Eidgenössische
Technische Hochschule
Zürich

Departement Informatik
Institut für
Wissenschaftliches Rechnen

Walter Gander
Dominik Gruntz

Derivation of Numerical Methods using Computer Algebra

July 1998

305

ETH Zürich
Departement Informatik
Institut für Wissenschaftliches Rechnen
Prof. Dr. W. Gander

Walter Gander
Institut für Wissenschaftliches Rechnen
Eidgenössische Technische Hochschule
CH-8092 Zürich
e-mail: gander@inf.ethz.ch

Dominik Gruntz
OBERON Microsystems AG
Technoparkstrs. 1
CH-8005 Zürich
e-mail: gruntz@oberon.ch

This report is also available via anonymous ftp from ftp.inf.ethz.ch
as doc/tech-reports/1998/305.ps.

Derivation of Numerical Methods using Computer Algebra

Walter Gander
Institut für Wissenschaftliches Rechnen
Eidgenössische Technische Hochschule
CH-8092 Zürich
gander@inf.ethz.ch

Dominik Gruntz
OBERON Microsystems AG
Technoparkstrs. 1
CH-8005 Zürich
gruntz@oberon.ch

July 1998

1 Introduction

At ETH Zürich we have redesigned our former courses on numerical analysis. We do not only run numerical programs but also introduce the students to computer algebra and make heavy use of computer algebra systems both in the lectures and the assignments.

Computer algebra may be used to generate numerical algorithms, to compute discretization errors, to simplify proofs, etc., but also to run examples and to generate plots.

We claim, that it is easier for students to follow a derivation which is carried out with the help of a computer algebra system than by hand. Computer algebra systems take over the hard hand work such as e.g. solving systems of equations. Students do not need to be concerned with all the details (and all the small glitches) of a manual derivation and can understand and keep the overview over the general steps of the derivation. A computer supported derivation is also more convincing than a presentation of the bare results without any reasoning.

Moreover, using computer algebra systems rather complex numerical formulas can be derived, far more complex than what can be done in class by hand. E.g. all useful Newton-Cotes rules can be computed without problems, in contrast to hand derivations, which usually end with Simpson's rule.

We will prove these statements with some examples taken from our introductory courses in scientific computing. We use Maple V Release 4, but the examples could also be reproduced e.g. with Mathematica, MuPad or any other computer algebra system.

2 Newton's Formula

One of the first formulas that students learn is Newton's iteration to solve a nonlinear equation $f(x) = 0$. Given an approximation x_k for the root s, a better approximation $x_{k+1} = F(x_k)$ can be obtained using the iteration function

```
> F := x -> x - f(x) / D(f)(x);
```

$$F := x \to x - \frac{\mathrm{f}(x)}{\mathrm{D}(f)(x)} \tag{1}$$

This iteration converges quadratically to a *single* root s of $f(x)$. This can be proven by computing the first derivative of $F(x)$ at $x = s$ which is zero.

```
> dF := D(F)(x);
```

$$dF := \frac{\mathrm{f}(x)\, D^{(2)}(f)(x)}{\mathrm{D}(f)(x)^2} \tag{2}$$

```
> subs(f(s) = 0, D(F)(s));
```

$$0$$

```
> subs(f(s) = 0, D(D(F))(s));
```

$$\frac{D^{(2)}(f)(s)}{\mathrm{D}(f)(s)}$$

The arbitrary precision floating point arithmetic, which is provided by most computer algebra systems, can be used to demonstrate what quadratic convergence means "in real life". As example, we compute the square root of 9 using Newton's iteration to solve equation $x^2 - 9 = 0$ starting with $x_0 = 1$. As expected, the number of correct digits doubles with each iteration.

```
> f := x -> x^2 - 9:
> Digits := 80:
> xk := 1.0: to 8 do xk := F(xk); lprint(xk); od:
```

```
5.0000000000000000000000000000000000000000000000000000000000000000000000000000000
3.4000000000000000000000000000000000000000000000000000000000000000000000000000000
3.0235294117647058823529411764705882352941176470588235294117647058823529411764706
3.0000915541313801785305561913481345845731288624399176012817578393224994277866789
3.0000000013969838622484784252588819771210853888469481349100703687663353906866013
3.0000000000000000003252606517456513302198682555233168260482212782050131565335491
3.0000000000000000000000000000000000001763241526233431261953104805833368516733180
3.0000000000000000000000000000000000000000000000000000000000000000000000000000518
```

The convergence result is only valid for single roots, as the first derivative of f appears in the denominator of (2), i.e. the result is only valid if $f'(s) \neq 0$. The behavior of Newton's iteration for an equation with multiple roots is the next topic we want to discuss using Maple. Let us assume that $f(x)$ has a zero of multiplicity n at $x = s$. We therefore define $f(x)$ to be

```
> f := x -> (x-s)^n * g(x);
```

$$f := x \to (x-s)^n\, \mathrm{g}(x)$$

where $g(s) \neq 0$. Again we inspect the first derivative of $F(x)$. If $F'(s) \neq 0$ then the iteration converges only linearly.

```
> dF;
```

$$\frac{(x-s)^n\, \mathrm{g}(x) \left(\frac{(x-s)^n\, n^2\, \mathrm{g}(x)}{(x-s)^2} - \frac{(x-s)^n\, n\, \mathrm{g}(x)}{(x-s)^2} + 2\, \frac{(x-s)^n\, n\, \mathrm{D}(g)(x)}{x-s} + (x-s)^n\, D^{(2)}(g)(x) \right)}{\left(\frac{(x-s)^n\, n\, \mathrm{g}(x)}{x-s} + (x-s)^n\, \mathrm{D}(g)(x) \right)^2}$$

Taking the limit of the above expression for $x \to s$ we obtain:

```
> limit(", x=s);
```

$$\frac{n-1}{n}$$

We have just proven, that Newton's iteration converges linearly with the factor $(n-1)/n$ if $f(x)$ has a zero of multiplicity n. Thus e.g. convergence is linear with the factor $1/2$ for double roots.

