Pracownia astrofizyki teoretycznej 1

Wykład 2

Mariusz Tarnopolski

Instytut Astronomii UMK

PATEO 1 ©2025



Metody numeryczne rozwiązywania równań różniczkowych

Rozwiązanie równania postaci

$$\dot{\mathbf{y}}(t) = \mathbf{f}\left[t, \mathbf{y}(t)\right]$$

z warunkiem początkowym

$$\mathbf{y}(t_0) = \mathbf{y}_0$$

Dyskretyzacja z krokiem h:

$$t_{n+1} = t_n + h$$

prowadzi do rozwiązań

$$\mathbf{y}_n \approx \mathbf{y}(t_n)$$

gdzie

$$\mathbf{y} \in \mathbb{R}^m$$

Metoda Eulera (*explicit Euler*)

Ekstrapolacja liniowa ze stałym krokiem h:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(t_n, \mathbf{y}_n)$$

Metoda Eulera (*explicit Euler*)

Ekstrapolacja liniowa ze stałym krokiem h:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(t_n, \mathbf{y}_n)$$

Dla $\ddot{\mathbf{r}} = \mathbf{a}(\mathbf{r})$, które można zapisać jako układ dynamiczny

$$\begin{cases} \dot{\mathbf{v}} &= \mathbf{a}(\mathbf{r}) \\ \dot{\mathbf{r}} &= \mathbf{v} \end{cases}$$

mamy

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{a}(\mathbf{r}_n)$$

$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\mathbf{v}_n$$

Metoda Eulera (*explicit Euler*)

Ekstrapolacja liniowa ze stałym krokiem h:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(t_n, \mathbf{y}_n)$$

Dla $\ddot{\mathbf{r}} = \mathbf{a}(\mathbf{r})$, które można zapisać jako układ dynamiczny

$$\begin{cases} \dot{\mathbf{v}} &= \mathbf{a}(\mathbf{r}) \\ \dot{\mathbf{r}} &= \mathbf{v} \end{cases}$$

mamy

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{a}(\mathbf{r}_n)$$

$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\mathbf{v}_n$$

Praktycznie nie zachowuje energii i nie jest odwracalny w czasie (tzn. gdy $\mathbf{y}_n \xrightarrow{+h} \mathbf{y}_{n+1} \xrightarrow{-h} \mathbf{y}'_n$ to $\mathbf{y}_n \neq \mathbf{y}'_n$), więc dla wszelkich faktycznych zastosowań (całkowanie długoterminowe) jest bezużyteczny (np. prowadzi do orbit spiralnych zamiast kołowych czy wzrostu amplitudy oscylatora harmonicznego z czasem).
Stanowi jednak dobrą bazę dydaktyczną dla porównań itd.

Metoda Eulera-Cromera

Prędkość jest aktualizowana w pierwszej kolejności i rezultat jest stosowany w aktualizacji położenia:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{a}(\mathbf{r}_n)$$

$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\mathbf{v}_{n+1}$$

Energia pozostaje ograniczona i oscyluje wokół dokładnej wartości (co jest istotne dla układów hamiltonowskich; metoda ta jest symplektyczna — zob. slajd 15); metoda ta nie jest jednak odwracalna w czasie.

Metoda Eulera-Cromera

Prędkość jest aktualizowana w pierwszej kolejności i rezultat jest stosowany w aktualizacji położenia:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{a}(\mathbf{r}_n)$$

$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\mathbf{v}_{n+1}$$

Energia pozostaje ograniczona i oscyluje wokół dokładnej wartości (co jest istotne dla układów hamiltonowskich; metoda ta jest symplektyczna — zob. slajd 15); metoda ta nie jest jednak odwracalna w czasie.

Równie uprawnionym matematycznie, choć rzadziej stosowanym, jest wariant z aktualizacją położenia w pierwszej kolejności:

$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\mathbf{v}_n$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{a}(\mathbf{r}_{n+1})$$

Będziemy stosować wariant pierwszy, tj. z aktualizacją prędkości w pierwszej kolejności.

Metoda Rungego-Kutty rzędu 4 (RK4)

$$
\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}(t_n, \mathbf{y}_n), \\
\mathbf{k}_2 &= \mathbf{f}\left(t_n + \tfrac{h}{2},\ \mathbf{y}_n + \tfrac{h}{2}\mathbf{k}_1\right), \\
\mathbf{k}_3 &= \mathbf{f}\left(t_n + \tfrac{h}{2},\ \mathbf{y}_n + \tfrac{h}{2}\mathbf{k}_2\right), \\
\mathbf{k}_4 &= \mathbf{f}\left(t_n + h,\ \mathbf{y}_n + h\mathbf{k}_3\right), \\
\mathbf{y}_{n+1} &= \mathbf{y}_n + \tfrac{h}{6}\left(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\right), \\
t_{n+1} &= t_n + h.
\end{aligned}
$$

Jest względnie stabilna dla małych h (tzn. błędy numeryczne nie nawarstwiają się lecz pozostają ograniczone), ale nie jest symplektyczna zatem nie zachowuje energii przy dłuższych całkowaniach oraz nie jest odwracalna w czasie.

