11. Numerische Quadratur

Aufgabe: Numerische Berechnung eines bestimmten Integrals

$$I \;:=\; I[f] \;:=\; \int_a^b f(x)\,\mathrm{d}x\,.$$

Dabei wird vorausgesetzt, dass f auf $[a,b]$ hinreichend oft stetig differenzierbar ist. Auf diese Voraussetzung sollte man bei der Anwendung eines Verfahrens zur numerischen Quadratur stets achten!

Das bestimmte Integral ist ein *linearer* und *positiver* Operator. Daher verlangt man, dass auch Quadraturformeln diese Eigenschaften besitzen.

Quadraturformel: $a \leq x_0 < \ldots < x_n \leq b$

$$I_n[f] = \sum_{i=0}^{n} g_i \, f(x_i) \qquad \textbf{Quadraturformel}$$

$$x_i \in [a,b], \quad i = 0,1,\ldots,n \qquad \textbf{Knoten}$$

$$g_i, \quad i = 0,1,\ldots,n \qquad \textbf{Gewichte}$$

Quadraturformeln dieser Art sind lineare Operatoren! Sie sind auch positiv, falls alle Gewichte g_i positiv sind.

Die Differenz

$$R_n[f] := I_n[f] - I[f]$$

heißt der **Quadraturfehler** der Formel $I_n[f]$.

11.1 Newton–Cotes Formeln

Man ersetzt den Integranden $f(x)$, $a \le x \le b$, durch ein interpolierendes Polynom, dessen Integral dann leicht ausgewertet werden kann.

Knoten äquidistant:

$$x_i \;=\; a + i\,h, \qquad h \;=\; \frac{1}{n}\,(b-a), \qquad i = 0, 1, \ldots, n\,.$$

Lagrange–Darstellung des Interpolationspolynoms:

$$\begin{aligned} I_n[f] \;&=\; \int_a^b \sum_{i=0}^{n} f(x_i) \prod_{k=0,\,k\neq i}^{n} \left(\frac{x - x_k}{x_i - x_k}\right) \mathrm{d}x \\ &=\; \sum_{i=0}^{n} f(x_i) \int_a^b \prod_{k\neq i} \left(\frac{x - x_k}{x_i - x_k}\right) \mathrm{d}x\,. \end{aligned}$$

Substitution: $x = a + th, \quad 0 \le t \le n, \qquad \Rightarrow$

$$I_n[f] \;=\; (b-a) \sum_{i=0}^{n} \alpha_{in}\, f(x_i)\,,$$

$$\alpha_{in} \;=\; \frac{1}{n} \int_0^n \prod_{k=0,\, k\neq i}^{n} \left(\frac{t-k}{i-k}\right) \,\mathrm{d}t\,. \qquad (11.1.1)$$

n	α_{in}					
1	$\frac{1}{2}$	$\frac{1}{2}$				(Trapezregel)
2	$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$			(Simpson–Regel)
3	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{1}{8}$		($\frac{3}{8}$– Regel)
4	$\frac{7}{90}$	$\frac{32}{90}$	$\frac{12}{90}$	$\frac{32}{90}$	$\frac{7}{90}$	(Milne–Regel)

Satz (11.1.2)

Die Newton–Cotes–Formel $I_n[f]$ integriert Polynome vom Grad $\leq n$ exakt.

Quadraturfehler (11.1.3)

n	$R_n[f]$
1	$h^3 \frac{1}{12} f^{(2)}(\xi_1)$
2	$h^5 \frac{1}{90} f^{(4)}(\xi_2)$
3	$h^5 \frac{3}{80} f^{(4)}(\xi_3)$
4	$h^7 \frac{8}{945} f^{(6)}(\xi_4)$

Beispiel (11.1.4)

Integralsinus: $\mathrm{Si}(x) = \int_0^x \frac{\sin t}{t}\,\mathrm{d}t$ an der Stelle $x = 1$.

