7 ODE Boundary Value Problems

In this chapter, we consider *two-point boundary value problems* (BVPs) for *ordinary differential equations* (ODEs)

$$y' = f(y)\ ,\quad f \in C^2\ ,\quad r(y(a), y(b)) = 0\ ,\quad r \in C^2\ ,$$

wherein both the right side f (autonomous for ease of writing) and the boundary conditions r are of dimension n and may be nonlinear. Algorithms for the solution of such problems can be grouped in two classes: initial value methods and global discretization methods. The presentation and notation here closely relates to Chapter 8 in the textbook [71].

Initial value methods. This kind of methods transforms the BVP into a sequence of *initial value problems* (IVPs), which are solved by means of numerical integrators. The most prominent method of this type is the *multiple shooting method*, which is a good choice only for problems, wherein a well-conditioned IVP direction exists, i.e. for so-called *timelike* BVPs. The name comes from the fact that in this problem class the independent variable t typically represents a time (or time related) variable. As a rule, there exists no generalization to boundary value problems for partial differential equations (PDEs).

Global discretization methods. Conceptually, this kind of BVP methods does not depend on any preferable direction and is therefore also applicable to cases, where a well-conditioned IVP direction does not exist, i.e. to so-called *spacelike* boundary value problems. In this type of BVP the independent variable t typically represents a space (or space related) variable, which implies that a generalization to BVPs for PDEs is possible. Such methods include, e.g., *finite difference* and *collocation methods*.

In Section 7.1, the realization of Newton and discrete continuation methods within the standard *multiple shooting* approach is elaborated. Gauss-Newton methods for parameter identification in ODEs are discussed in Section 7.2. For periodic orbit computation, Section 7.3 presents Gauss-Newton methods, both in the shooting approach (Sections 7.3.1 and 7.3.2) and in a collocation approach based on Fourier series (Galerkin-Urabe method in Section 7.3.3).

P. Deuflhard, *Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms*, Springer Series in Computational Mathematics 35,
DOI 10.1007/978-3-642-23899-4_7, © Springer-Verlag Berlin Heidelberg 2011

In Section 7.4 we concentrate on polynomial *collocation methods*, which have reached a rather mature status including affine covariant Newton methods. In Section 7.4.1, the possible discrepancy between discrete and continuous solutions is studied including the possible occurrence of so-called 'ghost solutions' in the nonlinear case. On this basis, the realization of *quasilinearization* seems to be preferable in combination with collocation. The following Section 7.4.2 is then devoted to the key issue that quasilinearization can be interpreted as an *inexact Newton method in function space*: the approximation errors in the infinite dimensional setting just replace the inner iteration errors arising in the finite dimensional setting. With this insight, an adaptive multilevel control of the collocation errors can be realized to yield an adaptive inexact Newton method in function space—which is the bridge to adaptive Newton multilevel methods for PDEs (compare Section 8.3).

Bibliographical Note. Affine invariant global Newton methods—now called *affine covariant* Newton methods—have first been developed in the frame of multiple shooting techniques by P. Deuflhard [60, 62, 61]. Therein they have turned out to be of crucial importance for the overall performance, especially in challenging real life optimal control problems. These Newton techniques have then quickly been adopted by U.M. Ascher and R.D. Russell within their adaptive collocation methods [8]—with comparable success, see also their textbook [9]. They have also played an important role within parameter identification algorithms and their convergence analysis as worked out by H.G. Bock [29, 31, 32] since 1981.

7.1 Multiple Shooting for Timelike BVPs

In this approach the interval $[a, b]$ is subdivided into a partition

$$\Delta = \{a = t_1 < t_2 < \cdots < t_m = b\}, \quad m > 2\,.$$

Let $x_j \in \mathbb{R}^n$, $j = 1, \ldots, m$ denote estimates of the unknown values at the nodes t_j. Then, in terms of the flow Φ, we may define those $m - 1$ sub-trajectories

$$y_j(t) = \Phi^{t,t_j} x_j\,, \quad t \in [t_j, t_{j+1}]\,, \quad j = 1, \ldots, m-1$$

that solve $(m-1)$ independent IVPs. The situation is illustrated in Figure 7.1. For the solution of the problem the sub-trajectories have to be joined continuously and hence at the intermediate nodes the n *continuity conditions*

$$F_j(x_j, x_{j+1}) = \Phi^{t_{j+1},t_j} x_j - x_{j+1} = 0, \quad j = 1, \ldots, m-1$$

have to hold.

In addition, we have to satisfy the n *boundary conditions*

$$F_m(x_1, x_m) = r(x_1, x_m) = 0.$$

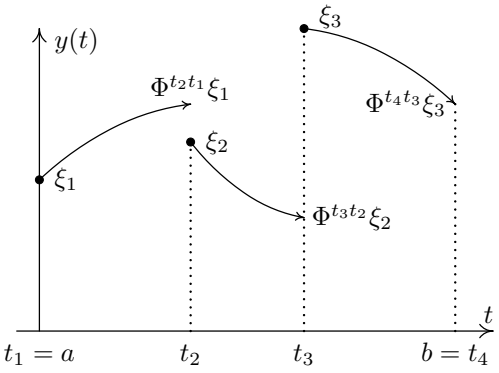

Fig. 7.1. Multiple shooting ($m = 4$).

The overall full nm-dimensional system is written in the form

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} \in \mathbb{R}^{n \cdot m}, \quad F(x) = \begin{pmatrix} F_1(x_1, x_2) \\ \vdots \\ F_m(x_1, x_m) \end{pmatrix} = 0. \tag{7.1}$$

This nonlinear system has a *cyclic* block structure as indicated in Figure 7.2. For the solution of the above cyclic nonlinear system (7.1) we compute the *ordinary Newton correction* as usual by solving the linear system

$$F'(x^k)\Delta x^k = -F(x^k)\,, \quad x^{k+1} = x^k + \Delta x^k\,, \quad k = 0, 1, \ldots\,.$$

The corresponding Jacobian matrix has the cyclic block structure

$$J = F'(x) = \begin{bmatrix} G_1 & -I & & \\ & \ddots & \ddots & \\ & & G_{m-1} & -I \\ A & & & B \end{bmatrix}.$$

Herein the matrices A, B are the derivatives of the boundary conditions r with respect to the boundary values $(x(a), x(b)) = (x_1, x_m)$. The propagation matrices G_j on each of the sub-intervals, also called Wronskian matrices, read

$$G_j = \frac{\partial \Phi^{t_{j+1}, t_j} x_j}{\partial x_j}\,, \quad j = 1, \ldots, m-1\,.$$

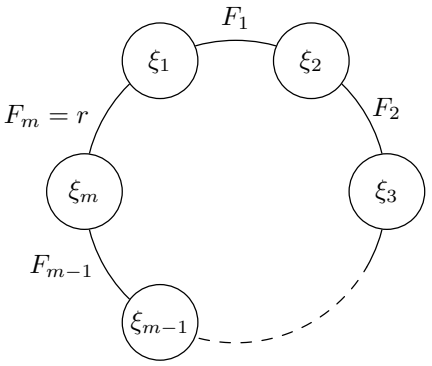

Fig. 7.2. Cyclic system of nonlinear equations

7.1.1 Cyclic linear systems

The block structure of the Jacobian matrix gives rise to a *block cyclic linear system* of the following kind:

$$\begin{array}{llllll} G_1\Delta x_1 & -\Delta x_2 & & & & = -F_1 \\ & \ddots & \ddots & & & \\ & & G_{m-1}\Delta x_{m-1} & -\Delta x_m & & = -F_{m-1} \\ A\Delta x_1 & & & +B\Delta x_m & & = -F_m = -r\,. \end{array}$$

If this linear system were just solved by some (sparse) direct elimination method, then global Newton methods as described in the preceding sections could be directly taken off the shelf.

For timelike BVPs, however, there exists an efficient alternative option, which opens the door to the construction of interesting specific Gauss-Newton methods. This option dates back to a suggestion of J. Stoer and R. Bulirsch [187]. It is often called *condensing* algorithm, since it requires only the decomposition of a 'condensed' (n, n)-matrix E instead of that of the total Jacobian (nm, nm)-matrix J. In order to convey the idea, we present the idea first for the case $m = 3$:

$$\begin{array}{llllll} (1) & G_1\Delta x_1 & -\Delta x_2 & & = & -F_1 \\ (2) & & G_2\Delta x_2 & -\Delta x_3 & = & -F_2 \\ (3) & A\Delta x_1 & & +B\Delta x_3 & = & -r\,. \end{array}$$

First we multiply (1) by G_2 from the left and add the result

$$G_2G_1\Delta x_1 - G_2\Delta x_2 = -G_2F_1$$

to equation (2). This gives

$$G_2G_1\Delta x_1 - \Delta x_3 = -(F_2 - G_2F_1)\,,$$

which after multiplication by B yields

$$BG_2G_1\Delta x_1 - B\Delta x_3 = -B(f_2 - G_2F_1).$$

Finally, by addition of equation (3) it follows that

$$\underbrace{(A + BG_2G_1)}_{=E}\,\Delta x_1 = \underbrace{-r - B(F_2 - G_2F_1)}_{=-u}\,.$$

Hence, in the general case $m \geq 2$ we obtain the following algorithm:

a) Evaluate by recursion over $j = 1, \ldots, m-1$

$E := A + BG_{m-1}\cdots G_1,$

$u := r + B\left[F_{m-1} + G_{m-1}F_{m-2} + \cdots + G_{m-1}\ldots G_2F_1\right].$

b) Solve the linear (n,n) − system (7.2)

$E\Delta x_1 = -u.$

c) Execute the explicit recursion

$\Delta x_{j+1} := G_j\Delta x_j + F_j, \quad j = 1, \ldots, m-1.$

The memory required by this algorithm is essentially $m \cdot n^2$. The computational cost is dominated by the accumulation of the matrix E as an $(m-1)$-fold product of (n,n)-matrices. Together with the decomposition of E this results in a cost of $O(m \cdot n^3)$ operations, where terms of order $O(n^2)$ have been neglected as usual.

The large sparse Jacobian matrix J and the small matrix E are closely connected as can be seen by the following lemma.

Lemma 7.1 *Notation as just introduced. Define* $W_j = G_{m-1}\cdots G_j$ *and* $E := A + BW_1$. *Then*

$$\det(J) = \det(E)\,. \tag{7.3}$$

Moreover, if E *is nonsingular, one has the decomposition*

$$LJR = S, \quad J^{-1} = RS^{-1}L \tag{7.4}$$

in terms of the block matrices

$$L := \begin{bmatrix} BW_2 \dots & B, & I \\ -I & & \\ \ddots & & \\ & -I, & 0 \end{bmatrix}, \qquad R^{-1} := \begin{bmatrix} I & & \\ -G_1, I & & \\ & \ddots\ddots & \\ & -G_{m-1}, I & \end{bmatrix},$$

$$S := \mathrm{diag}(E, I, \dots, I), \qquad S^{-1} = \mathrm{diag}(E^{-1}, I, \dots, I)\,.$$

Proof. The decomposition (7.4) is a direct formalization of the above block Gaussian elimination. The determinant relation (7.3) follows from

$$\det(L) = \det(R) = \det(R^{-1}) = 1\,.$$

With

$$\det(J) = \det(S)$$

the proof is completed. □

Interpretation. The matrix E is an approximation of the special *sensitivity matrix*

$$E(a) = \frac{\partial r}{\partial y_a} = A + BW(b, a)$$

corresponding to the BVP as a whole. Herein $W(\cdot, \cdot)$ denotes the propagation matrix of the variational equation. Generically this means that, whenever the underlying BVP has a locally uniqueness solution, a locally unique solution $x^* = (x_1^*, \dots, x_m^*)$ is guaranteed—independent of the partitioning Δ.

Separable linear boundary conditions. This case arises when part of the boundary conditions fix part of the components of x_1 at $t = a$ and part of the components of x_m at $t = b$. The situation can be conveniently described in terms of certain projection matrices P_a, $\bar{P}_a$, P_b, $\bar{P}_b$ such that

$$\begin{aligned} &\bar{P}_a A = P_a, \quad \bar{P}_a B = 0\,, \\ &\mathrm{rank}(\bar{P}_a) = \mathrm{rank}(P_a) = n_a < n\,, \end{aligned}$$

$$\begin{aligned} &\bar{P}_b B = P_b, \quad \bar{P}_b A = 0\,, \\ &\mathrm{rank}(\bar{P}_b) = \mathrm{rank}(P_b) = n_b < n \end{aligned}$$

with $n_a + n_b \le n$. Of course, we will choose initial guesses x_1^0, x_m^0 for the Newton iteration so that the separable boundary conditions

$$\bar{P}_a r = 0\,, \quad \bar{P}_b r = 0$$

automatically hold. Then the linearization of these conditions

$$A\Delta x_1 + B\Delta x_m = -r$$

directly implies

$$P_a\Delta x_1 = 0\,, \quad P_b\Delta x_m = 0\,.$$

Consequently, the variables $P_a x_1$ and $P_b x_m$ can be seen to satisfy

$$P_a x_1 = P_a x_1^0\,, \quad P_b x_m = P_b x_m^0$$

throughout the iteration. This part can be realized independent of any elimination method by carefully analyzing the sparsity pattern of the matrices A and B within the algorithm. As a consequence, the sensitivity matrix E also has the projection properties

$$\bar{P}_a E = 0, \quad \bar{P}_b E = 0, \qquad E P_a = 0, \quad E P_b = 0\,.$$

Iterative refinement sweeps. The block Gaussian elimination technique (7.2) seems to be highly efficient in terms of memory and computational cost. A closer look on its numerical stability, however, shows that the method becomes sufficiently robust only with the addition of some special iterative refinement called iterative refinement sweeps. We will briefly sketch this technique and work out its consequences for the construction of Newton and Gauss-Newton methods—for details see the original paper [70] by P. Deuflhard and G. Bader.

Let $\nu = 0, 1, \ldots$ be the indices of the iterative refinement steps. In lieu of the exact Newton corrections Δx_j the block Gaussian elimination will supply certain error carrying corrections $\Delta\tilde{x}_j^\nu$ so that iterative refinement will produce nonvanishing differences

$$dx_j^\nu \approx \Delta\tilde{x}_j^{\nu+1} - \Delta\tilde{x}_j^\nu\,.$$

In the present framework we might first consider the following algorithm:

(a) $du^\nu = dr^\nu + B\left[dF_{m-1}^\nu + G_{m-1}dF_{m-2}^\nu + \cdots + G_{m-1}\cdots G_2 dF_1^\nu\right]$,

(b) $E dx_1^\nu = -du^\nu$,

(c) $dx_{j+1}^\nu = G_j dx_j^\nu + dF_j^\nu \quad j = 1, \ldots, m.$

However, as shown by the detailed componentwise round-off error analysis in [70], this type of iterative refinement is only guaranteed to converge under the sufficient condition

$$\varepsilon(m-1)(2n+m-1)\kappa[a,b] \ll 1\,,$$

wherein ε denotes the relative machine precision and $\kappa[a,b]$ the IVP condition number over the whole interval $[a,b]$—to be associated with single instead

of multiple shooting. This too restrictive error growth can be avoided by a modification called *iterative refinement sweeps*. As before, this modification also begins with an implementation of iterative refinement for the 'condensed' linear system

$$E\Delta\tilde{x}_1 + u \approx 0\,.$$

Suppose, for the time being, that

$$\|d\tilde{x}_1\| \leq \text{eps},$$

where eps is the relative tolerance prescribed for the Newton iteration. Then some *sweep-index* $j_\nu \geq 1$ can be defined such that

$$\|d\tilde{x}_j^\nu\| \leq \text{eps}, \quad j = 1, \ldots, j_\nu.$$

If we now set part of the residuals deliberately to machine-zero, say,

$$dF_j^\nu = 0, \quad j = 1, \ldots, j_\nu - 1\,,$$

then this modified iterative refinements process can be shown to converge under the less restrictive sufficient condition

$$\varepsilon(m-1)(2n+m-1)\kappa_\Delta[a,b] < 1\,,$$

wherein now the quantity $\kappa_\Delta[a,b]$ enters, which denotes the maximum of the IVP condition numbers on each of the subintervals of the partitioning Δ. Obviously, this quantity reflects the IVP condition number to be naturally associated with multiple shooting. Under this condition it can be shown that

$$j_{\nu+1} \geq j_\nu + 1\,,$$

hence the process terminates, at the latest, after $m-1$ refinement sweeps.

