\Delta t^3\right)..$$

- Sum the forward and backward expansions

$$r_{n+1} = 2r_n - r_{n-1} + \left(\frac{F_n}{m}\right)\Delta t^2 + O\left(\Delta t^4\right)..$$

1. Use r_n to calculate F_n
2. Use r_n, r_{n-1} and F_n (step1) to calculate r_{n+1}

Verlet Algorithm

- Subtract the backward expansion from the forward

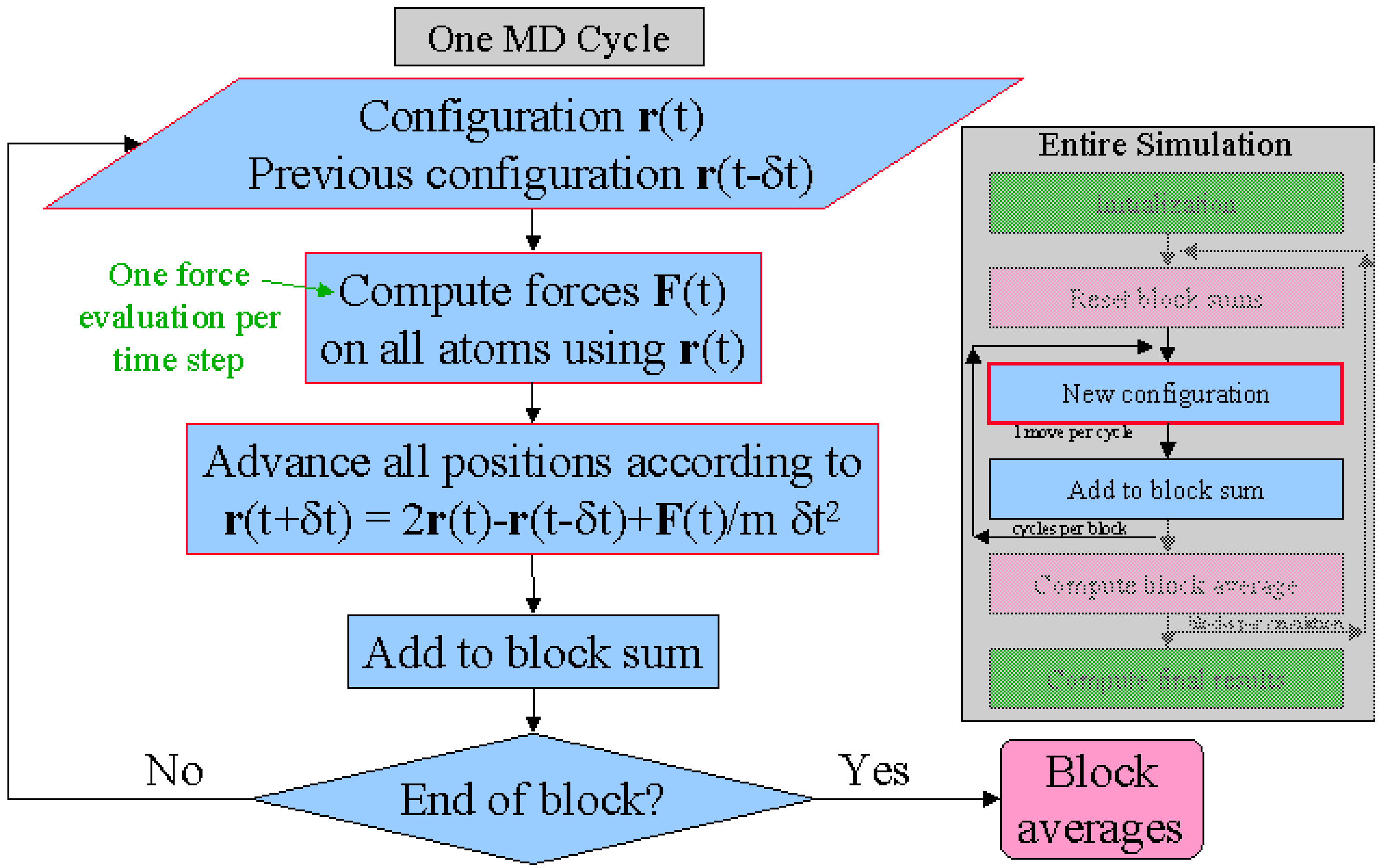
$$v_n = \frac{r_{n+1} - r_{n-1}}{2\Delta t} + O\left(\Delta t^2\right)$$