Wir verwenden die 3/8–Regel.

$$I_3[f] = (1-0)\left\{\frac{1}{8}f(0) + \frac{3}{8}f(\frac{1}{3}) + \frac{3}{8}f(\frac{2}{3}) + \frac{1}{8}f(1)\right\} = 0.946110921\,.$$

Fehlerabschätzung:

Wie genau ist diese Näherung? Wir verwenden die Fehlerdarstellung aus (11.1.3).

$$R_3[f] \;=\; \left(\frac{1}{3}\right)^5 \cdot \frac{3}{80} \cdot f^{(4)}(\xi) \;=\; \frac{1}{80 \cdot 81}\, f^{(4)}(\xi)\,.$$

Mit der Taylor–Entwicklung von $(\sin t)/t$ findet man

$$f^{(4)}(t) \;=\; \frac{1}{5} \;-\; \frac{1}{2!\,7}\,t^2 \;+\; \frac{1}{4!\,9}\,t^4 \;-\,+\;\ldots$$

und somit nach dem Leibniz–Kriterium: $\left| f^{(4)}(\xi) \right| \;\leq\; \frac{1}{5}\,.$

Insgesamt folgt: $|R_3(f)| \;\leq\; \frac{1}{5 \cdot 80 \cdot 81} \;\leq\; 3.1 \cdot 10^{-5}\,.$

Wir haben damit gezeigt, dass der obige Näherungswert auf vier Dezimalstellen (gerundet) korrekt ist!

Exakt: $\mathrm{Si}(1) = 0.946083070367.$

Intervallweise Anwendung

Um höhere Genauigkeiten zu erreichen, unterteilt man das Integrationsintervall $[a, b]$ und wendet die Quadraturformeln auf die einzelnen Teilintervalle an:

$$x_i \;=\; a + i\,h, \quad i = 0, 1, \ldots, N; \qquad h \;:=\; \frac{b-a}{N}\,.$$

Zusammengesetzte Trapezregel (11.1.5)

Die zusammengesetzte Trapezregel (Trapezsumme), lautet:

$$\begin{aligned} T(h) \;&=\; \sum_{i=0}^{N-1} \frac{h}{2}\,(f(x_i) + f(x_{i+1})) \\ &=\; h\,\left\{ \frac{f(a)}{2} + f(a+h) + \ldots + f(b-h) + \frac{f(b)}{2} \right\}. \end{aligned}$$

Quadraturfehler:

$$\left| T(h) - \int_a^b f(x)\,\mathrm{d}x \right| \leq \frac{b-a}{12}\, h^2 \,\| f^{(2)} \|_\infty . \qquad (11.1.6)$$

Zusammengesetzte Simpson–Regel (11.1.7)

Es sei N gerade. Wir wenden die Simpson–Regel auf die Teilintervalle $[x_{2i}, x_{2i+2}]$ mit Knoten x_{2i}, x_{2i+1}, x_{2i+2} an, $i = 0, 1, \ldots, N/2-1$. Man erhält:

$$\begin{aligned} S(h) &= \frac{h}{3} \sum_{i=0}^{N/2-1} \Big(f(x_{2i}) + 4\, f(x_{2i+1}) + f(x_{2i+2}) \Big) \\ &= \frac{h}{3} \{\, f(a) + 4\, f(a+h) + 2\, f(a+2h) + \ \ldots \\ &\qquad \ldots + 2\, f(b-2h) + 4\, f(b-h) + f(b) \,\} \ . \end{aligned}$$

Quadraturfehler:

$$\left| S(h) - \int_a^b f(x)\,\mathrm{d}x \right| \;\leq\; \frac{b-a}{180}\; h^4 \;\| f^{(4)} \|_\infty \,. \qquad (11.1.8)$$

Algorithmische Durchführung (für die Trapezsumme)

Wir berechnen die Trapzsummen $T(h_i)$ für die Schrittweitenfolge

$$h_i = h_{i-1}/2,\;\; i = 1, 2, \ldots, \qquad h_0 := b - a.$$

und verwenden

$$T(h_i) = \frac{T(h_{i-1})}{2} + h_i \left\{ f(a + h_i) + f(a + 3h_i) + \ldots + f(a + (N-1)h_i) \right\}$$

Das Verfahren wird abgebrochen, falls gilt:

$$| T(h_i) - T(h_{i-1}) | \;\leq\; \mathrm{TOL} \cdot | T(h_i) | \,. \qquad (11.1.9)$$

Dabei bezeichnet TOL die geforderte relative Genauigkeit.