Newton's iteration also has a nice geometrical interpretation. Starting with the approximation x_k, the next value x_{k+1} of the iteration is the intersection of the tangent at $f(x)$ in $[x_k, f(x_k)]$ with the x-axes. This property can also be proven with Maple. We set up an equation $p(x) = a\,x + b$ for the tangent line. $p(x)$ must interpolate $[x_k, f(x_k)]$ and must have the same derivative as $f(x)$ at $x = x_k$. With these two conditions the parameters a and b of the tangent $p(x)$ are defined.

```
> f := 'f':
> p := x -> a*x + b:
> solve({p(x[k]) = f(x[k]), D(p)(x[k]) = D(f)(x[k])}, {a, b});
```

$$\{b = -\mathrm{D}(f)(x_k)\,x_k + \mathrm{f}(x_k),\ a = \mathrm{D}(f)(x_k)\}$$

We have claimed that the intersection of the tangent $p(x)$ with the x-axes is the next Newton approximation. If the equation $p(x) = 0$ is solved again the iteration function (1) is obtained. This proves that the geometrical interpretation is correct.

```
> assign('');
> x[k+1] = expand(solve(p(t) = 0, t));
```

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

3 Secant Method

The secant method (3) approximates the derivative which appears in Newton's formula by a finite difference.

```
> F := (u, v) -> u - f(u) * (u-v) /  (f(u) - f(v));
```

$$F := (u,\ v) \to u - \frac{\mathrm{f}(u)\,(u-v)}{\mathrm{f}(u) - \mathrm{f}(v)}$$

```
> x[k+1] = F(x[k], x[k-1]);
```

$$x_{k+1} = x_k - \frac{\mathrm{f}(x_k)\,(x_k - x_{k-1})}{\mathrm{f}(x_k) - \mathrm{f}(x_{k-1})} \tag{3}$$

As with Newton's method, we want to analyse the convergence with Maple. Using equation (3) we obtain for the error $e_{k+1} = x_{k+1} - s$ the recurrence

$$e_{k+1} = F(s + e_k, s + e_{k-1}) - s.$$

The right hand side can be expanded into a multivariate Taylor series at $e_k = 0$ and $e_{k+1} = 0$. We assume that s is a single root, $(f'(s) \neq 0)$ and also $f''(s) \neq 0$ must hold. We set $f(s) = 0$ and compute the first term of the Taylor series expansion:

```
> f(s) := 0:
> e2 = normal(readlib(mtaylor)(F(s + e1, s +e0) - s, [e0, e1], 4));
```

$$e2 = \frac{1}{2}\,\frac{e0\,(D^{(2)})(f)(s)\,e1}{\mathrm{D}(f)(s)}$$

If we divide this leading coefficient by e_0 and e_1, we see that the limit of the quotient $e_2/e_0/e_1$ is a constant different from zero. We assume that the convergence coefficient is p and substitute $e_2 = K\,{e_1}^p$ and $e_1 = K\,{e_0}^p$.

```
> "/e1/e0;
```

$$\frac{e2}{e1\ e0} = \frac{1}{2}\,\frac{(D^{(2)})(f)(s)}{\mathrm{D}(f)(s)}$$

```
> expand(subs(e2 = K*e1^p, e1 = K*e0^p, "/K^p));
```

$$\frac{(e0^p)^p}{e0^p\ e0} = \frac{1}{2}\,\frac{(D^{(2)})(f)(s)}{K^p\,\mathrm{D}(f)(s)}$$

This equation is valid for all errors e_0. Since the right hand side is constant, the left hand side must also be independent of e_0. This is only the case, if the exponent of the power of $e0$ is zero. This condition is an equation for p which solved gives the well known convergence factor $p = (1+\sqrt{5})/2$ for the secant method.

```
> solve(ln(lhs(")), p);
```

$$\frac{1}{2}\sqrt{5}+\frac{1}{2},\ \frac{1}{2}-\frac{1}{2}\sqrt{5}$$

4 A New Iteration Formula

Having considered the Newton and the secant method, we now want to derive and analyze a new iteration method (a combination of the Newton and the secant method) to compute a root of a nonlinear equation. The new method shall use the function values and the first derivatives at two points. These four data define a degree three (Hermite) interpolation polynomial. A zero of this polynomial can be taken as next approximation of the root. Unfortunately, the explicit expression for this zero is rather complex, therefore we propose to use *inverse* interpolation for the given data. We then only need to evaluate the resulting polynomial at $y = 0$ to obtain the new approximation for the root. In Maple, this is done with the following commands.

```
> p := x -> a*x^3 + b*x^2 + c*x + d:
> solve({p(-f(x0))=x0, D(p)(-f(x0))=-1/D(f)(x0),
          p(-f(x1))=x1, D(p)(-f(x1))=-1/D(f)(x1)},
      {a,b,c,d}):
> assign(");
> p(0);
```

$$\begin{aligned}&(-\mathrm{f}(x1)^3\,\mathrm{D}(f)(x1)\,\mathrm{f}(x0) + \mathrm{f}(x1)^3\,x0\,\mathrm{D}(f)(x1)\,\mathrm{D}(f)(x0) - 3\,\mathrm{f}(x1)^2\,x0\,\mathrm{D}(f)(x1)\,\mathrm{D}(f)(x0)\,\mathrm{f}(x0)\\ &\quad + \mathrm{f}(x1)^2\,\mathrm{D}(f)(x1)\,\mathrm{f}(x0)^2 - \mathrm{f}(x1)^2\,\mathrm{D}(f)(x0)\,\mathrm{f}(x0)^2 + \mathrm{f}(x1)\,\mathrm{f}(x0)^3\,\mathrm{D}(f)(x0)\\ &\quad + 3\,\mathrm{f}(x1)\,x1\,\mathrm{D}(f)(x1)\,\mathrm{D}(f)(x0)\,\mathrm{f}(x0)^2 - \mathrm{f}(x0)^3\,x1\,\mathrm{D}(f)(x1)\,\mathrm{D}(f)(x0)) \Big/ ((\mathrm{f}(x1) - \mathrm{f}(x0))\\ &\quad \mathrm{D}(f)(x1)\,\mathrm{D}(f)(x0)\,(\mathrm{f}(x1)^2 - 2\,\mathrm{f}(x0)\,\mathrm{f}(x1) + \mathrm{f}(x0)^2))\end{aligned}$$