Algorytm Verleta (*velocity-Verlet*)

Dla układów równań drugiego rzędu bez jawnej zależności od czasu:

$$\ddot{\mathbf{r}} = \mathbf{a}(\mathbf{r})$$

gdzie $\mathbf{v} = \dot{\mathbf{r}}$:

$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\mathbf{v}_n + \tfrac{1}{2}h^2\,\mathbf{a}(\mathbf{r}_n),$$
$$\mathbf{v}_{n+1} = \mathbf{v}_n + \tfrac{h}{2}\Big[\mathbf{a}(\mathbf{r}_n) + \mathbf{a}(\mathbf{r}_{n+1})\Big].$$

Jest to algorytm symplektyczny, dobrze zachowuje energię w toku całkowania oraz jest odwracalny w czasie.

Algorytm *leapfrog* (żabiego skoku, skokowy)

Niech też $\ddot{\mathbf{r}} = \mathbf{a}(\mathbf{r})$.

$$\mathbf{v}_{n+\frac{1}{2}} = \mathbf{v}_n + \tfrac{h}{2}\,\mathbf{a}(\mathbf{r}_n),$$
$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\,\mathbf{v}_{n+\frac{1}{2}},$$
$$\mathbf{v}_{n+1} = \mathbf{v}_{n+\frac{1}{2}} + \tfrac{h}{2}\,\mathbf{a}(\mathbf{r}_{n+1}).$$

Również jest symplektyczny, mniej dokładny niż RK4 ale lepiej zachowuje energię w czasie długich całkowań; jest też odwracalny w czasie.

Algorytm *leapfrog* (żabiego skoku, skokowy)

Niech też $\ddot{\mathbf{r}} = \mathbf{a}(\mathbf{r})$.

$$\mathbf{v}_{n+\frac{1}{2}} = \mathbf{v}_n + \tfrac{h}{2}\,\mathbf{a}(\mathbf{r}_n),$$
$$\mathbf{r}_{n+1} = \mathbf{r}_n + h\,\mathbf{v}_{n+\frac{1}{2}},$$
$$\mathbf{v}_{n+1} = \mathbf{v}_{n+\frac{1}{2}} + \tfrac{h}{2}\,\mathbf{a}(\mathbf{r}_{n+1}).$$

Również jest symplektyczny, mniej dokładny niż RK4 ale lepiej zachowuje energię w czasie długich całkowań; jest też odwracalny w czasie.

Ideą jest aktualizacja pozycji i prędkości w naprzemiennych chwilach, stąd „żabie skoki". Krok h musi być stały by zachować symplektyczność i stabilność, jednak zmienny h bywa przydatny przy całkowaniu układów N ciał — trzeba wtedy stosować specjalne metody adaptatywne.

Algorytm *leapfrog* (żabiego skoku, skokowy)

Leapfrog i velocity-Verlet są algebraicznie równoważne

- Jeśli w algorytmie *leapfrog* (slajd 7) wstawimy wyrażenie na $\mathbf{v}_{n+\frac{1}{2}}$ do wyrażeń na $\mathbf{r}_{n+1}$ oraz $\mathbf{v}_{n+1}$, to dostaniemy **dokładnie** sformułowanie algorytmu *velocity-Verlet* (slajd 6).
- Zatem te dwa algorytmy, jeśli weźmiemy dla obu taki sam h, dadzą (z dokładnością do błędów numerycznych) takie same wyniki.
- Różnica w implementacji sprowadza się do sposobu organizacji danych w ramach tego samego schematu symplektycznego. W zastosowaniach mechaniki nieba nie należy się jednak spodziewać skrajnie różnych ewolucji orbit.
- W *velocity-Verlet* przechowujemy stan układu jako $(\mathbf{r}_n, \mathbf{v}_n)$ prowadzący do stanu $(\mathbf{r}_{n+1}, \mathbf{v}_{n+1})$; w *leapfrog* stan $(\mathbf{r}_n, \mathbf{v}_{n-\frac{1}{2}})$ prowadzi do stanu $(\mathbf{r}_{n+1}, \mathbf{v}_{n+\frac{1}{2}})$, co wymaga dodatkowej interpolacji by wyznaczyć np. $E_n = E(t_n)$. W zasadzie różnica w obu algorytmach jest historyczna.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Może być stosowany do $\dot{\mathbf{y}}(t) = \mathbf{f}(t, \mathbf{y})$:

