مبانی آنالیز عددی

بهنام هاشمی

دانشیار دانشگاه صنعتی شیراز

توجه: این متن، نسخه‌ی نهایی نیست و محتوای آن مرتبا در حال تغییر است؛ از جمله، ارجاع به منابعِ مورد استفاده، هنوز داده نشده است. لطفا هرگونه اشتباه احتمالی را به آدرس ایمیل hoseynhashemi@gmail.com ارسال فرمایید.

فهرست مطالب

فصل ۱

خطا در آنالیز عددی

۱.۱ منابع خطا در محاسبات علمی

بسیاری از رشته‌های مهندسی و علوم نیاز به محاسبات علمی دارند. نقطه‌ی شروع، معمولا یک مدل ریاضی است که شامل تعدادی پارامتر بوده و موقعیت مد نظر را توصیف می‌کند. بعنوان مثال، فرض کنید یک مهندس عمران قصد داشته باشد میزان فشارهای وارد بر یک پل فلزی را تجزیه و تحلیل کند. در عمل، جمع‌آوری داده‌ها از طریق اندازه‌گیری پارامترهایی که در مدل، معلوم درنظر گرفته می‌شوند صورت می‌گیرد. مثلا ممکن است مدل، نیاز به طول تیرآهن‌ها و کابل‌ها، زوایای بین آن‌ها و خواص مواد تشکیل‌دهنده‌ی هر قسمت داشته باشد. در همین مرحله‌ی ابتدایی، مقداری خطای اندازه‌گیری رخ می‌دهد چراکه ابزارهای اندازه‌گیری معمولا دقت کامل ندارند و هر اندازه‌گیری به صورت یک تقریب به اضافه یا منهای مقداری عدم‌اطمینان می‌باشد.

خودِ مدلی که انتخاب شده نیز می‌تواند یک منبع خطا باشد. ممکن است برخی فرضیاتِ ساده‌کننده، اِعمال شوند و یا تعدادی از پارامترهای با اهمیت کمتر نادیده گرفته شوند. مثلا ممکن است فرض شود که مواد بکاررفته در هر تیرآهن همگن هستند در حالی که در اصل چنین نبوده. خطای مدل‌سازی، نتیجه‌ی تفاوت پل واقعی و مدل قابل محاسبه‌ی مهندس است.

وقتی مدل، غیرخطی باشد یعنی روابط بین پارامترهای موثر به صورت ساده‌ی خطی نباشد، ممکن است الگوریتمی برای محاسبه‌ی جواب ارائه شود که جواب را به صورت حدی هم‌چون