Verlet Algorithm: Loose Ends

- How to get positon at "previous time step" when starting out?
- Simple approximation

$$\mathbf{r}(t_0 - \delta t) = \mathbf{r}(t_0) - \mathbf{v}(t_0)\delta t$$

Verlet Algorithm: Flow Chart



Verlet algorithm: Advantages

- Integration does not require the velocities, only position information is taken into account.
- Only a single force evaluation per integration cycle. (Force evaluation is the most computationally expensive part in the simulation).
- This formulation, which is based on forward and backward expansions, is naturally reversible in time (a property of the equation of motion).

- Time reversibility:
  - *forward time step*

$$\mathbf{r}(t+\delta t) = 2\mathbf{r}(t) - \mathbf{r}(t-\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t^2$$

  - *replace δt with $-\delta t$*

$$\mathbf{r}(t+(-\delta t)) = 2\mathbf{r}(t) - \mathbf{r}(t-(-\delta t)) + \tfrac{1}{m}\mathbf{F}(t)(-\delta t)^2$$

$$\mathbf{r}(t-\delta t) = 2\mathbf{r}(t) - \mathbf{r}(t+\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t^2$$

  - *same algorithm, with same positions and forces, moves system backward in time*

Verlet Algorithm: Disadvantages

- Error in velocity approximation is of the order of time step squared(large errors).

$$v_n = \frac{r_{n+1} - r_{n-1}}{2\Delta t} + O\left(\Delta t^2\right)$$

- Need to know r(n+1)to calculate v(n).
- Numerical imprecision in adding small and large numbers.

$$\overbrace{\mathbf{r}(t+\delta t)-\mathbf{r}(t)}^{O(\delta t^1)} = \overbrace{\underbrace{\mathbf{r}(t)}_{O(\delta t^0)} - \underbrace{\mathbf{r}(t-\delta t)}_{O(\delta t^0)}}^{O(\delta t^1)} + \underbrace{\frac{1}{m}\mathbf{F}(t)\delta t^2}_{O(\delta t^2)}$$

Leap Frog Algorithm

$$v_{n-\frac{1}{2}} \equiv v\left(t-\frac{\Delta t}{2}\right) \equiv \frac{r(t)-r(t-\Delta t)}{\Delta t} = \frac{r_n - r_{n-1}}{\Delta t}$$

$$v_{n+\frac{1}{2}} \equiv v\left(t+\frac{\Delta t}{2}\right) \equiv \frac{r(t+\Delta t)-r(t)}{\Delta t} = \frac{r_{n+1} - r_n}{\Delta t}$$

- Evaluate velocities at the midpoint of the position evaluations and Vice versa.

$$v_{n+\frac{1}{2}} = v_{n-\frac{1}{2}} + \left(\frac{F_n}{m}\right)\Delta t$$

Where v $_{n+1/2}$ is the velocity at t+(1/2)Δt

$$r_{n+1} = r_n + v_{n+\frac{1}{2}}\Delta t$$

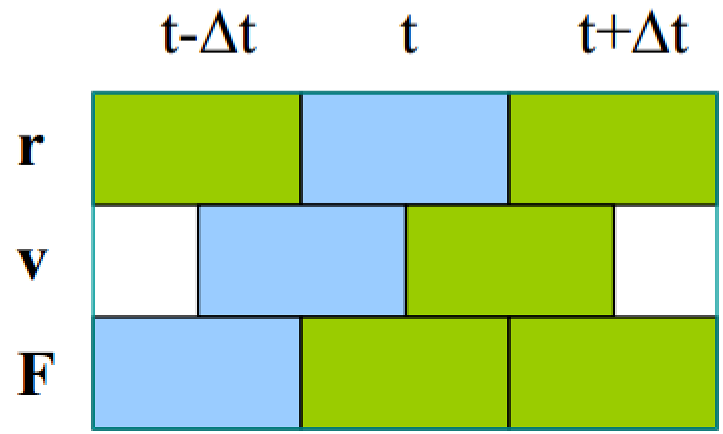
1. Use r(n) to calculate F(n).
2. Use F(n) and v(n-1/2)to calculate v (n+1/2).
3. Use r(n) and v(n+1/2) to calculate r(n+1).

