حل معادله‌ی دیفرانسیل مرتبه‌ی دوم

درس‌نامه‌ی کاربرد کامپیوتر در فیزیک

(دکتر اجتهادی، دانشگاه شریف، پاییز ۸۵)

در این بخش به حل معادلات دیفرانسیل مرتبه‌ی دوم می‌پردازیم و علاوه بر الگوریتم اویلر با الگوریتم‌های دیگری هم برای حل معادلات دیفرانسیل آشنا می‌شویم.

۱ الگوریتم اویلر

معادلات دیفرانسیل مرتبه‌ی دوم وقتی که با الگوریتم اویلر حل می‌شود فرق چندانی با حل معادله‌ی دیفرانسیل مرتبه‌ی اول ندارد. معادله‌ی دیفرانسیل مرتبه‌ی دوم زیر را در نظر بگیرید:

$$\ddot{y} = f(x, y, \dot{y}) \tag{۱}$$

این معادله را می‌شود در دو مرحله حل کرد. ابتدا معادله‌ی (۱) را به دو معادله‌ی دیفرانسیل مرتبه‌ی اول زیر تبدیل می‌کنیم:

$$\dot{y} = \frac{dy}{dt} \tag{۲}$$

$$\ddot{y} = \frac{d\dot{y}}{dt}$$

و سپس هر کدام را با الگوریتم اویلر حل می‌کنیم:

$$y_{n+۱} = y_n + \dot{y}_n h + O(h^۲) \tag{۳}$$

$$\dot{y}_{n+۱} = \dot{y}_n + f(x_n, y_n, \dot{y}_n)h + O(h^۲)$$

که در جمله‌ی دوم به جای مشتق دوم از تابع f استفاده کردیم. همان‌طور که می‌بینید در هر دو جمله بسط تیلور را تا مرتبه‌ی دوم h نوشتیم. اگر معادله‌ی (۱) رابطه‌ی نیوتن باشد، x زمان، y مکان، $\dot{y}$ سرعت، و $\ddot{y}$ شتاب خواهد بود و معادلات (۳) به صورت

$$x_{i+۱} = x_i + v_i h \tag{۴}$$

$$v_{i+1} = v_i + a_i h$$

در می‌آید. همان‌طور که می‌بینید الگوریتم اویلر را می‌توان هر معادلات دیفرانسیل با هر مرتبه‌ای تعمیم داد و نوشت. اگرچه الگوریتم اویلر ساده‌ترین الگوریتمی است که برای حل معادلات دیفرانسیل می‌توان پیشنهاد کرد، اما بهترین نیست. در ادامه با الگوریتم‌های دیگری که دارای خطای کمتری هستند، آشنا خواهیم شد.

۲ گسسته‌سازی

اگر بسط تابع f را در اطراف نقطه‌ی x_0 به صورت زیر نشان دهیم:

$$f(x_0) = f_0$$

$$f_{\pm 1} = f(x_0 \pm h) = f_0 \pm f'h + \frac{f''}{2}h^2 \pm \frac{f'''}{6}h^3 + O(h^4) \quad (5)$$

$$f_{\pm 2} = f(x_0 \pm 2h) = f_0 \pm 2f'h + 2f''h^2 \pm \frac{4}{3}f'''h^3 + O(h^4)$$

ساده‌ترین مشتق‌گیری، مشتق دونقطه‌ای است.

$$f' = \frac{f_1 - f_0}{h} + O(h) \quad (6)$$

می‌توان مشتق را به کمک سه نقطه (نقطه‌ی قبلی، نقطه‌ی مورد نظر و نقطه‌ی بعدی) به دست آورد و مشتق سه‌نقطه‌ای را این گونه نوشت:

$$f' = \frac{f_{+1} - f_{-1}}{2h} + O(h^2) \quad (7)$$

و مشتق پنج‌نقطه‌ای را به صورت زیر:

$$f' = \frac{1}{12h}(f_{-2} - 8f_{-1} + 8f_0 - f_2) + O(h^4) \quad (8)$$

همان‌طور که می‌بینید وقتی از تعداد نقاط بیشتری برای محاسبه‌ی مشتق حساب می‌کنیم، مشتق با دقت بیشتری محاسبه می‌شود.