$$
\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}\big[t_n, y_n\big], \\
\mathbf{k}_2 &= \mathbf{f}\Big[t_n + \tfrac{1}{4}h,\ \mathbf{y}_n + \tfrac{1}{4}h\,\mathbf{k}_1\Big], \\
\mathbf{k}_3 &= \mathbf{f}\Big[t_n + \tfrac{3}{8}h,\ \mathbf{y}_n + h\big(\tfrac{3}{32}\mathbf{k}_1 + \tfrac{9}{32}\mathbf{k}_2\big)\Big], \\
\mathbf{k}_4 &= \mathbf{f}\Big[t_n + \tfrac{12}{13}h,\ \mathbf{y}_n + h\big(\tfrac{1932}{2197}\mathbf{k}_1 - \tfrac{7200}{2197}\mathbf{k}_2 + \tfrac{7296}{2197}\mathbf{k}_3\big)\Big], \\
\mathbf{k}_5 &= \mathbf{f}\Big[t_n + h,\ \mathbf{y}_n + h\big(\tfrac{439}{216}\mathbf{k}_1 - 8\mathbf{k}_2 + \tfrac{3680}{513}\mathbf{k}_3 - \tfrac{845}{4104}\mathbf{k}_4\big)\Big], \\
\mathbf{k}_6 &= \mathbf{f}\Big[t_n + \tfrac{1}{2}h,\ \mathbf{y}_n + h\big(-\tfrac{8}{27}\mathbf{k}_1 + 2\mathbf{k}_2 - \tfrac{3544}{2565}\mathbf{k}_3 + \tfrac{1859}{4104}\mathbf{k}_4 - \tfrac{11}{40}\mathbf{k}_5\big)\Big].
\end{aligned}
$$

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Może być stosowany do $\dot{\mathbf{y}}(t) = \mathbf{f}(t, \mathbf{y})$:

$$
\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}\big[t_n, y_n\big], \\
\mathbf{k}_2 &= \mathbf{f}\Big[t_n + \tfrac{1}{4}h,\ \mathbf{y}_n + \tfrac{1}{4}h\,\mathbf{k}_1\Big], \\
\mathbf{k}_3 &= \mathbf{f}\Big[t_n + \tfrac{3}{8}h,\ \mathbf{y}_n + h\big(\tfrac{3}{32}\mathbf{k}_1 + \tfrac{9}{32}\mathbf{k}_2\big)\Big], \\
\mathbf{k}_4 &= \mathbf{f}\Big[t_n + \tfrac{12}{13}h,\ \mathbf{y}_n + h\big(\tfrac{1932}{2197}\mathbf{k}_1 - \tfrac{7200}{2197}\mathbf{k}_2 + \tfrac{7296}{2197}\mathbf{k}_3\big)\Big], \\
\mathbf{k}_5 &= \mathbf{f}\Big[t_n + h,\ \mathbf{y}_n + h\big(\tfrac{439}{216}\mathbf{k}_1 - 8\mathbf{k}_2 + \tfrac{3680}{513}\mathbf{k}_3 - \tfrac{845}{4104}\mathbf{k}_4\big)\Big], \\
\mathbf{k}_6 &= \mathbf{f}\Big[t_n + \tfrac{1}{2}h,\ \mathbf{y}_n + h\big(-\tfrac{8}{27}\mathbf{k}_1 + 2\mathbf{k}_2 - \tfrac{3544}{2565}\mathbf{k}_3 + \tfrac{1859}{4104}\mathbf{k}_4 - \tfrac{11}{40}\mathbf{k}_5\big)\Big].
\end{aligned}
$$

(Współczynniki z tzw. tablicy Butchera.)

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Rozwiązania dwóch rzędów — piątego [rząd $= p \Leftrightarrow$ błąd skaluje się z krokiem h jak $\mathcal{O}\left(h^{p+1}\right)$]:

$$\mathbf{y}_{n+1}^{(5)} = \mathbf{y}_n + h\left(\frac{16}{135}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{6656}{12825}\mathbf{k}_3 + \frac{28561}{56430}\mathbf{k}_4 - \frac{9}{50}\mathbf{k}_5 + \frac{2}{55}\mathbf{k}_6\right)$$

i czwartego:

$$\mathbf{y}_{n+1}^{(4)} = \mathbf{y}_n + h\left(\frac{25}{216}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{1408}{2565}\mathbf{k}_3 + \frac{2197}{4104}\mathbf{k}_4 - \frac{1}{5}\mathbf{k}_5 + 0\cdot\mathbf{k}_6\right)$$

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Rozwiązania dwóch rzędów — piątego [rząd $= p \Leftrightarrow$ błąd skaluje się z krokiem h jak $\mathcal{O}\left(h^{p+1}\right)$]:

$$\mathbf{y}_{n+1}^{(5)} = \mathbf{y}_n + h\left(\frac{16}{135}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{6656}{12825}\mathbf{k}_3 + \frac{28561}{56430}\mathbf{k}_4 - \frac{9}{50}\mathbf{k}_5 + \frac{2}{55}\mathbf{k}_6\right)$$

i czwartego:

$$\mathbf{y}_{n+1}^{(4)} = \mathbf{y}_n + h\left(\frac{25}{216}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{1408}{2565}\mathbf{k}_3 + \frac{2197}{4104}\mathbf{k}_4 - \frac{1}{5}\mathbf{k}_5 + 0\cdot\mathbf{k}_6\right)$$