$$v_n = \left(v_{n+1/2} + v_{n-1/2} \right) / 2$$

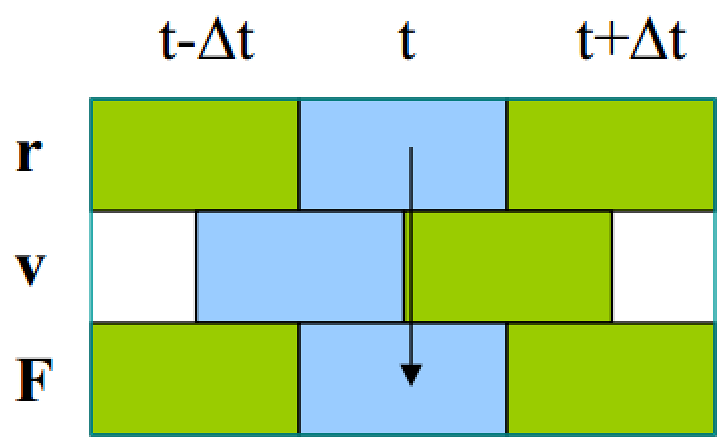
Instantaneous velocity at time t

Leap Frog: Flow Chart

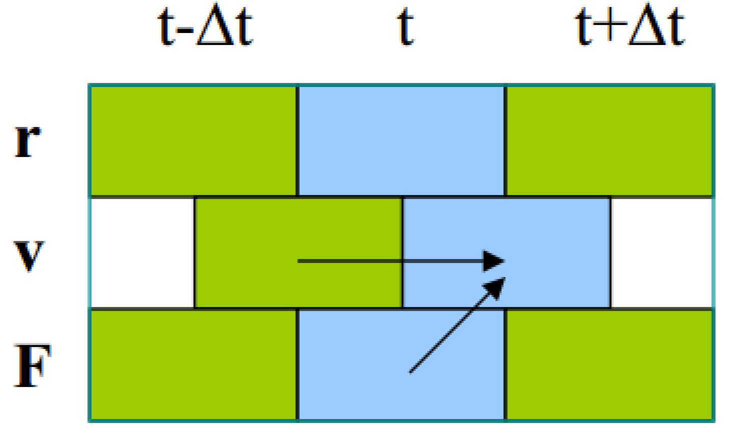
- Given current position, and velocity at last half-step



