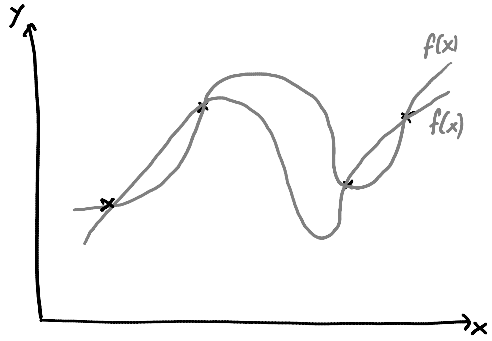
Lagrange and least squares

General fitting function:



$\forall \{x_i, y_i\}, i=1,\dots,N \rightarrow y_i = f(x_i)$

$$f(x) = \sum_{k=1}^{M} \alpha_k \cdot \phi_k(x)$$

$\phi_k(x) = x^{k-1} \quad \phi_k(x) = e^{\beta_k x} \quad \phi_k(x) = \dots$

Linear least squares:

Choose $\phi_k(x) \rightarrow$ pick α_k to solve: $f(x_i) = \sum_{k=1}^{M} \phi_k(x_i)\alpha_k = y_i \quad \forall i$

$$\underbrace{\begin{pmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & \phi_M(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \dots & \phi_M(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_N) & \phi_2(x_N) & \dots & \phi_M(x_N) \end{pmatrix}}_{\phi} \cdot \underbrace{\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_M \end{pmatrix}}_{\vec{\alpha}} = \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}}_{\vec{y}}$$

Lagrange Interpolation: $(M = N)$

$$l_k(x) = \frac{(x-x_1)(x-x_2)\dots(x-x_{k-1})(x-x_{k+1})\dots(x-x_N)}{(x_k-x_1)(x_k-x_2)\dots(x_k-x_{k-1})(x_k-x_{k+1})\dots(x_k-x_N)}$$

$\hookrightarrow l_k(x_k) = 1 \; ; \; l_k(x_i) = 0 \quad \forall i \neq k \rightarrow l = "\phi" = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$

$$f(x_i) = y_i = \sum_{k=1}^{N} \alpha_k l_k(x_i) = \alpha_i \rightarrow f(x) = \sum_{k=1}^{N} y_k \cdot l_k(x)$$

Error Minimization:

- $M = 1$, linear: $\underline{a} \cdot x = \underline{b} \qquad E = \sum_{i=1}^{N} (a_i - x b_i)^2 \qquad (a=x, b=y)$

  $\hookrightarrow x = (\underline{a}^T \cdot \underline{b})/(\underline{a}^T \cdot \underline{a})$

- $M < N$, linear: $\underline{\underline{A}} \cdot \underline{x} = \underline{b} \qquad E = \| \underline{\underline{A}}\underline{x} - \underline{b} \| \qquad (A = \phi)$

  $\hookrightarrow \underline{\underline{A}}^T\underline{\underline{A}} \cdot \underline{x} = \underline{\underline{A}}^T \cdot \underline{b} \rightarrow x = (\underline{\underline{A}}^T\underline{\underline{A}})^{-1}\underline{\underline{A}}^T\underline{b}$

- $M > N$, linear : $\underline{\underline{A}} \cdot \underline{x} = b$ $\quad E$ = "lenght"

  ↳ $\underline{p} = \underline{\underline{A}}(\underline{\underline{A}}^T\underline{\underline{A}})^{-1}\underline{\underline{A}}^T \cdot \underline{b} \rightarrow \underline{\underline{A}}\underline{x} = \underline{p} \rightarrow \underline{x} = \dots$

  projekt $\underline{b}$ on col.space of $\underline{\underline{A}}$

- Escamples : $\Delta x \cdot k = F \rightarrow \underset{\Delta x}{\begin{pmatrix}2\\3\\9\\5\end{pmatrix}} \cdot k = \underset{F}{\begin{pmatrix}4\\7\\9\\11\end{pmatrix}} \rightarrow k = (\Delta x^T \cdot \underline{F})/(\underline{\Delta x}^T \cdot \underline{x})$

