Métodos Numéricos

Um curso para o Mestrado Integrado em Engenharia Informática e Computadores da FEUP

Carlos Madureira
Cristina Vila
Maria de Lurdes Dinis
José Soeiro de Carvalho

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia de Minas

Este documento é uma versão provisória, de utilização exclusiva no âmbito da disciplina de Métodos Numéricos do Mestrado Integrado em Engenharia Informática e Computadores da Faculdade de Engenharia da Universidade do Porto.

Todas as correcções e contribuições são bem-vindas!

Carlos M. N. Madureira (`cmad@fe.up.pt`)

Maria Cristina C. Vila (`mvila@fe.up.pt`)

Maria de Lurdes Dinis (`mldinis@fe.up.pt`)

José M. S. Soeiro de Carvalho (`jmsoeiro@fe.up.pt`)

Versão 2.2 *Pública*
Impresso em 26 de Novembro de 2019

Baseado num documento obtido do commit GIT 0d3db1d, feito por Jose\begingroup \let \relax \relax \endgroup [Pleaseinsert\PrerenderUnicode{ÌĄ}intopreamble] Soeiro Carvalho em 2019-11-26 11:55:33 +0000

Conteúdo

Lista de Figuras

Lista de Tabelas

Introdução

Na prática da computação digital científica verificam-se numerosos e diversificados tipos de situações relacionadas com problemas de Análise Numérica. As que se revelam mais simples são aquelas em que se pretende construir um algoritmo para utilizar apenas uma vez, ou, quando muito, um número muito reduzido de vezes, para resolver um dado problema concreto e em que se tem uma ideia razoável do modo como vão desenvolver-se os cálculos. Nestes casos os problemas de *tempo de processamento* não se põem (pelo menos de modo agudo, sobretudo quando dispomos de um computador pessoal que podemos deixar a trabalhar durante a noite só para nós) e *as questões de precisão* são relativamente simples de tratar (até porque se dispõe, o deve dispor, *a priori* de ordens de grandeza para as soluções, o que permite controlar os erros mais óbvios); porém, a tentação de criar soluções extremamente dependentes das características particulares do problema e das peculiaridades da máquina é quase irresistível, de tal modo que os programas desenvolvidos não têm, em geral, qualquer transportabilidade.

Um segundo tipo de situação é o de um algoritmo destinado a correr diversas vezes em diversos tipos de situações todas incluídas em um mesmo contexto mas destinadas em princípio a um único utilizador, e difere da anterior no facto de existir uma potencial diversidade de dados de partida conducente a diferentes tipos de problemas numéricos. A principal dificuldade operacional reside, neste caso, no esforço de imaginação necessário para prever as diversas classes de situações que virão eventualmente a verificar-se, sob pena de se ter que rescrever sucessivas versões do mesmo algoritmo, uma para cada falhanço detectado. A situação torna-se nitidamente mais complicada quando o algoritmo se destina a ser corrido por diversos utilizadores, porque, nesse caso, é virtualmente impossível prever *a priori* o tipo de complicações numéricas que, em função de diferentes dados de partida, podem vir a ocorrer. Este é o caso típico dos *programas de biblioteca de uso geral*. Neste caso torna-se necessário usar dos maiores cuidados para cobrir tantas situações diferentes quantas seja possível, tendo ao mesmo tempo a habilidade de não produzir programas grosseiramente ineficientes ou tão complicados que desencorajem o utilizador comum. A real dificuldade de conciliar objectivos tão contraditórios e a subjectividade inevitável na ponderação das diferentes prioridades levam muitos utilizadores (incluindo os autores deste curso) a olhar com alguma desconfiança a tendência de certos outros para o uso sistemático de rotinas de biblioteca; o preço de uma tal atitude é, inevitavelmente, a necessidade de escrever os seus próprios algoritmos e programas.

Uma terceira classe de situações corresponde aos problemas computacionais tão grandes, tão complexos, tão particulares ou tão novos que se torna necessário, para os resolver, um conhecimento muito profundo da questão concreta que os gerou e o recurso à intuição física do proponente, que se supõe ser o seu melhor conhecedor. Esta classe de problemas dá, normalmente, origem a soluções particulares, que usam técnicas adequadas às peculiaridades de cada problema, ou de cada formulação particular do problema. O uso dessas soluções por outras pessoas que não o seu próprio autor exige um estudo muito aprofundado que pode, frequentemente, revelar-se infrutífero e/ou impraticável, dada a conhecida tendência dos programadores, sobretudo dos mais criativos, para documentarem mal os seus produtos.

Finalmente, uma outra classe de situações ocorre nas aplicações em tempo real (como os programas de controlo de um processo industrial); nestes casos, o imperativo fundamental, para além de uma precisão (que, ao contrário do que poderia pensar-se, nem sempre será crítica) será o de uma execução extremamente rápida; em casos deste tipo, a rapidez da execução pode mesmo compensar em larga medida uma certa imprecisão dos resultados, desde que não exagerada. Tal como no caso anterior, deve fazer-se um esforço importante para garantir que se dispõe de toda a informação relevante sobre o problema e sobre

os seus possíveis métodos de resolução, bem como de todo o tempo necessário para desenvolver uma solução realmente eficaz. Estes dois últimos casos constituem o domínio de eleição da aplicação das habilidades e artimanhas próprias da arte do analista e do programador.

Em resumo, existem três classes fundamentais de necessidades de algoritmos e programas computacionais:

- soluções expeditas, facilmente disponíveis ou implementáveis, embora possivelmente pouco eficientes, mas não imprecisas;
- soluções cuidadosamente pensadas, precisas, rápidas, seguras e de uso geral;
- soluções particulares para problemas particulares, em que as peculiaridades da aplicação em vista podem desempenhar papel central.

Como é óbvio, o presente curso só trata dos dois primeiros tipos de problemas. O nosso ponto de vista central será, portanto, o de que, embora o objectivo primário seja, como sempre, o de sacar resultados da máquina, esses resultados se destinam a fundamentar decisões, por vezes de importância vital, pelo que se torna necessário, antes de mais, compreender claramente e em profundidade o que tais números podem dizer, ou não dizer, o que podem valer, ou não valer. Assim, considerando que se trata de cadeira de mera introdução colocada muito cedo no plano do curso, a selecção de matérias tem em vista aquilo que de mais básico se prende com os objectivos centrais do exercício da profissão de engenheiro a nível superior, e não aquilo que está mais em voga ou mais bem documentado na literatura técnica e científica; secundariamente, a selecção das matérias nem pretende cobrir todas as necessidades que possam surgir no exercício da profissão, nem escolher os "melhores"métodos no sentido do uso eficiente dos recursos das máquinas - que, ainda por cima, mudam quase todos os dias -, mas apenas seleccionar aquelas matérias que melhor possam compensar o tempo e o esforço gastos no seu estudo.

Uma tal selecção é notavelmente dificultada

- pelo facto de *não existir, por trás da análise numérica, um corpo de doutrina coerente*, o que faz com que as soluções numéricas realmente eficientes e adaptadas sejam frequentemente produtos artesanais frutos de processos criativos baseados em toda uma experiência anterior do autor-utilizador;
- *pela enorme diversidade de métodos, técnicas, variantes e variações concebidas até ao presente* e que a maior parte dos tratadistas tende a apresentar a granel, sem qualquer senso crítico;
- pelo facto de muitas dessas técnicas serem *meros artifícios sem interesse geral*, facto que não pode ser revelado senão por um uso intensivo e constantemente crítico que o ensino convencional não encoraja.

A selecção de métodos aqui apresentada representa, portanto, um mero ponto de vista muito pessoal do autor, caucionado apenas pela experiência investigacional e docente da equipa que tem dirigido no Departamento de Engenharia de Minas ao longo das duas últimas décadas.

Por outro lado, o espaço de escolha encontra-se amputado pelo facto de existir no curriculum uma cadeira de Investigação Operacional, no âmbito da qual recaem os métodos cuja fundamentação é de carácter essencialmente algébrico - programação linear, algoritmos de árvores e redes, etc. Neste contexto, o capítulo sobre sistemas de equações lineares aparece como u, tanto estranho no curso, dado que a sua metodologia releva dos métodos algébricos.

A construção do curso no seu conjunto assenta sobre uma opção táctica muito clara, resultante de muitos anos de ensino da matéria ao nível da licenciatura: *total separação entre a Análise Numérica* - entendida como construção de algoritmos e análise do seu comportamento face à inevitável finitude da representação concreta dos números dentro da máquina - e *a programação desses algoritmos* no contexto de uma linguagem e/ou de uma máquina particular. Com efeito, toda a experiência passada nos mostrou

para além de qualquer dúvida razoável que a mistura imprudente das duas abordagens e dos respectivos pontos de vista e polarizações faz com que o estudante médio, engodado pela preocupação, não de todo ilegítima mas limitativa, da produtividade imediata, concentre a totalidade do seu esforço no segundo e esqueça por completo o primeiro, aquele que constitui, precisamente, o objectivo primordial da cadeira; acontece que o ponto de vista peculiar da programação valoriza desproporcionadamente os problemas da lógica e da economia do algoritmo e tende a remeter para segundo plano os problemas vitais da precisão, problemas que são de carácter muito menos racionalizável e, por esse facto, de abordagem muito mais árdua para o estudante de formação racionalista.

Um outro ponto de vista didáctico que confere um certo grau de originalidade à estrutura da cadeira é o de se praticar sistematicamente um certo tipo de "reconstrução"da história dos métodos da Análise Numérica com a intenção de mostrar como os métodos se encadeiam logicamente uns com os outros e, nessa perspectiva, podem constantemente ser redesenhados, o que favorece uma das atitudes mentais mais importantes para o candidato a analista numérico.

A necessidade do uso, na cadeira, de algoritmos implementados em computador, indispensável para a aquisição do seu completo domínio pelo estudante, cria um dilema extremamente difícil de ultrapassar: por um lado, os cuidados necessários em matéria de precisão são pouco compatíveis com programação de principiantes e, por outro lado, como se depreende do que ficou dito no início, temos sérias reservas quanto ao uso cegos de programas de biblioteca. Por isso, todos os exemplos e manipulações exibidas e todos os exercícios propostos se destinam a ser trabalhados à mão ou à máquina de calcular ou, na melhor das hipóteses, em folha de calculo; uma tal estratégia, quando bem praticada, permite seguir passo a passo eventuais incidentes resultantes de problemas de representação; no entanto, uma vez dominados os princípios e os conceitos, deve ser feito um esforço sério para formalizar a aprendizagem sob a forma de construção de programas informáticos que incorporem explicitamente todos os critérios de segurança e precaução que a teoria e a prática "à mão"aconselharam.

Finalmente, na concepção e implementação do curso, um outro problema se levantou: o da recente vulgarização dos *manipuladores simbólicos*, ferramentas informáticas que,

- ao contrário das vulgares linguagens de programação, têm a possibilidade de manipular directamente expressões simbólicas e não apenas números;
- permitem representar os números (dados, resultados intermédios e finais) com precisão "infinita"(dentro, naturalmente, dos limites de memória da máquina), representando os racionais sob a forma de quocientes de inteiros de comprimento adequado e mantendo em suspenso todas as operações irracionais; em opção, um modo de cálculo aproximado permite trabalhar com um número arbitrário de algarismos significativos.

Nestas condições, os manipuladores simbólicos permitem a verificação e o controlo do cálculo programado em linguagens convencionais e é um dos objectivos do curso levar o estudante a encará-los como tal.

Estamos mesmo em crer que, dentro da evolução previsível das capacidades de memória e das velocidades de cálculo dos computadores das próximas gerações, esta é uma das direcções genéricas em que se desenvolverá o cálculo científico nas próximas décadas, o que tornará obsoletas muitas das abordagens tradicionais da análise numérica. Pensamos, com efeito, que a generalização do uso dos manipuladores simbólicos corre o risco, se não for devidamente pensada, estudada e integrada, de vir a desempenhar, em relação ao ensino superior, o mesmo papel ambíguo, e por isso confusionista e dissolvente, que a generalização do uso das calculadoras de bolso desempenhou, em relação ao secundário, nas décadas de 70 e 80. Por outro lado, há que reconhecer que tais ferramentas informáticas se encontram apenas na fase de utilitários para a realização de cálculos avulsos e não podem ainda ser eficientemente integradas em programas construídos pelo utilizador; isso impede, naturalmente, a completa centragem nelas de uma cadeira que não pode dar-se ao luxo de ignorar as reais necessidades da actual computação científica e técnica. Assim, incapazes, neste momento, de desenvolver um curso completo ao longo desta linha

estratégica, os autores limitaram-se a fazer apenas algumas tímidas tentativas de introdução ao uso de manipuladores simbólicos muito acessíveis e que conhece como **`derive`**, A Mathematical Assistant ™, ou **`Maxima`**, a Computer Algebra System, a propósito de alguns problemas mais delicados levantados pelo desenvolvimento de um curso que é ainda fundamentalmente convencional.

O curso está estruturado por capítulos, em que se descreve um problema e se estudam algumas soluções numéricas:

O Erro em Análise Numérica Neste capítulo são abordados os processos de cálculo geradores de erro, quer por truncatura de processos infinitos, quer em consequência da própria representação. É dado algum destaque à representação em vírgula flutuante.

Zeros reais de uma função real A resolução de equações e sistemas de equações não lineares, conceptualmente um problema simples, introduz várias heurísticas relevantes para outros temas, nomeadamente as ideias de pesquisa, redução intervalar, ajuste, e os procedimentos iterativos de ponto fixo.

Sistemas de Equações Lineares O simplório método algébrico de Gauss para resolver sistemas de equações lineares, levanta um conjunto muito interessante de problemas quando aplicado a um cálculo concreto, nomeadamente pela dificuldade em detetar uma quantidade que vale zero, pela propagação de erros e pelo conceito de *resíduo*. São apresentadas alternativas algébricas e numéricas iterativas.

Quadratura e Cubatura Este capítulo trata do cálculo numérico de integrais definidos, em que os efeitos da truncatura resultante da transformação de uma diferencial num acréscimo são tornados bem claros. Introduz o conceito de *ordem* de um método e a heurística do *quociente de convergência*. Apresenta a ideia que os problemas de dimensão superior são resolúveis por redução de ordem, que um *integral duplo* é na realidade um *integral simples* de um *integral simples*.

Integração de equações diferenciais ordinárias A resolução numérica de equações diferenciais é um problema transversal a toda a engenharia. Os métodos são apresentados como especializações de um método universal, o método de Euler, modificado por uma de duas heurísticas: a descoberta da derivada média; ou a ideia que um bom preditor do futuro próximo está no passado recente.

Optimização No capítulo final sobre optimização, o problema é reduzido à optimização convexa, sendo explorados os problemas unidimensionais e em seguida trabalhados os problemas e métodos multidimensionais.

Os capítulos tem propostas de algoritmos, exemplos e exercícios, sendo propostas também aplicações de teste, que permitem comparar os resultados obtidos por processos numéricos com os resultados tipicamente mais precisos obtidos por procedimentos algébricos ou analíticos.

[DB74]

[CB81]

[Gol91]

[Ham71]

[Lie68]

[Moo66]

[Mul89]

[Wal90]

1 O Erro em Análise Numérica

Contents

Figures

Tables

1.1 Matemáticas e Análise Numérica

O senhor I. N. Génuo pretende fazer uma aplicação bancária de longo prazo para garantir o futuro da sua família; para isso, procura o Banco Caótico Português, que lhe oferece o seguinte plano-poupança:

> você faz uma entrega inicial de $(e-1)$ euros (sendo $e = 2.718281828...$) e ao fim de 1 ano multiplicamos o seu capital por 1 e subtraímos 1 euro para despesas administrativas; ao fim de 2 anos multiplicamos o seu capital por 2 e subtraímos 1 euro para despesas administrativas; ao fim de 3 anos multiplicamos o seu capital por 3 e subtraímos 1 euro para despesas administrativas; e assim sucessivamente, até que ao fim de 25 anos você pode retirar todo o dinheiro acumulado.
>
> Cuidadoso, o senhor I. N. Génuo não se compromete e vai para casa pensar. Chegado a casa pega na sua Casio FX702P para saber quanto terá ao fim de 25 anos e o resultado aterroriza-o: o saldo será devedor de `140 302 545 600 000` euros! Duvidando da sua competência de calculador, corre a um amigo informático que tem uma estação Sun Sparc e pede-lhe que calcule o saldo. A resposta é positiva: `4 645 987 753` euros! Satisfeito, corre ao Banco para subscrever esse interessante plano. Ao fim de 25 anos tem um desgosto: o Banco entrega-lhe `4` milésimos de cêntimo. Este é um exemplo típico de cálculo instável que, devido a erros de arredondamento nos dados iniciais e nos resultados intermédios, pode arrastar enormes variações no resultado final.
>
> J. M. Muller, Ordinateurs en quête d'arithmétique
>
> La Recherche, N^o Spécial, Juillet/Août 1995, adaptado

As *Matemáticas convencionais* (sobretudo quando consideradas no sentido em que são correntemente ensinadas nos nossos cursos superiores) diferem significativamente da *Análise Numérica*, na realidade muito mais significativamente que o que habitualmente se considera. A diferença mais evidente reside no uso constante da noção de *infinito* que nelas se faz,

- tanto a propósito da representação dos números, que são sempre considerados como representados com precisão infinita;
- como a propósito de processos de cálculo (passagem ao limite, cálculo de uma derivada, soma de uma série, cálculo de um integral definido),

enquanto que a computação, que é o objecto da Análise Numérica, é sempre feita por meio de dispositivos, manuais ou automáticos, dotados de precisão inelutavelmente finita e que operam em tempo finito (e, se possível, muito curto!).

Exemplo: 11.1 Matemática, números e processos

Números típicos das matemáticas:

$$\pi = 3.14159265358979323846264338327950288419716939937510\ldots$$

$$1/3 = 0.33333333333333333333333333333333333333333333333333\ldots$$

Processos típicos das matemáticas:

$$\frac{dy}{dt} = \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x}$$

$$\int_a^b f(x)\,dx = \lim_{\Delta x_i \to 0} \sum_{i=1}^{\frac{b-a}{\Delta x_i}} f(x_i)\Delta x_i$$

Mesmo erros de arredondamento muito pequenos, quando acumulados ao longo de processos de cálculo complexos, podem ter efeitos extremamente perversos, dando origem a perdas de precisão inesperadas, sem qualquer comum medida com as causas que lhes deram origem. Este facto, nem sempre devidamente apreciado pelos principiantes, é, no entanto, de importância fundamental, e a falta do seu reconhecimento expresso é causa frequente de muitas confusões e de graves acidentes; no dizer lapidar de [Ham71]

> "só a parte das matemáticas convencionais que se revela resistente a estes efeitos perversos tem interesse para a análise numérica, assim como para qualquer outra aplicação das matemáticas ao mundo real".

Só por este facto, a Análise Numérica seria, de direito próprio, parte fundamental das Matemáticas, no que se refere à sua aplicabilidade, embora constituindo o que temos que reconhecer como um ramo não-convencional destas últimas.

Outras diferenças entre as Matemáticas convencionais e a Análise Numérica são mais subtis ou mesmo mais encapotadas, por resultarem do uso em sentidos profundamente diferentes dos mesmos termos e expressões. Assim, por exemplo, como veremos muito em breve, a expressão *zero de uma função* pode ter diferentes sentidos em análise numérica, nenhum deles coincidente com o que tem em matemática.

Outras diferenças ainda são as que residem em profundas divergências de objectivos, de gostos e mesmo de conceitos de elegância entre as Matemáticas convencionais e a Análise Numérica. Assim, por exemplo, os teoremas de existência, tão caros aos matemáticos, raramente têm qualquer interesse em análise numérica; do mesmo modo, a Matemática prefere teoremas exactos e precisos, demonstráveis e demonstrados, enquanto a Análise Numérica frequentemente se satisfaz com resultados meramente plausíveis e com métodos heurísticos, intuitivos.

As consequências destas diferenças, nem sempre óbvias nem apreciadas, são, no entanto, sempre importantes e, em certos casos, podem prestar-se a confusões perigosas. Assim, por exemplo, enquanto em certos casos o resultado de um cálculo numérico pode considerar-se como uma aproximação satisfatória ao ideal matemático, em numerosos outros casos a diferença é tal que nem sequer se pode considerar o resultado como minimamente aproximado; no primeiro caso, a tendência corrente é para, sem mais análise, imputar a diferença aos "defeitos"inerentes à análise numérica; no segundo caso, por extensão, tende a pensar-se que se trata simplesmente de uma variante exagerada do caso anterior, isto é, a culpa é automaticamente atribuída ao cálculo numérico; ora, em muitos casos, se não mesmo na maioria, é precisamente o modelo matemático, por exacto que pareça, que está em falta, por se não adequar minimamente à realidade em estudo, e esta realidade tem uma dupla face: a do problema físico subjacente ao modelo matemático e a do sistema computacional subjacente à resolução do modelo matemático.

1.2 A representação dos números numa máquina

> **Vamos explorar agora a dicotomia numérico versus digital! ...** citando (apocrifamente ?) um conhecido compositor (1981).

Para além do facto importante e bem conhecido, e sobre o qual não insistiremos especialmente, de a representação interna dos números nas máquinas electrónicas ser quase sempre *não-decimal*, mas sim binária, octal ou hexadecimal, o analista numérico tem que ter bem presente que os computadores digitais usam duas classes de representações de números que a matemática convencional desconhece em absoluto. Para compreender este facto é preciso, em primeiro lugar, aceitar a ideia de que os números não existem na realidade exterior a nós, de que são apenas construções conceptuais. Assim, por exemplo Conte e de Boor em [CB81] afirmam:

> a Análise Matemática tornou-se uma disciplina exacta apenas quando deixou para trás as restrições do cálculo numérico e passou a tratar exclusivamente os problemas levantados por um modelo abstracto do sistema dos números, o modelo dito dos *números reais*. Este modelo foi formulado intencionalmente para tornar possível uma definição conceptualmente operacionalizável do conceito de limite, o que abre caminho para a resolução matemática de uma multidão enorme de problemas práticos, uma vez que esses problemas sejam traduzidos na linguagem do modelo. No entanto, permanece ainda de pé o problema da retroversão das soluções simbólicas ou abstractas encontradas para a linguagem das soluções práticas. É esta

> a tarefa assumida pela análise numérica que, ao assumi-la, assume as limitações do cálculo prático, às quais a análise matemática tão elegantemente fugiu. As soluções numéricas são, portanto, apenas tentativas e, na melhor das hipóteses, tentativas precisas apenas dentro de certos limites calculáveis.

Deste modo, na concepção desta classe de autores, a análise numérica não se prende apenas com a construção de métodos numéricos mas, talvez principalmente, com a construção de limites calculáveis para os erros.

Nas matemáticas puras, esse conceito de número não exige representação concreta, individualizada, mas apenas simbólica; pelo contrário, em todos os ramos das matemáticas aplicadas (com os quais a análise numérica tem sempre, directa ou indirectamente, alguma coisa que ver) exige-se uma representação concreta de cada número particular, representação que, para distinguir do conceito abstracto de *número*, designamos por vezes por *numeral*. Em consequência, o número de números representáveis concretamente, embora por vezes indefinido (como sucede no cálculo manual, em que podem utilizar-se tantos algarismos quantos quisermos e em que, portanto, não existe um limite definido para a representação), é sempre necessariamente finito; pelo contrário, nas máquinas esse número é perfeitamente limitado e mesmo, na maior parte dos casos, surpreendentemente pequeno.

Três classes de números são muito importantes para a informática:

inteiros este números são normalmente utilizados nas operações de contagem e enumeração, mas a sua importância na informática digital é enorme; efectivamente o "mundo"de uma máquina digital é discreto e representável por números inteiros;

racionais números que são obtidos pelo quociente entre dois quaisquer números inteiros (excluindo o zero como denominador);

reais porque modelam o mundo observável contínuo, como á escala humana parece ser o nosso; a sua utilização é imposta conceptualmente e não operacionalmente.

Exercício 1.1 O leitor tem a certeza de que sabe distinguir *enumeração* de *contagem*? A distinção, embora provavelmente não fundamental em Análise Numérica, é-o certamente em Programação.

A representação de quantidades inteiras não é problemática. Já vimos que é a *forma natural* como os computadores digitais lidam com elas. Mas os números inteiros *informáticos* não correspondem ao conjunto **Z** da matemática: nem o seu domínio é infinito – no caso mais corrente cobrem apenas o intervalo de -32767 a $+32767$, nem as operações sobre eles definidas têm as mesmas propriedades[1]. Usam-se correntemente inteiros construídos sobre palavras de 16, 32 ou 64 bits.

A representação e manipulação de números reais por computadores digitais tem uma característica surpreendente: é impossível!

Com efeito, não podemos representar perfeitamente o contínuo recorrendo ao discreto. A única solução é recorrer a aproximações. No caso dos números reais é recorrer à sua aproximação por racionais.

Exemplo: 11.2 Racionais redux

Um exemplo simples mostra as deficiências dessa aproximação: suponha que o conjunto dos inteiros é constituído apenas pelos seguintes números:

$$\{-3,-2,-1,0,1,2,3\}$$

[1] basta pensar qual será a soma dos dois maiores números do domínio!

Exemplo: 11.2 Racionais redux (cont.)

Portanto, o conjunto de todos os racionais (obtidos pela razão entre dois inteiros) incluiria apenas os seguintes inteiros e fracções:

$$\left\{-\frac{3}{2},-\frac{2}{3},-\frac{1}{2},-\frac{1}{3},0,\frac{1}{3},\frac{1}{2},\frac{2}{3},\frac{3}{2}\right\}$$

Estão excluídas todas as infinitas fracções que não se podem escrever recorrendo ao inteiros entre -3 e 3, como $\frac{4}{5}, \frac{1}{7}, \frac{28}{29}$, além de todos os inteiros e fracções mais pequenos que -3 ou maiores que 3.

Figura 1.1: Números racionais possíveis para o domínio inteiro [-3,3]

1.2.1 Representação, Codificação

A forma de representar internamente a informação está estritamente condicionada pelas características físicas do computador, especialmente por trabalhar com circuitos electrónicos descendentes do transístor, que podem estar em um de dois estados, ligado ou desligado.

Como é registada internamente a informação? Como vimos, o computador trabalha sobre estados eléctricos dos seus componentes internos, normalmente duas voltagens de referência (+5 e -5 Volts, por exemplo). A uma entidade que só pode tomar um deste valores chamamos um bit (de binary digit, ou, porque não, de bit - pedaço, pedacinho). Um bit é também a menor quantidade de informação.

A correspondência entre estes dois estados e os valores lógicos de *verdade* ou *falsidade* é evidente.

No caso dos números poderíamos associar cada um desses dois estados a um dos dígitos binários, 0 e 1. Essa codificação binária não constitui qualquer transtorno para a aritmética...

Por outro lado, um alfabeto apenas com as letras A e B ou uma paleta apenas com branco e preto[2] – cor ou a sua ausência – seriam muito limitativas.

Os problemas que podem ser resolvidos recorrendo ao computador raramente podem ser expressos em termos de bits, sendo mais normal serem expressos em termos de caracteres, números, símbolos e mesmo objectos mais complicados.

```
000
001
010
011
100
101
110
111
```

Como utilizar então esta base tão limitada para representar objectos tão diferentes? A solução consiste em recorrer a regras de codificação, utilizando agrupamentos de bits, em vez de bits isolados. Um agrupamento de três bits têm oito estados diferentes, permitindo um alfabeto de oito letras ou uma paleta de oito cores. Uma simples contagem permite verificar que oito estados não são suficientes para conter o alfabeto ocidental (composto de um mínimo de 28 caracteres minúsculos, outros tantos maiúsculos, alguns sinais de pontuação e, já agora, os 10 dígitos da numeração árabe), pelo que se utilizam correntemente agrupamentos de oito bits que podem assumir um de 256 estados diferentes. A cada um desses estados associamos um código, conforme o que se pretende representar seja um número, um dígito, uma letra, uma cor, etc. Os sistemas de códigos não são únicos, existindo no entanto algumas convenções

[2] ou já agora laranja e verde!

Voltagem - +	representação binária	representação decimal	Número natural	Número inteiro[5]	Caracter ASCII
--------	00000000	0	0	0	NULL
--+----+	00100001	33	33	33	!
-+-----+	01000001	65	65	65	B
-+-++-+-	01011010	90	90	90	Z
-++++-+-	01111010	122	122	122	
+--++-++	10011011	155	155	-27	

Tabela 1.1: Codificação de números e caracteres

e estandardizações para nos facilitar a vida (uf!). Por exemplo, para a representação de caracteres alfanuméricos, utilizam-se correntemente dois standards, ambos de origem americana, os códigos ASCII[3] e ANSI[4].

Como se pode ver, todos estes códigos são apenas convenções. O mesmo valor binário pode ser interpretado de diferentes maneiras, sendo essa interpretação da inteira responsabilidade do utilizador.

1.2.2 Representação de quantidades

Para representar quantidades recorremos a símbolos, cujo significado é determinados por regras de codificação, podendo depender da colocação e do contexto. Historicamente, não só os símbolos mudaram, como, mais importante ainda, as técnicas de codificação.

São comuns técnicas posicionais e não posicionais de escrita de números.

Um exemplo de notação não posicional é a utilizada na numeração romana, em que cada símbolo tem sempre o mesmo valor.

MMCDLXIV

Os símbolos utilizados são as iniciais das palavras romanas que designavam as quantidades milhar **M**, meio milhar **D**, centena **C**, meia centena **L**, dezena **X**, meia dezena **V**, unidade **I**. Para calcular a quantidade representada por um número romano é preciso somar todos os dígitos que o compõem, tendo em atenção que eles se escrevem em ordem decrescente da esquerda para a direita, e que se existir uma inversão de valores, um dígito menor à esquerda de um maior, este deve ser subtraído ao valor do seguinte.

2464

Com a técnica de representação ocidental comum, utilizando uma simbologia composta apenas de dez símbolos diferentes (os símbolos 0 1 2 3 4 5 6 7 8 9), somos capazes de representar quantidades conceptualmente ilimitadas[6]. Recorremos para isso a uma técnica de notação posicional, em que o valor associado ao símbolo depende da sua posição no número. No caso dos números inteiros, o dígito mais à direita corresponde às unidades, o imediatamente à esquerda às dezenas, ...

$$(\ldots a_3a_2a_1a_0,a_{-1}a_{-2}\ldots)_b = \ldots + a_3b^3 + a_2b^2 + a_1b^1 + a_0b^0 +, a_{-1}b^{-1} + a_{-2}b^{-2} + \ldots;$$

[3] ASCII - American Standard Code for Information Interchange
[4] ANSI - American National Standards Institute
[6] As limitações são apenas espaciais e temporais.

Normalmente, b é um inteiro maior que 1 e os a são inteiros no intervalo $0 \leq a_k < b$. Os dígitos à esquerda são *mais significativos* que os dígitos à direita[7].

Como avaliar então um número escrito em notação posicional? Notação posici

O número é escrito como uma sequência ordenada de dígitos, em que, quanto mais à esquerda, maior o valor associado ao dígito.

A cada dígito será atribuído um valor igual ao índice da sua posição na lista de símbolos multiplicada pela base elevada ao índice do dígito no número (sendo positivo o índice dos dígitos da parte inteira e negativo o índice do dígito da parte não inteira), sendo no fim somados todos os valores obtidos (ver exemplo na tabela 1.2).

Exemplo: 11.3 Notação posicional em duas bases

	A8F		
Número na base 16			
Símbolos	A	8	F
Índice do símbolo	A	8	F
Índice do dígito	2	1	0
Valor do dígito	$A \times 16^2$	8×16^1	$F \times 16^0$
Valor do número	10×16^2	8×16^1	15×16^0
na base 10	2560	128	15
Número na base 10			2703

Tabela 1.2: Notação posicional do mesmo valor em base 16 e base 10

Este tipo de notação posicional é utilizável qualquer que seja a base de numeração (ou o número de símbolos diferentes que a simbologia nos permitir utilizar). São correntemente utilizadas na informática as bases 2, 8, 10 e 16 .

Base	Designação	Símbolos	101_{10}
2	binário	0 1	1100101
8	octal	0 1 2 3 4 5 6 7	145
10	decimal	0 1 2 3 4 5 6 7 8 9	101
16	hexadecimal	0 1 2 3 4 5 6 7 8 9 A B C D E F	65

Tabela 1.3: Bases Numéricas

Muitas outras bases foram utilizadas historicamente, essencialmente para contagem, umas decalcadas de características humanas, tal como a base dez, dos dez dedos das mãos[8], ou a base três, das três articulações dos dedos, outras com raízes desconhecidas tal como a base vinte usada pelos Maias[9], outras ainda, como a base sessenta da Mesopotâmia, que ainda hoje se utilizam na notação sexagesimal da medida de ângulos ou do tempo.

[7] ver uma descrição detalhada em [Knu81].
[8] duvidoso, já que a base adequada seria 5, por permitir agrupamentos de mãos
[9] ou a contagem de javalis em catorzenas na época de Obelix [GU99] – talvez a antecipação do actual $12+2$ grátis.

Exemplo: 11.4 Contar com as mãos

As mãos são utilizadas muitas vezes na aritmética como um dispositivo auxiliar de contagem, permitindo-nos contar de um a dez (ou recorrendo à famosa IBM do Manelinho[a], contar até vinte). O seu uso como um dispositivo de notação posicional é muito mais eficaz. Consegue imaginar como?

A mão pode ser usada como um dispositivo posicional adoptando regras muito simples: um dedo completamente dobrado corresponde a zero, o dígito menos significativo é o mindinho, o mais significativo é o polegar. Se usarmos uma convenção binária – o dedo todo dobrado vale zero, todo esticado vale um – uma mão permite representar números entre 0 e $2^5 - 1$; se utilizarmos cada uma das articulações dos dedos e do pulso, poderemos usar base 4.

Quão enganadora a afirmação "Tenho de meu apenas aquilo que posso contar com uma mão"!

(Radiografia em http://uwmsk.org/RadAnat/HandPALabelled.html)

[a] personagem das tiras desenhadas "Mafalda", de Quino, que calçava sandálias de modo a poder contar também com os dedos dos pés.

1.2.3 Conversão entre bases

Para efectuar uma conversão de bases recorre-se à divisão euclidiana, divisão com resto, divisão inteira ou modular.

se for A a representação do número na base a e B a sua representação na base b então cada um dos dígitos de B será obtido pelo resto da divisão de A por b, em que A é substituído pelo quociente Q da divisão anterior.

Exemplo: 11.5

Um exemplo de aplicação é a conversão de 100_{10} para a base 3 descrita na tabela 1.4.

Iteração	4	← 3	← 2	← 1	← 0
Inicial (base 10)					100
Quociente (base 10)	0	← 1	← 3	← 11	← 33
Resto	1	← 0	← 2	← 0	← 1
Resultado (base 3)					10201

Tabela 1.4: Conversão de 100_{10} para base 3

Repare-se que este algoritmo pode ser expresso quer como uma iteração quer como uma recursão.

1.2.4 Notação

A notação de números não inteiros é feita segundo duas técnicas: em *vírgula fixa*, em que a vírgula separa as unidades das décimas e em *virgula flutuante*, em que o valor dos dígitos à esquerda e à direita da vírgula é determinado por um expoente associado ao número.

Um número representado em vírgula fixa é composto de duas partes: a parte inteira, à esquerda da vírgula, e a parte decimal à direita da vírgula.

Um número representado em vírgula flutuante é composto de três partes: a *parte fraccional* (ou *mantissa*) m, o *expoente* (ou *característica*)e, e a *base* b:

$$mantissa \times base^{expoente}$$

A *mantissa* é por sua vez composta de uma parte inteira e de uma parte decimal.

Duas constatações simples permitem reduzir bastante a complexidade desta representação:

- uma operação simples de normalização consistindo em escrever o número de maneira a que a parte inteira da mantissa seja sempre nula e que todos os dígitos da parte decimal sejam significativos (não existam zeros imediatamente á direita da vírgula)

  $$1/b \leq |m| < 1$$

  permite retirar da representação a parte inteira da mantissa e a vírgula;

- a *base* é condicionada pela base do sistema de numeração utilizado, sendo constante dentro do mesmo sistema[10], pelo que a sua escrita é dispensável.

Desta maneira, a representação em vírgula flutuante fica reduzida a duas partes, a parte decimal da mantissa e o expoente, que podem ambas ser representadas como números inteiros (ver tabela 1.5).

fixa	flutuante	
100.10	$= 100.10 \times 10^0$	
	$= 1001 \times 10^{-1}$	
	$= 0.1001 \times 10^3$	normalizada
100 10	$= 1001\ 3$	simplificada

Tabela 1.5: Vírgula fixa e flutuante

Se nos abstrairmos da virgula, um número representado em vírgula fixa também fica reduzido a duas componentes representáveis como inteiros.

Qual é então a notação preferível para o cálculo numérico? O factor Domínio (ver tabela 1.6) faz com que a escolha penda normalmente para a vírgula flutuante.

A *precisão*, ou *comprimento*, n da mantissa de um número em vírgula flutuante é em geral determinada pelo comprimento da palavra do computador e pode, portanto, variar largamente de máquina para máquina. Por outro lado, é necessário pensar que nem todas as máquinas operam necessariamente sobre a representação binária dos números.

Com efeito, os circuitos electrónicos utilizados nos computadores têm apenas dois estados estáveis (correspondentes, por exemplo, a dois valores da tensão eléctrica), que se notam, por convenção, 0 e 1, de modo que a aritmética mais corrente nos computadores é binária; no entanto, é possível utilizar uma representação interna *decimal codificada em binário* (binary coded decimal ou *BCD*), representando em binário cada algarismo decimal de um número por meio de quatro bits.

Exemplo: 11.6 Codificação BCD

$3141_{(10)} = 0011_{(2)}0001_{(2)}0100_{(2)}0001_{(2)}$

[10] no sistema de base 10 $1.27 \times 10^2 = 12.7 \times 10^1$ mas $1.27 \times 5^2 \neq 12.7 \times 5^1$.

	Vírgula fixa	Vírgula flutuante
Representação	Parte inteira com N_i e parte decimal com N_d dígitos, $\pm iiii, dddd$	Mantissa com N_m e expoente com N_e dígitos, $\pm 0, mmmm \times b^{\pm eeee}$
Para permitir a comparação, façamos $N = N_i + N_d = N_m + N_e$ (os números ocupam o mesmo espaço) e utilizemos a mesma base B.		
Domínio	depende de N_i, sendo aproximadamente $]-B^{N_i+1}, +B^{N_i+1}[$	depende de N_e, sendo aproximadamente $]-B^{B^{N_e}-1}, +B^{B^{N_e}-1}[$
Granularidade	é constante, valendo B^{-N_d}	é variável, valendo B^{p-N_m-1} (em que p é a potência de dois números consecutivos), podendo por isso tomar valores no intervalo $[B^{-B^{N_e}-N_m-1}, B^{B^{N_e}-N_m-1}]$ com um buraco em torno de zero de $2 \times B^{-N_e}$ o espaçamento relativo entre dois números consecutivos é constante e vale $B^{p-N_m-1}/B^p = B^{-N_m-1}$

Tabela 1.6: Comparação de notações em vírgula fixa e flutuante

Exemplo: 11.7 A MAKINA

Imagine uma máquina (que não existe, mas que vamos utilizar intensivamente daqui em diante) capaz de fazer cálculos em virgula flutuante normalizada, mas com capacidade de representação limitada, sendo a realização e registo das operações da responsabiliodade do operador.
Esta **MAKINA** trabalha com uma representação interna da forma: $\pm m \times 10^{\pm p}$ com $0.1 \leq m < 1$, com três dígitos na parte fracionária da mantissa e um dígito de expoente.
O seu interface tem à esquerda uma coluna apenas dedicada à indicação da operação , ao meio um conjunto de quatro colunas destinadas à escrita da mantissa, um número de três dígitos com sinal , e à direita um outro conjunto de duas colunas para escrever o expoente, um dígito e respetivo sinal ; tem ainda uma coluna dedicada a comentários .
Neste exemplo mostra-se como:

- escrever o valor de $\pi \approx 3,1416$
- calcular $10+11=21$, representada em vírgula flutuante normalizada:

$$0,100 \times 10^1 + 0,110 \times 10^1 = 0,210 \times 10^1$$

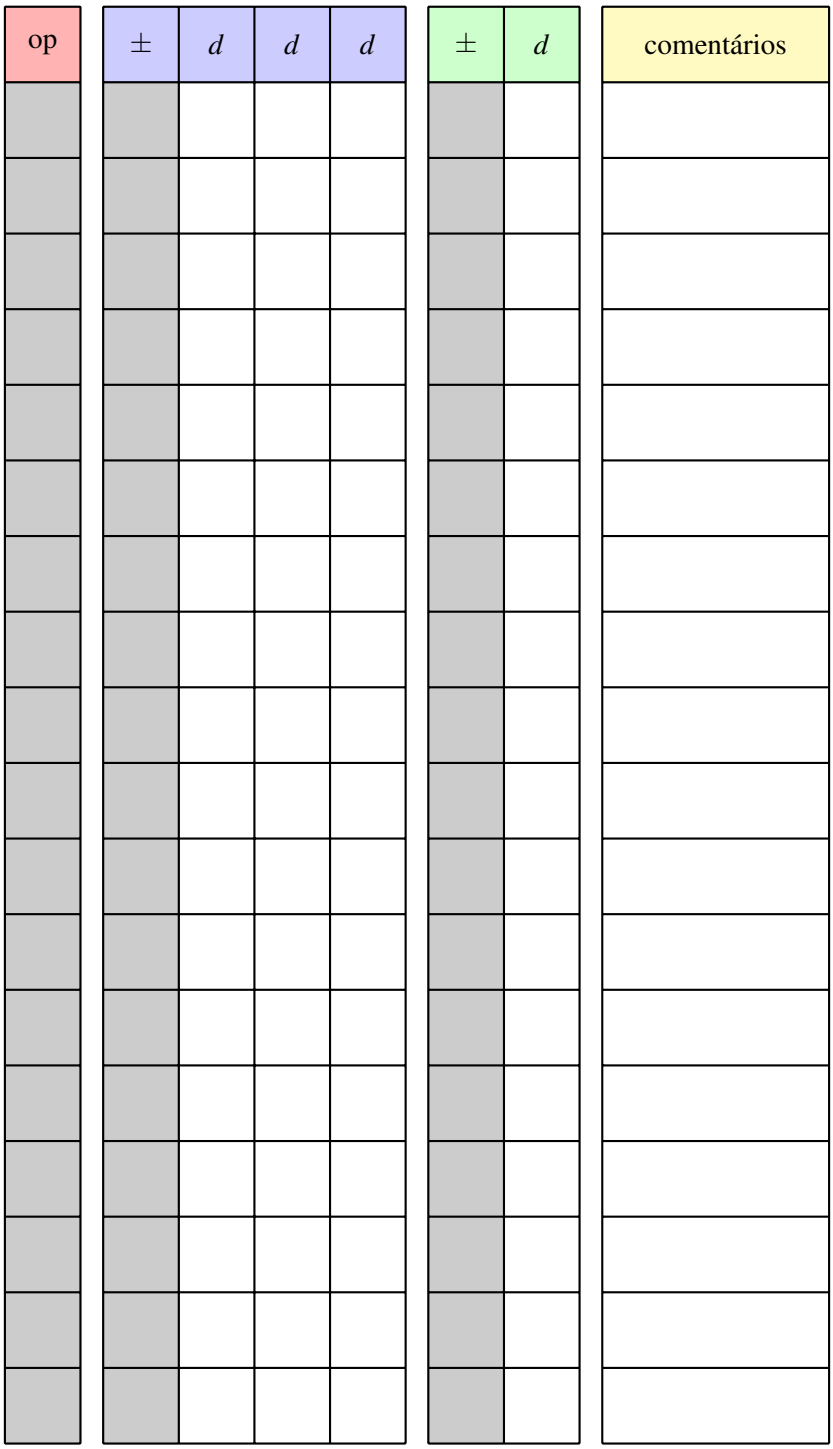
- calcular 74,2 + 92,7 = 166,9

op	±	*d*	*d*	*d*	±	*d*	comentários
π	+	3	1	4	+	1	0.314×10^1
	+	1	0	0	+	1	10
+	+	1	1	0	+	1	+11
=	+	2	1	0	+	1	=21
	+	7	4	2	+	2	74,2
+	+	9	2	7	+	2	92,7
=	+	1	6	6	+	3	166,9 !!!

Exercício 1.2 Reportando-se ao exemplo anterior, procure responder:

1. Nesta **MAKINA**, existem tantos números entre 0.100×10^{-3} e 0.999×10^{-3} como os que há entre 0.100×10^3 e 0.999×10^3, quando é perfeitamente evidente que este último intervalo é muito maior. Quantos são esses números?
2. Qual a diferença entre dois números consecutivos, em cada um dos casos?
3. Qual é o menor intervalo entre números representáveis sucessivos?
4. e o maior?

Uma medida da máxima precisão atingida pelo cálculo em vírgula flutuante é dado pela *unidade da*



.

Figura 1.2: A **MAKINA** para fazer cálculos.

última casa (u), definida como o menor número positivo tal que $u+1>1$.

Exemplo: 11.8 Unidade da última casa

O exemplo seguinte reporta-se a um tempo em que as implementações da representação em virgula flutuante não eram *nada* normalizadas, e em que, para conhecer os limites de um número era preciso mergulhar fundo no manual ou então experimentar ...