The resulting expression is still not very simple However, if the evaluation of f and f' is very expensive, it may still pay off since the convergence rate is 2.73 as we will see. For the convergence analysis we expand

$$e_{k+1} = F(s + e_k, s + e_{k-1}) - s.$$

into a multivariate Taylor series at $e_k = 0$ and $e_{k+1} = 0$ as we have done for the secant method.

```
> F := unapply(", x0, x1):
> f(s) := 0:
> e2 = readlib(mtaylor)(F(s+e0, s+e1)-s, [e0,e1], 8):
> eq := normal(");
```

$$eq := e2 = \frac{1}{24} e1^2 \left(-10\,(D^{(3)})(f)(s)\,\mathrm{D}(f)(s)\,(D^{(2)})(f)(s) + 15\,(D^{(2)})(f)(s)^3 + (D^{(4)})(f)(s)\,\mathrm{D}(f)(s)^2\right) e0^2 \Big/ \mathrm{D}(f)(s)^3$$

As before with the secant and Newton method we consider only single roots, i.e., we assume that $D(f)(s) \neq 0$. If this condition holds, then the above equation tells us that in the limit

```
> e2/e0^2/e1^2 = const;
```

$$\frac{e2}{e0^2\,e1^2} = const$$

Let us again introduce the convergence coefficient p and make the following substitutions:

```
> subs(e2=K*e1^p, e1=K*e0^p, ");
```

$$\frac{(K\,e0^p)^p}{K\,e0^2\,(e0^p)^2} = const$$

```
> simplify(", assume=positive);
```

$$K^{(-1+p)}\,e0^{(p^2-2-2\,p)} = const$$

This equation must hold for all errors e_0. Since K, p and $const$ are all constant, the exponent of e_0 must be zero.

```
> solve(p^2 - 2*p - 2 = 0, p);
```

$$1+\sqrt{3},\ 1-\sqrt{3}$$

```
> evalf(["]);
```

$$[2.732050808, -.732050808]$$

Thus, the convergence factor is $p = 1+\sqrt{3}$ and we have super-quadratic convergence.

With the help of Maple we want to demonstrate the above convergence rate for an example. We use our algorithm to compute the zero of the function $f(x) = e^x + x$ starting with $x_0 = 0$ and $x_1 = 1$. For every iteration we print the number of correct digits (first column) and its ratio to the number of correct digits in the previous step (second column). This ratio should converge to the convergence rate $p = 2.73$. We see that this is the case.

```
> f := x -> exp(x) + x;
```

$$f := x \to e^x + x$$

```
> solve(f(x)=0, x);
```

$$-\mathrm{LambertW}(1)$$

```
> Digits := 500:
> x0 := 0.0: x1 := 1.0:
  for i to 6 do x2 := evalf(F(x0,x1));
     d2 := evalf(log[10](abs(x2+LambertW(1)))):
     if i = 1 then lprint(evalf(d2,20))
     else lprint(evalf(d2,20), evalf(d2/d1,20))
     fi;
     x0 := x1: x1 := x2: d1 := d2:
  od:

-2.7349672475721192576
-7.8214175487946676893   2.8597847216407742880
-23.118600850923949542   2.9558070140989569391
-63.885801375143936026   2.7633939349141321183
-176.01456902838525572   2.7551437915728687417
-481.80650538330786806   2.7373103717659224742
```

5 Newton-Cotes Rules

Simpson's rule is a well known quadrature rule for approximating a definite integral

$$\int_a^b f(x)\,dx \approx (b-a)\left(\frac{1}{6}f(a)+\frac{2}{3}f\left(\frac{a+b}{2}\right)+\frac{1}{6}f(b)\right)$$

using three equidistant function values at the endpoints and in the middle of the integration interval. The formula is obtained *by interpolating the three function values and by computing the integral of the interpolating polynomial of degree two.* The following Maple statements may be used to derive Simpson's rule. We first define a polynomial of degree two. We then state the interpolation conditions and solve the resulting linear system for the coefficients of the polynomial. Finally we integrate the polynomial and simplify the result.

```
> p := x-> a0 + a1*x + a2*x^2:
> solve( {p(a)=f(a), p((a+b)/2)=f((a+b)/2), p(b)=f(b)}, {a0,a1,a2}):
> assign(");
> factor(int(p(x), x=a..b));
```

$$-\frac{1}{6}\left(\mathrm{f}(a)+4\,\mathrm{f}\left(\frac{1}{2}\,a+\frac{1}{2}\,b\right)+\mathrm{f}(b)\right)(a-b)$$

What is the error of this integration rule? Discretization errors can be computed very simply by appropriate series expansions. Let $h=(b-a)/2$, then we obtain

```
> En := subs(b = a+2*h, int(f(x), x=a..b) - ");
```

$$En := \int_a^{a+2\,h}\mathrm{f}(x)\,dx-\frac{1}{3}\left(\mathrm{f}(a)+4\,\mathrm{f}(a+h)+\mathrm{f}(a+2\,h)\right)h$$

```
> series(En, h);
```

$$-\frac{1}{90}\left(D^{(4)}\right)(f)(a)\,h^5+\mathrm{O}(h^6)$$

This shows that the error is proportional to h^5. For the composite Simpson rule (n intervals of length $2h$, $b-a=2nh$) the error therefore is $\frac{b-a}{180}h^4 f^{(4)}(\xi)$ with some ξ in (a,b).

Instead of computing the interpolation polynomial, we can use the Maple function `interp` to interpolate a polynomial through the points $(0,y_0),(h,y_1)$ and $(2h,y_2)$. By integration we obtain the same result as above.

```
> factor(int(interp([0,h,2*h], [y[0], y[1],y[2]], z), z=0..2*h)/(2*h));
```

$$\frac{1}{6}\left(y_2+4\,y_1+y_0\right)$$

Notice that we have normalized the rule by dividing through the interval length.