Whenever the above excluded case $j_0 = 0$ occurs, the iterative refinement cannot even start. This occurrence does not necessarily imply that the BVP as such is ill-conditioned—for a detailed discussion of this aspect see again the textbook [71].

Rank reduction. The iterative refinement sweeps cheaply supply a condition number estimate for the sensitivity matrix E via

$$\text{cd}(E) = \frac{\|d\tilde{x}_1^0\|}{\|\Delta\tilde{x}_1^0\|\varepsilon} \leq \text{cond}(E)\,.$$

Even without iterative refinement sweeps a cheap condition number estimate may be available: Assume that separable boundary conditions have been split off via the above described projection. Let E denote the remaining part of the sensitivity matrix which is then treated by QR-decomposition with

column pivoting—for details see, e.g., [77, Section 3.2.2]. In this setting the *subcondition number*

$$\mathrm{sc}(E) \leq \mathrm{cond}(E)$$

is easily computable.

If either

$$\varepsilon\,\mathrm{cd}(E) \geq \tfrac{1}{2} \quad \Longleftrightarrow \quad \|d\tilde{x}_1^0\| \geq \tfrac{1}{2}\|\Delta\tilde{x}_1^0\|,$$

which is equivalent to $j_0 = 0$ or

$$\varepsilon\,\mathrm{sc}(E) \geq \tfrac{1}{2}\,,$$

then

$$\varepsilon\,\mathrm{cond}(E) \geq \tfrac{1}{2}\,.$$

In other words: in either case E is *rank-deficient* and the condensed system is ill-conditioned. In this situation we may replace the condensed equation

$$E\Delta x_1 = -u$$

by the *underdetermined* linear least squares problem

$$\|E\Delta x_1 + u\|_2 = \min\,.$$

This linear system may be 'solved' by means of the Moore-Penrose pseudo-inverse as

$$\Delta x_1 = -E^+ u\,.$$

Upon leaving the remaining part of the condensing algorithm unaltered, the thus modified elimination process can be formally described by some generalized inverse

$$J^- = RS^+L \tag{7.5}$$

with R,S,L as defined in Lemma 7.1 and

$$S^+ = \begin{bmatrix} E^+ & & & \\ & I & & \\ & & \ddots & \\ & & & I \end{bmatrix}.$$

As can be easily verified (see Exercise 4.7), this generalized inverse is an *outer inverse* and can be uniquely defined by the set of four axioms

$$\begin{aligned} (J^-J)^T &= (RR^T)^{-1}J^-J(RR^T)\,, \\ (JJ^-)^T &= (L^TL)JJ^-(L^TL)^{-1}\,, \\ J^-JJ^- &= J^-\,, \\ JJ^-J &= J\,. \end{aligned} \tag{7.6}$$

This type of generalized inverse plays a role in a variety of more general BVPs, some of which are given in the subsequent Sections 7.2 and 7.3.

7.1.2 Realization of Newton methods

On the basis of the preceding sections we are now ready to discuss the actual realization of global Newton methods within multiple shooting techniques for timelike BVPs.

Jacobian matrix approximations. In order to establish the total Jacobian J, we must approximate the boundary derivative matrices A, B and the propagation matrices $G_1, \ldots, G_{m-1}$.

Boundary derivatives. Either an analytic derivation of r (not too rare case) or a *finite difference approximation*

$$A \doteq \frac{\delta r}{\delta x_1} \ , \quad B \doteq \frac{\delta r}{\delta x_m}$$

will be realized.

Propagation matrices. The propagation matrices G_j are also called Wronskian matrices. Whenever the derivative matrix $f_y(y)$ of the right side is analytically available, then numerical integration of the n variational equations

$$G_j' = f_y(y(t))G_j \ , \quad G_j(t_j) = I_n \tag{7.7}$$

might be the method of choice to compute them. If f_y is not available analytically, then some *internal* differentiation as suggested by H.G. Bock [31, 32] should be applied—see also [71, Section 8.2.1]. Its essence is a numerical differencing of the form

$$f_y(y) \doteq \frac{\delta f(y)}{\delta y} \ ,$$

which then enters into the numerical solution of *discrete variational equations* instead of (7.7). The actual realization of this idea requires special variants of standard integration software [112, 31, 32]. Note that any such approach involves, of course, the simultaneous numerical integration of $y' = f(y(t))$ to obtain the argument $y(t)$ in f_y.

Scaling. Formally speaking, each variable x_j will be transformed as

$$x_j \longrightarrow D_j^{-1} x_j \ ,$$

wherein the diagonal matrices $D_j > 0$ represent some carefully chosen scaling. Formal consequences are then

$$\begin{aligned} F_j &\longrightarrow D_{j+1}^{-1} F_j \ , \\ G_j &\longrightarrow D_{j+1}^{-1} G_j D_j =: \hat{G}_j \ . \end{aligned} \tag{7.8}$$

In actual computation, this means replacing

$$\begin{aligned} \|F_j\| &\longrightarrow \|D_{j+1}^{-1} F_j\| \ , \\ \|\Delta x_j\| &\longrightarrow \|D_j^{-1} \Delta x_j\| \ . \end{aligned}$$

Inner products and norms. In view of the underlying BVP we may want to modify the Euclidean inner product and norm by including information about the mesh $\Delta = \{t_1, \ldots, t_m\}$. For example, let $(\cdot, \cdot)$ denote some (possibly scaled) Euclidean inner product for the vectors $u = (u_1, \ldots, u_m)$, $v = (u_1, \ldots, v_m)$, $u_j, v_j \in \mathbb{R}^n$. Then we may define some *discrete L^2-product* by virtue of

$$\begin{aligned} |b-a|(u,v)_\Delta &= (u_1,v_1)|t_2-t_1| + (u_m,v_m)|t_m - t_{m-1}| \\ &\quad + \textstyle\sum_{j=2}^{m-1}(u_j,v_j)|t_{j+1}-t_{j-1}| \end{aligned} \tag{7.9}$$

and its induced *discrete L^2-norm* as

$$(u,u)_\Delta \equiv \|u\|_\Delta^2 .$$

Quasi-Newton updates. Any approximation of the Wronskian matrices G_j requires a computational cost of $\sim n$ trajectory evaluations. In order to save computing time per Wronskian evaluation, we may apply rank-1 updates as long as the iterates remain within the Kantorovich domain around the solution point—i.e., when the damping strategies in Section 2.1.4 supply

$$\lambda_k = \lambda_{k-1} = 1 .$$

Of course, the sparse structure of the total Jacobian J must be taken into account. We assume the boundary derivative approximations A and B as fixed. Then the *secant condition* (1.17) for the total Jacobian

$$(J_{k+1} - J_k)\Delta x^k = F(x^{k+1})$$

splits into the separate block secant conditions

$$(G_j^{k+1} - G_j^k)\Delta x_j^k = F_j(x_j^{k+1}, x_{j+1}^{k+1}) , \quad j = 1, \ldots, m-1 .$$

Upon applying the ideas of Section 2.1.4, we arrive at the following rank-1 update formula:

$$G_j^{k+1} = G_j^k + F_j(x_j^{k+1}, x_{j+1}^{k+1}) \frac{(\Delta x_j^k)^T}{\|\Delta x_j^k\|_2^2} , \quad j = 1, \ldots, m-1 . \tag{7.10}$$

In a *scaled* version of the update formula (7.10), we will either update the $\hat{G}_j$ from (7.8) directly or, equivalently, update G_j replacing

$$\frac{\Delta x_j^T}{\|\Delta x_j\|_2^2} \longrightarrow \frac{(D_j^{-2}\Delta x_j)^T}{\|D_j^{-1}\Delta x_j\|_2^2}$$

in the representation (7.10). As worked out in detail in Section 2.1.4 above, scaling definitely influences the convergence of the corresponding quasi-Newton iteration (compare also [59, Section 4.2]).

Adaptive rank strategy. Assume that the condensed matrix E has been indicated as being 'rank-deficient'. In this case we need not terminate the Newton iteration, but may continue by an intermediate Gauss-Newton step with a correction of the form

$$\Delta x^k = -F'(x^k)^- F(x^k)\,.$$

Upon recalling Section 4.1.1, the generalized inverse $F'(x)^-$ can be seen to be an *outer* inverse. Therefore Theorem 4.7 guarantees that the thus defined ordinary Gauss-Newton iteration converges locally to a solution of the system

$$\begin{aligned} &F_j(x_j, x_{j+1}) = 0\,, \quad j = 1, \ldots, m-1\,, \\ &\|r(x_1, x_m)\|_2 = \min\,. \end{aligned} \tag{7.11}$$

The modified trust region strategies of Section 4.3.5 can be adapted for the special projector

$$P := J^- J\,.$$

Note, however, that intermediate rank reductions in this context will *not* guarantee an increase of the feasible damping factors (compare Lemma 4.17 or Lemma 4.18), since P is generically *not* orthogonal (for $m > 2$). Nevertheless a significant increase of the damping factors has been observed in numerical experiments.

Obviously, the thus constructed Gauss-Newton method is associated with an underdetermined *least squares* BVP of the kind

$$\begin{aligned} &y' = f(y)\,, \\ &\|r(y(a), y(b))\|_2 = \min\,. \end{aligned}$$

Level functions. In the *rank-deficient* case, the *residual* level function

$$T(x|I) = \|F(x)\|^2 = \sum_{j=1}^{m} \|F_j(x)\|^2$$

no longer has the Gauss-Newton correction $\Delta x = -F'(x)^- F(x)$ as a descent direction. Among the practically interesting level functions, this property still holds for the above used *natural* level function $T(x|J^-)$ or for the *hybrid* level function

$$T(x|R^{-1}J^-) = \|R^{-1}J^- F(x)\|^2 = \|\Delta x_1\|^2 + \sum_{j=1}^{m-1} \|F_j(x)\|^2\,.$$

The proof of these statements is left as Exercise 7.4.

Bibliographical Note. The affine covariant Newton method as described here is realized, e.g., in the multiple shooting code `BVPSOL` due to P. Deuflhard and G. Bader [70] and the optimal control code `BOUNDSCO` due to J. Oberle [162]. Among these only `BVPSOL` realizes the Gaussian block elimination (Section 7.1.1) including the rank-deficient option with possible intermediate Gauss-Newton steps. Global sparse solution of the cyclic linear Newton systems is implemented in the code `BOUNDSCO` and as one of two options in `BVPSOL`; a rank-strategy is not incorporated within global elimination.

7.1.3 Realization of continuation methods

Throughout this section we consider parameter dependent two-point boundary value problems of the kind

$$\begin{aligned} y' &= f(y,\tau)\,, \\ r(y(a),y(b),\tau) &= 0\,, \end{aligned}$$

which give rise to some parameter dependent cyclic system of nonlinear equations

$$F(x,\tau) = 0\,.$$

Typical situations are that either the τ-family of BVP solutions needs to be studied or a continuation method is applied to globalize a local Newton method. Generally speaking, the parameter dependent mapping F is exactly the case treated in Section 5. Hence, any of the continuation methods described there can be transferred—including the automatic control of the parameter stepsizes $\Delta\tau$.

Newton continuation methods. Assume the BVP under consideration has no turning or bifurcation points—known either from external insight into the given scientific problem or from an a-priori analysis. Then Newton continuation methods as presented in Section 5.1 are applicable.

Classical continuation method. This algorithm (of order $p = 1$) deserves no further explanation. All the details of Section 5.1 carry over immediately.

Tangent continuation method. For this algorithm (of order $p = 2$) we need to solve the linear system

$$F_x(x,\tau)\dot{\tilde{x}}(\tau) = F_\tau(x,\tau)\,,$$

which is the same type of block cyclic linear system as for the Newton corrections. The above right hand term $F_\tau(x,\tau)$ can be computed by numerical integration of the associated variational equations or by *internal* numerical differentiation (cf. [31, 32] or [71, Section 8.2.1]).

Continuation via trivial BVP extension. A rather popular trick is to just extend the standard BVP such that

$$\begin{aligned} y' &= f(y,\tau), \quad \tau' = 0\,, \\ r(y(0),y(T),\tau) &= 0, \quad h(\tau) = 0 \end{aligned} \tag{7.12}$$

with $h'(\tau) \neq 0$. Any BVP solver applied to this extended BVP then defines some extended mapping

$$F(x,\tau) = 0, \quad h(\tau) = 0\,. \tag{7.13}$$

Let Δx^k denote the Newton correction for the equations with fixed τ. Then the Newton correction $(\Delta z^k, \Delta\tau)$ for (7.13) turns out to be

$$\Delta z^k = \Delta x^k + \Delta\tau^k \dot{\bar{x}}(\tau^k), \quad \Delta\tau^k = -\frac{h(\tau^k)}{h'(\tau^k)}\,. \tag{7.14}$$

The proof of this connection is left as Exercise 7.1. If one selects τ^0 such that $h(\tau^0) \neq 0$, then the extension (7.12) realizes a mixture of continuation methods of order $p = 1$ and $p = 2$. An adaptive control of the stepsizes $\Delta\tau$ here arises indirectly via the damping strategy of Newton's method.

Example 7.1 *Space shuttle problem.* This optimal control problem stands for a class of highly sensitive BVPs from space flight engineering. The underlying physical model (very close to realistic) is due to E.D. Dickmanns [87]. The full mathematical model has been documented in [81]. The stated mathematical problem is to find an optimal trajectory of the second stage of a Space Shuttle such that a prescribed maximum permitted skin temperature of the front shield is not exceeded. The real problem of interest is a study with respect to the temperature parameter, say τ. For technological reasons, the aim is to drive down the temperature as far as possible. This problem gave rise to a well-documented success story for error oriented Newton methods (earlier called affine 'invariant' instead of affine covariant).

The unconstrained trajectory goes with a temperature level of $2850°F$ (equivalent to $\tau = 0.072$). The original technological objective of NASA had been optimal flight trajectories at temperature level $2000°F$ (equivalent to $\tau = 0.$). However, the applied continuation methods just failed to continue to temperatures lower than $2850°F$! One reason for that failure can already be seen in the sensitivity matrix: the early optimal control code `OPTSOL` of R. Bulirsch [42], improved 1972 by P. Deuflhard [59] (essentially in the direction of error oriented Newton methods), revealed a subcondition number

$$\mathrm{sc}(E) = 0.2 \cdot 10^{10}$$

at that temperature. As a consequence, any traditional residual based Newton methods, which had actually been used at NASA within the frame of classical

continuation, are bound to fail. The reasons for such an expectation have been discussed in Sections 3.3.1 and 3.3.2.