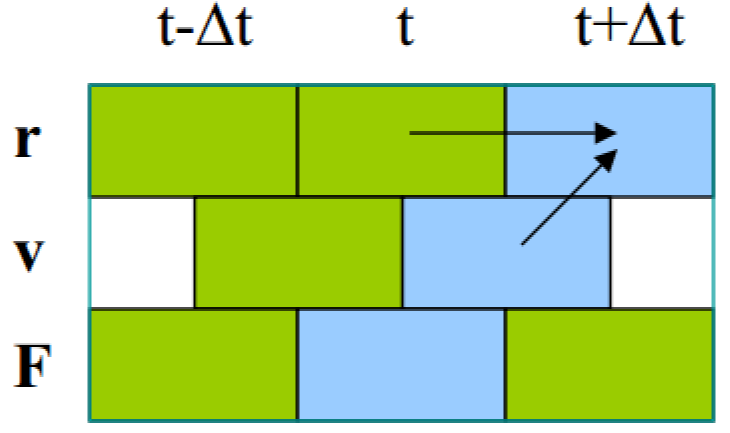
- Compute current force



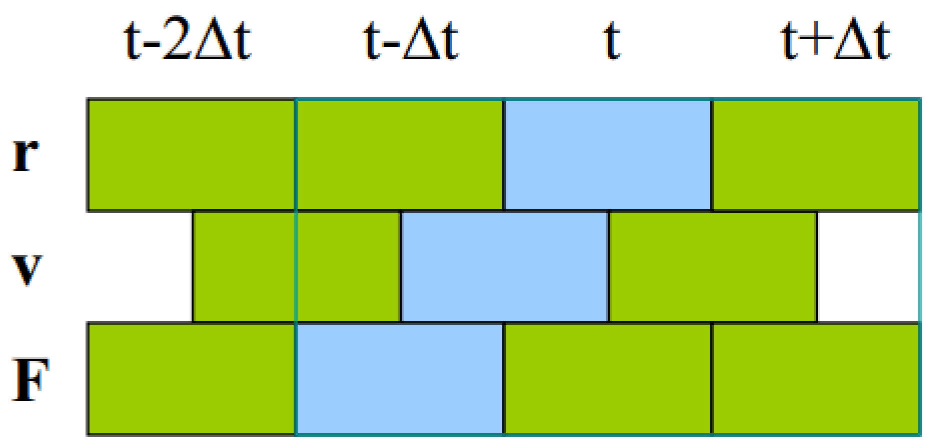
- Compute velocity at next half-step



- Compute next position



- Advance to next time step,
- repeat



Leap Frog : Advantages

- Eliminates addition of small numbers to large ones. Reduces the numerical error problem of the Verlet algorithm. Here $O(\Delta t1)$ terms are added to $O(\Delta t0)$ terms. Hence Improved evaluation of velocities.
- Direct evaluation of velocities gives a useful handle for controlling the temperature in the simulation.

Leap Frog : Disadvantages

- The velocities at time t are still approximate.
- Computationally a little more expensive than Verlet.

Velocity Verlet

$$r_{n+1} = r_n + v_n \Delta t + \frac{1}{2}\left(\frac{F_n}{m}\right)\Delta t^2$$

$$v_{n+1} = v_n + \frac{1}{2}\left(\frac{F_n}{m} + \frac{F_{n+1}}{m}\right)\Delta t^2$$

Derivation

$r(t+h)=r(t)+v(t)h+1/2a(t)h^2+O(h^3)$

$v(t+h)=v(t)+a(t)h+1/2b(t)h^2+O(h^3)$ (1)

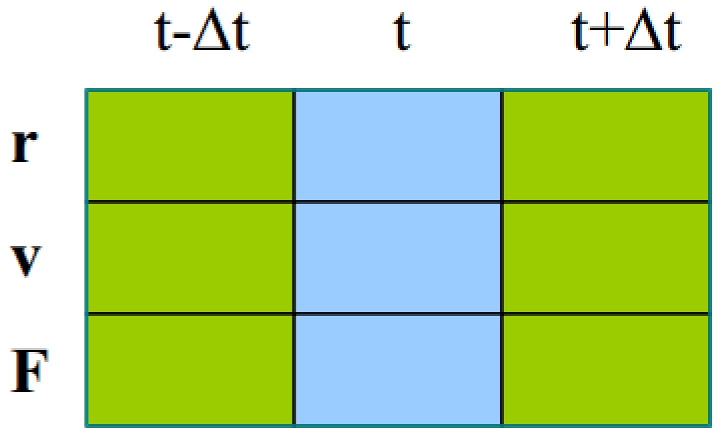
$v(t)=v(t+h)-a(t+h)h+1/2b(t+h)h^2+O(h^3)$ (2)

Subtracting (2) from (1),

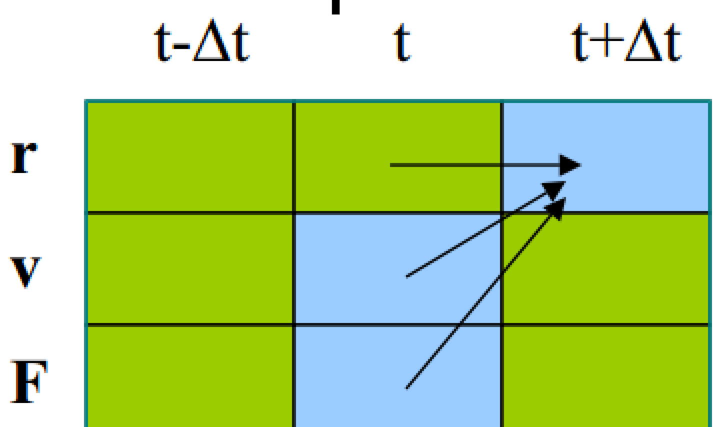
$2v(t+h)=2v(t)+h[a(t)+a(t+h)]+1/2[b(t)-b(t+h)]\ h^2$