More generally, we obtain *Newton-Cotes Rules* by interpolating $n+1$ function values with given equidistant nodes and by integrating the degree n interpolation polynomial. The following procedure generates such a $n+1$-point normalized Newton-Cotes Rule.

```
> Cotes := n -> factor(
   int(interp([seq(i*h, i=0..n)], [seq(y[i], i=0..n)], z), z=0..n*h))/(n*h):
```

With this procedure we can e.g. construct the Trapezoidal rule ($n=1$), the Milne Rule ($n=4$) or the Weddle Rule ($n=6$):

```
> Cotes(1);
```

$$\frac{1}{2}\left(y_1+y_0\right)$$

```
> Cotes(4);
```

$$\frac{1}{90}\left(7\,y_4+32\,y_3+12\,y_2+7\,y_0+32\,y_1\right)$$

```
> Cotes(6);
```

$$\frac{1}{840}\left(27\,y_4+216\,y_5+216\,y_1+27\,y_2+272\,y_3+41\,y_0+41\,y_6\right)$$

For $n=8$ we obtain the following equidistant 9-point rule which is used by the Matlab function `quad8`.

```
> Cotes(8);
```

$$-\frac{1}{28350}\left(4540\,y_4-5888\,y_7-989\,y_8-10496\,y_5-5888\,y_1+928\,y_2-10496\,y_3-989\,y_0+928\,y_6\right)$$

In [7] we can find one sided formulas that can also be generated with Maple in the same way. E.g. given four equidistant function values (three intervals), find an approximation for the integral over the 3rd interval:

```
> factor(int(interp([0,h,2*h,3*h], [y[0],y[1],y[2],y[3]], t), t=2*h..3*h));
```

$$-\frac{1}{24}\,h\left(-9\,y_3-19\,y_2+5\,y_1-y_0\right)$$

6 Approximating Derivatives

When replacing derivatives by finite differences one uses relations like e.g.

$$y''(x) \approx \frac{y(x-h) - 2y(x) + y(x+h)}{h^2}. \tag{4}$$

They are obtained by computing *derivatives of the corresponding interpolation polynomial.* Relation (4) is obtained by the Maple statement

```
> diff(interp([x-h,x,x+h],[y(x-h), y(x), y(x+h)],z),z$2);
```

$$\frac{\mathrm{y}(x+h) - 2\,\mathrm{y}(x) + \mathrm{y}(x-h)}{h^2}$$

Again, we can determine the discretization error with the help of a series expansion.

```
> "-D(D(y))(x);
```

$$\frac{\mathrm{y}(x+h) - 2\,\mathrm{y}(x) + \mathrm{y}(x-h)}{h^2} - (D^{(2)})(y)(x)$$

```
> series(",h);
```

$$\frac{1}{12}\,(D^{(4)})(y)(x)\,h^2 + \mathrm{O}(h^4)$$

Similarly, with the statements

```
> diff(interp([x,x+h,x+2*h],[y(x),y(x+h),y(x+2*h)], z), z):
> normal(subs(z=x, "));
```

$$-\frac{1}{2}\,\frac{3\,\mathrm{y}(x) + \mathrm{y}(x+2\,h) - 4\,\mathrm{y}(x+h)}{h}$$

an approximation for $y'(x)$ is obtained. The discretization error of this approximation is also of order h^2.

```
> series(" - D(y)(x), h, 4);
```

$$-\frac{1}{3}\,(D^{(3)})(y)(x)\,h^2 - \frac{1}{4}\,(D^{(4)})(y)(x)\,h^3 + \mathrm{O}(h^4)$$

7 Gauss-Quadrature

The idea of Gauss-Quadrature is to find nodes x_i and weights w_i so that the quadrature rule

$$\int_{-1}^{1} f(x)\,dx \approx \sum_{i=1}^{n} w_i\,f(x_i) \tag{5}$$

is exact for polynomials of degree as high as possible. For $n = 3$ we have to determine the six unknowns w_1, w_2, w_3, x_1, x_2 and x_3. We demand exact values for the integrals of the monomes x^j for $j = 0, \ldots, 5$ and obtain six (nonlinear) equations:

$$w_1 x_1^j + w_2 x_2^j + w_3 x_3^j = \int_{-1}^{1} x^j\,dx \tag{6}$$

```
> eqns := {seq(w1*x1^k+w2*x2^k+w3*x3^k = int(x^k, x=-1..1), k=0..5)};
```

$$eqns := \{w1\, x1 + w2\, x2 + w3\, x3 = 0,\ w1\, x1^2 + w2\, x2^2 + w3\, x3^2 = \frac{2}{3},$$
$$w1\, x1^3 + w2\, x2^3 + w3\, x3^3 = 0,\ w1\, x1^4 + w2\, x2^4 + w3\, x3^4 = \frac{2}{5},$$
$$w1\, x1^5 + w2\, x2^5 + w3\, x3^5 = 0,\ w1 + w2 + w3 = 2\}$$

We can solve this system with Maple:

```
> sols := solve(eqns, indets(eqns, name)):
> convert(sols[1], radical);
```

$$\{x1 = 0,\ w2 = \frac{5}{9},\ w3 = \frac{5}{9},\ w1 = \frac{8}{9},\ x3 = \frac{1}{5}\sqrt{3}\sqrt{5},\ x2 = -\frac{1}{5}\sqrt{3}\sqrt{5}\}$$

However, this brute force approach will not work for all values of n. For larger n the system of nonlinear equations become too complicated for Maple. One has to add some more sophisticated theory to compute the rules.

It is our goal to find nodes and weights to get an exact rule for polynomials of degree up to $2n-1$.

$$\int_{-1}^{1} P_{2n-1}(x)\ dx = \sum_{i=1}^{n} w_i P_{2n-1}(x_i) \tag{7}$$

We can argue as follows: consider the decomposition of P_{2n-1} by dividing by some polynomial $Q_n(x)$ of degree n:

$$P_{2n-1}(x) = H_{n-1}(x)Q_n(x) + R_{n-1}(x).$$

Then

$$\int_{-1}^{1} P_{2n-1}(x)\ dx = \int_{-1}^{1} H_{n-1}(x)Q_n(x)\ dx + \int_{-1}^{1} R_{n-1}(x)\ dx$$

Applying rule (7) on both sides and subtracting yields the following expression for the error:

$$\begin{aligned}\text{error} := &\int_{-1}^{1} H_{n-1}(x)Q_n(x)\ dx - \sum_{i=1}^{n} w_i H_{n-1}(x_i)Q_n(x_i)\\ &+ \int_{-1}^{1} R_{n-1}(x)\ dx - \sum_{i=1}^{n} w_i R_{n-1}(x_i)\end{aligned}$$

Now it is easy to see that we can make the error to zero by the following choices. First take $Q_n(x)$ as the orthogonal polynomial on the interval $[-1,1]$ to the scalar product

$$(f,g) = \int_{-1}^{1} f(x)g(x)\ dx.$$

By this choice and by the definition of orthogonal polynomial the first term in the error vanishes: $\int_{-1}^{1} H_{n-1}(x)Q_n(x)\ dx = 0$. Q_n is a *Legendre* Polynomial available in Maple as `orthopoly[P](n,x)`.