Algorithmus (11.1.10)

$n := 1; \quad h := b-a; \quad T_0 := \frac{h}{2}(f(a)+f(b));$

für $i = 1, 2, \ldots$

$h := h/2;$

$S := \sum_{j=1}^{n} f(a + (2j-1)\,h);$

$n := 2\,n;$

$T_i := \frac{1}{2} T_{i-1} + h \cdot S;$

Abbruch, falls $|T_i - T_{i-1}| \leq |T_i| \cdot \mathrm{TOL};$

end i

Beispiel (11.1.11) $\mathrm{Si}(1) = \int_0^1 \frac{\sin t}{t}\,\mathrm{d}t$

Trapezsummen

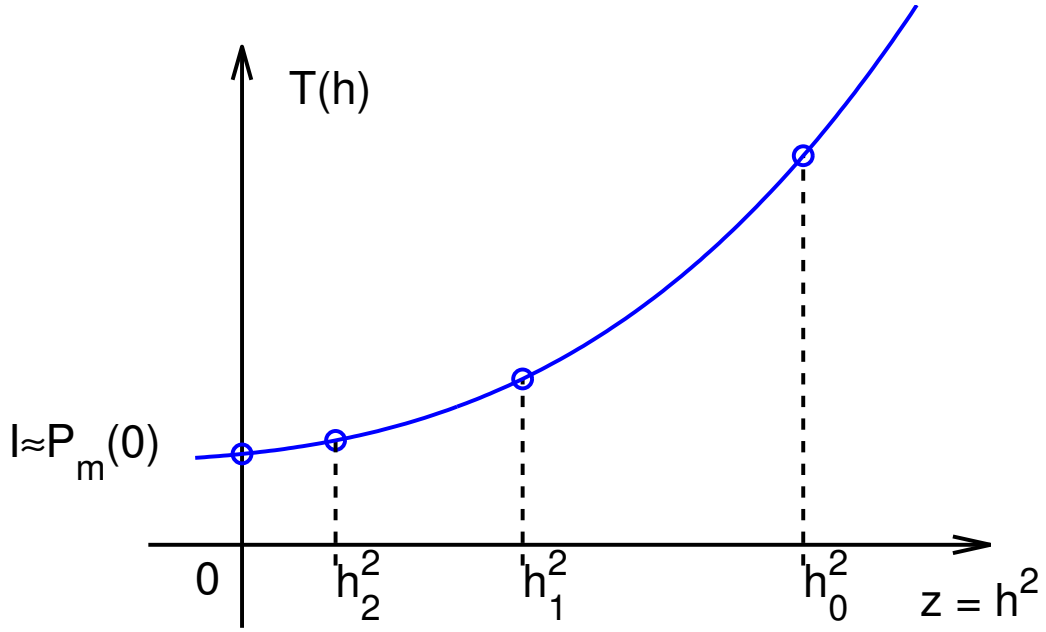
i	T(i)	nfc
0	.92073 54924	2
1	.93979 32848	3
2	.94451 35217	5
3	.94569 08636	9
4	.94598 50299	17
5	.94605 85610	33
6	.94607 96431	65
7	.94608 15385	129
8	.94608 26874	257
9	.94608 29746	513
10	.94608 30464	1025
11	.94608 30644	2049
12	.94608 30689	4097
13	.94608 30700	8193
14	.94608 30703	16385
15	.94608 30703	32769

Simpson-Summen

i	T(i)	nfc
0	.94614 58823	3
1	.94608 69340	5
2	.94608 33109	9
3	.94608 30854	17
4	.94608 30713	33
5	.94608 30704	65
6	.94608 30704	129

11.2 Extrapolation

Extrapolation ist ein **konvergenzbeschleunigendes** Verfahren. Es lässt sich immer dann anwenden, wenn sich die Näherungen $T(h)$ einer gesuchten Größe I (in unserem Fall : $T(h)$: Trapezsumme und I: bestimmtes Integral) verhalten „wie ein Polynom in h^2 “ mit $T(0) = I$.