O seguinte programa BASIC determina o valor de u em aritmética de precisão simples, diminuindo o valor de $u+1$ até não ser reconhecido pelo computador como maior que a unidade:

```
U=0.00001
DO
U=0.9999*U
LOOP WHILE (U+1) > 1
PRINT "U = ";U
```

Corrido em um PC-AT em BASICA em precisão simples dá $u = 5.96E-8$, isto é, 24 bits na mantissa, mas corrido em TURBOBASIC dá $u = 2.71E-20$, isto é, 64 bits na mantissa. Com efeito, o valor da unidade da última casa relaciona-se directamente com o número de bits usados na representação dos números em vírgula flutuante. Os PC usam 32 bits (4 bytes) para os números em precisão simples, dos quais 24 são atribuídos à mantissa, incluindo o sinal, e 8 ao expoente, incluindo também o sinal. Assim, o valor de u será $u_s = 2^{-24} = 5.96 \times 10^{-8}$ Em dupla precisão, o PC usa 64 bits (8 bytes) para representar os números em vírgula flutuante, dos quais 56 para a mantissa e 8 para o expoente. O valor de u em dupla precisão será então $u_d = 2^{-56} = 1.39 \times 10^{-17}$.

Exercício 1.3

1. Assegure-se que compreendeu claramente o funcionamento do algoritmo; em caso de necessidade, execute-o em modo "trace".
2. Qual será o resultado na **MAKINA**?
3. Se é verdade o que acima se disse sobre o PC, como é possível que o TURBOBASIC tenha o comportamento descrito?
4. Como procederia para determinar a unidade da última casa em uma máquina de calcular não-programável?
5. Seria capaz de escrever um programa BASIC para determinar a unidade da última casa em dupla precisão?
6. E em outra linguagem, para simples e dupla precisão?
7. E um programa para determinar o maior número representável?

Assim, o sistema de vírgula flutuante é nitidamente diferente dos sistemas habituais dos números da matemática, e um utilizador deve ficar prevenido de que essa diferença pode, por vezes, produzir efeitos surpreendentes.

Uma maneira possível de olhar os números da computação consiste em pensar que cada resultado de um cálculo surge, na representação computacional, um pouco preciso ou esborratado em torno do valor que matematicamente *deveria* ter. Este ponto de vista enquadra-se bem com a ideia que correntemente temos do uso prático dos números mesmo fora do contexto computacional (quando dizemos `uma tonelada`

de areia nunca pretendemos seriamente que seja a quantidade exacta até ao último grão e, mesmo que o fosse, a precisão não seria absoluta, sendo apenas da ordem de 10^{-9}). Um outro ponto de vista possível consiste em pensar que os únicos números existentes são os representáveis na máquina e que não existem mais nenhuns, o que corresponde a considerar que os números reais dos matemáticos são, para o programador e o analista numérico, inteiramente fictícios. Este ponto de vista pode parecer um tanto drástico, mas tem-se revelado extremamente útil ao analista numérico, nomeadamente quando tem que identificar os mecanismos que conduziram a um resultado errado ou simplesmente surpreendente.

Tipicamente, a finitude do intervalo coberto pelo expoente p pode dar origem aos acidentes conhecidos por *underflow* e *overflow* e que correspondem ao facto de o resultado de uma operação exceder o máximo ou o mínimo dos números representáveis na máquina. É relativamente comum - mas perigoso - pensar que não haverá mal em substituir um *underflow* por zero, salvo se, a seguir, entrar como divisor em uma divisão. Pelo contrário, em termos de senso comum, parece perigoso substituir um *overflow* pelo maior número representável na máquina (que não é, em geral, espantosamente grande) e, no entanto, muitos compiladores para uso científico fazem-no, sem sequer avisar o utilizador, para lhe poupar numerosas mensagens de erro ou paragens intempestivas do programa. Esta possibilidade não deve nunca ser perdida de vista, e um utilizador cuidadoso investiga, antes de mais nada, o comportamento de um compilador ou interpretador perante os *overflow* e *underflow* a fim de poder tomar as precauções adequadas para evitar surpresas desagradáveis.

Exercício 1.4 A linguagem de programação **PASCAL** define a primitiva **MAXINT**, que permite conhecer qual o maior valor inteiro positivo que a implementação concreta permite. Primitivas que permitem conhecer o *comprimento* de um objecto, permitem obter informação semelhante, desde que se saiba a *unidade de medida*! Os mesmos mecanismos não são aplicáveis em relação à representação em vírgula flutuante. Procure identificar, para as linguagens de programação que conhece, quais os mecanismos disponíveis para conhecer os limites das representações de números.

Exemplo: 11.9 Overflow

Na **MAKINA**, o maior número representável seria $0.999 \times 10^9 \approx 10^9$ enquanto o menor (em valor absoluto, com exclusão do zero) seria $0.100 \times 10^{-9} = 10^{-8}$.
Quando é que a função exponencial daria *overflow* na nossa máquina hipotética?
Quando fosse

$$\exp(x) > 0.999 \times 10^9$$

isto é, tomando logaritmos,

$$x > \ln 0.999 + 9 \times \ln 10 = -0.001 + 9 \times 2.303 = 20.7$$

os valores calculados com maior precisão seriam

$$\begin{aligned} -0,&00100050033358353350014298225406 83 \\ &+ 9 \times 2,30258509299404568401799145468 44 \\ &= 20,72226533661282762266178010990 5 \end{aligned}$$

Exercício 1.5 Qual é o valor correspondente para:

1. a sua máquina de calcular?

2. para as diferentes linguagens de programação, usadas com números em vírgula flutuante, em precisão simples e dupla ?
3. neste caso, dependerá da linguagem?

O raciocínio parece perfeitamente simples e claro; tem a certeza de que é geralmente correcto?

No entretanto, temos o direito de nos interrogar porque é que um intervalo representável que, nas piores condições, nunca será inferior a

$$\left[(-10^{38},-10^{-38}),(10^{-38},10^{38})\right]$$

pode com tanta frequência dar origem a problemas de *overflow* ou de *underflow*, quando rarissimamente os resultados dos problemas excedem os limites desses intervalos. A resposta óbvia é a de que tais acidentes surgem habitualmente nos resultados de cálculos intermédios. Assim, a precisão do resultado de um cálculo numérico fica altamente dependente do caminho para a ele chegar, isto é, do algoritmo adoptado para o implementar, o que leva o analista numérico a ter que tomar precauções que são inteiramente desconhecidas para o matemático convencional (nas matemáticas convencionais, a única situação comparável que nos ocorre é a da soma de séries simplesmente convergentes, que depende da ordem das parcelas).

Exemplo: 11.10 overflow e underflow

No cálculo de

$$1-\frac{1}{e^x+1}$$

para $x \geq 20.7$ (caso em que o valor da expressão seria 1 quando representado com a precisão da nossa máquina) teríamos *overflow*. A substituição do valor impossível pelo maior número representável na máquina daria o resultado final esperado, 1. Por outro lado, se utilizássemos a expressão matematicamente equivalente

$$\frac{1}{1+e^{-x}}$$

teríamos *underflow* e a sua substituição por zero daria também como resultado final o 1 esperado.

Exercício 1.6

1. Quantos números diferentes podem ser representados em vírgula flutuante na **`MAKINA`**?
2. Explique o sentido da expressão "*nos computadores, o número 0 está relativamente isolado dos seus vizinhos*".
   a) Os seus vizinhos está-lo-ão igualmente entre si?
   b) Que influência tem o comprimento da mantissa no fenómeno?
   c) e o comprimento da característica?
3. Em termos comparativos, que se passa com o maior número representável?
4. Estabeleça as semelhanças e diferenças entre o uso da representação de números em vírgula flutuante e a representação logarítmica.

1.2.5 Representação IEEE

A representação informática de números em vírgula flutuante foi normalizada pelo organismos IEEE e ISO, com a norma IEEE 754[11] [IEE19] , definindo regras de representação e de cálculo. A definição inicial de 1985 já sofreu várias alterações, sendo a mais actual de 2019..

O essencial da norma diz que um número em vírgula flutuante $mantissa \times 2^{expoente}$ deverá ser escrito na forma:

$$\pm 1.\mathtt{f} \times 2^{\mathtt{e}}$$

em que `f` é a parte fraccionária da mantissa, e sendo `e` obtido por

$$e = \text{expoente} + \text{viés}$$

O padrão define dois tipos:

Número em precisão simples Usa 32 bits repartidos da seguinte forma:

- bit `0`, um bit de sinal;
- bit `1` a `8`: oito bits de expoente (em excesso de 127);
- bit `9` a `31`: vinte e três bits para a parte fraccionária da mantissa, normalizada de maneira à parte inteira ser unitária;

Número em precisão dupla Usa 64 bits repartidos da seguinte forma:

- bit `0`, um bit de sinal;
- bit `1` a `11`: onze bits de expoente (em excesso de 1023);
- bit `12` a `63`: 52 bits para a parte fraccionária da mantissa, normalizada de maneira à parte inteira ser unitária;

> Na apresentação que se segue discutiremos apenas os números em precisão simples. A discussão dos números em dupla precisão apenas necessita de ser ajustada ao seu maior tamanho.

O padrão prevê os seguintes valores especiais:

- $+0$, -0 fazendo $e = 0$, com o sinal adequado;
- $-\infty$, $+\infty$ fazendo $e = 255$, com o sinal adequado;
- **`NaN`** fazendo $e = 255$ e $f \neq 0$; ***Not a Number*** é o resultado, por exemplo, da avaliação de $0/0$.

Como o expoente `e` está limitado ao intervalo `]0,255[`,
a potência do número $p \in [-126, +127]$ pelo que o menor número positivo em precisão simples é

$$\begin{aligned} 1.00000000000000000000000 \times 10_2^{-01111110} \\ = 2_{10}^{-126} \approx 1.2 \times 10^{-38} \end{aligned}$$

e o maior número positivo em precisão simples é

$$\begin{aligned} 1.11111111111111111111111 \times 10_2^{+11111110} \\ = (2 - 2^{-23}) \times 2_{10}^{+127} \approx 3.4 \times 10^{+38} \end{aligned}$$

[11] Publicada originalmente pelo Institute of Electrical and Electronics Engineers. O seu texto pode ser lido aqui: https://ieeexplore.ieee.org/document/8766229 existindo uma uma explicação adequada na Wikipedia em https://en.wikipedia.org/w/index.php?title=IEEE_754&oldid=917946439

Precisão

Qual a precisão numérica desta representação? O bit menos significativo de `f` tem peso 2^{-23}, pelo que uma mudança nesse bit causa uma variação em `f` igual a 2^{-23}. Consideremos um intervalo $[x_1, x_2]$, definido por dois números representáveis exactamente em precisão simples IEEE e consecutivos, por exemplo:

$$x_1 = 1.00000000000000000000000 \times 10_2^p$$
$$x_2 = 1.00000000000000000000001 \times 10_2^p$$

Como x_1 e x_2 são *números máquina*, representáveis exactamente na notação escolhida, e são consecutivos, não pode ser representado nenhum outro número entre eles. Qualquer quantidade entre x_1 e x_2, ou é representada por x_1 ou por x_2 [12]. Qualquer que seja a escolha, o erro cometido não excederá 2^{-23} no valor de f e 2^{-23+p} no valor do próprio número.

Se x for o número máquina

$$x = m \times 2^p$$

então o número máquina imediatamente superior é

$$x = (m + 2^{-23}) \times 2^p$$

e o número máquina imediatamente inferior é

$$x = (m - 2^{-23}) \times 2^p$$

A diferença (espaçamento) entre os números é

$$x_+ - x = x - x_- = 2^{-23} \times 2^p = 2^{-23+p}$$

sendo o espaçamento relativo entre x_+ e x

$$(x_+ - x)/x \approx 2^{-23} \approx 1.2 \times 10^{-7}$$

O espaçamento relativo é uma constante, aproximadamente igual a 2^{-23}, pelo que pode ser usado como uma medida da precisão de representação.

Arredondamento

O *arredondamento* ocorre sempre que a quantidade a representar não seja um *número máquina*, não sendo representável exactamente, o que obriga a escolher um dos dois *números máquina* enquadrantes. O modo de arredondamento padrão é *arredondamento para o mais próximo*, com *arredondamento para par* (para o número que tem um `0` no seu algarismo menos significativo) em caso de indecisão. Esta técnica de arredondamento garante um erro máximo de meia unidade do algarismo menos significativo.

Valores especiais

Os valores *infinito*, *NaN*, e *zero* tem representação e tratamento especiais:

[12] consoante a técnica de arredondamento usada.

Infinito é tratado como um número muito grande, sempre que tal fizer sentido; por exemplo, sendo x um número positivo em virgula flutuante:

$$\begin{aligned} +\infty + x &\rightarrow +\infty \\ +\infty \times x &\rightarrow +\infty \\ x/+\infty &\rightarrow +0 \\ +\infty/x &\rightarrow +\infty \end{aligned}$$

o mesmo se aplica ao caso de x negativo e de $-\infty$;

NaN Este valor é usado no caso de operações de resultado indeterminado:

$$\begin{aligned} 0/0 &\rightarrow \texttt{NaN} \\ (+\infty)-(+\infty) &\rightarrow \texttt{NaN} \\ x+\texttt{NaN} &\rightarrow \texttt{NaN} \end{aligned}$$

zero O padrão prevê duas formas de zero:

$+0$ que é o resultado da maioria das operações de que resulta zero;

-0 que é usado como indicador de um número negativo muito pequeno, inferior à precisão máquina.

Excepções

O padrão define vários tipos de excepções detectáveis, de que importa destacar o *overflow*, o *underflow* e *inexact*, que ocorre quando um resultado deixou de ser exacto em consequência de arredondamentos.

Exercício 1.7 Na definição do padrão IEEE diz-se que a normalização do número deve ser tal que satisfaça

$$\pm 1.\mathtt{f} \times 2^{\mathrm{e}}$$

enquanto que anteriormente tinhamos sugerido que deveria ser

$$\pm 0.\mathtt{f} \times b^{\mathrm{e}}$$

sendo neste caso o algarismo mais significativo de `f` diferente de zero. Porquê ?[13]

[13] Pense na base!

1.3 Erros

A representação finita (aproximada) de números em um dispositivo computacional (calculador humano, máquina de calcular mecânica ou electrónica, computador digital) conduz inevitável e sistematicamente ao aparecimento de *erros de arredondamento* [14], enquanto a representação finita de processos conduz ao aparecimento de *erros de truncatura* [15]; esta é a nomenclatura convencional, no entanto, dado que toda a representação de um número é um processo característico do cálculo numérico, e, portanto, enquanto processo finito é um processo de truncatura; assim, arredondamento e truncatura são, em termos práticos, essencialmente a mesma coisa.

O *erro absoluto* pode ser definido como a diferença entre o valor exacto de uma quantidade e o seu valor aproximado:

$$e(x) = x_{exacto} - x_{aproximado}$$

O *erro relativo* é a relação

$$r(x) = \frac{e(x)}{x_{exacto}}$$

Observe-se que, das duas equações, se pode deduzir

$$x_{exacto} = \frac{x_{aproximado}}{1 - r(x)}$$

expressão que, caso se conheça um majorante para o erro relativo e que este seja suficientemente pequeno, nos permite obter um intervalo enquadrante do valor exacto. [16]

1.4 Arredondamento e Truncatura

Quando se procede à *multiplicação* de dois números com três casas decimais, o produto tem seis casas decimais; se se quiser continuar a ter apenas as três casas decimais dos dados, é de boa prática proceder a um *arredondamento*, isto é, juntar 5 à primeira casa decimal a abandonar, antes de a abandonar; o resultado é que, se essa primeira casa a abandonar for 5 ou superior, teremos um aumento de uma unidade na última casa conservada (*arredondamento para cima*); se, porém, for inferior a 5, o valor da última casa conservada manter-se-á inalterado (*arredondamento para baixo*).

Deste modo, qualquer que seja o caso, o valor arredondado será mais próximo do valor exacto que o que obteríamos mediante uma simples truncatura, isto é, eliminando pura e simplesmente as casas decimais em excesso. Seja como for, cometeremos, em geral, pequenos erros: *erro de arredondamento* e *erro de truncatura*. A importância da pequena diferença entre os dois erros resulta da possibilidade de vir a acumular-se em operações sucessivas e de vir a ser amplificada pelo efeito de operações posteriores, como veremos.

Exemplo: 11.11 Multiplicação

Multiplicando dois valores

[14] erro devido à aproximação computacional
[15] erro devido à aproximação matemática
[16] Note-se que a dificuldade em satisfazer essas duas exigências torna a expressão praticamente inútil!

Exemplo: 11.11 Multiplicação (cont.)

$$\begin{array}{r} 0.236\times10^{1} \\ \times\ 0.127\times10^{2} \\ \hline 1652 \\ 472 \\ 236 \\ \hline 0.029972\times10^{3} \end{array}$$

Normalizando e calculando o valor para arredondar

$$\begin{array}{r} 0.29972\times10^{2} \\ +\,0.0005\ \times10^{2} \\ \hline 0.30022\times10^{2} \end{array}$$

Podemos verificar na tabela seguinte os valores obtidos por arredondamento e por truncatura e quais os erros cometidos.

×	arredondamento	truncatura
Valor	0.300×10^{2}	0.299×10^{2}
Erro absoluto	-0.280×10^{-1}	0.720×10^{-1}
Erro relativo	0.934×10^{-3}	0.240×10^{-2}

A mesma situação ocorre, naturalmente, na *divisão*. Note-se, de passagem, que as mesmas duas técnicas podem ser utilizadas na conversão de um dado (em geral sob forma decimal) na base própria da máquina.

Na própria *adição* podem ocorrer erros de arredondamento e de truncatura, mesmo quando as características dos números a adicionar são as mesmas, desde que possa haver transporte da casa mais significativa; no caso de os dois números serem de ordens de grandeza muito diferentes, a adição pode nem sequer ter qualquer efeito, no sentido de que o resultado será, pura e simplesmente, igual ao de maior valor absoluto.

Exemplo: 11.12 Adição

Somando dois valores

$$\begin{array}{r} 0.742\times10^{2} \\ +\ 0.927\times10^{2} \\ \hline 1.669\times10^{2} \end{array}$$

Normalizando e calculando o valor para arredondar

$$\begin{array}{r} 0.1669\times10^{3} \\ +\,0.0005\times10^{3} \\ \hline 0.1674\times10^{3} \end{array}$$

Podemos verificar na tabela seguinte os valores obtidos por arredondamento e por truncatura e quais os erros cometidos.

+	arredondamento	truncatura
Valor	0.167×10^{3}	0.166×10^{3}
Erro absoluto	0.100×10^{0}	0.900×10^{0}
Erro relativo	0.599×10^{-5}	0.539×10^{-4}

No caso de dois números de grandeza diferente, as *mantissas* não podem adicionar-se directamente, pelo que se torna necessária uma operação prévia de *alinhamento* que consiste em reduzir os dois números ao mesmo expoente; após a realização da adição das *mantissas*, o resultado é arredondado, o que implica perda de precisão que pode ir ao ponto de a soma ser igual à maior das parcelas, o que, em si, não será extremamente grave, salvo no caso de adições encadeadas, como sucede na soma de séries.

Exemplo: 11.13 Soma com perda de parcela

Para efectuar a soma:

$$0.742 \times 10^2 + 0.927 \times 10^{-2}$$

é necessário efectuar primeiro o alinhamento de expoentes

$$\begin{array}{r} 0.742 \quad \times 10^2 \\ +\ 0.0000927 \times 10^2 \\ \hline 0.7420927 \times 10^2 \end{array}$$

O valor renormalizado é 0.742×10^2

Porém, onde os erros deste tipo causam mais problemas é na *subtracção* de números da mesma ordem de grandeza: em consequência da anulação dos primeiros dígitos da mantissa, esta é deslocada para a esquerda até ao primeiro dígito não nulo, o que pode trazer bem para a esquerda os erros de truncatura e/ou de arredondamento. A rigor, neste caso, não há que falar em erro de arredondamento nem de truncatura, mas sim em erro intrínseco da própria técnica operatória quando realizada sobre dados arredondados ou truncados.

Exemplo: 11.14 Subtracção

$$\begin{array}{r} 0.314 \times 10^1 \\ -\ 0.313 \times 10^1 \\ \hline 0.001 \times 10^1 \end{array}$$

O valor renormalizado é 0.100×10^{-1}
Erros absoluto e relativo da operação: 0
Poderia pensar-se que, tal como no caso da adição, este efeito não tenha importância excepcional; porém, salvo se os valores dos termos forem exactos (isto é, dados iniciais ou resultados parciais sem erro) os dois últimos zeros da mantissa são artificiais e nem sequer temos a certeza do 1 que os antecede; com efeito,

- se o aditivo exacto fosse 0.3135×10^1 e o subtractivo exacto fosse 0.3134×10^1 , o resultado correcto seria 0.1×10^{-2} , um décimo do valor calculado 0.1×10^{-1};
- se o aditivo exacto fosse 0.3144×10^1 e o subtractivo exacto fosse 0.3125×10^1, o resultado correcto seria 0.19×10^{-1} , dezanove vezes o valor calculado.

Exercício 1.8

1. calcule os erros relativos de um e outro resultados e compare-os com os dos termos da subtracção;
2. repita o raciocínio para o caso de dois termos iguais, tais que o resultado calculado da subtracção é nulo (este é o caso extremo que dá ao fenómeno o nome de cancelamento); imagine as consequências se este resultado entrasse em seguida como factor de uma multiplicação por um número grande o como divisor em uma divisão;

3. seria capaz de adaptar os exemplos à **MAKINA**?

Este fenómeno de *cancelamento* constitui a principal fonte do erro numérico em cálculo científico, não criando erros por si próprio, mas actuando como um amplificador de erros preexistentes. Na maior parte dos cálculos instáveis, criam-se erros numéricos significativos por acumulação de arredondamentos e esses erros são depois amplificados pelo cancelamento. Esta situação é de tal modo frequente e grave que Lieberstein [Lie68] se refere a ela como

> a dificuldade patológica da análise numérica: enormes perdas de precisão resultantes da subtracção de representações finitas de números grandes quando a sua diferença é pequena.

Exemplo: 11.15 I.N.Génuo

Retomemos o caso do senhor I.N. Génuo e calculêmo-lo usando uma folha de cálculo, com precisões sucessivamente crescentes. Os resultados obtidos foram os seguintes:

algarismos significativos	capital acumulado
3	-4,375E+21
4	2,8188E+20
5	-2,83618E+19
6	2,660808E+18
8	2,39020731E+16
10	6,3525795660E+14
12	-7,016551748043E+11
13	8,4946582952843E+11
14	7,39053273591465E+10
15	7,390532735914650E+10
16	7,3905327359146500E+10

Não se entusiasme com a aparente estabilização (numa considerável fortuna!). Acontece que a folha de cálculo utilizada (MS Excel) tem uma precisão máxima de 14 algarismos significativos! Se pretender explorar o exemplo, pode recorrer ao ficheiro Ingenuo.xls .

Exercício 1.9 Calcule o problema do senhor I.N. Génuo com precisões ainda maiores. Investigue se o valor tenderá realmente a estabilizar.
(sugestão: utilize um manipulador numérico ou algébrico com precisão estipulável, como o **derive** ou o **Maxima**)

Exemplo: 11.16 Transformação de expressões

No cálculo de $(x+1)^{1/2}-x^{1/2}$ não há problemas de maior quando x é pequeno, mas pode ocorrer importante perda de precisão quando x é grande. Neste caso, pode com vantagem proceder-se à transformação

$$\frac{[(x+1)^{1/2}-x^{1/2}].[(x+1)^{1/2}+x^{1/2}]}{(x+1)^{1/2}+x^{1/2}} = \frac{1}{(x+1)^{1/2}+x^{1/2}}$$

Exemplo: 11.16 Transformação de expressões (cont.)

Podemos verificar os resultados executando a seguinte sequência de comandos no **maple**:

```
> Digits:=5;
```

$$Digits := 5$$

```
> f(x) := (x+1)^(1/2)-x^(1/2);
```

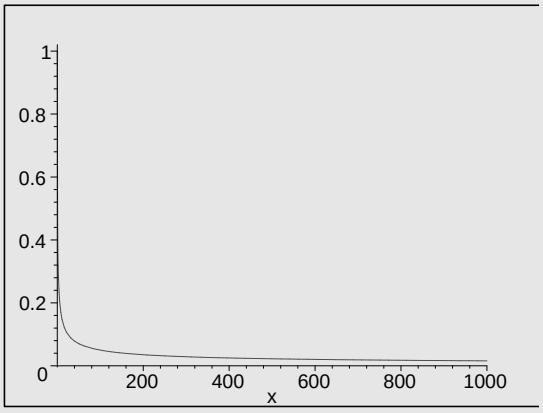
$$f(x) := \sqrt{x+1} - \sqrt{x}$$

```
> f1(x):= 1/((x+1)^(1/2)+x^(1/2));
```

$$f1(x) := \frac{1}{\sqrt{x+1} + \sqrt{x}}$$

Gráficos das duas funções

```
> plot(f(x) ,x= 0..  + 10^3);
> plot(f1(x),x= 0..  10^3);
```



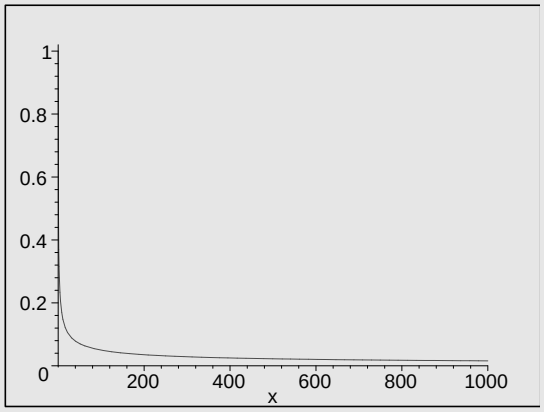
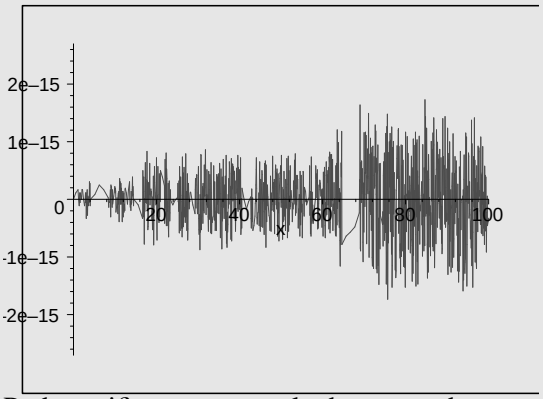


Gráfico da diferença das duas funções

```
> plot(f(x) -f1(x),x= 0..  + 1.0E2);
```



Pode verificar estes resultados e estudar a sua variação com o número de dígitos de precisão e com o valor de x usando o ficheiro Perda de Precisao.mws .

Uma vez cometido um erro de arredondamento no resultado de uma operação, esse erro vai contaminar os dados das operações subsequentes segundo um mecanismo de *propagação do erro*, mecanismo que depende do encadeamento particular de operações em cada caso. Existem, naturalmente, numerosos artifícios para diminuir os efeitos perversos do erro de arredondamento. A dificuldade principal é, porém, a de prever quando e onde é que o arredondamento vai criar problemas.

Um bom exemplo do tipo de artifício comparativamente elementar é o relativo à adição (em vírgula flutuante) de um número moderadamente elevado de números relativos: se nos limitarmos a adicioná-los

sucessivamente, arriscamo-nos a acumular muitos arredondamentos. Uma maneira de o evitar consiste em separar os somandos em duas classes, a dos positivos e a dos negativos, ordenando cada uma do maior para o mais pequeno; começa-se então a adição pelo mais pequeno de uma das classes, juntando-lhe sucessivamente membros da outra classe até que o sinal do resultado mude; a cada mudança de sinal do resultado passa-se à outra classe para tomar o número seguinte (imediatamente superior); deste modo, a soma acumulada é mantida tão pequena quanto possível; quando, finalmente, uma das classes se esgota, os números da restante podem ser adicionados ao resultado parcial sem cuidados especiais. Obviamente, este método exige esforço importante na separação e ordenação dos somandos de modo que, sempre que uma máquina disponha de representação de números em dupla precisão, tende a preferir-se o uso desta.

Para a pessoa habituada a calcular à mão com 4 ou 5 casas decimais, um erro de arredondamento na 8^a casa decimal, ou mesmo mais além, pode parecer coisa trivial e inofensiva ou até mesmo excelente, mas o facto é que os enormes encadeamentos de cálculos característicos da computação digital automática podem acarretar acumulações substanciais desses erros. O facto de ser praticamente impossível prever os efeitos de erros de arredondamento e/ou truncatura ao fim de longas e complexas cadeias de cálculos faz com que os espíritos simplistas considerem o processo de acumulação de erros como aleatório, argumentando que a lei dos grandes números torna aproximadamente iguais os casos de erros positivos e negativos, que assim tenderiam a compensar-se. A aparência altamente científica deste argumento não chega para esconder os factos de:

1. a experiência revelar que a existência de correlações (isto é, de dependências) internas entre os sucessivos resultados parciais de um cálculo científico complexo é muito mais comum que o que correntemente se pensa; tais correlações fazem com grande frequência que os erros ocorram dominantemente em um de dois sentidos, o que leva a uma efectiva acumulação;
2. aquilo que, no fundo, está em questão não é a probabilidade de um erro grande (que pode ser, efectivamente, pequena) mas a esperança matemática do custo que lhe está associado (isto é, a soma dos produtos das probabilidades de cada erro pelos respectivos custos), que pode ser insuportável.

Nestas condições, todo o cuidado é pouco nesta matéria e as belas folhas de números impressos pelo computador só são razoavelmente fiáveis quando forem tomadas todas as precauções possíveis. É este facto que nos leva, na perspectiva da computação automática, a consagrar tanto espaço de um curso de análise numérica ao problema dos erros.

Exemplo: 11.17 Expressões factorizadas e expandidas

Consideremos a expressão

$$\begin{aligned}(5x-14)^2(x^2+3) = & \\ &= 25x^4-140x^3+271x^2-420x+588 \\ &= 25x^4+271x^2+588-140x^3-420x\end{aligned}$$

e vejamos como o **maple** calcula estas diferentes expressões alternativas com 11 algarismos significativos nas vizinhanças da única raiz positiva, $x = 2.8$.

```
> Digits:=11;
```

$$Digits := 11$$

```
> par:= (x,y)->[x,y];
```

$$par := (x, y) \rightarrow [x, y]$$

Construção das expressões:

```
> ff:= x->(5.0*x-14.0)^2*(x^2+3.0);
> fexp:= expand(ff(x) );
```

Exemplo: 11.17 Expressões factorizadas e expandidas (cont.)

```
> fd1 := x->25.0*x^4+271.0*x^2+588.0-140.0*x^3-420.0*x;
```

$$ff := x \to (5.0x - 14.0)^2 (x^2 + 3.0)$$

$$fexp := 25.00x^4 + 271.000x^2 - 140.00x^3 - 420.000x + 588.000$$

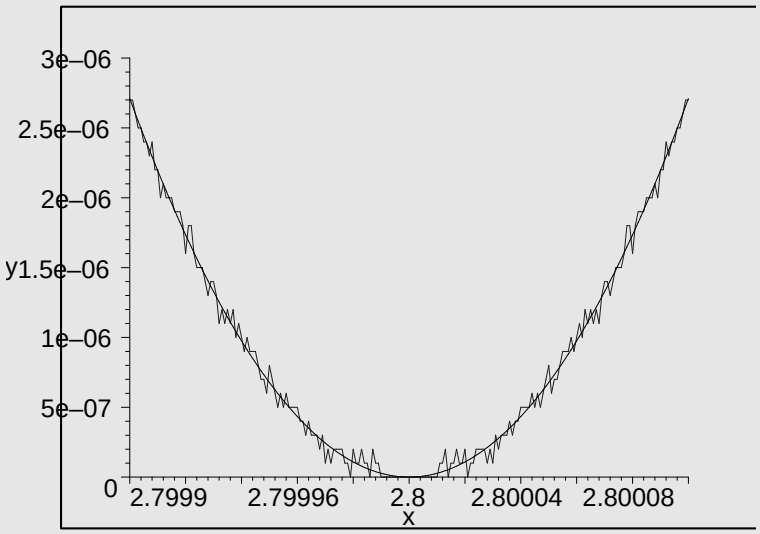
$$fd1 := x \to 25.0x^4 + 271.0x^2 + 588.0 - 140.0x^3 - 420.0x$$

Construção das listas de amostragem em x

```
> lx:= [seq(0.000001*i,i=2799900..  2800100)];
> lff:= [seq(f(x) , x= lx)];
> lfd1:= [seq( fd1(x), x= lx)];
> lxff:= zip(par,lx,lff);
> lxfd1:= zip(par,lx,lfd1);
```

Gráfico da forma factorizada fct(x) e da primeira forma expandida fd1(x)

```
> plot([lxff ,lxfd1], x=2.7999 ..
> 2.8001,y=0 ..3e-6, symbolsize= 4,
> style = line, adaptive=false, color=[black,blue]);
```



(veja o ficheiro Ordem-parcelas.mws)

Exercício 1.10

1. Discuta
   a) os méritos relativos de dispor de subrotinas de biblioteca para calcular $\arctan(x)$ e $\arcsin(x)$ por desenvolvimento em série de Taylor;
   b) proponha um critério de truncatura para esses desenvolvimentos;
   c) discuta os méritos relativos do cálculo das restantes funções trigonométricas inversas a partir de uma destas por meio de fórmulas adequadas.
2. Tendo em consideração o carácter discreto do conjunto dos números representáveis em vírgula flutuante, discuta os problemas relativos ao cálculo de $\text{sen}(x)$ por desenvolvimento em série de Taylor.

1.4.1 Diferentes maneiras de calcular uma expressão

Dada uma expressão analítica $f(x)$ de uma função, pode considerar-se que ela especifica um algoritmo de cálculo dos valores da variável dependente para cada valor da variável independente, x; usa-se então o termo *condição* (cf. [DB74]) para descrever a sensibilidade do valor calculado de $f(x)$ a variações do argumento, x. A condição pode, portanto, ser medida pela variação (relativa) máxima de $f(x)$ em termos da correspondente variação (relativa) de x, isto é, por

$$cond[f(x)\,] = max_{x^*} \frac{\left|\frac{f(x)\,-f(x^*)}{f(x)}\right|}{\left|\frac{x-x^*}{x}\right|}$$

em que x^* é a representação aproximada de x; se $f(x)$ puder ser desenvolvida aproximadamente na forma

$$f(x)\,-f(x^*) \cong f'(x).(x-x^*)$$

para pequenos valores de $|x-x^*|$ pode dar-se a esta expressão a forma mais simples:

$$cond[f(x)\,] \cong \left|\frac{x.f'(x)}{f(x)}\right|$$

No entanto, esta forma aproximada deve ser usada com grande cuidado porque, como se afirmou atrás, o modo e a ordem por que se realizam as diferentes operações em uma expressão não é de modo nenhum indiferente do ponto de vista do erro cometido e esta expressão aproximada não toma esse facto em consideração. Por outro lado, é muito pouco interessante o facto de, nesta aproximação, a condição se tornar infinita nas vizinhanças de uma raiz da expressão.

Exemplo: 11.18 Cálculo da condição

Retomemos o caso do polinómio do 4^o grau do exemplo anterior e calculemos a sua condição, usando para isso o **maple**.

```
> ff(x) :=(5*x-14)^2*(x^2+3);
```

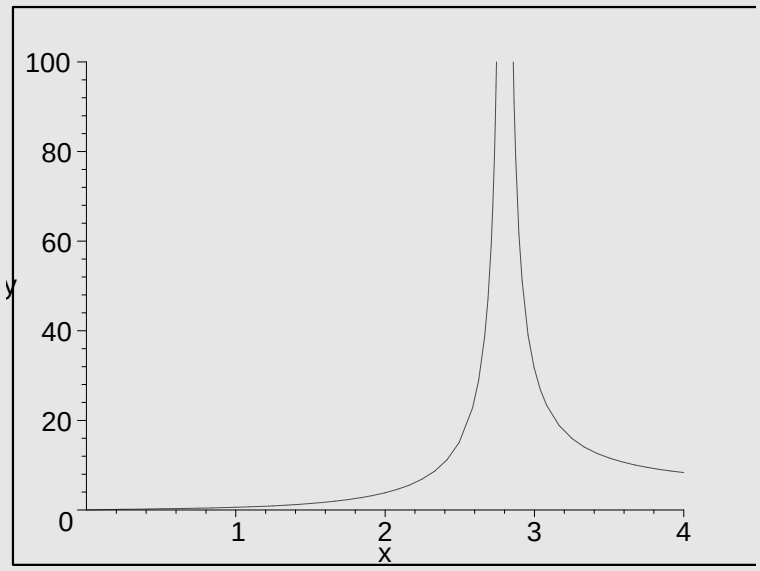
$$\mathrm{ff}(x) := (5x-14)^2\,(x^2+3)$$

```
> condff(x) :=abs((x*diff(ff(x) ,x))/ff(x) );
```

$$\mathrm{condff}(x) := \left|\frac{x\,(10\,(5x-14)\,(x^2+3)+2\,(5x-14)^2\,x)}{(5x-14)^2\,(x^2+3)}\right|$$

```
> plot(condff(x) ,x=0 ..  4, y=0 ..100);
```

Exemplo: 11.18 Cálculo da condição (cont.)



No gráfico da condição da forma factorizada do polinómio do exemplo anterior. Observemos o crescimento desmesurado da condição a partir de x = 2.5, precisamente a zona da raiz onde atrás observámos os estranhos fenómenos de perda de precisão. Verificaremos também sem dificuldade que a condição das três expressões diferentes da mesma função é a mesma, como seria de esperar. (veja o ficheiro Condição.mws)

Analisemos agora alguns casos exemplares desta situação: mesmo nos casos mais simples podem ocorrer situações de perda significativa de precisão, nomeadamente em casos relacionados com o já mencionado comportamento peculiar da subtracção. Embora pareça haver uma grande variedade de artifícios para tratar este tipo de problema, o leitor não terá, mediante um pouco de reflexão, grande dificuldade em identificar a maioria como variações do tipo geral de artifício necessário para calcular derivadas a partir da definição. Para evitar os percalços ligados a este tipo de situações, torna-se necessário apenas o uso da imaginação e o cuidado de procurar prever o que pode vir a suceder, antes de passar à fase de programação; a regra fundamental é a de, sempre que possível, evitar subtracções, mesmo que apareçam sob a forma de somas de termos de sinais diferentes. Os exemplos juntos são apenas de alguns dos métodos que o leitor deve já ter encontrado em álgebra e em análise matemática.

Exemplo: 11.19 Expressões alternativas

No cálculo de

$$\frac{\operatorname{sen}(x+\delta x)-\operatorname{sen}(x)}{\delta x}$$

para pequenos valores de x, há toda a vantagem em utilizar a expressão alternativa

$$\cos\left(x+\frac{\delta x}{2}\right).\frac{\operatorname{sen}\left(\frac{\delta x}{2}\right)}{\frac{\delta x}{2}}$$

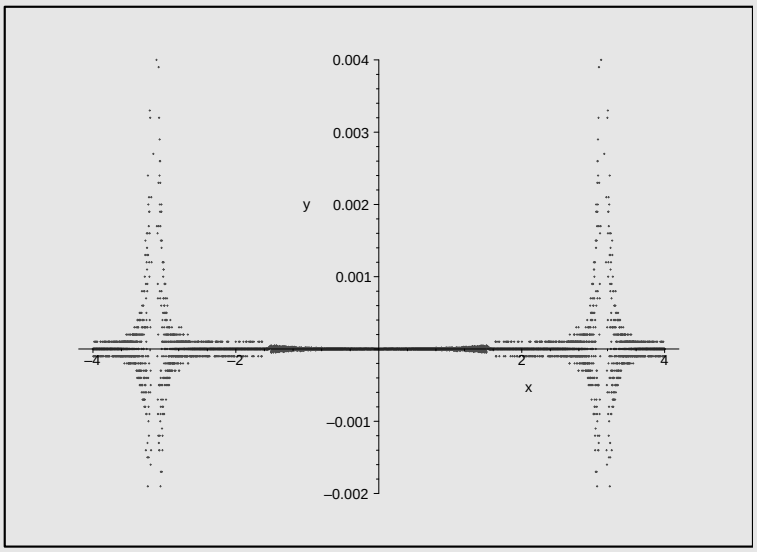
Do mesmo modo,

$$1-\cos(\delta x)=\frac{\operatorname{sen}^2(\delta x)}{1+\cos(\delta x)}=2.\operatorname{sen}^2\left(\frac{\delta x}{2}\right)$$

No entanto, é necessário o máximo cuidado: uma boa aproximação para pequenos valores de x pode ser desastrosa para valores altos, como mostra o gráfico, construído em **maple**:

Exemplo: 11.19 Expressões alternativas (cont.)

```
> Digits:= 5;
> s:= seq(0.001*i,i=-4000..  4000);
> p:= [seq([dx, 1-cos(dx)-((sin(dx)^2)/(1+cos(dx)))], dx=[s])];
> plot(p,x=-4.2 ..  4.2,y=-0.002 ..  0.004,
> symbolsize= 5, style = point,
> adaptive=false);
```



Porém, nem sempre simples rearranjos das expressões bastam para produzir novas expressões satisfatórias para o uso em cálculo numérico: por vezes é necessário recorrer a conhecimentos mais avançados das matemáticas, nomeadamente ao uso de desenvolvimento de funções em séries (de potências ou de outras funções simples). Neste caso, um critério simplista seria o de calcular e somar todos os termos (necessariamente decrescentes em valor absoluto a partir de certa ordem) que o cálculo numérico não apresente como nulos dentro da precisão utilizada; notar-se-á, porém, que

- no caso das séries de termos de sinais alternados, esse critério parece ter, para além da simplicidade, a vantagem de fixar o valor absoluto máximo do erro no limite da precisão da máquina; no caso das séries de termos de sinal constante, tal não se verifica, visto que o erro - correspondente, ele próprio, a uma série de termos de sinal constante - será em geral muito superior ao valor absoluto do primeiro valor abandonado;
- mesmo no caso mais favorável das séries de termos de sinais alternados, esse critério não só não dá garantias efectivas quanto ao erro como corresponde, em geral, a um esforço de cálculo inutilmente longo, na medida em que já se terá, de há muito, ultrapassado o limite em que as novas contribuições fazem crescer a soma;

Assim, um critério mais eficiente será o de abandonar o cálculo quando, persistentemente, a soma deixar de crescer, mesmo que as parcelas não sejam ainda nulas, o que corresponde ao limite efectivo do cálculo dentro da precisão da máquina, sob a condição de o ordenamento das parcelas ter sido adequado.

Exemplo: 11.20 Desenvolvimento em série de Taylor

Seja o desenvolvimento em série de Taylor da exponencial negativa e^{-x} até ao 5^o grau, e o correspondente erro:

```
> taylor( exp(-x), x=0, 6 );
```

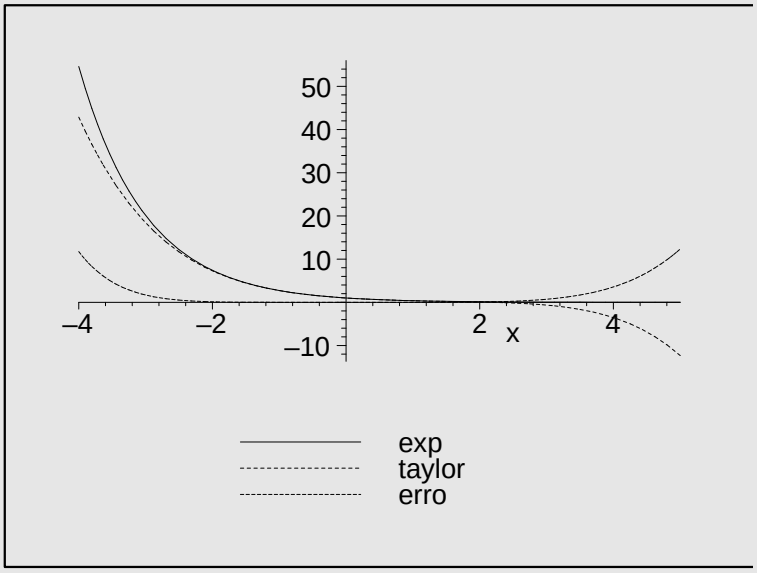
Exemplo: 11.20 Desenvolvimento em série de Taylor (cont.)

$$1 - x + \frac{1}{2}x^2 - \frac{1}{6}x^3 + \frac{1}{24}x^4 - \frac{1}{120}x^5 + \mathrm{O}(x^6)$$

```
> te(x):= convert(%,polynom);
```

$$\mathrm{te}(x) := 1 - x + \frac{1}{2}x^2 - \frac{1}{6}x^3 + \frac{1}{24}x^4 - \frac{1}{120}x^5$$

```
> plot( [exp(-x), te(x), exp(-x)-te(x)], x=
> -4 ..  5, linestyle= [1,2,3], legend=["exp", "taylor",
> "erro"]);
```



Exercício 1.11 Determine o número de termos do desenvolvimento em série de Taylor da exponencial negativa que devem ser usados para calcular e^{-1} na nossa máquina hipotética.
O termo geral do desenvolvimento é da forma

$$\frac{(-1)^k.x^k}{k!}$$

Exemplo: 11.21 Exponencial e Série de Taylor

O cálculo de $e^{a.x} - 1$ para valores relativamente grandes de $|a.x|$ pode, sem grandes problemas, ser abordado pelo cálculo da exponencial pelo método anterior. Porém, para o caso de $|a.x|$ ser pequeno, em que a exponencial se torna próxima da unidade, o erro originado pela subtracção pode tornar-se intolerável; neste caso, é preferível recorrer directamente ao desenvolvimento em série de Taylor da expressão toda

$$e^{a.x} - 1 = a.x + \frac{(a.x)^2}{2!} + \frac{(a.x)^3}{3!} + \ldots$$

Exercício 1.12 Proponha métodos de cálculo (x qualquer, h pequeno) para as funções:

$$\begin{aligned} f(x) &= \frac{1}{x+1} - \frac{1}{x} \\ f(x) &= \cos(x+h) - \cos(x) \\ f(x) &= (x+1)^{1/3} - x^{1/3} \\ f(x) &= \frac{1-\cos(h)}{h^2} \\ f(x) &= \ln \frac{1-x}{1+x} \end{aligned}$$

Exercício 1.13 Discuta os méritos do cálculo de um polinómio

$$p_n(x) = a_n.x^n + a_{n-1}.x^{n-1} + \ldots + a_1.x + a_0$$

na forma:

$$p_n(x) = (\ldots(a_n.x + a_{n-1}).x + \ldots + a_1).x + a_0$$

1.4.2 Os erros nos dados

Uma outra fonte de erro, em que raramente se pensa, mas que tem consequências muitas vezes surpreendentes é a que resulta de os coeficientes das expressões com que se trabalha em Engenharia provirem, directa ou indirectamente, de medidas cuja precisão é finita e, frequentemente, pequena. Esses erros, mesmo minúsculos, podem afectar decisivamente os resultados de um cálculo efectuado sobre os valores correspondentes.

