Numerical Methods

Laura Maria Quiros Conesa

October 2024

1 Basics

1.1 Convergence Orders

1. linear: bisection method
2. quadratic: newton method, trapezoidal and midpoint method
3. superlinear: secant method $\alpha = \frac{1}{2}(1+\sqrt{5})$, $K = \beta = \frac{\epsilon_n}{\epsilon_{n-1}^2}$
4. quadric: simpson's method

The Truncation Error is the difference between the exact integral and the result produced by using any approximation, abstracted away and expressed in terms of O-notation based on step size h.

2 Root-finding Methods

2.1 Bisection (!), Newton's and Secant Method

Bisection Method: If $f(a) * f(m) < 0$ means that either of them is 0 or negative, we can make $b = m$ halving the search area. If $f(a) * f(m) > 0$ then we make $a = m$.

Newtons Method: $x_{n+1} = x_n - \frac{f(x_n)}{f\prime(x_n)}$, Secant Method: $x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$

2.2 Is this iteration method successful? Fast convergence? Divergence?

$|g'(x_n)| < 1$ then it will converge. Otherwise it is not successful. We can tell how successful it is by comparing it to 0, which is value of optimal linear convergence

$$\begin{aligned} g(x) &= x_n + \alpha f(x) \\ g\prime(x) &= 1 + \alpha f\prime(x) \\ g\prime(x_n) &= 1 + \alpha f\prime(x_n) \\ |g\prime(x_n)| &= 0 \qquad \text{(then it's optimal } \alpha) \end{aligned}$$

and correct x_0!!. Divergence is when the function gets away from the root. So if $|g'(x_n)| > 1$ for some x_n and the subsequent x_{n+1} get higher and higher instead of oscillating, we can say it diverges.

2.3 Determine error estimate

First you need factor of error reduction $K = \frac{x_4 - x_3}{x_3 - x_2}$ then you do $\epsilon_4 = \frac{K}{1-K} * |(x_4 - x_3)|$ make sure to make the multiplication there, review `ANS` in calculator. Remember that for the bisection method, the constant K is always $\frac{1}{2}$ and for secant, error estimate can be $|\frac{x_2 - x_1}{x_2}|$

Order of error reduction is to compare factor K with convergence $|g\prime(x_0)|$. If $e_{n+1} \approx K \cdot e_n$, then linear. Otherwise consider other α.

Theoretical error reduction is $g\prime(\pi)$, reduction rate of exact error $\frac{x_4 - \pi}{x_3 - \pi}$ and convergence rate is K as explained above, for π is the exact solution.

2.3.1 How many iterations for an accuracy level

$$K^n * \epsilon_4 < 10^8, K^n < \frac{10^8}{\epsilon_4}, n < \frac{\log(\frac{10^8}{\epsilon_4})}{\log(K)}$$

2.4 Extrapolation

For linear convergence $T(n) = 2x_n - x_{n-1}$

3 Numerical Integration

3.1 Midpoint Rule, Trapezoidal Simpson Method

Midpoint: Height is $f(x_{i-1}+\frac{\Delta x}{2}) = f(m)$ and base is Δx. Area will be $\Delta x * f(m)$ per rectangle.

Trapezoidal: Individual area is $\frac{\Delta x}{2} * (f(x_{i-1}) + f(x_i))$.

Simpsons: $\text{Area}_i = \frac{h}{6}(f(x_{i-1}) + 4f(\frac{x_{i-1}+x_i}{2}) + f(x_i))$

Midpoint and trapezoidal second order convergence because error can be explained with 3.4.1. Simpsons is quadric with error $\frac{\hat{h}^4}{180}(b-a)M$.

3.2 Q-factor

$$q = |\frac{x_2 - x_3}{x_3 - x_4}|$$

If q not close to the desired value it might be because h is not small enough $\Rightarrow$ 3rd-order terms in ϵ not negligible w.r.t. 2nd-order term ch^2 and also it may be because function f not smooth enough or is not bound.

3.3 Richardson Extrapolation

$T_h = I_{h/2} + \frac{I_{h/2}-I_h}{3} = \frac{4}{3}I_{h/2} - \frac{1}{3}I_h$ and if you want to combine two you do $T \approx \frac{16}{15}T_h - \frac{1}{15}T_{h/2}$. Richardson extrapolation is allowed as long as $f(x)$ is stable aka q-factor is according to theory.

It does not make sense to further extrapolate when we are already running out of given decimals and the computations are too similar to each other.