Satz (11.2.1) (Euler–Maclaurinsche Summenformel)

Ist $f \in \mathrm{C}^{2m+2}[a,b]$, so gibt es Konstante $\tau_0, \ldots, \tau_m$ mit

$$T(h) \;=\; \tau_0 + \tau_1\, h^2 + \ldots + \tau_m\, h^{2m} + \alpha_{m+1}(h)\, h^{2m+2}\,.$$

Dabei ist $|\alpha_{m+1}(h)|$ beschränkt für alle Schrittweiten der Form $h = (b-a)/n,\ n \in \mathbb{N}$.

Die τ_k sind gegeben durch $\tau_0 = \int_a^b f(x)\,\mathrm{d}x$ und

$$\tau_k \;=\; B_{2k} \cdot \left(f^{(2k-1)}(b) - f^{(2k-1)}(a) \right)\,, \quad k = 1, 2, \ldots$$

wobei die B_{2k} die Bernoulli–Zahlen bezeichnen, vgl. (6.2.12).

Grundidee: Richardson (1927), Romberg (1955)

Man approximiert die Funktion $T(h)$ durch ein Interpolationspolynom $P_m(z)$ in der Variablen $z := h^2$ und wertet dieses in $z = h = 0$ aus. Im Allgemeinen ist dann $P_m(0)$ eine erheblich bessere Näherung für I als die zur Interpolation verwendeten Daten $T(h)$.

Fehlerabschätzung (11.2.2)

Unter geeigneten Voraussetzungen an die Schrittweitenfolge h_i gilt

$$P_m(0) \;=\; \int_a^b f(x)\,\mathrm{d}x \;+\; h_0^2\,h_1^2\,\ldots\,h_m^2 \cdot \sigma_m(h_0)\,,$$

wobei $\sigma_m(h)$ für $h \to 0$ beschränkt ist.

Algorithmus (11.2.3)

Zu einer Schrittweitenfolge $h_0 > h_1 > \ldots > h_m$ berechnet man das Interpolationpolynom $P_m(z)$ zu den Stützstellen $(z_i := h_i^2, T(h_i))$ und wertet dieses in $z = 0$ aus.

Mit den Algorithmus von Aitken, Neville, vgl. (7.2.12), folgt:

$$\begin{aligned} P_{k\,0} &= T(h_k), \qquad k = 0, 1, \ldots, m\,, \\ P_{k\,j} &= P_{k,j-1} + \frac{1}{(h_{k-j}/h_k)^2 - 1} \cdot (P_{k,j-1} - P_{k-1,j-1})\,, \end{aligned}$$

für $j = 1, \ldots, k$ mit $P_m(0) = P_{m,m}$.

Speziell für die Halbierungsfolge $h_0 := b - a$ und $h_k = h_{k-1}/2$, $k = 1, 2, \ldots, m$, lässt sich der Algorithmus noch weiter vereinfachen, vgl. Lehrbuch (15.2.4).

Beispiel (11.2.4) $\mathrm{Si}(1) = \int_0^1 \frac{\sin t}{t}\,\mathrm{d}t$

Trapezsummen-Extrapolation:

i	T(i)	P(i)	nfc
0	.92073 54924	.92073 54924	2
1	.93979 32848	.94614 58823	3
2	.94451 35217	.94608 30041	5
3	.94569 08636	.94608 30704	9
4	.94598 50299	.94608 30704	17

Man erkennt, dass Extrapolation eine Genauigkeit von zehn Dezimalstellen bereits mit neun Funktionsauswertungen liefert und damit das bisher beste Verfahren, die Simpson-Summe, deutlich übertrifft.