Second, choose as the nodes the (real) zeros of Q_n. Then the second term in the error will also vanish: $\sum_{i=1}^{n} w_i H_{n-1}(x_i)Q_n(x_i) = 0$.

Finally compute the weights according to Newton-Cotes by integrating the interpolation polynomial for R_{n-1}, which is of course again R_{n-1} by the uniqueness of the interpolation polynomial. Thus

$$\int_{-1}^{1} R_{n-1}(x)\ dx = \sum_{i=1}^{n} w_i R_{n-1}(x_i)$$

and the two last error term cancel.

So, we can compute a Gauss quadrature rule e.g. for $n = 12$ with the following Maple statements:

```
> X := sort([fsolve(orthopoly[P](12, x)=0, x)]);
```

$$X := [-.9815606342, -.9041172564, -.7699026742, -.5873179543, -.3678314990, -.1252334085, .1252334085, .3678314990, .5873179543, .7699026742, .9041172564, .9815606342]$$

```
> interp(X, [seq(y[i], i=1..12)], z):
> Q := int(", z=-1..1);
```

$$Q := .04717506586\, y_1 + .1069394295\, y_2 + .1600776434\, y_3 + .2031689029\, y_4 + .2334973032\, y_5 + .2491475198\, y_6 + .2491470907\, y_7 + .2334921379\, y_8 + .2031674530\, y_9 + .1600783833\, y_{10} + .1069391353\, y_{11} + .04717532540\, y_{12}$$

We note that numerical errors occur (the weights should be symmetric) because we are computing the rules here in a well-known unstable way. However, Maple offers us more precision by increasing the value of `Digits`. With two runs of the above statements with different precision we are able to obtain the rules correct to the amount of decimal digits we want.

The Gauss-Lobatto quadrature rule on $[-1, 1]$ using the end points and two intermediate points can be computed with a computer algebra system as follows ([1]). Considering the symmetry of the formula, we make the ansatz

```
> A := f -> int(f(t), t=-1..1) = a*(f(-1) + f(1)) + b*(f(-xi) + f(xi));
```

$$A := f \to \int_{-1}^{1} f(t)\, dt = a\,(f(-1) + f(1)) + b\,(f(-\xi) + f(\xi))$$

and require it to be exact for $f(x) = 1, x^2$ and x^4.

```
> solve({A(x->1), A(x->x^2), A(x->x^4)}, {a,b,xi});
```

$$\{a = \frac{1}{6},\ b = \frac{5}{6},\ \xi = \mathrm{RootOf}(5\,\_Z^2 - 1)\}$$

If we like to compute a Kronrod extension by adding three more points (by symmetry one of them will be 0), we could make the ansatz

```
> A := f -> int(f(t), t=-1..1) = a*(f(-1) + f(1)) + b*(f(-xi) + f(xi))
            + c*(f(-1/sqrt(5)) + f(1/sqrt(5))) + d*f(0);
```

$$A := f \to \int_{-1}^{1} f(t)\, dt = a\,(f(-1) + f(1)) + b\,(f(-\xi) + f(\xi)) + c\,(f(-\frac{1}{\sqrt{5}}) + f(\frac{1}{\sqrt{5}})) + d\, f(0)$$

and require exactness for $f(x) = 1, x^2, x^4, x^6$ and x^8:

```
> seq(A(unapply(x^(2*i),x)), i=0..4);
```

$$2 = 2\,a + 2\,b + 2\,c + d,\ \frac{2}{3} = 2\,a + 2\,b\,\xi^2 + \frac{2}{5}\,c,\ \frac{2}{5} = 2\,a + 2\,b\,\xi^4 + \frac{2}{25}\,c,\ \frac{2}{7} = 2\,a + 2\,b\,\xi^6 + \frac{2}{125}\,c,\ \frac{2}{9} = 2\,a + 2\,b\,\xi^8 + \frac{2}{625}\,c$$

```
> solve({"}, {a,b,c,d,xi});
```

$$\{\xi = \mathrm{RootOf}(3\,\_Z^2 - 2),\ b = \frac{72}{245},\ d = \frac{16}{35},\ c = \frac{125}{294},\ a = \frac{11}{210}\}$$

Thus our rule becomes

$$\int_{-1}^{1} f(t)\,dt = \frac{11}{210}(f(-1)+f(1)) + \frac{72}{245}(f(-\sqrt{\frac{2}{3}}) + f(\sqrt{\frac{2}{3}})) + \frac{125}{294}(f(-\frac{1}{\sqrt{5}}) + f(\frac{1}{\sqrt{5}})) + \frac{16}{35}f(0).$$

For a more elaborated treatment on generating Gauss Quadrature formulas symbolically we refer to [8].

8 Generation of Explicit Runge-Kutta Formulas

In this section we show how a computer algebra system can be used to derive explicit Runge-Kutta formulas. Such formulas are used to solve systems of differential equations of the first order. The solution of the initial value problem

$$y'(x) = f\big(x, y(x)\big), \qquad y(x_k) = y_k \tag{8}$$

can be approximated by a Taylor series around x_k, which is obtained from (8) by repeated differentiation and replacing $y'(x)$ by $f\big(x, y(x)\big)$ every time it appears.

$$\begin{aligned}
y(x_k + h) &= \sum_{i=0}^{\infty} y^{(i)}(x_k)\frac{h^i}{i!} \\
&= y(x_k) + hf\big(x_k, y(x_k)\big) + \frac{h^2}{2}\left(\frac{\partial}{\partial x} f\big(x, y(x)\big)\bigg|_{x=x_k}\right) + \ldots \\
&= y(x_k) + h\underbrace{\left(f\big(x_k, y(x_k)\big) + \frac{h}{2}\Big(f_x\big(x_k, y(x_k)\big) + f\big(x_k, y(x_k)\big)f_y\big(x_k, y(x_k)\big)\Big) + \ldots\right)}_{\Phi(x_k,\, y(x_k),\, h)}
\end{aligned} \tag{9}$$

The idea of the Runge-Kutta methods is to approximate the Taylor series (9) up to order m by using only *values* of $f\big(x, y(x)\big)$ and no derivatives of it.