Exemplo: 11.22 Erros nos dados

Seja a equação

$$a.x^2 + b.x + c = 0$$

cujos coeficientes a, b, c são determinados experimentalmente e têm os seguintes valores
$a = 1.50 \pm 0.01 \qquad b = -3.45 \pm 0.01 \qquad c = 1.98 \pm 0.01$
Se tomarmos apenas os valores centrais do intervalo temos a equação

$$1.50x^2 - 3.45x + 1.98 \quad \text{cujas raízes são} \qquad \begin{aligned} x_1 &= 1.10 \\ x_2 &= 1.20 \end{aligned}$$

Exemplo: 11.22 Erros nos dados (cont.)

Se, porém, tivermos em conta as variações possíveis dos coeficientes, podemos, para os respectivos valores extremos formar as equações

$$\begin{aligned}
1.51x^2-3.46x+1.99 \quad &\text{cujas raízes são} \quad && x_1=-1.15-0.07i\\
& && x_2=-1.15+0.07i\\
1.51x^2-3.46x+1.97 \quad &\text{cujas raízes são} \quad && x_1=-1.24\\
& && x_2=-1.06\\
1.51x^2-3.44x+1.99 \quad &\text{cujas raízes são} \quad && x_1=-1.14-0.14i\\
& && x_2=-1.14+0.14i\\
1.51x^2-3.44x+1.97 \quad &\text{cujas raízes são} \quad && x_1=-1.14-0.08i\\
& && x_2=-1.14+0.08i\\
1.49x^2-3.46x+1.99 \quad &\text{cujas raízes são} \quad && x_1=-1.28\\
& && x_2=-1.05\\
1.49x^2-3.46x+1.97 \quad &\text{cujas raízes são} \quad && x_1=-1.32\\
& && x_2=-1.00\\
1.49x^2-3.44x+1.99 \quad &\text{cujas raízes são} \quad && x_1=-1.15-0.05i\\
& && x_2=-1.15+0.05i\\
1.49x^2-3.44x+1.97 \quad &\text{cujas raízes são} \quad && x_1=-1.26\\
& && x_2=-1.05
\end{aligned}$$

o que significa que não só há fortes variações quantitativas nos valores das raízes como há, também, variações qualitativas, visto que existe uma zona de raízes complexas (que, num contexto físico, representam impossibilidades) e uma outra de raízes negativas (que, em certos contextos físicos, podem representar um tipo diferente de situação). Pode mesmo verificar-se que, dentro da zona da indeterminação dos coeficientes, existe uma zona em que as raízes são iguais: trata-se do caso $b=\pm 2.(a.c)^{1/2}$, isto é,

$$\begin{aligned}
a.x^2+2.(a.c)^{1/2}.x+c=0 \quad &\text{cujas raízes são} \quad && x_1=x_2=-\left(\frac{c}{a}\right)^{1/2}\\
a.x^2-2.(a.c)^{1/2}.x+c=0 \quad &\text{cujas raízes são} \quad && x_1=x_2=\left(\frac{c}{a}\right)^{1/2}
\end{aligned}$$

Nestas condições, o próprio sentido que possa ter a intenção de obter resultados precisos, mediante cálculo numérico, toma contornos muito nebulosos. Com efeito, sucede frequentemente – como no caso do exemplo acima, que os resultados calculados para diferentes valores dos parâmetros dentro dos respectivos intervalos de variação provável sejam qualitativamente diferentes; neste tipo de situação, a interpretação dos resultados torna-se altamente problemática e só um perfeito domínio do contexto concreto do problema pode permitir avançar neste terreno com um mínimo de segurança. Posto nestes termos, o problema é de natureza semântica e não puramente matemática, circunstância que raramente é apreciada em todo o seu valor e generalidade.

Observe-se, neste contexto, que instabilidades na resolução de equações, provenientes de variações dos seus parâmetros (coeficientes, expoentes, etc.), resultam não só de situações de incerteza externa mas também de efeitos internos da máquina, como os provindos da conversão de dados decimais em forma binária, o que, só por si, pode acarretar erros.

Exemplo: 11.23 A base e a exactidão da representação

Pense o leitor na conversão do número decimal (suposto exacto) 0.1 em forma binária, octal ou hexadecimal e na sua possível representação em vírgula flutuante, continuando a usar a nossa **MAKINA**.

0.1	convertido em	reconvertido para decimal	diferença
Decimal	$0.100 \times 10^{0}_{(10)} = 0.1_{10}$	$0.100 \times 10^{0}_{(10)}$	$0.000 \times 10^{0}_{(10)}$
Binário	$0.110 \times 10^{-11}_{(2)} \approx 0.1_{10}$	$0.938 \times 10^{-1}_{(10)}$	$0.600 \times 10^{-2}_{(10)}$
Octal	$0.631 \times 10^{-1}_{(8)} \approx 0.1_{10}$	$0.998 \times 10^{-1}_{(10)}$	$0.000 \times 10^{0}_{(10)}$
Hexadecimal	$0.199 \times A^{0}_{(16)} \approx 0.1_{10}$	$0.998 \times 10^{-1}_{(10)}$	$0.000 \times 10^{0}_{(10)}$

Surpreendido ?
Experimente outras conversões em https://www.exploringbinary.com/base-converter/

Exercício 1.14

1. Tendo em consideração o exemplo anterior, calcule com a precisão da nossa **MAKINA**, as raízes das equações:

$$x^2 - 100x - 4 = 0$$
$$x^2 - 60x + 2 = 0$$

2. Tendo em consideração o exemplo e os exercícios anteriores, discuta a resolução da equação quadrática geral $ax^2 + bx + c = 0$ dos pontos de vista da precisão dos resultados e da economia de cálculo.

Exercício 1.15 Em que outras bases tem $0.1_{(}10)$ representação exata ?

1.4.3 Cálculo dos erros

Na impossibilidade de evitar por completo os erros do cálculo numérico, como minimizá-los? Todos os exemplos anteriores mostram a que ponto o resultado de um cálculo pode desviar-se do valor exacto devido aos erros de arredondamento. O uso de máquinas de elevada precisão e o respeito de certas regras básicas da programação numérica limitam as consequências de certos problemas mas não as eliminam totalmente.

Na impossibilidade de eliminar por completo os erros, como estimá-los? Em primeiro lugar note-se que, se fosse possível estimar com razoável precisão o erro de um cálculo, seria, por definição, possível corrigi-lo. Assim sendo, apenas duas vias estão abertas:

A primeira é de carácter determinístico, dá um resultado certo: trata-se da chamada *aritmética intervalar*, baseada em uma ideia original de R. E. Moore [Moo66] e especialmente desenvolvida nos anos 80 por investigadores da IBM. O seu princípio é relativamente simples: suponhamos que pretendemos adicionar dois números máquina, X e Y (usamos maiúsculas para representar os números máquina e minúsculas para os números "reais"). O resultado dessa soma é, ou número máquina igual ou imediatamente inferior ao valor z da soma real $[Z_1 = \nabla(z)]$, ou o número máquina igual ou imediatamente superior a z $[Z_2 = \triangle(z)]$; assim, em vez de representar o número z por Z_1 ou Z_2 , passaremos a representá-lo pelo intervalo (Z_1, Z_2) ; procedendo sistematicamente deste modo, obteremos, no final, um intervalo exacto no qual o valor real se encontra necessariamente contido. A Universidade de Karlsruhe desenvolveu, a partir de 1986, software eficiente deste tipo; infelizmente, na grande maioria dos casos, os intervalos finais calculados são horrendamente pessimistas.

Exemplo: 11.24 Análise de erros

- Se somarmos ou subtrairmos duas quantidades x_1, x_2 afectadas de erros δx_1 e δx_2, o resultado aproximado será

  $$y = x_1 \pm x_2$$

  enquanto o resultado exacto será

  $$y + \delta y = x_1 + \delta x_1 \pm (x_2 \pm \delta x_2)$$

  de modo que o erro cometido em consequência da aproximação vale

  $$\delta y = \delta x_1 \pm \delta x_2$$

  Dado, porém, que não podemos conhecer o sinal dos erros, escreveremos apenas

  $$|\delta y| \leq |\delta x_1| + |\delta x_2|$$

  o que nos permite enunciar a regra de que o módulo do erro absoluto de uma soma (ou diferença) é majorado pela soma dos módulos dos erros absolutos dos operandos.

- Para a multiplicação, o resultado aproximado é

  $$y = x_1 . x_2$$

  e o exacto é

  $$y + \delta y = (x_1 + \delta x_1) . (x_2 + \delta x_2)$$

  , de modo que o erro absoluto vale

  $$\delta y = x_1 . \delta x_2 + x_2 . \delta x_1 + \delta x_1 . \delta x_2$$

Exemplo: 11.24 Análise de erros (cont.)

e o erro relativo

$$\Delta y \equiv \frac{\delta y}{y} = \frac{x_1.\delta x_2 + x_2.\delta x_1 + \delta x_1.\delta x_2}{x_1.x_2} = \Delta x_1 + \Delta x_2 + \Delta x_1.\Delta x_2$$

ou, supondo que os erros relativos dos operandos são significativamente menores que a unidade,

$$\Delta y \cong \Delta x_1 + \Delta x_2$$

- Para a divisão, o quociente aproximado é

$$y = \frac{x_1}{x_2}$$

e o quociente exacto

$$y + \delta y = \frac{x_1 + \delta x_1}{x_2 + \delta x_2} = \frac{(x_1 + \delta x_1).(x_2 - \delta x_2)}{x_2^2 - \delta x_2^2} \cong y + \frac{\delta x_1}{x_2} - y.\Delta x_2$$

e o erro relativo

$$\Delta y \cong \Delta x_1 - \Delta x_2$$

Conforme vimos pelo exemplo poderemos estabelecer as seguintes regras e observações:

1. O módulo do erro absoluto de uma soma (ou diferença) é majorado pela soma dos módulos dos erros absolutos dos operandos
2. A semelhança das expressões dos erros relativos do produto e do quociente e o facto de não conhecermos o sinal dos erros permite escrever uma expressão única para os dois casos: de onde a regra de que o módulo do erro relativo de um produto (ou quociente) é majorado pela soma dos módulos dos erros relativos dos operandos.
3. Embora estas regras não sejam particularmente difíceis de usar, a sua aplicação a cada passo de um cálculo complexo, como na aritmética intervalar, torná-lo-ia extremamente penoso. Por isso, muitos analistas numéricos estão interessados em simplificar estas regras, mesmo que à custa de alguma precisão. As regras aproximadas servirão para determinar, aproximadamente, o número de algarismos a reter num resultado, com base no número de algarismos exactos dos operandos; tais algarismos a reter chamam-se *algarismos significativos*. A validade da análise dos algarismos significativos depende, naturalmente, da base de numeração usada, porque desprezar um algarismo em uma base pode corresponder a desprezar dois ou mais em uma base menor; além disso, dois números que, em dada base, têm o mesmo número de algarismos significativos podem, em outra base, ter um número diferente. Assim, devido à possível perda de significado em consequência de a base de numeração eventualmente usada no cálculo não coincidir com aquela em que se apresentam os resultados (como sucede no cálculo electrónico), é de boa prática reter pelo menos um algarismo incerto, que se chama *dígito de guarda*, desprezando-o apenas no resultado final.
4. Da regra anterior relativa ao erro absoluto da soma e da diferença, resulta que esse erro é dominado pelo maior erro dos operandos; na pior das hipóteses, os dois operandos terão erros da mesma ordem, de modo que o erro do resultado será majorado pelo dobro do maior dos erros dos operandos. Por outro lado, poderemos normalmente admitir que o erro na representação de um número não será superior a meia unidade da última casa significativa, de modo que o operando dominante será aquele que tiver menor número de algarismos significativos à direita da vírgula. A regra para a determinação do número de algarismos significativos de uma soma ou diferença será, portanto,

a seguinte: o número de algarismos a reter na parte fraccionária do resultado de uma adição (ou subtracção) não será maior que o número de algarismos da parte fraccionária do operando que a tiver menos extensa.

5. De modo semelhante, deduzimos da regra anterior relativa os erros relativos do produto e do quociente que o erro relativo é dominado pelo do operando que tiver maior erro relativo. Para obter uma aproximação em termos de algarismos significativos, utilizamos a representação em vírgula flutuante: $x = m.r^p$, em que a mantissa m não tem zeros imediatamente à direita da vírgula e r é a base de numeração usada; então

$$r^{-1} \leq |m| \leq 1$$

Por outro lado, se x tiver q algarismos significativos, então a sua mantissa, m, terá esse mesmo número de algarismos; logo, a grandeza do erro absoluto de m não será maior que $r^{-q}/2$ de modo que

$$\delta x \leq \tfrac{1}{2}.r^{p-q}$$

Dado que $|m| \geq r^{-1}$, resulta $|x| \geq r^{p-1}$, logo

$$|\Delta x| = \left|\frac{\delta x}{x}\right| \leq \tfrac{1}{2}.r^{1-q}$$

Deste modo, a grandeza do erro relativo de um operando depende apenas do número de algarismos significativos desse operando, donde resulta a regra: o número de algarismos significativos de um produto (ou quociente) é majorado pelo número de algarismos significativos do operando que os tiver em menor número.

6. Esta análise de algarismos significativos é tipicamente útil para o cálculo manual, mas pode utilizar-se em cálculo automático; porém, para este fim, as linguagens convencionais não são convenientes, tornando-se necessário o recurso a linguagens de processamento de listas.

Uma outra via, dita *método de perturbação*, é de carácter probabilístico e começou a ser explorada em meados dos anos 70 na Universidade Pierre et Marie Curie de Paris. O seu princípio é também simples e engenhoso: a melhor maneira de testar o efeito, sobre o resultado final, das perdas de informação por arredondamento consiste em simulá-la, perturbando cada cálculo elementar por vários erros aleatórios da ordem de grandeza dos arredondamentos e estudando estatisticamente os resultado. O Instituto Francês do Petróleo desenvolveu, a partir de 1987, um package deste tipo; infelizmente, o cálculo torna-se excessivamente lento para todas as aplicações práticas excepto as de maior responsabilidade.

Exercício 1.16

1. Calcule $\text{sen}(x)$ para $x = 2$ rad por meio de um desenvolvimento em série de Taylor. Calcule cada termo da série com sete algarismos significativos. Some os termos e compare o resultado com o valor correcto a nove decimais: 0.909297427. Comente.

2. Calcule $\cos(x) - \cos(0.1)$ a sete algarismos significativos para $x = 0$, $x = 0.05$, $x = 0.095$ e $x = 0.995$,

   a) usando uma tábua ou uma máquina de calcular

   b) usando a identidade

$$\cos(x) - \cos(y) = 2.\text{sen}\left(\frac{y+x}{2}\right).\text{sen}\left(\frac{y-x}{2}\right)$$

   compare os resultados e comente.

3. Na programação de computadores é frequente a utilização de constantes numéricas para tornar a computação mais eficiente. Por exemplo, usando

---

1.5 Conclusão

O objectivo do presente capítulo foi, essencialmente, o de chamar a atenção do estudante para o facto de não poder nunca confiar-se cegamente nos resultados fornecidos por um computador e para o facto de os problemas de precisão do cálculo numérico automático serem ainda matéria de investigação científica e tecnológica. Os factos centrais a destacar são:

- o de uma longa sucessão de cálculos poder conduzir a um resultado grosseiramente errado;
- o de as consequências deste primeiro facto poderem ser extremamente graves: os cálculos em vírgula flutuante são utilizados para projectar edifícios, pontes, barragens, automóveis e aviões.

Vimos também, por outro lado, que, antes de pôr em exploração uma nova aplicação, é possível e desejável, embora caro e trabalhoso, testá-la do ponto de vista da precisão.

O grande matemático A. Householder dizia que tinha medo de andar de avião porque sabia que tinha sido projectado usando aritmética de vírgula flutuante; trata-se de um comentário extremamente sério não só a um estado de coisas ainda insatisfatório mas, sobretudo, às práticas descuidadas e irresponsáveis que, infelizmente, são correntes.

[Ham71]

2 Zeros reais de uma função real

Contents

Figures

Tables

O problema de achar as raízes reais de uma dada função real contínua é um dos mais frequentes problemas de cálculo numérico em Engenharia.

Exemplo: 22.1 Raízes

Para a equação

$$x^2+80x+1=0$$

as raízes são, com 5 algarismos significativos, $x_1=-79.987$ e $x_2=-0.012502$ Ora, a fórmula resolvente normal dá, na precisão da nossa máquina hipotética

$$x'_1=\frac{-b-\sqrt{b^2-4.a.c}}{2.a}=\frac{-0.800\times10^2-\sqrt{0.640\times10^4-0.400\times10^1}}{0.200\times10^1}$$
$$x'_1=-0.800\times10^2$$
$$x'_2=\frac{-b+\sqrt{b^2-4.a.c}}{2.a}=\frac{-0.800\times10^2+\sqrt{0.640\times10^4-0.400\times10^1}}{0.200\times10^1}$$
$$x'_2=\pm0.000\times10^{-9}$$

sendo manifesto que $f(x'_1)=1$ e $f(x'_2)=1$ mas

$$\left(x-x''_1\right).\left(x-x''_2\right)=x^2+80x+1$$

Que tipo de situação é que o leitor preferiria?

Para o matemático, um zero de uma função $y=f(x)$ é qualquer número ξ que, substituído na expressão de $f(x)$, produz um resultado exactamente nulo: $f(\xi)=0$. Porém, em cálculo numérico, dada a finitude dos números e da sua precisão, há apenas um conjunto discreto de números que podemos ensaiar como candidatos a zeros, e habitualmente sucede que nenhum deles produz exactamente zero para o valor da função.

Porém, em cálculo numérico, o significado de uma raiz é extremamente ambíguo: veja-se (no capítulo 1), por exemplo, o que se passa com o cálculo das formas desenvolvidas do polinómio do 4^o grau $(5x-14)^2.\left(x^2+3\right)$ nas vizinhanças do zero teórico $x=2.8$.

Usando o **Maxima** [1]:

wxMaxima 2.1: Significado de uma raiz

Para ilustrar os problemas do cálculo desta expressão, foi necessário organizar o cálculo do **Maxima** de maneira a evitar algumas das suas optimizações: reorganização das expressões, recorrendo à decomposição em funções, gráfico a partir de pontos discretos. Note-se que a solução numérica proposta pelo **Maxima**, um método intervalar baseado na mudança de sinais em pontos da função, só é possível porque perto da raiz da função surgem valores negativos espúrios; como é facilmente verificável, a função é sempre positiva!

(% i1) f(x):=(5*x-14)^2*(x^2+3);

(% o1) $$\mathrm{f}(x):=(5x-14)^2\left(x^2+3\right)$$

(% i2) f1(x):=25*x^4;

(% o2) $$\mathrm{f1}(x):=25x^4$$

(% i3) f2(x):=271*x^2;

(% o3) $$\mathrm{f2}(x):=271x^2$$

[1] (ver o ficheiro ZEROS-2-2)

wxMaxima 2.1: Significado de uma raiz (cont.)

```
(% i4)  f3(x):= 588-140*x^3;
```

(% o4) $$f3(x) := 588 - 140x^3$$

```
(% i5)  f4(x):= -420*x;
```

(% o5) $$f4(x) := (-420)x$$

```
(% i6)  "Cálculo numérico das raízes, a partir de vários intervalos."$
(% i7)  find_root("(f1(x)+f2(x)+f3(x)+f4(x)), x, 2.799999964, 2.80001);
```

(% o7) 2.799999964

```
(% i8)  find_root("(f1(x)+f2(x)+f3(x)+f4(x)), x, 2.79, 2.800000029);
```

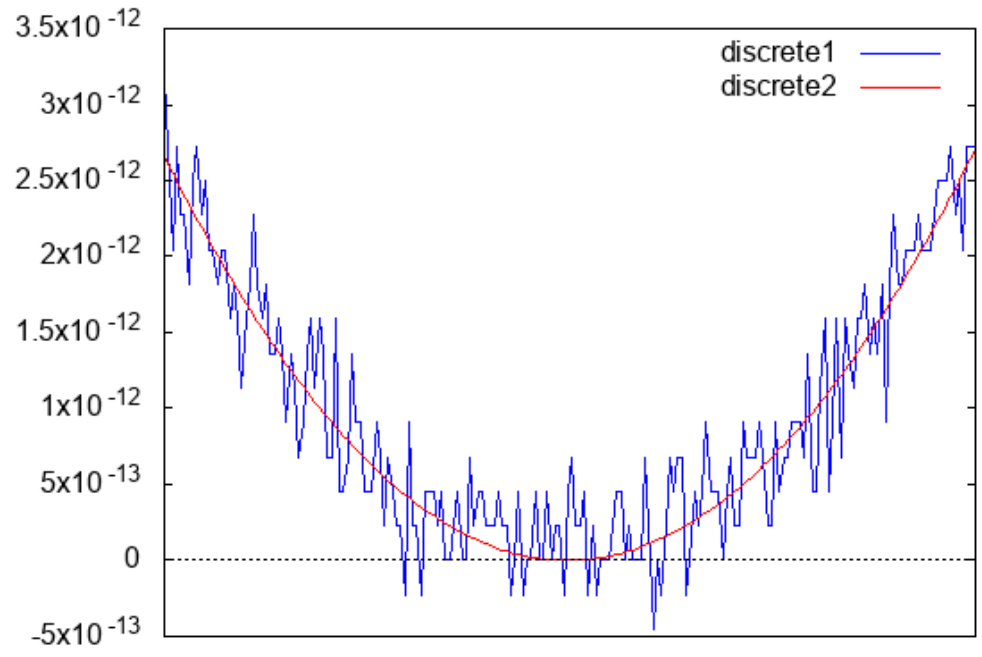
(% o8) 2.799999971779375

```
(% i9)  find_root("(f1(x)+f2(x)+f3(x)+f4(x)), x, 2.70, 2.800000029);
```

(% o9) 2.799999981316271

```
(% i10) "Construção das listas de pontos para o gráfico."$
(% i11) xx:makelist(x/1000000000+2.7999999,x,1,200),numer$
(% i12) yy:makelist(f1(x)+f2(x)+f3(x)+f4(x),x,xx),numer$
(% i13) zz:makelist(f(x),x,xx),numer$
(% i14) wxplot2d([[discrete,xx,yy],[discrete,xx,zz]],
        [plot_format, gnuplot],
        [gnuplot_preamble, "set zeroaxis; unset xtics"])$
```

(% t14)



Assim sendo, temos o direito (e o dever) de nos interrogar: o que pretendemos dos zeros de uma função:

- que sejam números ξ_i tão próximos quanto possível do valor teórico?
- ou, pelo contrário, que nos pontos ξ_i o valor calculado de $|f(\xi_i)|$ seja tão pequeno quanto possível?

Aquilo que pretendemos depende, evidentemente, do contexto de cada problema particular, mas é evidente que a pergunta não é inocente, até porque a estratégia do cálculo numérico depende fortemente da resposta; com efeito, no primeiro caso estaremos perante um problema de *resolução de equações*, enquanto no segundo o problema será de *optimização*. Seja como for, uma consideração cuidadosa dos casos concretos que se nos deparam na prática mostrará claramente que são mais frequentes os do segundo que os do primeiro tipo, ao contrário do que poderíamos pensar e do que correntemente se pratica.

Quanto à técnica a utilizar para calcular uma raiz de uma equação (ou, o que é o mesmo, o zero de uma função), teremos que distinguir o caso de se dispor de uma solução por via analítica, caso em que, porque a estrutura do algoritmo está, em termos genéricos, fixada, o problema se reduz ao cálculo de uma expressão, como o é o caso da equação do 2ºgrau; diz-se, neste caso, que se dispõe de um *método directo* ou *cerrado*, isto é, de um processo que envolve apenas a execução finita e não-iterativa de operações aritméticas e lógicas. Embora um método directo possa ter alternativas de percurso (como acabámos de ver para a equação do 2ºgrau), eventualmente a escolher com base nos próprios valores obtidos durante o cálculo, distingue-se por cada passo ser executado, em geral, apenas uma vez e o resultado ser exacto, a menos dos erros introduzidos pela representação dos números no processo de cálculo utilizado. Como se sabe, os casos de resolubilidade analítica são em número muito reduzido, embora sejam frequentes nas mais simples das situações correntes:

- das equações algébricas, apenas têm solução analítica geral as de grau inferior ao quinto;
- das equações racionais, apenas aquelas que possam reduzir-se a equações algébricas resolúveis (e, mesmo assim, com os problemas marginais da possível introdução de raízes estranhas pela multiplicação da equação por expressões que contêm a incógnita);
- para as equações irracionais e transcendentes não existem soluções gerais e são raros os casos particulares resolúveis analiticamente.

No caso de não estar disponível uma solução analítica em forma cerrada, diremos que o problema se reduz, na generalidade, a procurar um par de números, x_1, x_2 suficientemente próximos um do outro e entre os quais se encontre uma e uma só raiz da equação, o que exige um *método indirecto* ou *aberto*. O problema geral da resolução numérica indirecta deveria, portanto, comportar duas partes distintas:

1. - encontrar um intervalo que enquadre uma e uma só raiz;
2. - reduzir esse intervalo a dimensões compatíveis com a nossa necessidade (e possibilidade) de precisão.

Na prática, dados os efeitos dos arredondamentos que inevitavelmente ocorrem no cálculo da função, não teremos sequer, em geral, uma única mudança de sinal em torno do zero verdadeiro, mas eventualmente, uma pequena sucessão de números consecutivos que dão uma sucessão de mudanças de sinal nos valores calculados numericamente (reveja-se o caso da função do 4ºgrau do Capítulo anterior). Embora, em princípio, uma tal situação possa confundir-se com uma sucessão de zeros muito próximos, na prática não existirão dificuldades especiais, desde que tenhamos uma razoável ideia da ordem de grandeza dos erros numéricos cometidos no cálculo numérico da função. Esta consideração mostra desde já como foram importantes o tempo e o esforço que no Capítulo anterior consagrámos ao problema dos erros de arredondamento, dado que eles são inevitáveis nos processos de calculo numérico de resolução de equações na medida em que o cancelamento exacto de termos positivos e negativos faz parte da própria natureza do problema.

2.1 Isolamento das raízes

O seguinte teorema bem conhecido dá uma boa medida das ambiguidades e das dificuldades associadas ao problema do isolamento das raízes:

> Se uma função contínua $f(x)$ toma valores de sinais opostos nos extremos de um intervalo (a,b), isto é, se $f(a)\times f(b)<0$, então esse intervalo contém um número ímpar de raízes da equação $f(x)=0$; se, pelo contrário, toma valores do mesmo sinal, isto é, se $f(a)\times f(b)>0$, então esse intervalo contém um número par de raízes (incluindo nenhuma raiz).

No caso de, nos extremos do intervalo, a função tomar valores de sinal idêntico, não podemos, naturalmente, garantir que não existem raízes nesse intervalo (podem existir duas ou, mais geralmente, um número par). A garantia de que não existem raízes no intervalo só pode ser obtida mediante recurso a informação suplementar, como, por exemplo, a de que a função muda de sinal algures no intervalo.

Exemplo: 22.2 Isolamento de raízes

Consideremos o problema de isolar as raízes da equação

$$f(x) = x^4 - 4x + 1 = 0$$

dado que $f(-\infty) = +\infty > 0$ e $f(+\infty) = +\infty > 0$, nada podemos concluir.
Porém, se considerarmos a derivada

$$f'(x) = 4x^3 - 4$$

resulta que $f'(1) = 0$ e, por outro lado, $f(1) = -2 < 0$, de modo que a equação tem, pelo menos, duas raízes, uma no intervalo $(-\infty, 1)$, outra no intervalo $(1, +\infty)$. Usemos as facilidades do **Maxima** para visualizar os comportamentos da função e da sua derivada na figura 2.1[a].

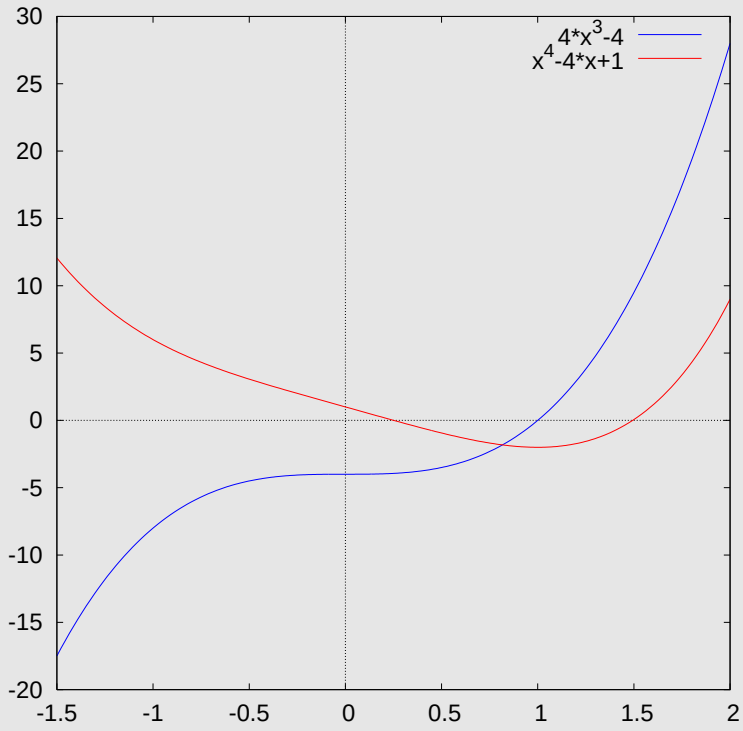


Figura 2.1: Isolamento de raízes

[a] (ver o ficheiro ZEROS-2-3)

wxMaxima 2.2: Função e derivada

```
(% i1)  x^4-4*x+1;
```

(% o1) x^4-4x+1

```
(% i2)  diff(%, x,1);
```

(% o2) $4x^3-4$

```
(% i3)  plot2d([%o1,%o2], [x,-1.5,2],
        [plot_format, gnuplot],
        [gnuplot_preamble, "set xrange [-1.5:2]; set size ratio 1; set zeroaxis;"])$
```

No caso dos sinais opostos, a única coisa que o teorema garante é a existência de um número ímpar de raízes. A garantia de que existe uma raiz única só pode ser obtida mediante recurso a informação mais pormenorizada e raramente acessível, como, por exemplo, a de que a primeira derivada mantém o sinal no intervalo considerado, o que corresponde a saber que a função é monótona no intervalo.

Exemplo: 22.3 Isolamento de raízes II

Seja a função

$$f(x) = x^4 - 5x^2 + 1$$

tal que $f(-\infty) = +\infty > 0$ e $f(+\infty) = +\infty > 0$.
A sua derivada,

$$f'(x) = 4x^3 - 10x$$

tem um zero óbvio em $x = 0$, no qual a função vale $f(0) = 4 > 0$, pelo que continuamos sem saber nada sobre a eventual existência de raízes. Porém, $f'(-\infty) = -\infty < 0$ e $f'(+\infty) = +\infty > 0$, de modo que a derivada tem, pelo menos, um zero no intervalo $(-\infty, 0)$ e, portanto, também um outro no intervalo $(0, +\infty)$, o que completa o máximo de três que pode ter (por ser do 3ºgrau). Como anteriormente, utilizaremos as facilidades gráficas do **Maxima** para estudar o comportamento da função e da sua derivada na figura 2.2[a]:

Exemplo: 22.3 Isolamento de raízes II (cont.)

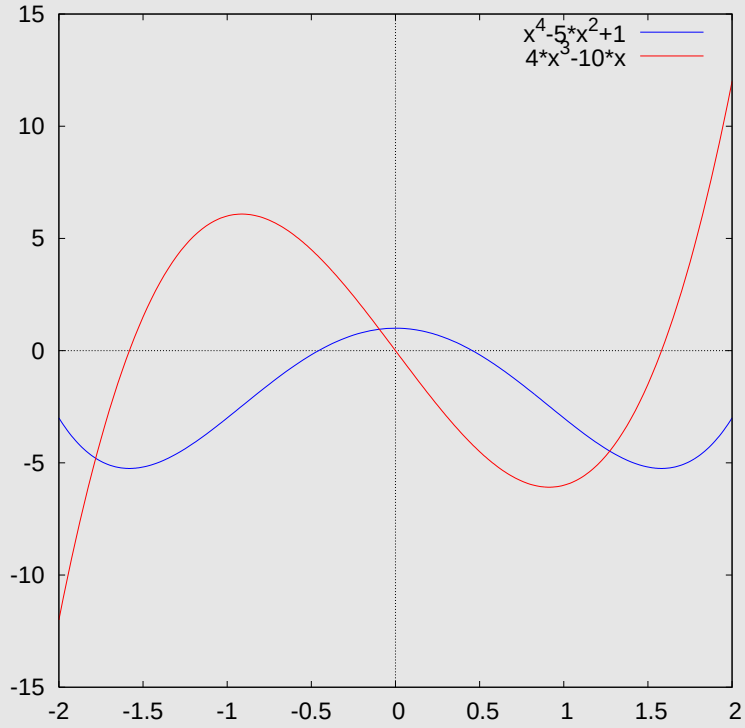


Figura 2.2: Isolamento de raízes II

[a] (ver o ficheiro ZEROS-2-4)

wxMaxima 2.3: Função e derivada

```
(% i1)  x^4-5*x^2+1;
```

(% o1) $x^4 - 5x^2 + 1$

```
(% i2)  diff(%, x);
```

(% o2) $4x^3 - 10x$

```
(% i3)  plot2d([%o1,%o2], [x,-2,2],
        [plot_format, gnuplot],
        [gnuplot_preamble, "set xrange [-2:2]; set size ratio 1; set zeroaxis;"])$
```

Nestas condições, a única solução geral que se pode aconselhar para tratar o problema do isolamento das raízes é a de utilizar todos os recursos ao nosso alcance para obter um conhecimento suficientemente exacto do comportamento da função (e, eventualmente, das suas derivadas) no intervalo de interesse, o qual, em muitos casos, será fixado com precisão suficiente pelas constrições impostas pela física, pela economia, ou pelo simples bom senso, no contexto do problema concreto que temos diante de nós; só na total falta de tais critérios, o intervalo terá que ser tomado como $]-\infty,+\infty[$.

Assim, por exemplo, se existe e é conhecida uma derivada contínua e se os zeros desta puderem ser mais facilmente calculados que os da função original, o processo de isolamento das raízes pode ser acelerado;

com efeito, bastará apenas tomar os sinais da função nos zeros da derivada e nos extremos do intervalo para obter importante informação suplementar.

Exemplo: 22.4 Isolamento de raízes III

Determinar o número de raízes reais da equação

$$f(x) = x + e^x = 0$$

Dado que $f'(x) = 1 + e^x > 0$
e que $f(-\infty) = -\infty < 0,\ \ f(+\infty) = +\infty > 0$,
concluímos que a equação tem uma e uma só raiz real, como mostra a figura 2.3[a].

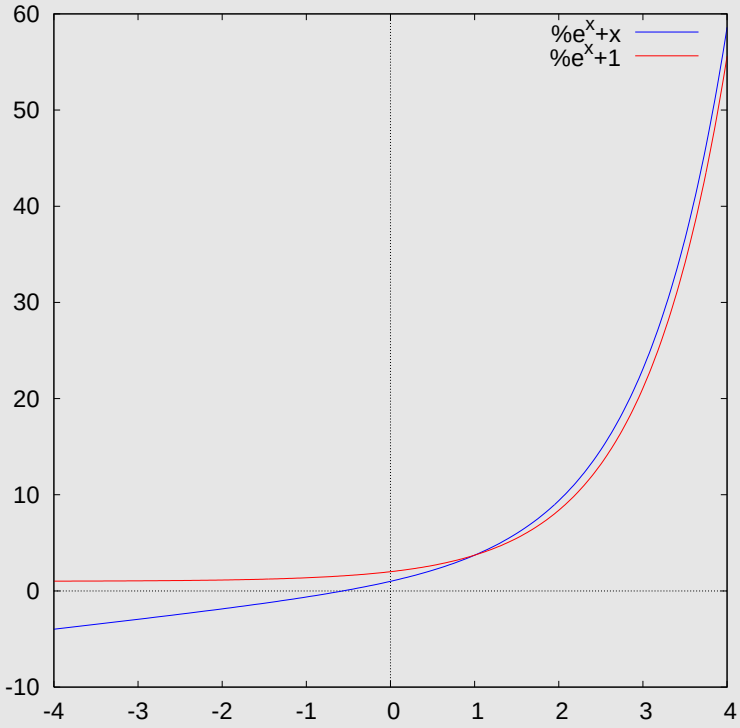


Figura 2.3: Isolamento de raízes III

[a] (ver o ficheiro ZEROS-2-5)

wxMaxima 2.4: Uma só raiz

```
(% i1)  x+%e^x;
```

(% o1) $\%e^x + x$

```
(% i2)  diff(%, x);
```

(% o2) $\%e^x + 1$

```
(% i3)  plot2d([%o1,%o2], [x,-4,4],
        [plot_format, gnuplot],
        [gnuplot_preamble, "set xrange [-4:4]; set size ratio 1; set zeroaxis;"])$
```

Do mesmo modo, se o nosso problema é algébrico e de grau n, sabemos antecipadamente que tem n e só n raízes, das quais alguns pares poderão ser complexos conjugados e as restantes serão reais.

No caso das funções para as quais dispomos de uma expressão analítica, um utilitário matemático do tipo **`matlab`**, **`maple`**, **`derive`** ou **`Maxima`**, (especialmente estes últimos, dadas as suas possibilidades de cálculo simbólico) dotado de uma boa rotina de traçado de gráficos com possibilidade de *zoom* constituirá um auxiliar precioso nesta tarefa. No caso de a função ter vários parâmetros livres que modificam o seu comportamento, pode ser necessário muito trabalho, apoiado em uma boa dose de conhecimentos e intuição matemática para levar a tarefa a cabo. Pelo contrário, se a função é definida por um algoritmo não exprimível sob forma cerrada, tornar-se-á em geral necessário escrever um programa informático que permita explorar os seus valores em diferentes pontos e traçar o respectivo gráfico. Neste caso, em particular, torna-se necessário estar muito atento à possibilidade de a função apresentar descontinuidades.

Seja como for, o problema apresenta-se desde o início com a possibilidade de dois níveis de dificuldade muito distintos:

1. quando se trata de resolver apenas o problema isolado de determinação das raízes de uma equação particular, pode estudar-se cada caso particular *de per si*;

2. no caso de a resolução da equação ser apenas parte de um problema mais vasto e, portanto, o respectivo algoritmo constituir apenas uma rotina dentro de um programa, toda a questão do isolamento da raiz terá que ser resolvida na generalidade, na fase da análise do problema, isto é, antes de passar à implementação informática da solução; como se compreende, esta situação é de longe mais complexa que a anterior.

Dadas estas circunstâncias particularmente desfavoráveis, o objectivo do isolamento da raiz é, na prática corrente, substituído pelo objectivo menos ambicioso de encontrar um intervalo em que a função muda de sinal.

Exemplo: 22.5 Isolamento de raízes IV

Isolar os zeros da função

$$p_3(x) = x^3 - 6x + 2$$

considerando a expressão analítica como uma mera especificação de cálculo de valores e sem a utilizar para outros fins.
Comecemos por observar que se trata de um polinómio do 3ºgrau e, portanto, contínuo e com um máximo de 3 raízes distintas. Em seguida, tabulemos (ver tabela 2.1) alguns valores encontrados, começando por $\pm\infty$, em seguida 0, depois ± 10 (supondo que estamos especialmente interessados em raízes pequenas) e depois por bissecções sucessivas aproximadas.

Exemplo: 22.5 Isolamento de raízes IV (cont.)

	1º passo		2º passo		3º passo		4º passo		5º passo	
	x	$f(x)$	x	$f(x)$	x	$f(x)$	x	$f(x)$	x	$f(x)$
valor	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
raízes			**0 a 2**		**0 a 2**		**0 a 2**		**0**	
valor			-10	-938	-10	-938	-10	-938	-10	-938
raízes					**0 a 2**		**0 a 2**		**0**	
valor					-5	-93	-5	-93	-5	-93
raízes	**1 a 3**		**1 a 3**				**0 a 2**		**0**	
valor							-3	-7	-3	-7
raízes					**1 a 3**		**1 a 3**		**1**	
valor									-1	+7
raízes									**0**	
valor	0	+2	0	+2	0	+2	0	+2 +2	0	+2
raízes									**1**	
valor									+1	-3
raízes					**0 a 2**		**0 a 2**		**1**	
valor							+3	+11	+3	+11
raízes	**0 a 2**		**0 a 2**				**0 a 2**		**0**	
valor					+5	+97	+5	+97	+5	+97
raízes					**0 a 2**		**0 a 2**		**0**	
valor			+10	+942	+10	+942	+10	+942	+10	+942
raízes			**0 a 2**		**0 a 2**		**0 a 2**		**0**	
valor	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$

Tabela 2.1: Isolamento de raízes

De repente, a separação das raízes aparece completada. A equação tem, portanto, 3 raízes reais, situadas nos intervalos $(-3,-1)$, $(0,+1)$, $(+1,+3)$. Do ponto de vista da separação, nada mais é preciso, mas é evidente que os intervalos poderiam continuar a ser reduzidos usando o mesmo método.

Exercício 2.1

1. Determine, por meios gráficos, valores grosseiramente aproximados das raízes positivas da equação $\tan(x) = x$

2. Determine a dois algarismos significativos

   a) as duas primeiras raízes positivas de $x.\operatorname{sen}(x) = 1$

   b) a raiz de $x.e^x = 1$

   c) a menor raiz positiva de $\tan(x) = x^2 + 1$

   d) Mostre que não há raízes reais da equação $e^x = 1 + \ln(x)$

2.2 Método da Bissecção

Suposto identificado um intervalo (a,b) em cujos extremos a função tem sinais opostos, o método mais simples de reduzir esse intervalo é o *método da bissecção* que constitui também, e de longe, o método mais seguro para implementar em uma rotina informática de uso geral. Neste método, a redução do intervalo é realizada mediante o cálculo da função no ponto médio do intervalo, $f\left(\left(x_1+x_2\right)/2\right)$. Se este valor não for nulo (porque, se fosse o caso, teríamos logo encontrado a raiz "exacta"), então ocorrerá necessariamente uma mudança de sinal em uma (e uma só) das metades do intervalo original; a redução desejada far-se-á substituindo o intervalo original por essa metade; deste modo, $\left(x_1+x_2\right)/2$ substitui x_1 ou x_2, conforme o caso, e o processo repete-se.

Algoritmo 2.1

```
DADOS a0 , b0  : f(a0 ).f(b0 )<0
    PARA n=0,1,2,... ATÉ ONDE FOR CONVENIENTE
        SEJA m=(an +bn )/2
        SE f(an ).f(m )<0  SEJAM an+1 = an ; bn+1 = m
        CASO CONTRÁRIO SEJAM an+1 = m; bn+1 = bn
```

Exemplo: 22.6 Bissecção

Vamos usar o método da bissecção para melhorar a raiz da equação

$$x^4+2x^3-x-1=0$$

situada no intervalo $(0,1)$, trabalhando a cinco algarismos significativos.

a	f(a)	b	f(b)	intervalo
0,000000E+00	-1,000000E+00	1,00000E+00	1,000000E+00	1,000000E+00
5,000000E-01	-1,187500E+00	1,00000E+00	1,000000E+00	5,000000E-01
7,500000E-01	-5,898440E-01	1,00000E+00	1,000000E+00	2,500000E-01
7,500000E-01	-5,898440E-01	8,75000E-01	5,102540E-02	1,250000E-01
8,125000E-01	-3,039400E-01	8,75000E-01	5,102540E-02	6,250000E-02
8,437500E-01	-1,355730E-01	8,75000E-01	5,102540E-02	3,125000E-02
8,593750E-01	-4,461470E-02	8,75000E-01	5,102540E-02	1,562500E-02
8,593750E-01	-4,461470E-02	8,67188E-01	2,615420E-03	7,813000E-03
8,632820E-01	-2,114390E-02	8,67188E-01	2,615420E-03	3,906000E-03
8,652350E-01	-9,301190E-03	8,67188E-01	2,615420E-03	1,953000E-03
8,662120E-01	-3,349080E-03	8,67188E-01	2,615420E-03	9,760000E-04
8,667000E-01	-3,691450E-04	8,67188E-01	2,615420E-03	4,880000E-04
8,667000E-01	-3,691450E-04	8,66944E-01	1,122560E-03	2,440000E-04
8,667000E-01	-3,691450E-04	8,66822E-01	3,765620E-04	1,220000E-04
8,667000E-01	-3,691450E-04	8,66761E-01	3,672460E-06	6,100000E-05
8,667310E-01	-1,796890E-04	8,66761E-01	3,672460E-06	3,000000E-05
8,667310E-01	-1,796890E-04	8,66746E-01	-8,801060E-05	1,500000E-05
8,667390E-01	-1,307950E-04	8,66746E-01	-8,801060E-05	7,000000E-06

Tabela 2.2: Exemplo de bissecção sucessiva
A melhor aproximação à raiz é, portanto, $x=0.86674$.