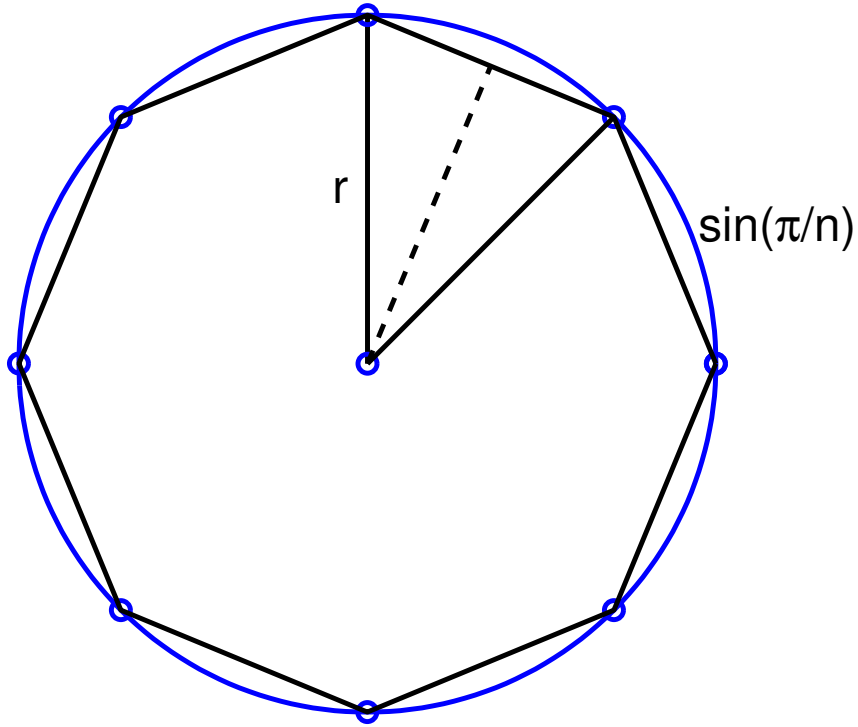
Beispiel (11.2.5) (Berechnung von π)

Sei K ein Kreis mit Radius $r = 1/2$ und bezeichne U_n den Umfang des in K einbeschriebenen regelmäßigen n–Ecks. Dann gilt

$$U_n \;=\; n \cdot \sin\left(\frac{\pi}{n}\right) .$$

Mit Hilfe des Additionstheorems für sin kann man hiermit eine Rekursion zur Berechnung von U_{2n} aus U_n aufstellen:

$$\begin{aligned} U_4 &:= 2\sqrt{2}\,; \qquad C_4 := 1/\sqrt{2} \\ C_{2n} &:= \sqrt{(1+C_n)/2} \\ U_{2n} &:= U_n/C_{2n}, \qquad n = 4, 8, 16, \ldots \end{aligned}$$



$$r = 0.5$$

$$U_n = n \cdot \sin\left(\frac{\pi}{n}\right)$$

U_n besitzt eine asymptotische Entwicklung bezüglich der „Schrittweite“ $h_n := 1/n$. Die Taylor–Entwicklung von $\sin(\pi/n)$ liefert nämlich

$$U_n \;=\; \pi \;-\; \frac{\pi^3}{3!}\left(\frac{1}{n}\right)^2 \;+\; \frac{\pi^5}{5!}\left(\frac{1}{n}\right)^4 \;-\; +\; \ldots$$

Man kann daher das Extrapolationsverfahren auf U_n, $n = 4, 8, 16, \ldots$, anwenden und man erhält mit Algorithmus (15.2.4) die folgende Tabelle. $P_m(0)$ bezeichnet hierbei wieder den extrapolierten Wert.

n	U_n	$P_m(0)$
4	2.8284 27125	2.8284 27125
8	3.0614 67459	3.1392 47570
16	3.1214 45152	3.1415 90393
32	3.1365 48491	3.1415 92653
64	3.1403 31157	3.1415 92654

Bemerkungen (11.2.6)

a) Außer den obigen beiden Verfahrensklassen sind auch Quadraturformeln gebräuchlich, die neben den Gewichten $g_{i,n}$ auch die Knoten x_i optimieren, so dass Polynome möglichst hohen Grades exakt integriert werden. Dieser Ansatz führt auf die so genannte **Gauß– Quadratur**.

