9. Numerical Methods

- *the Taylor series*
- *finite-difference calculus*
- *interpolation and extrapolation*
- *solving single equations*
- *solving sets of equations*

9. Numerical Methods

a. *Introduction*

The material in this chapter has not been part of the traditional focus of experimentalists. It is included here because, with continuing improvement in the power and accessibility of computers for data analysis, these tools have growing involvement in the analysis and interpretation of data and in the fitting of theoretical models and concepts to data. The methods are valuable in the many cases where analytical solution is not possible or feasible.

b. *The Taylor series*

Many numerical methods are based on the Taylor expansion, so a brief review of the Taylor series is included here. The Taylor series expansion of a function $f(x)$ is

$$f(x) = f(x_0) + \sum_{k=1}^{\infty} \frac{(x - x_0)^k}{k!} f^{(k)}(x_0) \tag{9.1}$$

where $f^{(k)}$ is the kth derivative of the function. Differing approximations to the function are obtained from this series by truncation. If the series is truncated at the *nth* term, the maximum error in the approximation is

$$\frac{|x - x_0|^{n+1}}{n!} \left| f^{(n+1)} \right|_{max} \tag{9.2}$$

where the "max" subscript indicates the maximum value of the derivative in the interval from x to x_0. The error is said to be of order $(x - x_0)^{n+1}$.

The Taylor expansion must be used with some caution because the series does not converge for all values and sometimes converges very slowly.

c. *Finite-difference derivatives*

Finite-difference formulas make it possible to use arithmetic operations to determine derivatives. They are based on the Taylor series truncated at various orders in the expansion. The first-order expansion, of course, just gives the form

$$f'(x) = \frac{f(x+\delta) - f(x)}{\delta} + \mathcal{O}(\delta) \tag{9.3}$$

or

$$f'(x) = \frac{f(x) - f(x-\delta)}{\delta} + \mathcal{O}(\delta) \tag{9.4}$$

where $\mathcal{O}(\delta)$ indicates that the error in this approximation is of order δ. If an index is used to indicate consecutive values at intervals δ, so that $f_{i+1} = f(x+\delta)$ if $f_i = f(x)$, then

$$f_i'(x) = \frac{(f_{i+1} - f_i)}{\delta} \tag{9.5}$$

is called the first forward difference and

$$f_i'(x) = \frac{(f_i - f_{i-1})}{\delta} \tag{9.6}$$

is the first backward difference formula for the derivative.

Similar expressions can be found for higher-order derivatives. For example, the second derivative is

$$f_i'' = \frac{f_{i+1}' - f_i'}{\delta} + \mathcal{O}(\delta) \tag{9.7}$$

$$= \frac{\frac{f_{i+2}-f_{i+1}}{\delta} - \frac{f_{i+1}-f_i}{\delta}}{\delta} + \mathcal{O}(\delta) \tag{9.8}$$

$$= \frac{f_{i+2} - 2f_{i+1} + f_i}{\delta^2} + \mathcal{O}(\delta) \quad . \tag{9.9}$$

The corresponding backward-difference formula is

$$f_i'' = \frac{f_{i-2} - 2f_{i-1} + f_i}{\delta^2} + \mathcal{O}(\delta) \quad . \tag{9.10}$$

The preceding formulas were obtained by taking only the first term in the Taylor series containing the desired derivative. More accurate formulas can be obtained by retaining more terms. For example,

$$f'(x) = \frac{f(x+\delta) - f(x)}{\delta} - \frac{\delta}{2} f''(x) + \cdots \tag{9.11}$$

$$= \frac{f(x+\delta)-f(x)}{\delta} - \frac{\delta}{2}\left[\frac{f(x+2\delta)-2f(x+\delta)+f(x)}{\delta^2} + \mathcal{O}(\delta)\right] + \mathcal{O}(\delta^2) \qquad (9.12)$$

or

$$f_i' = \frac{-f_{i+2}+4f_{i+1}-3f_i}{2\delta} + \mathcal{O}(\delta^2) \ . \qquad (9.13)$$

Another improvement is to use *central differences*:

$$f_{i+1} = f_i + \delta f_i' + \frac{\delta^2}{2!}f_i'' + \cdots \qquad (9.14)$$

$$f_{i-1} - f_i - \delta f_i' + \frac{\delta^2}{2!}f_i'' + \cdots \qquad (9.15)$$

$$f_i' = \frac{f_{i+1}-f_{i-1}}{2\delta} + \mathcal{O}(\delta^2) \ . \qquad (9.16)$$

The centered-difference formula is more accurate than the forward or backward-difference formulas, but still involves only two terms. Table 9.1 lists some finite-difference formulas for derivatives.