Cada iteração reduz a metade o comprimento do intervalo, de modo que 10 iterações, por exemplo, reduzem-no de um factor de $2^{10} = 1024 > 10^3$, enquanto que 20 iterações o reduzem de $2^{20} = 1048576 > 10^6$.

E como se decide a paragem do processo? Existem basicamente cinco tipos de critérios distintos, aplicáveis de acordo com o objectivo em vista:

critério de precisão absoluta

$$|x_1 - x_2| \leq \varepsilon$$

Consiste em parar quando o intervalo que contêm a raiz for menor que um dado valor (pequeno) pré-definido (sob a condição evidente de ε ser maior que a distância dos dois números representáveis consecutivos na vizinhança da raiz, sob pena de o processo não terminar);

critério de precisão relativa

$$|(x_1 - x_2)/x_1| \leq \varepsilon \qquad \text{ou} \qquad |(x_1 - x_2)/x_2| \leq \varepsilon$$

Consiste em parar quando a razão entre o intervalo que contêm a raiz e a própria raiz (ou um valor aproximado) for menor que um dado valor (pequeno) pré-definido (sob a condição de a raiz não ser demasiado próxima de zero, sob pena de o processo não terminar);

critério de anulação da função

$$|f(x_1) - f(x_2)| \leq \varepsilon$$

Consiste em parar quando diferença entre o valor da função nos extremos do intervalo que contêm a raiz for menor que um dado valor (pequeno) pré-definido (sob a condição de a derivada da função não ter, nas vizinhanças da raiz, valor absoluto demasiado alto, sob pena de o processo não terminar);

critério da precisão máxima Consiste em parar quando é atingida a precisão limite da representação numérica no aparelho de cálculo; nesse caso a amplitude do intervalo é no máximo o valor da última casa. O erro absoluto máximo é esse valor.

critério do número de iterações

$$n = N$$

Consiste em parar quando tiverem sido feitas N iterações.

Devido às restrições aos três primeiros tipos de critérios, que nem sempre são fáceis de implementar, aconselha-se vivamente o uso de um dos dois últimos.

Para calcular o número de iterações efetivas para o último critério poderemos usar a regularidade da redução intervalar, a partir intervalo inicial, procurando que a máquina estabeleça o intervalo mais pequeno que pode representar. Com efeito, se, na máquina utilizada, a representação em vírgula flutuante for da forma $m.2^p$ e (a,b) for o intervalo inicial, deverá ser

$$n \leq n_m + \log_2(b-a)$$

em que n_m é o número de bits da mantissa[2].

[2]Usando a norma IEEE para float, o número de bits da mantissa – n_m – é 23. O número de iterações N depende então da amplitude do intervalo inicial, e contas bastante simples mostram que, para $b-a = 1$ então $N = 23$.

Exemplo: 22.7 Iterações de Bisseção

se constatarmos que $\log_b 1 = 0$, qualquer que seja a base b e que $\log_b x = \frac{\log_k x}{\log_k b}$, podemos facilmente construir uma tabela de valores de N em função da amplitude do intervalo inicial $b-a$:

$(b-a)$	N
100	30
10	26
1	23
0,1	20
0,01	17

Se não quisermos trabalhar directamente com a representação interna (binária) da máquina, que não é acessível às linguagens de programação normais, uma boa regra é tomar

$$n \leq 3.3 \times n_M + \log_{10}(b-a)$$

em que n_M é o número de dígitos da mantissa da representação decimal externa (isto é, dos outputs).

A verificação do *critério de Bolzano* para a existência de raiz, a deteção da mudança de sinal, pode ser feita pela multiplicação do valor da função nos extremos do intervalo, isto é, por $f(x_1) \times f(x_2) < 0$. Computacionalmente muito mais eficiente é, no entanto, a comparação directa dos sinais, recorrendo a funções que extraiam o bit de sinal da mantissa, ou mesmo a encadeamentos de condicionais de comparação de sinais ou expressões lógicas complexas

```
se   ( f(a) <= 0   E  f(m) >= 0 )  OU  ( f(a) > 0  E  f(m) <0)   então
raiz está no intervalo [ a, m ]
caso contrário
raiz está no intervalo ] m , b ]
```

Terminada deste modo a iteração, qual o valor que deve ser escolhido como melhor aproximação à raiz:

- o meio `m` do intervalo final, que conduz ao menor erro absoluto máximo?
- o extremo do intervalo que corresponde ao menor valor absoluto da função, sujeito à verificação da derivada na vizinhança, mas não esquecendo o caráter fundamental desta verificação;
- qualquer um dos valores `a, m, b`, no caso de fecho pela precisão limite?
- o intervalo `[a,b]` que contém a raiz?

O método da bissecção tem duas importantes vantagens do ponto de vista do cálculo automático:

- por um lado, a facilidade de programação;
- por outro lado, o facto de, mesmo quando as raízes não estão bem isoladas, o método permitir sempre encontrar uma delas, embora não denuncie a existência das outras (é precisamente por causa da existência desta propriedade que nos permitimos habitualmente relaxar a condição de isolamento da raiz para a condição de a função ter sinais contrários nos extremos do intervalo).

Apesar da sua extrema robustez, o método da bissecção tem um ponto fraco que é frequentemente ignorado: a existência de sinais contrários nos valores da função nos extremos do intervalo só garante a existência de um zero no intervalo se a função for contínua; no caso de uma função descontínua pode ocorrer um polo de primeira ordem e, então, o método da bissecção reduzirá o intervalo inicial a um pequeno intervalo na vizinhança desse polo, sem denunciar o facto (salvo se, acidentalmente, ocorrer um *overflow*); por isso, após a terminação da iteração, devemos sempre prever um teste dos valores de $f(a_n)$ e de $f(b_n)$ antes de aceitarmos o intervalo como enquadrante de um zero.

wxMaxima 2.5: find_root()

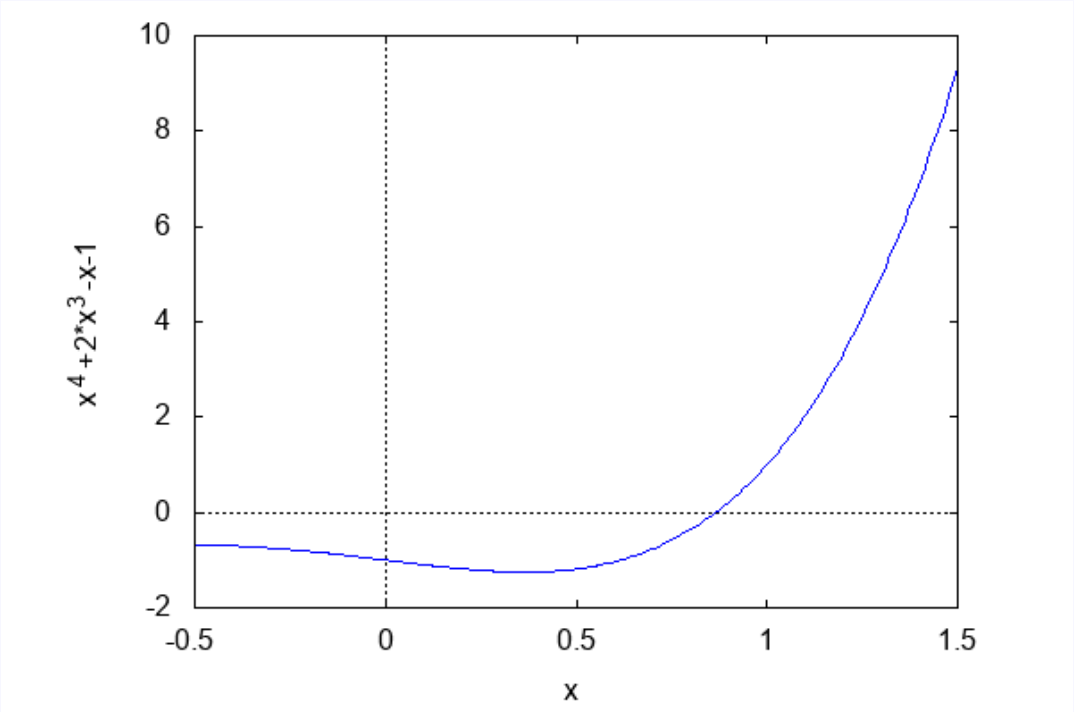
a função `find_root( f(x), x, a, b)` calcula uma raiz da função $f(x)$ no intervalo $[a,b]$ usando uma bisseção, levada ao limite da precisão de representação.

(% i5) f(x):= x^4 +2*x^3-x -1;

(% o5) $$f(x) := x^4 + 2x^3 - x - 1$$

(% i9) wxplot2d([f(x)], [x,-.5,1.5],
[gnuplot_postamble, "set zeroaxis;"])$

(% t9)



(% i10) find_root(f(x) , x , 0, 1.5);

(% o10) 0.866760399173862

(% i11) f(%);

(% o11) $-3.3306690738754710^{-16}$

Exercício 2.2

1. Construa um fluxograma para o método da bissecção, incluindo todos os cuidados na terminação.

2. Escreva um programa para o fluxograma anterior e teste-o sobre os exercícios e problemas anteri-

ores, ou sobre uma das seguintes funções:

$$\begin{aligned} f(x) &= 2^{\sqrt{x}} - 10x + 1 \\ f(x) &= x - \ln x - x \\ f(x) &= e^{\sin x}\cos(2x+1) \\ f(x) &= \cot x \sin(3x) - x - 1 \end{aligned}$$

3. No exemplo **??** os valores da amplitude do intervalo inicial deveriam ser potências de 2. Concorda (porquê ...)?
4. Discuta qual dos critérios de fecho descritos é mais adequado para implementação manual.

Excelente pela simplicidade e robustez para implementação informática, o método da bissecção é pouco apreciado no cálculo manual, devido a uma certa lentidão de convergência. Tal lentidão resulta essencialmente do facto de o algoritmo utilizar relativamente mal a informação que ele próprio vai gerando; com efeito, calculando os valores da função nos dois extremos do intervalo, utiliza apenas o sinal desses valores para o cálculo do novo ponto. O método que se apresenta em seguida procura explicitamente obviar a esse inconveniente, embora à custa de um cálculo ligeiramente mais trabalhoso.

2.3 Método da Corda

A ideia por trás do *método da corda*, também chamado *método da falsa posição* ou *regula falsi*, é muito simples: ao tentar reduzir o intervalo, se um valor extremo for grande (em valor absoluto) e o outro for pequeno, então o zero encontra-se, provavelmente mais perto do valor pequeno que do grande. Uma maneira simples da implementar este conceito consiste em traçar uma recta que passa pelos pontos extremos do intervalo, $(x_1, f(x_1))$ e $(x_2, f(x_2))$ e utilizá-la como aproximação da função. Esta recta tem por equação

$$y(x) = f(x_1) + \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1)$$

e tem o zero

$$x = \frac{x_1.f(x_2) - x_2.f(x_1)}{f(x_2) - f(x_1)}$$

que constitui o próximo ponto a ensaiar e, portanto, o próximo ponto de divisão do intervalo. Encontrado este novo ponto, calcula-se o valor $f(x)$ e abandona-se o ponto antigo em que a função tiver o mesmo sinal que neste, tal como se fazia na bissecção.

Algoritmo 2.2

```
Algoritmo da corda
DADOS a0 , b0  : f(a0 ).f(b0 )<0
    PARA n=0,1,2,... ATÉ ONDE FOR CONVENIENTE
        SEJA w =(an*f(bn) - bn*f(an))/(f(bn)-f(an))
        SE f(an ).f(w )<0  SEJAM an+1 = an ; bn+1 = w
        CASO CONTRÁRIO SEJAM an+1 = w; bn+1 = bn
```

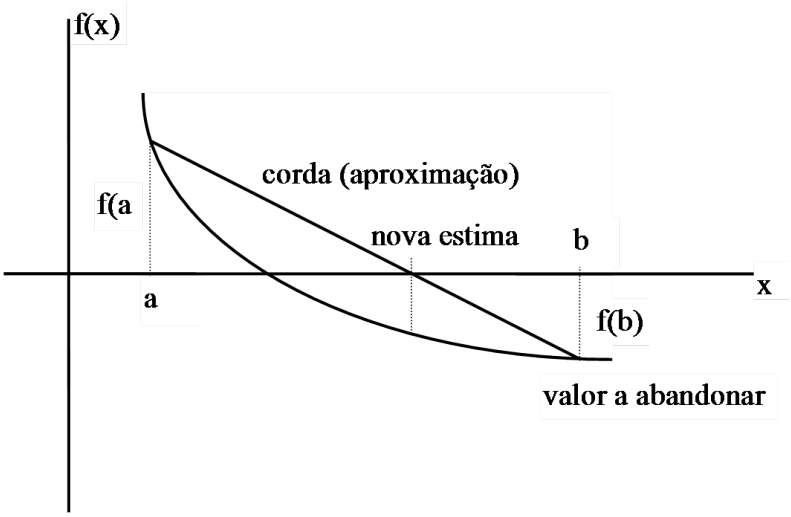


Figura 2.4: Interpretação geométrica do método da corda

O inconveniente desta técnica em relação à da bissecção consiste em não ser possível prever *a priori* o momento da paragem, o que obriga, a cada iteração, a testar se já se verifica a condição de terminação pretendida, o que pode corresponder a uma perda de tempo significativa, principalmente se o cálculo dos valores da função for relativamente expedito.

Exemplo: 22.8 Método da Corda

Vamos usar o método da corda para melhorar a raiz da equação

$$x^4 + 2x^3 - x - 1 = 0$$

situada no intervalo $(0,1)$, trabalhando com cinco algarismos significativos. Para efeitos de comparação, note-se que este é o mesmo problema que propusemos atrás para o método da bissecção.

a	f(a)	b	f(b)	x	intervalo
0,00000E+00	-1,00000E+00	1,00000E+00	1,00000E+00	5,00000E-01	1,00000E+00
5,00000E-01	-1,18750E+00	1,00000E+00	1,00000E+00	7,71429E-01	5,00000E-01
7,71429E-01	-4,99123E-01	1,00000E+00	1,00000E+00	8,47530E-01	2,28571E-01
8,47530E-01	-1,13991E-01	1,00000E+00	1,00000E+00	8,63132E-01	1,52470E-01
8,63132E-01	-2,20505E-02	1,00000E+00	1,00000E+00	8,66085E-01	1,36868E-01
8,66085E-01	-4,12384E-03	1,00000E+00	1,00000E+00	8,66635E-01	1,33915E-01
8,66635E-01	-7,66330E-04	1,00000E+00	1,00000E+00	8,66737E-01	1,33365E-01
8,66737E-01	-1,43018E-04	1,00000E+00	1,00000E+00	8,66756E-01	1,33263E-01
8,66756E-01	-2,68891E-05	1,00000E+00	1,00000E+00	8,66760E-01	1,33244E-01
8,66760E-01	-2,43988E-06	1,00000E+00	1,00000E+00	8,66760E-01	1,33240E-01
8,66760E-01	-2,43988E-06	1,00000E+00	1,00000E+00	8,66760E-01	1,33240E-01

Tabela 2.3: Aplicação do método da corda

A observação da tabela 2.3 e do gráfico da figura 2.5 permite compreender o desenvolvimento do método.

Exemplo: 22.8 Método da Corda (cont.)

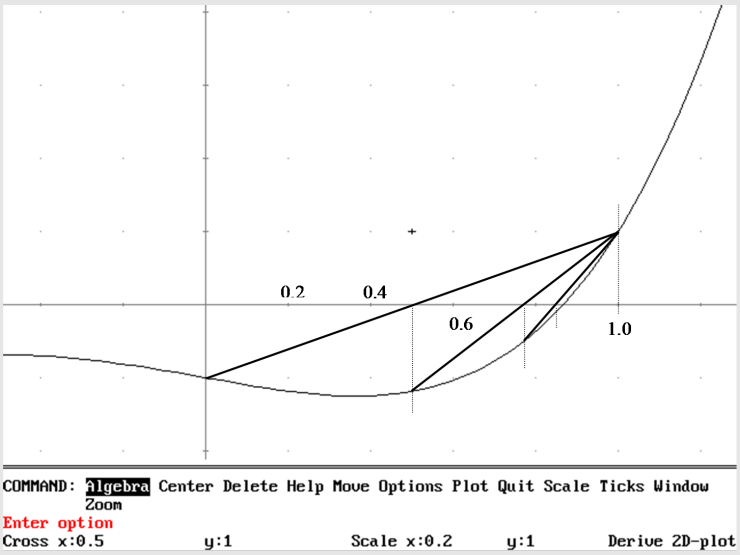


Figura 2.5: Aplicação do método da corda

O algoritmo termina na décima iteração porque se obtêm duas estimativas idênticas da raiz e não é possível, com esta precisão de representação, progredir para além deste ponto. A terminação obtém-se em 10 iterações, enquanto no método da bissecção, para o mesmo problema, precisamos de 17.

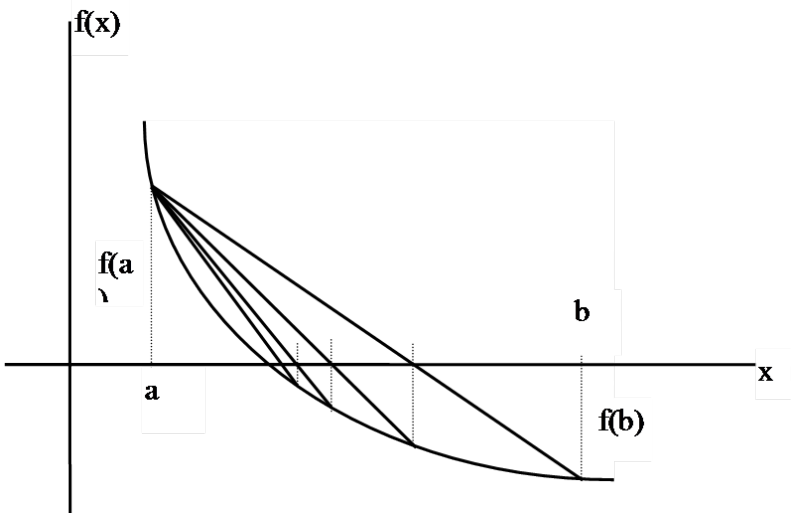


Figura 2.6: Convergência do método da corda

Para estudar a convergência do processo, admitamos que a raiz está isolada e que a segunda derivada da função tem sinal constante no intervalo (a,b) [3]. Suponhamos também, o que não arrasta perda de generalidade, que $f''(x) > 0$ (o caso $f''(x) < 0$ reduz-se a este fazendo $f(x) = -f(x)$, o que não afecta a raiz procurada). A curva é, portanto, côncava para cima, como na figura 2.6, e, portanto, situa-se inteiramente abaixo da corda.

[3] Se assim não sucedesse, o método levar-nos-ia, apesar de tudo, rapidamente a esta situação.

Neste caso acontece que o extremo a permanece fixo e as aproximações sucessivas são

$$x_{n+1} = x_n - \frac{f(x_n)}{f(x_n) - f(a)}.(x_n - a)$$

e formam uma sucessão monotonamente decrescente tal que

$$a < \xi < \ldots < x_{n+1} < x_n \ldots < x_1 < b$$

Concluímos, portanto, após uns momentos de reflexão, que

- o ponto que se torna fixo é o extremo em que o sinal da função coincide com o da sua segunda derivada;
- as aproximações sucessivas encontram-se, em relação à raiz, do lado em que a função tem sinal oposto ao da sua segunda derivada.

Além disso, implementado nesta forma, o método tem ainda um ponto fraco tornado evidente pelo resultado anterior: a partir do momento em que um extremo permanece fixo, a convergência é retardada pelo facto de a corda se aproximar cada vez mais da vertical. Por isso, é frequente usar uma variante que consiste em, a cada iteração, reduzir a metade o valor da função que se conserva[4].

Uma outra variante que frequentemente se ouve aconselhar é a de conservar sempre os dois últimos pontos calculados, dentro da ideia de que os valores mais recentes serão os mais próximos e, portanto, a redução do intervalo é mais drástica. Porém, pode suceder, e sucede frequentemente, que os dois pontos venham a cair do mesmo lado da raiz, o que invalida o princípio do método e o transforma, de *método de interpolação*, em *método de extrapolação*, com todos os riscos inerentes. Por este facto, uma tal variante, correntemente distinguida pelo nome de *método da secante*, é formalmente desaconselhável.

Exercício 2.3

1. Construa um fluxograma do método da corda e escreva um programa para esse fluxograma, propondo ao utilizador a escolha entre diferentes critérios de terminação; teste o programa sobre os exemplos e os exercícios propostos para o método da bissecção. Compare e comente os resultados.
2. Modifique o programa anterior para obter o método da corda melhorado e teste-o sobre os mesmos casos. Compare e comente os resultados.
3. A escolha, no método da corda melhorado, do divisor 2 para a ordenada do ponto fixo é, evidentemente, arbitrária. Discuta outras escolhas possíveis do divisor e as condições em que podem ser usadas com vantagem.
4. Mostre, por meio de diagramas, como pode falhar o método da secante.

2.4 Método da tangente

Os métodos discutidos atrás são *métodos intervalares* e têm o inconveniente óbvio de exigir o passo prévio de isolamento da raiz ou, pelo menos, de identificação de um intervalo em que a função mude de sinal, o que, como vimos, nem sempre é tarefa fácil.

Por isso se criou um método, dito *método da tangente* ou *método de Newton*, que parte apenas de um valor plausível, embora eventualmente grosseiramente errado, da raiz. Conceptualmente, pode ser considerado

[4] conhecido por vezes como método de Ilinois.

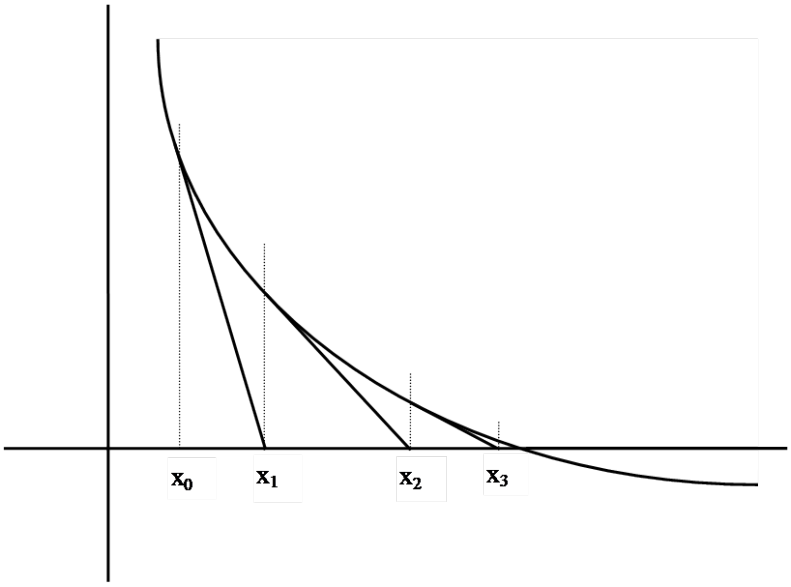


Figura 2.7: Método da tangente

como apenas uma extensão, ou passagem ao limite, do método da secante e tem, portanto, todos os inconvenientes potenciais deste: em termos geométricos, consiste em substituir o gráfico da função não pela secante que liga os pontos extremos do intervalo, mas pela tangente no ponto considerado, usando o zero desta como nova aproximação à raiz.

Seja x_k o valor aproximado actual; a equação da tangente à curva nesse ponto é

$$y(x) = f(x_k) + f'(x_k).(x - x_k)$$

de modo que o valor $y(x) = 0$ é

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

que constitui a nova aproximação.

Exemplo: 22.9 Método da Tangente

Seja, uma vez mais, o cálculo da raiz da equação

$$x^4 - 2x^3 - x - 1 = 0$$

tal que

$$f'(x) = 4x^3 - 6x^2 - 1$$

Comecemos pelo extremo direito $x = 1$, construindo a tabela 2.4.

n	x_n	$f(x_n)$ $x_n^4 - 2x_n^3 - x - 1$	$f'(x_n)$ $4x_n^3 - 6x_n^2 - 1$	x_{n+1} $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$
0	1	1	9	0,888889
1	0,888889	0,140071	6,55007	0,867504
2	0,867504	0,00455051	6,12679	0,866761
3	0,866761	3,67246E-06	6,11235	0,86676
4	0,86676	-2,43988E-06	6,11233	0,86676
5	0,86676	-2,43988E-06	6,11233	0,86676

Exemplo: 22.9 Método da Tangente (cont.)

Tabela 2.4: Aplicação do método da tangente

A convergência deu-se em 4 iterações o que, em comparação com os métodos anteriores, é muito bom.
Tentemos, porém, partir do outro extremo $x = 0$,construindo a tabela 2.5.

n	x_n	$f(x_n)$ $x_n^4 - 2x_n^3 - x - 1$	$f'(x_n)$ $4x_n^3 - 6x_n^2 - 1$	x_{n+1} $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$
0	0	-1	-1	-1
1	-1	-1	1	0
2	0	-1	-1	-1
3	-1	-1	1	0
4	0	-1	-1	-1

Tabela 2.5: Aplicação do método da tangente, com ponto inicial diferente.

O que se passou desta vez?
O gráfico da figura 2.8 ilustra a situação.

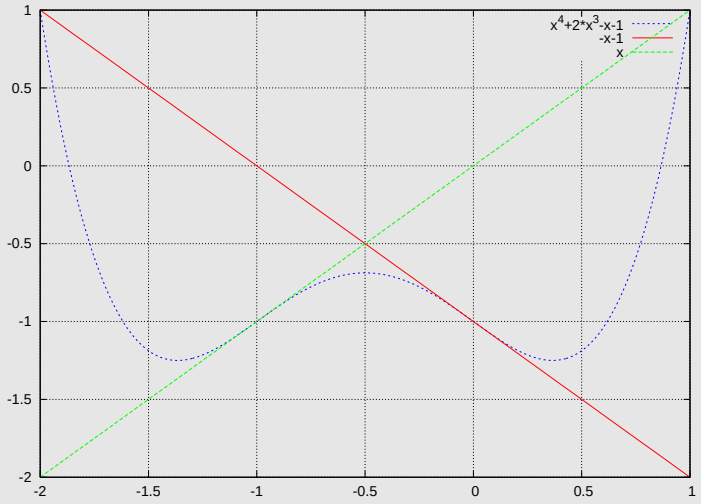


Figura 2.8: Não convergência no método da tangente

Quando funciona bem, o método da tangente é excelente, até porque, nas vizinhanças da raiz, tende, a cada iteração, a dobrar o número de algarismos exactos da solução. Porém, as suas limitações são muitas e muito severas, como mostra o exemplo anterior. A existência dessas limitações faz com que o método da tangente deva ser evitado, a menos que se conheça muito bem a estrutura local da função, o que é um tanto contraditório, porque, se essa estrutura local fosse bem conhecida não seria certamente difícil encontrar um intervalo de partida para um dos métodos anteriores.

Um inconveniente óbvio do método da tangente é o de exigir o conhecimento da derivada da função, e o seu cálculo em cada iteração; por este facto, tem sido sugerido que, em vez de calcular o valor da derivada em cada ponto, se trabalhe sempre com o seu valor no ponto original; uma tal estratégia, diminuindo o esforço de cálculo, aumenta o número de iterações necessárias para obter uma dada precisão. Além disso, em certos casos, pode tender a aumentar a robustez do método, mas uma tal propriedade depende, uma vez mais, da estrutura local da função.

Em suma, trata-se de um método indiscutivelmente útil, mas apenas para usar em terreno muito bem conhecido, e sempre com grande precaução.

2.5 Método de iteração de Picard-Peano

Todos os métodos apresentados até agora são, evidentemente, iterativos, visto usarem repetidamente o mesmo algoritmo para obterem aproximações sucessivamente melhores ao resultado pretendido e todos eles repousam na mesma ideia de substituir a função por uma aproximação adequadamente simplificada. O método, também iterativo, que vamos apresentar agora repousa sobre uma ideia completamente diferente e constitui, no plano conceptual, um dos mais importantes métodos de resolução numérica de equações, embora a sua aplicação prática tenda a ser um tanto limitada.

A ideia básica é a seguinte: suponhamos uma equação $f(x) = 0$ e, por não sabermos resolvê-la analiticamente, transformêmo-la de modo a dar-lhe a forma

$$x = g(x)$$

Se, por qualquer processo, tivermos obtido uma aproximação x_0 da raiz e a substituirmos no segundo membro, obtemos um valor $x_1 = g(x_0)$ que pode de novo ser utilizado para produzir $x_2 = g(x_1)$ e assim sucessivamente:

$$x_n = g(x_{n-1})$$

Se esta sucessão for convergente (o que, como veremos, não é de modo nenhum garantido), isto é, se existir o limite

$$\xi = \lim_{n\to\infty}(x_n)$$

então, por passagem ao limite na expressão do termo geral, obtemos

$$\lim_{n\to\infty}(x_n) = g\left(\lim_{n\to\infty}(x_{n-1})\right) = g\left(\lim_{n\to\infty}(x_n)\right)$$

isto é,

$$\xi = g(\xi)$$

o que prova que o limite, se existir, é, efectivamente, raiz da equação proposta.

Geometricamente, o método de iteração pode ser interpretado do seguinte modo: tracemos em um plano (x,y) os gráficos das funções

$$y = x$$
$$y = g(x)$$

e cada raiz ξ real da equação $x = g(x)$ será a abcissa de um ponto de intersecção da curva $y = g(x)$ com a recta $y = x$.

1. Se começarmos com uma abcissa x_0 , calculamos $y_0 = g(x_0)$, isto é, subimos de x_0 até encontrar a curva $y = g(x)$ em A_0;
2. em seguida, fazemos $x_1 = y_0$, isto é, deslocamo-nos horizontalmente até encontrar a recta $y = x$ em B_1 e descemos de novo até encontrar o eixo dos xx em $x1$;
3. Em seguida, calculamos de novo $y_1 = g(x_1)$, isto é, subimos de x_1 a A_1, e assim sucessivamente, subindo a "*escada*"A_0, B_1, A_1, B_1, A_2, B_3, A_3, ...(ver figura 2.9) até chegarmos ao ponto de intersecção da curva com a recta que, evidentemente, representa a raiz da equação proposta.

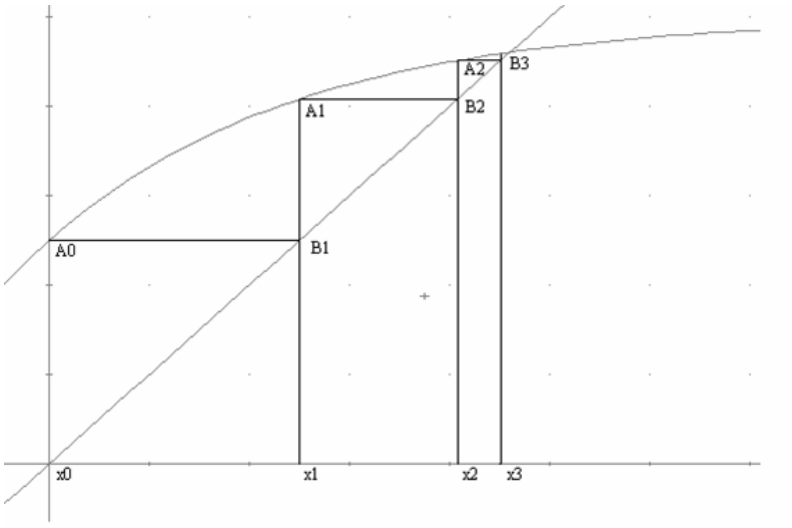


Figura 2.9: Método de Picard-Peano em escada

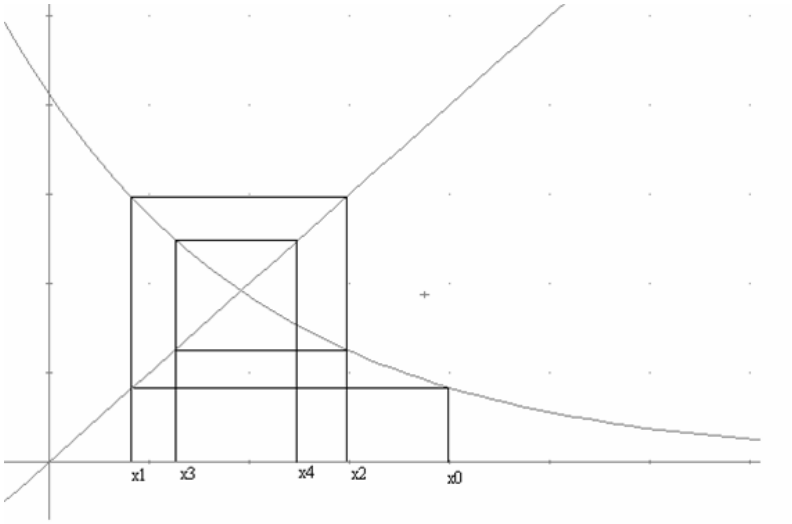


Figura 2.10: Método de Picard-Peano em teia de aranha

Uma outra configuração possível é a da "*teia de aranha*"(ver figura 2.10), correspondente ao caso de $y = g(x)$ ser decrescente, caso em que, em vez de uma sucessão de aproximações que tendem monotonamente para a raiz, obtemos uma sucessão oscilante que tende para a raiz enquadrando-a sucessivamente de mais perto.

Porém, nem sempre o método resulta convergente (não se esqueça o leitor que, na prova anterior, pusemos explicitamente a condição *se o limite* ξ *existir* ...), como resulta claramente das figuras seguintes (ver figuras 2.11 e 2.12).

Há, pois, para poder utilizar-se o método com segurança, que investigar as condições em que converge. Uma simples análise das figuras mostrará facilmente que a condição de convergência corresponde a ser

$$|g'(x)| < 1$$

Uma outra questão interessante que pode pôr-se é a de saber se, dado um conjunto de iterações ainda

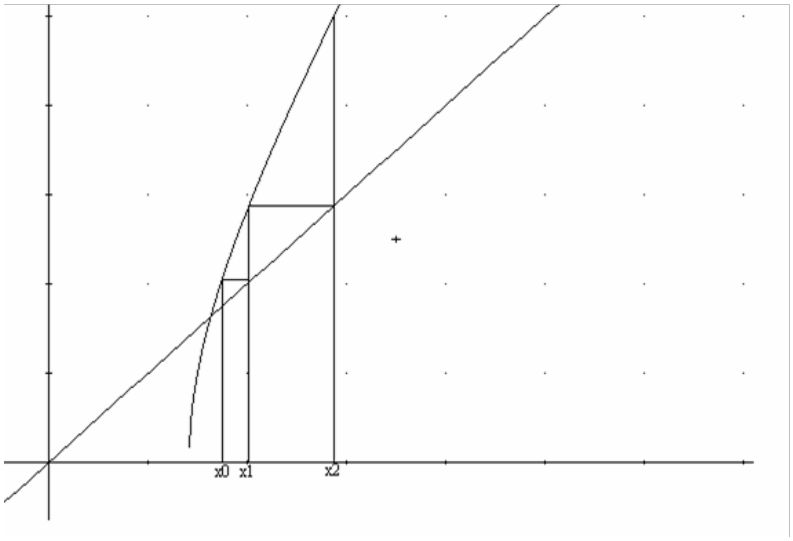


Figura 2.11: Método de Picard-Peano em escada divergente

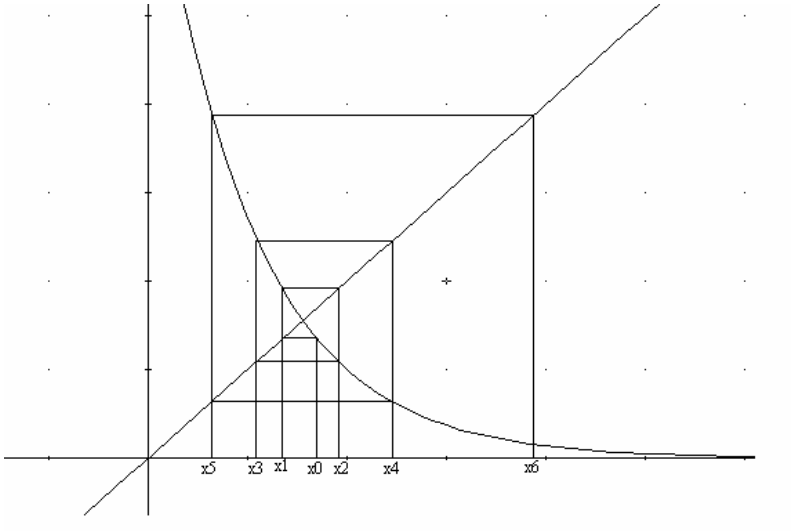


Figura 2.12: Método de Picard-Peano em teia de aranha divergente

não suficientemente aproximado da raiz, será possível acelerar o processo, extrapolando as tendências observadas.

Imaginemos uma coluna de iterações

$$\begin{aligned} & x_0 \\ & x_1 = g(x_0) \\ & x_2 = g(x_1) \\ & \cdots \\ & x_k = g(x_{k-1}) \\ x_{k+1} &= g(x_k) \end{aligned}$$

e suponhamos, antes de mais, que pretendemos interpolar entre x_k e x_{k+1}; para isso, teríamos que formar

$$(1-w).x_k + w.x_{k+1} = (1-w).x_k + w.g(x_k)$$

com $0 < w < 1$. Pois bem, para extrapolar, faremos exactamente o mesmo, mas com $w > 1$:

$$x^*_{k+1} = (1-w).x_k + w.g(x_k)$$

e procuremos determinar o melhor valor possível de w que será, naturalmente, função de k. Para isso, consideraremos que existe um número $z_k \in (x, x_k)$ ou $z_k \in (x_k, x)$ tal que

$$\begin{aligned}
\varepsilon_{k+1} &= \left(x^*_{k+1} - x\right) \\
&= (x_k - x) + w.\left[g(x_k) - g(x) - (x_k - x)\right] \\
&= (x_k - x) + w.\left[\frac{g(x_k) - g(x)}{x_k - x} - \frac{x_k - x}{x_k - x}\right].(x_k - x) \\
&= (x_k - x) + w.\left[g'(z_k) - 1\right].(x_k - x) \\
&= \left[1 + w.\left(g'(z_k) - 1\right)\right].\varepsilon_k
\end{aligned}$$

A escolha óptima de w seria aquela que anulasse o factor multiplicativo de ε_k; porém, dado que z_k não é conhecido, a nossa melhor opção é tomar $z_k = x_k$, isto é, fazer,

$$1 + w.\left(g'(x_k) - 1\right) = 0$$

ou, supondo $g'(x_k) \neq 1$

$$w = \frac{1}{1 - g'(x_k)}$$

isto é, considerando que $f(x) = x - g(x)$ e $f'(x) = 1 - g'(x)$

$$\begin{aligned}
x_{k+1} &= x_k - \frac{x_k}{1 - g'(x_k)} + \frac{g(x_k)}{1 - g'(x_k)} \\
&= x_k - \frac{f(x_k)}{f'(x_k)}
\end{aligned}$$

o que corresponde, senão à iteração óptima, pelo menos à melhor que podemos calcular.

Ora, esta iteração de Picard-Peano melhorada não é senão a iteração proposta pelo método da tangente. Por esta razão, preferiremos a variante de Newton, sempre que a derivada de $f(x)$ seja calculável sem complicação de maior.

Exercício 2.4 Da condição geral de convergência, $|g'(x)| < 1$, será capaz de deduzir uma condição particular adequada ao caso da variante de Newton?

Exemplo: 22.10 Aplicação de teste

As funções para testar um algoritmo de resolução de $f(x) = 0$ devem ter soluções conhecidas, por exemplo:

- $y = x$, cuja solução é 0, é um óptimo teste para a bisseção, mas um péssimo teste para os outros métodos;
- $y = x^3$, cuja solução também é 0, não permite facilmente construir uma expressão de Picard-

Exemplo: 22.10 Aplicação de teste (cont.)

Peano, mas pode ser usada para o método de Newton, onde resulta numa expressão particularmente simples

.

2.6 Resolução Iterativa de Sistemas de Equações

No caso geral, não linear, o problema da resolução numérica de sistemas de equações é um problema muito complexo, para o qual não existe grande variedade de estratégias disponíveis. Com efeito, quando falham os métodos analíticos (directos) ou semi-analíticos (substituições sucessivas até se obter uma única equação), o único método com razoável possibilidade de utilização genérica (e, mesmo assim, com as dificuldades que desde já se antevêem) é uma extensão óbvia do método de iteração, naturalmente, sempre que possível, na sua versão melhorada de método de Newton.

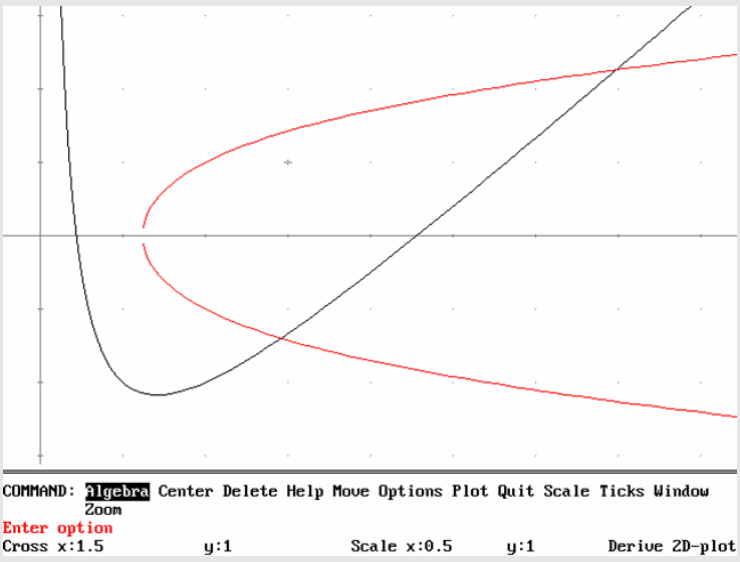
Exemplo: 22.11 Resolução de um sistema de duas equações a duas incógnitas

Seja o sistema

$$\begin{cases} f_1(x,y) = 2x^2 - x.y - 5x + 1 = 0 \\ f_2(x,y) = x + 3\log_{10}(x) - y^2 = 0 \end{cases}$$

de que pretendemos conhecer as raízes positivas, aproximadas a 5 casas significativas. Comecemos por traçar gráficos aproximados das funções f_1, f_2 e, nas suas intersecções obteremos, como primeira aproximação das raízes,

$$\begin{cases} x'_0 = +3,5 \\ y'_0 = +2,3 \end{cases} \qquad \begin{cases} x''_0 = +1,46 \\ y''_0 = -1,41 \end{cases}$$



wxMaxima 2.6: Repfresentaçãso do sistem

representação gráfica do sistema

```
(% i1)  load(implicit_plot)$
(% i2)  f1(x,y) := 2*x^2 -x*y -5*x+1;
```

(% o2) $$f1(x,y) := 2x^2 - xy + (-5)x + 1$$

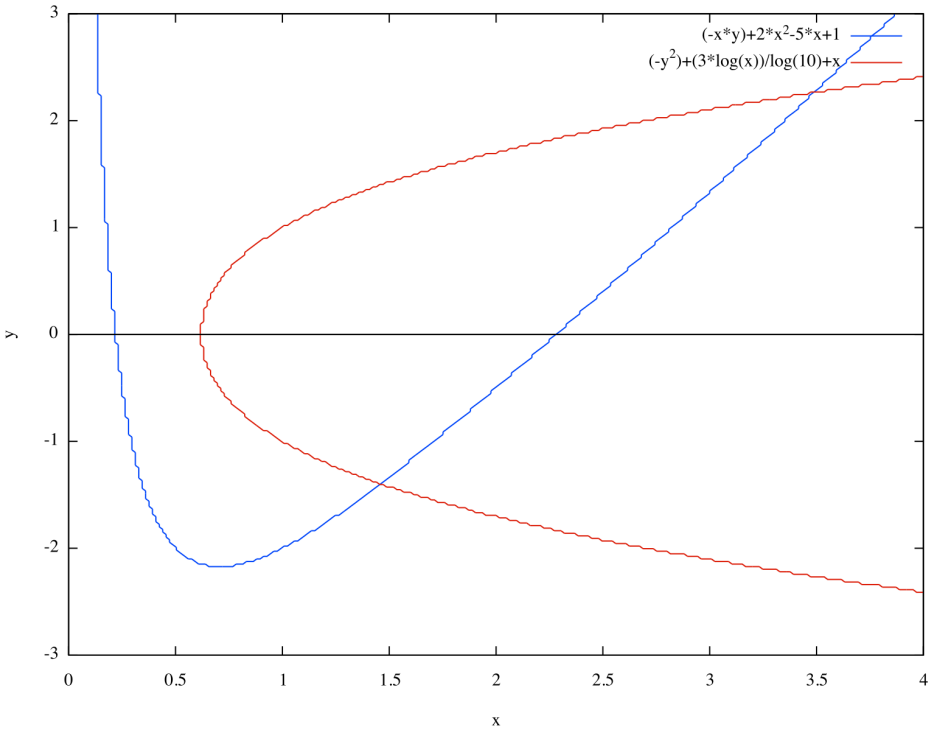
```
(% i3)  f2(x,y):= x+ 3*log(x)/log(10)-y^2;
```

(% o3) $$f2(x,y) := x + \frac{3\log(x)}{\log(10)} - y^2$$

```
(% i4)  implicit_plot([f1(x,y),f2(x,y)], [x,0,4], [y,-3,3])$
```

rat: replaced 2.302585092994046 by 11249839/4885743 = 2.302585092994044

(% t4)



Sejam dadas duas equações simultâneas em duas incógnitas

$$\begin{cases} f_1(x,y) = 0 \\ f_2(x,y) = 0 \end{cases}$$

de que pretendemos encontrar raízes reais com um dado grau de precisão. Sejam

$$\begin{cases} x = x_0 \\ y = y_0 \end{cases}$$

valores aproximados de uma raiz isolada, valor que foi obtido por um processo qualquer. Se representar-

mos o sistema na forma

$$\begin{cases} x = g_1(x,y) \\ y = g_2(x,y) \end{cases} \tag{2.1}$$

e construirmos aproximações sucessivas

$$\begin{cases} x_1 = g_1(x_0,y_0) \\ y_1 = g_2(x_0,y_0) \end{cases} \qquad \begin{cases} x_2 = g_1(x_1,y_1) \\ y_2 = g_2(x_1,y_1) \end{cases} \quad \cdots \quad \begin{cases} x_{n+1} = g_1(x_n,y_n) \\ y_{n+1} = g_2(x_n,y_n) \end{cases}$$

e se o processo de iteração convergir, isto é, se existir o duplo limite

$$\begin{cases} \xi = \lim_{n\to\infty}(x_n) \\ \eta = \lim_{n\to\infty}(y_n) \end{cases}$$

então

$$\begin{cases} \lim_{n\to\infty}(x_n) = \lim_{n\to\infty}(g_1(x_n,y_n)) \\ \lim_{n\to\infty}(y_n) = \lim_{n\to\infty}(g_2(x_n,y_n)) \end{cases}$$

e, portanto,

$$\begin{cases} \xi = g_1(\xi_n) \\ \eta = g_2(\eta) \end{cases}$$

isto é, o limite em questão é raiz do sistema proposto.