Mit dieser Methode lassen sich auch gewisse Singularitäten im Integranden behandeln – auch ist die numerische Berechnung uneigentlicher Integrale hiermit mitunter möglich.

b) Für praktische Anwendungen wird schließlich mit einer automatisch angepassten, im Allg. nichtäquidistanten Partitionierung des Integrationsintervalls gearbeitet (so genannte **adaptive Verfahren**).

11.3 Numerische Berechnung von Fourier–Koeffizienten

Zu berechnen seien die Fourier–Koeffizienten einer glatten, 2π–periodischen Funktion $f : \mathbb{R} \to \mathbb{R}$

$$a_k = \frac{1}{\pi} \int_0^{2\pi} f(t)\, \cos(k\,t)\, \mathrm{d}t, \quad b_k = \frac{1}{\pi} \int_0^{2\pi} f(t)\, \sin(k\,t)\, \mathrm{d}t \;\; (k \geq 0)\,.$$

Anwendung der Trapezsumme zu einem äquidistanten Gitter ($h := 2\pi/n,\;\; t_j := j\,h,\;\; f_j := f(t_j)$) ergibt

$$\begin{aligned} a_k &\approx \frac{1}{\pi} \cdot \frac{2\pi}{n} \cdot \left\{ \frac{f_0}{2} + \sum_{j=1}^{n-1} f_j \cos(k\,t_j) + \frac{f_n}{2} \right\} \\ &= \frac{2}{n} \sum_{j=0}^{n-1} f_j \cos(k\,j\,h) \;\; =: \;\; A_k \end{aligned} \tag{11.3.1}$$

und analog:

$$b_k \;\approx\; \frac{2}{n}\sum_{j=0}^{n-1} f_j \sin(k\,j\,h) \;=:\; B_k\,. \qquad (11.3.2)$$

Damit besteht die Aufgabe darin, Summen der folgenden Form zu berechnen:

$$\sigma(t) \;:=\; \sum_{k=0}^{n-1} f_k \cos(k\,t), \quad \mu(t) \;:=\; \sum_{k=0}^{n-1} f_k \sin(k\,t)\,. \qquad (11.3.3)$$

Da n groß sein kann und zudem die σ und μ i. Allg. für viele t–Werte berechnet werden müssen, kommt eine naive Auswertung von (11.3.3) wegen des hohen Aufwandes nicht in Betracht.

A. Der Algorithmus von Goertzel

Kernstück ist die folgende *Dreiterm-Rekursion* für $c_k := \cos(kt)$ und $s_k := \sin(kt)$:

$$\begin{aligned} c_{k+1} &= 2\,c_1\,c_k - c_{k-1}, \qquad k = 1, 2, \ldots, \\ s_{k+1} &= 2\,c_1\,s_k - s_{k-1}, \qquad k = 1, 2, \ldots \end{aligned} \tag{11.3.4}$$

Startwerte sind $c_0 := 1$, $c_1 := \cos t$, $s_0 := 0$ und $s_1 := \sin t$.

Mittels so genannter *adjungierter Summation* ergibt sich hieraus:

Algorithmus von Goertzel (11.3.5)

$u_n = 0$; $u_{n-1} = f_{n-1}$; $c_1 = \cos(t)$;
für $k = n-2, n-3, \ldots, 1$
$u_k := f_k + 2\,c_1\,u_{k+1} - u_{k+2}$,
end k;
$\sigma := f_0 + c_1\,u_1 - u_2$,
$\mu := u_1 \sin(t)$.

Gerald Goertzel:

Gerald Goetzel war ein US-amerikanischer theoretischer Physiker. Er wurde am 18.8.1919 geboren und starb am 17.7.2002 in White Plans bei New York. Er studierte am Stevens Institut of Technology (New Jersey) und promovierte an der New York University. Er arbeitete am Manhatten Project mit und war danach für viele Jahre am IBM's Research Division tätig.