Finite-difference formulas are often used to evaluate derivatives of functions when analytical expressions are not available. For example, the procedures for nonlinear least-squares fitting often need derivatives of the chisquare function with respect to the parameters, and these are sometimes easier to evaluate numerically than analytically. Because the error in the derivatives scales with the size of δ, the step used for evaluating such derivatives should be small, but not so small that rounding errors are introduced by the digital representation of the values in the computer.

d. *Interpolation and extrapolation*

Often experimental results are available for selected conditions, but values are needed for intermediate conditions. For example, the fall speeds of raindrops are measured for specific diameters, but calculation of a rainrate from a measured drop size distribution requires fall speeds for intermediate diameters. The estimation of such intermediate values is called interpolation. Another common use for interpolation is when the functional dependence is so complicated that explicit evaluation is costly in terms of computer time. In such cases, it may be more efficient to evaluate the function at selected points spanning the region of interest and then use interpolation, which can be very efficient, to determine intermediate values.

Extrapolation is the extension of such data beyond the range of the measurements. It is much more difficult, and can result in serious errors if not used and interpreted properly. For example, a high-order polynomial may provide a very good fit to a data set over its range of validity, but if higher powers than needed are included, the polynomial may diverge rapidly from smooth behavior outside the range of the data.

1). Finite-difference interpolation formulas

Table 9.1: Finite-difference derivatives

The entries in this table are the coefficients of the corresponding function value f_{i+j} in the expression for the derivative. For example, the second-order centered-difference expression for f''' is $(-f_{i-2} + 2f_{i-1} - 2f_{i+1} + f_{i+2})/(2\delta^3)$ where δ is the interval between consecutive values f_i.

derivative	f_{i-3}	f_{i-2}	f_{i-1}	f_i	f_{i+1}	f_{i+2}	f_{i+3}	f_{i+4}
centered ($O(\delta^2)$)								
$\delta f_i'$			-1	0	1			
$\delta^2 f_i''$			1	-2	1			
$2\delta^3 f_i'''$		-1	2	0	-2	1		
centered ($O(\delta^4)$)								
$12\delta f_i'$		1	-8	0	8	-1		
$12\delta^2 f_i''$		-1	16	-30	16	-1		
$8\delta^3 f_i'''$	1	-8	13	0	-13	8	-1	
forward ($O(\delta)$)								
$\delta f_i'$				-1	1			
$\delta^2 f_i'$				1	-2	1		
$\delta^3 f_i'''$				-1	3	-3	1	
forward ($O(\delta^2)$)								
$2\delta f_i'$				-3	4	-1		
$\delta^2 f_i''$				2	-5	4	-1	
$2\delta^3 f_i'''$				-5	18	-24	14	-3

Interpolation formulas based on the Taylor series use the finite-difference formulas of the preceding section. One might consider starting with

$$f(x - x_i) = f_i + (x - x_i)\left[\frac{f_{i+1} - f_i}{\delta}\right] + \frac{(x - x_i)^2}{2!}\left[\frac{f_{i+2} - 2f_{i+1} + f_i}{\delta^2}\right]\cdots \tag{9.17}$$

However, a better expression results from keeping terms that are of consistent order in the expected error. Equation (9.12) can be rewritten as

$$f_i' = f_F' - \frac{\delta}{2!}f_i'' - \frac{\delta^3}{3!}f_i''' + \cdots \tag{9.18}$$

where f_F' is the first-order finite difference formula (9.6) evaluated at x_i. Similarly,

$$f_i'' = f_F'' - \delta f_i''' + \cdots \tag{9.19}$$

If the third derivative is included in the interpolation formula, the third-derivative corrections to the finite difference formulas for the first and second derivatives should also be included. The result of regrouping the series to consistent order is the Gregory-Newton interpolation formula:

$$f(x-x_i) = f_i + (x-x_i)f'_F + \frac{(x-x_i)(x-x_{i+1})}{2!}f''_F + \frac{(x-x_i)(x-x_{i+1})(x-x_{i+2})}{3!}f''_F \cdots \tag{9.20}$$

where the finite-difference formulas with subscripts F are the forward-difference formulas with error limits of order $\mathrm{O}(\delta)$ from Table 9.1, evaluated at x_i. The backward-interpolation formula is similar except backward difference formulas are used and terms involving $(x - x_{i+1})$ become $(x - x_{i-1})$, etc. Both of these interpolation formulas can be used for extrapolation beyond the limits of the available data, but usually should be used only for distances of about the data spacing.