۳ واحدهای کاهش‌یافته

ذره‌ای را در نظر بگیرید که در حال سقوط آزاد است و نیروی اصطکاکی برابر $f_d(v) = -kv$ به آن وارد می‌شود. برای این ذره می‌توان نوشت:

$$F = mg - kv \quad (9)$$

می‌دانیم که این ذره وقتی به سرعت حد می‌رسد که نیروی جاذبه و اصطکاک هوا به تعادل برسند:

$$f_d(v) = -kv_f = -mg \tag{۱۰}$$

و در نتیجه سرعت حد خواهد بود:

$$v_f = \frac{mg}{k}$$

اگر رابطه‌ی (۹) را بر حسب v_f بازنویسی کنیم و mg را به سمت چپ معادله بیاوریم:

$$\frac{F}{mg} = 1 - \frac{v}{v_f}$$

این بدین معنی است که در دستگاهی کار می‌کنیم که v_f واحد سرعت و mg واحد نیرو است. پس رابطه‌ی (۳) به معادله‌ی زیر تبدیل می‌شود.

$$F = 1 - v \tag{۱۱}$$

نکته‌ای که در کاهش واحدهای مسئله باید به آن توجه کرد این است که بی‌شمار کمیت مشخصه نمی‌توان تعیین کرد؛ چون در این صورت به تناقض می‌رسیم. برای مثال در مسئله‌ی پرتابه‌ی با اصطکاک چون فقط سه کمیت L و T و M در مسئله دخیل هستند پس تنها سه کمیت مشخصه می‌توانیم تعریف کنیم.

۴ الگوریتم اویلر—کرامر[1]

این الگوریتم در واقع همان الگوریتم اویلر است، با این تفاوت که هر بار ابتدا سرعت را حساب می‌کنیم و بعد مکان را. این بدین معنی است که بر خلاف الگوریتم اویلر، به جای اینکه از شیب در نقطه‌ی i استفاده کنیم، از شیب در نقطه‌ی $i + 1$ استفاده می‌کنیم.

$$v_{n+1} = v_n + a_n h \tag{۱۲}$$

$$x_{n+1} = x_n + v_{n+1} h$$

مرتبه‌ی خطای این الگوریتم با الگوریتم اویلر یکی است اما در مسائل نوسانی و مواقعی که این الگوریتم نسبت به الگوریتم اویلر پایدارتر است.

۵ الگوریتم نقطه‌ی متوسط[2]

در الگوریتم اویلر تقریب پایین از شیب نقطه زده شد و در الگوریتم اویلر—کرامر تقریب بالا. ایده‌ای که در این الگوریتم مطرح می‌شود این است که از متوسط این دو تقریب به عنوان تقریب بهتری از مقدار شیب نمودار در آن نقطه می‌توان استفاده کرد.

$$v_{n+1} = v_n + a_n h \tag{۱۳}$$

[1] Euler-Cromer
[2] Mean algorithm

$$x_{n+1} = x_n + \frac{1}{2}(v_n + v_{n+1})h$$

همان‌طور که می‌شود دید واقعاً هم این اصلاح باعث می‌شود که الگوریتم با دقت بیشتری مکان را تخمین بزند:

$$x_{n+1} = x_n + \frac{1}{2}(v_n + v_{n+1} + a_n h)h = x_n + v_n h + \frac{1}{2}a_n h^2 \tag{14}$$

که بدین معنی است که برای x خطای این الگوریتم از مرتبه‌ی ۳ است.

