Pracownia astrofizyki teoretycznej 1

Wykład 2

Mariusz Tarnopolski

Instytut Astronomii UMK

PATEO 1 ©2025

Metody numeryczne rozwiązywania równań różniczkowych

Rozwiązanie równania postaci

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

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:

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

Metoda Eulera (*explicit Euler*)

Ekstrapolacja liniowa ze stałym krokiem h:

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

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

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

mamy

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

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

Metoda Eulera (*explicit Euler*)

Ekstrapolacja liniowa ze stałym krokiem h:

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

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

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

mamy

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

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

Niemal nie zachowuje energii i nie jest odwracalny w czasie, więc dla wszelkich praktycznych zastosowań bezużyteczny (np. spirale zamiast orbit kołowych).
Stanowi jednak dobrą bazę dydaktyczną dla porównań itd.

Metoda Eulera-Cromera — symplektyczna

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

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

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

Zachowuje energię w przypadku rozwiązań oscylacyjnych ale nie jest odwracalna.

Metoda Eulera-Cromera — symplektyczna

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

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

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

Zachowuje energię w przypadku rozwiązań oscylacyjnych ale nie jest odwracalna.
Równie uprawnionym matematycznie, choć rzadziej stosowanym, jest wariant z aktualizacją położenia w pierwszej kolejności:

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

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

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 stabilny dla małych h (tzn. błędy numeryczne nie nawarstwiają się lecz pozostają ograniczone), ale nie zachowuje energii (co jest istotne dla układów hamiltonowskich) w dłuższych skalach czasu oraz nie jest odwracalny 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 (zob. dalej), dobrze zachowuje energię w toku całkowania oraz jest odwracalny.

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 niż algorytm Verleta; jest odwracalny.

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 niż algorytm Verleta; jest odwracalny.

Ideą jest aktualizacja pozycji i prędkości w naprzemiennych chwilach, stąd „żabie skoki". Krok h musi być stały by zachować stabilność, jednak zmienny h jest przydatne przy całkowaniu układów N-ciał.

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 kroku $n+1$ 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 kroku $n+1$ 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 kroku $n+1$ 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 obliczany jako

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

zaś błąd znormalizowany:

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

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 $\mathbf{y}\left(t_{n+1}\right)$.

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

Dla praktycznych aspektów mechaniki nieba (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)

Po kilku orbitach sprawdzić zachowanie energii; jeśli:

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

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

Po kilku orbitach sprawdzić zachowanie energii; jeśli:

- dryft jest zbyt duży, zmniejszyć $\varepsilon_{\mathrm{rel}}$ i/lub $\varepsilon_{\mathrm{abs},i}$;
- h jest zbyt mały, tolerancje są zbyt duże;
- h jest zbyt duży, tolerancje są za małe;
- $\Delta y_{\mathrm{norm}} \leqslant 1$ — 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

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

gdzie $f_{\mathrm{safety}} \approx 0.8 \div 0.95$ zapobiega zbyt agresywnym zmianom kroku, zaś dla RKF45 jest $p = 4$.

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

Po kilku orbitach sprawdzić zachowanie energii; jeśli:

- dryft jest zbyt duży, zmniejszyć $\varepsilon_{\mathrm{rel}}$ i/lub $\varepsilon_{\mathrm{abs},i}$;
- h jest zbyt mały, tolerancje są zbyt duże;
- h jest zbyt duży, tolerancje są za małe;
- $\Delta y_{\mathrm{norm}} \leqslant 1$ — 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

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

gdzie $f_{\mathrm{safety}} \approx 0.8 \div 0.95$ zapobiega zbyt agresywnym zmianom kroku, zaś dla RKF45 jest $p = 4$.
Jest symplektyczny, ale nie odwracalny w czasie.

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, dla dostatecznie małych kroków h, odwzorowaniem symplektycznym.