Stirling's formula is an analogous formula evaluated with central-difference derivatives. It is usually preferable for interpolation within the range of a table. It is

$$f(x-x_i) = f_i + (x-x_i)f'_C + \frac{(x-x_i)^2}{2}f''_C + \frac{(x-x_i)((x-x_i)^2-1)}{3!}f'''_C + \frac{(x-x_i)^2((x-x_i)^2-1)}{4!}f''''_C + \cdots \tag{9.21}$$

where the derivatives with subscript C are the centered-difference formulas with accuracy of order $\mathrm{O}(\delta^2)$ from Table 9.1, evaluated at x_i.

2). Lagrange interpolation

An alternate method of interpolation is to use polynomial fits to the available values to interpolate between those values. If there are N data values, a polynomial of degree $N-1$ can be found that will pass through all the points. The Lagrange polynomials provide a convenient alternative to solving the simultaneous equations that result from requiring the polynomials to pass through the data values. The Lagrange interpolation formula is

$$f(x) = \sum_{i=1}^{N} f(x_i)P_i^L(x) \tag{9.22}$$

where $f(x_i)$ are the known values of the function and $f(x)$ is the desired value of the function. The Lagrange polynomial P_i^L is the polynomial of order $N-1$ that has the value 1 when $x = x_i$ and 0 for all $x_{j\neq i}$:

$$P_i^L(x) = \frac{\prod_{j\neq i}(x-x_j)}{\prod_{j\neq i}(x_i-x_j)} \quad . \tag{9.23}$$

This is a particularly convenient way to interpolate among tabulated values with polynomials.

An advantage of Lagrange interpolation is that the method does not need evenly spaced values in x. However, it is usually preferable to search for the nearest value in the table and then use the lowest-order interpolation consistent with the functional form of the data. High-order polynomials that match many entries in the table simultaneously can introduce undesirable rapid fluctuations between tabulated values. If used for extrapolation with a high-order polynomial, this method may give serious errors.

3). Whittacker's interpolation formula

For time series analysis, the discrete samples in a series were represented in section 8h by the function

$$g(t) = f(t)I(t) \tag{9.24}$$

where

$$I(t) = \sum_m \delta(t - m\Delta T) \quad . \tag{9.25}$$

The Fourier transform of the delta function was a series of delta functions in frequency space, and this led to mixing together of various frequency components or to aliasing. However, if the Fourier transform of $f(t)$, $\tilde{f}(\nu)$, is zero for frequencies above the Nyquist frequency, no contamination of the signal by aliasing occurs, and the true Fourier transform can be recovered from

$$\tilde{f}(\nu) = \tilde{g}(\nu)\tilde{h}(\nu) \tag{9.26}$$

where

$$\begin{aligned} \tilde{h}(\nu) &= \Delta, \quad |\nu| \leq \frac{1}{2\Delta T} \\ &= 0 \quad \text{otherwise.} \end{aligned} \tag{9.27}$$

The Fourier transform of $\tilde{h}(\nu)$ is

$$h(t) = \frac{\sin(\pi t/\Delta)}{\pi t/\Delta} \tag{9.28}$$

so, in the case where there is no variance at frequencies higher than the Nyquist frequency, the underlying function can be recovered from the discrete series by using the formula

$$f(t) = \int_{-\infty}^{\infty} h(t')g(t - t')dt' \quad . \tag{9.29}$$

This is Whittacker's interpolation formula, often used for interpolation between values of a time series for this reason. When (9.24) is substituted in (9.29) the result is

$$f(t) = \sum_{n=-\infty}^{\infty} \frac{\sin[\pi(t - n\Delta T)/\Delta T]}{\pi(t - n\Delta T)/\Delta T} f(n\Delta T) \quad . \tag{9.30}$$

Because the terms decrease in magnitude as $1/n$, and oscillate in sign, the summation converges fast enough to be practical.

4). The cubic spline

A problem with polynomial interpolation is that higher-order polynomials sometimes produce undesirable fluctuations when the polynomials are forced to fit the data exactly. Small errors in the data can then have undesirable effects on interpolated values. The spline provides a technique for obtaining a smoother interpolation formula. A cubic spline $s(x)$ is constructed for each interval between data points by determining the four polynomial coefficients as follows. Two requirements are that the endpoints of the polynomial match the data:

$$s(x_i) = f(x_i) \quad \text{and} \quad s(x_{i+1}) = f(x_{i+1}) \ .$$

The two other constraints arise from the requirement that the first and second derivatives be the same as in adjoining intervals. (These constraints are shared with the nearby data intervals, so only provide two constraints.) It is conventional to specify that the second derivatives vanish at the endpoints of the data set. This then specifies a set of simultaneous equations to be solved for the interpolating function. Computer routines are readily available to perform these interpolations.