$$C + D \cdot t_i = y_i \rightarrow \begin{pmatrix}1 & t_1\\1 & t_2\\ \vdots & \\1 & t_N\end{pmatrix} \cdot \begin{pmatrix}C\\D\end{pmatrix} = \begin{pmatrix}y_1\\y_2\\ \vdots \\y_N\end{pmatrix} \rightarrow \begin{pmatrix}N & \Sigma t_i\\ \Sigma t_i & \Sigma t_i^2\end{pmatrix}\begin{pmatrix}C\\D\end{pmatrix} \begin{pmatrix}\Sigma y_i\\ \Sigma y_i \cdot t_i\end{pmatrix}$$

$$\begin{pmatrix}\mu_1 & 0 & 0 & 0\\0 & \mu_2 & 0 & 0\\0 & 0 & 0 & 0\\0 & 0 & 0 & 0\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix} = \begin{pmatrix}b_1\\b_2\\b_3\end{pmatrix} \rightarrow p = \begin{pmatrix}b_1\\b_2\\0\end{pmatrix} \rightarrow \begin{pmatrix}\mu_1 & 0 & 0 & 0\\0 & \mu_2 & 0 & 0\\0 & 0 & 0 & 0\\0 & 0 & 0 & 0\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix} = \begin{pmatrix}b_1\\b_2\\0\end{pmatrix} \rightarrow \underline{x} = \begin{pmatrix}b_1/\mu_1\\b_2/\mu_2\\0\\0\end{pmatrix}$$

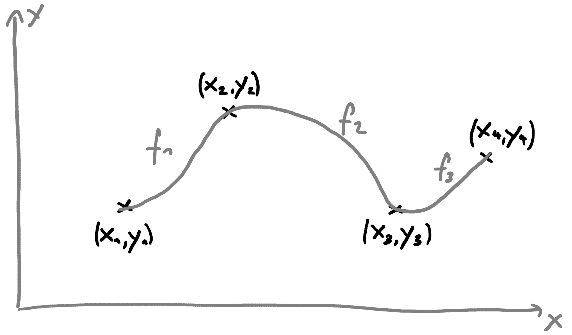
Splines

Cubic Splines:

$\{x_i, y_i\}$, $i = 1 \dots N$

$$f_i(x) = \begin{cases} x_i \le x < x_{i+1} : ax^3 + bx^2 + cx + d \\ \text{else} \quad : 0 \end{cases}$$

$$f(x) = \sum_{i=1}^{N-1} f_i(x)$$



$$f''(x) = f_i'' \cdot \frac{x_{i+1} - x}{\Delta x_i} + f_{i+1}'' \frac{x - x_i}{\Delta x_i} \qquad (\Delta x_i = x_{i+1} - x_i) \quad (\text{linear } f'' \text{ between } x_i \dots x_{i+1} \text{ with } f_i'' = f''(x_i))$$

↳ $$f(x) = f_i'' \cdot \frac{(x_{i+1} - x)^3}{6\Delta x_i} + f_{i+1}'' \frac{(x - x_i)^3}{6\Delta x_i} + C_i(x - x_i) + D_i$$

$$C_i = \frac{y_{i+1} - y_i}{\Delta x_i} - (f_{i+1}'' - f_i'') \cdot \frac{\Delta x_i}{6} \; ; \quad D_i = y_i - f_i'' \cdot \frac{\Delta x_i^2}{6} \qquad (\text{from } f(x_i) = y_i)$$

$$\frac{\Delta x_{i-1}}{6} \cdot f_{i-1}'' + \frac{\Delta x_{i-1} - \Delta x_i}{3} \cdot f_i'' + \frac{\Delta x_i}{6} \cdot f_{i+1}'' = \frac{\Delta y_i}{\Delta x_i} - \frac{\Delta y_{i-1}}{\Delta x_{i-1}} \qquad (\text{from } f \text{ continuous})$$