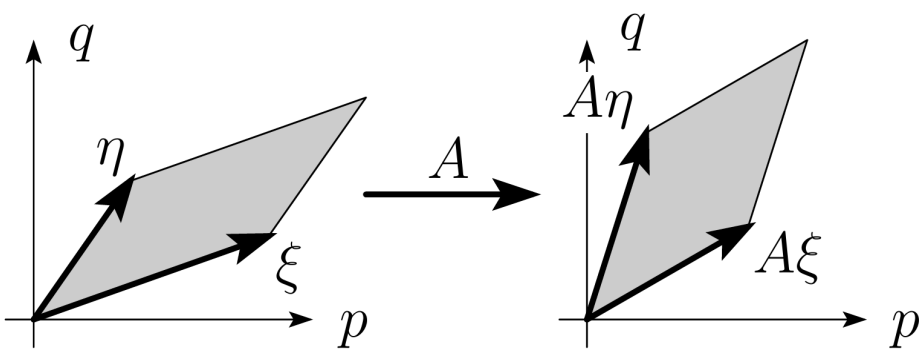
Symplektyczność (pobieżnie)

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, dla dostatecznie małych kroków h, 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 $J = \begin{pmatrix} 0 & \mathbb{1} \\ -\mathbb{1} & 0 \end{pmatrix} \Leftrightarrow \det A = 1.$



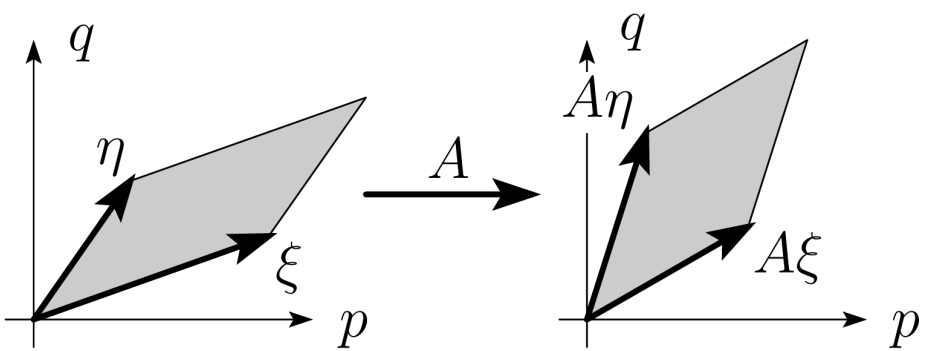
Symplektyczność (pobieżnie)

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, dla dostatecznie małych kroków h, 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 $J = \begin{pmatrix} 0 & \mathbb{1} \\ -\mathbb{1} & 0 \end{pmatrix} \Leftrightarrow \det A = 1$.



Odwzorowania nieliniowe są symplektyczne jeśli ich macierze Jacobiego są symplektyczne. Geometrycznie: powierzchnię można przybliżać sumą mnogościową 2D równoległoboków.

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

(A) Zaimplementować algorytmy:

1. *explicit Euler*,
2. Eulera-Cromera,
3. RK4,
4. Verleta,
5. *leapfrog*,
6. RKF45.

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

(A) Zaimplementować algorytmy:

1. *explicit Euler*,
2. Eulera-Cromera,
3. RK4,
4. Verleta,
5. *leapfrog*,
6. RKF45.

(B) Porównać ich działanie dla równań:

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

Porównać z rozw. dokładnymi (analitycznymi). Przyjąć nietrywialne war. pocz.

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

(A) Zaimplementować algorytmy:

1. *explicit Euler*,
2. Eulera-Cromera,
3. RK4,
4. Verleta,
5. *leapfrog*,
6. RKF45.

(B) Porównać ich działanie dla równań:

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

Porównać z rozw. dokładnymi (analitycznymi). Przyjąć nietrywialne war. pocz.

(C) Zapisać równanie ruchu zagadnienia 2 ciał, dla orbity kołowej $e = 0$ z $a = 1$, 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. *explicit Euler*,
2. Eulera-Cromera,
3. RK4,
4. Verleta,
5. *leapfrog*,
6. RKF45.

(B) Porównać ich działanie dla równań:

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

Porównać z rozw. dokładnymi (analitycznymi). Przyjąć nietrywialne war. pocz.

(C) Zapisać równanie ruchu zagadnienia 2 ciał, dla orbity kołowej $e = 0$ z $a = 1$, 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 $x(0) = a(1 - e)$, $y(0) = 0$. Wyliczyć $\mathbf{v}(0)$ z całki energii (całki siły żywej, *vis-viva*) — rozłożyć na stosowne składowe.