Porém, se o limite não existir, então o processo iterativo diverge e o método de iteração não pode ser usado, salvo mediante uma nova escolha mais adequada das funções $g_1(\ldots)$, $g_2(\ldots)$. O teorema seguinte, que não demonstraremos, dá uma condição de convergência:

Teorema 2.1 *Se, em uma vizinhança $(a \le x \le A, b \le y \le B)$ existir uma e uma só raiz (ξ,η), se $g_1(\ldots)$, $g_2(\ldots)$ forem diferenciáveis, se existirem tanto (x_0,y_0) como todas as aproximações sucessivas (x_n,y_n) e se forem válidas as desigualdades*

$$\left|\frac{\partial g_1}{\partial x}\right| + \left|\frac{\partial g_2}{\partial x}\right| \le q_x < 1 \qquad \left|\frac{\partial g_1}{\partial y}\right| + \left|\frac{\partial g_2}{\partial y}\right| \le q_y < 1$$

então o processo converge;
o teorema vale também para as desigualdades

$$\left|\frac{\partial g_1}{\partial x}\right| + \left|\frac{\partial g_2}{\partial y}\right| \le q_x < 1 \qquad \left|\frac{\partial g_1}{\partial y}\right| + \left|\frac{\partial g_2}{\partial x}\right| \le q_y < 1$$

A variante de Newton aplica-se na seguinte forma: sendo

$$\begin{cases} f_1(x,y) = 0 \\ f_2(x,y) = 0 \end{cases}$$

e escrevendo

$$\begin{cases} x = x_n + h_n \\ y = y_n + k_n \end{cases}$$

obtemos

$$\begin{cases} f_1(x_n+h_n, y_n+k_n)=0 \\ f_2(x_n+h_n, y_n+k_n)=0 \end{cases}$$

Se o *determinante jacobiano*

$$J(x_n,y_n)=\begin{vmatrix} f'_{1,x}(x_n,y_n) & f'_{1,y}(x_n,y_n) \\ f'_{2,x}(x_n,y_n) & f'_{2,y}(x_n,y_n) \end{vmatrix}$$

não for nulo, o sistema dá

$$h_n=-\frac{\begin{vmatrix} f_1(x_n,y_n) & f'_{1,y}(x_n,y_n) \\ f_2(x_n,y_n) & f'_{2,y}(x_n,y_n) \end{vmatrix}}{J(x_n,y_n)} \qquad k_n=-\frac{\begin{vmatrix} f'_{1,x}(x_n,y_n) & f_1(x_n,y_n) \\ f'_{2,x}(x_n,y_n) & f_2(x_n,y_n) \end{vmatrix}}{J(x_n,y_n)}$$

de modo que

$$\begin{cases} x_{n+1}=x_n-\dfrac{f_1(x_n,y_n).f'_{2,y}(x_n,y_n)-f_2(x_n,y_n).f'_{1,y}(x_n,y_n)}{f'_{1,x}(x_n,y_n).f'_{2,y}(x_n,y_n)-f'_{2,x}(x_n,y_n).f'_{1,y}(x_n,y_n)} \\ y_{n+1}=y_n-\dfrac{f_2(x_n,y_n).f'_{1,x}(x_n,y_n)-f_1(x_n,y_n).f'_{2,x}(x_n,y_n)}{f'_{1,x}(x_n,y_n).f'_{2,y}(x_n,y_n)-f'_{2,x}(x_n,y_n).f'_{1,y}(x_n,y_n)} \end{cases}$$

Exercício 2.5 Aplique o método de iteração ao sistema

$$\begin{cases} f_1(x,y)=2x^2-x.y-5x+1=0 \\ f_2(x,y)=x+3\log_{10}(x)-y^2=0 \end{cases}$$

na forma

$$\begin{cases} x=\sqrt{\dfrac{x.(y+5)-1}{2}}=g_1(x,y) \\ y=\sqrt{x+3\log_{10}(x)}=g_2(x,y) \end{cases}$$

usando os valores iniciais

$$\begin{cases} x'_0=+3,5 \\ y'_0=+2,3 \end{cases} \qquad \begin{cases} x''_0=+1,46 \\ y''_0=-1,41 \end{cases}$$

Aplique ao mesmo sistema a variante de Newton e compare resultados e desempenhos.

---

wxMaxima 2.7: Outra forma de calcular

Alguns exemplos não são perfeitos!
O sistema de duas equações a duas incógnitas que é proposto para resolução pelos métodos iterativos de Picard-Peano e Newton pode ser resolvido por um método misto de substituição, eliminando uma variável (neste caso y), depois resolvendo por um método numérico o problema reduzido apenas com a variável x, e por fim substituir as soluções na expressão que permite calcular y.
Pode ver esse cálculo aqui:

wxMaxima 2.7: Outra forma de calcular (cont.)

(% i1) f1: 2*x^2 -x*y-5*x+1;

(f1) $$-xy+2x^2-5x+1$$

(% i2) f2: x+3*log(x)/log(10)-y^2,numer;

(f2) $$-y^2+1.302883445709755\log(x)+x$$

escrever a primeira equação f1=0 como y = g1(x)

(% i3) -(+2*x^2-5*x+1)/(-x);

(% o3) $$-\frac{-2x^2+5x-1}{x}$$

(% i4) g1: ratsimp(%);

(g1) $$\frac{2x^2-5x+1}{x}$$

substituir g1 em f2 para obter uma equação apenas em x

(% i5) g2: subst(g1, y, f2);

(g2) $$1.302883445709755\log(x)-\frac{\left(2x^2-5x+1\right)^2}{x^2}+x$$

Construir o iterador de newton, apenas para desenhar um gráfico e identificar raízes

(% i6) x-g2 / diff(g2,x,1);

(% o6) $$x-\frac{1.302883445709755\log(x)-\frac{\left(2x^2-5x+1\right)^2}{x^2}+x}{\frac{2\left(2x^2-5x+1\right)^2}{x^3}-\frac{2(4x-5)\left(2x^2-5x+1\right)}{x^2}+\frac{1.302883445709755}{x}+1}$$

(% i7) itnew: ratsimp(%);

rat: replaced 1.302883445709755 by 14657229/11249839 = 1.302883445709756rat: replaced 1.302883445709755 b

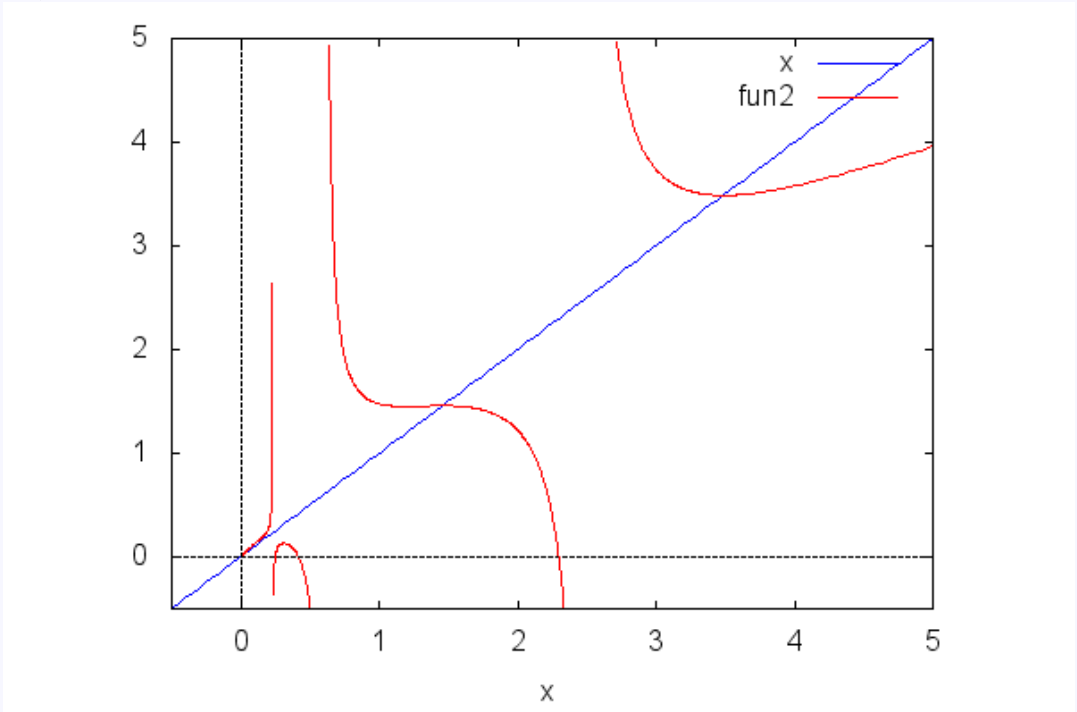
(itnew) $$\frac{14657229x^3\log(x)+44999356x^5-340902560x^3+224996780x^2-33749517x}{89998712x^4-236246619x^3-14657229x^2+112498390x-22499678}$$

(% i8) wxplot2d([x, itnew], [x,-0.5,5],[y,-0.5,5],
[gnuplot_postamble, "set zeroaxis;"])$

plot2d: expression evaluates to non-numeric value somewhere in plotting range.plot2d: some values were clipped.

wxMaxima 2.7: Outra forma de calcular (cont.)

(% t8)



O gráfico mostra duas soluções perto de 1.5 e 3.5. Parece também indicar 0 como raiz, mas tente calcular log(0) ...
Se usarmos a implementação do método de Newton para fazer os cálculos para a raiz da esquerda e depois para a da direita da expressão g2, escolhendo guesses adequados (observe com cuidado o gráfico):

(% i9) load(newton1);

(% o9)

/Applications/Maxima.app/Contents/Resources/opt/share/maxima/5.43.0/share/numeric/newton1.mac

(% i10) x1: newton(g2,x,1.5,0.001);

(x1) 1.458890152604047

(% i11) x2: newton(g2,x,3,0.001),numer;

(x2) 3.487442850379197

Calcular y usando a expressão g1, depois de transformada em função Maxima:

(% i12) fung1(x) := (2*x^2-5*x+1)/x;

(% o12) $$\text{fung1}(x) := \frac{2x^2-5x+1}{x}$$

(% i13) y1:fung1(x1);

(y1) −1.396767127842263

wxMaxima 2.7: Outra forma de calcular (cont.)

(% i14) y2:fung1(x2);

(y2) 2.261628750867807

Verificar as soluções (x1,y1) e (x2,y2)

(% i15) funf1(x,y):=2*x^2 -x*y-5*x+1;

(% o15) $\text{funf1}\,(x,y) := 2x^2 - xy + (-5)\,x + 1$

(% i16) funf2(x,y):= x+3*log(x)/log(10)-y^2;

(% o16) $\text{funf2}\,(x,y) := x + \dfrac{3\log(x)}{\log(10)} - y^2$

(% i17) funf1(x1,y1), numer;

(% o17) 0.0

(% i18) funf2(x1,y1),numer;

(% o18) $-4.78286195981780810^{-7}$

(% i19) funf1(x2,y2), numer;

(% o19) 0.0

(% i20) funf2(x2,y2), numer;

(% o20) $-4.58038099893087710^{-7}$

3 Sistemas de Equações Lineares

Contents

Tables

3.1 Eliminação Gaussiana

O problema de encontrar a solução numérica para um sistema de equações lineares simultâneas

$$\begin{cases} a_{11}.x_1 + a_{12}.x_2 + \ldots + a_{1n}.x_n = b_1 \\ a_{21}.x_1 + a_{22}.x_2 + \ldots + a_{2n}.x_n = b_2 \\ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}.x_1 + a_{n2}.x_2 + \ldots + a_{nn}.x_n = b_n \end{cases}$$

é um problema muito frequente, em larga medida porque tanto cientistas como engenheiros frequentemente não resistem à perigosa tentação de linearizar os seus problemas intrinsecamente não-lineares, cuja resolução é sempre incómoda e problemática.

O facto de dedicarmos um capítulo exclusivamente a este problema corresponde a reconhecer, mas não a endossar, esta tendência e também a reconhecer o seu carácter muito especial. Com efeito, do ponto de vista estrito da Análise Numérica, o problema dos sistemas lineares coloca-se, em certo sentido, no extremo oposto do da resolução de equações não-lineares que examinámos no Capítulo anterior: a sua solução canónica é de carácter directo mas iterativo; porém, devido ao risco de acumulação dos inevitáveis erros de arredondamento, quer dos dados quer dos resultados intermédios, o resultado final encontra-se frequentemente sujeito a erros grosseiros, pelo que há necessidade de tomar precauções especiais a fim de evitar essas situações desagradáveis.

Notemos por **A** a matriz quadrada dos coeficientes e por **b** e **x** as matrizes-colunas dos termos independentes e das incógnitas:

$$\mathbf{A} = \begin{Vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{Vmatrix} \quad \mathbf{b} = \begin{Vmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{Vmatrix} \quad \mathbf{x} = \begin{Vmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{Vmatrix}$$

de modo que o sistema pode ser escrito na forma simbólica $\mathbf{A}.\mathbf{x} = \mathbf{b}$ isto é,

$$\sum_{j=1}^{n} a_{ij}.x_j = b_i \qquad (i = 1,2,\dots,n)$$

Se a matriz quadrada $\mathbf{A}$ for não-singular, isto é, se $\det\{\mathbf{A}\} \neq 0$, então, como sabemos, o sistema tem solução x única que pode ser calculada pela *regra de Cramer*, a qual obriga a

$$N_n = N_{n-1} + (2n-1))$$

operações algébricas elementares (somas, subtracções, multiplicações e divisões). Se o leitor se der ao trabalho de calcular esta expressão recorrente até $n = 10$, por exemplo, dar-se-á conta, perante a enormidade dos números, da necessidade de encontrar um método computacionalmente mais económico. Um desses métodos é o *método de eliminação de Gauss*, habitualmente estudado nos cursos elementares de Álgebra e presente em muitas bibliotecas de rotinas matemáticas, que se desenvolve do seguinte modo:

- divida-se a primeira equação por a_{11}, para tornar unitário o primeiro coeficiente;
- multiplique-se esta nova primeira equação por $-a_{21}$ e some-se à segunda, obtendo-se uma nova segunda equação sem a incógnita x_1; em seguida multiplique-se a nova primeira equação por $-a_{31}$ e some-se à terceira, obtendo-se uma nova terceira equação, também sem a incógnita x_1 ; continue-se deste modo até que, após $n-1$ passos, se tenham $n-1$ equações que não contêm a incógnita x_1;
- repita-se o processo, trabalhando agora sobre a segunda coluna de coeficientes, de modo a eliminar a incógnita x_2 nas $n-2$ últimas equações, e assim sucessivamente, até que, se tudo correr bem, a última equação contém apenas a incógnita x_n com coeficiente unitário e , portanto, resolvida; o conjunto destes passos corresponde à fase de *triangularização* da matriz dos coeficientes;
- substitua-se o valor de x_n assim achado na penúltima equação (que tem apenas x_{n-1}, x_n) e resolva-se esta em ordem a x_{n-1} ; substituam-se os valores de x_{n-1}, x_n na penúltima equação e resolva-se esta em ordem a x_{n-2} ; continue-se deste modo até ter resolvido a primeira equação em ordem a x_1 ; o conjunto destes passos corresponde à fase de *substituição para trás* que culmina com a *diagonalização* da matriz do sistema.

Note-se de passagem que, neste processo, dividimos cada equação (depois de devidamente reduzida) pelo seu coeficiente em posição diagonal, num total de n divisões; ora, cada divisão divide o valor do determinante do sistema pela mesma quantidade e as outras operações (simples combinações lineares de linhas) não alteram o valor do determinante; além disso, o determinante do sistema triangularizado vale uma unidade, de modo que o determinante do sistema pode ser calculado pelo produto dos divisores utilizados, o que constitui uma regra de cálculo bem mais eficaz que a conhecida *regra de Sarrus*.

Exemplo: 33.1 Aplicação do Método de Gauss

Resolver o sistema:

$$\begin{cases} 3x+6y+9z=39 & (1)\\ 2x+5y-2z=3 & (2)\\ x+3y-z=2 & (3)\end{cases}$$

Divida-se (1) por 3:

$$\begin{cases} x+2y+3z=13 & (1')\\ 2x+5y-2z=3 & (2)\\ x+3y-z=2 & (3)\end{cases}$$

Multiplique-se $(1')$ por -2 e some-se a (2):

$$\begin{cases} x+2y+3z=13 & (1')\\ y-8z=-23 & (2')\\ x+3y-z=2 & (3)\end{cases}$$

Multiplique-se $(1')$ por -1 e some-se a (3):

$$\begin{cases} x+2y+3z=13 & (1')\\ y-8z=-23 & (2')\\ y-4z=-11 & (3')\end{cases}$$

Multiplique-se $(2')$ por -1 e some-se a $(3')$:

$$\begin{cases} x+2y+3z=13 & (1')\\ y-8z=-23 & (2')\\ 4z=12 & (3'')\end{cases}$$

Divida-se $(3'')$ por 4:

$$\begin{cases} x+2y+3z=13 & (1')\\ y-8z=-23 & (2')\\ z=3 & (3''')\end{cases}$$

Substitua-se $z=3$ em $(2')$:

$$\begin{cases} x+2y+3z=13 & (1')\\ y=1 & (2'')\\ z=3 & (3''')\end{cases}$$

substitua-se $z=3$ e $y=1$ em $(1')$:

$$\begin{cases} x=2 & (1'')\\ y=1 & (2'')\\ z=3 & (3''')\end{cases}$$

O valor do determinante do sistema é $3\times 1\times 4=12$.

Uma condição suficiente para que o algoritmo de Gauss seja viável é a de que nenhum dos divisores seja nulo; se tal condição se não verificar em dado passo, pode sempre pensar-se em trocar a ordem das colunas ainda não triangularizadas; se mesmo esse expediente falhar, o sistema não é resolúvel.

Algoritmo 3.1

```
  DADOS W(n,n+1)= A(n,n);b(n)
  INICIALIZE-SE p DE MODO QUE p(i) = i   (i = 1, 2, ..., n)
        PARA k =  1, 2, ..., n-1 {
                ACHAR O MENOR i >=  k TAL QUE W(i,k) != 0
                SE NÃO EXISTIR, A NÃO É INVERTÍVEL; PARAR
                CASO CONTRÁRIO, TROCAR p(k)  COM p(i)  E AS LINHAS k E I DE W
                PARA i = k+1, ..., n {
                        SEJAM m = W(k,k) , W(i,k) = W(i,k)/m
                                PARA j = k+1, ..., n+1 {
                                        SEJA W(i,j) = W(i,j) - m. W(k,j)
                                }
                }
        }
SE W(n,m)= 0 , A NÃO É INVERTÍVEL; PARAR
SEJA x(n) = b(n)
PARA k = n-1, ..., 1 {
    SEJA s= 0
    PARA j = k+1, ..., n {
        SEJA s = s + W(k,j).x(j)
    }
        SEJA x(k) = (b(k) - s)/W(k,k)
}
```

É fácil calcular o número de operações algébricas elementares necessárias para resolver um sistema de n equações pelo método de Gauss:

$$N_{Gn} = \frac{2}{3}.n.(n+1).(n+2) + n.(n-1)$$

e podemos compará-lo com o número de operações exigidas pela regra de Cramer. A economia obtida a partir de $n = 4$ torna-se verdadeiramente espectacular.

Exemplo: 33.2 Organização de cálculos

Uma das formas possíveis de organizar os cálculos é a que se apresenta a seguir. Repare na indexação das linhas, indicando **iteração.equação.transformação** e no registo da operação de que resulta a linha. Operações sobre colunas podem ser registadas na célula de título da coluna. São óbvias algumas regras de previsão e arrumação do espaço para os cálculos.

Operando sobre a totalidade da matriz:

índice	operação linha	x_1	x_2	x_3	b
1.1		0.100	0.700	-0.300	-19.300
1.1.1	**1.1** / 0.100	1.000	7.000	-3.000	-193.000
1.1.2	**1.1.1** × -(0.300)	- 0.300	- 2.100	0.900	57.900
1.1.3	**1.1.1** × -(3.000)	-3.000	21.000	9.000	579.000
1.2		0.300	-0.200	10.000	71.400
1.2.1	**1.2 + 1.1.2**	0	-2.3000	10.900	129.300
1.3		3.0	-0.100	-0.200	7.850
1.3.1	**1.3 + 1.1.3**	0	20.900	8.800	656.850

Operando sobre a submatriz [2-4]

Exemplo: 33.2 Organização de cálculos (cont.)

índice	operação linha	x_1	x_2	x_3	b
2.1	**1.1.1**	1.000	7.000	-3.000	-193.000
2.2	**1.2.1**	0	-2.300	10.900	129.300
2.2.1	**2.2** / (-2.300)	0	1.000	-4.739	-56.217
2.2.2	**2.2.1** × -(20.900)	0	-20.900	99.045	1174.935
2.3	**1.3.1**	0	20.900	8.800	656.850
2.3.1	**2.3** + **2.2.2**	0	0	107.845	1831.785

Operando sobre a submatriz [3-4]

índice	operação linha	x_1	x_2	x_3	b
3.1	**1.1.1**	1.000	7.000	-3.000	-193.000
3.2	**2.2.1**	0	1.000	-4.739	-56.217
3.3	**2.3.1**	0	0	107.845	1831.785
3.3.1	**3.3** / (107.845)	0	0	1.000	16.985

índice	operação linha	x_1	x_2	x_3	b
4.1	**1.1.1**	1.000	7.000	-3.000	-193.000
4.2	**2.2.1**	0	1.000	-4.739	-56.217
4.3	**3.3.1**	0	0	1.000	16.985

3.2 O Erro no Método de Gauss

O método de Gauss é, entre os métodos da Análise Numérica, um tanto invulgar pelo facto de, teoricamente, produzir uma solução exacta em um número finito de passos. No entanto, não seremos tão ingénuos que vamos supor que é isso exactamente que se passa na prática. Por um lado, e logo à partida, há os erros contidos na conversão dos dados (coeficientes, termos independentes) e que, embora raramente pensemos neles, podem ter consequências sérias.

Exemplo: 33.3 Representação binária exacta

Tome-se o caso de um número tão simples como $0.1_{(10)}$ que não tem representação binária exacta.

Exercício 3.1 Para confirmar a afirmação anterior, calcule $1_{(2)}$ / $110_{(2)}$.

Por outro lado, há os inevitáveis erros de arredondamento no decorrer das operações próprias do algoritmo, que podem produzir erros graves mesmo em um sistema de ordem baixa e, portanto, com poucas operações, como mostram os exemplos seguintes:

Exemplo: 33.4 Erros graves

Seja o sistema:

$$\begin{cases} x+\frac{1}{3}y=1 \\ 2x+\frac{2}{3}y=2 \end{cases}$$

que é indeterminado (porque o seu determinante é nulo) e que na nossa máquina imaginária seria representado na forma

$$\begin{cases} 0.100\times10^{1}x+0.333\times10^{0}y=0.100\times10^{1} \\ 0.200\times10^{1}x+0.667\times10^{0}y=0.200\times10^{1} \end{cases}$$

Exemplo: 33.4 Erros graves (cont.)

que é perfeitamente determinado (porque o determinante vale 0.100×10^{-2}) e se resolve na forma

$$\begin{cases} 0.100 \times 10^{1} x + 0.333 \times 10^{0} y = 0.100 \times 10^{1} \\ 0.100 \times 10^{-2} y = 0 \end{cases}$$

e dá

$$\begin{cases} x = 1 \\ y = 0 \end{cases}$$

o que é obviamente falso.
Note-se como é irónico que a técnica de arredondamento ($\frac{2}{3} \rightarrow 0.667 \times 10^{0}$) , destinada a diminuir o erro nos resultados finais, seja, neste caso, a responsável por um erro catastrófico: se, em vez de arredondar, tivéssemos truncado, o sistema continuaria indeterminado.

Exemplo: 33.5 Mais erros graves

Resolver, na **MAKINA**, o sistema

$$\begin{cases} x + 400y = 801 \\ 200x + 200y = 600 \end{cases}$$

cuja solução óbvia é

$$\begin{cases} x = 1 \\ y = 2 \end{cases}$$

Multiplicando a primeira equação por -200 somando o resultado à segunda, obtemos

$$\begin{cases} 0.100 \times 10^{1} x + 0.400 \times 10^{3} y = 0.801 \times 10^{3} \\ -0.798 \times 10^{5} y = -0.159 \times 10^{6} \end{cases}$$

o que dá

$$\begin{cases} x = 0.500 \times 10^{1} \\ y = 0.199 \times 10^{1} \end{cases}$$

O erro relativo em x é, portanto, de 400% e resulta da equação de substituição

$$x = 801 - 400y$$

que multiplica por 400 o erro cometido em y.

Torna-se, portanto, necessário, uma vez obtida a solução, tentar uma avaliação do erro cometido.

Por outro lado, não basta estimar esta *estabilidade externa* (isto é, em relação aos potenciais erros dos coeficientes e dos termos constantes): é necessário também estimar a *estabilidade interna* (isto é, em relação aos erros de arredondamento no decorrer do cálculo); notar-se-á que esta segunda forma de estabilidade não depende apenas das características do sistema de equações em questão, mas também do método de resolução adoptado e da precisão da representação dos números na máquina.

O erro na resolução de um sistema pode ser estudado por duas abordagens, o erro distribuído pelas equações e o erro distribuído pelas incógnitas.

Estamos a resolver o sistema de equações $A.x = b$. Da sua resolução numérica resulta uma solução aproximada x_0, que não satisfaz exatamente o sistema: $A.x_0 \approx b$.

Para restabelecer a identidade é necessário introduzir uma parcela de *erro*, que pode ser colocada em dois locais da equação:

$A.x_0 = b + \varepsilon$ este *erro* é designado como *resíduo*, e pode ser entendido como *o que falta a cada equação para se transformar numa identidade*, sendo calculável como

$$\varepsilon_0 = b - A.x_0$$

O resíduo, que é uma medida da qualidade da resolução do sistema, deve ser sempre calculado!

$A.(x_0 + \delta) = b$ este *erro* é associado à ideia de *estabilidade interna* do sistema, e pode ser entendido como *o que falta a cada solução para que a resolução do sistema seja exata*.

Vejamos então como obter uma estimativa da estabilidade interna: seja x_0 uma solução aproximada (a nossa solução!) tal que

$$x = x_0 + \delta$$

e teremos

$$A.(x_0 + \delta) = b$$

isto é

$$A.\delta = b - A.x_0 = \varepsilon$$

em que ε é obviamente a coluna dos resíduos (cujas componentes são os resíduos das equações, isto é, as diferenças entre os segundos membros e os valores calculados dos primeiros). Esta relação mostra que os δ podem ser obtidos por eliminação gaussiana sobre um sistema com matriz idêntica à do problema inicial, uma vez conhecida a solução deste. Note-se, porém, que esta segunda resolução será, por sua vez, afectada por erros do mesmo tipo dos da resolução do sistema inicial.

Destas considerações resulta claramente que, por trás da sua aparência directa e simples, o método de eliminação de Gauss contém muitas subtilezas ocultas.

É interessante também estudar o efeito da precisão dos próprios coeficientes e termos independentes na resolução dos sistema e valores da solução, o estudo da *estabilidade externa*. Para isso, partindo de novo do problema da forma

$$A.x = b$$

consideremos o problema perturbado

$$(A + \delta A).(x + \delta x) = b + \delta b$$

em que δA e δb são os erros (pequenos!) da matriz do sistema e dos termos constantes e δx é o erro resultante para a solução. Teremos, portanto, sucessivamente:

$$\begin{aligned} A.x + A.\delta x + \delta A.x + A.\delta x &= b + \delta b \\ A.\delta x + \delta A.x + \delta A.\delta x &= \delta b \end{aligned}$$

e, se os erros nos dados forem suficientemente pequenos, poderemos desprezar $\delta A.\delta x$ em face de $A.\delta x$ e $\delta A.x$, pelo que:

$$A.\delta x = \delta b - \delta A.x$$

o que mostra que é possível calcular os δx (aproximadamente, se os erros forem pequenos!) mediante o próprio processo de eliminação gaussiana sobre um sistema de matriz idêntica à do problema inicial e só com segundos membros diferentes, mas só depois de conhecer a solução x.

Exemplo: 33.6 Estabilidade externa

Seja o sistema

$$\begin{cases} 7x+8y+9z=24 \\ 8x+9y+10z=27 \\ 9x+10y+8z=27 \end{cases}$$

e resolvamo-lo em **Maxima** pelo método de eliminação gaussiana, usando a colecção de funções para álgebra linear[a]:

```
(%i1) load(linearalgebra);
```

mediante os comandos `echelon(M)`[b] e `rowop(M,i,j,t)`[c]:

```
(%i2) A:matrix(
 [7,8,9],
 [8,9,10],
 [9,10,8]
);
```

(%o2)
$$\begin{pmatrix} 7 & 8 & 9 \\ 8 & 9 & 10 \\ 9 & 10 & 8 \end{pmatrix}$$

```
(%i3) B:matrix(
 [24],
 [27],
 [27]
);
```

(%o3)
$$\begin{pmatrix} 24 \\ 27 \\ 27 \end{pmatrix}$$

```
(%i4) AB:addcol(A,B);
```

(%o4)
$$\begin{pmatrix} 7 & 8 & 9 & 24 \\ 8 & 9 & 10 & 27 \\ 9 & 10 & 8 & 27 \end{pmatrix}$$

```
(%i5) AB:echelon(AB);
```

(%o5)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{24}{7} \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

```
(%i6) AB:rowop(AB,2,3,2);
```

(%o6)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{24}{7} \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Exemplo: 33.6 Estabilidade externa (cont.)

```
(%i7) AB:rowop(AB,1,3,9/7);
```

(%o7)
$$\begin{pmatrix} 1 & \frac{8}{7} & 0 & \frac{15}{7} \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

```
(%i8) AB:rowop(AB,1,2,8/7);
```

(%o8)
$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

A solução é, portanto,

$$\begin{cases} x = 1 \\ y = 1 \\ z = 1 \end{cases}$$

e a estabilidade externa pode ser estimada mediante:

$$A.\delta x = \delta b - \delta A.x$$

em que δa é o erro absoluto dos coeficientes da equação e δb o dos segundos membros. Calculando mais uma vez em **Maxima**

```
(%i9) DA:zeromatrix(3,3)+da;
```

(%o9)
$$\begin{pmatrix} da & da & da \\ da & da & da \\ da & da & da \end{pmatrix}$$

```
(%i10) DB:zeromatrix(3,1)+db;
```

(%o10)
$$\begin{pmatrix} db \\ db \\ db \end{pmatrix}$$

```
(%i11) X:col(AB,4);
```

(%o11)
$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

```
(%i12) BP:DB-DA.X;
```

(%o12)
$$\begin{pmatrix} db - 3\,da \\ db - 3\,da \\ db - 3\,da \end{pmatrix}$$

```
(%i13) AP:addcol(A,BP);
```

Exemplo: 33.6 Estabilidade externa (cont.)

(%o13)
$$\begin{pmatrix} 7 & 8 & 9 & db-3da \\ 8 & 9 & 10 & db-3da \\ 9 & 10 & 8 & db-3da \end{pmatrix}$$

```
(%i14) AP:echelon(AP);
```

(%o14)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{db-3da}{7} \\ 0 & 1 & 2 & db-3da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i15) AP:rowop(AP,2,3,2);
```

(%o15)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{db-3da}{7} \\ 0 & 1 & 0 & db-3da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i16) AP:rowop(AP,1,3,9/7);
```

(%o16)
$$\begin{pmatrix} 1 & \frac{8}{7} & 0 & \frac{db-3da}{7} \\ 0 & 1 & 0 & db-3da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i17) AP:rowop(AP,1,2,8/7);
```

(%o17)
$$\begin{pmatrix} 1 & 0 & 0 & 3da-db \\ 0 & 1 & 0 & db-3da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

teremos, portanto,

$$\begin{cases} \delta x = 3\delta A - \delta b \\ \delta y = -3\delta A + \delta b \\ \delta z = 0 \end{cases}$$

o que mostra o facto surpreendente de (para este sistema particular!) a incógnita z ser insensível a (pequenos!) erros dos dados. Se supusermos que $\delta A = \delta b = 0.5$ teremos

$$\begin{cases} \delta x = 1 \\ \delta y = -1 \\ \delta z = 0 \end{cases}$$

[a] ver ficheiro SISTEMAS-estabilidade

[b] `echelon(M)` devolve uma matriz triangular superior de diagonal unitária obtida por eliminação gaussiana a partir da matriz `M`

[c] `rowop(M,i,j,t)` opera sobre a linha `i` da matriz `M` por $R_i \leftarrow R_i - t * R_j$

Exercício 3.2 Se o valor 0.5 que acabámos de usar fosse tomado como um majorante do erro dos dados, o resultado seria um majorante do erro do resultado?

Exemplo: 33.7 Estabilidade interna

A estabilidade interna do mesmo sistema pode, por sua vez, ser estimada mediante[a]:

```
(%i18) AX0:addcol(A,B-A.X0);
```

$$(\%o18)\qquad \begin{pmatrix} 7 & 8 & 9 & 0 \\ 8 & 9 & 10 & 0 \\ 9 & 10 & 8 & 0 \end{pmatrix}$$

```
(%i19) AX0:echelon(AX0);
```

$$(\%o19)\qquad \begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i20) AX0:rowop(AX0,2,3,2);
```

$$(\%o20)\qquad \begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i21) AX0:rowop(AX0,1,3,9/7);
```

$$(\%o21)\qquad \begin{pmatrix} 1 & \frac{8}{7} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i22) AX0:rowop(AX0,1,2,8/7);
```

$$(\%o22)\qquad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

o que significa que, os erros internos são nulos, como seria de esperar visto termos usado o modo de cálculo exacto, com representação racional.
Se, porém, tivéssemos calculado a seis casas significativas em vírgula flutuante, a solução do problema inicial seria, como o leitor pode verificar

$$\begin{cases} x &= 0.999257 \times 10^{0} \\ y &= 0.100069 \times 10^{1} \\ z &= 0.999973 \times 10^{0} \end{cases}$$

o que representa

$$\begin{cases} \delta x &= -0.743000 \times 10^{-3} \\ \delta y &= +0.690000 \times 10^{-4} \\ \delta z &= -0.270000 \times 10^{-4} \end{cases}$$

A precisão externa, calculada nos mesmo termos e com estes dados seria:

$$\begin{cases} \delta x &= +0.100046 \times 10^{0} \\ \delta y &= -0.100043 \times 10^{0} \\ \delta z &= +0.166583 \times 10^{-4} \end{cases}$$

Exemplo: 33.7 Estabilidade interna (cont.)

(note-se que estes números não podem comparar-se com os anteriores) A precisão interna, por seu lado, seria,

$$\begin{cases} \delta x &= +0.743674 \times 10^{-3} \\ \delta y &= -0.690639 \times 10^{-4} \\ \delta z &= +0.270489 \times 10^{-4} \end{cases}$$

Não há, portanto (como seria de esperar dado o carácter meramente aproximado do cálculo) total concordância entre estes valores e os efectivamente observados; este resultado mostra, como já tínhamos deixado apontado, que há que pôr de parte toda a esperança de, por este processo, suprir os erros de cálculo; com efeito, neste caso, se corrigíssemos os valores calculados das raízes por estes desvios calculados, a solução y pioraria substancialmente.

[a] ver ficheiro SISTEMAS-estabilidade

Porém, o método de Gauss não é de todo desprovido de esperança: o conceito de *vector dos resíduos*, ε , sugere, dada a linearidade do problema, uma técnica iterativa para melhoramento da solução:

1. seja x_0 uma primeira solução pelo método de Gauss e calcule-se $\varepsilon_0 = b - A.x_0$;
2. resolva-se em seguida o sistema $A.y_0 = \varepsilon_0$ e faça-se $x_1 = x_0 + y_0$, a nova aproximação;
3. calcule-se em seguida $\varepsilon_1 = b - A.x_1$;
4. verifique-se se $\|\varepsilon_1\| < \|\varepsilon_1\|$, usando por exemplo a *norma euclidiana*, decidindo se a nova aproximação é uma melhoria e se portanto vale a pena reiterar até que se obtenha um vector de resíduos ε_n satisfatóri, ou então abandonar a tentativa.

Dada, porém, a natureza sempre aproximada dos cálculos, não se podem depositar demasiadas esperanças neste esquema um pouco ingénuo, que pretende substituir o erro efetivo por um erro estimado [1], pelo que o número de iterações deve ser fixado sensatamente.

3.3 Técnicas Clássicas para Minimização dos Erros

Para evitar estes erros típicos do método de Gauss, têm, ao longo dos tempos, sido imaginados numerosos esquemas que, no fundo, se reduzem a combinações ou variantes das técnicas que descreveremos em seguida. No entanto, nenhum desses esquemas é perfeito. Na maior parte dos casos trata-se, fundamentalmente de evitar o aparecimento de valores muito altos como multiplicadores dos erros e de valores muito baixos como divisores das equações.

pivotagem parcial Já encontramos o conceito de *pivotagem parcial* quando, no algoritmo de eliminação, tratámos de evitar o embaraço de uma eventual divisão por zero e, para isso, fomos levados a permutar equações; no sentido mais geral, esta técnica consiste em, na eliminação de cada coluna, escolher, não o primeiro coeficiente não-nulo, mas o maior (em valor absoluto) coeficiente dessa coluna e em, naturalmente, usar a equação correspondente para proceder à eliminação. O efeito da pivotagem parcial é apenas o de uma reordenação das equações, embora essa reordenação não precise ser fisicamente implementada.

pivotagem total A *pivotagem total* consiste em escolher, não o maior coeficiente da coluna que se pretende eliminar, mas o maior de todos os coeficientes das equações ainda não tratadas. Existe razoável consenso entre os analistas numéricos no sentido de que a vantagem marginal produzida

[1] que se supõe pequeno, mas que é calculado por um processo que distribui mal quantidades pequenas ...

pela pivotagem total não compensa o esforço computacional envolvido no reordenamento total da matriz.

escalagem de linhas A *escalagem de linhas* consiste em, antes de iniciar o processo de eliminação, dividir cada equação por uma potência de 10 adequada, de modo a que o maior coeficiente se encontre entre 0.1 e 10; em termos mais gerais: se, em dada máquina, se utilizar a base β (o caso mais frequente é $\beta = 2$) para a aritmética de vírgula flutuante, divida-se cada equação pela potência de β que torna o seu maior coeficiente compreendido entre β^{-1} e β . Um outro esquema, mais simples de programar mas menos eficiente, consiste em dividir cada equação pelo seu mais alto coeficiente (ao contrário da técnica anterior, esta produz desde logo erros de arredondamento). Na ausência de erros, qualquer destas técnicas não afecta a solução teórica do sistema.

escalagem de colunas A *escalagem de colunas* é semelhante à de linhas, mas afecta as soluções no sentido de que, se uma coluna i for dividida por β^{α} , a solução x_i encontrada no final deve ser multiplicada por β^{α} .

Em relação às técnicas de escalagem, pode mostrar-se (cf. [JR75]) que, quando se usa aritmética de vírgula flutuante, a escalagem por uma potência da base de numeração é completamente inócua, salvo quando usada em conjunto com a pivotagem.

Exemplo: 33.8 Escalagem

Seja o sistema anterior

$$\begin{cases} x + 400y = 801 \\ 200x + 200y = 600 \end{cases}$$

que, escalado por linhas, dá

$$\begin{cases} 0.100 \times 10^{-2}x + 0.400 \times 10^{0}y = 0.801 \times 10^{0} \\ 0.200 \times 10^{0}x + 0.200 \times 10^{0}y = 0.600 \times 10^{0} \end{cases}$$

A técnica de pivotagem parcial exige que se comece a eliminação na 2^a equação e não na 1^a:

$$\begin{cases} 0.399 \times 10^{0}y = 0.798 \times 10^{0} \\ 0.100 \times 10^{1}x + 0.100 \times 10^{1}y = 0.300 \times 10^{2} \end{cases}$$

de onde resulta

$$\begin{cases} y = 0.200 \times 10^{1} \\ x = 0.100 \times 10^{1} \end{cases}$$

Note-se que, deste modo, um eventual pequeno erro de arredondamento em y (que neste caso não ocorreu) não seria amplificado por um coeficiente desastradamente alto, como sucedia no exemplo anterior.

3.4 Ordem e Condição de um Sistema

Como sabemos, a existência de uma *dependência linear entre linhas* (equações) de um sistema faz com que a sua solução seja *indeterminada*; na prática corrente do método de Gauss, esta situação é detectada pelo aparecimento de uma linha de zeros na matriz; na versão de pivotagem sistemática, porém, tal detecção não é automática; é, por isso, de boa prática, tentar sempre detectar a existência de um zero na coluna e, nesse caso, investigar se a respectiva linha é toda nula (incluindo o termo independente); se for este o caso, torna-se evidente que a equação correspondente é *linearmente dependente* das anteriores e que a variável que pretendíamos eliminar passa a constituir um *parâmetro* da solução do sistema (esta situação pode ocorrer mais que uma vez no decorrer da resolução de um mesmo sistema); se todos os

coeficientes forem nulos mas o termo independente o não for, a equação é *incompatível* com as anteriores e, portanto, o sistema é impossível.

Pode também suceder que surja uma coluna de zeros, precisamente aquela sobre a qual pretendíamos proceder à eliminação; isto significa, obviamente, que existe uma *dependência linear entre colunas*; neste caso o sistema é também indeterminado na incógnita correspondente (o que não impede que possa também ser impossível por outras razões).

Em qualquer dos casos (dependência de linhas ou de colunas), a *ordem* do sistema (ordem do maior determinante não nulo da matriz dos coeficientes) cai de uma unidade.

A ordem do sistema é, portanto, um indicador deste tipo de anomalias, mas um indicador pouco interessante porque não é capaz de distinguir uma situação de indeterminação de outra de impossibilidade. De qualquer modo, a consideração é irrelevante dado que, como vimos na página 86, a melhor maneira de calcular um determinante consiste em aplicar o método de Gauss à respectiva matriz.

Exemplo: 33.9 Dependência linear de colunas

Dependência linear de colunas (ver ficheiro SISTEMAS-deplincol):

```
(%i1) m:matrix(
 [2,4,1,13],
 [1,2,-1,2],
 [1,2,2,11]
);
```

$$(\%o1)\qquad \begin{pmatrix} 2 & 4 & 1 & 13 \\ 1 & 2 & -1 & 2 \\ 1 & 2 & 2 & 11 \end{pmatrix}$$

```
(%i2) echelon(m);
```

$$(\%o2)\qquad \begin{pmatrix} 1 & 2 & \frac{1}{2} & \frac{13}{2} \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Exemplo: 33.10 Dependência linear entre linhas

Dependência linear entre linhas (ver ficheiro SISTEMAS-deplinlin):

```
(%i1) m:matrix(
 [2,4,1,13],
 [1,2,.5,6.5],
 [1,1,1,6]
);
```

$$(\%o1)\qquad \begin{pmatrix} 2 & 4 & 1 & 13 \\ 1 & 2 & 0.5 & 6.5 \\ 1 & 1 & 1 & 6 \end{pmatrix}$$

```
(%i2) echelon(m);
```

$$(\%o2)\qquad \begin{pmatrix} 1 & 2 & \frac{1}{2} & \frac{13}{2} \\ 0 & 1 & -\frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Detectada que seja uma dependência linear, um bom programa deve identificá-la tão de perto quanto possível, porque é altamente provável que o utilizador pretendesse, ou supusesse, que o sistema tivesse a maior ordem possível. Em termos numéricos, o problema da ordem de um sistema é um problema complexo, até porque, no decorrer do cálculo, os efeitos do arredondamento tanto podem fazê-lo descer (por exemplo, quando ocorre perda total de significado em uma subtracção de linhas) como subir (como no exemplo atrás, relativo ao arredondamento dos dados).