The general form of a *s-stage explicit Runge-Kutta method* is

$$\begin{aligned}
k_1 &= f(x, y), \\
k_2 &= f(x + c_2\, h, y + h\, a_{2,1}\, k_1), \\
&\vdots \\
k_s &= f(x + c_s\, h, y + h \sum_{j=1}^{s-1} a_{s,j}\, k_j), \\
\Phi(x, y, h) &= \sum_{i=1}^{s} b_i k_i, \\
y_{k+1} &= y_k + h\,\Phi(x_k, y_k, h),
\end{aligned} \tag{10}$$

where s is the number of "stages" and $a_{i,j}$, b_i and c_i are real coefficients

To derive the coefficients of such a method, the series expansion of (9) and (10) are equated. This leads to a set of nonlinear equations for the parameters $a_{i,j}$, b_i and c_i which has to be solved.

For this derivation we have to compute the Taylor series expansions of (9) and (10). Maple knows how to expand a function with two parameters that both depend on x, but we have to inform Maple, that $y'(x)$ has to be replaced by $f\big(x, y(x)\big)$ whenever it appears. We do this by overwriting the derivative of the operator `y`.

```
> D(y) := x -> f(x,y(x)):
> taylor(y(x+h), h=0, 3);
```

$$\mathrm{y}(x)+\mathrm{f}(x,\,\mathrm{y}(x))\,h+(\frac{1}{2}\,D_1(f)(x,\,\mathrm{y}(x))+\frac{1}{2}\,D_2(f)(x,\,\mathrm{y}(x))\,\mathrm{f}(x,\,\mathrm{y}(x)))\,h^2+\mathrm{O}(h^3)$$

In this result, $D_1(f)(x,y(x))$ stands for the derivative of f with respect to the first argument, i.e. $f_x(x,y(x))$. In order to make the result more readable, we define some alias substitutions.

```
> alias(F = f(x,y(x)), Fx = D[1](f)(x,y(x)), Fy = D[2](f)(x,y(x)),
      Fxx = D[1,1](f)(x,y(x)), Fxy = D[1,2](f)(x,y(x)),
      Fyy = D[2,2](f)(x,y(x))):
```

We are now ready to derive the parameters of a Runge-Kutta formula for $s=3$ which is of order $m=3$.

```
> m := 3:
> taylor(y(x+h),h=0,m+1);
```

$$\mathrm{y}(x)+F\,h+(\frac{1}{2}\,Fx+\frac{1}{2}\,Fy\,F)\,h^2+(\frac{1}{6}\,Fxx+\frac{1}{3}\,F\,Fxy+\frac{1}{6}\,F^2\,Fyy+\frac{1}{6}\,Fy\,Fx+\frac{1}{6}\,Fy^2\,F)\,h^3+\mathrm{O}(h^4)$$

```
> TaylorPhi := normal((convert(",polynom) - y(x))/h);
```

$$TaylorPhi := F+\frac{1}{2}\,h\,Fx+\frac{1}{2}\,h\,Fy\,F+\frac{1}{6}\,h^2\,Fxx+\frac{1}{3}\,h^2\,F\,Fxy+\frac{1}{6}\,h^2\,F^2\,Fyy+\frac{1}{6}\,h^2\,Fy\,Fx+\frac{1}{6}\,h^2\,Fy^2\,F$$

The variable `TaylorPhi` corresponds to Φ in equation (9).

For the Runge-Kutta scheme we get the following Taylor series. Note that we keep the parameters $a_{i,j}$, b_i and c_i in symbolic form. `RungeKuttaPhi` corresponds to Φ in equation (10).

```
> k1 := taylor(f(x,      y(x)),                    h=0, m):
> k2 := taylor(f(x+c2*h, y(x)+h*(a21*k1)),         h=0, m):
> k3 := taylor(f(x+c3*h, y(x)+h*(a31*k1+a32*k2)),h=0, m):
> RungeKuttaPhi := convert(series(b1*k1+b2*k2+b3*k3,h,m),polynom);
```

$$\begin{aligned}RungeKuttaPhi &:= b1\,F+b2\,F+b3\,F\\&+(b2\,(Fx\,c2+Fy\,a21\,F)+b3\,(Fx\,c3+Fy\,a31\,F+Fy\,a32\,F))\,h+(\\&b2\,(\frac{1}{2}\,Fxx\,c2^2+c2\,Fxy\,a21\,F+\frac{1}{2}\,a21^2\,F^2\,Fyy)+b3(\frac{1}{2}\,Fxx\,c3^2+c3\,Fxy\,a31\,F\\&+c3\,Fxy\,a32\,F+\frac{1}{2}\,a31^2\,F^2\,Fyy+a31\,F^2\,Fyy\,a32+\frac{1}{2}\,a32^2\,F^2\,Fyy+Fy\,a32\,Fx\,c2\\&+Fy^2\,a32\,a21\,F))h^2\end{aligned}$$

The difference `d` of the two polynomials `TaylorPhi` and `RungeKuttaPhi` should be zero. We consider `d` as a polynomial in the unknowns `h`, `F`, `Fx`, `Fy`, `Fxx`, etc. and set the coefficients of that polynomial to zero. This gives us a nonlinear system of equations which has to be solved.

```
> d := expand(TaylorPhi-RungeKuttaPhi):
> eqns := {coeffs(d, [h,F,Fx,Fy,Fxx,Fxy,Fyy])};
```

$$eqns := \{-\frac{1}{2}\,b3\;c3^2 - \frac{1}{2}\,b2\;c2^2 + \frac{1}{6},\; 1 - b2 - b3 - b1,\; -b2\;a21 - b3\;a32 + \frac{1}{2} - b3\;a31,$$
$$\frac{1}{3} - b3\;c3\;a32 - b3\;c3\;a31 - b2\;c2\;a21,$$
$$-b3\;a31\;a32 + \frac{1}{6} - \frac{1}{2}\,b2\;a21^2 - \frac{1}{2}\,b3\;a32^2 - \frac{1}{2}\,b3\;a31^2,\; \frac{1}{2} - b3\;c3 - b2\;c2,\; -b3\;a32\;c2 + \frac{1}{6},$$
$$-b3\;a32\;a21 + \frac{1}{6}\}$$

```
> solve(eqns, indets(eqns));
```