۶ الگوریتم نقطه‌ی میانی[3]

در این الگوریتم به جای اینکه از نقطه‌ی قبل یا بعدی

$$v_{n+\frac{1}{2}} = v_{n-\frac{1}{2}} + a_n h \tag{15}$$

$$x_{n+1} = x_n + v_{n+\frac{1}{2}} h$$

مشکل این الگوریتم این است که خودشروع نیست. یعنی با داشتن مقادیر اولیه‌ی سرعت و مکان، الگوریتم نمی‌تواند شروع به محاسبه‌ی سرعت و مکان‌های بعدی بکند. برای شروع الگوریتم باید $v_{\frac{1}{2}}$ را هم بدهیم. یعنی a_0 هم برای شروع نیاز است.

$$v_{\frac{1}{2}} = v_0 + \frac{1}{2}a_0 h \tag{16}$$

در واقع می‌توان نشان داد که با این جمله‌ی اضافی این الگوریتم هم‌مرتبه و شبیه الگوریتم اویلر—کرامر است.

۷ الگوریتم ورلت[4]

اگر x_{n+1} و x_{n-1} را بنویسیم:

$$x_{n+1} = x_n + v_n h + \frac{1}{2}a_n h^2 + \beta_n h^3 + \gamma_n h^4$$

$$x_{n-1} = x_n - v_n h + \frac{1}{2}a_n h^2 - \beta_n h^3 + \gamma_n h^4$$

و با هم جمع کنیم:

$$x_{n+1} = 2x_n - x_{n-1} + a_n h^2 + O(h^4) \tag{17}$$

[3] mid-point algorithm
[4] Verlet

شکل ۱: مقایسه‌ی الگوریتم‌های مختلف در محاسبه‌ی مقدار تابع در قدم بعدی به کمک شیب نمودار

به این روش محاسبه‌ی مکان در هر قدم، «الگوریتم ورلت» گویند. در این الگوریتم مکان در هر قدم از روی دو قدم قبلی به دست می‌آید. با این کار می‌توان خطا را تا مرتبه‌ی $O(h^4)$ کاهش داد. برای محاسبه‌ی سرعت در این الگوریتم، در معادله (؟) اگر مقادیر x_{n+1} و x_{n+1} را از هم کم کنیم، داریم:

$$x_{n+1} - x_{n+1} = 2v_n h + O(h^3)$$

که از روی آن به راحتی می‌توان سرعت را محاسبه کرد:

$$v_n = \frac{x_{n+1} - x_{n-1}}{2h} \tag{18}$$

این الگوریتم، الگوریتم مناسبی برای محاسبه‌ی سرعت نیست؛ چون برای محاسبه‌ی v_n دو جمله‌ی هم‌مرتبه‌ی x_{n+1} و x_{n-1} را داریم از هم کم می‌کنیم و در نتیجه خطای گرد کردن ظاهر می‌شود. علاوه بر این مشکل این الگوریتم هم خودشروع نیست. در این الگوریتم ....

۸ الگوریتم ورلت سرعتی[5]

علت نام‌گذاری این الگوریتم بدین صورت بدین خاطر است که این الگوریتم همان الگوریتم ورلت است با این تغییر که سرعت در این الگوریتم به روش دیگری محاسبه می‌شود. محاسبه‌ی سرعت در این الگوریتم شبیه محاسبه‌ی مکان در الگوریتم نقطه‌ی متوسط است. بعلاوه بر خلاف الگوریتم ورلت، این الگوریتم خودشروع هم هست. به خاطر این مزیت‌ها، این الگوریتم بیشتر در حل مسائل مکانیک نیوتنی مورد استفاده قرار می‌گیرد. مکان و سرعت در این الگوریتم به صورت زیر محاسبه می‌شوند:

$$x_{n+1} = x_n + v_i h + \frac{1}{2} a_n h^2 + O(h^4) \tag{19}$$

[5] Velocity Verlet

$$v_{n+1} = v_n + \frac{1}{2}(a_n + a_{n+1})h$$

می‌توان نشان داد که دو معادله‌ی محاسبه‌ی مکان و سرعت در ورلت سرعتی با دو معادله‌ی الگوریتم ورلت معادل هستند. بدین معنی که از هر کدام از زوج معادلات شروع کنیم می‌توانیم به زوج دیگر معادلات برسیم. برای مثال