↳ $N-2$ equations for f_i''. → 2 boundary conditions

- natural : $f_1'' = f_N'' = 0$
- parabolic : $f_1'' = f_2''$; $f_N'' = f_{N-1}''$

$$\begin{pmatrix} 1 & \alpha & 0 & \cdots & & & 0 \\ \frac{\Delta x_1}{6} & \frac{\Delta x_1 - \Delta x_2}{3} & \frac{\Delta x_2}{6} & 0 & & & 0 \\ 0 & \frac{\Delta x_2}{6} & \frac{\Delta x_2 - \Delta x_3}{3} & \frac{\Delta x_3}{6} & 0 & \cdots & 0 \\ 0 & 0 & \frac{\Delta x_3}{6} & \frac{\Delta x_3 - \Delta x_4}{3} & \frac{\Delta x_4}{6} & \cdots & 0 \\ \vdots & & & \ddots & & & \vdots \\ 0 & \cdots & 0 & \frac{\Delta x_{N-1}}{6} & \frac{\Delta x_{N-1} - \Delta x_N}{3} & \frac{\Delta x_N}{6} & \\ 0 & & \cdots & 0 & \alpha & 1 & \end{pmatrix} \cdot \begin{pmatrix} f_1'' \\ f_2'' \\ f_3'' \\ f_4'' \\ \vdots \\ f_{N-1}'' \\ f_N'' \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{\Delta y_1}{\Delta x_1} - \frac{\Delta y_2}{\Delta x_2} \\ \vdots \\ \frac{\Delta y_{N-1}}{\Delta x_{N-1}} - \frac{\Delta y_N}{\Delta x_N} \\ 0 \end{pmatrix}$$

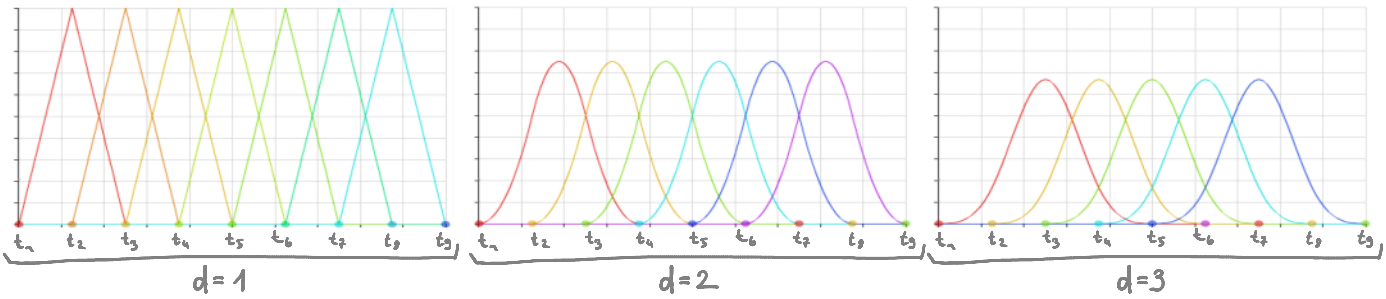
natural: $\alpha = 0$; parabolic: $\alpha = -1$

B-Splines:

$S_{d,t}(x) = \sum_{i=1}^{n} \alpha_i \cdot B_{i,d,t}(x)$ (Spline $S_{d,t}$ as summ of Basis Splines $B_{i,d,t}$)

d: polynomial degree in one Spline
i: # of B-Splines
t: $t_1, t_2, \dots, t_{m+d+1}$ knot vector = intervals of poly. func.

$$B_{i,0,t}(x) = \begin{cases} 1 & : \ t_i \le x \le t_{i+1} \\ 0 & : \ \text{else} \end{cases} \quad ; \quad B_{i,d,t}(x) = \frac{x - t_i}{t_{i+d} - t_i} \cdot B_{i,d-1,t}(x) + \frac{t_{i+d+1} - x}{t_{i+d+1} - t_{i+1}} \cdot B_{i+1,d-1,t}(x)$$



Clamping: $t_1, \dots, t_{d+1} = C_1$; $t_{n+1}, \dots, t_{M+d+1} = C_2 \Rightarrow B_{1,d,t}(t_1) = 1, \ B_{M,d,t}(t_{M+d+1}) = 1$
$B_{1,d,t}(t_{i+1}) = 0, \ B_{M,d,t}(t_{i \neq M+d+1}) = 0$

NURBS:

$\vec{p}_i = \{x_i, y_i\}$ $(i = 1, \dots, N)$ 2D-data points

$B_{i,d,t}(s)$ N B-Splines of deg. d
t_j $(j = 1, \dots, N+d+1)$ knot vector
ω_i $(i = 1, \dots, N)$ Weight of each data point