$v(t+h)=v(t)+h/2[a(t)+a(t+h)]+O(h^3)$

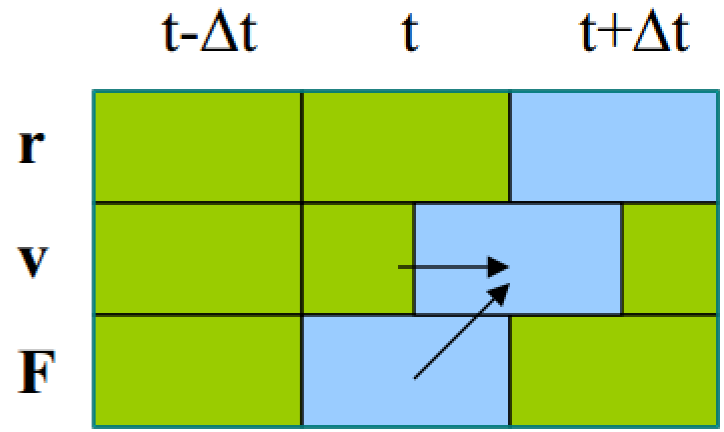
- Given current position, velocity, and force



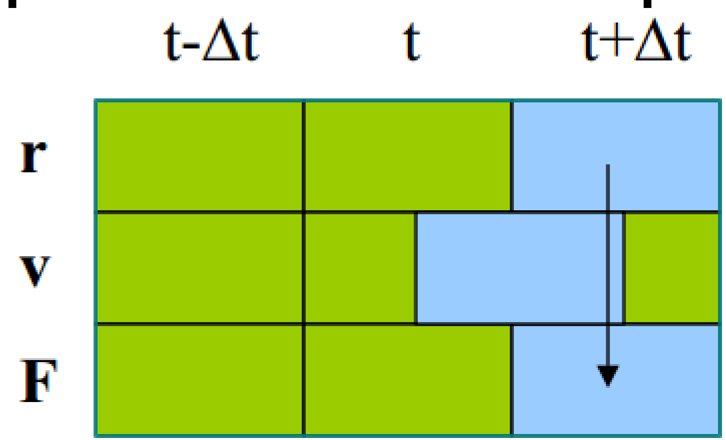
- Compute new position



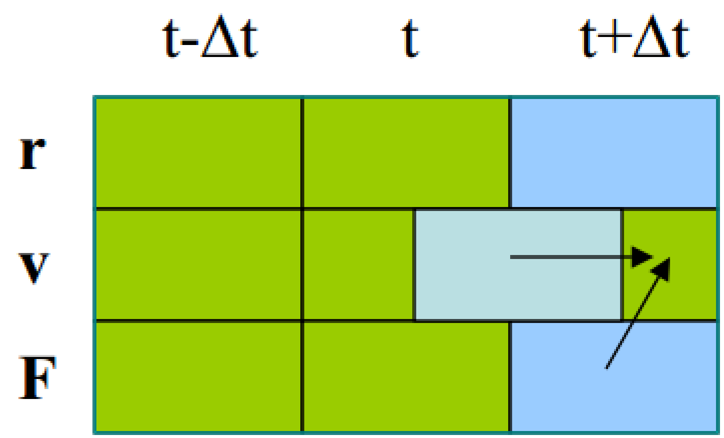
- Compute velocity at half step



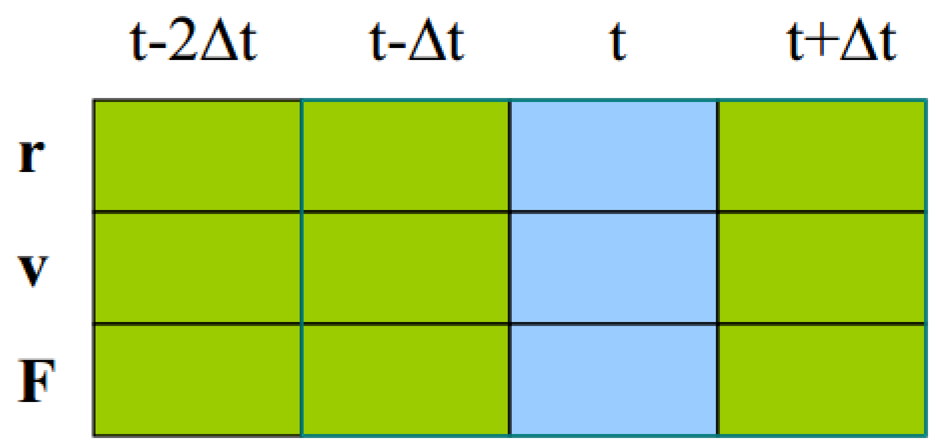
- Compute force at new position



- Compute velocity at full step



- Advance to next time step, repeat



Verlet Algorithm: Advantages

- is a second order integration scheme, i. e. the error term is O ((∂t)3).
- is explicit, i. e. without reference into the future: The system at time t+∂t can be calculated directly from quantities known at time t.
- is self-starting, i. e. without reference into the far past: The system at time ∂t can be calculated directly knowing only the system at time t = 0.
- allows ∂t to be chosen differently for each time. This can be very useful when the accelerations vary strongly over time.
- requires only one evaluation of the accelerations per timestep.