3.4 Errors

3.4.1 Global error theorem

$\epsilon = \frac{b-a}{24} \cdot f''_{\max} \cdot (\frac{1}{N})^2$ for N is number of rectangles added. When is it useful? When we have a bounded integral aka the double derivative is stable and it will lead to optimal convergence. Optimal convergence: Look for singularities like divisions of 0 and oscillatory behavior added by trigonometric operations in $f''(0)$. Then it's not bound. Careful with signs and chain rules.

1. Error estimate based on I $\epsilon_h = |\frac{1}{3}(I_h - I_{h/2})|$
2. How to know which estimate is better: M belongs to 1 location while the one based in I_n is over the whole range
3. How many intervals are required (powers of 2) for the method to achieve accuracy of extrapolation value X? Intervals increase to the rate of 1/4 for quadratic methods so $(1/4)^m \cdot \epsilon < 10^{-9}$ and then we follow like 2.3.1. that is m times the number of segments from the epsilon for the total amount

4 ODE

q-factor $q = |\frac{i_h - i_{h/2}}{i_{h/2} - i_{h/4}}|$

error estimate on fine grid is calculated with first-order extrapolation $2y_{h/2} - y_h$ with second order $1/3(y_{h/2} - y_h)$. Error estimate in later iter could be bigged due to error acumulation. which grid for this error we use $n^2 < \frac{10^k}{\epsilon}$ if second order method and then $n * \Delta_X$ equals number of segments. For heun K is always $(\frac{1}{4})^n * \epsilon < 10^k$.

4.1 Runge-Kutta RK methods

for extrapolating if method is 2nd order $4/3y_{h/2} - 1/3y_h$ and if 3rd order $8/7y_{h/2} - 1/7y_h$ order of convergence for extrapolation solution is same as method order.

4.2 Heun's Method RK2

$y_{n+1} = y_0 + \frac{1}{2}(h * f(x_n, y_n) + h * f(x_{n+1}, y_0 + h * f(x_{n+1}, y_n))$ Often high numbers are visible in the table corner if N too small. Oscillations may also indicate that is not advisable to use it.

4.3 Euler Method

We cannot use formula $|1 + ah| < 1$ if the function doesn't have shape $y\prime = ay$ otherwise this is how we calculate stability. It may have shape $1/1 - h * f(x_n, y(x_n))$ for explicit. explicit euler $y(x_{n+1}) = y(x_n) + h * f(x_n, y(x_n))$ implicit euler $y(x_{n+1}) = y(x_n) + h * f(x_n, y(x_{n+1}))$ leads to problems because it may lead to complex numbers when using quadratic eq.

5 Curve Fitting

5.1 Coordinate transformation

pay attention to if they want -1,0,1 or some other specific thing. Overall, $\hat{x} = \frac{x-x_i}{x_i}$ for x_i is the x you want to center around. This is necessary to have more balanced and easier to solve set of equations (matrix).

5.2 Fit

Linear and polynomial fit

$$M_N a_0 + M_{N+1} a_1 + M_{N+2} a_2 + \cdots + M_{2N} a_N = F_N$$

$$M_k = \sum_{i=1}^{M} x_i^k, F_k = \sum_{i=1}^{M} f_i \cdot x_i^k$$

Exponential: remember to always linearise the y_i first!

why do we say "approximate" f(0)? bc it's a curve that goes straight through the data.

6 Iterative Differential

6.1 Max norm of initial residual

$r_0 = Ax_0 - b = (r_1, r_2, r_3)^T$ then the max norm is the max value in the absolute vector

6.2 Determine LU factorisation

$$A = \begin{pmatrix} a_1 & c_2 & 0 & & 0 \\ b_2 & a_2 & c_2 & & 0 \\ 0 & b_3 & a_3 & c_3 & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & a_n \end{pmatrix} = LU = \begin{pmatrix} \alpha_1 & 0 & 0 & & 0 \\ b_2 & \alpha_2 & 0 & & 0 \\ 0 & b_2 & \alpha_2 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & \alpha_n \end{pmatrix} \begin{pmatrix} 1 & \gamma_1 & 0 & & 0 \\ 0 & 1 & \gamma_2 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ \vdots & & & 1 & \gamma_{n-1} \\ 0 & & & 0 & 1 \end{pmatrix}$$

6.2.1 Matrix to differential eq, make matrix-vector system

Make use of the Neumann approximation $y\prime(i-1) \approx \frac{y_i - y_{i-1}}{h} = g$ for the boundary conditions $y\prime(x) = g$. You need to use double derivative discretisation $\frac{y_{i-1} - 2y_i + y_{i+2}}{\Delta_x^2}$ and remember internal points are the points between the initial and final point. To make matrix tridiagonal /prepare for TDMA, you really only need to do row/column elimination. Some value of α may lead to or singular matrix ($y_1 = y_2 = y_3 = y_4$)

6.2.2 Jacobi Theorem

If A is diagonally dominant, then Jacobi converges with linear rate $\mu < 1$, independent of start vector x^0. For $= 1$, it may not converge but we have no proof to say it will diverge. We calculate $\mu = \max(\frac{\sum_{i \neq j} |a_i j|}{|a_{ii}}$) and to get the number of iter until factor 10^k error reduced, we need $\mu^n < 10^k$.