Bemerkungen (11.3.5)

a) Der Algorithmus von Goertzel benötigt für jede Auswertung von $\sigma(t)$, $\mu(t)$ etwa n Multiplikationen und $2n$ Additionen. Zur Berechnung aller Fourier–Koeffizienten A_k, B_k, $k = 0, 1, \ldots, n$, werden demnach $O(n^2)$ elementare Operationen benötigt.

b) Für $t \approx k\,\pi$, $k \in \mathbb{Z}$, ist der Algorithmus von Goertzel numerisch instabil!
Es gibt jedoch eine geringfügig aufwendigerer stabile Variante des Goertzelschen Algorithmus.

c) Mit dem Algorithmus von Goertzel lässt sich zugleich die Aufgabe der **Interpolation durch trigonometrische Polynomen** lösen: Setzt man nämlich zu vorgegebenen Stützstellen (t_k, f_k), $k = 0, 1, \ldots, n-1$:

$$A_j := \frac{2}{n} \sum_{k=0}^{n-1} f_k \cos(k\,t_j)\,, \quad B_j := \frac{2}{n} \sum_{k=0}^{n-1} f_k \sin(k\,t_j)\,,$$

so interpolieren die folgenden trigonometrischen Polynome die vorgegebenen Daten: Für ungerades $n = 2\,m + 1$:

$$P_n(t) \;=\; \frac{A_0}{2} + \sum_{k=1}^{m} \left[A_k \cos(k\,t) + B_k \sin(k\,t)\right] .$$

Für gerades $n = 2\,m$:

$$P_n(t) \;=\; \frac{A_0}{2} + \sum_{k=1}^{m-1} \left[A_k \cos(k\,t) + B_k \sin(k\,t)\right] + \frac{A_m}{2} \cos(m\,t) .$$

B. Die schnelle Fourier–Transformation (FFT)

Beschränkt man sich auf die Auswertung von $\sigma(t)$ und $\mu(t)$ auf dem festen Gitter $t_j = jh$, $h = (2\,\pi)/n$, so lassen sich Algorithmen angeben, die dies mit deutlich geringerem Aufwand, nämlich mit $O(n \cdot \log_2 n)$ elementaren Operationen leisten.

Verfahren dieser Art heißen **Verfahren der schnellen Fourier–Transformation**. Sie gehen auf Cooley und Tukey (1969) zurück.

Zur Darstellung des Verfahrens verwenden wir die komplexe Schreibweise und berechnen zu vorgegebenen Funktionswerten $f_j = f(t_j) \in \mathbb{C}$ mit $t_j = j\,h$, $h = 2\pi/n$, die **diskreten (komplexen) Fourier–Koeffizienten**:

$$\Gamma_k := \frac{1}{n} \sum_{j=0}^{n-1} f_j \, \mathrm{e}^{-i\,j\,k\,2\pi/n}\,, \quad k = 0, 1, \ldots, n-1\,.$$

Für den Algorithmus setzen wir voraus, dass n eine Zweierpotenz ist: $n = 2^r$. Die Aufspaltung der obigen Summe (16.3.12) in zwei Teilsummen mit den geraden und dem ungeraden Indizes ergibt mit $m := n/2$:

$$\begin{aligned} \Gamma_k &= \frac{1}{n}\left(\sum_{j=0}^{m-1} f_{2j}\,\mathrm{e}^{-i\,(2j)\,k\,2\pi/n} + \sum_{j=0}^{m-1} f_{2j+1}\,\mathrm{e}^{-i\,(2j+1)\,k\,2\pi/n}\right) \\ &= G_k + \mathrm{e}^{-i\,k\,\pi/m}\,U_k, \quad k = 0,1,\ldots,n-1\,, \end{aligned}$$

wobei:

$$G_k := \frac{1}{n}\sum_{j=0}^{m-1} f_{2j}\,\mathrm{e}^{-i\,j\,k\,2\pi/m}\,, \quad U_k := \frac{1}{n}\sum_{j=0}^{m-1} f_{2j+1}\,\mathrm{e}^{-i\,j\,k\,2\pi/m}\,.$$