- Współczynniki w tablicy Butchera są tak dobrane, by wektory $\mathbf{k}_i$ były dzielone między rozwiązania rzędów 4. i 5. $\Rightarrow$ liczone tylko raz na krok.
- Za $\mathbf{y}_n$ bierzemy wartość z poprzedniego kroku — czyli *de facto* $\mathbf{y}_n^{(5)}$ (po spełnieniu kryteriów i ustaleniu h — por. dalej); drugie rozw., $\mathbf{y}_n^{(4)}$, jest konstruowane w oparciu o te same $\mathbf{k}_i$ i służy do oszacowania błędu lokalnego dla oceny wartości h — zob. następne slajdy.
- **Czyli**: w kolejnym, $(n+1)$-szym kroku używa się tego rozw. wyższego rzędu $\mathbf{y}_n$, które zostało uznane za wystarczająco dokładne — krótko mówiąc, $\mathbf{y}_n = \mathbf{y}_n^{(5)}$.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Rozwiązania dwóch rzędów — piątego [rząd $= p \Leftrightarrow$ błąd skaluje się z krokiem h jak $\mathcal{O}\left(h^{p+1}\right)$]:

$$\mathbf{y}_{n+1}^{(5)} = \mathbf{y}_n + h\left(\frac{16}{135}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{6656}{12825}\mathbf{k}_3 + \frac{28561}{56430}\mathbf{k}_4 - \frac{9}{50}\mathbf{k}_5 + \frac{2}{55}\mathbf{k}_6\right)$$

i czwartego:

$$\mathbf{y}_{n+1}^{(4)} = \mathbf{y}_n + h\left(\frac{25}{216}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{1408}{2565}\mathbf{k}_3 + \frac{2197}{4104}\mathbf{k}_4 - \frac{1}{5}\mathbf{k}_5 + 0\cdot\mathbf{k}_6\right)$$

- Współczynniki w tablicy Butchera są tak dobrane, by wektory $\mathbf{k}_i$ były dzielone między rozwiązania rzędów 4. i 5. $\Rightarrow$ liczone tylko raz na krok.
- Za $\mathbf{y}_n$ bierzemy wartość z poprzedniego kroku — czyli *de facto* $\mathbf{y}_n^{(5)}$ (po spełnieniu kryteriów i ustaleniu h — por. dalej); drugie rozw., $\mathbf{y}_n^{(4)}$, jest konstruowane w oparciu o te same $\mathbf{k}_i$ i służy do oszacowania błędu lokalnego dla oceny wartości h — zob. następne slajdy.
- **Czyli**: w kolejnym, $(n+1)$-szym kroku używa się tego rozw. wyższego rzędu $\mathbf{y}_n$, które zostało uznane za wystarczająco dokładne — krótko mówiąc, $\mathbf{y}_n = \mathbf{y}_n^{(5)}$.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Rozwiązania dwóch rzędów — piątego [rząd $= p \Leftrightarrow$ błąd skaluje się z krokiem h jak $\mathcal{O}\left(h^{p+1}\right)$]:

$$\mathbf{y}_{n+1}^{(5)} = \mathbf{y}_n + h\left(\frac{16}{135}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{6656}{12825}\mathbf{k}_3 + \frac{28561}{56430}\mathbf{k}_4 - \frac{9}{50}\mathbf{k}_5 + \frac{2}{55}\mathbf{k}_6\right)$$

i czwartego:

$$\mathbf{y}_{n+1}^{(4)} = \mathbf{y}_n + h\left(\frac{25}{216}\mathbf{k}_1 + 0\cdot\mathbf{k}_2 + \frac{1408}{2565}\mathbf{k}_3 + \frac{2197}{4104}\mathbf{k}_4 - \frac{1}{5}\mathbf{k}_5 + 0\cdot\mathbf{k}_6\right)$$

- Współczynniki w tablicy Butchera są tak dobrane, by wektory $\mathbf{k}_i$ były dzielone między rozwiązania rzędów 4. i 5. $\Rightarrow$ liczone tylko raz na krok.
- Za $\mathbf{y}_n$ bierzemy wartość z poprzedniego kroku — czyli *de facto* $\mathbf{y}_n^{(5)}$ (po spełnieniu kryteriów i ustaleniu h — por. dalej); drugie rozw., $\mathbf{y}_n^{(4)}$, jest konstruowane w oparciu o te same $\mathbf{k}_i$ i służy do oszacowania błędu lokalnego dla oceny wartości h — zob. następne slajdy.
- **Czyli**: w kolejnym, $(n+1)$-szym kroku używa się tego rozw. wyższego rzędu $\mathbf{y}_n$, które zostało uznane za wystarczająco dokładne — krótko mówiąc, $\mathbf{y}_n = \mathbf{y}_n^{(5)}$.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Błąd lokalny jest estymowany jako