In 1973, H.-J. Pesch attacked this problem by means of `OPTSOL`, which in those days still contained classical continuation with *empirical* stepsize control, but already error oriented Newton methods [59] with *empirical* damping strategy. With these techniques at hand, the first successful continuation steps to $\tau < 0.072$ were at all possible—nicely illustrating the geometric insight from Section 3.3.2. However, computing times had been above any tolerable level, so that H.-J. Pesch eventually terminated the continuation process at $\tau = 0.0080$ with a final empirical stepsize $\Delta\tau = -0.0005$. Further improvements were possible by replacing

- the classical Newton damping strategy by an adaptive one [63], similar to the adaptive trust region predictor given in Section 3.3.3 and
- the classical continuation method with empirical stepsize selection by their adaptive counterparts as presented in Section 5.1.

For the last continuation step performed by H.-J. Pesch, Table 7.1 shows the comparative computational amount for different continuation methods (counting full trajectories to be computed within the multiple shooting approach).

Continuation method	*Newton method*	*work*
classical	residual based	failure
classical	error oriented, empirical damping	~ 340
classical	error oriented, adaptive trust region	114
trivial BVP extension	error oriented, adaptive trust region	48
tangent	error oriented, adaptive trust region	18

Table 7.1. Space Shuttle problem: Fixed continuation step from $\tau = 0.0085$ to $\tau = 0.0080$. Comparative computational amount for different Newton continuation methods.

In 1975, an adaptive error oriented Newton method [81] in connection with the trivial BVP extension made it, for the first time, possible to solve the original NASA problem for temperature level $2000°F$ ($\tau = 0.$). Results of technical interest have been published by E.D. Dickmanns and H.-J. Pesch [88]. The performance of this computational technique for temperatures even below the NASA objective value is documented in Table 7.2. As can be seen,

this kind of continuation technique, even though it succeeds to solve the problem, still performs a bit rough.

Continuation sequence			*work*		*Remarks*
0.0	→	-0.0050	60	fail	switching structure totally disturbed
0.0	→	-0.0010	60	fail	negative argument in log-function
0.0	→	-0.0005	80		
-0.0005	→	-0.0010	63		
-0.0010	→	-0.0020	50		
-0.0020	→	-0.0050	65		
-0.0050	→	-0.0200	30	fail	switching structure totally disturbed
-0.0050	→	-0.0100	30	fail	as above
-0.0050	→	-0.0100	67	fail	Newton method fails to converge
0.0050	→	-0.0080	66		prescribed final parameter
0.0	→	-0.0080	571		overall amount

Table 7.2. Space Shuttle problem: Adaptive continuation method [81] via trivial BVP extension (7.12).

A much smoother and faster behavior occurs when adaptive tangent continuation as worked out in Section 5.1 is applied—just see Table 7.3. With this method the temperature could be lowered even down to $1700°F$. Starting from these data, H.G. Bock [30] computed an achievable temperature of only $890°C$ from the multiple shooting solution of a Chebyshev problem assuming that all state constraints of the problem are to be observed.

Continuation sequence			*work*	*Remarks*
0.0	→	-0.0035	47	ordinary Newton method
-0.0035	→	-0.0057	32	throughout the computation;
-0.0057	→	-0.0080[a]	31	switching structure never disturbed
0.0	→	-0.0080	110	overall amount

[a] stepsize cut off to prescribed final value $\tau = -0.0080$.

Table 7.3. Space Shuttle problem: Adaptive tangent continuation [61]. See also Section 5.1 here.

Remark 7.1 It may be interesting to hear that none of these 'cooler' space shuttle trajectories has been realized up to now. In fact, the author of this book has presented the optimal $2000°F$ trajectories in 1977 within a seminar at NASA, Johnson Space Flight Center, Houston; the response there had been that the countdown for the launching of the first space shuttle (several

years ahead) had already gone too far to make any substantial changes. The second chance came when Europe thought about launching its own space shuttle HERMES; in fact, the maximum skin temperature assumed therein turned out to be the same as for the present NASA flights! Sooner or later, a newcomer in the space flight business (and this is big business!) will exploit this kind of knowledge which permits him (or her) to build a space shuttle in a much cheaper technology.

Gauss-Newton continuation method. As soon as *turning* or *bifurcation points* might arise, any Newton continuation method is known to be inefficient and Gauss-Newton techniques come into play—compare Section 5.2. The basic idea behind these techniques is to treat the parameter dependent nonlinear equations as an *underdetermined* system in terms of the extended variable $z = (x, \tau) = (x_1, \ldots, x_m, \tau)$. In addition to the Wronskian approximations G_j we therefore need the derivatives

$$g_j := \frac{\partial \Phi^{t_j+1,t_j}(\tau)x_j}{\partial \tau}\,, \quad j = 1, \ldots, m\,.$$

With these definitions the Jacobian $(nm, nm+1)$-matrix now has the block structure

$$J = \begin{bmatrix} G_1 & -I & & & g_1 \\ & \ddots & \ddots & & \vdots \\ & & G_{m-1} & -I & g_{m-1} \\ A & & & B & g_m \end{bmatrix}.$$

Based on this structure, *Gaussian block elimination* offers a convenient way to compute a Gauss-Newton correction

$$\widehat{\Delta z} = -J^- F$$

in terms of the outer inverse J^- already introduced in (7.5). The computation of $\widehat{\Delta z}$ can be conveniently based on a QR-decomposition of the $(n, n+1)$-matrix $[E, g]$, where

$$\begin{aligned} E &:= A + BG_{m-1} \cdot \cdots \cdot G_1\,, \\ g &:= g_m + B(g_{m-1} + \cdots + G_{m-1} \cdot \cdots \cdot G_2 g_1)\,. \end{aligned}$$

With this we obtain a variant of the condensing algorithm

$$\begin{aligned} \begin{pmatrix} \widehat{\Delta x_1} \\ \widehat{\Delta \tau} \end{pmatrix} &= -[E, g]^+ u\,, \\ \widehat{\Delta x_{j+1}} &= G_j \widehat{\Delta x_j} + g_j \widehat{\Delta \tau} + F_j\,, \quad j = 1, \ldots, m-1\,. \end{aligned} \tag{7.15}$$

Of course, iterative refinement sweeps must be properly added, see Section 7.1.1. This kind of Gauss-Newton continuation would need to be coupled by

some extra step-size control in the continuation parameter τ—which is not worked out here.

Instead we advocate a realization of the standard Gauss-Newton continuation method as developed in Section 5.2. In the spirit of (5.28) and (5.29), a Gauss-Newton correction in terms of the Moore-Penrose pseudo-inverse can be easily computed via

$$\Delta z = -J^+ F = \widehat{\Delta z} - \frac{(t, \widehat{\Delta z})}{(t,t)} t \,, \tag{7.16}$$

wherein t now denotes any *kernel vector* satisfying $Jt = 0$. Let $t = (t_1, \ldots, t_m, t_\tau)$ denote the partitioning of a kernel vector. Then components (t_1, t_τ) can be computed from the $(n, n+1)$-system

$$E t_1 + g t_\tau = 0$$

again via the QR-decomposition. The remaining components are once more obtained via explicit recursion as

$$t_{j+1} = G_j t_j + g_j t_\tau \;, \quad j = 1, \ldots, m-1 \,.$$

Insertion of the particular correction $\widehat{\Delta z}$ and the kernel vector t finally yields the Gauss-Newton correction Δz. The local convergence analysis as well as the corresponding step-size control in τ can then be copied from Section 5.2 without further modification.

As for the inner products arising in the above formula, the discrete L^2-product $(\cdot, \cdot)_\Delta$ defined in (7.9) looks most promising, since it implicitly reflects the structure of the BVP.

Detection of critical points. Just as in Section 5.2.3, certain determinant pairs need to be computed. This is especially simple in the context of the QR-decomposition of the matrix $[E, g]$ in (7.15).

Bibliographical Note. More details are given in the original paper [73] by P. Deuflhard, B. Fiedler, and P. Kunkel. There also a performance comparison of `MULCON`, a multiple shooting code with Gauss-Newton continuation as presented here, and `AUTO`, a collocation code with pseudo-arclength continuation due to E. Doedel [89] is presented: the given numerical example nicely shows that the *empirical* pseudo-arclength continuation is robust and reliable, but too cautious and therefore slower, whereas the *adaptive* Gauss-Newton continuation is also robust and reliable, but much faster. Moreover, continuous analogs of the augmented system of G. Moore for the characterization of bifurcations are worked out therein.

7.2 Parameter Identification in ODEs

This section deals with the *inverse problem* in ordinary differential equations (ODEs): given a system of n nonlinear ODEs

$$y' = f(y,p)\,, \quad y(0) \text{ given } , \quad p \in \mathbb{R}^q\,,$$

determine the unknown parameter vector p such that the solution $y(t,p)$ 'fits' given experimental data

$$(\tau_j, z_j) \qquad z_j \in \mathbb{R}^n\,, \quad j = 1,\ldots,M\,.$$

Let

$$\delta y(\tau_j,p) := y(\tau_j,p) - z_j\,, \quad j = 1,\ldots,M\,.$$

denote the *pointwise deviations* between model and data with prescribed statistical tolerances $\delta z_j,\ j = 1,\ldots,M$. If some of the components of z_j are not available, this formally means that the corresponding components of δz_j are infinite. In nonlinear least squares, the deviations are measured via a discrete (weighted) l_2-product $(\cdot,\cdot)$, which leads to the problem

$$(\delta y, \delta y) = \frac{1}{M}\sum_{j=1}^{M} \|D_j^{-1}\delta y(\tau_j,p)\|_2^2 = \min$$

with diagonal weighting matrices

$$D_j := \operatorname{diag}(\delta z_{j1},\ldots,\delta z_{jn})\,, \quad j = 1,\ldots,M\,.$$

If we define some nonlinear mapping F by

$$F(p) := \begin{bmatrix} D_1^{-1}\delta y(\tau_1,p) \\ \vdots \\ D_M^{-1}\delta y(\tau_M,p) \end{bmatrix},$$

then our least squares problem reads

$$\|F(p)\|_2^2 \equiv (\delta y, \delta y) = \min\,.$$

If *all* components at every data point τ_j have been measured (rare case), then $F : \mathbb{R}^q \longrightarrow \mathbb{R}^L$ with $L = nM$. Otherwise some $L < nM$ occurs.

We are thus guided to some *constrained nonlinear least squares problem*

$$y' = f(y,p), \quad (\delta y, \delta y) = \min\,,$$

where the ODEs represent the equality constraints. This problem type leads to a modification of the standard multiple shooting method.

The associated Jacobian (L, q)-matrix $F'(p)$ must also be computed exploiting its structure numerically

$$F'(p) = \begin{bmatrix} D_1^{-1} y_p(\tau_1) \\ \vdots \\ D_l^{-1} y_p(\tau_l) \end{bmatrix} . \tag{7.17}$$

Herein the *sensitivity matrices* y_p each satisfy the *variational equation*

$$y_p' = f_y(y, p) y_p + f_p(y, p)$$

with initial values $y_p(t_j) = 0, t_j \in \Delta$. Of course, we will naturally pick m multiple shooting nodes out of the set of M measurement nodes, which means that

$$\Delta := \{t_1, \ldots, t_m\} \subseteq \{\tau_1, \ldots, \tau_M\}$$

with, in general, $m \ll M$. As before, sub-trajectories $\Phi^{t,t_j}(p)x_j$ are defined per each subinterval $t \in [t_j, t_{j+1}]$ via the initial value problem

$$y' = f(y, p) , \quad y(t_j) = x_j .$$

Figure 7.3 gives a graphical illustration of the situation for the special case $M = 13$, $m = 4$.

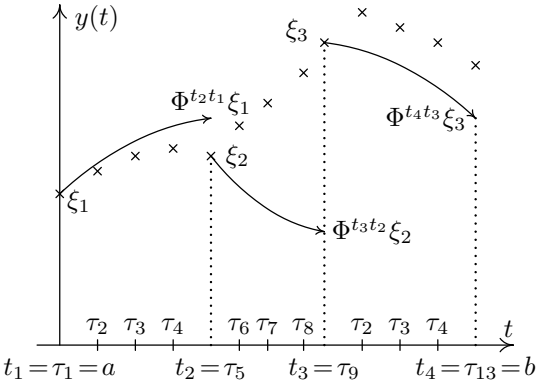

Fig. 7.3. Multiple shooting for parameter identification ($M = 13, m = 4$).

Unknowns to be determined are $(x, p) = (x_1, \ldots, x_m, p)$. If we introduce the convenient notation

$$r(x_1, \ldots, x_m, p) := \begin{bmatrix} D_1^{-1}(\Phi^{\tau_1,t_1}(p)x_1 - z_1) \\ \vdots \\ D_{M-1}^{-1}(\Phi^{\tau_{M-1},t_{m-1}}(p)x_{m-1} - z_{M-1}) \\ D_M^{-1}(x_m - z_M) \end{bmatrix} ,$$

we arrive at the parameter identification problem in its multiple shooting version

$$F_j(x_j, x_{j+1}, p) := \Phi^{t_{j+1},t_j}(p)x_j - x_{j+1} = 0\,, \quad j = 1, \ldots, m-1\,,$$
$$\|r(x_1, \ldots, x_m, p)\|_2^2 = \min\,.$$

Obviously, this is a constrained nonlinear least squares problem with the continuity equations as nonlinear constraints, an overdetermined extension of (7.11).

For ease of notation we write the whole mapping as

$$F(x,p) = \begin{bmatrix} F_1(x_1, x_1, p) \\ \vdots \\ F_{m-1}(x_{m-1}, x_m, p) \\ r(x_1, \ldots, x_m, p) \end{bmatrix}$$

and its block structured Jacobian matrix (ignoring weighting matrices for simplicity) as

$$J = \begin{bmatrix} G_1 & -I & & & P_1 \\ & \ddots & \ddots & & \vdots \\ & & G_{m-1} & -I & P_{m-1} \\ B_1 & \ldots & B_{m-1} & B_m & P_m \end{bmatrix}.$$

The above matrices P_j, $j = 1, \ldots, m$ represent the parameter derivatives of the mapping F consisting just as in (7.17) of sensitivity matrices; their length and initial values depend on the available measurement data and on the selection of the multiple shooting nodes out of the measurement modes—details are omitted here, since they require clumsy notation.

Upon recalling (7.11) and (7.5), the corresponding *constrained Gauss-Newton* corrections are defined as

$$(\Delta x^k, \Delta p^k) = -J(x^k, p^k)^- F(x^k, p^k)$$

or, more explicitly, via the block system

$$G_j \Delta x_j - \Delta x_{j+1} + P_j \Delta p = -F_j\,, \quad j = 1, \ldots, , m-1\,,$$
$$\|B_1 \Delta x_1 + \cdots + B_m \Delta x_m + P_m \Delta p + r\|_2^2 = \min\,.$$

Gaussian block elimination. Proceeding as in the simpler BVP case, we here obtain

$$\begin{pmatrix} \Delta x_1 \\ \Delta p \end{pmatrix} = -[E, P]^+ u\,,$$

wherein the quantities E, P, u are computed recursively from

$$\bar{P}_m := P_m, \bar{B}_m := B_m\,,$$
$$j = m-1,\dots,1 : \bar{P}_j := \bar{P}_{j+1} + \bar{B}_{j+1}P_j, \quad \bar{B}_j := B_j + \bar{B}_{j+1}G_j\,,$$
$$E := \bar{B}_1\,, \quad P := \bar{P}_1\,,$$
$$u := r + B_m[F_{m-1} + \cdots + G_{m-1}\cdot\cdots\cdot G_2F_1]\,.$$

The remaining correction components follow from

$$\Delta x_{j+1} = G_j\Delta x_j + P_j\Delta p + F_j\,, \quad j = 1,\dots,m-1\,.$$

In analogy with (7.5) and with the notation for the block matrices L, R introduced there, the generalized inverse J^- can be formally written as

$$J^- = RS^-L, \quad S^- = \text{diag}\,([E,P]^+, I, \cdots, I).$$

Bibliographical Note. Since 1981, this version of the multiple shooting method for parameter identification in differential equations has been suggested and driven to impressive perfection by H.G. Bock [29, 31, 32] and his coworkers. It is implemented in the program `PARFIT`. A single shooting variant especially designed for parameter identification in large chemical reaction kinetic networks has been worked out in detail by U. Nowak and the author [158, 159] in the code `PARKIN`.