$$\lim_{n\to\infty} G(n)$$

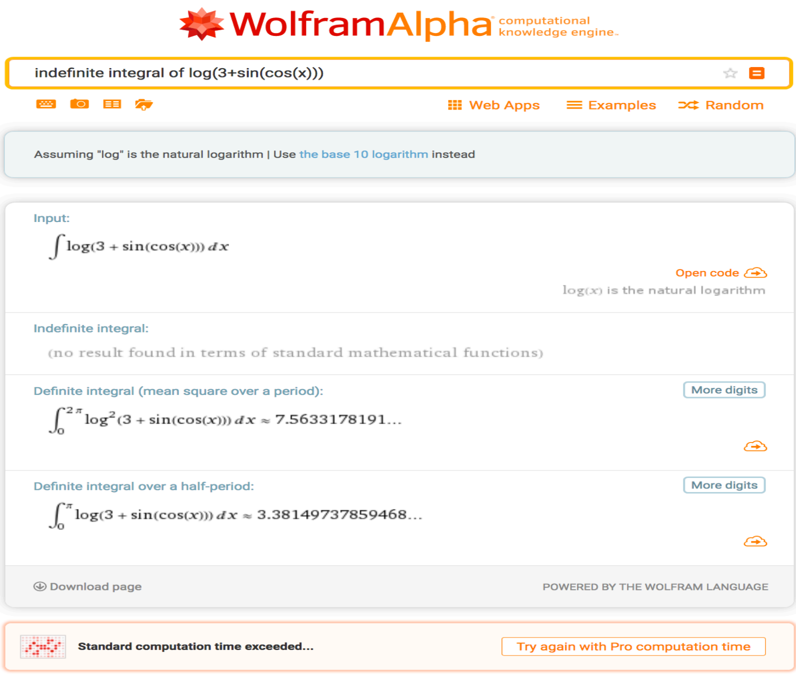
بیان می‌کند که در آن $G(n)$ مثلا جواب بعد از n تکرار است. معمولا چنین حدی را نمی‌توان در عمل محاسبه کرد و ممکن است به تقریبی که بعد از تعدادی متناهی مرحله به دست می‌آید، رضایت دهیم مثلا تصمیم بگیریم که $G(150)$ تقریبی به اندازه‌ی کافی خوب برای اهداف ماست. چنین عملی که به ریاضیاتِ مساله مربوط می‌شود، خطای گسسته‌سازی یا برشی یا خطای تقریب را معرفی می‌کند. موقعیتی مشابه، وقتی است که در هنگام محاسبه‌ی مشتق یک تابع در یک نقطه، بجای اینکه طول گام h را به صفر میل دهیم، به تقریبی که با یک h کوچک به دست می‌آید رضایت دهیم.

در نهایت، نوعا الگوریتمِ انتخاب‌شده برای حل مساله در یک کامپیوتر پیاده‌سازی و اجرا می‌شود. به طور کلی می‌توان دو رویکرد متفاوت از انواع محاسبات را درنظر داشت. یکی محاسبات نمادین است و دیگری محاسبات عددی. محاسبات نمادین به نوعی شبیه است به آنچه یک ریاضی‌کار محضِ ایده‌آل‌گرا با قلم و کاغذ انجام می‌دهد: کاش بتوان مساله را به صورت تحلیلی حل کرد! زیبایی این رویکرد در دقت بی‌نهایتش است و زشتیش در سرعت پایین آن[1] متخصصین علوم کامپیوتر و نظریه‌ی گراف تلاش‌های فراوانی کرده‌اند تا نرم‌افزارهایی تولید کنند که چنین نوع محاسباتی را اجرا می‌نماید. قدرتمندترین نرم افزارهای محاسبات نمادین عبارتند از مَتِمَتیکا[2]، مِیپِل[3] و جعبه ابزار محاسبات نمادین مَتلَب[4]. هرچند این نرم‌افزارها گاها توانایی حیرت‌انگیزی در حل برخی مسائل داشته و در چنین مواردی بسیار مفید و ارزشمند تلقی می‌شوند اما مشکل اصلی آنها همان است که قبلا گفتیم: پایین بودن سرعت اجرا در حل بیشتر محاسبات علمی مورد نیاز در زندگی روزمره و یا این‌که اصولا چنین مسائلی در بسیاری مواقع هیچ جواب تحلیلیِ به فرم بسته‌ای ندارند که بتوان آن را با محاسبات نمادین به دست آورد. لازم نیست راه درازی برویم تا مساله‌ای بیابیم که محاسبات نمادین از حل آن عاجز باشد: کافی است یک انتگرال معین کمی پیچیده را بخواهیم، یا فقط پنج مقدار ویژه‌ی یک ماتریس با ده هزار سطر و ستون که اندازه‌ی آن‌ها بزرگ‌تر از ۹۹۹۵ مقدار ویژه‌ی دیگر است، یا حل یک معادله‌ی دیفرانسیل معمولی غیرخطی یا یک معادله‌ی دیفرانسیل با مشتقات جزیی. از آن‌جا که متمتیکا یک نرم‌افزار محاسبات نمادین است قاعدتا غیرمنطقی به نظر نمی‌رسد که آن را برای محاسبه‌ی یک انتگرال نامعین بکار ببریم. در این‌جا تلاش کرده‌ایم جواب

$$\int \log(3+\sin(\cos(x)))dx$$

را بیابیم.