$$x_{n+1} = 2x_n - x_{n-1} + a_n h^2 \tag{20}$$

$$= x_n + \frac{1}{2}(x_{n+1} - x_{n-1}) - \frac{1}{2}x_{n-1} - \frac{1}{2}x_{n+1} + x_n + a_n h^2$$

$$= x_n + v_n h - \frac{1}{2}a_n h^2 + a_n h^2$$

$$= x_n + v_n h + \frac{1}{2}a_n h^2$$

۹ الگوریتم بیمَن[۶]

در این الگوریتم

$$x_{n+1} = x_n + v_n \Delta t + \frac{1}{6}(4a_n - a_{n-1})h^2 \tag{21}$$

$$v_{n+1} = v_n + \frac{1}{6}(2a_{n+1} + 5a_n - a_{n-1})h$$

۱۰ الگوریتم حدس و امتحان

۱۱ الگوریتم رونگ—کوتا[۷]

برای معادله‌ی دیفرانسیل مرتبه‌ی اول یک الگوریتم داریم به نام «رونگ—کوتا ۲»[۸].

$$\frac{dy}{dx} = f(x, y) \tag{22}$$

[۶] Beeman
[۷] Rung-Kutta
[۸] Rung-Kutta 2

$$y_{n+1} = y_n + k_2 \quad (23)$$

$$k_1 = f(x_n, y_n)h$$

$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})h$$

برای الگوریتم «رونگ—کوتا ۴» داریم:

$$y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4 + O(h^5) \quad (24)$$

$$k_1 = f(x_n, y_n)h$$

$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})h$$

$$k_3 = f(x_n + \frac{h}{2}, y_n + \frac{k_2}{2})h$$

$$k_4 = f(x_n + h, y_n + k_3)h$$

۱۲ ناپایداری ناشی از الگوریتم

نوع دیگری از ناپایداری‌ها به دلیل نوع الگوریتم است. این ناپایداری به دلیل آن است که شکل الگوریتم، جواب‌های مجازی می‌دهد. برای مثال داریم معادله‌ی دیفرانسیل مرتبه یک را حل می‌کنیم ولی شکل الگوریتم به صورتی نوشته شده که انگار یک معادله دیفرانسیل مرتبه دو دارد حل می‌شود و دو جواب به ما می‌دهد. که یکی از جواب‌ها، جواب واقعی معادله‌ی دیفرانسیل مرتبه‌ی یک است و جواب دیگر جواب مجازی است که باعث می‌شود جواب‌های غلطی به دست بیاید.

به عنوان مثال معادله (۱۷) را اگر به صورت

$$y_{n+1} = y_n + f(x_n, y_n)h + O(h^2)$$

$$y_{n-1} = y_n - f(x_n, y_n)h + O(h^2)$$

شکل ۲: ناپایداری ناشی از الگوریتم

از جمع دو عبارت بالا داریم:

$$y_{n+1} = y_{n-1} + 2f(x_n, y_n) + O(h^3) \tag{۲۵}$$

$$y = Ar^n \tag{۲۶}$$

حال اگر مقدار y را در معادله‌ی (۲۰) جایگذاری کنیم، داریم:

$$Ar^{n+1} = Ar^{n-1} - 2Ar^n h$$

$$r^2 - 1 + 2rh = 0 \tag{۲۷}$$

که از حل آن داریم:

$$r = -h \pm \sqrt{h^2 + 1} \simeq -h \pm 1 \tag{۲۸}$$

که دارای دو جواب هست:

$$r_+ = 1 - h < 1 \tag{۲۹}$$

$$r_- = |-(1+h)| > 1$$

$$y = A_+ R_+^n + A_- R_-^n$$

با گذشت زمان جواب درست را از دست می‌دهیم و تنها جواب اشتباه باقی می‌ماند که یک ناپایداری به جواب القا می‌کند.