$p(s) = \{x(s), y(s)\}$ $(t_{d+1} \le s \le t_{N+1})$ curve filling data points

$$\vec{p}(s) = \sum R_{i,d,t}(s) \vec{p}_i \qquad R_{i,d,t}(s) = \frac{B_{i,d,t}(s)\,\omega_i}{\sum_{j=1}^{N} B_{j,d,t}(s)\,\omega_j} \qquad (\omega_i = 1 \rightarrow R_{i,d,t}(s) = B_{i,d,t}(s))$$

Basis Functions

Orthogonal Functions:

Increase Precision without changing lower-order Func.

$$y(x) = \sum_{i=1}^{M} a_i \varphi_i(x) \quad \text{with} \quad \langle \varphi_i(x), \varphi_j(x) \rangle = \int_{-\infty}^{\infty} \varphi_i(x) \cdot \varphi_j(x) \cdot p(x)\, dx = \delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

$$\hookrightarrow a_i = \frac{1}{N} \sum_{n=1}^{N} y_n \cdot \varphi_i(x_n)$$

Generate $\varphi_i(x)$ from set of $g_i(x)$ (Gram-Schmit):

① $\varphi_1(x) = g_1(x) \Big/ \frac{1}{N} \sum_{n=1}^{N} g_1(x_n) \cdot g_1(x_n)\, dx$

② $\tilde{\varphi}_2(x) = g_2(x) - \varphi_1(x) \cdot \frac{1}{N} \sum_{n=1}^{N} \varphi_1(x_n)\, g_2(x_n) dx$

③ $\varphi_2(x) = \tilde{\varphi}_2(x) \Big/ \frac{1}{N} \sum_{n=1}^{N} \tilde{\varphi}_2(x_n) \cdot \tilde{\varphi}_2(x_n)\, dx$

Radial Basis Functions:

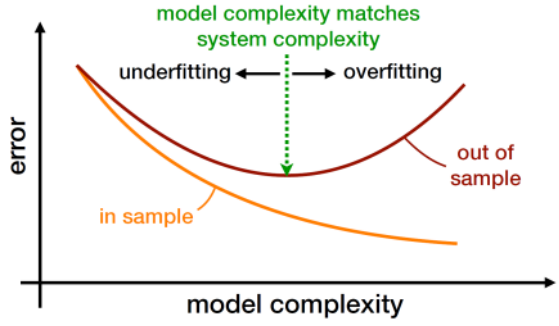
$$y(x) = \sum_{i=1}^{M} a_i \cdot \varphi(\|x - c_i\|)$$

φ: 1 basis function $\left(\varphi(r) = r\,;\, r^n\,;\, e^{-r^2}\,;\, \ldots\right)$

c_i: centers a_i: parameters

$$\begin{pmatrix} \varphi(\|x_1 - c_1\|) & \varphi(\|x_1 - c_2\|) & \cdots & \varphi(\|x_1 - c_M\|) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi(\|x_N - c_1\|) & \varphi(\|x_N - c_2\|) & \cdots & \varphi(\|x_N - c_M\|) \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ \vdots \\ a_M \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$$

Cross-Validation:



1: Divide training data $Z = \{x_i, y_i\,;\, i = 1 \ldots N\}$ in k samples of size $\approx N/k \Rightarrow Z_1, \ldots, Z_k$

2: For each Z_i use all other data to estimate model f_i then get square error on Z_i: $r_i = \frac{k}{N} \sum_{Z_i} (f_i(x) - y)^2$

3: Compute prediction error $R = \frac{1}{k} \sum_{i=1}^{k} r_i$

Nonlinear Systems

Goal: Find root of a function or a function-system.