Gear Predictor-Corrector Algorithm

As the name implies, solving the equations of motion is carried out in two stages. First, use Taylor series to estimate the changes in positions and their time derivatives (in this case up to third order) over a small time step δt ...

Predictor stage

$$\left\{\begin{aligned}
\vec{r}_i^{\,p}(t+\delta t) &= \vec{r}_i(t)+\vec{v}_i(t)\delta t+\frac{1}{2}\vec{a}_i(t)\delta t^2+\frac{1}{6}\vec{b}_i(t)\delta t^3\\
\vec{v}_i^{\,p}(t+\delta t) &= \vec{v}_i(t)+\vec{a}_i(t)\delta t+\frac{1}{2}\vec{b}_i(t)\delta t^2\\
\vec{a}_i^{\,p}(t+\delta t) &= \vec{a}_i(t)+\vec{b}_i(t)\delta t\\
\vec{b}_i^{\,p}(t+\delta t) &= \vec{b}_i(t)
\end{aligned}\right.$$

... Then, calculate the forces and accelerations from the predicted positions and calculate a correction term ...

$$\vec{a}_i^c(t+\delta t) = \frac{1}{m}\vec{f}_i^c \qquad \Delta\vec{a}_i(t+\delta t) = \vec{a}_i^c(t+\delta t) - \vec{a}_i^p(t+\delta t)$$

... Which is then used to calculate the corrected positions (and their time derivatives)