Iterative refinement sweeps. In order to start the iterative refinement sweeps, we require some iterative correction of the above condensed least-squares system. A naive iterative correction approach, however, would not be suitable for large residuals

$$\bar{r} = E\Delta x_1 + P\Delta p + u\,.$$

Thus we recommend an algorithm proposed by Å. Bjørck [25] to be adapted to the present situation. In this approach the above linear least-squares problem is first written in the form of the augmented linear system of equations

$$\begin{aligned} -\bar{r} + E\Delta x_1 + P\Delta p + u &= 0\,,\\ [E,P]^T\bar{r} &= 0 \end{aligned}$$

in the variables Δx_1, Δp, $\bar{r}$. The iterative correction is then applied to this system. If it converges—which means that the condensed linear least-squares problem is regarded as well-posed—then, without any changes, the iterative refinement sweeps for the explicit recursion can be applied in the same form as presented in the standard situation treated in Section 7.1.1.

7.3 Periodic Orbit Computation

In this section we are interested in continuous solutions of *periodic boundary value problems*:

$$\begin{aligned} y' &= f(y)\,, \\ r(y(0), y(T)) &:= y(T) - y(0) = 0 \end{aligned} \tag{7.18}$$

with (hidden) time variable t and (unknown) period T. Here we treat only the situation that f is *autonomous*, the nonautonomous case is essentially standard. In this case f satisfies the *variational equation*

$$f' = f_y \cdot f\,,$$

which can be formally solved as

$$f(y(t)) = W(t,0)\; f(y(0))\,, \tag{7.19}$$

wherein $W(\cdot,\cdot)$ is once more the propagation matrix of the variational equation. Insertion of the periodicity condition $y(T) = y(0)$ then yields

$$f(y(T)) = f(y(0)) = W(T,0) f(y(0))\,,$$

or, equivalently, with $E = E(0) = W(T,0) - I$ inserted:

$$Ef(y(0)) = 0\,.$$

Obviously, the sensitivity matrix E is *singular* and $f(y(0))$ is a right eigenvector associated with eigenvalue zero, if only $f(y(0)) \neq 0$. The singularity of E reflects the fact that the phase or time origin is undetermined, causing a special nonuniqueness of solutions: whenever $y(t),\ t \in [0,T]$ is a periodic solution, then $y(t+t_0),\ t \in [0,T],\ t_0 \neq 0$, is a different periodic solution, even though it is represented by the same *orbit*. Obviously, there exists a continuous solution set generated by S^1-symmetry. In this situation, we will naturally aim at computing the *orbit* directly, which means computing *any* trajectory $y(t+t_0),\ t \in [0,T]$ without fixing the phase t_0.

In what follows we will assume that the eigenvalue zero of E is *simple*, which then implies that

$$\operatorname{rank}[E, f(y(0))] = n$$

and

$$\ker[E, f] = (f, 0)\,. \tag{7.20}$$

7.3.1 Single orbit computation

First we treat the case when a single periodic orbit is wanted. In order to convey the main idea, we start with the derivation of a special Gauss-Newton method in the framework of single shooting ($m = 2$).

Single Shooting. In this approach one obtains the *underdetermined* system of n equations

$$F(z) = \Phi^T x - x = 0$$

in the $n+1$ unknowns $z = (x, T)$ is generated. The corresponding Jacobian $(n, n+1)$-matrix has the form

$$F'(z) = [r_x, r_T] = [E, f(x(T))]\,,$$

or, after substituting a periodic solution,

$$F'(z) = [E, f(x(0))]\,.$$

Under the assumption made above the Jacobian matrix has full row rank and its Moore-Penrose pseudoinverse $F'(z)^+$ has full column rank. Hence, instead of a Newton method we can construct the *Gauss-Newton* iteration

$$\Delta z^k = -F'(z^k)^+ F(z), \quad z^{k+1} = z^k + \Delta z^k, \quad k = 0, 1, \ldots\,.$$

Also under the full rank assumption this iteration will converge *locally quadratically* to *some* solution z^* in the 'neighborhood' of a starting point z^0, i.e., to an arbitrary point on the orbit. This point determines the whole orbit uniquely.

Multiple shooting. In multiple shooting we need to have fixed nodes. So we introduce the dimensionless independent variable

$$s := \frac{t}{T} \in [0, 1]\,. \tag{7.21}$$

Let

$$\Delta := \{0 = s_1 < s_2 < \cdots < s_m = 1\}$$

denote the given partitioning with mesh sizes

$$\Delta s_j := s_{j+1} - s_j,\ j = 1, \ldots, m-1\,.$$

Then the following conditions must hold

$$\begin{aligned} F_j(x_j, x_{j+1}, T) &:= \Phi^{T\Delta s_j} x_j - x_{j+1} = 0\,,\ \ j = 1, \ldots, m-1\,, \\ r(x_1, x_m) &:= x_m - x_1 = 0\,. \end{aligned}$$

The subtrajectories can be formally represented by

$$\Phi^{T\Delta s_j} x_j = x_j + \int_{s=0}^{T\Delta s_j} f(y(s))ds\,.$$

With the Wronskian (n, n)-matrices

$$G_j := \frac{\partial \Phi^{T\Delta s_j} x_j}{\partial x_j} = W(s_{j+1}, s_j)\Big|_{\Phi^{T(s-s_j)}x_j)}$$

and the n-vectors

$$g_j := \frac{\partial \Phi^{T\Delta s_j} x_j}{\partial T} = \Delta s_j f(\Phi^{T\Delta s_j} x_j)$$

the underdetermined linear system to be solved in each Gauss-Newton step has the form

$$\begin{array}{rrrrrrr}
G_1\Delta x_1 & -\Delta x_2 & & & +g_1\Delta T & = & -F_1 \\
& \ddots & \ddots & & \vdots & & \vdots \\
& & G_{m-1}\Delta x_{m-1} & -\Delta x_m & +g_{m-1}\Delta T & = & -F_{m-1} \\
-\Delta x_1 & & & +\Delta x_m & & = & 0\,.
\end{array}$$

Gaussian block elimination. The 'condensing' algorithm as described in Section 7.1.1 will here lead to the small underdetermined linear system

$$E\Delta x_1 + g\Delta T + u = 0\,,$$

where

$$\begin{aligned}
E &= G_{m-1} \cdot \cdots \cdot G_1 - I\,, \\
g &:= g_{m-1} + \cdots + G_{m-1} \cdot \cdots \cdot G_2 g_1\,, \\
u &:= F_{m-1} + \cdots + G_{m-1} \cdot \cdots \cdot G_2 F_1\,.
\end{aligned}$$

At a solution point $z^* = (x_1^*, \ldots, x_m^*, T^*)$ we obtain

$$E^* = W(1,0) - I\,, \quad g^* = f(x_1^*)\,,$$

just recalling that

$$\begin{aligned}
&f(x_m^*) = W(s_m, s_j) f(x_j^*) = f(x_1^*), \\
&\Delta s_1 + \cdots + \Delta s_{m-1} = 1\,.
\end{aligned}$$

Hence, the $(n, n+1)$-matrix

$$[E^*, g^*] = [E(0), f(y(0))]$$

has again full row rank n. Consequently, an extension of the single shooting rank-deficient Gauss-Newton method realizing the Jacobian outer inverse J^- can be envisioned.

Better convergence properties, however, are expected by the standard Gauss-Newton method which requires the Moore-Penrose pseudoinverse J^+ of the total block Jacobian. As in the case of parameter dependent BVPs, we again compute a kernel vector $t = (t_1, \ldots, t_m, t_T)$, here according to

$$[E, g] \begin{pmatrix} t_1 \\ t_T \end{pmatrix} = 0\,,$$
$$t_{j+1} = G_j t_j + g_j t_T,\ j = 1, \ldots, m-1$$

and combine it with iterative refinement sweeps, of course. The computation of the actual Gauss-Newton correction then follows from

$$\Delta z = \widehat{\Delta z} - \frac{(t, \widehat{\Delta z})}{(t, t)} t\,,$$

where $\widehat{\Delta z}$ is an arbitrary particular correction vector satisfying

$$F'(z^k)\widehat{\Delta z} + F(z^k) = 0\,.$$

Simplified Gauss-Newton method. At the solution point z^*, the above equations lead (up to some normalization factor) to the known solution

$$(t_1, t_T) = (f(x_1^*), 0)\,.$$

Upon inserting this result into (7.19), we immediately arrive at

$$t_j = f(x_j^*)\,,\quad j = 2, \ldots, m\,.$$

This inspires an intriguing modification of the above Gauss-Newton method: we may insert the 'iterative' kernel vector

$$t_T^k = 0,\ t_j^k = f(x_j^k)\ \ j = 1, \ldots, m$$

into the expression (7.16). In this way we again obtain some pseudo-inverse and, in turn, thus define some associated Gauss-Newton method.

Bibliographical Note. The multiple shooting version realizing the Jacobian outer inverse J^- has been suggested in 1984 by P. Deuflhard [64] and implemented in the code `PERIOD`. The improvement realizing J^+ has been proposed in 1994 by C. Wulff, A. Hohmann, and P. Deuflhard[199] and realized in the orbit continuation code `PERHOM`—for details see the subsequent Section 7.3.2. The same paper also contains a possible exploitation of symmetry for equivariant orbit problems, following up the work of K. Gatermann and A. Hohmann [95] for equivariant steady state problems.

7.3.2 Orbit continuation methods

In this section we consider the computation of families of orbits for the parameter dependent periodic boundary value problem

$$y' = f(y, \lambda)\,,$$
$$r(y(0), y(T)) := y(T) - y(0) = 0\,,$$

where λ is the embedding parameter and T the unknown period. Note that the S^1-symmetry now only holds for *fixed* λ, so that the orbits can be explicitly parametrized with respect to λ. Throughout this section, let $\{\lambda_\nu\}$ denote the parameter sequence and

$$\Delta\lambda_\nu := \lambda_{\nu+1} - \lambda_\nu$$

the corresponding continuation step sizes to be automatically selected.

Single shooting. In order to convey the main geometrical idea, we again start with this simpler case. There we must solve the sequence of problems

$$F(x, T, \lambda) := \Phi^T(\lambda)x - x = 0$$

for the parameters $\lambda \in \{\lambda_\nu\}$.

Classical continuation method. This continuation method, where the previous orbit just serves as starting guess for the Gauss-Newton iteration to compute the next orbit, can be implemented without any further discussion, essentially as described in Section 5.1.

Tangent continuation method. The realization of this method deserves some special consideration. The Jacobian $(n, n+2)$-matrix has the following substructure

$$[F_x, F_T, F_\lambda] = [E, f, p]$$

with E, f as introduced above and p defined as

$$p := \frac{\partial \Phi^T(\lambda)x}{\partial \lambda}\,.$$

Let $t = (t_x, t_T, t_\lambda)$ denote any kernel vector satisfying

$$Et_x + f \cdot t_T + p \cdot t_\lambda = 0\,.$$

Under the above assumption that $[E, f]$ has full row rank n, we here know that

$$\dim \ker[E, f, p] = 2\,.$$

A natural basis for the kernel will be $\{t^1, t^2\}$ such that

$$t^1 := \ker[E, f]\,, \quad t^2 \perp t^1\,, \tag{7.22}$$

wherein

$$t^i = (t^i_x, t^i_T, t^i_\lambda)\,, \quad i = 1, 2\,.$$

From (7.20) and (7.22) we are directly led to the representations (ignoring any normalization)

$$t^1_x = f\,, \quad t^1_T = 0\,, \quad t^1_\lambda = 0$$

and

$$\begin{pmatrix} t_x^2 \\ t_T^2 \end{pmatrix} = -[E, f]^+ p \,, \quad t_\lambda^2 := 1 \,.$$

Upon recalling that t^1 reflects the S^1-symmetry of each orbit, tangent continuation will mean to continue along t^2. In the notation of Section 5 this means that guesses $(\hat{x}, \hat{T})$ can be predicted as

$$\begin{pmatrix} \hat{x}(\lambda_{\nu+1}) - \bar{x}(\lambda_\nu) \\ \hat{T}(\lambda_{\nu+1}) - t(\lambda_\nu) \end{pmatrix} = \left. \begin{pmatrix} t_x^2 \\ t_T^2 \end{pmatrix} \right|_{\lambda_\nu} \cdot \Delta\lambda_\nu \,.$$

These guesses are used as starting points for the Gauss-Newton iteration as described in Section 7.3.1. Since $t_T^1 = t_\lambda^1 = 0$, the property $t^1 \perp t^2$ also implies $t_x^1 \perp t_x^2$, which means that

$$\hat{x}(\lambda_{\nu+1}) - \bar{x}(\lambda_\nu) \perp f(\bar{x}(\lambda_\nu)) \,. \tag{7.23}$$

The geometric situation is represented schematically in Figure 7.4.

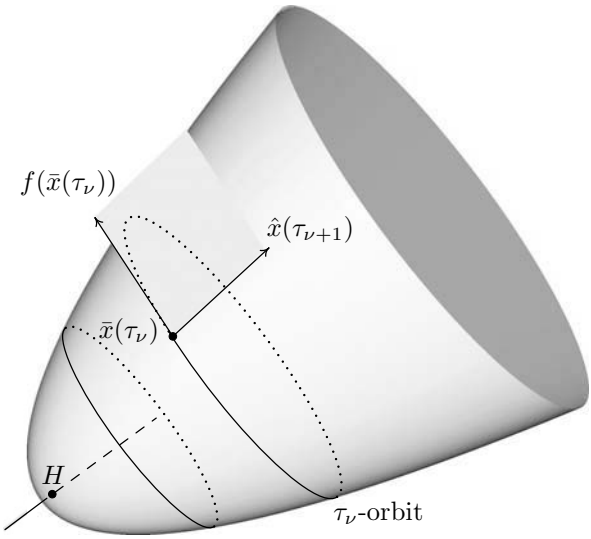

Fig. 7.4. Orbit continuation: H Hopf bifurcation point, — stable steady states, - - - unstable steady states.

Multiple Shooting. As in (7.21) we again deal with fixed nodes

$$\Delta := \{0 = s_1 < s_2 < \cdots < s_m = 1\}$$

instead of variable nodes $t_j = s_j T, \; j = 1, \ldots, m$.