$$\Delta \mathbf{y} = \mathbf{y}_{n+1}^{(5)} - \mathbf{y}_{n+1}^{(4)},$$

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Błąd lokalny jest estymowany jako

$$\Delta \mathbf{y} = \mathbf{y}_{n+1}^{(5)} - \mathbf{y}_{n+1}^{(4)},$$

zaś estymator błędu znormalizowanego to

$$\Delta y_{\mathrm{norm}} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\frac{\Delta y_i}{\varepsilon_{\mathrm{abs},i} + \varepsilon_{\mathrm{rel}}\ \max(|y_{n,i}|, |y_{n+1,i}|)}\right)^2}$$

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

gdzie tolerancje bezwzględna i względna są w ogólności definiowane jako:

$$\varepsilon_{\mathrm{abs},i} = |y_{\mathrm{computed},i} - y_{\mathrm{true},i}|$$

$$\varepsilon_{\mathrm{rel}} = \frac{|\mathbf{y}_{\mathrm{computed}} - \mathbf{y}_{\mathrm{true}}|}{|\mathbf{y}_{\mathrm{true}}|}$$

Można przyjąć, że $\mathbf{y}_{\mathrm{computed}}$ to $\mathbf{y}_{n+1}^{(5)}$, zaś $\mathbf{y}_{\mathrm{true}}$ jest (nieznanym) dokładnym rozwiązaniem $\mathbf{y}(t_{n+1})$.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Dla praktycznych aspektów mechaniki nieba Układu Słonecznego (w jednostkach AU):

- $\varepsilon_{\mathrm{rel}} = 10^{-8}$
- $\varepsilon_{\mathrm{abs},i} = 10^{-10} \div 10^{-12}$

i to te wielkości (czyli parametry **wprowadzone przez użytkownika — dostosowane do konkretnego zagadnienia**) wchodzą do wyrażenia na Δy_{norm} na poprzednim slajdzie.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Dla praktycznych aspektów mechaniki nieba Układu Słonecznego (w jednostkach AU):

- $\varepsilon_{\mathrm{rel}} = 10^{-8}$
- $\varepsilon_{\mathrm{abs},i} = 10^{-10} \div 10^{-12}$

i to te wielkości (czyli parametry **wprowadzone przez użytkownika — dostosowane do konkretnego zagadnienia**) wchodzą do wyrażenia na Δy_{norm} na poprzednim slajdzie.

Caveat emptor!

Te tolerancje oczywiście należy dostosować do konkretnego zagadnienia — egzotyczne układy egzoplanetarne mogą wymagać przyjęcia znacząco innych wartości. Powyższe wartości są podane jako przykładowe punkty wyjścia. Zmiana tych wartości może się okazać konieczna nawet dla US.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Po kilku orbitach sprawdzić zachowanie E w celach diagnostycznych.[1] Jeśli:

- dryft jest zbyt duży, zmniejszyć $\varepsilon_{\mathrm{rel}}$ i/lub $\varepsilon_{\mathrm{abs},i}$;
- h jest zbyt mały to tolerancje są zbyt małe (można zwiększyć);
- h jest zbyt duży to tolerancje są za duże (można zmniejszyć);
- $\Delta y_{\mathrm{norm}} \leqslant 1$ — przyjąć h oraz $\mathbf{y}_{n+1} = \mathbf{y}^{(5)}_{n+1}$;
- $\Delta y_{\mathrm{norm}} > 1$ — odrzucić h, przyjąć mniejszy i sprawdzić ponownie.

[1] W praktyce badawczej należałoby monitorować wszystkie wielkości zachowywane: L, A itd., ale też błędy położenia, prędkości, fazy itp.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Po kilku orbitach sprawdzić zachowanie E w celach diagnostycznych.[1] Jeśli:

- dryft jest zbyt duży, zmniejszyć $\varepsilon_{\mathrm{rel}}$ i/lub $\varepsilon_{\mathrm{abs},i}$;
- h jest zbyt mały to tolerancje są zbyt małe (można zwiększyć);
- h jest zbyt duży to tolerancje są za duże (można zmniejszyć);
- $\Delta y_{\mathrm{norm}} \leqslant 1$ — przyjąć h oraz $\mathbf{y}_{n+1} = \mathbf{y}^{(5)}_{n+1}$;
- $\Delta y_{\mathrm{norm}} > 1$ — odrzucić h, przyjąć mniejszy i sprawdzić ponownie.

Aktualizacja kroku wg reguły (dla RKF45 jest $p = 4$)

$$h_{\mathrm{new}} = f_{\mathrm{safety}} \cdot h_{\mathrm{old}} \cdot \left(\frac{1}{\Delta y_{\mathrm{norm}}}\right)^{1/(p+1)}$$

gdzie $f_{\mathrm{safety}} \approx 0.8 \div 0.95$ zapobiega zbyt agresywnym zmianom kroku.