Premise:

- $f(x)$ continuous + $\text{sign}(f(a)) \neq \text{sign}(f(b)) \Rightarrow f(x^*)=0 \; ; \; x^* \in [a,b]$
- Conditioning $1/f'(x^*)$: well-cond. ill-cond.
- Convergence rate: $E^{(k)} = x^{(k)} - x^*$: error ; (k) : k'th iteration

$$\lim_{k\to\infty} \frac{\|E^{(k+1)}\|}{\|E^{(k)}\|^r} = C \; ; \; C \neq \pm\infty$$

- $r=1 \rightarrow$ linear
- $r>1 \rightarrow$ superlin.
- $r=2 \rightarrow$ quadratic

Bisection Method

```
while (b − a) > tol do
    m ← (a + b)/2
    if sign(f(a)) = sign(f(m)) then
        a ← m
    else
        b ← m
    end if
end while
```

$\text{sign}(f(a)) \neq \text{sign}(f(b))$

tol: error tollerance

$r = 1$ (linear)

$C = 0.5$

Newtons Method

$x^{(k)}$: first guess

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

- $f'(x^{(k)}) \approx \frac{f(x^{(k)}) - f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$ (Secant Method)
- $f'(x^{(k)}) =$ actual (Newton Method) $(r = 1.839)$

Nonlinear Systems

$$\vec{F}(\vec{x}) = \begin{pmatrix} f_1(\vec{x}) \\ f_2(\vec{x}) \\ \vdots \\ f_N(\vec{x}) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \vec{0}$$

$$J(\vec{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\vec{x}) & \cdots & \frac{\partial f_1}{\partial x_N}(\vec{x}) \\ \vdots & & \vdots \\ \frac{\partial f_N}{\partial x}(x) & \cdots & \frac{\partial f_N}{\partial x_N}(\vec{x}) \end{pmatrix}$$

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - J^{-1}(\vec{x}^{(k)}) \cdot \vec{F}(\vec{x}^{(k)})$$

Input:

$\vec{x}^{(0)}$, {vector of length N with initial approximation}
tol, {tolerance: stop if $\|\vec{x}^{(k)} - \vec{x}^{(k-1)}\| < tol$}
k_{max}, {maximal number of iterations: stop if $k > k_{max}$}

Output:

$\vec{x}^{(k)}$, {solution of $\vec{F}(\vec{x}^{(k)}) = \vec{0}$ within tolerance tol} (or a message if $k > k_{max}$ reached)

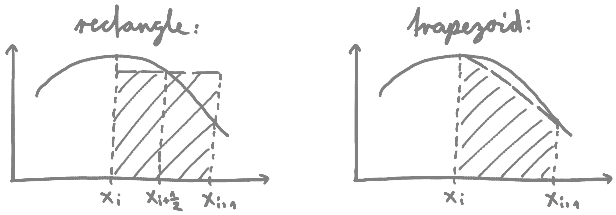
Steps:

```
k ← 1
while k ≤ k_max do
    Calculate F(x^(k−1)) and N × N matrix J(x^(k−1))
    Solve the N × N linear system J(x^(k−1))y = −F(x^(k−1))
    x^(k) ← x^(k−1) + y
    if ||y|| < tol then
        break
    end if
    k ← k + 1
end while
```

- **Modified Newton Method:** $J(\vec{x}^{(k)}) = J^0 = J(\vec{x}^{(0)}) \; \forall k$
- **Quasi Newton Method:** $J(\vec{x}^{(k+1)}) = J(\vec{x}^{(k)}) + \frac{(\Delta\vec{F} - J(\vec{x}^{(k)}) \cdot \Delta\vec{x})(\Delta\vec{x})^T}{(\Delta\vec{x})^T \cdot (\Delta\vec{x})}$

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

Numerical Integration



$$I = \int_a^b f(x)\,dx \approx \sum_{i=0}^{N-1} I_i$$

- Rectangle Rule: $I_{Ri} = f\left((x_i + x_{i+1})/2\right) \cdot \Delta_i$
- Trapezoidal Rule: $I_{Ti} = \frac{f(x_i) + f(x_{i+1})}{2} \cdot \Delta_i \rightarrow I \approx \frac{\Delta x}{2}\left(f_0 + 2\cdot\sum_{i=1}^{N-1} f_i + f_N\right)$
- Simpson's Rule: $I_{Si} = \frac{f(x_i) + 4 f\left((x_i + x_{i+1})/2\right) + f(x_{i+1})}{6} \cdot \Delta_i \rightarrow I \approx \frac{\Delta x}{3}\left(f_0 + 4\sum_{\substack{i=1 \\ \text{odd}}}^{N-1} f_i + 2\sum_{\substack{i=2 \\ \text{even}}}^{N-2} f_i + f_N\right)$

Newton-Cotes formulas: (integrate Lagrange approx.)