Switching notation, let now $t = (t_1, \ldots, t_m, t_T, t_\lambda)$ denote a selected kernel vector satisfying

$$G_j t_j - t_{j+1} + g_j \cdot t_T + p_j t_\lambda = 0\,, \quad j = 1, \ldots, m-1\,,$$
$$t_m = t_1\,,$$

wherein G_j, g_j, and p_j are essentially defined as in single shooting ($m = 2$). For $m > 2$, the Jacobian nullspace is still two-dimensional. For continuation we again choose the tangent vector

$$t = (f(x_1), \ldots, f(x_m), 0, 0),\ x = \bar{x}(\lambda_\nu)\,.$$

Gaussian block elimination. In this setting, we will compute

$$\begin{pmatrix} \widehat{\Delta x_1} \\ \widehat{\Delta T} \end{pmatrix} := -[E, f]^+ p \cdot \Delta\lambda_\nu$$

with

$$p := p_{m-1} + \cdots + G_{m-1} \cdot \cdots \cdot G_2 p_1$$

and, recursively, for $j = 1, \ldots, m-2$:

$$\widehat{\Delta x}_{j+1} = G_j \widehat{\Delta x_j} + g_j \widehat{\Delta T} + p_j \Delta\lambda_\nu\,.$$

As a straightforward consequence, we may verify that

$$\widehat{\Delta x_m} = \widehat{\Delta x_1}\,.$$

The thus defined continuation

$$\hat{x}_j(\lambda_{\nu+1}) - \bar{x}_j(\lambda_\nu) = \widehat{\Delta x_j}\,, \quad j = 1, \ldots, m, \quad \hat{T}(\lambda_{\nu+1}) - t(\lambda_\nu) = \widehat{\Delta T}$$

clearly satisfies the *local* orthogonality property

$$\hat{x}_1(\lambda_{\nu+1}) - \bar{x}_1(\lambda_\nu) \perp f(\bar{x}_1(\lambda_\nu))\,,$$

which is *biased* towards the node $s_1 = 0$. Therefore, already from a geometrical point of view, the above continuation method should be modified such that the *global* orthogonality

$$\hat{x}(\lambda_{\nu+1}) - \bar{x}(\lambda_\nu) \perp f(\bar{x}(\lambda_\nu))$$

holds as a natural extension of (7.23). This directly leads us to the following orbit continuation method

$$\hat{x}(\lambda_{\nu+1}) - \bar{x}(\lambda_\nu) = \widehat{\Delta x} - \frac{(f_\nu, \widehat{\Delta x})}{(f_\nu, f_\nu)} f_\nu$$

with $f_\nu = f(\bar{x}(\lambda_\nu))$. Once more, the periodicity condition

$$\hat{x}_m(\lambda_{\nu+1}) = \hat{x_1}(\lambda_{\nu+1})$$

can be shown to hold.

Hopf bifurcations. The *detection* of Hopf bifurcation points H is an easy computational task. First, orbit continuation beyond H is impossible, as long as we come from the side of the periodic orbits, which we here do—see Figure 7.4. Second, at H, the condition $f = 0$ leads to $\operatorname{rank}[E, f] < n$—a behavior that already shows up in a neighborhood of H. Third, the automatic stepsize control will lead to a significant reduction of the stepsizes $\Delta\lambda_\nu$. As soon as a Hopf bifurcation point seems to close by, its precise computation can be done switching to the augmented system suggested by A.D. Jepson [124].

Bibliographical Note. The above orbit continuation has been worked out in 1994 by C. Wulff, A. Hohmann, and P. Deuflhard [199] and realized in the code `PERHOM`. The same paper also covers the detection and computation of Hopf bifurcations and period doublings. Particular attention is paid to the computational exploitation of symmetries following up work of M. Dellnitz and B. Werner [50] and of K. Gatermann and A. Hohmann [95], the latter for steady state problems only, the former including Hopf bifurcations as well.

7.3.3 Fourier collocation method

In quite a number of application fields the desired periodic solution y to period $T = 2\pi/\omega$ is just expanded into a Fourier series according to

$$y(t) = \tfrac{1}{2}a_0 + \sum_j \left(a_j \cos(j\omega t) + b_j \sin(j\omega t)\right) .$$

Such a solution living in the infinite dimensional function space L_2 cannot be directly computed from the periodic BVP (7.18). Instead one aims at computing some *Fourier-Galerkin approximation* y_m out of the finite dimensional subspace $U_m \subset L_2$ according to the finite Fourier series ansatz

$$y_m(t) = \tfrac{1}{2}a_0 + \sum_{j=1}^{m} \left(a_j \cos(j\omega t) + b_j \sin(j\omega t)\right) , \tag{7.24}$$

where $a_j, b_j \in \mathbb{R}^n$. Insertion into the *approximate periodic* BVP

$$y_m' = f(y_m) , \qquad y_m(T) = y_m(0)$$

will require the coefficients of the derivative defined via

$$y_m'(t) = \sum_{j=1}^{m} \left(a_j' \cos(j\omega t) + b_j' \sin(j\omega t)\right)$$

with

$$a_j' = j\omega b_j , \qquad b_j' = -j\omega a_j$$

as well as those of the right side defined via

$$f(y_m(t)) = \sum_{j=1}^{m} (\alpha_j \cos(j\omega t) + \beta_j \sin(j\omega t)) \; . \tag{7.25}$$

Upon inserting the two expansions into the BVP (7.24), we arrive at the system of $N = 2m + 1$ relations

$$j\omega b_j = \alpha_j \, , \quad j = 0, \ldots, m \, , \qquad -j\omega a_j = \beta_j \, , \quad j = 1, \ldots, m \, .$$

From the theory of Fourier transforms we know that the right side coefficients can be computed via

$$\alpha_j = \frac{2}{T} \int_{t=0}^{T} f(y_m(t)) \cos(j\omega t) \; dt \, , \quad \beta_j = \frac{2}{T} \int_{t=0}^{T} f(y_m(t)) \sin(j\omega t) \; dt \, . \tag{7.26}$$

Obviously, this Galerkin approach involves a continuous Fourier transform which inhibits the construction of an approximation scheme for y_m.

Therefore, already in 1965, M. Urabe [188] suggested to replace the continuous Fourier transform by a *discrete Fourier transform*, i.e., by trigonometric interpolation over equidistant nodes

$$t_k = T \frac{k}{N} \, , \quad k = 0, 1, \ldots, N \, .$$

Formally speaking, the integrals in (7.26) are then approximated by their trapezoidal sums defined over the selected set of nodes. As a consequence, we now substitute the Galerkin approximation y_m by a *Galerkin-Urabe approximation* defined via the modified Fourier series expansion

$$\hat{y}_m(t) = \tfrac{1}{2} \hat{a}_0 + \sum_{j=1}^{m} \Big(\hat{a}_j \cos(j\omega t) + \hat{b}_j \sin(j\omega t) \Big) \tag{7.27}$$

and the corresponding expansion for its derivative $\hat{y}'_m(t)$. Instead of (7.25) we now have a modified expansion

$$f(\hat{y}_m(t)) = \sum_{j=1}^{m} \Big(\hat{\alpha}_j \cos(j\omega t) + \hat{\beta}_j \sin(j\omega t) \Big)$$

and instead of the representation (7.26) we obtain the well-known trigonometric expressions

$$\hat{\alpha}_j = \frac{2}{N} \sum_{k=0}^{N-1} f(\hat{y}_m(t_k)) \cos(j\omega t_k) \, , \quad \hat{\beta}_j = \frac{2}{N} \sum_{k=0}^{N-1} f(\hat{y}_m(t_k)) \sin(j\omega t_k) \, . \tag{7.28}$$

We again require the $N = 2m + 1$ relations

$$j\omega\hat{b}_j = \hat{\alpha}_j \ , \quad j = 0, \ldots, m \ , \qquad -j\omega\hat{a}_j = \hat{\beta}_j \ , \quad j = 1, \ldots, m \, . \tag{7.29}$$

As before, we insert the expansion for $\hat{y}_m$ into the formal representation (7.28) so that the coefficients $\hat{\alpha}_j, \hat{\beta}_j$ drop out and we arrive at a system of nN equations, in general nonlinear. Originally, Urabe had suggested this approach, also called *harmonic balance method*, for nonautonomous periodic BVPs where T (or ω, respectively) is given in the problem so that the nN unknown coefficients $(\hat{a}_0, \hat{a}_1, \hat{b}_1, \ldots, \hat{a}_m, \hat{b}_m)$ can, in principle, be computed.

For *autonomous* periodic BVPs as treated here, system (7.29) turns out to be underdetermined with $nN + 1$ unknowns $(\hat{a}_0, \hat{a}_1, \hat{b}_1, \ldots, \hat{a}_m, \hat{b}_m, \omega)$ to be computed. We observe that in this kind of approximation local nonuniqueness shows up just as in the stated original problem: given a solution with computed coefficients $\hat{a}_j, \hat{b}_j$, then any trajectory defined by the modified coefficients

$$\tilde{a}_0 = \hat{a}_0 \ , \quad \tilde{a}_j = \hat{a}_j \cos(j\omega\tau) + \hat{b}_j \sin(j\omega\tau) \ , \quad \tilde{b}_j = \hat{b}_j \cos(j\omega\tau) - \hat{a}_j \sin(j\omega\tau) \, ,$$

is also a solution, shifted by τ. Therefore we may simply transfer the Gauss-Newton methods for single orbit computation or for orbit continuation (see the preceding sections) to the nonlinear mapping as just defined.

Numerical realization of Gauss-Newton method. For ease of writing, we here ignore the difference between y_m and $\hat{y}_m$ and skip all 'hats' in the coefficients. Then the underdetermined system has the form

$$F(z) = \begin{pmatrix} \dfrac{2}{N} \displaystyle\sum_{k=0}^{N-1} f(y_m(t_k)) \\ \vdots \\ \dfrac{2}{N} \displaystyle\sum_{k=0}^{N-1} f(y_m(t_k)) \cos(j\omega t_k) + j\omega b_j \\ \dfrac{2}{N} \displaystyle\sum_{k=0}^{N-1} f(y_m(t_k)) \sin(j\omega t_k) - j\omega a_j \\ \vdots \end{pmatrix} = 0 \, , \quad z = \begin{pmatrix} a_0 \\ \vdots \\ a_j \\ b_j \\ \vdots \\ \omega \end{pmatrix}$$

in terms of a mapping $F : \mathbb{R}^{nN+1} \longrightarrow \mathbb{R}^{nN}$. Herein z additionally enters via the expression (7.27) understood to be inserted for y_m. For the corresponding Jacobian $(nN, nN + 1)$-matrix we may write block columnwise

$$F'(z) = (F_{a_0}(z), F_{a_1}(z), F_{b_1}(z), \ldots, F_{a_m}(z), F_{b_m}(z), F_\omega(z)) \ .$$

Assume we have already computed the Fourier series expansion of the (n, n)-matrix $f_y(y_m)$, say with coefficients

$$A_j = \frac{2}{N}\sum_{k=0}^{N-1} f_y(y_m(t_k))\cos(j\omega t_k)\,, \quad B_j = \frac{2}{N}\sum_{k=0}^{N-1} f_y(y_m(t_k))\sin(j\omega t_k)\,.$$

Then a straightforward calculation reveals that

$$F_\omega(z) = \begin{pmatrix} 0 \\ \vdots \\ ja_j \\ -jb_j \\ \vdots \end{pmatrix}, \quad F_{a_0}(z) = \begin{pmatrix} \frac{1}{2}A_0 \\ \vdots \\ \frac{1}{2}B_j \\ \\ \frac{1}{2}A_j \\ \vdots \end{pmatrix},$$

and, for $l = 1, \ldots, m$:

$$F_{a_l}(z) = \begin{pmatrix} A_l \\ \vdots \\ \frac{1}{2}(B_{l+j} - B_{l-j}) + j\omega\delta_{jl}I_n \\ \\ \frac{1}{2}(A_{l+j} + A_{l-j}) \\ \vdots \end{pmatrix},$$

and

$$F_{b_l}(z) = \begin{pmatrix} B_l \\ \vdots \\ \frac{1}{2}(-BA_{l+j} + A_{l-j}) \\ \\ \frac{1}{2}(B_{l+j} + B_{l-j}) - j\omega\delta_{jl}I_n \\ \vdots \end{pmatrix}.$$

In this representation, certain indices run out of the permitted index set: whenever an index $l > m$ appears, then replace A_l by A_{N-l} and B_l by $-B_{N-l}$; whenever $l < 0$, then replace A_l by A_{-l} and B_l by $-B_{-l}$. In this way all computations can be performed using the FFT algorithm for f and f_y, assuming that an iterate y_m is at hand—which it is during the Gauss-Newton iterations for single orbit computation or orbit continuation (see preceding sections).

Adaptivity device. Up to now, we have not discussed the number m of terms necessary to obtain an approximations to prescribed accuracy. Let $|\cdot|$ denote the $L_2[0,T]$-norm then we have

$$\varepsilon_m = |y - y_m| = \left(\sum_{j=m+1}^{\infty} \left(a_j^2 + b_j^2\right)\right)^{\frac{1}{2}}.$$

If we repeat the Galerkin-Urabe procedure with m replaced by $M \gg m$ then we may choose the computationally available term

$$[\varepsilon_m] = |y_M - y_m| = \left(\sum_{j=m+1}^{M} \left(a_j^2 + b_j^2\right) \right)^{\frac{1}{2}} \leq \varepsilon_m$$

as a reasonable error estimate. Upon again ignoring the difference between the Fourier coefficients y_m and $\hat{y}_m$, we may apply a well-known approximation result from Fourier analysis: assume that the unknown function y and all its approximations y_m are analytic, then the coefficients obey some exponential decay law, which we write in the form

$$\varepsilon_m \doteq Ce^{-\gamma m}\,,$$

where the coefficients C, γ are unknown a-priori and need to be estimated. For this purpose, let the optimal number m^* be such that

$$\varepsilon_{m^*} \doteq \mathrm{TOL}$$

and assume that this is not yet achieved for the actual index m. Then a short calculation (see also Exercise 7.6 for more details) shows that m^* can be estimated by the adaptive rule (with some further index $l \ll m$)

$$m^* \doteq m + (m-l)\frac{\log([\varepsilon_m]/\,\mathrm{TOL})}{\log([\varepsilon_l]/[\varepsilon_m])}\,. \tag{7.30}$$

Only with such an adaptivity device added, the Galerkin-Urabe (also: harmonic balance) method can be expected to supply reliable computational results.

From (7.28) and (7.29) we may readily observe that in the Galerkin-Urabe approach the BVP (7.24) has been tacitly replaced by the *discrete boundary value problem*

$$\hat{y}_m'(t_k) = f\left(\hat{y}_m(t_k)\right)\ , \quad k = 0, 1, \ldots, N\,, \quad \hat{y}_m(T) = \hat{y}_m(0)\,,$$

wherein $t_0 = 0, t_N = T$ by definition. The boundary conditions are implicitly taken into account by the Fourier ansatz. One of the conditions, at $t = 0$ or at $t = T$, can be dropped due to periodicity so that there are nN so-called *collocation conditions* left. By construction, the method inherits the symmetry of the BVP with respect of an interchange of the boundaries $t = 0$ and $t = T$—indicating that this computational approach treats periodic BVPs as *spacelike* BVPs—as opposed to the preceding Sections 7.3.1 and 7.3.2 where multiple shooting approaches for timelike BVPs have been discussed. The following Section 7.4 is fully devoted to collocation methods for spacelike BVPs—there, however, in connection with polynomial approximation.