e. Roots of equations

Cases often arise where a desired solution can be specified by an equation, but the equation is difficult or impossible to solve analytically. In this section, some techniques for solution of such equations will be discussed. Three examples that have arisen in the author's recent research are:

1. Green (1975) gives an equation that relates the semi-major axis a of a raindrop to its volume-equivalent radius a_0. The equation, after some transformation, can be written as

$$a_0^2 - \frac{\sigma}{g\rho_w}\left[\left(\frac{a}{a_0}\right)^7 - 2\left(\frac{a}{a_0}\right)^2 + \frac{a}{a_0}\right] = 0 \ . \tag{9.31}$$

   To estimate rainrate or radar reflectivity, one needs to be able to find the volume-equivalent radius from a measurement of the semi-major axis. However, the equation involves a high-order polynomial and so is difficult to solve.

2. For reversible adiabatic ascent of a cloud parcel, the quantity Θ_q (the wet-equivalent potential temperature; cf. Paluch 1979) is conserved. This provides a basis for finding the temperature of a cloud parcel at some altitude above cloud base (specified by a pressure and by the requirement that the humidity correspond to water saturation in cloud) if the cloud-base pressure and temperature are known. From this one can find the liquid water content corresponding to adiabatic ascent to that level. This procedure requires solution of the equation

$$f(T) = \Theta_q(p,T) - \Theta_{q,base} = 0 \tag{9.32}$$

   where $\Theta_q(p,T)$ involves the pressure p and temperature T in factors that appear both multiplying and in the argument of an exponential expression:

$$XXX \ . \tag{9.33}$$

3. To construct skew-T or other thermodynamic diagrams, it is necessary to construct lines of constant Θ_e by finding the appropriate temperature corresponding to a given Θ_e and pressure:

$$f(T) = \Theta_e(T, p) - \Theta_{e,0} = 0. \tag{9.34}$$

In all these cases, the problems reduce to finding solutions to equations of the form $f(x) = 0$. Some methods for finding such solutions numerically are discussed in this section.

An invaluable aid in choosing an appropriate method of solution is to plot the function. This preliminary step will locate approximate roots and reveal the extent to which complicated procedures will be needed, e.g., to deal with multiple roots. It should almost always be the first step in numerical solution of such equations.

1). Newton's method

The first-order Taylor expansion of the equation $f(x) = 0$ is

$$f(x_0) + (x - x_0)f'(x_0) = 0 \tag{9.35}$$

An estimate of the root, x^*, can be found from values of the function and its derivative:

$$\hat{x}^* = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{9.36}$$

The formula can then be used iteratively to obtain improving estimates of the root:

$$\hat{x}^*_{n+1} = \hat{x}^*_n - \frac{f(\hat{x}^*_n)}{f'(\hat{x}^*_n)} \tag{9.37}$$

Figure 9.1 illustrates the basis for the method. Numerical versions of Newton's method are very convenient and simple to use. These usually use finite-difference formulas to evaluate the derivative.

Sometimes it is better to find roots of the function $f(x)/f'(x)$. This function will have roots at the same locations as $f(x)$, as long as the first derivative is finite, but the convergence will be faster for the case of multiple roots. To illustrate the problem, consider the equation

$$f(x) = (x - 5)^3 = 0$$

When Newton's method is used, convergence is very slow because the first derivative (which is in the denominator of Eq. (9.37)) approaches zero at the root. Starting at 10, the series is: 10, 8.333, 7.222, 6.481, 5.988, 5.658, 5.439, 5.292, 5.195, 5.130, ..., and after 20 terms is 5.002. If instead $f(x)/f'(x)$ is used in (9.44), the series reaches the correct answer exactly in one step.

As an example, consider the function (9.31) that describes the drop semi-major axis. A plot of this relationship, for $a = 0.3$, is shown in Fig. 9.2. The function is monotonic in the interval chosen, and well suited to solution by Newton's method. If the starting value is a_0=0.3, Newton's method leads to a value of a_0=0.26563 after 7 steps, and the steps change by less than 0.00001 after that step.

The disadvantages of Newton's method are that high-order roots can cause convergence to be slow, and the sequence may take undesirable jumps between roots or take a very large step upon encountering an inflection point.