6.3 Jacobi, Gauss-Seidel and SOR Method

Jacobi $x_i^{(m+1)} = \frac{1}{a_{ii}}\{b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(m)}\}, i = 1 \cdots n$, Gauss-Seidel $x_i^{(m+1)} = \frac{1}{a_{ii}}\{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(m+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(m)}\}, i = 1 \cdots n$ and SOR is gauss seidel but then we do $x_i^{m+1} = \omega * \hat{x}_i + (1 - \omega) * x_i^m$. SOR Method will diverge if we see oscilating values or overshooting in the x-vector (0,1,0,1) to (1,2,3,2). Also if the matrix is not diagonally dominant

7 Partial Differential Equations

7.1 Euler explicit and implicit

explicit: $u_m^{n+1} = ru_m^n + (1-2r)u_m^n + ru_{m+1}^n$

implicit: $u_m^n = -ru_m^{n+1} + (1-2r)u_m^{n+1} - ru_{m+1}^{n+1}$ 1 pro: larger Δ_t possible, no inestabilities, 1 con: each timestep $Ax = b$ has to be solved which may take long time.

crank-nicholson: $-u_{m+1}^{n+1} + (2/r-2)u_m^{n+1} - u_{m+1}^{n+1} = u_{m-1}^n + (2/r-2)u_m^n + u_{m+1}^n$ pros: it has $\mathcal{O}(\Delta_t^2, \Delta_x^2)$ instead of $\mathcal{O}(\Delta_t, \Delta_x^2)$, better accuracy in Δ_t and implicit method, no stability limit(always stable) cons: more complicated, set up and solve linear system each timestep. Solve for i-th row with source term: $\frac{T_i^{n+1}-T_i^n}{\Delta_t} = \frac{k}{2}\frac{T_{i-1}^{n+1}-2T_i^{n+1}+T_{i+2}^{n+1}}{\Delta_x^2} + \frac{k}{2}\frac{T_{i-1}^{n}-2T_i^{n}+T_{i+2}^{n}}{\Delta_x^2} + Q(x_i, t_n)$

7.2 stability restriction

$0 \leq r \leq \frac{1}{2}$ is necessary and sufficient for stability. Unconditionally stable if source term enhances.

7.2.1 Diffuse condition

restriction for number of segments w timestep info, timestep limit for number of segments info, does it change with amplitude change, is it wise to use max timestep $R = \frac{\Delta_t * D}{\Delta_x^2} < \frac{1}{2}$ A large timestep for a $\mathcal{O}(\Delta_t)$ method is not advisable since it will cause lower acurracy.

7.2.2 Partial diff discretisation

$$\frac{T_i^{n+1} - T_i^n}{\Delta_t} = k\frac{T_{i-1}^{n+1} - 2T_i^{n+1} + T_{i+2}^{n+1}}{\Delta_x^2} + Q(x_i, t_n) \tag{1}$$

$$T_i^{n+1} = T_i^n + \frac{k * \Delta_t}{\Delta_x^2}(T_{i-1}^{n+1} - 2T_i^{n+1} + T_{i+2}^{n+1}) + \Delta_t * Q(x_i, t_n)$$

(add discretisation)

$$T_i^{n+1} = T_i^n + \frac{k * \Delta_t}{\Delta_x^2}(T_{i-1}^{n+1} - 2T_i^{n+1} + T_{i+2}^{n+1}) + \Delta_t * (T_i^n - T_{i-1}^n)$$

($i^{n+1} - i^n$ in cn method)

get the coefficients and look at the one for i. i.e. $T_i = 1 - 2R - \mu, \mu = \frac{\Delta_t * U}{\Delta_x}$, that has to be higher than 0. If this leads to $R < 1/2 + \Delta_t$ then it is enhanced by the new discretisation. If the Δ_t is now more limited we can say that we have more strict stability for different gradient discretisation. does same spatial acurracy lead to better computing times: compare timestep restrictions if any, otherwise not faster

7.3 Source term

explain added source term to model, add it for which f_n time and between which x_i positions. It is added at the end multiplied by Δ_t due to the multiplication of it in our discretisation of the partial diff eq.