Exercício 3.3 Resolver os sistemas:

1. $$\begin{cases} x+y+z & =6 \\ 3x-y+z & =-2 \\ 2x+0y+z & =2 \end{cases}$$

2. $$\begin{cases} x+3y+2z & =6 \\ x-y+0z & =-2 \\ x+0y+z & =2 \end{cases}$$

3. $$\begin{cases} x+z & =a \\ x+z & =b \\ x+z & =c \end{cases}$$

Dado que, em termos numéricos, a identificação do que é um valor nulo é sempre, em princípio, um problema muito complicado, sucede que a determinação da ordem de um sistema é, também, um problema complicado. No entanto, é óbvio que, por razões práticas, temos que construir um conceito que possa, pelo menos de forma parcial ou de modo aproximado, ajudar-nos a resolver o problema. Este conceito é o de *condição de um sistema*, o qual, por sua vez, dada a própria natureza do problema, é um conceito mal definido. A ideia geral é a de que um sistema é mal condicionado se pequenas variações dos coeficientes podem dar origem a variações desproporcionadamente grandes das soluções; ideia corresponde, portanto, aproximadamente à de *instabilidade externa* que atrás analisámos e o seu carácter vago resulta, naturalmente, do carácter vago das noções de "grande"e "pequeno"; em um outro sentido, o conceito de condição pode ser considerado como uma tentativa de formalizar um pouco a ideia de dependência linear face à inevitabilidade dos arredondamentos.

Se o mau condicionamento de um sistema resulta da própria natureza do processo físico que ele descreve, preferiremos falar em *instabilidade*. No entanto, nem todos os maus condicionamentos provêm do sistema físico, mas sim da formulação matemática que para ele escolhemos: em certo sentido, a discussão do Capítulo **??**Um sobre o cálculo numérico de expressões mostrou já como diferentes expressões analíticas do mesmo problema matemático podem conduzir a cálculos com precisões completamente diferentes; com efeito, em casos extremos, pode mesmo suceder que a própria estratégia de cálculo seja responsável pelo mau condicionamento; por exemplo, sabe-se que a pivotagem pode transformar um sistema bem condicionado em um mal condicionado.

Existem numerosos mal-entendidos acerca dos sistemas mal-condicionados:

- um deles é o de que um sistema adequadamente escalado que tem um determinante pequeno é necessariamente mal-condicionado;

- um outro é o de que o mau condicionamento se liga com a existência de um pequeno ângulo entre os gráficos de duas ou mais das equações.

Definiremos a *norma espectral* de uma matriz quadrada $\mathbf{A}$ como:

$$\|\mathbf{A}\| = \max_{\|\mathbf{x}\|=1} \|\mathbf{A}.\mathbf{x}\|$$

em que

$$\|\mathbf{x}\| = \sum_{i=1}^{n} x_i^2$$

é a *norma euclidiana* do vector $\mathbf{x}$.

exemplo:
Interpretação: Se $\mathcal{T}(\mathbf{x}) = \mathbf{A}.\mathbf{x}$ for a transformação linear produzida sobre o vector $\mathbf{x}$ pela matriz quadrada $\mathbf{A}$, a norma espectral da matriz pode ser interpretada como o máximo alongamento que sofre o raio de uma hiperesfera unitária que sofre essa transformação.

Definiremos a condição de uma matriz quadrada não-singular (isto é, invertível) como:

$$\text{cond}(\mathbf{A}) = \|\mathbf{A}\| \, . \left\|\mathbf{A}^{-1}\right\|$$

Esta definição não é, porém, adequadamente operacional no contexto do nosso problema, na medida em que implica o cálculo de $\mathbf{A}^{-1}$, que é, precisamente, o problema que pretendemos resolver. Em um tratamento mais sofisticado, o cálculo de $\text{cond}(\mathbf{A})$ baseia-se na teoria das forma quadráticas hermitianas, que não podemos aqui abordar; diremos apenas que pode ser estimado através das relações

$$\|\mathbf{A}\| \leq \sqrt{\sum_{j=1}^{n} \sum_{i=1}^{n} \left|a_{ij}\right|^2}$$

e

$$\text{cond}(\mathbf{A}) \geq \frac{\max_j \left\|a_{.j}\right\|}{\min_j \left\|a_{.j}\right\|}$$

O significado concreto da condição é dado pelas suas seguintes propriedades:

- se $\mathbf{A}$ for não-singular e

  $$\mathbf{A}.\mathbf{x} = \mathbf{b}$$
  $$\mathbf{A}.(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$$

  será

  $$\frac{\|\delta\mathbf{x}\|}{\mathbf{x}} \leq \text{cond}(\mathbf{A}).\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}$$

  o que significa que $\text{cond}(\mathbf{A})$ é um majorante da amplificação do erro relativo introduzido na solução pelo erro relativo dos segundos membros;

- Sendo

$$\mathbf{A}.\mathbf{x} = \mathbf{b}$$
$$\mathbf{A}.(\mathbf{x}+\delta\mathbf{x}) = \mathbf{b}+\delta\mathbf{b}$$

será

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}+\delta\mathbf{x}\|} \leq \text{cond}(\mathbf{A}).\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|} = r$$

e, se for $r < 1$, será ainda

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{r}{1-r}$$

o que significa essencialmente a mesma coisa para os erros induzidos por $\mathbf{A}$.

O ponto a reter é, portanto, o seguinte: se a condição da matriz dos coeficientes do sistema for elevada, deveremos ter pouca confiança na solução, mesmo que os cálculos sejam executados de modo muito exacto; naturalmente, a inevitável presença de arredondamentos só pode piorar esta situação. Em particular, usando aritmética de base 2 em vírgula flutuante com n_m bits na mantissa, então um valor da condição da matriz do sistema maior que 2^{n_m-1} indica que a solução encontrada terá, provavelmente, pouco significado. Pode, além disso, mostrar-se (cf. [Wil65]) que a solução $\mathbf{x}$ calculada pelo método de Gauss é solução exacta de uma equação da forma

$$(\mathbf{A}+\delta\mathbf{A}).\mathbf{x} = \mathbf{b}$$

com

$$\|\delta\mathbf{A}\| = \max_i \sum_{j=1}^{n} \left|\delta a_{ij}\right|$$

aproximadamente proporcional a n^3 . Este é um primeiro exemplo de uma análise inversa de erros, em que se mostra que a solução calculada aproximadamente para o nosso problema é solução exacta de um problema ligeiramente perturbado.

Todas estas considerações apontam para o facto de as fraquezas do método de Gauss e das suas variantes e "aperfeiçoamentos"serem bem mais graves que o que correntemente se supõe e que o que a sua presença sistemática nas bibliotecas de subrotinas matemáticas pode levar a crer. Na realidade, trata-se de um método que hoje já se não deve recomendar especialmente. O leitor aceitará que todo o tempo e esforço que lhe pedimos até ao momento se destinou a desenvolver a sensibilidade e o espírito crítico para as peculiaridades dos métodos de cálculo numérico e, por outro lado, para pôr em guarda contra o risco da excessiva confiança nos resultados que saem do computador com a benção de uma grande softwarehouse. Na realidade, o método que sugerimos como preferível é o que se apresenta de seguida.

O método de eliminação gaussiana constituiu objecto de extensos e profundos estudos, cuja história dá uma boa medida do que é, em termos reais, a construção das matemáticas. Como recorda Herman H. Goldstine (cf. [Gol90]:

> Dada uma matriz quadrada $n \times n$, real e invertível, $\mathbf{A}$, e uma matriz coluna $\mathbf{b} \in \mathbb{R}$, pretendemos calcular $\mathbf{x} = \mathbf{A}^{-1}.\mathbf{b}$.
>
> O processo conhecido como eliminação gaussiana, que constitui uma resposta ao problema, presta-se a implementação em computadores digitais automáticos e é também uma maneira de factorizar $\mathbf{A}$ em um produto $\mathbf{L}.\mathbf{U}$ em que $\mathbf{L}$ é triangular inferior e $\mathbf{U}$ é triangular superior.

Uma vez conhecidas **L** e **U**, a solução **x** é obtida resolvendo os dois sistemas triangulares

$$\mathbf{L}.\mathbf{c} = \mathbf{b}$$
$$\mathbf{U}.\mathbf{x} = \mathbf{c}$$

sendo a truncatura a única fonte de erro.

Em 1943 o famoso estatístico Hotelling (a resolução de sistemas de equações lineares é um problema constante e importante em Estatística) publicou uma análise que mostrava que o erro no cálculo do inverso aproximado, **X**, podia crescer como 4^{n-1}; em especial, mostrou, de modo heurístico e não muito rigoroso, que o método de Gauss exige cerca de $k + 0.6 \times n$ dígitos durante o cálculo para se obter uma precisão de k dígitos; assim, para inverter uma matriz de ordem 100 seriam necessários 70 dígitos se se quisesse uma precisão final de 10 dígitos. Começou então a temer-se que a eliminação gaussiana fosse instável com o erro de truncatura e iniciou-se uma procura frenética de algoritmos alternativos.

Em 1947, Goldstine e von Neumann, em um formidável artigo de 80 páginas, corrigiram em certa medida esta ideia. Certos estudiosos escolheram esta data como a do nascimento da moderna Análise Numérica. Entre outras coisas, este artigo mostrava como o uso sistemático de normas matriciais podia permitir a análise de erros; porém, um seu lamentável efeito involuntário foi o de deixar a impressão de que só matemáticos do calibre de Goldstine e von Neumann seriam capazes de empreender tais análises e, pior ainda, que tal trabalho era espantosamente maçador. Em termos técnicos, o principal resultado estabelecido era o de que, se **A** for simétrica e definida positiva, então o inverso calculado, **X**, satisfaz a

$$\|\mathbf{A}.\mathbf{X} - \mathbf{I}\| \leq 14.2 \times n^2 \times \varepsilon \times \mathrm{cond}(\mathbf{A})$$

com

$$\mathrm{cond}(\mathbf{A}) = \|\mathbf{A}\| \, . \, \left\|\mathbf{A}^{-1}\right\|$$

em que $\|.\|$ é a norma espectral e ε é a unidade de truncatura do computador. Só quando **A** é demasiado próxima de singular é que o algoritmo falha e não produz nenhum **X**, mas isso é exactamente o que seria de esperar. O interessante neste resultado era o carácter polinomial em n do erro e o mais difícil era estabelecer o valor do coeficiente numérico. Por outro lado, ninguém até hoje foi capaz de mostrar que um inverso computacionalmente correcto, **X**, tenha necessariamente que ser o inverso de uma matriz próxima de **A**, isto é, que

$$\mathbf{X} = (\mathbf{A} + \mathbf{E})^{-1}$$

com

$$\|\mathbf{E}\| / \|\mathbf{A}\| \ll 1$$

Na realidade, é altamente provável que tal condição não se cumpra na generalidade; o que pode mostrar-se é que cada coluna de **X** é a coluna correspondente da inversa de uma matriz próxima de **A**; infelizmente, será, em geral, uma matriz diferente para cada coluna.

Infelizmente, estes resultados referem-se apenas a um caso particular, o das matrizes simétricas e definidas positivas.

A experiência prática, anterior a 1963, da resolução de sistemas de equações com calculadoras mecânicas com n até 18 convenceu o matemático James Hardy Wilkinson, do National Physical Laboratory do Reino Unido, e os seus colegas L. Fox e E. T. Goodwin, de que a eliminação gaussiana dá excelentes resultados mesmo quando **A** está longe de ser simétrica e definida positiva. Escrevendo $\mathbf{L}.\mathbf{U} = \mathbf{A} + \mathbf{K}$, em que **K** é uma pequena "matriz de erro", os resultados que então publicou foram que a solução calculada, **z**, satisfaz a

$$(\mathbf{A} + \mathbf{K}).\mathbf{z} = \mathbf{b}$$

com (se os produtos internos forem acumulados em dupla precisão antes da truncatura final)

$$\|\mathbf{K}\| \leq g.\varepsilon(2.005 \times n^2 + n^3 + \frac{1}{4} \times \varepsilon \times n^4).|\mathbf{A}\|$$

em que g é o *factor de crescimento*, definido como o cociente entre o maior valor intermédio gerado no processo e o elemento máximo de $\mathbf{A}$. O correspondente majorante dos resíduos é

$$\|\mathbf{b} - \mathbf{A}.\mathbf{z}\| \leq g.\varepsilon(2.005 \times n^2 + n^3).|\mathbf{z}\mathbf{A}\|$$

sob a condição de ser

$$\varepsilon \times n \ll$$

A importante quantidade g é extremamente fácil de calcular no próprio decorrer da execução do algoritmo.

3.5 Método de Khaletsky

Dado um sistema

$$\mathbf{A}.\mathbf{x} = \mathbf{b}$$

procuremos representar $\mathbf{A}$ na forma do produto de uma matriz triangular inferior, $\mathbf{B}$, por uma matriz triangular superior de diagonal unitária, $\mathbf{C}$:

$$\mathbf{A} = \mathbf{B}.\mathbf{C}$$

com

$$\begin{Vmatrix} b_{11} & 0 & \dots & 0 \\ b_{21} & b_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & b_{nN} \end{Vmatrix} \qquad \begin{Vmatrix} 1 & c_{12} & \dots & c_{1n} \\ 0 & 1 & \dots & c_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{Vmatrix}$$

e é imediato verificar, procedendo à multiplicação, que

$$\begin{aligned} b_{i1} &= a_{i1} \\ b_{ij} &= a_{ij} - \sum_{k=1}^{j-1} b_{ik}.c_{kj} \quad (i \geq j) \\ c_{1i} &= \frac{a_{1i}}{b_{11}} \\ c_{ij} &= \frac{a_{ij} - \sum_{k=1}^{i-1} b_{ik}.c_{kj}}{b_{ii}} \quad (i < j) \end{aligned}$$

Então o vector procurado, $\mathbf{x}$, pode ser calculado a partir dos sistemas encadeados

$$\mathbf{B}.\mathbf{y} = \mathbf{b}$$
$$\mathbf{C}.\mathbf{x} = \mathbf{y}$$

sistema que é facilmente resolúvel por ter matrizes triangulares:

$$\begin{aligned} y_1 &= a_{1.n+1} \\ y_i &= \frac{a_{1.n+1} - \sum_{k=1}^{i-1} b_{ik}.y_k}{b_{ii}} \\ x_n &= y_n \\ x_i &= y_i - \sum_{k=i+1}^{n} b_{ik}.y_k \end{aligned}$$

Resulta evidente que os números y_i se calculam mediante o mesmo esquema que os c_{ij} e os x_i mediante o mesmo esquema que os b_{ij}, o que facilita notavelmente a programação. Além disso, o *método de Khaletsky* exige menos memória que o método de Gauss e torna-se mesmo particularmente simples:

- no caso das *matrizes simétricas* $a_{ki} = a_{ik}$, em que

$$c_{ij} = \frac{b_{ji}}{b_{ii}} \qquad (i < j)$$

- no caso de *matrizes definidas positivas* (isto é, tais que $\mathbf{x}^{\mathcal{T}}.\mathbf{A}.\mathbf{x} > 0$ qualquer que seja $\mathbf{x} \neq 0$), em que $\mathbf{C} = \mathbf{B}^{\mathcal{T}}$, o que reduz o cálculo à primeira fase, isto é, a cerca de metade.

Infelizmente, devido à sua menor divulgação e ao facto de, irracionalmente, muitos autores continuarem convencidos de que apenas se aplica ao caso das matrizes definidas positivas, não existem ainda estudos exaustivos sobre o erro deste método.

Exercício 3.4

1. Investigue as condições em que o método de Khaletsky pode falhar por envolver divisores nulos, e que tipos de soluções remediais se podem imaginar para esses casos.
2. Escreva um programa para resolver sistemas pelo método de Khaletsky e use-o sobre os exemplos trabalhados pelo método de Gauss, comparando tempos de execução, consumo de memória e resultados.

3.6 Métodos iterativos

Um método iterativo de resolução de sistemas de equações lineares pode ser vantajoso em relação a um método directo, quando, por exemplo, o número de equações é muito grande, e então os erros de truncatura e arredondamento propagados ao longo da aplicação de um método directo podem não compensar os erros afectos a uma solução aproximada resultante da aplicação de um método iterativo.

Pensemos no método de Picard-Peano ou da iteração simples que estudámos no capítulo anterior. É fácil estender a sua aplicação a um sistema de equações lineares.

Seja dado o sistema

$$\begin{cases} a_{11}x_1 & +a_{12}x_2 & + & \dots & +a_{1n}x_n = b_1 \\ a_{21}x_1 & +a_{22}x_2 & + & \dots & +a_{2n}x_n = b_2 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ a_{n1}x_1 & +a_{n2}x_2 & + & \cdots & a_{nn}x_n = b_n \end{cases}$$

que em notação condensada toma a forma:

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \ , \quad (1 \leq i \leq n)$$

Se em cada equação isolarmos uma incógnita no primeiro membro, teremos:

$$a_{ii}x_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} -a_{ij}x_j + b_i \ , \quad (1 \leq i \leq n)$$

Como estamos lembrados, do capítulo 2, uma das maiores limitações dos métodos iterativos é o facto de por vezes (não tão poucas vezes assim!), estes se afastarem progressivamente das soluções procuradas, divergindo. Portanto, também aqui, teremos que verificar as condições de convergência. Estas derivam das condições de convergência do método de Picard-Peano, sendo que para equações lineares as funções $g_i(x)$, do sistema de equações 2.1 são funções lineares e portanto as suas derivadas são constantes. Matematicamente, as condições de convergência dos métodos iterativos traduzem-se por:

$$\frac{1}{|a_{ii}|}\sum_{\substack{j=1\\ j\neq i}}^{n}|a_{ij}|<1\ ,\quad (1\leq i\leq n)$$

Estas condições de convergência, correspondem, em álgebra matricial, à imposição de a matriz dos coeficientes das incógnitas ser diagonalmente dominante (o módulo de cada elemento da diagonal principal é superior à soma dos restantes elementos da linha correspondente).

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \vdots & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{bmatrix}$$

A matriz A será de diagonal dominante se se verificar:

$$\begin{cases} |a_{11}|>\sum_{j=2}^{n}|a_{1j}|\\ |a_{22}|>\sum_{\substack{j=1\\ j\neq 2}}^{n}|a_{2j}|\\ \vdots\\ |a_{nn}|>\sum_{j=1}^{n-1}|a_{nj}|\end{cases}$$

3.6.1 Método de Gauss-Jacobi

A transposição do método de Picard-Peano para os sistemas de equações lineares denomina-se método de Gauss-Jacobi, que tem por fórmula de recorrência:

$$x_i^{(k)}=\frac{1}{a_{ii}}\left[-\sum_{\substack{j=1\\ j\neq i}}^{n}a_{ij}x_j^{(k-1)}+b_i\right]\ ,\quad (1\leq i\leq n)$$

3.6.2 Método de Gauss-Seidel

O método anterior pode ser melhorado, se em cada iteração, à medida que vamos obtendo uma nova aproximação para a incógnita x_{i-1} esta entrar no cálculo de x_i ainda na mesma iteração. Este melhoramento, designado por Gauss-Seidel, permite uma convergência mais rápida e traduz-se matematicamente por:

$$x_i^{(k)}=\frac{1}{a_{ii}}\left[-\sum_{\substack{j=1\\ j<i}}^{n}a_{ij}x_j^{(k)}-\sum_{\substack{j=1\\ j>i}}^{n}a_{ij}x_j^{(k-1)}+b_i\right]\ ,\quad (1\leq i\leq n)$$

É possível, em alguns casos, acelerar o processo de convergência do método de Gauss-Seidel através da introdução de um factor de relaxação w. A esta variante conhecida por sobrerelaxação sucessiva (SOR - Successive Overrelaxation) corresponde a fórmula de recorrência:

$$x_i^{(k)} = w\left\{\frac{1}{a_{ii}}\left[-\sum_{\substack{j=1\\ j<i}}^{n} a_{ij}x_j^{(k)} - \sum_{\substack{j=1\\ j>i}}^{n} a_{ij}x_j^{(k-1)} + b_i\right]\right\} + (1-w)x_i^{(k-1)} \quad , \quad (1 \leq i \leq n)$$

Note-se que para $w = 1$, estamos na presença do método de Gauss-Seidel.

Exemplo: 33.11 Aplicação de Gauss-Seidel

Seja o sistema de equações lineares:

$$\begin{cases} 7x+2y & = 24 \\ 4x+10y+z & = 27 \\ 5x-2y+8z & = 27 \end{cases}$$

Podemos verificar que a matriz dos coeficientes das incógnitas é diagonalmente dominante:

$$A = \begin{bmatrix} 7 & 2 & 0 \\ 4 & 10 & 1 \\ 5 & -2 & 8 \end{bmatrix} \qquad \text{Com efeito:} \quad \begin{bmatrix} 1^{\text{a}}\text{linha} & |7| > |2|+0 \\ 2^{\text{a}}\text{linha} & |10| > |4|+|1| \\ 3^{\text{a}}\text{linha} & |8| > |5|+|2| \end{bmatrix}$$

Isolando cada incógnita na equação correspondente, virá:

$$\begin{cases} x = \frac{24-2y}{7} \\ y = \frac{27-4x-z}{10} \\ z = \frac{27-5x+2y}{8} \end{cases}$$

Que corresponderá às seguintes fórmulas de recorrência:

Para o método de Gauss-Jacobi

$$\begin{cases} x_{n+1} = \frac{24-2y_n}{7} \\ y_{n+1} = \frac{27-4x_n-z_n}{10} \\ z_{n+1} = \frac{27-5x_n+2y_n}{8} \end{cases}$$

Para o método de Gauss-Seidel

$$\begin{cases} x_{n+1} = \frac{24-2y_n}{7} \\ y_{n+1} = \frac{27-4x_{n+1}-z_n}{10} \\ z_{n+1} = \frac{27-5x_{n+1}+2y_{n+1}}{8} \end{cases}$$

No quadro 3.1 apresentam-se os resultados dos processo iterativos para os dois métodos, com os respectivos resíduos, para quatro iterações.

Exemplo: 33.11 Aplicação de Gauss-Seidel (cont.)

Gauss - Jacobi				Gauss - Seidel		
	Guess inicial				Guess inicial	
x	0			x	0	
y	0			y	0	
z	0			z	0	
	1ªiteração	Resíduos			1ªiteração	Resíduos
x	3.428571	5.400000		x	3.428571	2.657143
y	2.700000	17.089286		y	1.328571	1.564286
z	3.375000	11.742857		z	1.564286	0.000000
	2ªiteração				2ªiteração	
x	2.657143	3.417857		x	3.048980	0.009184
y	0.991071	4.553571		y	1.323980	0.236097
z	1.907143	0.439286		z	1.800383	0.000000
	3ªiteração				3ªiteração	
x	3.145408	0.910714		x	3.050292	0.048269
y	1.446429	2.007972		y	1.299845	0.006854
z	1.962054	1.530612		z	1.793529	0.000000
	4ªiteração				4ªiteração	
x	3.015306	0.401594		x	3.057187	0.004146
y	1.245631	0.711735		y	1.297772	0.004828
z	1.770727	0.248916		z	1.788701	0.000000

Tabela 3.1: Aplicação dos Métodos de Gauss-Jacobi e Gauss-Seidel

Exercício 3.5 Verifique se o sistema do exemplo da secção 3.1 pode ser resolvido por métodos iterativos.

4 Quadratura e Cubatura

Contents

Figures

4.1 Introdução

Uma situação muito corrente tanto na Ciência como na Engenharia é a de a solução de um problema ser dada na forma de um integral que não pode ser calculado em forma cerrada (isto é, analítica) quer porque a integranda tem forma analítica intratável quer porque a integranda é dada sob a forma de uma tábua de valores ou, o que vem a dar no mesmo, é especificada através de um algoritmo que não tem expressão analítica cerrada. Há dois casos completamente diferentes desta situação:

- um é o caso do *integral indefinido*

$$F_a(x) = \int_a^x f(t)\,.dt$$

que, do ponto de vista numérico, exige a construção de uma tábua de valores e que consideraremos no Capítulo seguinte como caso particular, e particularmente simples, de integração de uma equação diferencial ordinária;

- o segundo caso é o do *integral definido*

$$F_a = \int_a^b f(x).dx$$

que é um número único.

A ideia geral para o cálculo numérico de um integral definido consiste basicamente em substituir a integranda $y = f(x)$ por uma função que a aproxime satisfatoriamente no intervalo $[a,b]$ e que seja dotada de expressão analítica primitivável, $g(x)$, escrevendo

$$\int_a^b f(x).dx \cong \int_a^b g(x).dx = [G(x)]_a^b$$

Posto nestes termos, o problema reduz-se ao problema de determinar uma aproximação adequada, $g(x)$, à função $y = f(x)$ no intervalo $[a,b]$. Uma solução frequentemente apontada que contempla de modo muito simples a questão da integrabilidade consiste em

1. definir $y = f(x)$ pelos seus valores em $n+1$ pontos do intervalo (a,b);
2. dar a $g(x)$ a forma de um polinómio $p_n(x)$ de grau n e obrigar este a passar pelos $n+1$ pontos conhecidos.

Deste modo, a solução será exacta para qualquer função $f(x) = a, a.x, ..., a.x^n$ e, naturalmente, também para qualquer combinação linear destas formas, isto é, para qualquer polinómio de grau igual ou inferior a n. No caso mais geral de uma função $y = f(x)$ qualquer, a solução será apenas aproximada e, frequentemente, apenas grosseiramente aproximada. Encontramo-nos, assim, perante o problema genérico muito complexo de definir uma aproximação analítica óptima a uma função dada e, naturalmente, antes de mais, de definir em que sentido é que essa função deve ser óptima. Infelizmente, não nos é possível abordar aqui esse problema genérico, pelo que teremos que procurar uma alternativa mais simples, embora com a consciência de que, assim, nos limitamos a soluções relativamente toscas. Essa alternativa será procurada dentro da hipótese de que a divisão do intervalo (a,b) em n subintervalos é suficiente fina para que o problema da qualidade das aproximações

$$g_i(x): \ x \in (x_{i-1}, x_i)$$

não seja crítico; devido à própria natureza desta hipótese, os métodos que exporemos em seguida aplicam-se apenas ao caso das funções contínuas; no caso das funções com descontinuidades de 1ª espécie (saltos), aplicar-se-ão, em separado, a cada um dos subintervalos resultantes da partição do intervalo original pelo ponto de descontinuidade; no caso das descontinuidades de espécie superior, esses métodos não são, pura e simplesmente, aplicáveis. Por razões de simplicidade, consideraremos apenas o caso da divisão do intervalo original em partes iguais:

$$\begin{array}{ll} x_{i+1} - x_i = h & \\ x_0 = a & y_i = f(x_i) \\ x_{n+1} = x_0 + n.h = b & \delta y_i = y_{i+1} - y_i \end{array}$$

4.2 Regra dos Trapézios

Neste método, a ideia central consiste em substituir, em cada intervalo, o arco da curva pela sua corda, calculando em seguida a área sob a poligonal assim definida.

Evidentemente, para o primeiro trapézio é

$$\int_{x_0}^{x_1} y.dx = \frac{h}{2}.[y_1 + y_0]$$

de modo que, para o segundo trapézio será:

$$\int_{x_1}^{x_2} y.dx = \frac{h}{2}.[y_2 + y_1]$$

e assim sucessivamente, até

$$\int_{x_{n-1}}^{x_n} y.dx = \frac{h}{2}.[y_n + y_{n-1}]$$

Somando todos os termos, temos

$$\int_{x_0}^{x_n} y.dx = \frac{h}{2}.[y_0 + 2.y_1 + ... + 2.y_{n-1} + y_n] \tag{4.1}$$

Esta fórmula é pouco interessante para o cálculo automático, sendo mais adequada:

$$\int_{x_0}^{x_n} y.dx = \frac{h}{2} \times \left[y_0 + y_n + 2 \times \sum_{\substack{i=1 \\ i=i+2}}^{n-1} y_i \right] \tag{4.2}$$

4.2.1 A "fórmula"do erro

A estimativa do erro cometido pode fazer-se considerando que $y = g(x)$ é apenas a aproximação de primeira ordem no desenvolvimento de $y = f(x)$ em série de Taylor em torno de x_i :

$$f(x)_i = f(x_i) + (x - x_i).f'(x_i) + \frac{(x - x_i)^2}{2}.f''(x_i) + \frac{(x - x_i)^3}{3!}.f'''(x_i) + \cdots$$

Substituindo esta expressão em ambos os membros de

$$\int_{x_i}^{x_{i+1}} y.dx = \frac{h}{2}.[y_{i+1} - y_i]$$

obtém-se o seguinte valor para o erro:

$$\varepsilon_i = -\frac{h^3}{12}.f''(x_i) + \cdots$$

ao qual, mediante o uso do teorema da média do cálculo diferencial (teorema de Lagrange), pode dar-se a forma:

$$\varepsilon_i = -\frac{h^3}{12}.f''(\xi_i) \qquad (x_i \leq \xi_i \leq x_{i+1})$$

Somando sobre todos os valores de i, obtemos

$$\begin{aligned}\varepsilon &= -\frac{n.h^3}{12}.f''(\xi) \\ &= -\frac{(x_n - x_0).h^2}{12}.f''(\xi) \qquad (x_0 \leq \xi_i \leq x_n) \\ &= -\frac{(x_n - x_0)^3}{12n^2}.f''(\xi)\end{aligned} \tag{4.3}$$

que fornece um majorante para o erro, pelo facto de depender de $f''(\xi)$, o máximo da derivada no intervalo.

Esta expressão é importante porque mostra que

- para uma dada amplitude do intervalo, o erro varia com o inverso do quadrado do número de intervalos ou, de outro modo, com o quadrado do passo de integração, pelo que o método dos trapézios se diz de *segunda ordem*;
- para um dado número de passos de integração (e sob a hipótese muito forte de que o majorante da segunda derivada se mantém), o erro cresce com o cubo da amplitude do intervalo, pelo que, se multiplicarmos o intervalo por k e quisermos manter a precisão, devemos multiplicar o número de passos por $k^{3/2}$ (o que mostra as sérias dificuldades que teremos se quisermos abordar por este método o problema do integral impróprio).

Exemplo: 44.1 Regra dos Trapézios

Seja o integral

$$I_{\pi/2} = \int_0^{\pi/2} \text{sen}(x).dx = 1$$

cujo erro *calculado* pode ser majorado pela expressão (4.3), mas cujo valor *observado* pode ser obtido pela diferença ao resultado da integração analítica. Calculando I pela regra dos trapézios (ver Integral_seno.xls), dá:

n	80	40	20	10	5
h	1,963495E-02	3,926991E-02	7,853982E-02	1,570796E-01	3,141593E-01
S	9,999679E-01	9,998715E-01	9,994859E-01	9,979430E-01	9,917618E-01
Erro Calc.	5,046595E-05	2,018638E-04	8,074551E-04	3,229820E-03	1,291928E-02
Erro Obs.	3,212782E-05	1,285138E-04	5,140948E-04	2,057014E-03	8,238231E-03

Vemos que, efectivamente, o erro diminui à medida que aumenta o número de passos da integração. Dado que, neste caso, foi possível estimar f'' com grande rigor, a estimativa do erro encontra-se bastante próxima do erro real.

Seja agora o integral

$$I_{\pi} = \int_0^{\pi} \text{sen}(x).dx = 2$$

que, calculado pela regra dos trapézios, dá:

n	80	40	20	10	5
h	3,926991E-02	7,853982E-02	1,570796E-01	3,141593E-01	6,283185E-01
S	1,999743E+00	1,998972E+00	1,995886E+00	1,983524E+00	1,933766E+00
Erro Calc.	4,037276E-04	1,614910E-03	6,459641E-03	2,583856E-02	1,033543E-01
Erro Obs.	2,570276E-04	1,028190E-03	4,114027E-03	1,647646E-02	6,623440E-02

O erro cometido é oito vezes maior que no caso anterior, como seria de esperar visto que a amplitude do intervalo duplicou.

Seja agora o integral

$$I_{3\pi/2} = \int_0^{3\pi/2} \text{sen}(x).dx = 1$$

Exemplo: 44.1 Regra dos Trapézios (cont.)

que, calculado pela regra dos trapézios, dá:

n	80	40	20	10	5
h	5,890486E-02	1,178097E-01	2,356194E-01	4,712389E-01	9,424778E-01
S	9,997108E-01	9,988431E-01	9,953693E-01	9,814256E-01	9,248584E-01
Erro Calc.	1,362581E-03	5,450322E-03	2,180129E-02	8,720515E-02	3,488206E-01
Erro Obs.	2,891653E-04	1,156862E-03	4,630663E-03	1,857436E-02	7,514159E-02

O erro observado é agora apenas nove vezes maior que o do primeiro exemplo, enquanto poderia esperar-se que fosse $3^3 = 27$ vezes maior; trata-se aqui de um típico efeito de compensação parcial dos erros em tramos de curvatura contrária. Pelo contrário, o erro calculado, é, efectivamente, 27 vezes maior, mas absolutamente fictício; torna-se evidente que, se não forem tomadas especiais precauções para garantir a sua validade, a "fórmula"do erro pode, para efeitos práticos, tornar-se absolutamente inútil.
Seja o integral

$$I_{2\pi} = \int_0^{2\pi} \operatorname{sen}(x)\,.dx = 0$$

que, calculado pela regra dos trapézios, dá:

n	80	40	20	10	5
h	7,853982E-02	1,570796E-01	3,141593E-01	6,283185E-01	1,256637E+00
S	-3,616790E-16	5,911324E-16	1,533436E-16	-7,697835E-17	-1,444196E-17
Erro Calc.	3,229820E-03	2,583856E-02	2,067085E-01	1,653668E+00	1,322934E+01
Erro Obs.	3,616790E-16	-5,911324E-16	-1,533436E-16	7,697835E-17	1,444196E-17

O efeito de compensação dos erros é, aqui, particularmente nítido, como seria de esperar. Um efeito mais surpreendente surge, porém: o aumento do número de passos de integração não aumenta de modo nenhum a precisão dos resultados porque os erros não provêm da aproximação do método em si (truncatura do processo de passagem ao limite, truncatura da série de Taylor) mas apenas dos efeitos de arredondamento do cálculo na máquina.

4.2.2 Controlo do erro

Na impossibilidade de calcular um erro ou, sequer, um seu majorante correspondente à aplicação do método a um dado caso, temos, necessariamente, que procurar um modo de avaliar o erro ou, pelo menos, mantê-lo sob controlo.

Um caminho possível e que é frequente ouvir-se recomendar é o seguinte: utilizar a regra dos trapézios para dois espaçamentos h e $h' = h/2$ e comparar os resultados; se estes não diferirem significativamente (o nível de significado terá sempre que ser fixado pelo utilizador em face das suas necessidades concretas mas, por razões de realismo, deve sempre ficar razoavelmente longe da precisão da representação dos números na máquina), poderemos razoavelmente esperar que tenhamos já um resultado correcto; no entanto, a experiência mostra que este critério é, mais frequentemente que o que gostaríamos, demasiado optimista; com efeito:

- o erro não depende apenas da amplitude do intervalo e do passo de integração, mas da própria forma da integranda, através do parâmetro incontrolável ξ ;
- a fórmula do erro só é válida para valores de *h suficientemente pequenos*, condição que não podemos saber *a priori* o que significa em cada caso concreto.

Um critério mais exigente, e que responde explicitamente a esta última consideração, parte de três cálculos, correspondentes a h, $h' = h/2$ e $h'' = h/4$, a que corresponderão os resultados calculados S, S' e

S'', tais que:

$$I = S + \varepsilon = S' + \varepsilon' = S'' + \varepsilon''$$

e em que, de acordo com a fórmula do erro, esperamos que

$$\varepsilon \approx 4.\varepsilon' \approx 16.\varepsilon''$$

de modo que

$$S' - S = \varepsilon - \varepsilon' \approx \tfrac{3}{4}.\varepsilon$$

$$S'' - S' = \varepsilon' - \varepsilon'' \approx \tfrac{3}{4}.\varepsilon' \approx \tfrac{3}{16}.\varepsilon$$

ou

$$\frac{S' - S}{S'' - S'} \approx 4 \tag{4.4}$$

Assim, se os números $(S' - S)$ e $(S'' - S')$ estiverem entre si na razão aproximada de 4 para 1, isso significa que, provavelmente, os passos de integração se encontram já todos dentro do domínio de validade da fórmula do erro; nestas condições, $(S'' - S')$ pode servir como estimador de ε'' na forma

$$S'' - S' \approx 3\varepsilon'' \tag{4.5}$$

o que nos permitirá estimar se S'' se encontra já dentro da precisão desejada. Como é evidente, pode acontecer que o cumprimento da condição (4.4) exija uma precisão superior àquela em que estamos interessados, o que corresponderia a um certo desperdício de esforço de cálculo, certamente um preço justo a pagar pela segurança.

Exemplo: 44.2 Cociente de convergência

Apliquemos este raciocínio aos integrais do exemplo anterior (ver Integral_seno.xls): Para $I_{\pi/2}$ temos:

n	80	40	20	10	5
h	1,963495E-02	3,926991E-02	7,853982E-02	1,570796E-01	3,141593E-01
S	9,999679E-01	9,998715E-01	9,994859E-01	9,979430E-01	9,917618E-01
cociente	4,000386E+00	4,001543E+00	4,006184E+00		
ε''	3,212865E-05	1,285270E-04	5,143063E-04		
Erro Obs.	3,212782E-05	1,285138E-04	5,140948E-04	2,057014E-03	8,238231E-03

o que significa que, a partir de $n = 20$, já o passo é suficiente para funcionar como infinitesimal . Por outro lado, observar-se-á que esta nova estimativa do erro é francamente razoável.

Para I_π temos:

n	80	40	20	10	5
h	3,926991E-02	7,853982E-02	1,570796E-01	3,141593E-01	6,283185E-01
S	1,999743E+00	1,998972E+00	1,995886E+00	1,983524E+00	1,933766E+00
cociente	4,001543E+00	4,006184E+00	4,024930E+00		
ε''	2,570540E-04	1,028613E-03	4,120812E-03		
Erro Obs.	2,570276E-04	1,028190E-03	4,114027E-03	1,647646E-02	6,623440E-02

o que significa que, também neste caso, $n = 20$ é perfeitamente suficiente.

O mesmo acontece para $I_{3\pi/2}$:

n	80	40	20	10	5
h	5,890486E-02	1,178097E-01	2,356194E-01	4,712389E-01	9,424778E-01
S	9,997108E-01	9,988431E-01	9,953693E-01	9,814256E-01	9,248584E-01
cociente	4,003475E+00	4,013960E+00	4,056830E+00		
ε''	2,892322E-04	1,157934E-03	4,647900E-03		
Erro Obs.	2,891653E-04	1,156862E-03	4,630663E-03	1,857436E-02	7,514159E-02

Para $I_{2\pi}$, porém, o cálculo deixa claramente de ser de confiança dado o número ridiculamente baixo de casas significativas que retivemos no resultado:

Exemplo: 44.2 Cociente de convergência (cont.)

n	80	40	20	10	5
h	7,853982E-02	1,570796E-01	3,141593E-01	6,283185E-01	1,256637E+00
S	-3,616790E-16	5,911324E-16	1,533436E-16	-7,697835E-17	-1,444196E-17
cociente	-4,594706E-01	5,261029E-01	-2,715173E-01		
ε''	-3,176038E-16	1,459296E-16	7,677398E-17		
Erro Obs.	3,616790E-16	-5,911324E-16	-1,533436E-16	7,697835E-17	1,444196E-17

Para n crescente, o cociente (4.4) começa em geral por diminuir até próximo do valor teórico, por efeito da diminuição do erro de truncatura, voltando mais tarde a aumentar, o que testemunha domínio do erro de arredondamento sobre o erro de truncatura. Se, porém, o crescimento se inicia antes de o cociente se ter sequer aproximado do valor teórico, isso indica claramente que o método não é aplicável com a precisão de representação da máquina que estamos a utilizar.

Exemplo: 44.3 Evolução do Quociente de Convergência para a Regra dos Trapézios

Vamos calcular o integral definido

$$I = \int_0^{20*\pi+0.5} \frac{x\sin(x)}{5}\,\mathrm{d}x$$

primeiro recorrendo ao **`Maxima`** para obter uma solução analítica.

Exemplo: 44.3 Evolução do Quociente de Convergência para a Regra dos Trapézios (cont.)

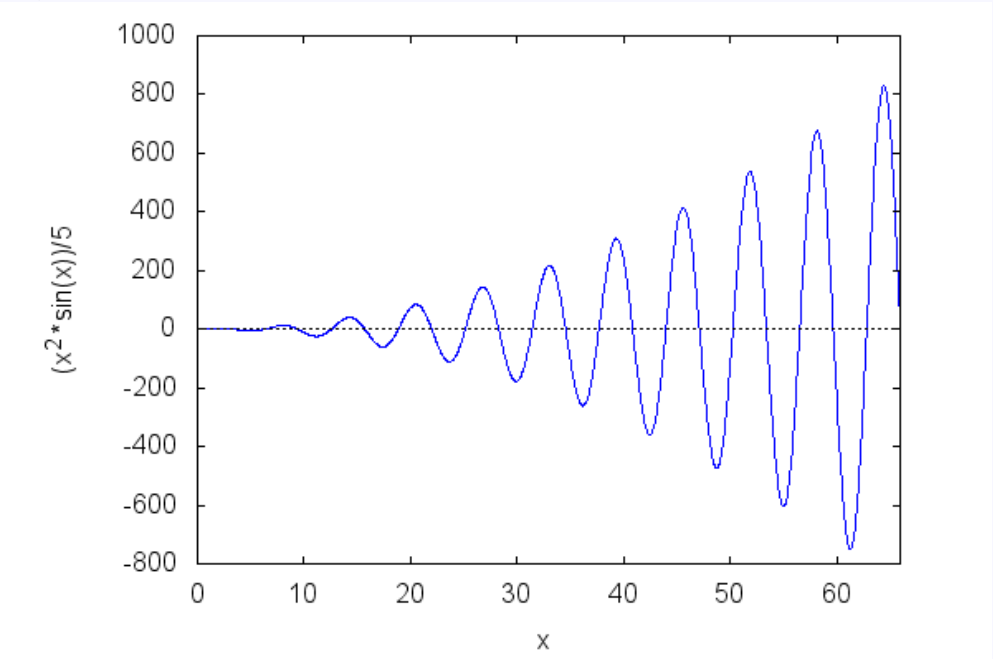
wxMaxima 4.1: Integral analítico

(% i1) f(x):= x^2*sin(x)/5;

(% o1) $$f(x) := \frac{x^2 \sin(x)}{5}$$

(% i2) wxplot2d([f(x)], [x,0,21*%pi],
[gnuplot_postamble, "set zeroaxis;"])$

(% t2)



(% i3) integrate(f(x), x);

(% o3) $$\frac{2x\sin(x) + \left(2 - x^2\right)\cos(x)}{5}$$

(% i4) integrate(f(x), x, 0, 20*%pi+0.5),numer;

(% o4) $$-691.8871279250529$$

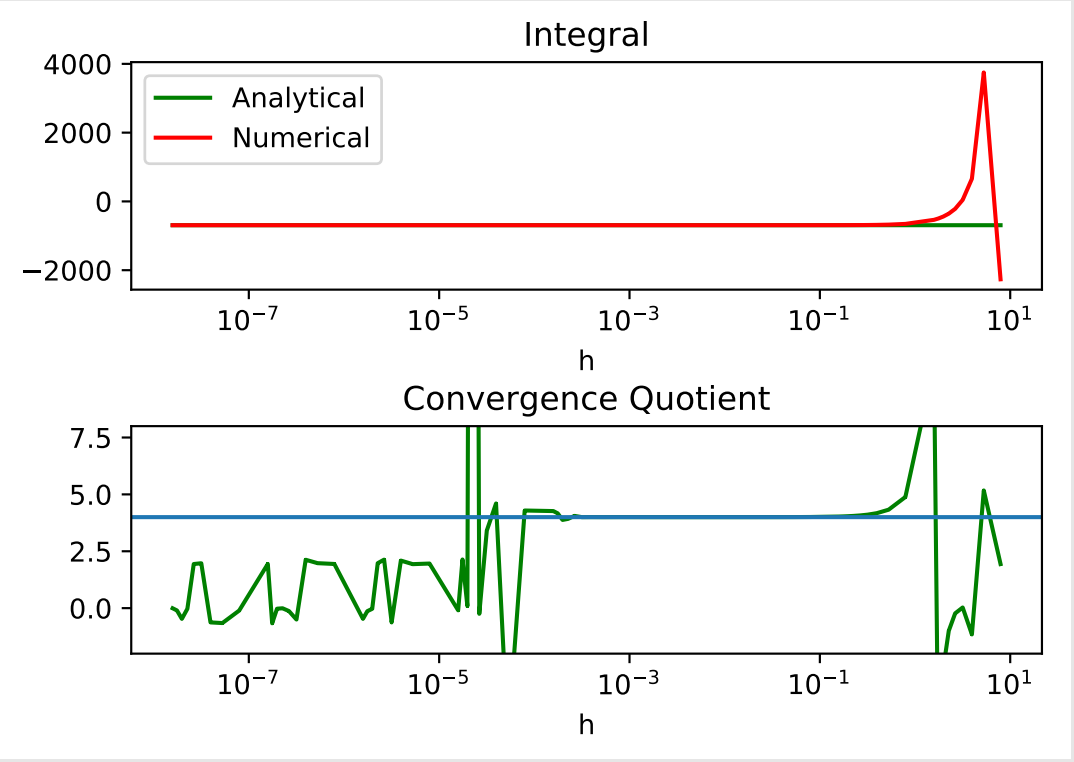
Agora vamos fazer o mesmo mas usando a regra dos trapézios.
Estamos interessados em observar o comportamento do *quociente de convergência* - *QC*, pelo que vamos calcular o integral usando o número de intervalos $n, n*2, n*4$ igual a (2, 4, 8), depois (4, 8, 16), depois (6, 12, 24), e assim sucessivamente, até (1e9, 2e9, 4e9). O valor do passo correspondente a $n*4$ varia entre aproximadamente 1e-8 e 3e1

Tenha cuidado ao ler os gráficos seguintes. As abcissas são logarítmicas, de base 10, e por vezes, as ordenadas também.