[1] W praktyce badawczej należałoby monitorować wszystkie wielkości zachowywane: L, A itd., ale też błędy położenia, prędkości, fazy itp.

Algorytm Rungego-Kutty-Fehlberga 4(5) (RKF45)

Po kilku orbitach sprawdzić zachowanie E w celach diagnostycznych.[1] Jeśli:

- dryft jest zbyt duży, zmniejszyć $\varepsilon_{\mathrm{rel}}$ i/lub $\varepsilon_{\mathrm{abs},i}$;
- h jest zbyt mały to tolerancje są zbyt małe (można zwiększyć);
- h jest zbyt duży to tolerancje są za duże (można zmniejszyć);
- $\Delta y_{\mathrm{norm}} \leqslant 1$ — przyjąć h oraz $\mathbf{y}_{n+1} = \mathbf{y}^{(5)}_{n+1}$;
- $\Delta y_{\mathrm{norm}} > 1$ — odrzucić h, przyjąć mniejszy i sprawdzić ponownie.

Aktualizacja kroku wg reguły (dla RKF45 jest $p = 4$)

$$h_{\mathrm{new}} = f_{\mathrm{safety}} \cdot h_{\mathrm{old}} \cdot \left(\frac{1}{\Delta y_{\mathrm{norm}}}\right)^{1/(p+1)}$$

gdzie $f_{\mathrm{safety}} \approx 0.8 \div 0.95$ zapobiega zbyt agresywnym zmianom kroku.
Nie jest symplektyczny ani odwracalny w czasie; jego zaletą jest za to adaptacyjny dobór kroku h poprzez kontrolę błędu.
Spośród przedstawionych algorytmów RKF45 zazwyczaj najlepiej się sprawdza w mechanice nieba.

[1] W praktyce badawczej należałoby monitorować wszystkie wielkości zachowywane: L, A itd., ale też błędy położenia, prędkości, fazy itp.

Symplektyczność (pobieżnie)

Mając hamiltonian $\mathcal{H}(\mathbf{p},\mathbf{q})$, równania ruchu są postaci

$$\dot{\mathbf{p}} = -\frac{\partial \mathcal{H}(\mathbf{p},\mathbf{q})}{\partial \mathbf{q}}, \quad \dot{\mathbf{q}} = \frac{\partial \mathcal{H}(\mathbf{p},\mathbf{q})}{\partial \mathbf{p}}$$

Integrator $\mathbf{y}_{n+1} = \mathbf{\Phi}_h(\mathbf{y}_n)$ jest symplektyczny jeśli potok fazowy układu hamiltonowskiego $\mathbf{y} \to \mathbf{\Phi}_h(\mathbf{y})$ jest odwzorowaniem symplektycznym.

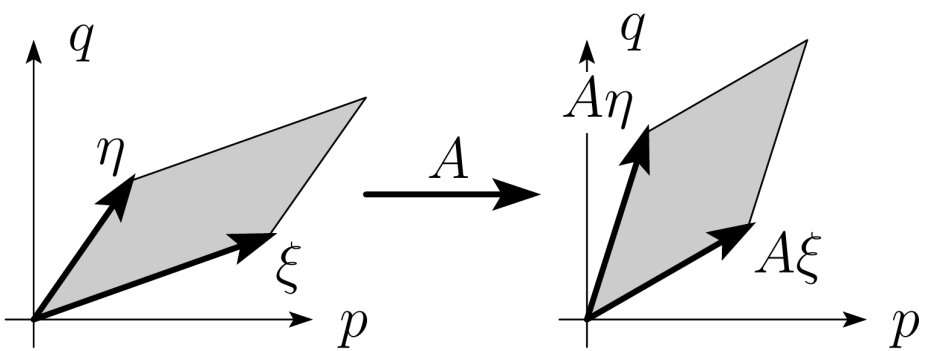
Symplektyczność (pobieżnie)

Mając hamiltonian $\mathcal{H}(\mathbf{p},\mathbf{q})$, równania ruchu są postaci

$$\dot{\mathbf{p}} = -\frac{\partial \mathcal{H}(\mathbf{p},\mathbf{q})}{\partial \mathbf{q}}, \quad \dot{\mathbf{q}} = \frac{\partial \mathcal{H}(\mathbf{p},\mathbf{q})}{\partial \mathbf{p}}$$

Integrator $\mathbf{y}_{n+1} = \mathbf{\Phi}_h(\mathbf{y}_n)$ jest symplektyczny jeśli potok fazowy układu hamiltonowskiego $\mathbf{y} \to \mathbf{\Phi}_h(\mathbf{y})$ jest odwzorowaniem symplektycznym.