Pick $n+1$ points in $[a,b]$ $\left(x_i = a + i\cdot\frac{(b-a)}{n},\ i = 0,\dots,n\right)$

$P(x) = \sum_{k=0}^{n} f(x_k)\cdot L_k^n(x)$ (Lagrange approx.: $L_k^n(x) = \frac{(x-x_0)\dots(x-x_{k-1})(x-x_{k+1})\dots(x-x_N)}{(x_k-x_0)\dots(x_k-x_{k-1})(x_k-x_{k+1})\dots(x_k-x_N)}$)

$$I \approx (b-a)\cdot\sum_{k=0}^{n} C_k^n \cdot f(x_k) \quad ; \quad C_k^n = \frac{1}{b-a}\int_a^b L_k^n(x)\,dx$$

Properties of C_k:
- $\sum_{k=0}^{n} C_k^n = 1$
- $C_k^n = C_{n-k}^n$
- $n=1$: $C_0^1 = \frac{1}{2}$ $C_1^1 = \frac{1}{2}$
- $n=2$: $C_0^2 = \frac{1}{6}$ $C_1^2 = \frac{2}{3}$ $C_2^2 = \frac{1}{6}$

Error analysis:

$$f(x) = f(x_{i+1/2}) + \frac{1}{1!}(x - x_{i+1/2})\cdot f'(x_{i+1/2}) + \frac{1}{2!}(x - x_{i+1/2})^2 \cdot f''(x_{i+1/2}) + \frac{1}{3!}(x - x_{i+1/2})^3 \cdot f'''(x_{i+1/2}) + \dots$$ (Taylor series at $x_{i+1/2}$)

- Rectangle Rule: $I_{Ri} = I_i - \frac{1}{24} f'''(x_{i+1/2})\Delta_i^3 + \mathcal{O}(\Delta_i^5)$
- Trapezoidal Rule: $I_{Ti} = I_i + \frac{1}{12} f'''(x_{i+1/2})\Delta_i^3 + \mathcal{O}(\Delta_i^5)$
- Simpson's Rule: $I_{Si} = I_i + \mathcal{O}(\Delta_i^5)$

Richardson extrapolation / error estim.:

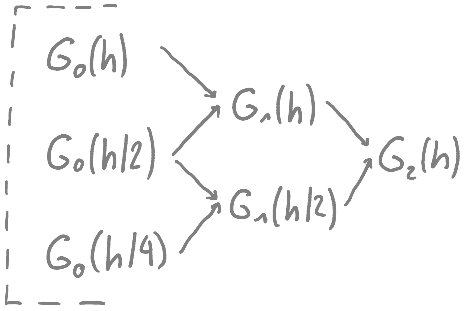
G: quantity to determine $G_0(h)$: Approximation depending on param. h $G_0(0) = G$

$$G_0(h) = G + c_1 h + c_2 h^2 + \dots \quad \text{(Taylor)}$$
$$G_0(h/2) = G + \tfrac{1}{2} c_1 h + \tfrac{1}{4} c_2 h^2 + \dots$$

$$G_1(h) = 2G_0(h/2) - G_0(h) = G + c_2' h^2 + c_3' h^3 = G + \mathcal{O}(h^2)$$
$$G_2(h) = \tfrac{1}{3}(4G_1(h/2) - G_1(h)) = \dots = G + \mathcal{O}(h^3)$$

$$G_n(h) = \frac{1}{2^n - 1}\left(2^n G_{n-1}(h/2) - G_{n-1}(h)\right) = G + \mathcal{O}(h^{n+1})$$

$$\varepsilon(h/2) \approx G(h/2) - G(h) \quad \text{(error estimation)}$$

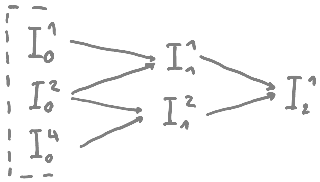


Romberg integration: (based on Richardson extrapolation)

