

If denominator recomputed:
 N^2 FLOPS needed to evaluate $Q(x)$ at some x
 If EXACT Model is known: (cheap setup)
 We can directly minimize:
 $E_2 = \sum \sum \left[\sum (\alpha x_i + \beta) - Q(x_i) \right]^2$
 LSQR Model exact model

CMEAT:

Function Fitting:

- ↳ Data $\{x_i, y_i\}, i=1, \dots, N \xrightarrow{\text{FIND}} f(x)$, such that $f(x_i) = y_i$
- ↳ ANSATZ: $f(x) = \sum_{q=1}^M \alpha_q \Phi_q(x)$
 - $\alpha_q \dots$ coeffs
 - $\Phi_q \dots$ basis func.
- ↳ Example: $\Phi_q = x^{q-1}, \Phi_q = e^{\beta_q x}, \Phi_q = \cos((q-1)x)$

$$f(x_i) = y_i = \sum \alpha_q \Phi_q(x_i)$$

$$\begin{pmatrix} \Phi_1(x_1) & \dots & \Phi_M(x_1) \\ \vdots & & \vdots \\ \Phi_1(x_N) & \dots & \Phi_M(x_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_M \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$$

INTERPOLATION HAS TO PASS THROUGH DATA

CASE: $\begin{cases} M=N \rightarrow \text{exact} \\ M < N \rightarrow \text{LSQ} \\ M > N \rightarrow \text{SVT} \end{cases}$

Lagrange Interpolation:

- ↳ Construct N polynomials of degree $(q-1)$, s.t. $l_q(x_i) = \delta_{qi}$
- ↳ $l_q(x) = \frac{(x-x_1)(x-x_2)\dots(x-x_{q-1})(x-x_{q+1})\dots(x-x_N)}{(x_q-x_1)(x_q-x_2)\dots(x_q-x_{q-1})(x_q-x_{q+1})\dots(x_q-x_N)}$
- ↳ This choice yields $\alpha_i = y_i \rightarrow f(x) = \sum_{q=1}^N y_q l_q(x)$
- ↳ Error: $|y(x) - f(x)| = \left| \frac{y^{(N)}(\xi)}{N!} \prod_{i=1}^N (x-x_i) \right|$, for some $\xi \in [x_1, x_N]$

- ↳ Properties: i) Cannot handle noise (high order polynomial)
- ii) Minimize error using the roots of n -degree Chebyshev polynomials as sampling points.

PERFECT INTERPOLATION OF A STRAIGHT LINE

$$x_i = -\cos\left(\frac{2i-1}{n} \pi\right)$$

$$\text{iii) Extension higher dim.: } l_q(x, y) = \prod_{i \neq q} \frac{(x-x_i)^2 + (y-y_i)^2}{(x_q-x_i)^2 + (y_q-y_i)^2}$$

Linear least Squares (LSQ): (COEFF. MUST ENTER LINEARLY)

- ↳ Problem: $N \begin{bmatrix} A \\ 1 \end{bmatrix} \begin{bmatrix} x \\ b \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $A \in \mathbb{R}^{N \times M}, M < N \rightarrow$ inconsistent system

- ↳ Idea: Minimize average error: $\text{grad}(E) = \text{grad}(\|Ax - b\|) = 0$

- ↳ Solution: For the optimal $\vec{x}$, the vector $A\vec{x} - b$

$P = A(A^T A)^{-1} A^T$ projection matrix

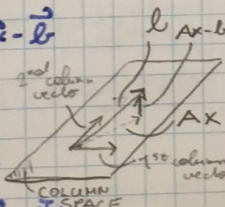
$\vec{p} = P\vec{b}$ is the projection of $\vec{b}$ of the column space of A !

$P^2 = P, P^T = P$

$$(A\vec{g})^T (A\vec{x} - \vec{b}) = 0 = \vec{g}^T (A^T A \vec{x} - A^T \vec{b})$$

$$A^T A \vec{x} = A^T \vec{b} \rightarrow \vec{x} = (A^T A)^{-1} A^T \vec{b}$$

- ↳ Example: Fit $\{x_i, y_i\}$ using $Q(x) = A \exp(-[\frac{x-\mu}{\sigma}]^2) \rightarrow \log(Q(x)) = \log A - (\frac{x-\mu}{\sigma})^2$