7.4 Polynomial Collocation for Spacelike BVPs

In the collocation approach the interval $[a, b]$ is subdivided into a partition

$$\Delta = \{a = t_1 < t_2 < \cdots < t_m = b\}, \quad m > 2\,,$$

where each subinterval $I_j = [t_j, t_{j+1}]$ of length $\tau_j = t_{j+1} - t_j$ is further subdivided by s internal nodes, the so-called *collocation points*

$$t_{ji} = t_j + c_i \tau_j\ ,\ i = 1, \ldots, s\ , \quad 0 \le c_1 < \cdots < c_s \le 1$$

corresponding to some quadrature rule of order p with nodes c_i. Let Δ_* denote the union of all collocation points—to be distinguished from the above defined coarse mesh Δ.

Let u denote the collocation polynomial to be computed for the given BVP. The collocation polynomial is defined via the n boundary conditions

$$F_m = r(u(a), u(b)) = 0$$

typically assumed to be linear separated (cf. [9, 71])

$$F_m = Au(a) + Bu(b) - d = 0\,, \tag{7.31}$$

so that all components arising therein can be fixed. At the 'internal' nodes we require the $(m-1)sn$ 'local' collocation conditions

$$F_{ji} = u'(t_{ji}) - f(u(t_{ji})) = 0, \qquad t_{ji} \in \Delta_* \tag{7.32}$$

often in the scaling invariant form (i.e. invariant under rescaling of the variable t)

$$F_{ji} = \tau_j\,(u'(t_{ji}) - f(u(t_{ji}))) = 0, \qquad t_{ji} \in \Delta_*\,.$$

Finally, the $(m-1)n$ 'global' collocation conditions

$$F_j = u(t_{j+1}) - u(t_j) - \tau_j \sum_{l=1}^{s} b_l f(u(t_{jl})) = 0\ , \quad t_j \in \Delta \tag{7.33}$$

must hold, wherein the quadrature rule implies the relation

$$\sum_{l=1}^{s} b_l = 1\,.$$

By construction, the collocation approach is invariant under $a \leftrightarrow b$ whenever a *symmetric* quadrature rule with

$$c_i = 1 - c_{s+1-i}\ , \quad b_i = b_{s+1-i}\ , \qquad i = 1, \ldots, s$$

is selected. In fact, symmetric collocation methods are realized in nearly all public domain codes, since they permit the highest possible convergence orders p by one out of the following two options:

- *Gauss methods.* Here collocation points are selected as the nodes of Gauss-Legendre quadrature. This leads to the highest possible order $p = 2s$. The simplest case with $s = 1$ is just the implicit midpoint rule. Since $c_0 > 0$ and $c_s < 1$, the nodes of the coarse mesh are not collocation points, i.e., $\Delta_* \cap \Delta = \emptyset$, and hence only $u \in C^0[a, b]$ is obtained.
- *Lobatto methods.* Here the collocation points are selected as the nodes of Lobatto quadrature. The attainable order is $p = 2s - 2$. The simplest case is the implicit trapezoidal rule with $p = s = 2$. Since $c_0 = 0$ and $c_s = 1$, the nodes of the coarse mesh are included in the set of collocation points, i.e., $\Delta_* \cap \Delta = \Delta$. The lower order (compared with the Gauss methods) comes with better global smoothness, since here $u \in C^1[a, b]$.

In algorithmic implementations, Gauss methods are usually preferred due to their more robust behavior in nonsmooth BVPs.

Bibliographical Note. Efficient collocation methods have been implemented in the classical code `COLSYS` of U.M. Ascher, J. Christiansen, and R.D. Russell and its more recent variant `COLNEW` by G. Bader and U.M. Ascher [16]. An adaptive Gauss-Newton continuation method (as described in Section 5.2) has been implemented in the code `COLCON` by G. Bader and P. Kunkel [17]. An advanced *residual based inexact* Gauss-Newton continuation method has been designed for collocation by A. Hohmann [120] and realized in the rather robust research code `COCON`. Unfortunately, that line of development has not continued toward a fully satisfactory general purpose collocation code. We will resume the topic, in Section 7.4.2 below, in the frame of *error oriented inexact* Newton methods. Recently, this concept has regained importance in a novel multilevel algorithm for optimal control problems based on function space complementarity methods—a topic beyond the present scope, for details see M. Weiser and P. Deuflhard [197].

7.4.1 Discrete versus continuous solutions

From multiple shooting techniques we are accustomed to the fact that, whenever the underlying BVP has a locally unique solution, the discrete system also has a locally unique solution—just look up Lemma 7.1 and the interpretation thereafter. *This need not be the case for global discretization methods.* Here additional 'spurious' discrete solutions may occur that have nothing to do with the unique continuous BVP solution. In [71, Section 8.4.1] this situation has been analyzed for the special method based on the implicit trapezoidal rule, the simplest Lobatto method already mentioned above. In what follows we want to give the associated analysis for the whole class of collocation methods. We will mainly focus on Gauss collocation methods, which are the ones actually realized in the most efficient collocation codes. Our results do, however, also apply to other collocation schemes.

Throughout this section we assume—without proof—that the BVP has a *unique* solution $y \in C^{p+1}[a,b]$ and that the collocation polynomial u is globally continuous, i.e., $u \in C^0[a,b]$, but only piecewise sufficiently differentiable, $u \in C^{p+1}[t_j, t_{j+1}]$, which we denote by $u \in C_\Delta^{p+1}[a,b]$. In what follows we restrict our attention to Gauss methods, which means $p = 2s$. Consequently, the discretization error $\epsilon(t) = u(t) - y(t), t \in [a,b]$ satisfies $\epsilon \in C_\Delta^{p+1}[a,b]$. In order to study its behavior, we introduce norms over the grids Δ, Δ_*, for example

$$|\epsilon|_\Delta = \max_{t \in \Delta} \|\epsilon(t)\|$$

in terms of some vector norm $\|\cdot\|$. Let

$$\tau = \max_{j=1,\ldots,m-1} \tau_j$$

denote the maximum mesh size on the coarse grid Δ. With these preparations we are now ready to state a convergence theorem for Gauss collocation methods.

Theorem 7.2 *Notation as just introduced. Consider a BVP on $[a,b]$ with linear separated boundary conditions and a right side f that is p-times differentiable with respect to its argument. Assume that the BVP has a unique solution $y \in C^{p+1}[a,b]$. Let this solution be well-conditioned with bounded interval condition number $\bar{\rho}$. Define a global Lipschitz constant ω via*

$$\|f_y(v) - f_y(w)\| \le \omega \|v - w\| . \tag{7.34}$$

Consider a Gauss collocation scheme based on a quadrature rule of order $p = 2s$. Let the discrete BVP have a collocation solution $u \in C_\Delta^{p+1}[a,b]$ that is consistent with the BVP solution y. Let γ, γ^* denote error coefficients corresponding to the Gauss quadrature rule and depending on the smoothness of the right hand side f. Then, for

$$\tau \le \left(2\omega\gamma^*(\bar{\rho}|b-a|)^2\right)^{-\frac{1}{s+1}} , \tag{7.35}$$

the following results hold:

(I) *At the local (internal) nodes the pointwise approximation satisfies*

$$|u - y\,|_{\Delta_*} \le 2\bar{\rho}|b-a|\gamma^*\tau^{s+1} . \tag{7.36}$$

(II) *At the global nodes superconvergence holds in the sense that*

$$|u - y\,|_\Delta \le \bar{\rho}|b-a|\left(2\omega(\bar{\rho}|b-a|\gamma^*\tau)^2 + \gamma\right)\tau^{2s} . \tag{7.37}$$

Proof. I. We begin with deriving a perturbed variational equation for the discretization error $\epsilon \in C_\Delta^{p+1}[a,b]$. For $t \in I_j = [t_j, t_{j+1}]$ we may write

$$\epsilon'(t) - f_y(y(t))\epsilon(t) = \delta f(t) + \delta\varphi(t)\,, \tag{7.38}$$

where

$$\delta f(t) = u'(t) - f(u(t))$$

and

$$\begin{aligned}\delta\varphi(t) &= f(u(t)) - f(y(t)) - f_y(y(t))\epsilon(t) \\ &= \int_{\Theta=0}^{1} \big(f_y(y(t) + \Theta\epsilon(t)) - f_y(y(t))\big)\epsilon(t)\, d\Theta\,.\end{aligned}$$

The above first term δf vanishes at the collocation points and gives rise to the local upper bound for the interpolation error (see, e.g., [71, Thm. 7.16])

$$\max_{\sigma\in I_j} \|\delta f(\sigma)\| \le \overline{\gamma}\tau_j^s\,. \tag{7.39}$$

The second term $\delta\varphi$ contains the nonlinear contribution and satisfies the pointwise estimate

$$\|\delta\varphi(t)\| \le \tfrac{1}{2}\omega\|\epsilon(t)\|^2\,. \tag{7.40}$$

By variation of constants (see Exercise 7.7) the differential equation (7.38) can be formally solved for $t \in [t_j, t_{j+1}]$ to yield

$$\epsilon(t) = W(t, t_j)\epsilon(t_j) + \int_{\sigma=t_j}^{t} W(t,\sigma)\,(\delta f(\sigma) + \delta\varphi(\sigma))\; d\sigma\,,$$

where $W(\cdot,\cdot)$ denotes the (Wronskian) propagation matrix, the solution of the unperturbed variational equation—see [71, Section 3.1.1]. This, however, is just a representation of the solution of the IVP on each subinterval. Therefore, any estimates based on this formula would bring in the IVP condition number, which we want to avoid for spacelike BVPs.

For this reason, we need to include the boundary conditions, known to be linear separated, so that

$$\delta r = A\epsilon(t_1) + B\epsilon(t_m) = 0\,.$$

Upon combining these results, we obtain the formal global representation

$$\epsilon(t) = \int_{\sigma=a}^{b} G(t,\sigma)\,(\delta f(\sigma) + \delta\varphi(\sigma))\; d\sigma\,,$$

where $G(\cdot,\cdot)$ denotes the *Green's function* of the (linear) variational BVP. Its global upper bound is the *condition number* of the (nonlinear) BVP (as defined in [71, Section 8.1.2]):

$$\bar{\rho} = \max_{t,\bar{t}\in[a,b]} \|G(t,\bar{t})\| \,.$$

Since the BVP is assumed to be well-conditioned, the above condition number is bounded. Note that, by definition, the Green's function has a jump at $t = \bar{t}$ such that

$$G(t,t^+) - G(t,t^-) = I \,.$$

For the subsequent derivation we decompose the integral into subintegrals over each of the subintervals such that

$$\epsilon(t) = \sum_{k=1}^{m-1} \int_{\sigma=t_k}^{t_{k+1}} G(t,\sigma)\,(\delta f(\sigma) + \delta\varphi(\sigma))\; d\sigma \,.$$

II. We are now ready to derive upper bounds for the discretization error. For $t_j \in \Delta$, we may directly apply the corresponding Gauss quadrature rule, since the above jumps occur only at the boundaries of each of the subintegrals. Along this line we obtain

$$\epsilon(t_j) = \sum_{k=1}^{m-1} \tau_k \Big(\sum_{l=1}^{s} b_l G(t_j, t_{kl}) \delta\varphi(t_{kl}) + \Gamma_k(\cdot)\tau_k^{2s} \Big) \,.$$

The argument in the remainder term $\Gamma_k(\cdot)$ is dropped, since we are only interested in its global upper bound, say

$$\|\Gamma_k(\cdot)\| \le \bar{\rho}\gamma \,.$$

Introducing pointwise norms on the coarse grid Δ, and exploiting (7.40) and (7.36), we are then led to

$$\|\epsilon(t_j)\| \le \bar{\rho}|b-a| \left(|\delta\varphi|_{\Delta_*} + \gamma\tau^{2s} \right) ,$$

which yields

$$|\epsilon|_{\Delta} \le \bar{\rho}|b-a| \left(\tfrac{1}{2}\omega|\epsilon|_{\Delta_*}^2 + \gamma\tau^{2s} \right) . \tag{7.41}$$

Next we consider arguments $t = t_{ji} \in \Delta_*$. In this case the jumps do occur inside one of the subintegrals. Hence, we have to be more careful in our estimate. We start with

$$\|\epsilon(t)\| \le \sum_{k=1}^{m-1} \Big\| \int_{\sigma=t_k}^{t_{k+1}} G(t,\sigma)\,(\delta f(\sigma) + \delta\varphi(\sigma))\; d\sigma \Big\|$$

from which we immediately see that the integrals over I_k for $k \neq j$ can be treated as before

$$\Big\| \int_{\sigma=t_k}^{t_{k+1}} G(t,\sigma)\,(\delta f(\sigma) + \delta\varphi(\sigma))\; d\sigma \Big\| \le \bar{\rho}\tau_k \left(\gamma\tau_k^{2s} + \tfrac{1}{2}\omega|\epsilon|_{\Delta_*}^2 \right) .$$

For $k = j$, however, we just obtain

$$\left\| \int_{\sigma=t_j}^{t_{j+1}} G(t,\sigma)\left(\delta f(\sigma) + \delta\varphi(\sigma)\right)\, d\sigma \right\| \leq \bar{\rho}\tau_j \left(\max_{\sigma\in I_j} \|\delta f(\sigma)\| + \tfrac{1}{2}\omega|\epsilon|^2_{\Delta_*} \right) .$$

If we recall (7.39) and define the quantity

$$\gamma^*\tau = (\gamma\tau^s + \overline{\gamma}\frac{\tau}{|b-a|}) ,$$

we end up with the quadratic inequality

$$|\epsilon|_{\Delta_*} \leq \bar{\rho}|b-a|(\tfrac{1}{2}\omega|\epsilon|^2_{\Delta_*} + \gamma^*\tau^{s+1}) . \tag{7.42}$$

For the solution of this inequality, we introduce the majorant $|\epsilon|_{\Delta_*} \leq \bar{\epsilon}$ generating the quadratic equation

$$\bar{\epsilon} = \bar{\rho}|b-a|(\tfrac{1}{2}\omega\bar{\epsilon}^2 + \gamma^*\tau^{s+1}) .$$

For the discriminant to be nonnegative we need to require

$$\tau^{s+1} \leq \frac{1}{2\omega\gamma^*(\bar{\rho}|b-a|)^2} ,$$

which is just statement (7.35) of the theorem. This situation is represented graphically in Figure 7.5.

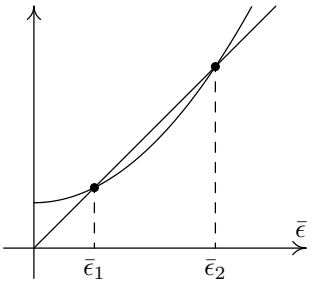

Fig. 7.5. Left and right side of quadratic inequality (7.42).

With the notations

$$\alpha = \frac{1}{\bar{\rho}|b-a|\omega}, \quad \tau_c = \left(2\omega\gamma^*(\bar{\rho}|b-a|)^2\right)^{-\frac{1}{s+1}}$$

we obtain the two majorant roots

$$\bar{\epsilon}_1 = \frac{\alpha(\tau/\tau_c)^{s+1}}{1+\sqrt{1-(\tau/\tau_c)^{s+1}}}\,, \quad \bar{\epsilon}_2 = \alpha\left(1+\sqrt{1-(\tau/\tau_c)^{s+1}}\right) = 2\alpha - \bar{\epsilon}_1\,.$$

For $\tau \in [0, \tau_c[$ we obtain the bounds

$$0 \le \bar{\epsilon}_1 \le \alpha^-\,, \quad 2\alpha \ge \bar{\epsilon}_2 \ge \alpha^+\,.$$

The situation is depicted in Figure 7.6.