$I_0^1, I_0^2, I_0^4, I_0^8, \dots \; (= G_0(h), G_0(\tfrac{h}{2}), G_0(\tfrac{h}{4}), \dots)$: Rect, Trap or Simps. interp. on 1, 2, 4, 8.. segments

$$I_1^n = \frac{4I_0^{2n} - I_0^n}{3} \quad (= G_2(\tfrac{h}{n}) \text{ as } G_1 = G_0)$$

$$I_K^n = \frac{4^K I_{K-1}^{2n} - I_{K-1}^n}{4^K - 1} \quad (\varepsilon(k) = I_K^2 - I_K^1)\ ?$$



Adaptive Quadrature:

Algorithm 1 Adaptive integration.

Steps:

Subdivide the interval of the integration into sub-intervals
for all sub-intervals **do**
  Compute sub-integral, estimate the error with Richardson procedure described earlier.
  if accuracy is worse than desired **then**
    Subdivide the interval
  else
    Leave the interval untouched
  end if
end for

"be more accurate where error is large."

2-Point Gauss quadrature rule: (interp. rule like Rect, Trap, Simps.)

$$\int_a^b f(x)\,dx \approx c_1 f(x_1) + c_2 f(x_2) \qquad (f(x) \approx a_0 + a_1 x + a_2 x^2 + a_3 x^3)$$

$$\rightarrow c_1 = \frac{b-a}{2}\,,\quad c_2 = \frac{b-a}{2}\,,\quad x_1 = \frac{-1}{\sqrt{3}}\cdot\frac{b-a}{2} + \frac{b+a}{2}\,,\quad x_2 = \frac{1}{\sqrt{3}}\cdot\frac{b-a}{2} + \frac{b+a}{2}$$

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

Hermite interpolation: (Gauss quadrature)

Given $i = 1, \dots, n$ Points find $2n-1$ deg polynom $f(x)$.

$$y_i = f(x_i),\ y_i' = f'(x_i) \rightarrow f(x) = \sum_{k=1}^{n} U_k(x)\cdot y_k + \sum_{k=1}^{n} V_k(x)\, y_k' \leftarrow \begin{pmatrix} U_k(x_j) = \delta_{kj} & U_k'(x_j) = 0 \\ V_k(x_j) = 0 & V_k'(x_j) = \delta_{kj} \end{pmatrix}$$

$$U_k(x) = \left(1 - 2\,L_k'(x_k)\cdot(x - x_k)\right)\cdot L_k(x)^2$$
$$V_k(x) = (x - x_k)\cdot L_k(x)^2$$
} Hermite Polinomials ?

$$\int_{-1}^{1} f(x)\,dx = \sum_{k=1}^{n} U_k \cdot f(x_k) \qquad \left(U_k = \int_{-1}^{1} U_k(x)\,dx \quad V_k = \int_{-1}^{1} V_k(x)\,dx = 0\right)$$

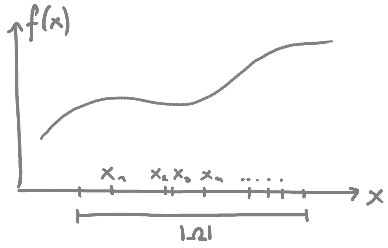
$$U_k = 2 / (1 - x_k^2)\,(P_n'(x_k))^2 \qquad (P_n: \text{Legendre polynomials})$$

$$\varepsilon = \frac{2^{2n+1}\,(n!)^4}{(2n+1)\,(2n!)^3}\cdot f^{(2n)}(\xi) \qquad (\text{Error})$$

Montecarlo sampling:

$$I = |\Omega| \cdot \langle f \rangle \approx |\Omega| \cdot \frac{1}{M} \cdot \sum_{i=1}^{n} f(x_i) = |\Omega| \cdot \langle f \rangle_M$$

($|\Omega|$: interval width; $\langle f \rangle$: avg. val; x_i: M random points in interval)



$$\varepsilon_M = \sqrt{\frac{1}{M-1}\cdot\left(\langle f^2 \rangle_M - \langle f \rangle_M^2\right)} \qquad (\text{error est.}: \langle f \rangle - \langle f \rangle_n < \varepsilon_M \text{ with } 68\%)$$