[1] کندی سرعت محاسبات نمادین نتیجه‌ی پیچیدگی محاسباتی ترکیبیاتی آن است.
[2] Mathematica
[3] Maple
[4] Symbolic Computation Toolbox in MATLAB



شکل ۱.۱: تلاشی ناموفق برای محاسبه‌ی یک انتگرال نامعین با (نسخه‌ی رایگان) موتور محاسباتی آنلاین متمتیکا در www.wolframalpha.com

در شکل ۱.۱ حداقل سه نکته قابل مشاهده است: یکی شکایت نرم‌افزار از این‌که اصولا جوابی به فرم استاندارد برای این پرسش وجود ندارد جایی که می‌گوید:

no result found in terms of $\cdots$

دیگری این شکایت که زمانی بیش از آنچه انتظار می‌رفته برای محاسبه‌ی این انتگرال لازم بوده جایی که در پایین شکل می‌گوید

Standard computation time exceeded...

نکته‌ی سوم این‌که خود موتور محاسبات نمادین (بدون آن‌که از آن خواسته باشیم)، به رویکرد محاسبات عددی روی آورده و دو نمونه از انتگرال‌های معین را که ((با روش‌های عددی محاسبه کرده)) نشان می‌دهد! مهندسی و علوم کاربردی به صورتی روزافزون با حل چهار مساله‌ای که در بالا ذکر کردیم و نظایر آنها درگیر هستند و رویکردی که عموما در عمل استفاده می‌شود رویکرد دوم است یعنی محاسبات عددی با دقت متناهی[1]. در این رویکرد از تعدادی مشخص و محدود رقم در ذخیره‌ی اعداد و اجرای محاسبات با آن‌ها بهره می‌گیریم. نتیجه‌ی فوریِ این واقعیت، معرفی چهارمین نوع خطا در محاسبات علمی یعنی خطای گردکردن

[1] بعضی مواقع از محاسبات نمادین برای محاسبه‌ی کمیتهای حساس و سپس بکارگیری حاصل در کدهایی که محاسبات عددی را اجرا می‌کنند نیز استفاده می‌شود.

است. بعنوان مثالی ساده، عددی بنام π به صورتی دقیق در محاسبات با دقت متناهی وجود ندارد! به وضوح اعداد گنگ (که می‌دانیم بسط اعشاری نامختوم و غیرتکراری دارند) را نمی‌توان در تعدادی محدود از بیت‌های حافظه‌ی یک کامپیوتر جا داد. داستان خطاهای گردکردن البته فراتر از این مورد (مشکل نمایش اعداد گنگ) است. هدف ما در این درس آشنایی با مبانی این رویکرد محاسباتی است. محاسبات عددی می‌توانند بسیار سریع و مفید باشند اما بدون ایراد هم نیستند! در این محاسبات، بجای کار کردن با اعداد حقیقی (یا مختلط) که یک مجموعه‌ی ناشمارای نامتناهی را می‌سازد، با زیرمجموعه‌ای متناهی و شمارا از آن کار می‌کنیم (که آن‌ها را اعداد ماشین خواهیم نامید)[1]. پس با چهار منبع خطا در محاسبات علمی آشنا شدیم:

- خطای اندازه‌گیری
- خطای مدل‌سازی
- خطای گسسته سازی (برشی یا تقریب)
- خطای گردکردن

دو مورد اول، موضوع این درس نیستند. آنچه در سرتاسر این درس بدان خواهیم پرداخت، خطاهای گسسته‌سازی و گردکردن می‌باشند. در این بین خطای گسسته‌سازی جنبه‌ی ریاضی–الگوریتمی بیشتری دارد اما خطای گردکردن نیز که منشأ آن بیشتر کامپیوتری بوده و ناشی از کار با اعداد است را می‌توان با ابزار ریاضی تجزیه و تحلیل کرد. در اینجا کمی بیشتر در مورد خطای گسسته‌سازی که تا انتهای درس با آن سر و کار خواهیم داشت بحث می‌کنیم.