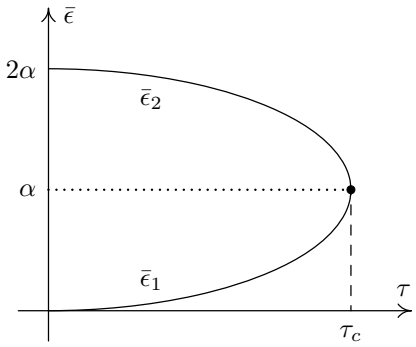

Fig. 7.6. Error bounds for consistent discrete solution ($\bar{\epsilon} \le \bar{\epsilon}_1$) and spurious or 'ghost' solutions ($\bar{\epsilon} \ge \bar{\epsilon}_2$).

First we pick $\bar{\epsilon}_1$, the root consistent with the continuous BVP, and are led to the approximation result

$$|\epsilon|_{\Delta_*} \le \bar{\epsilon}_1 \le 2\bar{\rho}|b-a|\gamma^*\tau^{s+1}\,,$$

which verifies statement (7.36). Second we study the root $\bar{\epsilon}_2$. Again under the meshsize constraint (7.35), the quadratic inequality can be seen to characterize a further *discrete solution* branch by

$$|\epsilon|_{\Delta_*} \ge \bar{\epsilon}_2\,.$$

Obviously, the proof permits the existence of inconsistent discrete solutions. However, if such solutions exist, they are well-separated from the consistent ones as long as $\tau < \tau_c$.

As a final step of the proof, we may just insert the upper bound (7.36) into (7.41) and arrive at

$$\begin{aligned} |\epsilon|_\Delta &\le \bar{\rho}|b-a|\left(\tfrac{1}{2}\omega(\bar{\rho}|b-a|\gamma^*\tau^{s+1})^2 + \gamma\tau^{2s}\right) \\ &\le \bar{\rho}|b-a|\left(\tfrac{1}{2}\omega(\bar{\rho}|b-a|\gamma^*\tau)^2 + \gamma\right)\tau^{2s}\,. \end{aligned}$$

This is the desired superconvergence result (7.37) and thus completes the proof.

Of course, the theorem does not state anything about the situation when the meshsize restriction (7.35) does *not* hold, i.e. for $\tau > \tau_c$. □

The above discussed possible occurrence of *'ghost' solutions*, i.e. of *inconsistent discrete solutions*, which have nothing to do with the continuous solution, has been experienced by computational scientists, both in ODEs and in PDEs. In words, the above theorem states that a computed collocation solution u is a valid approximation of the BVP solution y only, if the applied mesh is 'sufficiently fine' and if this solution 'essentially is preserved' on successively finer meshes. This means that the actual uniqueness structure of a collocation solution cannot be revealed by solving just one finite-dimensional problem. Rather, successive mesh refinement is additionally needed as an algorithmic device to decide about uniqueness. This paves the way to Newton methods in function space, also called *quasilinearization*—to be treated in the next section.

7.4.2 Quasilinearization as inexact Newton method

Instead of a Newton method for the discrete nonlinear system (7.31), (7.32), and (7.33), the popular collocation codes realize some *quasilinearization* technique, i.e., a Newton method in function space. Of course, approximation errors are unavoidable, which is why *inexact* Newton methods in function space are the correct conceptual frame.

We start with the *exact* ordinary Newton iteration in *function space*

$$y_{k+1}(t) = y_k(t) + \delta y_k(t)\,, \quad k = 0, 1, \ldots\,.$$

Herein the Newton corrections δy_k satisfy the linearized BVP, which is the perturbed variational equation with linear separated boundary conditions:

$$\begin{aligned} \delta y_k' - f_y(y_k(t))\delta y_k &= -\left(y_k'(t) - f(y_k(t))\right)\,, \quad t \in [a,b]\,, \\ A\delta y_k(a) + B\delta y_k(b) &= 0\,. \end{aligned}$$

Newton's method in the infinite dimensional function space can rarely be realized, apart from toy problems. Instead we study here the corresponding *exact* Newton method in *finite dimensional space*, i.e., the space spanned by the collocation polynomials on the grids Δ and Δ_* (as introduced in the preceding section). Formally, this method replaces $y, \delta y$ by their polynomial representations $u, \delta u$, where

$$\begin{aligned} \delta u_k'(t_{ji}) - f_y(u_k(t_{ji}))\delta u_k(t_{ji}) &= -\delta f_k(t_{ji}) = -\left(u_k'(t_{ji}) - f(u_k(t_{ji}))\right)\,, \\ A\delta u_k(a) + B\delta u_k(b) &= 0\,. \end{aligned} \tag{7.43}$$

If we include a damping factor λ_k to expand the local domain of convergence of the ordinary Newton method, we arrive at

$$u_{k+1}(t) = u_k(t) + \lambda_k \delta u_k(t)\,. \tag{7.44}$$

Note that here Theorem 7.2 applies with global Lipschitz constant $\omega = 0$. As a consequence there are *no spurious solutions* δu to be expected—which clearly justifies the use of quasilinearization. Nevertheless, the chosen mesh might be not 'fine enough' to represent the solution correctly—see the discussion on mesh selection below.

Linear band system. Within each iteration, dropping the index k, we have to solve the finite-dimensional linear system consisting of the (scaling invariant) local collocation conditions

$$F_{ji} = \tau_j \left(\delta u'(t_{ji}) - f_y(u(t_{ji}))\delta u(t_{ji}) + \delta f(t_{ji})\right) = 0\,,$$

the global collocation conditions

$$F_j = \delta u(t_{j+1}) - \delta u(t_j) - \tau_j \sum_{l=1}^{s} b_l \left(f_y(u(t_{jl}))\delta u(t_{jl}) - \delta f(t_{jl})\right) = 0 \tag{7.45}$$

and the boundary conditions

$$F_m = A\delta x_1 + B\delta x_m = 0\,.$$

In most implementations, the local collocation conditions are realized in the equivalent initial value problem form

$$F_{ji} = \delta u(t_{ji}) - \delta u(t_j) - \tau_j \sum_{l=1}^{s} a_{il} \left(f_y(u(t_{jl}))\delta u(t_{jl}) - \delta f(t_{ji})\right)\,. \tag{7.46}$$

In this case, the local variables $u(t_{j1}), \ldots, u(t_{js})$ can be condensed—which means expressed in terms of the global variables $u(t_j)$. However, unlike the equations (7.45) and (7.43), the part (7.46) is not symmetric with respect to $a \leftrightarrow b$. The separated boundary conditions can be dropped by fixing the proper components of the boundary values.

The remaining linear system has block tridiagonal structure. It is usually solved by some global direct elimination method such as the modified band solver due to J.M. Varah [191].

In the already mentioned Gauss-Newton continuation method for parameter dependent BVPs (cf. [17]), the discretized linear system is just enhanced by a further column, which requires a slight modification of the elimination method; the corresponding rank-deficient Moore-Penrose pseudoinverse is then simply realized via the representation (5.29) as presented in Section 5.2.

Norms. As in Section 7.4.1 above we here also write $\|\cdot\|$ for a local vector norm. With $|\cdot|$ we will mean the canonical C^0-norm defined as

$$|\epsilon|_{0,[a,b]} = \max_{t\in[a,b]} \|\epsilon(t)\|\,,$$

whose discretization is

$$|\epsilon|_{0,\Delta} = \max_{t\in\Delta} \|\epsilon(t)\|\,.$$

If we scale the L_2-norm appropriately, we find that

$$|\epsilon|_{2,[a,b]} = \left(\frac{1}{b-a}\int_a^b \|\epsilon(t)\|^2 dt\right)^{\frac{1}{2}} \leq |\epsilon|_{0,[a,b]}\,.$$

From this, we may turn to the corresponding discretization

$$|\epsilon|_{2,\Delta} = \left(\frac{1}{b-a}\sum_{j=1}^{m-1}\tau_j\sum_{l=1}^{s} b_l\|\epsilon(t_{jl})\|^2\right)^{\frac{1}{2}}$$

and obtain the corresponding relation

$$|\epsilon|_{2,\Delta} \leq |\epsilon|_{0,\Delta^*}\,.$$

Note that, due to Gaussian quadrature, we have the approximation property

$$|\epsilon|_{2,\Delta} = |\epsilon|_{2,[a,b]} + O(\tau^{2s}) \leq |\epsilon|_{0,[a,b]} + O(\tau^{2s})\,.$$

Below we will not distinguish between any of these essentially equivalent norms and write subscripts only where necessary.

Discretization error estimates. The above exact finite dimensional Newton method can also be viewed as an *inexact* Newton method in *function space*, if we include the discretization errors into a unified mathematical frame. In fact, any *adaptive* collocation method will need to control the arising discretization error

$$\epsilon(t) = \delta u(t) - \delta y(t)$$

at least via some estimate of it, in some suitable norm (see above). For convenience, we again use the notation $I_j = [t_j, t_{j+1}]$ for the subintervals of the coarse mesh Δ.

Before we start, let us draw a useful consequence of Theorem 7.2: we interpret the Gaussian collocation method as an implicit Runge-Kutta method (compare, e.g., [115, 71]) and thus immediately obtain (for $k < s$)

$$\delta u^{(k)}(t) - \delta y^{(k)}(t) = \delta y^{(s+1)}(t_j)\tau_j^{s+1}P_s^{(k)}\left(\frac{t-t_j}{\tau_j}\right)(1+O(\tau_j)) + O(\tau_j^{2s})\,, \quad (7.47)$$

where P is a polynomial of degree s—see, e.g., [9, Section 9.3]. Since the local collocation polynomials are of order s, their derivative $\delta u^{(s)}$ is piecewise constant, i.e.,

$$\delta u^{(s)}(t) = \text{const}_j\,, \quad \delta u^{(s+1)}(t) = 0\,, \quad t \in I_j\,.$$

At first glance, we do not seem to have a reasonable approximation of $\delta y^{(s+1)}$ that could be used to estimate the error bound (7.47). However, we may overcome this lack of information by defining piecewise linear functions δz_s via the interpolation conditions at the subinterval midpoints $\bar{t}_j = t_j + \frac{1}{2}\tau_j$ such that

$$\delta z_s(\bar{t}_j) = \delta u^{(s)}(\bar{t}_j)\,.$$

The situation is depicted schematically in Figure 7.7. The derivative $\delta z_s'$ is piecewise constant with jumps at the subinterval midpoints $\bar{t}_j$.

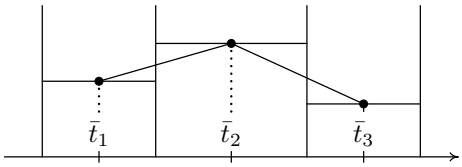

Fig. 7.7. Piecewise linear approximation δz_s of piecewise constant function $\delta u^{(s)}$

Since $P_s^{(s)}(\frac{1}{2}) = 0$, (7.47) implies the superconvergence result

$$\delta u^{(s)}(\bar{t}_j) - \delta y^{(s)}(\bar{t}_j) = O(\tau_j^{2s})\,,$$

from which we obtain the approximation property

$$\delta z_s'(t) = \delta y^{(s+1)}(t_j) + O(\tau_j)\,, \quad t \in I_j\,.$$

Hence, in first order of the local mesh size τ_j, we obtain the componentwise estimate

$$|\delta u(t) - \delta y(t)| \doteq |\delta z_s'(t_j)|\tau_j^{s+1}\,, \quad t \in I_j\,.$$

If we average according to the rule

$$|\delta z_s'(\bar{t}_1)| = |\delta z_s'(t_1)|\,, \quad |\delta z_s'(\bar{t}_m)| = |\delta z_s'(t_m)|\,,$$
$$|\delta z_s'(\bar{t}_j)| = \tfrac{1}{2}\left(|\delta z_s'(t_j)| + |\delta z_s'(t_{j+1})|\right)\,, \quad j = 2, \ldots, m-1\,,$$

we arrive at the cheaply computable local discretization error estimates

$$\max_{t \in I_j} |\delta u(t) - \delta y(t)| \doteq \epsilon_j = |\delta z_s'(\bar{t}_j)|\tau_j^{s+1}\,. \tag{7.48}$$

In [120], A. Hohmann has suggested a realization with variable order $p_j = 2s_j$ in different subintervals I_j; for such an $h-p$-strategy, the above estimation

technique will no longer be applicable. Instead, the Gauss interpolation polynomial is extended by the additional boundary nodes t_j, t_{j+1}. This kind of collocation polynomial is of order $s_j + 2$. Its difference to the computed solution can then serve as an estimate of order $s_j + 1$ replacing (7.48).

Global mesh selection. This issue is crucial in every global discretization method, at least when the numerical solution of really challenging spacelike BVPs is envisioned, including the possible occurrence of internal or boundary layers. Typically, a starting mesh Δ will be defined at the beginning of the discretization process, which may already contain some information about the expected behavior of the solution. Within each quasilinearization step the componentwise global discretization error estimate

$$|\epsilon| = \max_{j=1,..,m-1} \epsilon_j$$

can be computed as indicated above. In order to minimize this maximum subject to the constraint

$$\sum_{j=1}^{m-1} \tau_j = b - a\,,$$

the well-known greedy algorithm leads to the requirement of *equidistribution* of the local discretization errors. Therefore, any reasonable mesh selection device will aim, at least asymptotically, at

$$\epsilon_j = C_j \tau_j^{s+1} \approx \text{const}\,, \quad j = 1, \ldots, m-1$$

with coefficients C_j as defined above; an equivalent formulation is

$$\epsilon_j \approx \overline{\epsilon} = \frac{1}{m-1} \sum_{l=1}^{m-1} \epsilon_l\,.$$

There are various options to realize this equidistribution principle. In the code family `COLSYS` the number m of nodes is adapted such that the global error estimate eventually satisfies

$$|\epsilon| \le \text{TOL}$$

in terms of the user prescribed error tolerance TOL. Some codes also permit nonnested successive meshes.

In view of the theoretical approximation results (as given, e.g., in the previous section), `COLSYS` uses a further *global* mesh refinement criterion based on an affine covariant Newton method including a damping strategy [60, 63, 9]. Whenever the damping factor turns out to be 'too small', i.e., whenever $\lambda_k < \lambda_{\min}$ occurs, for some prescribed threshold value $\lambda_{\min} \ll 1$, then a new mesh with precisely halved local stepsizes is generated. Note that in Newton

methods, which are *not* affine covariant, such a criterion may be activated in the wrong situation: not caused by the nonlinearity of the problem, but by the ill-conditioning of the discrete equations—which automatically comes with the fact that the BVP operator is noncompact.

Local mesh refinement. Instead of the mesh selection devices realized within `COLSYS` we here want to work out an alternative option in the spirit of *adaptive multilevel methods* for partial differential equations (to be treated in Section 8.3 below). Let

$$\Delta_0 \subset \Delta_1 \subset \cdots \subset \Delta_d$$

denote a sequence of *nested* meshes. By construction, the mesh Δ_0 is just the given initial mesh, while the mesh Δ_1 is obtained by halving of all subintervals. All further meshes can be obtained using an adaptivity device based on *local extrapolation*—a technique that has been suggested by I. Babuška and W.C. Rheinboldt [12] already in 1978, there for finite element methods in partial differential equations. For details we here recur to Section 9.7.1 of the elementary textbook [77], where this technique is explained in the simple context of numerical quadrature. Assume we have already computed local error estimates on two consecutive meshes, on the given mesh Δ and its coarser predecessor Δ^-. Let $I := (t_l, t_m, t_r) \in \Delta^-$ denote an interval bisected into subintervals $I_l, I_r \in \Delta$, where

$$I_l := \big(t_l, \tfrac{1}{2}(t_l + t_m), t_m\big) \quad \text{and} \quad I_r := \big(t_m, \tfrac{1}{2}(t_r + t_m), t_r\big) \ .$$

Repeated refinement leads to a recursive *binary tree*—see Figure 7.8.