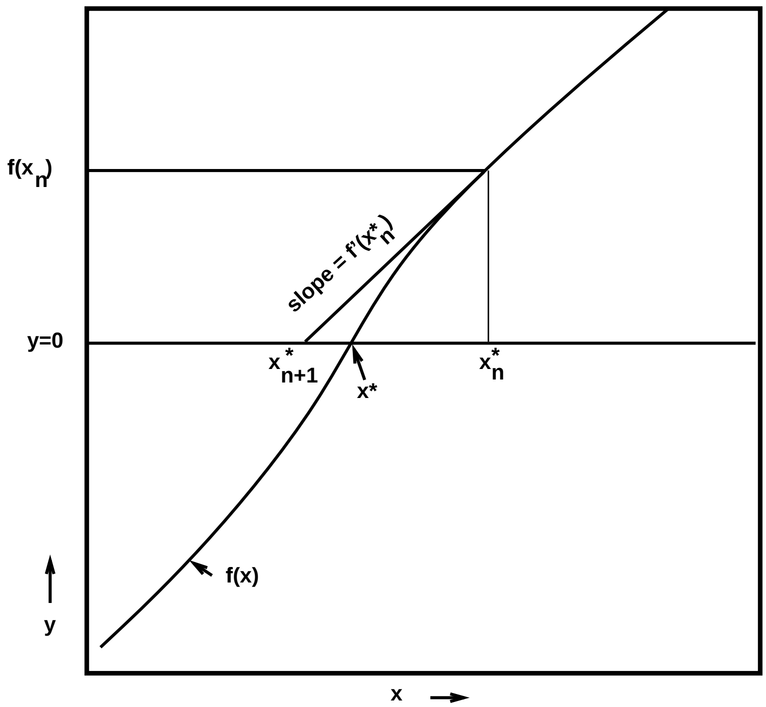

Fig. 9.1: A step in Newton's method. From an estimate x_n^* of the root of the equation $y = f(x) = 0$, construct an improved estimate of the root x_{n+1}^* by finding the intersection of the line $y = 0$ and the line through the point $(x,\ f(x))$ with slope $f'(x_n^*)$.

2). Interpolation

Roots of equations can also be found by interpolation if the identifications of dependent and independent variables are interchanged: From known values of the function, one interpolates to the point where the value of the *function*, rather than the independent variable, is zero. Lagrange interpolation is particularly suited to this approach because the values of the function will usually not form an evenly spaced array, and Lagrange interpolation is still convenient with unequal spacing between data values.

To illustrate the procedure, consider the equation

$$f(x) = x - e^{-x} = 0 \tag{9.38}$$

To start the problem, plot the function or make some guesses spanning the root:

x	$f(x)$
0.1	0.804837
0.5	0.1065306
0.7	-0.2034147
0.9	-0.4934303

The answer lies between 0.5 and 0.7. A fairly good estimate of the answer can be obtained by using $f(x)$ as the independent variable and interpolating among the tabulated values to

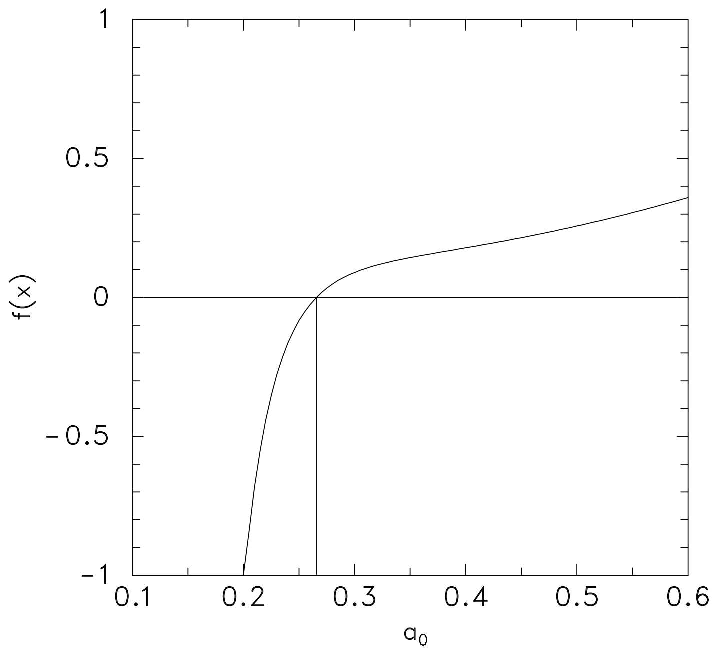

Fig. 9.2: Plot of Green's (1975) equation, (9.31), characterizing the drop semi-major axis a_0 in terms of the volume-equivalent radius a, for $a = 0.3$ cm. The solution to the equation corresponds to a_0=0.2656, as shown.

find x corresponding to $f(x) = 0$. Lagrange interpolation with the above points produces x=0.567, while the true root is 0.56714.... To improve the estimate, values at 0.555, 0.560, 0.565, 0.570, and 0.575 were taken; when the interpolation procedure was repeated, the result was 0.56714.