Cubic Splines:

$$\Delta_i = x_{i+1} - x_i$$

- ↳ Idea: Global func. $\rightarrow$ high oscillatory (no noise resistance), instead use piecewise low order polynomials $\rightarrow$ cubic (for instance)
- ↳ $\{x_i, y_i\}_{i=1, \dots, N} \rightarrow$ Cubic Pol. for $x \in [x_i, x_{i+1}] \forall i \rightarrow f''(x) =$ piecewise linear
- ↳ $f''(x_i) = f''_i \hat{=} N$ UNKNOWN, $f''(x) = f''_i \frac{(x_{i+1}-x)}{\Delta_i} + f''_{i+1} \frac{(x-x_i)}{\Delta_i}, x \in [x_i, x_{i+1}]$
- ↳ $f(x) = f''_i \frac{(x_{i+1}-x)^3}{6\Delta_i} + f''_{i+1} \frac{(x-x_i)^3}{6\Delta_i} + (\frac{y_{i+1}-y_i}{\Delta_i} - (f''_{i+1}-f''_i)\frac{\Delta_i}{6})(x-x_i) + (y_i - f''_i \frac{\Delta_i^2}{6})$
- ↳ $f'(x) = f''_i [\frac{(x-x_i)^2}{2\Delta_i} - \frac{\Delta_i}{6}] - f''_{i+1} [\frac{(x_{i+1}-x)^2}{2\Delta_i} - \frac{\Delta_i}{6}] + \frac{y_{i+1}-y_i}{\Delta_i}$
- ↳ Tri-diagonal system: $\Delta_{i-1} f''_{i-1} + 2(\Delta_{i-1} + \Delta_i) f''_i + \Delta_i f''_{i+1} = \frac{y_{i+1}-y_i}{\Delta_i} - \frac{y_i-y_{i-1}}{\Delta_{i-1}}$
- ↳ B.C.: i) Natural: $f''_1 = f''_N = 0$ ii) Periodic: $f''_1 = f''_N$ $N-2$ equations
- iii) Clamped: $f'_1 = f'_N = 0$ iii) Periodic: $f''_1 = f''_N$

B-Splines: (Basis for NURBS, which solve recomputation problem)

- ↳ M B-Splines (degree d) require a knot vector $t \in \mathbb{R}^{M+d+1}$

- ↳ RECURSIVE: $B_{i,0} = \begin{cases} 1, & x \in [t_i, t_{i+1}] \\ 0, & \text{else} \end{cases}$

$$\sum_{i=1}^M B_{i,d} = 1 \forall x$$

$$B_{i,d} = \frac{x-t_i}{t_{i,d}-t_i} B_{i,d-1}(x) + \frac{t_{i,d+1}-x}{t_{i,d+1}-t_{i+1}} B_{i+1,d-1}(x)$$

- ↳ Any Spline can be written as $S_d(x) = \sum_{i=1}^M \alpha_i B_{i,d}(x)$

- ↳ Continuous derivatives up to degree $(d-1)$ (if $t_i \dots$ distinct)

- ↳ Clamped: Set first and last $(d+1)$ entries of knot vector equal!

NURBS:

- ↳ Use B-Splines to define parameterization of interpolating curve

- ↳ N data points $\vec{p}_i = (x_i, y_i)$, define N B-Splines (degree d)

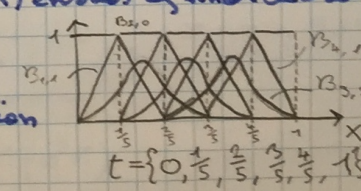
$$\text{Then } \vec{p}(s) = \sum_{i=1}^N R_{i,d}(s) \vec{p}_i, R_{i,d} = \frac{B_{i,d}(s) \cdot w_i}{\sum B_{i,d}(s) \cdot w_i}, w_i = \text{weight}$$

- ↳ If $w_i = 1 \rightarrow B_{i,d} = R_{i,d}$

$$\vec{p}(s) = \begin{pmatrix} x(s) \\ y(s) \end{pmatrix}$$

Result lies on different space!

B-Splines $\rightarrow s \rightarrow x(s)$



Unusually: $t \in [0, 1]$ equally spaced!
 Every dupl. entry $\rightarrow$ deg -1
 CONVENTION: $0 = 0$
 So $t \in \mathbb{R}^{m+d+1}$

Multivariate Interpolation:

→ Data: $\{x_a, y_a, z_a\}$, find $f(x, y) = \sum_i \sum_j x_{ij} \Phi_i(x) \Phi_j(y)$
such that $z_a = \sum_i \sum_j \alpha_{ij} \Phi_i(x_a) \Phi_j(y_a)$

→ Using NURBS: $\vec{p}_{ij} = \begin{pmatrix} x_{ij} \\ y_{ij} \\ z_{ij} \end{pmatrix}_{i=1, \dots, N, j=1, \dots, M}$, $du dv \rightarrow \vec{t}_u \vec{t}_v$

$$\vec{p}(u, v) = \sum_i \sum_j R_i du(u) R_j dv(v) \cdot \vec{p}_{ij}$$

→ Shepard's method: $f(x, y) = \frac{\sum_{a=1}^N (z_a \cdot g(x - x_a, y - y_a))}{\sum_{a=1}^N g(x - x_a, y - y_a)}$
 $g(x, y) \dots$ usually radially symmetric!
e.g. $g(x, y) = (x^2 + y^2)^{-\frac{\mu}{2}}$, $\mu \in \mathbb{R}$

→ Coon's patch: "Warehouse of industry" (70s/80s)