Die G_k und U_k brauchen nun aber nur für $k = 0,1,\ldots,m-1$ berechnet zu werden. Wegen der Periodizität der Exponentialfunktion gilt:

$$G_{k+m} = G_k, \quad U_{k+m} = U_k\,, \quad k = 0,1,\ldots,m-1\,.$$

Wir fassen den Reduktionsschritt nochmals zusammen:

Reduktionsschritt (11.3.6)

Anstelle der Berechnung der Γ_k, $k = 0, 1, \ldots, n-1$, berechne man mit $m := n/2$ für $k = 0, 1, \ldots, m-1$:

$$G_k = \frac{1}{n} \sum_{j=0}^{m-1} f_{2j}\, \mathrm{e}^{-i\,j\,k\,2\pi/m}, \quad U_k = \frac{1}{n} \sum_{j=0}^{m-1} f_{2j+1}\, \mathrm{e}^{-i\,j\,k\,2\pi/m}$$

$$\Gamma_k = G_k + \mathrm{e}^{-i\,k\,\pi/m}\, U_k, \quad \Gamma_{m+k} = G_k - \mathrm{e}^{-i\,k\,\pi/m}\, U_k\,.$$

Das Verfahren iteriert nun diesen Reduktionsschritt, bis nur noch triviale Fourier–Transformationen mit $m = 1$ auszuführen sind. Die einzelnen Fourier–Transformationen mit $m = 2, 4, 8, \ldots, 2^r$ werden dann nach (11.3.6) aus diesen zusammengesetzt. Der Algorithmus besteht daher aus zwei Teilen: In einem ersten Schritt werden die f_j so umsortiert, dass bei den anschließenden Reduktionsschritten jeweils benachbarte Werte zusammengefasst werden können.

Algorithmus (11.3.7) (FFT; 1. Teil)

$d_0 := f_0/n; \quad \overline{j} := 0;$
for $j = 1, 2, \ldots, n-1$
  for $m := n/2$ while $m + \overline{j} \geq n$ do $m := m/2;$
  $\overline{j} := \overline{j} + 3m - n;$
  $d_{\overline{j}} := f_j/n;$
end j

Erläuterung: Im Schritt Nr. j wird zum Index j der neue Index $\overline{j}$ berechnet. Dabei wird in der for–Schleife zunächst der „alte“ $\overline{j}$–Index auf führende Einsen getestet. Nach Durchlaufen der Schleife ist dann:

$$\begin{aligned} \overline{j} &= (1, \ldots, 1, 0, j_{\ell+2}, \ldots, j_r)_2 \\ m &= (0, \ldots, 0, 1, 0, \ldots 0)_2 \end{aligned}$$

Hieraus lässt sich nun leicht der „neue“ $\overline{j}$–Index berechnen.

Algorithmus (11.3.8) (FFT; 2. Teil)

for $\ell = 1, 2, \ldots, r$
  $m := 2^{\ell-1}$; $m_2 := 2\,m$;
  for $k = 0, 1, \ldots, m-1$
    $c := \exp(-i\,k\,\pi/m)$
    for $j = 0,\ m_2,\ 2\,m_2, \ldots,\ n - m_2$
      $g := d_{j+k}$; $u := c\,d_{j+k+m}$;
      $d_{j+k} := g + u$; $d_{j+k+m} := g - u$;
end $(\ell,\ k,\ j)$

James W. Cooley:
James W. Cooley ist ein US-amerikanischer Mathematiker. Er wurde am 18.9.1926 geboren, studierte am Manhattan College in New York und an der Columbia University, wo er auch promovierte. Ab 1953 arbeitete er am Institute for Advanced Studies in Princton, danach am Courant Institute der New York University und sodann für viele Jahre am IBM's Research Center.

John W. Tukey:
John W. Tukey war ein US-amerikanischer Mathematiker, der am 16.7 1915 in New Bedford (Massachusetts) geboren wurde und am 26.7.2000 in New Brunswick (New Jersey) starb. Er studierte an der Brown University (Rhode Island) und wurde an der Princeton University (New Jersey) promoviert. Später wurde er Professor in Princeton und einer der Grüder der dortigen Statistik Fakultät.