Similar accuracy can be reached via Newton's method in four steps. Lagrange interpolation can be very useful, however, when only tabular values are available. In the preceding example, the initial result was close to the true root even though the tabulated values were widely spaced about the root.

3). Other iterative procedures

Suppose that the equation to be solved, $f(x) = 0$, can be separated into two parts, one of which can be solved for x:

$$f(x) = f_1(x) + f_2(x) = 0 \tag{9.39}$$

where the equation $f_1(x) = -f_2(x)$ could be solved for specified $f_2(x)$. If the inverse of $f_1(x)$, $f_1^{-1}(x)$, is known, then the solution can be written as

$$x = g(x) = f_1^{-1}\Big(-f_2(x)\Big) \tag{9.40}$$

where the right side of this equation defines $g(x)$. In the case where $f_2(x)$ varies slowly with x, it may be possible to use (9.40) iteratively to find the root of (9.39).

Figures 9.3 illustrate that the root corresponds to the intersection of the two functions. As an example, consider the equation

$$x^2 = 0.01 - x^5 - x^7 \; . \tag{9.41}$$

If $f_2(x) = -0.01 + x^5 + x^7$ and $f_1(x) = x^2$, an iterative solution can be found from

$$x_{n+1} = g(x) = \sqrt{0.01 - x_n^5 - x_n^7} \; . \tag{9.42}$$

If the starting guess is 0, the iterative sequence is 0,0.1,0.09995,0.0999496, ..., so convergence is very rapid. This technique is often useful in cases where some part of the equation is complicated but introduces only a weak dependence.

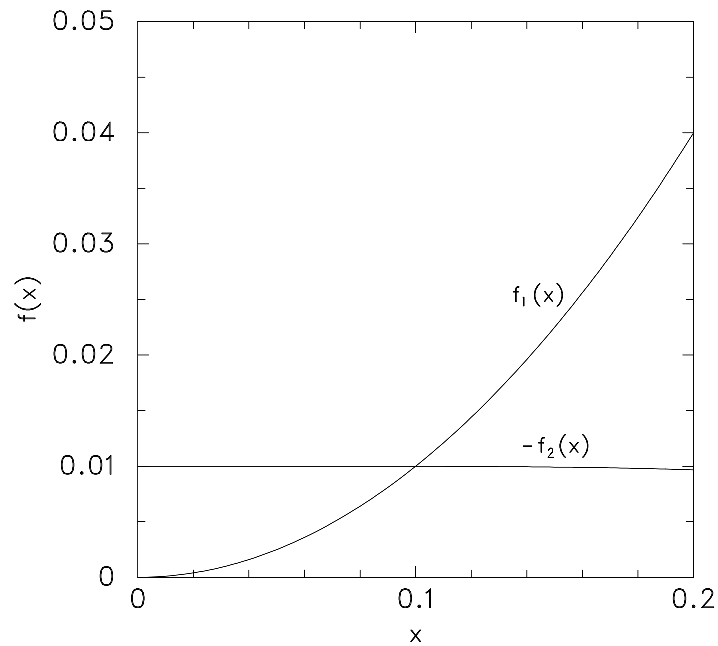

Fig. 9.3: The solution of the equation $f_1(x) + f_2(x)$ is the intersection of the two functions shown (here, for $f_1 = x^2$ and $f_2 = -0.01 + x^5 + x^7$).

The equation used by Green (1975), (9.31), provides a good example of the usefulness of this approach. It may be written as

$$a_0^3 = k\left[\left(\frac{a}{a_0}\right)^6 - 2\left(\frac{a}{a_0}\right) + 1\right] \tag{9.43}$$

where $k \approx ca$ where c=0.0765 cm^{-1}. The problem is to find a_0 given a. To obtain a form that will converge, rewrite the equation to use the term with largest derivative with respect to a_0 as $f_1(x)$:

$$\left(\frac{a_0}{a}\right)^6 = \left[2\left(\frac{a}{a_0}\right) - 1 + \frac{a_0^3}{k}\right]^{-1} \tag{9.44}$$

An iteration that leads rapidly to a solution is then

$$a_{0n+1} = a\left[\frac{a_{0n}^3}{k} + \frac{2a}{a_{0n}} - 1\right]^{-1/6} \tag{9.45}$$

For example, if a=0.3, the sequence for a_0 that starts with a as the first estimate is 0.3, 0.2635, 0.26566, 0.265632, ..., which converges even faster than Newton's method for this problem.