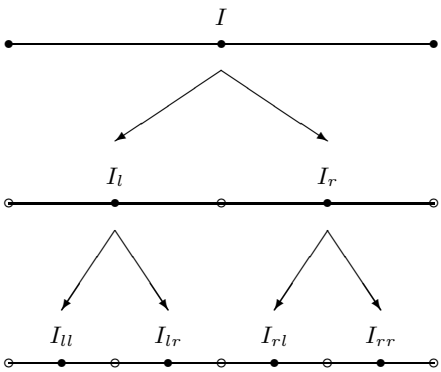

Fig. 7.8. Double refinement of subinterval $I := (t_l, t_m, t_r)$

In contrast to our above approximation results, we make the following more general assumption for the local discretization error

$$\epsilon(I_j) \doteq C\tau_j^\gamma \tag{7.49}$$

with a local order γ and a local problem dependent constant C to be roughly identified; note that above we derived $\gamma = s$ in the upper bounds, which may be unrealistic in a specific example. Let I be a subinterval obtained by refinement; then we denote the starting interval from the previous mesh by I^-, i.e., $I_r^- = I_l^- = I$. Dropping the local index j, assumption (7.49) then implies

$$\epsilon(I^-) \doteq C(2\tau)^\gamma = 2^\gamma C\tau^\gamma \doteq 2^\gamma \epsilon(I)\,,$$

from which we conclude that

$$\epsilon(I_l) \doteq C\tau^\gamma 2^{-\gamma} \doteq \epsilon(I)\epsilon(I)/\epsilon(I^-)\,.$$

Thus, through *local extrapolation*, we have obtained a local *error prediction*

$$\epsilon^+(I) := \frac{\epsilon^2(I)}{\epsilon(I^-)} \approx \varepsilon(I_l)\,.$$

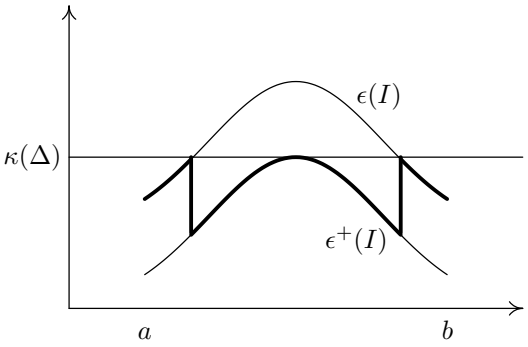

Fig. 7.9. Error distributions: before refinement: $\epsilon(I)$, prediction after *global* refinement: $\epsilon^+(I)$, prediction after *adaptive* refinement: bold line.

We can therefore estimate in advance, what effect a refinement of an interval $I \in \Delta$ would have. We only have to fix a *threshold value* for the local errors, above which we refine an interval. In order to do this, we take the maximal local error, which we would obtain from a *global refinement*, i.e., refinement of *all* subintervals $I \in \Delta$, and define

$$\kappa(\Delta) := \max_{I\in\Delta} \epsilon^+(I)\,.$$

In order to illustrate the situation, we plot the computed estimated errors $\epsilon(I)$ together with the predicted errors $\epsilon^+(I)$ in a smoothed histogram, see Figure 7.9.

Following the aim of local error equidistribution, we do not need to refine near the left and right boundary; rather, refinement will pay off only in the center region. We thus arrive at the following *refinement rule*: Refine only those intervals $I \in \Delta$, for which

$$\epsilon(I) \geq \kappa(\Delta)\,.$$

This yields the error distribution displayed in bold line in Figure 7.9, which is narrower than both the original distribution and the one to be expected from global refinement.

Clearly, repeated refinement will ultimately generate some rough error equidistribution, provided the local error estimation technique is reliable, which here means that the finest meshes need to be 'fine enough' to activate the superconvergence properties of Gauss collocation methods.

Error matching. If we view quasilinearization as an inexact Newton method in function space, we need to match the Newton corrections δu_k and the discretization errors $\delta u_k - \delta y_k$. In his dissertation [120], A. Hohmann worked out a residual based inexact Newton method using a Fredholm basis for the local representation of the collocation polynomials and obtained some rather robust algorithm. Here, however, we want to realize the *error oriented* local (Section 2.1.5) and global Newton methods (Section 3.3.4) within the collocation code `COLSYS`—transferring the finite dimensional case therein to the present infinite dimensional one.

At each iteration index k, identify $u_k = y_k$—as a common starting point, say—and define the two Newton corrections in function space

$$F'(u_k)\delta u_k = -F(u_k) + r_k\,, \quad F'(u_k)\delta y_k = -F(u_k)\,. \tag{7.50}$$

In contrast to the setting in finite dimensional inexact Newton methods (see Chapters 2 and 3), here the residual r_k is not generated by some inner iteration, but by the discretization error such that

$$F'(u_k)(\delta u_k - \delta y_k) = r_k\,.$$

In collocation, the situation is characterized by the fact that we have cheap computational estimates of the relative *discretization error*

$$\delta_k = \frac{|\delta u_k - \delta y_k|}{|\delta u_k|}$$

available in some (approximate) norm $|\cdot|$—compare (3.50). This quantity essentially depends on the selected mesh. For successively fine meshes, this quantity will approach zero, which is part of the asymptotic mesh independence to be discussed in detail in Section 8.1 below (see also Exercise 8.3).

The iteration (7.44) is realized as a finite dimensional Newton iteration with adaptive trust region (or damping) strategy—as long as the mesh is kept

unchanged. However, in order to really solve the BVP and not just some discrete substitute system with unclear approximation quality, the mesh should be adapted along with the iteration: the idea advocated here is to aim at some asymptotic confluence with an exact Newton iteration based on the correction δy_k. In view of (3.55) and the analysis in Section 3.3.4, we will require that, for some $\rho \leq 1$,

$$\delta_k \leq \frac{\rho}{2(1+\rho)} \leq \tfrac{1}{4} \text{ for } \overline{\lambda}_k < 1\,. \tag{7.51}$$

As soon as the iteration swivels in the *ordinary* Newton phase with $\lambda = 1$ throughout, then a more careful consideration is needed, which can be based on the following theoretical estimates.

Theorem 7.3 *Let δu_k and δy_k denote the inexact and exact Newton corrections as defined in* (7.50)*. Let ω be the affine covariant Lipschitz constant defined via*

$$|F'(u)^{-1}\,(F'(v) - F'(w))\,| \leq \omega |v - w|\,.$$

Then, with the notation of the present section, we obtain

(I) *for the ordinary exact Newton method in finite dimension the quadratic convergence result*

$$|\delta u_{k+1}| \leq \tfrac{1}{2} h_k^\delta |\delta u_k|$$

with $h_k^\delta = \omega |\delta u_k|$,

(II) *for the stepwise 'parallel' ordinary inexact Newton method in function space, as defined in* (7.50)*, the mixed convergence results*

$$|\delta y_{k+1}| \leq \frac{\frac{1}{2} h_k^\delta + (1 + h_k^\delta)\delta_k}{1 - \delta_k} |\delta y_k|$$

and, under the additional matching assumption for some safety factor $\rho \leq 1$,

$$\delta_k \leq \tfrac{1}{2} \rho \frac{h_k^\delta}{1 + h_k^\delta}\,, \tag{7.52}$$

the modified quadratic convergence results

$$|\delta y_{k+1}| \leq \tfrac{1}{2}(1+\rho) \frac{h_k^\delta}{1 - \delta_k} |\delta y_k|\,,$$

where contraction is realized, if $h_0^\delta < \dfrac{2\,(1 - \delta_0)}{1 + \rho}$.

Proof. Part (I) of the theorem is standard. Part (II) is a slight modification of Theorem 2.11 in Section 2.1.5, there derived for the inner iterative solver `GBIT`, which does not satisfy any orthogonality properties. The definition of

h_k^δ is different here; it can be directly inserted into the intermediate result (2.54). □

Clearly, we will select the same value of ρ in both (7.51) for the damped Newton method and (7.52) for the ordinary Newton method.

Recall that the derivation of (7.51) required that λ_k needs to be chosen according to an adaptive trust region strategy based on computationally available Kantorovich estimates $[h_k^\delta] \le h_k^\delta$. We are still left with the construction of such estimates. Since the right hand upper bound in (7.52) is a monotone increasing function of h_k^δ, we may then construct an adaptive matching strategy just replacing h_k^δ by its lower bound $[h_k^\delta]$. From the above lemma we directly obtain the a-posteriori estimates

$$[h_k^\delta]_1 = 2\Theta_k \le h_k^\delta \,, \text{ where } \Theta_k = \frac{|\delta u_{k+1}|}{|\delta u_k|}$$

and the a-priori estimate

$$[h_k^\delta] = 2\Theta_{k-1}^2 \le h_k^\delta \,.$$

With these preparations we are led to the following informal

Error matching algorithm. As long as the finite dimensional global Newton method is still damped, we realize $\delta_k \le 1/4$ via appropriate mesh selection as given above. Let the index $k = 0$ characterize the beginning of the local Newton method with $\lambda_0 = 1$. Then the following steps are required (skipping emergency exits to avoid infinite loops):

1. $k = 0$: Given u_0, compute δu_0 and its norm $|\delta u_0|$. Compute the discretization error estimate δ_0, e.g., via the suggestion (7.48).
2. **If** $\delta_0 > \frac{1}{4}$, then refine the mesh and goto 1,
   else $u_1 = u_0 + \delta u_0$.
3. $k \ge 1$: Given u_k, compute δu_k, its norm $|\delta u_k|$ and the contraction factor
   $$\Theta_{k-1} = \frac{|\delta u_k|}{|\delta u_{k-1}|} \,.$$
   If $\Theta_{k-1} > 1$, then realize an adaptive trust region strategy with $\lambda < 1$ as described in Section 3.3.4,
   else compute the discretization error estimate δ_k.
4. **If**
   $$\delta_k > \min\left(\frac{\rho}{2(1+\rho)}, \frac{\rho\Theta_{k-1}^2}{1+\Theta_{k-1}^2}\right) \,,$$
   then refine the mesh and goto 3,
   else $u_{k+1} = u_k + \delta u_k \to u_k$ and goto 3.

If the adaptive collocation algorithm is performed including the matching strategy as described here, the amount of work will clearly increase from step to step, since the required discretization error estimate will ask for finer and finer (adaptive) meshes. Of course, the process may be modified such that at some iterate the value of δ_k is frozen—with the effect of eventually obtaining unreasonably accurate results u_k.

Remark 7.2 An adaptive collocation method as sketched here has not been implemented so far. However, this kind of ideas has entered into the adaptive multilevel collocation method used within a function space complementarity approach to constrained optimal control problems that has recently been suggested and worked out by M. Weiser and P. Deuflhard [197].

Exercises

Exercise 7.1 Given the extended mapping (7.12), derive the expressions (7.14) for the associated Newton corrections. Sketch details of the corresponding adaptive trust region method. In which way is a control of the actual parameter stepsize performed? Why should the initial iterate τ^0 satisfy $h(\tau^0) \neq 0$?

Exercise 7.2 In order to compute a periodic orbit, one may apply a gradient method [156] as an alternative to the Gauss-Newton method described in Section 7.3. Let, in a single shooting approach, the functional

$$\varphi := \tfrac{1}{2}\|r\|_2^2$$

be minimized. Then

$$\operatorname{grad}\varphi = [E(0),\ f(y(T))]^T r$$

in the autonomous case. Show that

$$E^T r = u(0) - u(T)$$

for some u satisfying

$$u' = -f_y(y)^T u\,, \quad u(T) = r\,.$$

Discuss the computational consequences of this relation. Why is, nevertheless, such a gradient method unsatisfactory?

Exercise 7.3 In multiple shooting techniques, assume that local rank reduction leads to a replacement of the Jacobian inverse J^{-1} by the generalized inverse J^- defined in (7.5).

a) Verify the four axioms (7.6) using the Penrose axioms for the Moore-Penrose pseudo-inverse.

b) In the notation of Section 4.3.5, let $\lambda_k \sim [\omega_k]^{-1}$ denote a damping factor estimate associated with the Jacobian inverse J_k^{-1} and $\lambda'_k \sim [\omega'_k]^{-1}$ the analog estimate associated with the generalized inverse J_k^-. Give a computationally economic expression for λ'_k. Show that, in general, the relation $\lambda'_k > \lambda_k$ need not hold, unless $m = 2$.

Exercise 7.4 Consider a local rank reduction in multiple shooting techniques as defined by J^- in (7.5) and the axioms (7.6). Show that for the Gauss-Newton direction

$$\Delta x = -J^- F \,.$$

the residual level function

$$T(x|I) := \|r\|_2^2 + \sum_{j=1}^{m-1} \|F_j\|_2^2$$

is, in general, no longer an appropriate *descent function* in the sense of Lemma 3.11, whereas both the natural level function $T(x|J^-)$ and the hybrid level function $T(x|R^{-1}J^-)$ still are.

Exercise 7.5 Consider a singular perturbation problem of the type

$$\varepsilon y'' + f(t)y' + g(t)y = h(t)\,, \quad y(0) = y_0\,, \quad y(T) = y_T$$

having one internal layer at some $\tau \in]0, T[$ with $f(\tau) = 0$. Assume that the initial value problem is well-conditioned from τ to 0 and from τ to T so that numerical integration in these directions can be conveniently performed. Consider a multiple shooting approach with $m = 3$ nodes $\{0, \tau, T\}$ that involves numerical integration in the well-conditioned directions. Study the associated Newton method in detail with respect to necessary Jacobian approximations, condensed linear system solver, and iterative refinement sweeps. Discuss extensions for $m > 3$, where the above 3 nodes are among the selected multiple shooting nodes. Interpret this approach in the light of Lemma 7.1.

Exercise 7.6 Consider the Fourier collocation method (also: Urabe or harmonic balance method) as presented in Section 7.3.3.

a) Given the asymptotic decay law

$$\varepsilon_m \doteq Ce^{-\gamma m}\,,$$

verify the computationally available estimate (7.30) for the optimal number m^* of terms needed in the Fourier series expansion (7.24). How many

different Galerkin-Urabe approximations are at least required for this estimate?

b) In the derivation of Section 7.3.3 we ignored the difference between the Galerkin approximation y_m with Fourier coefficients a_j, b_j and the actually computed Galerkin-Urabe approximation with coefficients a_j^m, b_j^m depending on the truncation index m. Modify the error estimates so that this feature is taken into account.

Exercise 7.7 Given a perturbed variational equation in the form

$$\epsilon'(t) - f_y(y(t))\epsilon(t) = \delta f(t)\,, \quad t \in [a,b]\,,$$

prove the closed analytic expression

$$\epsilon(t) = W(t,a)\epsilon(a) + \int_{\sigma=a}^{t} W(t,\sigma)\delta f(t) d\sigma\,.$$

Hint: Apply the variation of constants method recalling that the propagation matrix $W(t,a)$ is a solution of the unperturbed variational equation.