از مهمترین قضایا در سرتاسر ریاضیات، قضیه‌ی تیلور است که ابتدا یک فرم ساده‌ی آن را مرور می‌کنیم. دلیلِ نیاز ما به قضیه‌ی تیلور در این بخش، استفاده از آن برای توصیف خطای گسسته‌سازی است اما اهمیت این قضیه در این درس بسیار فراتر است: بعنوان نمونه در فصل بعد در تحلیل خطای گسسته‌سازیِ درونیابیِ چندجمله‌ای از قضیه‌ی تیلور استفاده خواهیم کرد و یا بعدا در ساختن برخی از فرمول‌های مشتق‌گیری عددی نیز قضیه‌ی تیلور را بکار خواهیم برد.

[1] هر سه نرم‌افزاری که ذکر کردیم را می‌توان هم برای اجرای محاسبات نمادین استفاده کرد و هم محاسبات عددی. با این حال متمتیکا و میپل به لحاظ تاریخی با هدف محاسبات نمادین ابداع و توسعه یافته‌اند و متلب با هدف محاسبات عددی. متمتیکا و میپل به خاطر قدرتشان در محاسبات نمادین شناخته می‌شوند و متلب بخاطر توانایی‌اش در محاسبات عددی بخصوص از نوع محاسبات ماتریسی.

قضیه‌ی ۱.۱.۱. فرض کنید $f(x)$ دارای مشتق تا مرتبه‌ی $n+1$ بر بازه‌ی $[a,b]$ بوده و $x_0 \in [a,b]$. آن‌گاه برای هر $x \in [a,b]$، نقطه‌ای هم‌چون ζ بین x_0 و x موجود است به‌طوری که

$$f(x) = p_n(x) + R_n(x),$$

که در آن

$$p_n(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \ldots + \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n,$$

و

$$R_n(x) = \frac{f^{(n+1)}(\zeta)}{(n+1)!}(x-x_0)^{n+1}.$$

$p_n(x)$ چندجمله‌ای تیلور درجه n تابع f در مجاورت نقطه‌ی x_0 و $R_n(x)$ باقیمانده‌ی متناظر با چندجمله‌ای $p_n(x)$ نامیده می‌شود. برای دسته‌ی بزرگی از توابع مهم هم‌چون توابع تام (که بینهایت‌بار مشتق‌پذیر هستند) n را می‌توان به دلخواه بزرگ گرفت. یعنی وضعیتی بیش از شرایط ذکرشده در نسخه‌ی بالا از قضیه‌ی تیلور برقرار است و یک سری بینهایت-جمله‌ای برای بسط تابع f در نقطه‌ی x حول x_0 موجود است:

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k = \underbrace{\sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k}_{p_n(x)} + \underbrace{\sum_{k=n+1}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k}_{R_n(x)}$$

از آن‌جا که در $p_n(x)$ تنها $n+1$ جمله‌ی ابتدایی این سریِ نامتناهی را نگاه داشته‌ایم، مناسب است که باقیمانده‌ی $R_n(x)$ را بعنوان خطای گسسته‌سازی یا برشی متناظر با $p_n(x)$ نیز در نظر بگیریم: از درس ریاضی ۱ می‌دانیم که وقتی $x_0 = 0$، سری مک‌لورن بعنوان حالت خاصی از سری تیلور حاصل می‌شود:

$$f(x) = f(0) + f'(0)x + \frac{f''(0)}{2!}x^2 + \ldots + \frac{f^{(n)}(0)}{n!}x^n + \ldots$$