The examples of iterative sequences shown in Fig. 9.4 illustrate the necessity of isolating the slowest-varying component as function $f_1(x)$, so that the function $g(x)$ will have as small a derivative as possible. For $|dg(x)/dx| \geq 1$, the sequence does not converge.

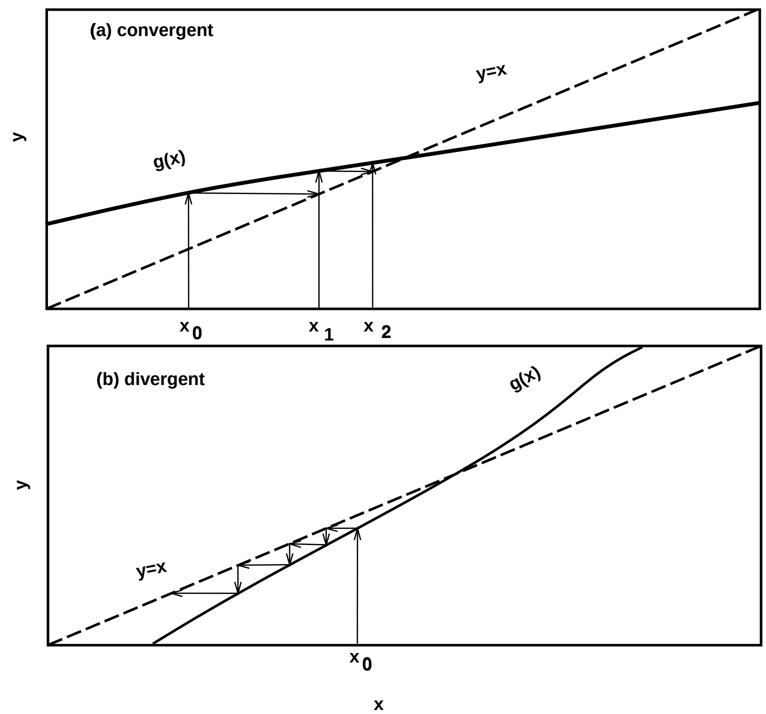

Fig. 9.4: Schematic representation of iterative sequences for a convergent case (a) and a divergent case (b). In each case, $x_{n+1} = g(x_n) = f_1^{-1}(-f_2(x_n))$ gives the sequence of estimates.

EXAMPLE 9.1: *The calculation of the wet-bulb temperature illustrates the use of these techniques. The defining equation for the wet-bulb temperature T_{wb} is*

$$f(T_{wb}) = T_{wb} - T - \frac{L_v}{C_p}\left(r - r_s(T_{wb})\right) = 0 \tag{9.46}$$

where T is the temperature, L_v the latent heat of vaporization, C_p the specific heat of air at constant pressure, r the water vapor mixing ratio, and $r_s(T_{wb})$ the saturation mixing ratio at the wet-bulb temperature. The saturation mixing ratio is a complicated function of temperature that is exponential even in the Clausius-Clapeyron approximation, and is better expressed by the more complicated Goff-Gratch formula (cf. List, 1958, p. 350), so this equation does not have a simple analytical solution.

Newton's method provides a good solution to (9.46) under most conditions, and is particularly fast at low humidity. At high humidity, the (approximately exponential) dependence of the saturation mixing ratio on wet-bulb temperature can introduce *instabilities. To increase the range over which the solution converges, it is sometime helpful to average consecutive estimates in an iterative procedure; for example, the series*

$$T_{wb,n+1} = \frac{T_{wb,n} + T + \frac{L_v}{C_p}[r - r_s(T_{wb,n})]}{2} \tag{9.47}$$

converges for a wider range of temperatures than does the corresponding series without averaging of consecutive terms.

EXERCISE 9.1: *Calculate the wet-bulb temperature as a function of dewpoint for a pressure of 800 mb and a temperature of 10°C.*

EXERCISE 9.2: *Calculate the liquid water content that would be produced in adiabatic ascent from a cloud base of 800 mb and 10°C to a final pressure of 500 mb.*

EXAMPLE 9.2: *Example for further discussion: calculate the threshold temperature for contrail formation at 400 mb, if the ratio of moisture to heat produced during combustion of jet fuel is 0.0295* $g_{H_2O} kg_{air}^{-1} (°C)^{-1}$.