Os gráficos abaixo mostram o valor do integral, calculado numericamente, em comparação com

Exemplo: 44.3 Evolução do Quociente de Convergência para a Regra dos Trapézios (cont.)

o seu valor calculado analiticamente recorrendo à primitivação, e o valor do *quociente de convergência*. .



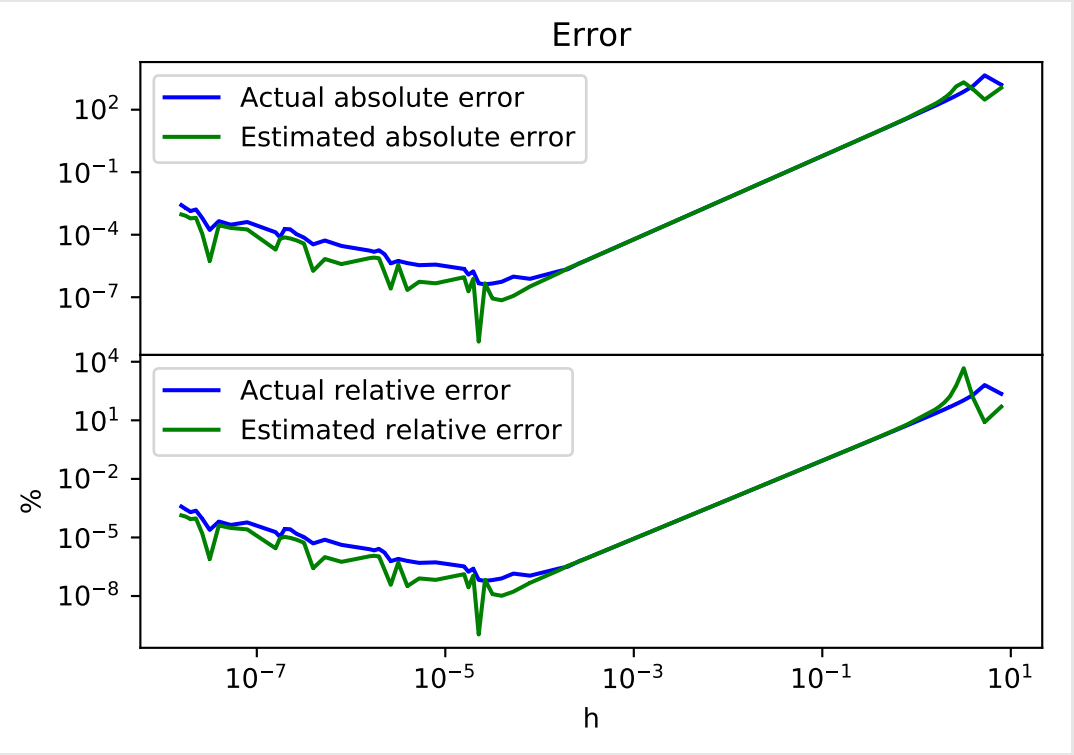
Como foi calculado valor analítico do integral, $I = -691.8871279250529$, foi possível calcular dois erros, um *observado*,

$$Erro\ Absoluto_{observado} = I - S''$$
$$Erro\ Relativo_{observado} = \frac{Erro\ Absoluto_{observado}}{I}$$

e um erro *estimado*, recorrendo ao *quociente de convergência*:

$$Erro\ Absoluto_{estimado} \approx \frac{S'' - S'}{3}$$
$$Erro\ Relativo_{estimado} = \frac{Erro\ Absoluto_{estimado}}{S''}$$

Exemplo: 44.3 Evolução do Quociente de Convergência para a Regra dos Trapézios (cont.)



Se olharmos para os gráficos dos erros absolutos e relativo, poderemos ver que:

- a região em que o QC se aproxima do valor teórico é bastante extensa, entre 1e-3 e 1e-1, perdendo qualquer significado para h abaixo de 1e-4 ou acima de 1e0;
- nessa região, o erro estimado é bastante parecido com o erro observado, sendo assim um bom estimador;
- o erro de truncatura, consequência de um número de iterações insuficientes, domina o lado direito do gráfico;
- o erro de arredondamento, consequência do arredondamento e da soma de quantidades muito pequenas, domina o lado esquerdo do gráfico, crescendo lentamente à medida que o h baixa para valores muito pequenos;
- o menor erro absoluto, da ordem de 1e-5, corresponde a um passo da ordem de 1e-4.

4.3 Regra de Simpson

Um defeito óbvio da regra dos trapézios é o de cometer um erro sistemático em intervalos em que a segunda derivada da integranda mantém sinal constante. Para o evitar e para aumentar, na generalidade, a precisão da aproximação, foi criado um algoritmo novo, a regra de Simpson que, em vez de substituir a curva pelas cordas definidas por cada par de pontos consecutivos, a substitui pelas parábolas definidas por cada trio de pontos consecutivos.

É fácil estabelecer que essa parábola é da forma

$$y = y_i + \frac{x - x_i}{h} . (y_{i+1} - y_i) + \frac{(x - x_i) . (x - x_{i+1})}{2h^2} . (y_{i+2} - 2y_{i+1} + y_i)$$

em que $h = x_{i+1} - x_i$ de onde resulta que

$$\begin{aligned}\int_{x_i}^{x_{i+2}} y.dx &= \\ &= \int_{x_i}^{x_{+2}} \left[y_i + \frac{x - x_i}{h} . (y_{i+1} - y_i) + \frac{(x - x_i) . (x - x_{i+1})}{2h^2} . (y_{i+2} - 2y_{i+1} + y_i) \right] .dx \\ &= 2h. \left[y_i + y_{i+1} - y_i + \frac{y_{i+2} - 2y_{i+1} + y_i}{6} \right] \\ &= \frac{h}{3} [y_i + 4y_{i+1} + y_{i+2}]\end{aligned}$$

de modo que a soma dá

$$\int_{x_0}^{x_{2n}} y.dx = \frac{h}{3} . [y_0 + 4y_1 + 2y_2 + 4y_3 + ... + 4y_{2n-3} + 2y_{2n-2} + 4y_{2n-1} + y_{2n}] \tag{4.6}$$

A fórmula mais adequada para o cálculo automático é, no entanto:

$$\int_{x_0}^{x_{2n}} y.dx = \frac{h}{3} . \left[y_0 + y_{2n} + 4 \times \sum_{\substack{i=1 \\ i=i+2}}^{2n-1} y_i + 2 \times \sum_{\substack{i=2 \\ i=i+2}}^{2n-2} \right] \tag{4.7}$$

Como ser constata da fórmula, o número de intervalos tem que ser par.

Exemplo: 44.4 Descubra as diferenças !

A fórmula para a *regra dos trapézios* em dois passos de amplitude h é:

$$S_t = \frac{h}{2} [y_0 + 2y_1 + y_2]$$

A fórmula para a *regra de Simpson* em um passo com dois intervalos de amplitude h é:

$$S_s = \frac{h}{3} [y_0 + 4y_1 + y_2]$$

Consegue descobrir as diferenças?

4.3.1 A "fórmula"de erro

Quanto à fórmula do erro, procedendo como anteriormente, obtemos:

$$\varepsilon_i = -\frac{h^5}{90} . f''''(\xi_i) \qquad (x_i \leq \xi_i \leq x_{i+2})$$

e, somando sobre todos os valores de i, obtemos

$$\begin{aligned}\varepsilon &= -\frac{n.h^5}{90} . f''''(\xi) \\ &= -\frac{(x_n - x_0) . h^4}{90} . f''''(\xi) \qquad (x_0 \leq \xi_i \leq x_{2n}) \\ &= -\frac{(x_n - x_0)^5}{90n^4} . f''''(\xi)\end{aligned} \tag{4.8}$$

de onde resulta que:

- para uma dada amplitude do intervalo, o erro varia com o inverso da quarta potência do número de intervalos ou, de outro modo, com a quarta potência do passo de integração, pelo que o método dos trapézios se diz de *quarta ordem*; dado o facto de a complexidade computacional da fórmula de Simpson ser da mesma ordem da dos trapézios e a melhoria significativa da precisão que ela representa, preferi-la-emos em todos os casos;
- para um dado número de passos de integração (e sob a hipótese muito forte de que o majorante da terceira derivada se mantém), o erro cresce com a quinta potência da amplitude do intervalo, pelo que, se multiplicarmos o intervalo por k e quisermos manter a precisão, devemos multiplicar o número de passos por $k^{5/4}$.

.

4.3.2 Controlo do erro

Naturalmente, no caso mais corrente de não dispormos de um majorante da quarta derivada, poderemos sempre, como anteriormente, recorrer ao critério

$$\frac{S' - S}{S'' - S'} \approx 16 \tag{4.9}$$

com $h = 2 * h' = 4 * h''$, e, quando ele for cumprido, poderemos estimar

$$S'' - S' \approx 15\varepsilon'' \tag{4.10}$$

Exemplo: 44.5 Regra de Simpson

Sejam os integrais do exemplo anterior:

n	80	40	20	10
$I_{\pi/2}$	1.00000	1.00000	1.00000	1.00000
I_{π}	2.00011	2.00001	2.00000	2.00000
$I_{3\pi/2}$	1.00028	1.00002	1.00000	1.00000
$I_{2\pi}$	0	-2.1×10^{-13}	-2.6×10^{-13}	-4.1×10^{-12}

Observa-se, em relação ao exemplo anterior, um notável aumento da precisão, salvo no último integral, em que dominam os erros de arredondamento sobre os de truncatura.

Exemplo: 44.6 Evolução do Quociente de Convergência para a Regra de Simpson

Voltando a usar o integral definido

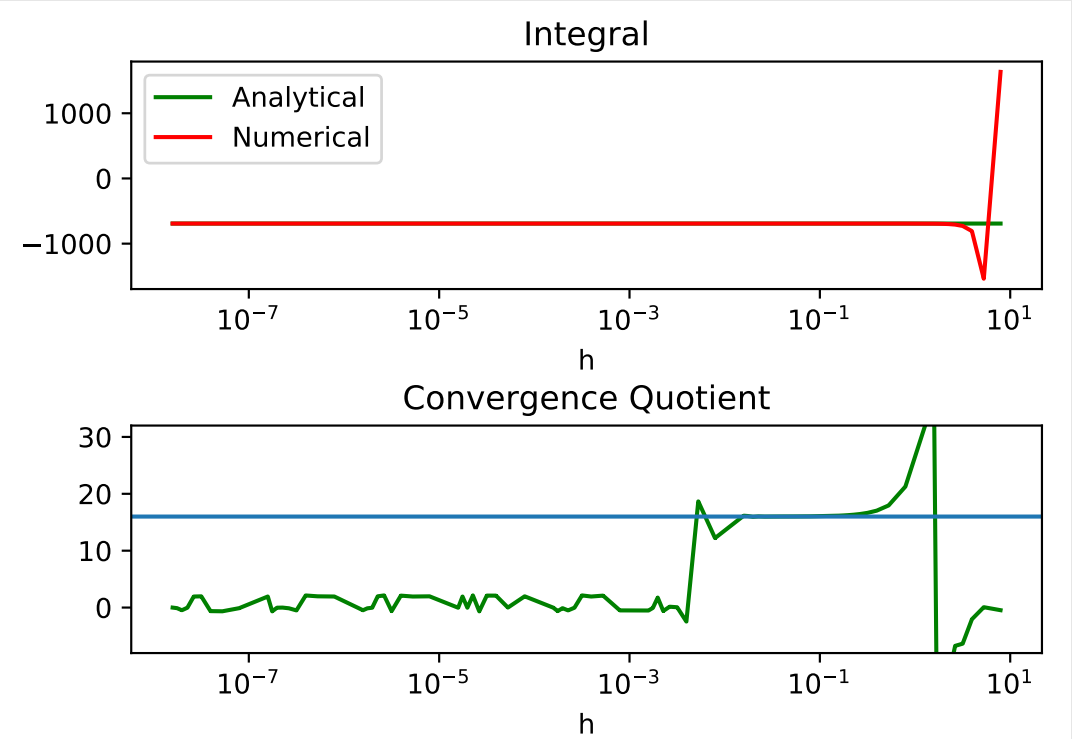
$$I = \int_0^{20*\pi+0.5} \frac{x\sin(x)}{5}\,\mathrm{d}x$$

cujo valor já foi calculado, sendo $I = -691.8871279250529$ vamos refazer o estudo do *quociente de convergência* mas usando agora a regra de Simpson.

Tenha cuidado ao ler os gráficos seguintes. As abcissas são logarítmicas, de base 10, e por vezes, as ordenadas também.

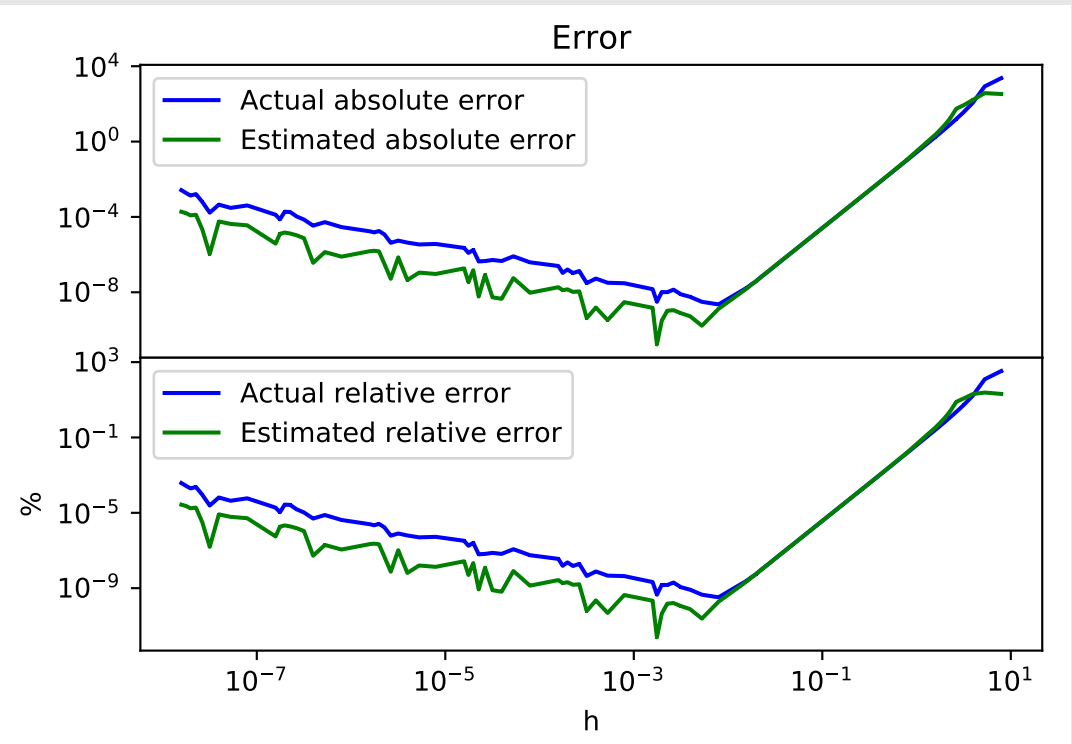
Os gráficos abaixo mostram o valor do integral, calculado numericamente, em comparação com o seu valor calculado analiticamente recorrendo à primitivação, e o valor do *quociente de convergência*. .

Exemplo: 44.6 Evolução do Quociente de Convergência para a Regra de Simpson (cont.)



Os erros *observado* e *estimado* podem ser estudados nos gráficos seguintes, sendo agora:

$$Erro\ Absoluto_{estimado} \approx \frac{S'' - S'}{15}$$



Exemplo: 44.6 Evolução do Quociente de Convergência para a Regra de Simpson (cont.)

- a região em que o QC se aproxima do valor teórico é menos extensa, entre 1e-2 e 1e0
- nessa região, o erro estimado é bastante parecido com o erro observado, sendo assim um bom estimador;
- o erro de truncatura domina o lado direito do gráfico;
- o erro de arredondamento domina o lado esquerdo do gráfico, crescendo lentamente à medida que o h baixa para valores muito pequenos;
- o menor erro absoluto, da ordem de 1e-8, é atingido para h da ordem de 1e-2
- comparando com a regra dos trapézios, um determinado grau de precisão é atingido muito mais cedo, com h maior, o que faz com que o erro de arredondamento também se manifeste mais cedo.

Exercício 4.1

1. O leitor calculará I_π com a precisão adequada, tanto pelo método dos trapézios como pelo método de Simpson, de modo a poder calcular e comparar os cocientes de convergência e os erros calculados.
2. Será o leitor capaz de estabelecer um cociente de convergência para um método de integração de ordem n e de estabelecer o correspondente estimador do erro?
3. Enumere as dificuldades numéricas do cálculo do cociente de convergência, e quais as suas consequências em termos de utilização? (observe com cuidado o seu comportamento no exemplo 4.2.2)

4.4 Integrais impróprios

Um integral

$$\int_a^b f(x).dx$$

diz-se impróprio se um dos limites (ou ambos) não é um número finito. Por razões de simplicidade, limitaremos a nossa discussão ao caso do integral

$$\int_a^\infty f(x).dx$$

visto que o caso

$$\int_{-\infty}^b f(x).dx = -\int_{+\infty}^b f(-x).dx = \int_b^{+\infty} f(-x).dx$$

se reduz a este sem qualquer dificuldade.

O *integral impróprio* diz-se *convergente* se existir I tal que

$$I = \lim_{b=\infty} \left(\int_a^b f(x).dx \right)$$

Se tal limite não existir ou for finito, então o integral diz-se *divergente* e não tem qualquer sentido, quer em termos analíticos quer numéricos. No que segue, suporemos, portanto, que o integral é convergente. Em cada caso particular, porém, o leitor terá que certificar-se previamente, por todos os meios ao seu alcance, da convergência do integral: com efeito, a abordagem por cálculo numérico sem tal certeza expõe-no a sérios riscos; para o efeito, a Análise Matemática fornece critérios e testes de convergência adequados; porém, em situações concretas, o próprio significado físico do integral e a natureza das aproximações e simplificações realizadas ao construir o modelo matemático do real poderão ajudar a determinar a validade de uma hipótese de convergência sem necessidade do recurso a testes analíticos.

O cálculo do integral impróprio, quando convergente, consegue-se, com uma precisão pre-especificada, se o representarmos na forma

$$\int_a^{\infty} f(x)\,.dx = \int_a^b f(x)\,.dx + \int_b^{\infty} f(x)\,.dx$$

Com efeito, o primeiro integral, qualquer que seja b finito, é um integral ordinário e o segundo tende para 0 quando b cresce; na prática, o valor de b pode ser estabelecido como aquele a partir do qual, dentro da precisão preestabelecida, já se não observa (persistentemente!) variação do resultado; porém, para isso, deve ir-se fazendo variar o passo de integração com a amplitude do intervalo de modo a manter constante o erro, tendo em consideração a ordem do método utilizado.

Exercício 4.2 Com base no que ficou dito, estabeleça uma estratégia para calcular um integral da forma

$$\int_{-\infty}^{\infty} f(x)\,.dx$$

supondo que se sabe que é convergente.

4.5 Integrais singulares

Quando o intervalo de integração é finito mas a integranda (x) tem um número finito de descontinuidades infinitas (singularidades) nesse intervalo, estamos em presença do que chamamos um *integral singular*. Dado que é sempre possível (embora nem sempre fácil) partir o intervalo de integração em partes tais que, em cada uma, exista apenas uma singularidade, podemos limitar o nosso estudo ao caso de uma singularidade única . Um artifício adequado para o cálculo de um valor aproximado do integral singular é o *método de Kantorovich* que consiste em extrair da integranda $f(x)$ uma certa função $g(x)$ que

- tenha uma singularidade do mesmo tipo e ordem da de $f(x)$;
- seja integrável analiticamente;
- a diferença $f(x) - g(x)$ seja contínua.

Sob estas hipóteses, teremos

$$\int_a^b f(x)\,.dx = \int_a^b g(x)\,.dx + \int_a^b (f(x) - g(x))\,.dx$$

No presente estudo consideraremos apenas o caso de

- integrais da forma

$$\int_a^b \frac{f(x)}{(x-x_0)^{\alpha}}.dx \qquad a < x_0 < b \qquad 0 < \alpha < 1$$

- sendo $f(x)$ contínuo em (a,b)
- e $f(x)$ dotada de derivadas contínuas em (a,b) até à ordem $n+1$.

Sob estas condições, podemos escrever $f(x)$ na forma de desenvolvimento em série de Taylor truncado:

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!} . (x-x_0)^k + \varepsilon(x)$$

com

$$\varepsilon(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} . (x-x_0)^{n+1} \qquad a < \xi < b$$

de onde resulta

$$\begin{aligned}
&\int_a^b \frac{f(x)}{(x-x_0)^\alpha} .dx = \\
&\quad = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!} . \int_a^b (x-x_0)^{k-\alpha} .dx + \int_a^b \frac{\varepsilon(x)}{(x-x_0)^\alpha} .dx \\
&\quad = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{(k+1-\alpha)} . \left[(b-x_0)^{k+1-\alpha} - (a-x_0)^{k+1-\alpha} \right] + \int_a^b \frac{f^{(n+1)}(\xi)}{(n+1)!} . (x-x_0)^{n+1-\alpha} .dx \\
&\quad = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{(k+1-\alpha)} . \left[(b-x_0)^{k+1-\alpha} - (a-x_0)^{k+1-\alpha} \right] + f^{(n+1)}(\xi) \int_a^b \frac{(x-x_0)^{n+1-\alpha}}{(n+1)!} .dx
\end{aligned}$$

Este último integral é, por força das hipóteses, um integral próprio calculável analiticamente, pelo que o problema está resolvido a menos da incerteza introduzida por ξ ; para diminuir esta incerteza, podemos reduzir $b-a$ (continuando a impor $a < \xi < b$) a um valor tão pequeno quanto necessário para que a variação de $f^{n+1}(\xi)$ seja insignificante face à precisão exigida.

Exercício 4.3 Investigue as eventuais dificuldades da aplicação do método de Kantorovich ao caso de a singularidade ocorrer precisamente em um dos extremos do intervalo de integração. Aplique os resultados a que chegar ao cálculo de

$$\int_0^{0.5} x^{-0.5} . (1-x)^{-0.5} .dx$$

4.6 Cubatura

As fórmulas da cubatura aplicam-se ao cálculo de *integrais duplos*

$$\int_a^A dx . \int_b^B f(x,y) .dy$$

Uma das mais simples e úteis resulta de aplicar a *regra de Simpson* nos seguintes termos: sendo o domínio de integração *aABb* um rectângulo de lados paralelos aos eixos, partamo-lo em quatro partes

iguais pelas medianas dos lados e calculemos os integral interior pela regra de Simpson:

$$\begin{aligned}
&\int_a^A dx.\int_b^B f(x,y).dy = \\
&\quad = \int_a^A \frac{h_y}{3}.[f(x,y_0)+4f(x,y_1)+f(x,y_2)].dx \\
&\quad = \frac{h_y}{3}.\left[\int_a^A f(x,y_0).dx+4.\int_a^A f(x,y_1).dx+\int_a^A f(x,y_2).dx\right]
\end{aligned}$$

Aplicando de novo a regra de Simpson a estes integrais, obtemos

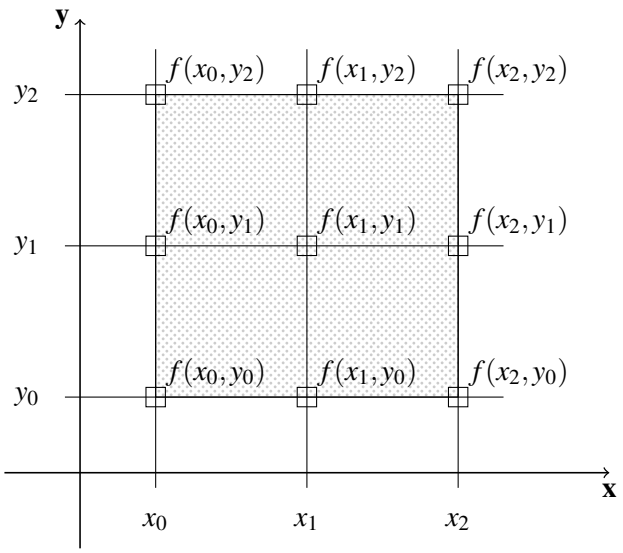


Figura 4.1: Pontos usados na Regra de Simpson

$$\begin{aligned}
&\int_a^A dx.\int_b^B f(x,y).dy = \\
&\quad = \frac{h_x.h_y}{9}.\left\{\begin{array}{l}[f(x_0,y_0)+4f(x_1,y_0)+f(x_2,y_0)]+\\ +4.[f(x_0,y_1)+4f(x_1,y_1)+f(x_2,y_1)]+\\ +[f(x_0,y_2)+4f(x_1,y_2)+f(x_2,y_2)]\end{array}\right\} \\
&\quad = \frac{h_x.h_y}{9}.\left\{\begin{array}{l}[f(x_0,y_0)+4f(x_0,y_2)+f(x_2,y_0)+f(x_2,y_2)]+\\ +4.[f(x_1,y_0)+4f(x_0,y_1)+f(x_2,y_1)+f(x_1,y_2)]+16f(x_1,y_1)\end{array}\right\}
\end{aligned}$$

com

$$h_x = \frac{A-a}{2} \qquad h_y = \frac{B-b}{2}$$

A fórmula de Simpson da cubatura é, portanto, da forma:

$$\iint f(x,y).dx.dy = \frac{h_x.h_y}{9}.[\Sigma_0+4\Sigma_1+16\Sigma_2]$$

em que:

$\Sigma_0 =$ soma dos valores de f nos vértices da malha
$\Sigma_1 =$ soma dos valores de f nos pontos médios dos lados da malha
$\Sigma_2 =$ valor de f no centro da malha

Se o rectângulo de integração é grande, a divisão a meio não chega, de modo que o domínio é divido em $m \times n$ rectângulos, a cada um dos quais se aplica a fórmula anterior. Se o domínio de integração não é rectangular, considera-se o menor rectângulo que o contém e aplica-se o mesmo método fazendo $f = 0$ fora do domínio.

Exercício 4.4

1. Desenvolva uma fórmula dos Trapézios para o integral duplo.
2. Desenvolva uma fórmula de Simpson para o integral triplo.
3. Verifique, usando uma estratégia de cubatura de trapézios, que o volume de um cubo de lado $l = 2$ é realmente 8.

   Lembre-se que o volume de um cubo pode ser descrito pelo integral duplo

   $$\int_0^l \int_0^l l \, \mathrm{d}y \, \mathrm{d}x$$

4. Verifique se um cilindro de revolução de altura $h = 1$ e raio da base $r = 1$ tem como volume π. Talvez seja interessante usar coordenadas cilíndricas!
5. Para calcular o volume subtendido por um parabolóide de revolução
   a) usaria uma estratégia de trapézios ou de Simpson
   b) coordenadas retangulares, cilíndricas ou esféricas ?
   c) se este fosse descrito pela expressão

      $$z = x^2 + y^2$$

      qual seria o volume abaixo de $z = 1$?

5 Integração de equações diferenciais ordinárias

Contents

5.1 O significado de uma solução: Método de Euler

As equações diferenciais são muito frequentes nos problemas de Ciência e Engenharia, desde que, com a invenção de Newton e Leibniz, nos habituámos a exprimir as propriedades dos objectos e sistemas, fenómenos e processos em termos de derivadas, sobretudo

- espaciais (que, dão, no caso geral, origem a equações em derivadas parciais),
- temporais (que dão origem a equações diferenciais ordinárias).

Destas últimas, as equações mais correntes são as equações do movimento de sistemas, habitualmente de 2^a ordem, isto é, contendo, além da função desconhecida, as suas derivadas de 1^a e 2^a ordem.

Nos cursos elementares de análise matemática aborda-se o problema genérico da resolução de equações diferenciais e aprendem-se as técnicas de resolução sob forma cerrada de certas classes particulares dessas equações; infelizmente, porém, a larga maioria das equações diferenciais que se encontram na prática não pode ser resolvida por meios analíticos, o que torna imprescindível o recurso aos métodos numéricos que, tendo embora o defeito de serem apenas aproximados, são implementáveis em computadores digitais, onde têm soluções-padrão relativamente rápidas e eficientes. Na realidade, boa parte da motivação dos técnicos e cientistas para construírem os primeiros computadores digitais automáticos proveio precisamente da necessidade de resolver rápida e precisamente problemas de cálculo balístico; hoje em dia, o cálculo automático é usado intensivamente para resolver as equações do movimento dos mísseis balísticos e dos satélites artificiais, da teoria dos circuitos eléctricos, da deformação de pilares, vigas e cascas, da estabilidade de aeronaves e embarcações, etc.

Dada uma equação diferencial ordinária de primeira ordem:

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

o que entendemos por uma solução?
Em termos genéricos e sem preocupações de precisão, entendemos por solução de uma equação diferencial ordinária de primeira ordem uma curva $g(x,y) = 0$ tal que, se calcularmos a derivada de $g(x,y)$ em ordem a x

$$g'_x(x,y) = \frac{dg(x,y)}{dx}$$

em um ponto (x,y), o valor calculado coincide com o valor especificado pela equação diferencial para esse ponto.

Exemplo: 55.1 Integral como solução

O integral indefinido

$$y(x) = \int_a^x f(t).dt$$

é a solução da equação diferencial ordinária de primeira ordem $y'(x) = f(x)$ de um tipo particular em que $f(x,y)$ não depende de y.

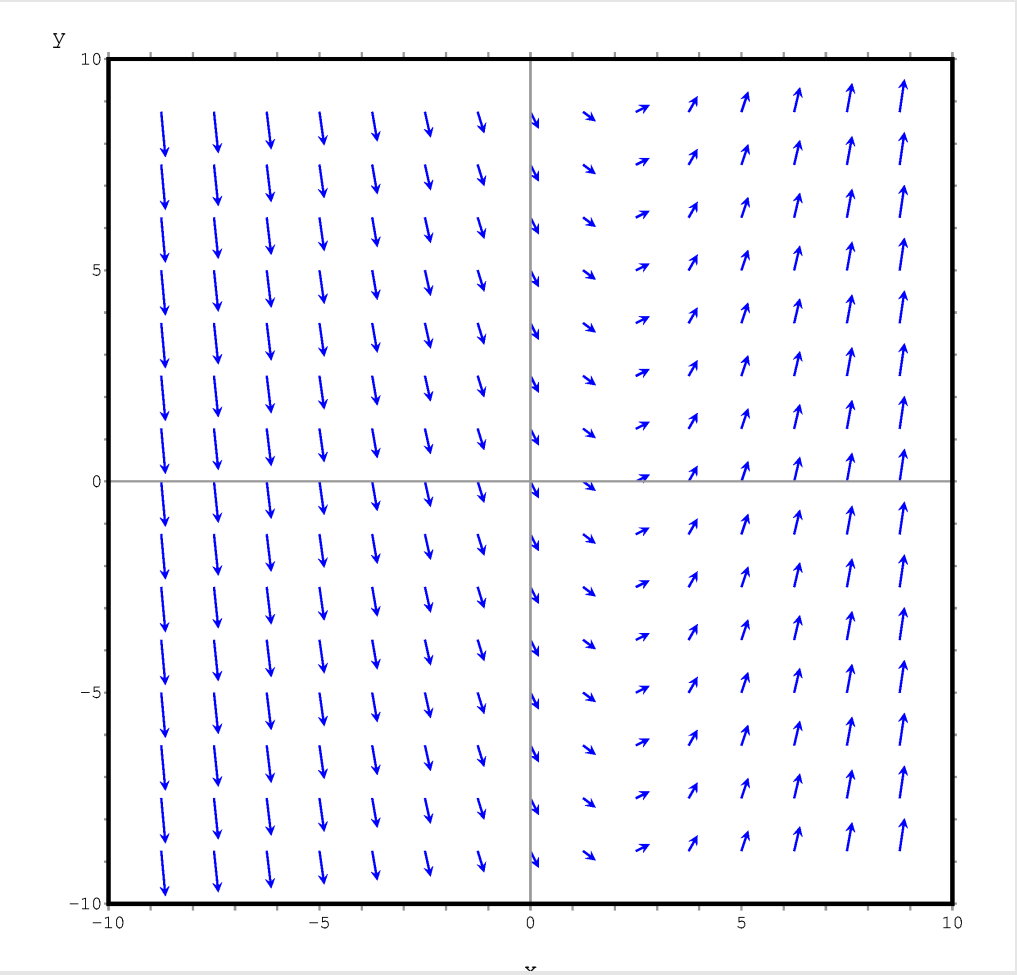
Esta simples interpretação sugere desde logo um método tosco para resolver graficamente a equação diferencial: em cada ponto (x,y) de uma rede rectangular de pontos, calculamos a derivada tal como é especificada pela equação e, em torno do ponto, traçamos um pequeno segmento cuja inclinação sobre o eixo dos xx é igual ao valor dessa derivada. Obtemos assim aquilo que chamamos um *campo de direcções*.

Exemplo: 55.2 Campo de direcções

Seja a equação diferencial $y' = x - 2$ e calculemos o campo de direcções usando o comando plotdf() do **Maxima**:

```
(% i1)
load(plotdf)$
(% i2)  plotdf(x-2)$
```

Exemplo: 55.2 Campo de direcções (cont.)



Notaremos, antes de mais, que, pelo facto de se tratar de uma equação particular em que $f(x,y)$ não depende de y, todas as direcções são independentes de y. O mesmo não acontece com a equação

$$y' = \left(\sqrt[3]{x} + \sqrt[3]{y}\right)^3$$

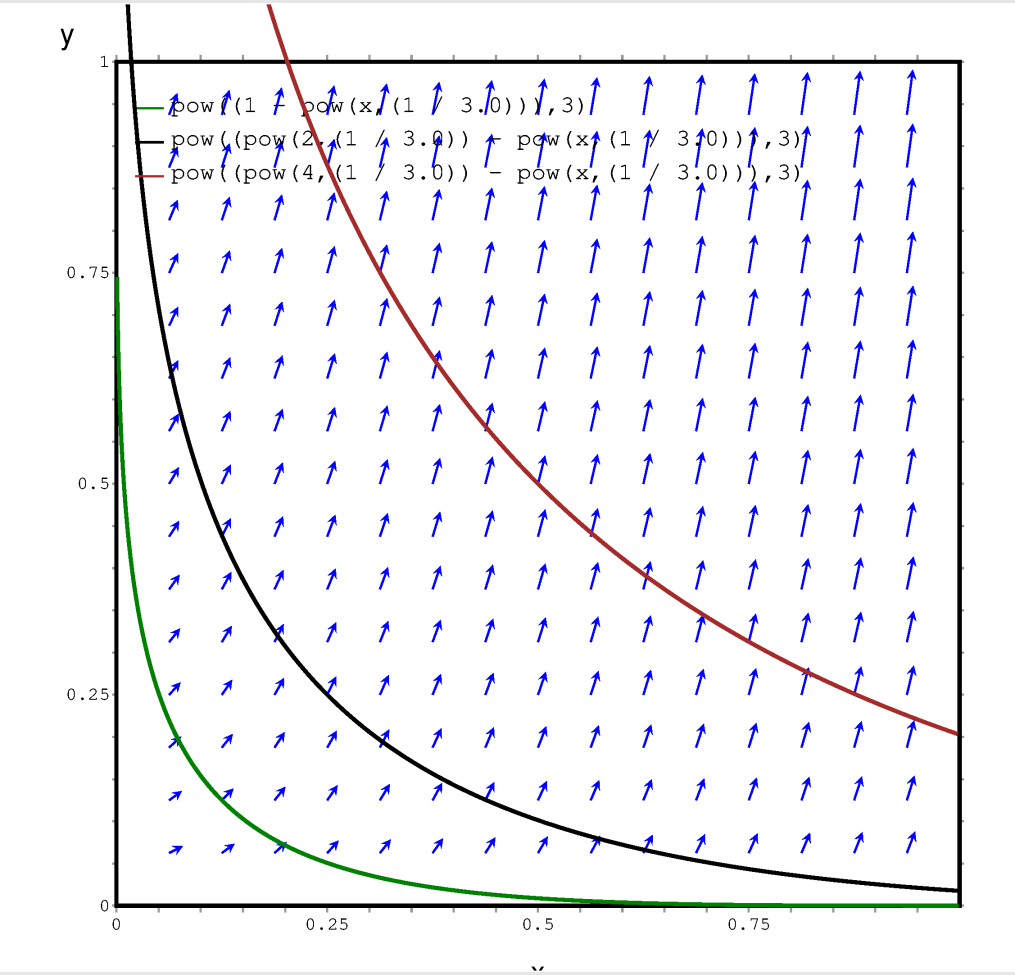
```
(% i1)  load(plotdf)$
(% i2)  f:(x^(1/3 )+ y^(1/3))^3;
```

$$\left(y^{\frac{1}{3}} + x^{\frac{1}{3}}\right)^3 \tag{f}$$

```
(% i3)  plotdf(f,[x,0,1],[y,0,1],[xfun,"(1-x^(1/3.0))^3;(2^(1/3.0 )- x^(1/3.0))^3;
        (4^(1/3.0 )- x^(1/3.0))^3"])$
```

cujo campo de direções é:

Exemplo: 55.2 Campo de direcções (cont.)



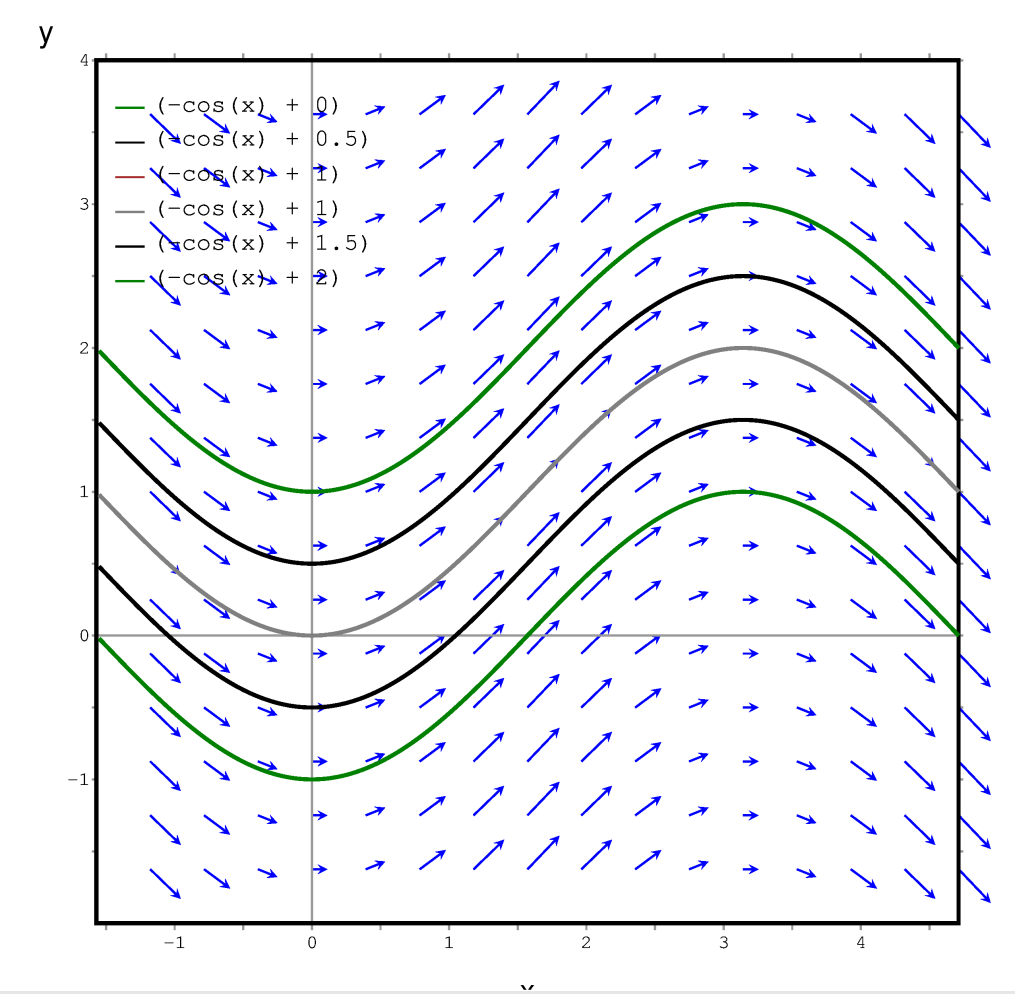
e sobre o qual marcámos as isoclinas (lugares geométricos dos pontos de igual inclinação) para $y' = 1, 2, 4$ cuja observação facilita a interpretação e o uso do campo de direcções (assim, por exemplo, os máximos e os mínimos de uma solução encontram-se necessariamente sobre a isoclina zero).

Exercício 5.1 Como se identificam, sobre um campo de direcções calibrado com isoclinas, os pontos de inflexão das soluções?

Exemplo: 55.3 Soluções particulares

Para o ajudar, mostra-se em seguida o campo de direcções da equação diferencial $y' = sin(x)$ ao qual sobrepusemos algumas soluções:

Exemplo: 55.3 Soluções particulares (cont.)



O problema está então reduzido a traçar uma curva contínua nesse campo, tal que, em cada ponto, seja tangente aos pequenos segmentos traçados.

Resulta então evidente que por cada ponto passa apenas uma tal curva e que por diferentes pontos passam diferentes curvas. Assim, a solução da equação diferencial não é uma única curva (uma única função $y = g(x)$), mas toda uma família de curvas (o que equivale, em termos analíticos, a admitir a existência de uma constante de integração na solução geral). Para especificar uma das curvas da família basta especificar um ponto por onde ela passa.

Se, por qualquer razão (e como é frequente nas aplicações) estivermos apenas interessados em uma curva particular, é evidente que podemos limitar o traçado do campo de direcções às imediações dos locais por onde irá passar. Assim, começaremos em um ponto inicial (x_0, y_0) e nele calculamos a inclinação $y'_0 = f(x_0, y_0)$ e avançamos um pouco a partir desse ponto ao longo da direcção assim definida; podemos então considerar o novo ponto atingido como um novo ponto de partida e continuar a proceder deste modo. Após um número suficiente destes pequenos passos, teremos a solução no intervalo finito que nos interessa.

Exemplo: 55.4 Solução passo a passo

Retomemos a equação diferencial $y' = x - 2$.
Para a mesma equação, consideremos a solução particular $\frac{x^2}{2} - 2x + 3$ que corresponde à condição

Exemplo: 55.4 Solução passo a passo (cont.)

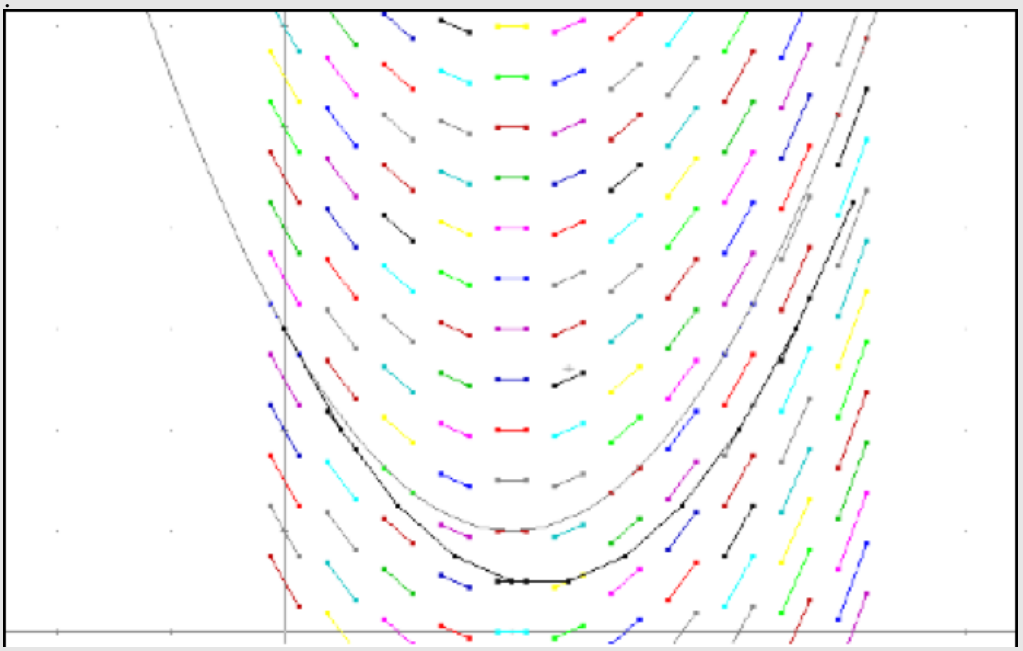
inicial

$$\begin{cases} x = 0 \\ y = 3 \end{cases}$$

e sobreponhamos o seu gráfico ao campo de direcções já determinado:
Façamos agora a reconstituição ponto a ponto da solução, tal como sugerido anteriorente e tracemos o resultado deste processo sobre o mesmo gráfico , tomando um passo

$$h = \Delta x = 0.5$$

.