$$\{b3 = b3,\; b1 = \frac{1}{4}\,\frac{12\,b3\;c3^2 - 1 - 12\,b3\;c3 + 4\,b3}{3\,b3\;c3^2 - 1},\; b2 = -\frac{3}{4}\,\frac{1 - 4\,b3\;c3 + 4\,b3^2\;c3^2}{3\,b3\;c3^2 - 1},$$
$$c2 = \frac{2}{3}\,\frac{3\,b3\;c3^2 - 1}{-1 + 2\,b3\;c3},\; a32 = \frac{1}{4}\,\frac{-1 + 2\,b3\;c3}{b3\,(3\,b3\;c3^2 - 1)},\; a31 = \frac{1}{4}\,\frac{12\,b3^2\;c3^3 - 6\,b3\;c3 + 1}{b3\,(3\,b3\;c3^2 - 1)},$$
$$a21 = \frac{2}{3}\,\frac{3\,b3\;c3^2 - 1}{-1 + 2\,b3\;c3},\; c3 = c3\},$$
$$\{a32 = a32,\; b3 = \frac{3}{4},\; b1 = \frac{1}{4},\; c3 = \frac{2}{3},\; a21 = \frac{2}{9}\,\frac{1}{a32},\; c2 = \frac{2}{9}\,\frac{1}{a32},\; a31 = \frac{2}{3} - a32,\; b2 = 0\}$$

For $s = 3$, we found two (parameterized) solutions which can be represented by the following coefficient schemes. Note that in the first solution, the unknowns c_3 and b_3 are free parameters, i.e. they can take on any value. This is indicated by the entries `c3 = c3` and `b3 = b3` in the solution set. For the second solution, $a_{3,2}$ is a free parameter (see also Figure 1). From the second solution we get the method of Heun of third order if we set $a_{3,2} = 2/3$.

$$\begin{array}{c|ccc}
0 & & & \\
\frac{2}{3}\,\frac{3\,b_3\,c_3{}^2-1}{2\,b_3\,c_3-1} & \frac{2}{3}\,\frac{3\,b_3\,c_3{}^2-1}{2\,b_3\,c_3-1} & & \\
c_3 & \frac{1-6\,b_3\,c_3+12\,b_3{}^2c_3{}^3}{4\,b_3\,(3\,b_3\,c_3{}^2-1)} & \frac{2\,b_3\,c_3-1}{4\,b_3\,(3\,b_3\,c_3{}^2-1)} & \\
\hline
 & \frac{12\,b_3\,c_3{}^2-1+4\,b_3-12\,b_3\,c_3}{12\,b_3\,c_3{}^2-4} & \frac{3}{4}\,\frac{(2b_3\,c_3-1)^2}{1-3b_3\,c_3{}^2} & b_3
\end{array}
\qquad
\begin{array}{c|ccc}
0 & & & \\
\frac{2}{9a_{3,2}} & \frac{2}{9a_{3,2}} & & \\
\frac{2}{3} & \frac{2}{3} - a_{3,2} & a_{3,2} & \\
\hline
 & \frac{1}{4} & 0 & \frac{3}{4}
\end{array}$$

Figure 1: Three Stage Runge-Kutta Methods of Order 3.

For further information on this topic we refer to [4].

9 Methods of Ritz and Galerkin

This section shows an example where the computer algebra systems takes over the hard hand work and the students can understand the general steps of the method. Consider the boundary value problem

$$\mathcal{D}(u(x)) = -u''(x) + 2u(x) = f(x), \quad u(0) = 0, u'(\pi) = 0. \tag{11}$$

For $f(x) = \frac{17}{4}\sin(\frac{3}{2}\,x)$ the solution is

```
> DG := u -> -D(D(u)) + 2*u - f:
> f  := x -> 17/4*sin(3/2*x):
> dsolve({DG(u)(x)=0, u(0)=0, D(u)(Pi)=0}, u(x));
```

$$\mathrm{u}(x) = \sin(\frac{3}{2}\,x)$$

```
> assign("):
```

Equation (11) is obtained if we want to minimize the functional

$$L = \int_0^\pi u'(x)^2 + 2u(x)^2 - 2f(x)u(x)\,dx. \tag{12}$$

In the method of *Ritz* the solution u is approximated by an ansatz (a linear combination of known functions $\phi_j(x)$):

$$y(x) = \sum_{j=1}^{n} c_j \phi_j(x). \tag{13}$$

Introducing $y(x)$ in (12) we obtain a quadratic form in the unknown coefficients c_j. Minimizing this quadratic form gives us values for c_j and an approximation $y(x)$.

The principle is easy to describe but to compute a concrete example is rather tedious, even if we choose only $n = 2$, $\phi_1(x) = x - \frac{x^2}{2\pi}$ and $\phi_2(x) = x\exp(-\frac{x}{\pi})$. Both ansatz-functions do satisfy the boundary conditions.

With Maple we can compute and plot the approximation with the following statements:

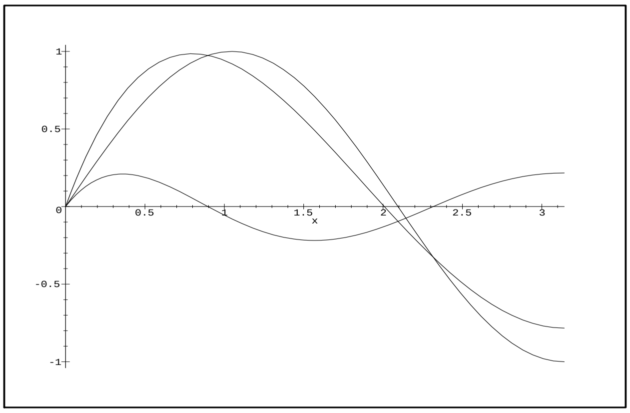
```
> y := x -> c1*(x-x^2/(2*Pi)) + c2*x*exp(-x/Pi):
> L := int(diff(y(x),x)^2+2*y(x)^2-2*f(x)*y(x),x=0..Pi):
> evalf(L);
```

$$9.315538005\, c1^2 + 5.69163633\, c2^2 + 14.5140300\, c1\, c2 - 1.582748541\, c2 \\ - .8016693429\, c1$$

```
> v := linalg[grad](L, [c1,c2]):
> solve({v[1],v[2]}, {c1,c2}):
> evalf("); assign(");
```

$$\{c2 = 12.52353174,\ c1 = -9.713086137\}$$

```
> plot({y(x), u(x), y(x)-u(x)}, x=0..Pi, color=black);
```



Not every differential equation minimizes a functional. In the method of *Galerkin* one tries to solve (11) also by the ansatz (13). The goal is to choose the coefficients c_j such that the residual

$r(x) = \mathcal{D}y(x) - f(x)$ becomes small in some sense. For this we have to choose another set of functions $\{\psi_j(x)\}$. The coefficients c_j are now computed in such a way that the *residual is orthogonal to the space spanned by the functions* $\psi_j(x)$:

$$\int_0^\pi r(x)\psi_j(x)\,dx = 0, \quad j = 1, \ldots, n. \tag{14}$$

We obtain this way again a system of linear equations for the coefficients c_j.