---

f. Solution of simultaneous equations

1). Gauss-Seidel iteration

Simultaneous equations that are linear in the coefficients can be solved by the method of matrix inversion. The alternative of Gauss-Seidel iteration has two advantages:

- Errors do not accumulate during the calculation. If the procedure converges, it approaches the correct answer without rounding errors such as can occur during inversion of large matrices. As a result, very large sets of equations can be solved; the method has been applied to sets of thousands of equations.
- The method can be used for nonlinear sets of equations.

To apply the method, the set of equations must be written in a form so that each of the variables is given by one of the equations in the set. For example, the set might be written in the form

$$x_1 = f_1(x_2, x_3, \ldots)$$
$$x_2 = f_2(x_1, x_3, \ldots)$$
$$x_3 = f_3(x_1, x_2, x_4, \ldots)$$

etc. Then the sequence of equations is solved by using the values from all the preceding steps at each step in the solution, and the procedure is repeated iteratively.

An example where the method is effective is in the calculation of the lifted condensation level (LCL) of an unsaturated air parcel. This is needed, for example, in the calculation of equivalent potential temperature for an unsaturated air parcel. The LCL is the point at which the parcel, lifted adiabatically, reaches saturation with respect to water. During this ascent, potential temperature and mixing ratio remain constant, so finding the LCL is equivalent to finding the temperature and pressure for which a saturated air parcel would have the same potential temperature and mixing ratio as the observed parcel. If the observed temperature, pressure, and dewpoint are T, p, and T_d, and the temperature and pressure at the LCL are T_L and P_L, these conditions are equivalent to

$$p_L = p\left(\frac{T_L}{T}\right)^{c_p/R} \tag{9.48}$$

$$e_s(T_L) = e_s(T_d)\frac{p_L}{p} \tag{9.49}$$

If the last equation is solved for T_L (given p_L), for example by Lagrange interpolation, then this system of equations readily converges to the conditions at the LCL. The sequence for T=20°C, T_d=10°C, and p=800 mb is:

T_L	p_L
293.15	800.0
283.26	709.51
281.37	693.09
281.03	690.13
280.96	689.59
280.95	689.49
"	689.47
"	"

2). The Newton-Raphson method

The iterative method described earlier, Newton's method, can be extended to sets of simultaneous equations. In a single dimension, the slope was used as a first approximation to find the change needed to move to the root of the equation. With multiple dimensions, the analog to the slope is the multidimensional gradient, and the Newton-Raphson method consists of using the gradient to obtain successively better approximations.

The Taylor expansion of

$$f_i(x_1, x_2, x_3, \ldots) = 0 \tag{9.50}$$

can be written to first order using vector notation as

$$f_i(\mathbf{x}) = f_i(\mathbf{x}_0) = \Delta f_i \cdot (\mathbf{x} - \mathbf{x}_0). \tag{9.51}$$

An approximation to the equations to be solved is then, in matrix notation,

$$\mathbf{f} = -\mathbf{F}\delta \tag{9.52}$$

where $\mathbf{f}$ is the matrix of functions of the form (9.50) to be solved, evaluated at an estimated location for the root $\mathbf{x}_0$, $\mathbf{F}$ is the matrix of derivatives of the functions $\mathbf{f}$ with respect to the variables $\mathbf{x}$ for those values $\mathbf{x}_0$, and δ is the matrix of differences $(\mathbf{x} - \mathbf{x}_0)$ between the values that solve the equation and the current values.

The values of δ are estimates of the needed correction to an estimate of the solution, so the form

$$\delta = -\mathbf{F}^{-1}\mathbf{f} \tag{9.53}$$

will provide successively better estimates of the solution if applied iteratively. This method is readily applied to non-linear equations, and can use finite-difference estimates of the derivatives to evaluate the gradients.

g. Techniques for numerical integration

These techniques will not be covered here, but should be studied elsewhere because they are valuable tools in the analysis of data. Gaussian quadrature is particularly powerful and is used frequently. The Runge-Kutta method and predictor-corrector schemes are among the most important methods for numerical integration of differential equations. These are standard topics in books on numerical methods. For general numerical integration, the author has found the Cash-Karp algorithm described on p. 719 of Press et al. (1992) to be particularly efficient.

SOURCES AND FURTHER READING

Abramowitz, M. and I. A. Stegun, 1972: Handbook of Mathematical Functions. Dover Publications, New York, 1046 pp.

Green, A. W., 1975: An approximation for the shapes of large raindrops. *J. Appl. Meteor.,* **14,** 1578–1583.

Hornbeck, R. W., 1975: *Numerical Methods.* Quantum Publishers, Inc., New York, 310 pp.

Paluch, I. R., 1979: The entrainment mechanism in Colorado cumuli. *J. Atmos. Sci.,* **36,** 2467–2478.

page intentionally left blank