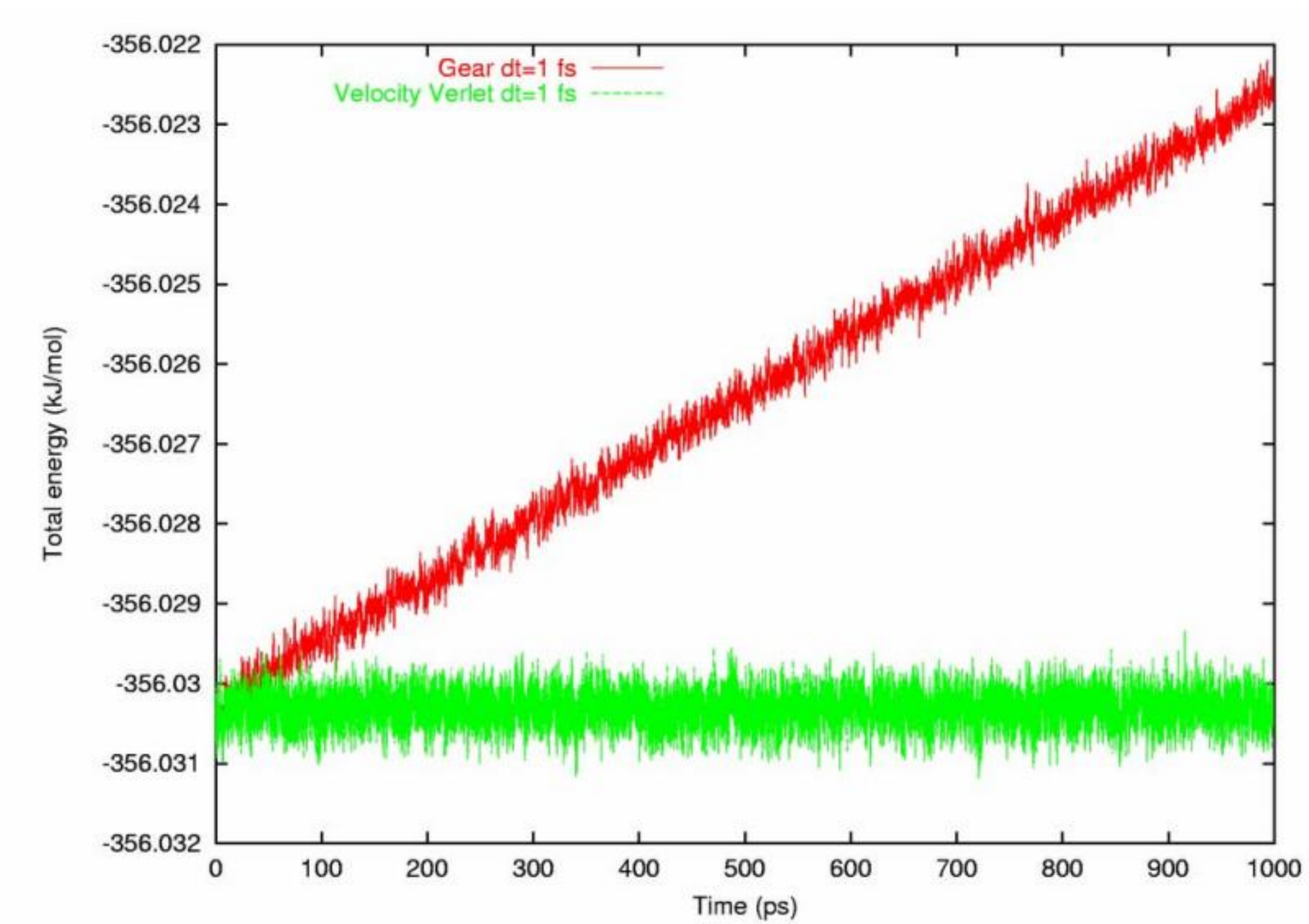
Corrector stage

$$\begin{cases} \vec{r}_i^c(t+\delta t) = \vec{r}_i^p(t+\delta t) + c_0\Delta\vec{a}_i(t+\delta t) \\ \vec{v}_i^c(t+\delta t) = \vec{v}_i^p(t+\delta t) + c_1\Delta\vec{a}_i(t+\delta t) \\ \vec{a}_i^c(t+\delta t) \qquad \text{(Got this already)} \\ \vec{b}_i^p(t+\delta t) = \vec{b}_i^p(t+\delta t) + c_3\Delta\vec{a}_i(t+\delta t) \end{cases}$$

- The correction coefficients have been determined by gear and tabulated.

No. Eqs	d					
	$\mathbf{r}$	$\mathbf{v} = d\mathbf{r}/dt$	$\mathbf{a} = d^2\mathbf{r}/dt^2$	$d^3\mathbf{r}/dt^3$	$d^4\mathbf{r}/dt^4$	$d^5\mathbf{r}/dt^5$
3	0	1	1			
4	1/6	5/6	1	1/3		
5	19/120	3/4	1	1/2	1/12	
6	3/20	251/360	1	11/18	1/6	1/60

Gear Predictor-Corrector Algorithm: Advantages



Gear Predictor-Corrector Algorithm: Disadvantages

- Not time reversible.
- Not symplectic, therefore does not conserve phase space volume.
- Not energy conserving, implies over time there is a gradual energy drift.

Thank You

Velocity Corrected Verlet

$$r(t+2\Delta t) = r(t) + 2v(t)\Delta t + \dot{v}(t)(2\Delta t)^2/2! + \ddot{v}(t)(2\Delta t)^3/3! + \ldots$$

$$r(t+\Delta t) = r(t) + v(t)\Delta t + \dot{v}(t)(\Delta t)^2/2! + \ddot{v}(t)(\Delta t)^3/3! + \ldots$$

$$r(t-\Delta t) = r(t) - v(t)\Delta t + \dot{v}(t)(\Delta t)^2/2! - \ddot{v}(t)(\Delta t)^3/3! + \ldots$$

$$r(t-2\Delta t) = r(t) - 2v(t)\Delta t + \dot{v}(t)(2\Delta t)^2/2! - \ddot{v}(t)(2\Delta t)^3/3! + \ldots$$

$$12v(t)\Delta t = 8[r(t+\Delta t) - r(t-\Delta t)] - [r(t+2\Delta t) - r(t-2\Delta t)]$$

$$v(t) = \frac{v(t+\Delta t/2) - v(t-\Delta t/2)}{2} + \frac{\Delta t}{12}[\dot{v}(t-\Delta t) - \dot{v}(t+\Delta t)]$$