Odwzorowanie liniowe A jest symplektyczne, jeśli 2D równoległobok w przestrzeni fazowej pod wpływem potoku zachowuje powierzchnię, tj. $A^\intercal J A = J$, gdzie $J = \begin{pmatrix} 0 & \mathbf{1} \\ -\mathbf{1} & 0 \end{pmatrix} \Leftrightarrow \det A = 1$ dla 2D prz. fazowej.



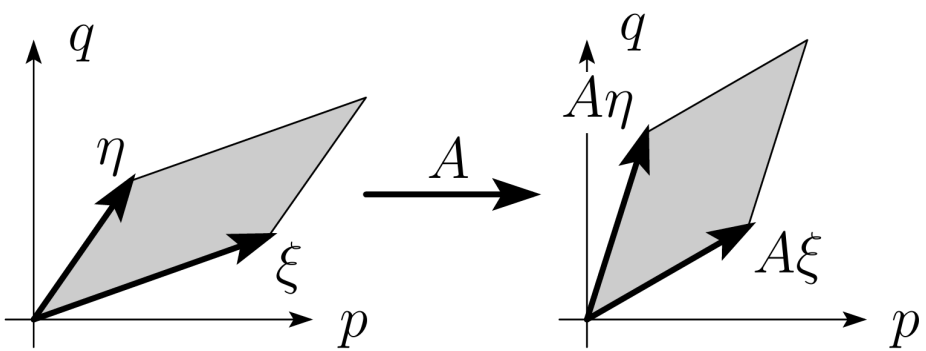
Symplektyczność (pobieżnie)

Mając hamiltonian $\mathcal{H}(\mathbf{p},\mathbf{q})$, równania ruchu są postaci

$$\dot{\mathbf{p}} = -\frac{\partial \mathcal{H}(\mathbf{p},\mathbf{q})}{\partial \mathbf{q}}, \quad \dot{\mathbf{q}} = \frac{\partial \mathcal{H}(\mathbf{p},\mathbf{q})}{\partial \mathbf{p}}$$

Integrator $\mathbf{y}_{n+1} = \mathbf{\Phi}_h(\mathbf{y}_n)$ jest symplektyczny jeśli potok fazowy układu hamiltonowskiego $\mathbf{y} \to \mathbf{\Phi}_h(\mathbf{y})$ jest odwzorowaniem symplektycznym.

Odwzorowanie liniowe A jest symplektyczne, jeśli 2D równoległobok w przestrzeni fazowej pod wpływem potoku zachowuje powierzchnię, tj. $A^\intercal J A = J$, gdzie $J = \begin{pmatrix} 0 & \mathbf{1} \\ -\mathbf{1} & 0 \end{pmatrix} \Leftrightarrow \det A = 1$ dla 2D prz. fazowej.



Odwzorowania nieliniowe są symplektyczne jeśli ich macierze Jacobiego $\mathbf{D}\mathbf{\Phi}_h$ są symplektyczne. Geometrycznie: powierzchnię można przybliżać sumą mnogościową 2D równoległoboków.

Symplektyczność (pobieżnie) — przykład (Euler-Cromer)

Jeden stopień swobody ≡ 2D przestrzeń fazowa

$$\begin{pmatrix} v' \\ r' \end{pmatrix} = \mathbf{\Phi}_h(v,r) = \begin{pmatrix} v + ha(r) \\ r + hv + h^2 a(r) \end{pmatrix}$$

$$\mathbf{D}\mathbf{\Phi}_h(v,r) = \frac{\partial(v',r')}{\partial(v,r)} = \begin{pmatrix} 1 & h\frac{\mathrm{d}a}{\mathrm{d}r} \\ h & 1 + h^2\frac{\mathrm{d}a}{\mathrm{d}r} \end{pmatrix}$$

Jawnym rachunkiem sprawdzamy, że $[\mathbf{D}\mathbf{\Phi}_h(v,r)]^\mathsf{T}\, J \mathbf{D}\mathbf{\Phi}_h(v,r) = J$, co dla odwz. 2D jest równoważne warunkowi $\det \mathbf{D}\mathbf{\Phi}_h = 1$, który oczywiście też jest spełniony, zatem metoda Eulera-Cromera w istocie jest symplektyczne.

Symplektyczność (pobieżnie) — przykład (Euler-Cromer)

Jeden stopień swobody ≡ 2D przestrzeń fazowa

$$\begin{pmatrix} v' \\ r' \end{pmatrix} = \mathbf{\Phi}_h(v,r) = \begin{pmatrix} v + ha(r) \\ r + hv + h^2 a(r) \end{pmatrix}$$

$$\mathbf{D\Phi}_h(v,r) = \frac{\partial(v',r')}{\partial(v,r)} = \begin{pmatrix} 1 & h\frac{\mathrm{d}a}{\mathrm{d}r} \\ h & 1 + h^2\frac{\mathrm{d}a}{\mathrm{d}r} \end{pmatrix}$$

Jawnym rachunkiem sprawdzamy, że $[\mathbf{D\Phi}_h(v,r)]^\mathsf{T}\, J \mathbf{D\Phi}_h(v,r) = J$, co dla odwz. 2D jest równoważne warunkowi $\det \mathbf{D\Phi}_h = 1$, który oczywiście też jest spełniony, zatem metoda Eulera-Cromera w istocie jest symplektyczne.