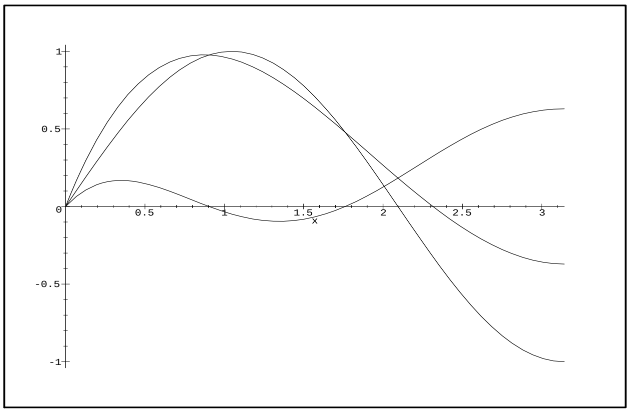
```
> c1 := 'c1': c2 := 'c2':
> psi1 := x -> sin(x):
> psi2 := x -> cos(x):
> eq := {int(DG(y)(t)*psi1(t), t=0..Pi), int(DG(y)(t)*psi2(t), t=0..Pi)}:
> evalf(eq);
```

$$\{-1.038400155\, c2 - 5.100000003 - 2.000000001\, c1,$$
$$5.051451973\, c1 - 3.400000002 + 4.146353669\, c2\}$$

```
> solve(eq, {c1, c2}):
> evalf("); assign(");
```

$$\{c1 = -8.098032911,\, c2 = 10.68573206\}$$

```
> plot({y(x), u(x), y(x)-u(x)}, x=0..Pi, color=black);
```



10 Conclusions

In this paper we have given some examples on how we use computer algebra systems in our scientific computing courses. Many numerical methods and classical proofs can be developed with only a few statements in a computer algebra system.

However, it might be difficult sometimes to find the right ones, this method therefore is in general still restricted to the reproduction of classical results. The use of a computer algebra system also requires much experience, as it is not always easy to find an elegant way to the result one is expecting.

We also made the experience that it may be particularly complicated to convince a computer algebra system to perform a specific task. As an example take the convergence analysis of section 3. We came across the expression $(a^p)^p/a^p/a$ and were interested in the exponent if this expression is written as a^b. How is b obtained? Right, taking the logarithm to the base a. The result however does not simplify, even not after using the *simplify* command.

```
> b := log[a]((a^p)^p/a^p/a);
```

$$b := \frac{\ln\left(\frac{(a^p)^p}{a^p\, a}\right)}{\ln(a)}$$

```
> simplify(b);
```

$$\frac{\ln\left(\frac{(a^{(-p+1)})^{(-p)}}{a}\right)}{\ln(a)}$$

What is the problem? Maple does not know as much as we do. Maple cannot simplify this expression as it assumes a and p to be complex numbers. Obviously, a and p are real and positive in our context, but Maple has to be informed about this fact using the *assume* facility. This can be done directly in the *simplify* command for all the indeterminants which appear in the expression to be simplified or for each unknown with the *assume* command. The symbol ˜ signals that assumptions have been made on a variable.

```
> simplify(b, assume=positive);
```

$$-1 + p^2 - p$$

```
> assume(a > 0);   # a is assumed to be real and positive
> assume(p, real); # b is assumed to be real
> simplify(b);
```

$$p\tilde{}^{2} - p\tilde{} - 1$$

Look also at the discussion of the discretization error in chapter 6. We have computed a series expansion of an expression comparable to the following one:

```
> f(x+h) - f(x) - h*diff(f(x), x);
```

$$\mathrm{f}(x+h) - \mathrm{f}(x) - h\,(\frac{\partial}{\partial x}\,\mathrm{f}(x))$$

Which leading term do you expect if this expression is expanded into a series? Maple gives you the following answer:

```
> series(", h, 2);
```

$$(\mathrm{D}(f)(x) - (\frac{\partial}{\partial x}\,\mathrm{f}(x)))\,h + \mathrm{O}(h^2)$$

This result may be surprising for a student. The leading coefficient is indeed zero, but Maple does not recognize this zero automatically. In general, it is a particularly difficult problem to recognize zeros, but in this example the above result can be simplified using a special option to the command `simplify`.

```
> simplify(", diff);
```

$$\mathrm{O}(h^2)$$

Computer algebra systems still have to progress. They are not yet a replacement for paper and pencil.

We also do not want to conceal that computer algebra systems still have bugs and produce erroneous results or results which are only valid under some assumptions. It is also very important to demonstrate this fact to students. Students should learn, that results cannot blindly be trusted. Whenever a numerical method is derived, the result has to be compared with ones expectations.

Nevertheless, a computer algebra system is a very powerful tool to be used in teaching numerical methods. Many examples in this article have proven it. Further examples on the use of computer algebra systems can be found e.g. in [3, 2, 5].

References

[1] Walter Gander and Walter Gautschi, *Adaptive Quadrature - Revisited*, Research Report Computer Science Department ETH, Zürich, 1998.

[2] Walter Gander and Dominik Gruntz, *The Billiard Problem*, Int. J. Math. Educ. Sci. Technol, 23(6), p. 825-830, 1992.

[3] Walter Gander and Jiri Hřebíček, ed. *Solving Problems in Scientific Computing Using Maple and Matlab*, Springer, Berlin etc., 3rd edition, 1997.

[4] Dominik Gruntz, *Symbolic Computation of Explicit Runge-Kutta Formulas*, chapter 19 in [3], p. 281–296, 1997.

[5] Dominik Gruntz, *Automatic Differentiation and Bisection,* MapleTech, The Maple Technical Newsletter, **4**(1), 14–19, Birkhäuser, 1997.

[6] Michael T. Heath, *Scientific Computing*, McGraw-Hill, 1997.

[7] Eduard Stiefel, *Einführung in die numerische Mathematik*, Teubner, 1976.

[8] Urs von Matt, *Gauss Quadrature*, chapter 18 in [3], p. 251–279, 1997.