بعنوان مثال، در ریاضی ۱ دیدیم که سری مک‌لورنِ تابع نمایی $f(x) = \exp(x)$ عبارت است از

$$\exp(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!} = \underbrace{1 + x + \frac{x^2}{2!} \ldots + \frac{x^n}{n!}}_{p_n(x)} + \underbrace{\frac{x^{n+1}}{(n+1)!} + \frac{x^{n+2}}{(n+2)!} + \ldots}_{R_n(x)}. \qquad (۱.۱)$$

و یکی از نکات معجزه‌آسای قضیه‌ی تیلور این است که بینهایت جمله‌ی موجود در باقیمانده را می‌توان به شکلی بسیار ساده در تنها یک جمله چپاند! در مورد تابعِ نمایی بالا، قضیه‌ی تیلور، وجود نقطه‌ای چون ζ بین صفر و (نقطه‌ی انتخاب‌شده‌ی) x را تضمین می‌کرد به طوری که:

$$R_n(x) = \frac{e^{\zeta}}{(n+1)!}x^{n+1}.$$

مثال ۱.۱.۱. به کمک قضیه‌ی تیلور، تقریبی برای $\sqrt{e}$ و کران بالایی برای خطای گسسته‌سازیِ تقریبِ خود به‌دست آورید.

چون $e^{1/2} = \sqrt{e}$، پس قرار می‌دهیم $x = \frac{1}{2}$. هم‌چنین می‌توان با انتخاب $x_0 = 0$، از سری مک‌لورن برای تقریب مقدار تابع نمایی در نقطه‌ی x استفاده کرد. حال فرض کنید در فرمول (۱.۱)، قرار دهیم $n = 3$. پس با چهارجمله‌ی ابتدایی، چندجمله‌ای تیلور درجه سه را استفاده کرده و داریم:

$$\sqrt{e} = p_3(1/2) + DE$$

که در آن

$$p_3(1/2) = 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{48} = \frac{79}{48}$$

تقریبی است که برای $\sqrt{e}$ در نظر گرفته‌ایم و DE خطای گسسته‌سازی یعنی اختلاف بین مقدار دقیق $\sqrt{e}$ و تقریب $\frac{79}{48}$ است که طبق قضیه‌ی تیلور برابر است با

$$DE = \frac{e^{\zeta}}{4!}(1/2)^4 = \frac{e^{\zeta}}{24 \times 16} = \frac{e^{\zeta}}{384},$$

جایی‌که در آن ζ بین صفر و $\frac{1}{2}$ است. چون تابع نمایی، صعودی است داریم $e^{\zeta} < e^{1/2} < e^1 < 3$ و در نتیجه کران بالایی برای خطای گسسته‌سازی عبارت است از

$$DE < \frac{3}{384} \approx 0.78 \times 10^{-2}.$$

در این مثال، مسئله‌ی یافتن مقدار $\sqrt{e}$ را داشتیم، الگوریتمی که برای حل مساله بکار گرفتیم، استفاده از چندجمله‌ای تیلور درجه سه تابع نمایی بود و خطای گسسته‌سازی، از جایگزین‌کردن یک سری بینهایت جمله‌ای با یک چندجمله‌ای درجه سه ناشی شد. در این مثال تا وقتی تقریب $\frac{79}{48}$ را به همین صورتِ کسری

نگه‌داشته و با تقریبی همچون 1.64583 جایگزین نکنیم، خطای گردکردنی رخ نداده است. می‌توان نشان داد که چند رقم ابتدایی $\sqrt{e}$ در اصل عبارتند از 1.648721270700128. پس می‌بینیم که سه رقم ابتدایی 1.64 از تقریب ما درست بوده و می‌توان دید که خطای گسسته‌سازی درواقع تقریبا برابر است با 2.9×10^{-3} که طبیعتا از کران بالایی که بدست آوردیم کمتر است.