Ponadto, jawnym rachunkiem

$$\begin{pmatrix} v'' \\ r'' \end{pmatrix} = \mathbf{\Phi}_{-h}(v',r') = \mathbf{\Phi}_{-h} \circ \mathbf{\Phi}_h(v,r) \neq \begin{pmatrix} v \\ r \end{pmatrix}$$

zatem nie jest odwracalna w czasie.
(Warunek odwracalności można zapisać jako $\mathbf{\Phi}_{-h} \circ \mathbf{\Phi}_h = \mathbf{1}$.)

Zad. 0' (*Tablica*)

Zbadać symplektyczność oraz odwracalność w czasie algorytmów:

1. Eulera
2. *velocity-Verlet*

Zad. 2 (**raport**) — całkowanie numeryczne równań ruchu

(A) Zaimplementować algorytmy:

1. Eulera,
2. Eulera-Cromera,
3. RK4,
4. velocity-Verleta lub *leapfrog*,
5. RKF45.

Zad. 2 (**raport**) — całkowanie numeryczne równań ruchu

(A) Zaimplementować algorytmy:

1. Eulera,
2. Eulera-Cromera,
3. RK4,
4. velocity-Verleta lub *leapfrog*,
5. RKF45.

(B) Porównać ich działanie dla równań:

1. $\dot{y}(t) = y(t)$
2. $\ddot{y}(t) + y(t) = 0$

Porównać z rozw. analitycznymi.

Przyjąć nietrywialne war. pocz.

UWAGA! $\dot{y} = y \Leftrightarrow \ddot{y} = \dot{y} = y$ gdy $y(0) = \dot{y}(0)$.

Zad. 2 (**raport**) — całkowanie numeryczne równań ruchu

(A) Zaimplementować algorytmy:

1. Eulera,
2. Eulera-Cromera,
3. RK4,
4. velocity-Verleta lub *leapfrog*,
5. RKF45.

(B) Porównać ich działanie dla równań:

1. $\dot{y}(t) = y(t)$
2. $\ddot{y}(t) + y(t) = 0$

Porównać z rozw. analitycznymi. Przyjąć nietrywialne war. pocz.
UWAGA! $\dot{y} = y \Leftrightarrow \ddot{y} = \dot{y} = y$ gdy $y(0) = \dot{y}(0)$.

(C) Zapisać równanie ruchu zagadnienia 2 ciał, dla orbity kołowej $e = 0$ z wielką półosią $a_{\mathrm{orb}} = 1$, we współ. kartezjańskich. Rozwiązać przy użyciu zaimplementowanych algorytmów. Przyjąć warunki pocz.: $x(0) = 1$, $y(0) = 0$, $v_x(0) = 0$, $v_y(0) = 1$. Rozważyć różne kroki h. Zbadać stałość energii E oraz wektorów $\mathbf{L}$ i $\mathbf{A}$ w czasie. Sprawdzić stabilność kształtu orbity w czasie.

Zad. 2 (**raport**) — całkowanie numeryczne równań ruchu

(A) Zaimplementować algorytmy:

1. Eulera,
2. Eulera-Cromera,
3. RK4,
4. velocity-Verleta lub *leapfrog*,
5. RKF45.

(B) Porównać ich działanie dla równań:

1. $\dot{y}(t) = y(t)$
2. $\ddot{y}(t) + y(t) = 0$

Porównać z rozw. analitycznymi. Przyjąć nietrywialne war. pocz. **UWAGA!** $\dot{y} = y \Leftrightarrow \ddot{y} = \dot{y} = y$ gdy $y(0) = \dot{y}(0)$.

(C) Zapisać równanie ruchu zagadnienia 2 ciał, dla orbity kołowej $e = 0$ z wielką półosią $a_{\mathrm{orb}} = 1$, we współ. kartezjańskich. Rozwiązać przy użyciu zaimplementowanych algorytmów. Przyjąć warunki pocz.: $x(0) = 1$, $y(0) = 0$, $v_x(0) = 0$, $v_y(0) = 1$. Rozważyć różne kroki h. Zbadać stałość energii E oraz wektorów $\mathbf{L}$ i $\mathbf{A}$ w czasie. Sprawdzić stabilność kształtu orbity w czasie.

(D) To samo co w (C), ale dla $e = 0.1$. Przyjąć $x(0) = a_{\mathrm{orb}}(1 - e)$, $y(0) = 0$. Wyliczyć $\mathbf{v}(0)$ z całki energii (całki siły żywej, *vis-viva*) — rozłożyć na stosowne składowe.