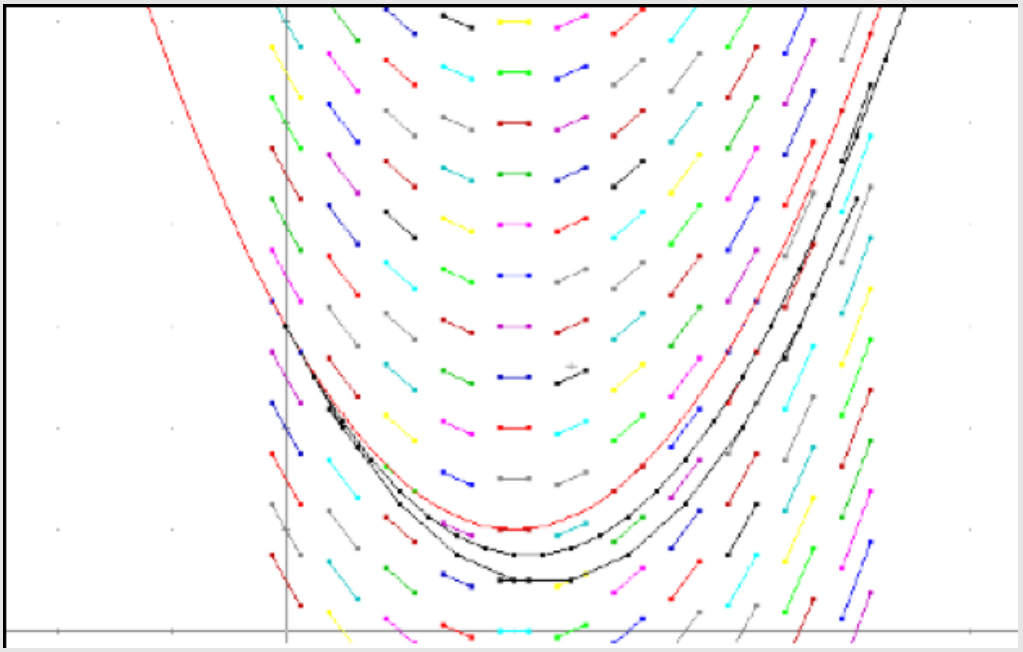


Deve agora ser perfeitamente claro que, quanto menores forem os *pequenos passos*, tanto menor será o erro cometido por efeito da truncatura e tanto maior será o esforço despendido e o risco de acumulação de erros de arredondamento.

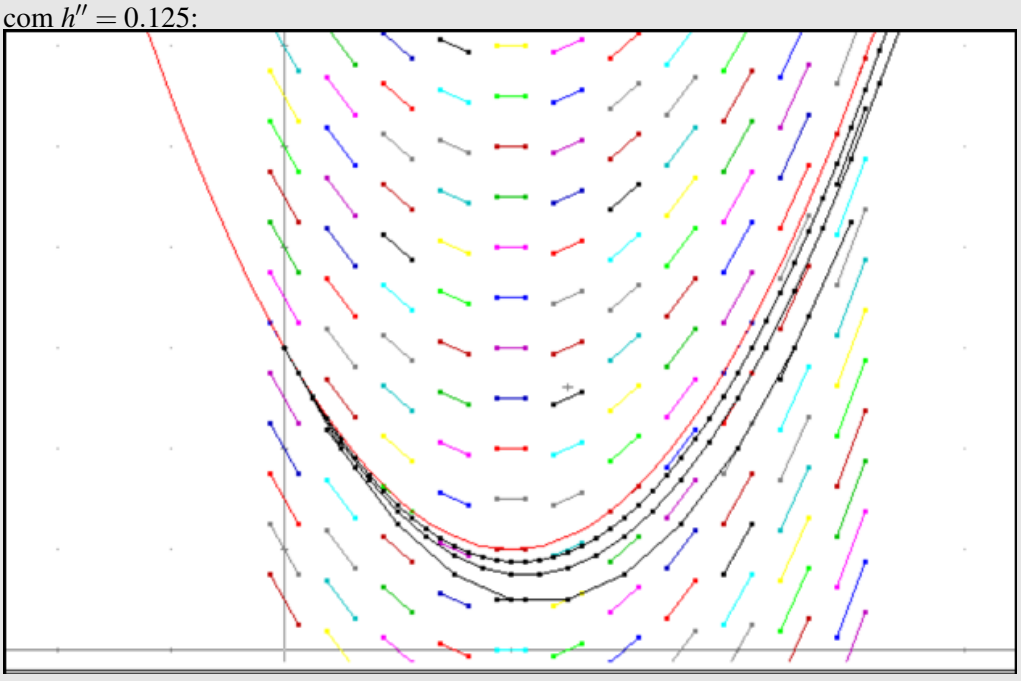
Exemplo: 55.5 Solução com passo menor

Tomemos ainda o caso anterior e executemos o processo de integração com um passo $h' = 0.25$:

Exemplo: 55.5 Solução com passo menor (cont.)



e verificaremos de imediato a melhor qualidade da aproximação. Vejamos ainda o que se passa com $h'' = 0.125$:



É também óbvio que não precisamos de proceder por meios gráficos: poderemos, simplesmente, construir uma tabela numérica com as coordenadas dos pontos sucessivos, procedendo do modo que se segue.

5.2 Método de Euler

Este método elementar de integração numérica de equações diferenciais é conhecido por *método de Euler* e corresponde, obviamente, em termos analíticos, a utilizar a *fórmula dos acréscimos finitos*, isto é, um desenvolvimento em série de Taylor limitado à primeira ordem.

O algoritmo para o *método de Euler* é então:

1. Calcular o incremento de y usando y'_n que é a inclinação da curva no ponto (x_n, y_n), e Δx_n, o incremento de x no intervalo n:

$$\Delta y_n = f(x_n, y_n)\, \Delta x_n$$

2. Calcular o ponto seguinte:

$$\begin{cases} y_{n+1} & \leftarrow & y_n + \Delta y_n \\ x_{n+1} & \leftarrow & x_n + \Delta x_n \end{cases}$$

e repetir esses passos até cobrir por completo o intervalo que nos interessa.

Trata-se, portanto, de um *método de primeira ordem*, no sentido de que o erro cometido no cálculo de y é proporcional a h, se este for suficientemente pequeno.

Poderemos também aqui aplicar o critério do *quociente de convergência*, aplicado agora intervalo a intervalo com $h = 2 * h' = 4 * h''$:

$$\frac{S' - S}{S'' - S'} \approx 2 \tag{5.1}$$

, e, quando ele for cumprido, poderemos estimar o erro absoluto

$$S'' - S' \approx \varepsilon'' \tag{5.2}$$

Exemplo: 55.6 Euler e Q.C.

Consideremos ainda o exemplo da equação anterior $y' = x - 2$ e tabelemos os resultados da integração pelo método de Euler para $h = 1$, $h' = 0.5$, $h'' = 0.25$ e calculemos os cocientes de convergência e os erros respectivos:

x	y exacto	y (h=0.25)	erro (h=0.25)	y (h=0.5)	y h=1.0	$\frac{(S'-S)}{(S''-S')}$	erro calc h=0.25
0	3	3		3	3		
1	1.5	1.375	0.125	1.25	1	2	0.125
2	1	0.75	0.25	0.5	0	2	0.25
3	1.5	1.125	0.375	0.75	0	2	0.375
4	3	2.5	0.5	2	1	2	0.5
5	5.5	4.875	0.625	4.25	3	2	0.625

Verificamos que:

- o critério mínimo de convergência é satisfeito (cociente = 2) e os erros calculados coincidem com os observados:
- os erros são ainda muito grandes, pelo que há todo o interesse em diminuir o passo de integração.

Calculemos então o integral para $h = .5$, $h' = 0.25$, h" $= 0.125$, juntamente com os cocientes de convergência e os erros respectivos:

Exemplo: 55.6 Euler e Q.C. (cont.)

x	y exacto	y (h=0.25)	erro (h=0.25)	y (h=0.5)	y h=1.0	$\frac{(S'-S)}{(S''-S')}$	erro calc h=0.25
0	3	3	0	3	3		
0.5	2.125	2.09375	0.03125	2.0625	2	2	0.03125
1	1.5	1.4375	0.0625	1.375	1.25	2	0.0625
1.5	1.125	1.03125	0.09375	0.9375	0.75	2	0.09375
2	1	0.875	0.125	0.75	0.5	2	0.125
2.5	1.125	0.96875	0.15625	0.8125	0.5	2	0.15625
3	1.5	1.3125	0.1875	1.125	0.75	2	0.1875
3.5	2.125	1.90625	0.21875	1.6875	1.25	2	0.21875
4	3	2.75	0.25	2.5	2	2	0.25
4.5	4.125	3.84375	0.28125	3.5625	3	2	0.28125
5	5.5	5.1895	0.3105	4.875	4.25	1.99	0.3145

Verificamos que:

- os erros se reduziram a metade, como seria de esperar dado que o método é de primeira ordem e o passo se reduziu a metade;
- o erro máximo = 0.3145 (que, neste caso, corresponde ao do último resultado) é ainda muito elevado; Para termos um erro máximo da ordem de uma centésima teríamos que fazer

$$h'' = \frac{0.01}{0.3145} \times 0.125 \approx 0.003$$

isto é, que dar cerca de 42 vezes mais passos.

Exemplo: 55.7 Integral de $\operatorname{sen} x$

Calculemos pelo método de Euler o integral indefinido

$$x = \int_0^x \operatorname{sen} x \, \mathrm{d}x$$

até $x = 2$, planeando o trabalho de modo a poder comparar os resultados com os do cálculo do integral definido do capítulo anterior. Certifiquemo-nos de que nos encontramos dentro do critério mínimo de convergência, calculemos os erros, identifiquemos os pontos em que o erro é máximo e escolhamos um passo que garanta uma precisão mínima de uma centésima. Comparemos os resultados com os do cálculo do integral definido do capítulo anterior.
Os quadros seguintes dão alguns resultados preliminares obtidos, como anteriormente, com o comando EULER de DERIVE e com precisão de 6 algarismos significativos.
Comecemos por $h = 2\pi/5$, $h' = 2\pi/10$, $h" = 2\pi/20$:

x	y observado	y $2\pi/20$	erro $2\pi/20$	y $2\pi/10$	y $2\pi/5$	QC	erro estimado	delta (erro) %
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000			
1.25664	0.69098	0.53589	-0.15509	0.36932	0.00000	2.21713	0.16657	7.40
2.51327	1.80902	1.70178	-0.10724	1.56444	1.19513	2.68902	0.13734	+28.07
3.76991	1.80902	1.88644	+0.07742	1.93376	1.93376	0.00000	-0.04732	-38.88
5.02655	0.69098	0.83468	+0.14370	0.96688	1.19513	1.72654	-0.13220	-8.00
6.28319	0.00000	0.00000	0.00000	0.00000	0.00000		0.00000	

É perfeitamente claro que $h" = 2\pi/20$ é um passo insuficiente, porque o cociente de convergência se afasta ainda muito do valor teórico (2); em consequência, o erro calculado tem muito pouca

Exemplo: 55.7 Integral de sen$x$ **(cont.)**

precisão.
Reduzamos o passo a metade:

x	y observado	y $2\pi/40$	erro $2\pi/40$	y $2\pi/20$	y $2\pi/10$	QC	erro estimado	delta (erro) %
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000			
0.62832	0.19098	0.14443	-0.04656	0.09708	0.00000	2.05051	0.04734	+1.69
1.25664	0.69098	0.61487	-0.07612	0.53589	0.36932	2.10920	0.07898	+3.75
1.88496	1.30902	1.23162	-0.07740	1.14884	0.96688	2.19809	0.08278	+6.96
2.51327	1.80902	1.75913	-0.04989	1.70178	1.56444	2.39477	0.05735	+14.96
3.14159	2.00000	1.99588	-0.00412	1.98352	1.93376	4.02589	0.01236	+200.00
3.76991	1.80902	1.85146	+0.04244	1.88644	1.93376	1.35277	-0.03498	-17.58
4.39823	1.30902	1.38102	+0.07200	1.44762	1.56444	1.75405	-0.06660	-7.50
5.02655	0.69098	0.76426	+0.07327	0.83468	0.96688	1.87717	-0.07043	-3.89
5.65487	0.19098	0.23675	+0.04577	0.28174	0.36932	1.94687	-0.04498	-1.72
6.28319	0.00000	0.00000	0.00000	0.00000	0.00000	0.31260	0.00000	

A situação melhorou claramente, mas a convergência é ainda defeituosa, sobretudo na região central.
Continuemos, portanto:

x	y observado	y $2\pi/80$	erro $2\pi/80$	y $2\pi/40$	y $2\pi/20$	cociente	erro estimado	delta (erro) %
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000			
0.31416	0.04894	0.03678	-0.01216	0.02457	0.00000	2.01240	0.01221	+0.41
0.62832	0.19098	0.16780	-0.02318	0.14443	0.09708	2.02526	0.02338	+0.85
0.94248	0.41221	0.38023	-0.03198	0.34783	0.28174	2.03938	0.03241	+1.32
1.25664	0.69098	0.65328	-0.03770	0.61487	0.53589	2.05589	0.03841	+1.88
1.57080	1.00000	0.96022	-0.03978	0.91940	0.83468	2.07583	0.04081	+2.59
1.88496	1.30902	1.27099	-0.03803	1.23162	1.14884	2.10262	0.03937	+3.53
2.19911	1.58779	1.55519	-0.03260	1.52097	1.44762	2.14348	0.03422	+4.98
2.51327	1.80902	1.78500	-0.02402	1.75913	1.70178	2.21685	0.02587	+7.72
2.82743	1.95106	1.93791	-0.01315	1.92277	1.88644	2.39960	0.01514	+15.16
3.14159	2.00000	1.99897	-0.00103	1.99588	1.98352	4.00000	0.00309	+200.00
3.45575	1.95106	1.96218	+0.01112	1.97131	1.98352	1.33735	-0.00913	-17.92
3.76991	1.80902	1.83116	+0.02214	1.85146	1.88644	1.72315	-0.02030	-8.32
4.08407	1.58779	1.61873	+0.03094	1.64805	1.70178	1.83254	-0.02932	-5.25
4.39823	1.30902	1.34569	+0.03667	1.38102	1.44762	1.88508	-0.03533	-3.66
4.71239	1.00000	1.03875	+0.03875	1.07648	1.14884	1.91784	-0.03773	-2.63
5.02655	0.69098	0.72795	+0.03697	0.76426	0.83468	1.93971	-0.03631	-1.79
5.34071	0.41221	0.44377	+0.03156	0.47491	0.53590	1.95902	-0.03113	-1.34
5.65487	0.19098	0.21397	+0.02298	0.23675	0.28174	1.97411	-0.02279	-0.86
5.96903	0.04894	0.06105	+0.01211	0.07311	0.09708	1.98743	-0.01206	-0.41
6.28319	0.00000	0.00000	0.00000	0.00000	0.00000	-0.33891	0.00000	

Os resultados aproximam-se agora de uma precisão satisfatória (erro máximo de 4 centésimas) o que permite prever que com um passo quatro vezes mais pequeno estejamos nos valores pretendidos; porém, aparece um problema novo: o cociente de convergência afasta-se ainda muito do valor teórico na região central, nas vizinhanças de um ponto de inflexão da integranda.
Aqui a truncatura é mais grave, ou são já problemas de arredondamento?
Procuraremos esclarecer este problema utilizando para a integração a folha de cálculo EXCEL, com a sua facilidade de impor precisão arbitrária de cálculo:

Exemplo: 55.7 Integral de sen$x$ (cont.)

	12 significativos			6 significativos			3 significativos		
x	$2\pi/80$	$2\pi/40$	$2\pi/20$	$2\pi/80$	$2\pi/40$	$2\pi/20$	$2\pi/80$	$2\pi/40$	$2\pi/20$
3.141592653589E-1	1.987			1.987			1.986		
6.283185307177E-1	1.974	1.956		1.974	1.947		1.974	1.944	
9.424777960765E-1	1.959			1.959			1.965		
1.256637061436E+0	1.941	1.886	1.727	1.941	1.877	1.727	1.942	1.877	1.731
1.570796326796E+0	1.918			1.918			1.973		
1.884955592156E+0	1.885	1.746		1.885	1.754		1.971	1.746	
2.199114857516E+0	1.832				1.832			1.929	
2.513274122876E+0	1.724	1.343	0.000	1.724	1.353	0.000	1.944	1.343	-0.064
2.827433388236E+0	1.338			1.338			1.500		
3.141592653596E+0	4.006	4.167		4.008	4.024		4.333	3.923	
3.455751918956E+0	2.399			2.399			2.118		
3.769911184316E+0	2.216	2.404	2.689	2.217	2.395	2.689	2.148	2.379	2.703
4.084070449676E+0	2.144			2.144			2.028		
4.398229715036E+0	2.103	2.193	2.103	2.198	2.075		2.184		
4.712388980396E+0	2.076			2.076			2.027		
5.026548245756E+0	2.056	2.114	2.217	2.056	2.109	2.217	1.997	2.107	2.227
5.340707511116E+0	2.039			2.040			1.969		
5.654866776476E+0	2.025	2.064		2.026	2.050		1.940	2.044	
5.969026041836E+0	2.012			2.013			1.852		
6.283185307196E+0	5.329		3.825	-0.685	-1.530	1.345	5.528	2.073	1.687

Torna-se, portanto, evidente que o problema não é de arredondamento, mas sim de truncatura.

Resulta do que ficou dito que o método de Euler, embora extremamente simples de implementar e de controlar conceptualmente, tem muito pequena precisão (é de primeira ordem) e dá, portanto, origem a cálculos extremamente longos.

Por outro lado, deve também ter ficado claro que a pequena precisão resulta da acumulação de atrasos nos intervalos em que o sinal da curvatura se mantém; a razão desta característica desfavorável prende-se claramente com o facto de usarmos, ao longo de cada passo, um valor da derivada que só vale para o seu extremo inicial.

Este resultado é consequência da substituição ingénua da diferencial, dx, por um acréscimo, Δx , não tão pequeno como isso.

A solução é-nos sugerida pelo *teorema de Lagrange*, que nos diz que

$$\Delta y = f'_{\text{média}} \Delta x$$

A descoberta da *derivada média* é então o objetivo de qualquer melhoramento .

Consideraremos duas heurísticas:

- a ideia que uma boa estima do futuro próximo é o passado recente, condicionado pelo *bom comportamento* [1], e que resulta nos métodos chamados *preditores - corretores*, que apresentaremos como um melhoramento ao método de Euler;
- por outro lado o meio está a virtude, isto é, uma boa estima da *derivada média* pode ser a *média das derivadas*, eventualmente ponderada, calculadas ao longo do intervalo; daqui resultam os métodos da família *Runge-Kutta*.

[1] Como anedota, o canal televisivo dos EUA com melhores resultados na previsão meteorológica previa para o dia seguinte o tempo que estava no próprio dia. Ajudava que estava no deserto do Nevada ...

Exercício 5.2 Visto que o passo não é ainda suficiente para o que se pretendia, o leitor prosseguirá o exercício, realizando a integração em EXCEL para $h'' = 2\pi/320$, certificando-se de que se encontra dentro do critério mínimo de convergência, calculando os erros, identificando os pontos em que o erro é máximo e escolhendo um passo que garanta uma precisão mínima de uma centésima. Se os altos valores do cociente de convergência persistirem, encarará a hipótese de reduzir (ou aumentar?) o passo apenas nas vizinhanças dos pontos mais críticos, acedendo assim ao conceito de *método de integração de passo variável*. Este ponto é particularmente importante porque:

- pode suceder, porque estamos em um ponto em que a primeira derivada da função é muito pequena, que um passo pequeno resulte em um acréscimo tão pequeno face ao valor da função que haja perda significativa da precisão na adição;

- fica assim mostrado que, em termos de cálculo numérico, os passos pequenos nem sempre são desejáveis. Comparará em seguida os resultados com os do cálculo do integral definido do capítulo anterior.

5.3 Um melhoramento do Método de Euler

Com base no diagnóstico apresentado anteriormente, é possível construir um *método de Euler modificado* que corresponde a uma estratégia de cálculo melhor e mais económica: suponhamos que temos o par de pontos (x_{n-1}, y_{n-1}) e (x_n, y_n) ; como devemos calcular (x_{n+1}, y_{n+1}) ? Uma maneira simples consiste em

i) calcular o declive da tangente no ponto índice n e usar esse valor para, a partir do ponto $n-1$ calcular um ponto *previsto* em $n+1$, associando a tangente em n à tangente no *meio* do intervalo $[x_{n-1}, x_{n+1}]$, o que corresponde a usar a tangente no meio do duplo intervalo para o valor do integral no seu extremo, o que é mais preciso que usar a tangente no início;

ii) usando agora o ponto *previsto* para aí calcular a tangente obtemos dois valores do declive, em (x_n, y_n) e (x_{n+1}, p) cuja média utilizaremos para dar o passo *definitivo*, o que corresponde a usar a média das inclinações nos dois extremos do intervalo extrapolado como aproximação à inclinação média no intervalo.

Calcular declive

$$y'_{n1} = f(x_n, y_n)$$

Calculando em seguida um *valor previsto*:

$$p_{n+1} = y_{n-1} + 2y'_n \times \Delta x_n$$

Usando este valor previsto, calculamos a derivada no ponto previsto

$$p'_{n+1} = f(x_{n+1}\ ,\ p_{n+1})$$

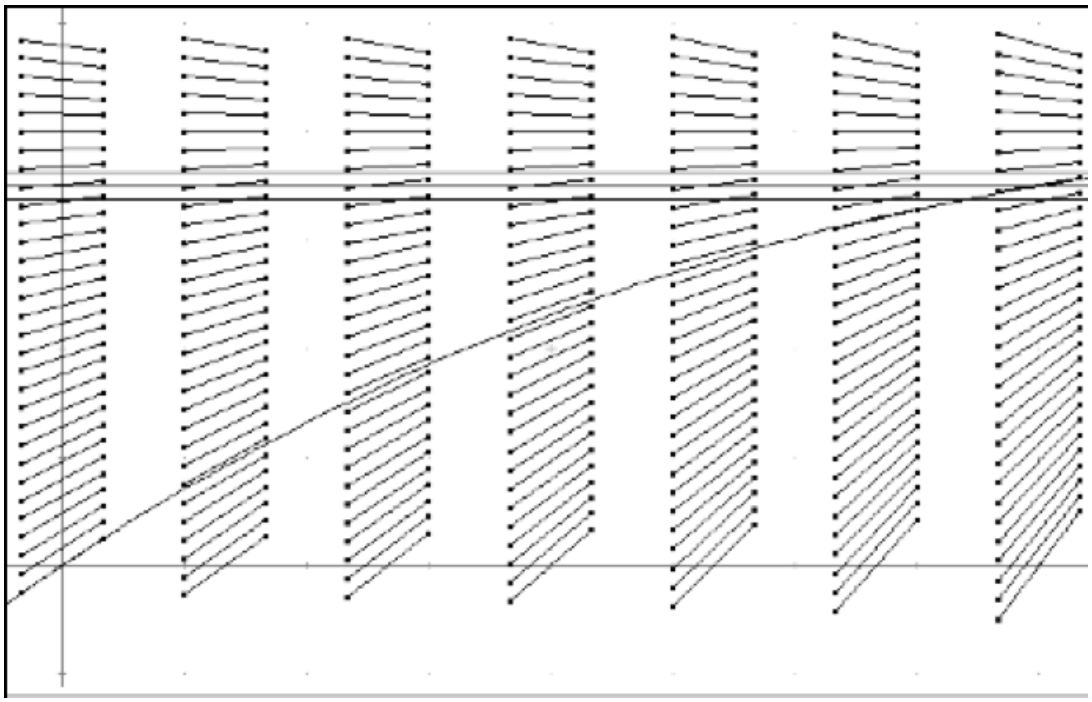
calculando em seguida o *incremento corrigido* de y em x_{n+1}

$$\Delta y_n = \frac{p'_{n+1} + y'_n}{2} \times \Delta x_n$$

aplicando no fim as expressões recorrentes para obter o novo ponto:

$$\begin{cases} y_{n+1} & \leftarrow & y_n + \Delta y_n \\ x_{n+1} & \leftarrow & x_n + \Delta x_n \end{cases}$$

Representação gráfica de um método de Euler melhorado (preditor-corrector).



O método tem, obviamente, o inconveniente de exigir dois pontos de partida e não um; o ponto extra , (x_{n-1}, y_{n-1}) no nosso caso, pode ser calculado por um processo qualquer, desde que suficientemente preciso; por exemplo:

- o método de Euler simples com um passo muito curto;
- um método um pouco mais preciso, baseado em um desenvolvimento em série de Taylor mais longo.

$$y(x+h) = y(x) + h.y'(x) + \frac{h^2}{2}.y''(x) + \frac{h^3}{6}.y''(x) + \ldots$$

calculando as derivadas na forma (o que implica supor que dispomos de uma forma analítica de $f(x,y)$, o que nem sempre será o caso):

$$\begin{aligned} y' &= f(x,y) \\ y'' &= \frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y}.f(x,y) = f'_x + f.f'_y \\ y''' &= f''_x + f''_{xy} + f'_x + f.f'_y \end{aligned}$$

- um método particularmente engenhoso que consiste em usar o próprio corrector, do seguinte modo: dado(x_0, y_0), estimamos (x_1, y_1) na forma simples

$$\begin{aligned} x_{-1} &= x_0 - h \\ y_{-1} &= y_0 - h.y'_0 \\ y'_{-1} &= f(x_1, y_1) \end{aligned}$$

e em seguida usamos o corrector

$$\begin{aligned} y_{-1} &= y_0 - h.\frac{y'_0 + y'_{-1}}{2} \\ y'_{-1} &= f(x_1, y_1) \\ y'_{-1} &= f(x_1, y_1) \end{aligned}$$

sucessivamente até y_{-1} estabilizar dentro da precisão da máquina; se isso não acontecer dentro de meia dúzia de tentativas, teremos que diminuir h.

Pode demonstrar-se, raciocinando como no capítulo anterior, que o erro cometido pela aproximação do preditor é

$$\varepsilon_p = \frac{n.h^3}{3}.f'''(\xi_p)$$

e o erro cometido pela aproximação do corrector é

$$\varepsilon_c = -\frac{n.h^3}{12}.f'''(\xi_c)$$

ou, considerando que y''' conserva o sinal no interior do passo h (e, se assim não fosse, deveríamos reduzir o passo),

$$\begin{aligned} |\varepsilon| &= |\varepsilon_p| + |\varepsilon_c| \\ &= h^3.\left[\frac{1}{3}.\left|f'''(\varepsilon_p)\right| + \frac{1}{12}.\left|f'''(\varepsilon_c)\right|\right] \\ &= \frac{5h^3}{12}.y'''(\xi) \end{aligned}$$

de onde concluímos que o método é de terceira ordem.

Coerentemente com a abordagem anterior, não usaremos esta expressão para calcular o erro (embora disponhamos da facilidade de calcular um majorante de $y'''(\xi)$, não dispomos, à partida, de garantias de que h se comporta já como um infinitésimo de modo a que esta aproximação seja garantidamente válida) mas usaremos a técnica do cociente de convergência e a avaliação do erro que lhe está associada.

Certos autores convencionais recomendam que, a cada passo, se vá calculando ξ para efeitos de controlo da precisão, usando

$$y''' = f''_x + f''_{xy} + f'_x + f.f'_y$$

Outros chegam mesmo ao ponto de sugerir que se adicionem os erros estimados aos valores calculados, a fim de os melhorar. Todas estas técnicas são ingénuas na medida em que estas estimativas dos erros avaliam apenas os efeitos da truncatura, ignorando os efeitos dos arredondamentos que, em muitos casos podem ser dominantes. Assim, por muito que pese aos matemáticos, pensamos que os métodos deste tipo valem o que valem e não são especialmente recomendáveis pelo facto de para eles ser possível uma estimativa do erro.

Por outro lado, não devemos ignorar a importância deste método: na realidade, ele constitui o embrião de uma família de métodos de integração de equações diferenciais ordinárias chamados *métodos preditores-correctores*, ideia muito fecunda que não podemos desenvolver aqui.

5.4 Métodos de Runge-Kutta

Ao contrário dos métodos do tipo preditor-corrector, os métodos do tipo Runge-Kutta não dispõem do conforto um tanto duvidoso de uma "fórmula"para o erro, mas são mais expeditos ao nível da programação, nomeadamente porque não exigem técnicas separadas para o arranque (que, aliás, não ocorre apenas no início do intervalo, mas de todas as vezes que é necessários diminuir o passo).

5.4.1 Método de Runge-Kuta de Segunda Ordem

No caso mais simples, um *método de Runge-Kutta* de segunda ordem, muitas vezes designado por *RK2*, pode visualiza-se do seguinte modo (o raciocínio formal que lhe está subjacente não cabe no escopo do nosso curso):

1. começa-se por calcular y' no início do intervalo: $y'_n = f(x_n, y_n)$
2. a partir daí constrói-se uma estima de y' no meio do intervalo

$$y'_a = f\left(x_n + \frac{\Delta x_n}{2}\ ,\ y_n + \frac{h}{2}.y'_n\right)$$

3. Calcular o incremento de y usando y'_a , e Δx_n, o incremento de x no intervalo n:

$$\Delta y_{n+1} = y'_a\ \Delta x_n$$

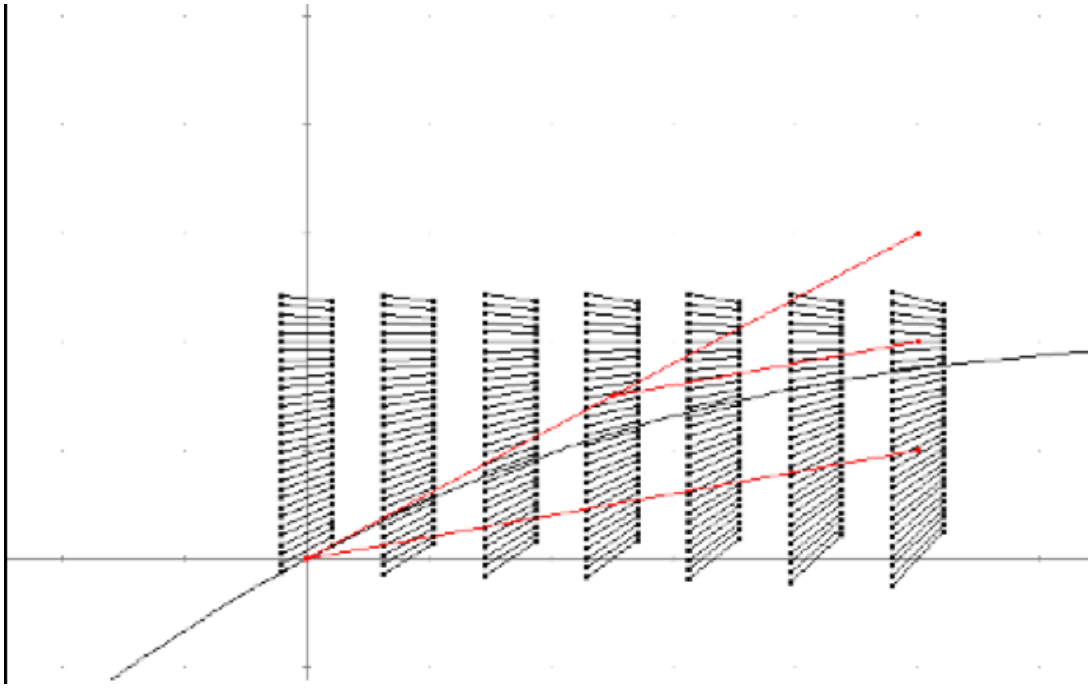
4. Calcular o ponto seguinte:

$$\begin{cases} y_{n+1} & \leftarrow & y_n + \Delta y_n \\ x_{n+1} & \leftarrow & x_n + \Delta x_n \end{cases}$$

   e repetir esses passos até cobrir por completo o intervalo que nos interessa.

Notar-se-á, antes de mais, que esta técnica não é, de modo nenhum, equivalente à aplicação da do *método de Euler* em duas etapas, uma a cada metade do passo de integração.

Pode ver em baixo os gráficos de um Euler directo, de um Euler de meio passo e um Runge-Kutta de segunda ordem.



Para testar a validade do valor de h escolhido em cada passo, deve utilizar-se uma técnica semelhante à esboçada para os métodos anteriores, usando três cálculos de y para h, $h/2$, $h/4$ e verificando se a razão das diferenças sucessivas é aproximadamente igual a 2 levantado à ordem do método.

$$\frac{S' - S}{S'' - S'} \approx 4 \tag{5.3}$$

, e, quando ele for cumprido, poderemos estimar o erro absoluto

$$S'' - S' \approx 3.\varepsilon'' \tag{5.4}$$

Exercício 5.3

Como se disse, este *método de Runge-Kutta*, que, na realidade, é apenas um melhoramento óbvio do método de Euler, é de segunda ordem, enquanto o anterior método preditor-corrector é de terceira. Conseguirá o leitor descobrir de onde provém a diferença?

5.4.2 Método de Runge-Kuta de Quarta Ordem

É possível construir, mediante um raciocínio semelhante, um método de Runge-Kutta de terceira ordem, mas o interesse é pequeno porque o trabalho de cálculo envolvido é pouco menor que o de um *método de Runge-Kutta* de quarta ordem, conhecido por *RK4*.

Este pode ser estabelecido do seguinte modo:

1. começa-se por, usando y'_n no início do passo de integração, calcular uma primeira estima δ_1 para o incremento de Δy_n ;
$$\delta_1 = \Delta x_n . f\left(x_n\ ,\ y_n\right)$$

2. a partir desta estima, calcula-se uma primeira estima dey' no meio do passo e, a partir deste valor, uma segunda estima δ_2 para o incremento de Δy_n :
$$\delta_2 = \Delta x_n . f\left(x_n + \frac{\Delta x_n}{2}\ ,\ y_n\ + frac\delta_1 2\right)$$

3. calcula-se uma segunda estima de y'_n no meio do passo, usando a anterior, e a partir deste valor, uma terceira estima δ_3 para o incremento de Δy_n :
$$\delta_3 = \Delta x_n . f\left(x_n + \frac{\Delta x_n h}{2}\ ,\ y_n + \frac{\delta_2}{2}\right)$$

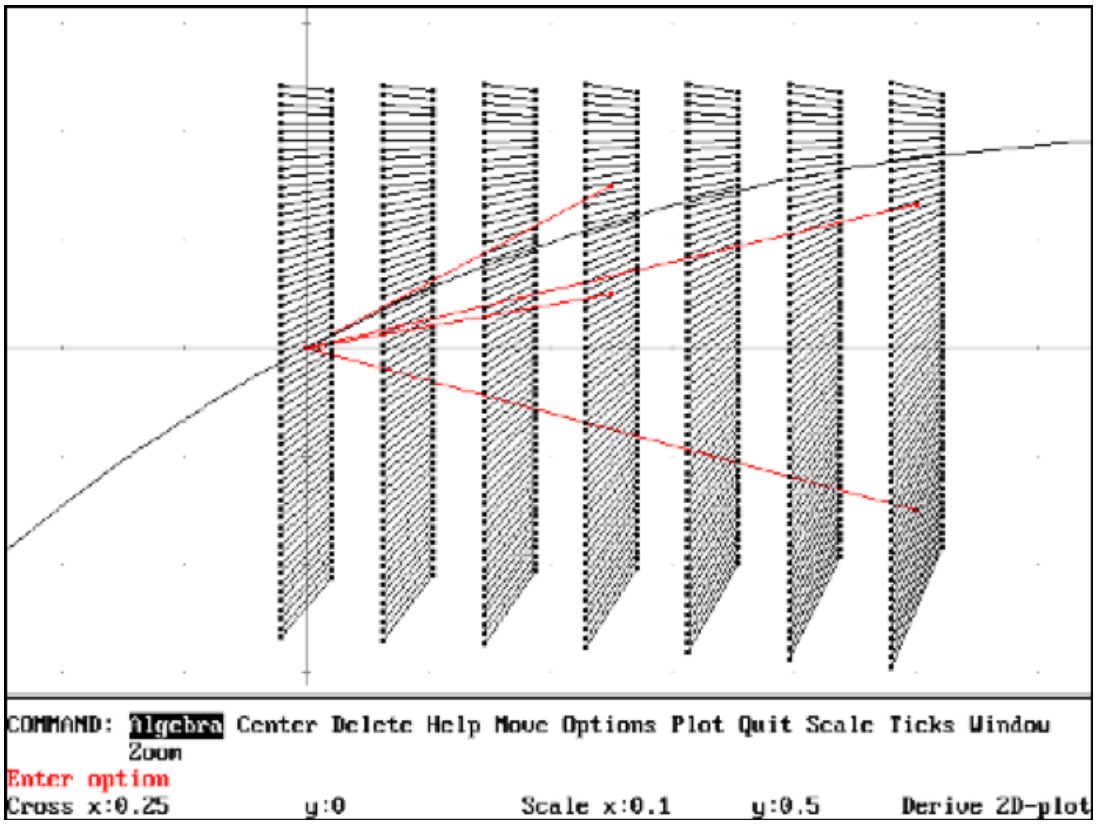
4. finalmente, obtém-se uma última estima de y'_n no fim do passo, usando a anterior, e a partir deste valor, uma quarta estima δ_4 para o incremento de Δy_n :
$$\delta_4 = \Delta x_n . f\left(x_n + \Delta x_n\ ,\ y_n + \delta_3\right)$$

5. calcula-se a média ponderada das diversas estimas δ_i estabelecendo os pesos de modo a garantir um erro proporcional a $\left(\Delta x_n\right)^4$
$$\Delta y_n = \frac{\delta_1}{6} + \frac{\delta_2}{3} + \frac{\delta_3}{3} + \frac{\delta_4}{6}$$

6. e usa-se essa média para calcular o valor y_{n+1} em x_{n+1}

7. Calcular o ponto seguinte:
$$\begin{cases} y_{n+1} & \leftarrow & y_n + \Delta y_n \\ x_{n+1} & \leftarrow & x_n + \Delta x_n \end{cases}$$

e repetir esses passos até cobrir por completo o intervalo que nos interessa.



Para testar a validade do valor de h escolhido em cada passo, deve utilizar-se uma técnica semelhante à esboçada para os métodos anteriores, usando três cálculos de y para h, $h/2$, $h/4$ e verificando se a razão das diferenças sucessivas é aproximadamente igual a 2 levantado à ordem do método.

$$\frac{S' - S}{S'' - S'} \approx 16 \tag{5.5}$$

, e, quando ele for cumprido, poderemos estimar o erro absoluto

$$S'' - S' \approx 15.\varepsilon'' \tag{5.6}$$

Exercício 5.4

1. Integre as equações dos exemplos e exercícios anteriores pelos métodos de Runge-Kutta de segunda e quarta ordens e compare os resultados e os tempos de cálculo entre si e com os anteriores.
2. Como implementaria uma estratégia de passo variável no RK4 ?

5.5 Forma geral

Como vimos, a forma geral para os vários métodos consiste na aplicação repetida de um sistema de expressões recorrentes que formam um passo do método:

Em primeiro lugar obter os acréscimos nas várias variáveis, mediante a aplicação simultânea e repetida de:

- Cálculo do acréscimo δy_n pela aplicação do método escolhido;
- Escolha do acréscimo δx_n, usando uma estratégia de:
  i) passo constante;
  ii) passo variável pela avaliação de QC;
  iii) passo variável ...

Finalmente aplicar as expressões recorrentes que permitem obter (x_{n+1}, y_{n+1}) a partir de (x_n, y_n):

$$\begin{cases} y_{n+1} & \leftarrow & y_n + \Delta y_n \\ x_{n+1} & \leftarrow & x_n + \Delta x_n \end{cases}$$

5.6 Sistemas de Equações e Equações de Ordem Superior

Embora a nossa discussão se tenha centrado no caso de uma equação única e de primeira ordem, os resultados que obtivemos estendem-se, sem necessidade de considerações especiais, ao caso de um sistema de *n* equações diferenciais ordinárias de primeira ordem:

$$\begin{cases} y_1' = f_1(x, y_1, y_2, \cdots, y_n) \\ y_2' = f_2(x, y_1, y_2, \cdots, y_n) \\ \quad \cdots \\ y_n' = f_1(x, y_1, y_2, \cdots, y_n) \end{cases}$$

fazendo, simplesmente, qualquer que seja o método de resolução adotado, o mesmo encadeado de operações, realizando-as em paralelo sobre todas as equações.

Exemplo: 55.8 Sistema de equações diferenciais

Vamos integrar o sistema de equações diferenciais

$$\begin{cases} z' = & x + y + z \\ y' = & 2.x - y - z \end{cases}$$

usando o *método de Euler*, com as expressões recorrentes

$$\begin{cases} z_{n+1} \leftarrow & z_n + (x_n + y_n + z_n).\Delta x_n \\ y_{n+1} \leftarrow & y_n + (2.x_n - y_n - z_n).\Delta x_n \\ x_{n+1} \leftarrow & x_n + \Delta x_n \end{cases}$$

começando em $(x_0, y_0, z_0) = (0, 1, 1)$, com passo constante $\Delta x = h = 1$
As três primeiras iterações são:

Exemplo: 55.8 Sistema de equações diferenciais (cont.)

iter	x	y	z	y'	z'
0	0	1	1	2	-2
1	1	3	-1	3	0
2	2	6	-1	7	-1
3	3	13	-2	14	-5

Equações de ordem superior, por exemplo, de segunda ordem

$$y" = f(x, y, y')$$

podem reduzir-se à primeira ordem mediante o expediente de fazer

$$y' = z$$

(que, por si, constitui uma equação diferencial de primeira ordem) e

$$z' = f(x, y, z)$$

que é uma outra equação diferencial de primeira ordem que forma sistema com a anterior:

$$\begin{cases} z' = f(x, y, z) \\ y' = z \end{cases}$$

Exemplo: 55.9 Sistema de duas equações de segunda ordem

Reduzamos o sistema de duas equações de segunda ordem

$$\begin{cases} y'' = f\left(x, y, y', z, z'\right) \\ z'' = g\left(x, y, y', z, z'\right) \end{cases}$$

a um sistema de quatro equações de primeira ordem:

$$\begin{cases} u' = f\left(x, y, u, z, v\right) \\ y' = u \\ v' = g\left(x, y, u, z, v\right) \\ z' = v \end{cases}$$

Exemplo: 55.10 Equação diferencial de segunda ordem

Vamos integrar a equação diferencial de segunda ordem

$$y'' = 2.x - y - 0.5.y'$$

começando por a transformar num sistema de equações diferenciais de primeira ordem, fazendo $y' = z$:

Exemplo: 55.10 Equação diferencial de segunda ordem (cont.)

$$\begin{cases} z' = & 2.x - y - 0.5.z \\ y' = & z \end{cases}$$

Vamos usar o *método de Euler*, com as expressões recorrentes

$$\begin{cases} z_{n+1} \leftarrow & z_n + (2x_n - y_n - 0.5z_n).\Delta x_n \\ y_{n+1} \leftarrow & y_n + z_n.\Delta x_n \\ x_{n+1} \leftarrow & x_n + \Delta x_n \end{cases}$$

começando em $(x_0, y_0, y'_0) = (0, 1, 1)$, com passo constante $\Delta x = h = 1$
As três primeiras iterações são:

iter	x	y	z	y'	z'
0	0	1	1	-1,5	1
1	1	-0,5	2	1,5	2
2	2	1	4	1	4
3	3	2	8	0	8

Bibliografia

[CB81] Conte and De Boor. *Elementary Numerical Analysis, an algorithmic approach*. Mc-Graw-Hill International Editions, Singapore, 1981.

[DB74] Dahlquist and Björck. *Numerical Methods*. Prentice-Hall, Inc., Englewood Cliffs, 1974.

[Gol90] Herman H. Goldstine. *Remembrance of things past*. ACM Press - Addison-Wesley, Reading, Massachusetts, 1990.

[Gol91] Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1), March 1991.

[GU99] René Goscinny and Albert Uderzo. *Le Tour de Gaule d'Astérix*. Hachette, Paris, 1999.

[Ham71] Hamming. *Introduction to applied numerical analysis*. McGraw-Hill-Kogakusha, Ltd., Tokyo, 1971.

[IEE19] IEEE. Ieee standard for floating-point arithmetic. *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, pages 1–84, July 2019.

[JR75] Jensen and Rowland. *Methods of Computation*. Scott, Foresman & Co, Glenview, 1975.

[Knu81] D.E. Knuth. *The Art of Computer Programming: Seminumerical algorithms*. Addison-Wesley series in computer science and information processing. Addison-Wesley, 1981.

[Lie68] Lieberstein. *A course in numerical analysis*. Harper and Row, New York, 1968.

[Max19] Maxima. Maxima, a computer algebra system. version 5.43.0, 2019.

[Moo66] R.E. Moore. *Interval analysis*. Prentice-Hall series in automatic computation. Prentice-Hall, 1966.

[Mul89] Muller. *Arithmétique des ordinateurs*. Masson, Paris, 1989.

[Sof] Soft WhareHouse. Derive, a mathematical assistant ^{TM}. actualmente disponíve nas calculadoras TI-Nspire CAS.

[Wal90] Wallis. *Improving floating-point programming*. Wiley, New York, 1990.

[Wik19] Wikipedia contributors. Ieee 754 — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=IEEE_754&oldid=917946439, 2019. [Online; accessed 26-September-2019].

[Wil65] Wilkinson. *Rounding errors in algebraic processes*. Clarendon Press, Oxford, 1965.

[Zac96] Joseph L. Zachary. *Introduction to scientific programming: computational problem solving using Maple and C*. Springer Verlag New York Inc., New York, 1 edition, 1996.

Índice