Known: Values on a square, i.e. $z(x, 0), z(0, y), z(x, 1), z(1, y)$

$$\begin{aligned} z(x, y) = & y \cdot z(x, 1) + (1-y) \cdot z(x, 0) + x \cdot z(1, y) + (1-x) \cdot z(0, y) \\ & - (1-x)(1-y) \cdot z(0, 0) - (1-x) \cdot y \cdot z(0, 1) - (1-y) \cdot x \cdot z(1, 0) \\ & - x \cdot y \cdot z(1, 1) \end{aligned}$$

Orthogonal Basis Functions: (Recall: $f(x) = \sum \alpha_i \Phi_i$)

→ Advantage: Each α_i depends only on Φ_i

→ CASE 1: Functional form $y(x) = \text{KNOWN}$, $x \in \Omega$

FOURIER'S TRICK

$$\begin{aligned} \langle y(x) | \Phi_j \rangle &= \langle \sum \alpha_i \Phi_i | \Phi_j \rangle = \int_{\Omega} \Phi_i(x) \Phi_j(x) dx = \delta_{ij} \\ &= \sum \alpha_i \langle \Phi_i | \Phi_j \rangle = \sum \alpha_i \delta_{ij} = \alpha_j = \langle y(x) | \Phi_j(x) \rangle \end{aligned}$$

→ CASE 2: Data not given $\{x_i, y_i\}$, $i = 1, \dots, N$

Assume: $y_i = y(x_i) \cdot p(x_i)$ $\begin{cases} y(x) \dots \text{True correlation} \\ p(x) \dots \text{Prob. density of measurements} \end{cases}$

$$\text{Claim: } p(x) = \frac{1}{N} \sum_{i=1}^N \delta(x - x_i)$$

$$\text{Choose: } \langle \Phi_i | \Phi_j \rangle_p = \int_{-\infty}^{\infty} \Phi_i \Phi_j p(x) dx = \delta_{ij}$$

$$\alpha_j = \frac{1}{N} \sum_{i=1}^N y_i \Phi_j(x_i)$$

Gram-Schmidt: (Construction of orthonormal set)

→ Choose appropriate scalar product: $\langle \cdot | \cdot \rangle \rightarrow \text{Norm} \|\cdot\| = \sqrt{\langle \cdot | \cdot \rangle}$

→ STEPS: i) $\Phi_1 = \frac{g_1}{\|g_1\|}$... "normalized"
ii) $\tilde{\Phi}_2 = g_2 - \Phi_1 \langle \Phi_1 | g_2 \rangle$... "orthogonalized"
 $\Phi_2 = \frac{\tilde{\Phi}_2}{\|\tilde{\Phi}_2\|}$... "normalized"
iii) $\tilde{\Phi}_3 = g_3 - \Phi_1 \langle \Phi_1 | g_3 \rangle - \Phi_2 \langle \Phi_2 | g_3 \rangle$
 $\Phi_3 = \frac{\tilde{\Phi}_3}{\|\tilde{\Phi}_3\|}$

ORIGINAL SET
 $\{g_1, g_2, g_3\}$

→ Examples: i) Hermite ... Orthogonal w.r.t. e^{-x^2}

ii) Laguerre ... w.r.t. e^{-x}

iii) Chebyshev ... w.r.t. $\sqrt{1-x^2}$

Radial Basis Functions: (RBF)

→ Avoid oscillating behavior → Choose RBF that depend on

distance from some center: $y(x) = \sum_i \Phi_i(x; c_i, \sigma_i) \rightarrow y(x) = \sum_i \alpha_i \Phi_i$

Center can be arbitrarily chosen!

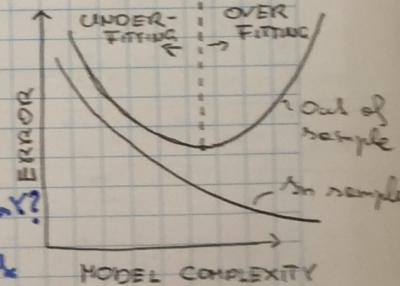
Choosing a model (Gen-Validation):

→ Until now: How to fit model X, Y
to same data?

→ Now: How to choose between model X or Y ?

→ Idea: i) "Train" the model using one part

ii) Test the model using the remaining part



→ K-Fold Gen-Validation:

i) Divide Data $Z = \{(x_i, y_i), i=1, \dots, N\}$ into K samples $Z_1, \dots, Z_K$ (size $\sim \frac{N}{K}$)

ii) For every Z_i : → Use remaining data to train model (i.e. $Z_i = Z \setminus \{Z_i\}$)
→ Average error for data in Z_i : $\alpha_i = \frac{1}{N} \sum_{j \in Z_i} (f_j(x_j) - y_j)^2$

iii) Estimate error by averaging α_i : $R = \frac{1}{K} \sum_i \alpha_i$

DATA
MODEL

PROOF:

$$\begin{aligned} \langle y(x) | \Phi_j \rangle_p &= \sum_i \alpha_i \langle \Phi_i | \Phi_j \rangle_p \\ &= \sum_i \alpha_i \delta_{ij} = \alpha_j \\ &= \langle y(x) | \Phi_j(x) \rangle_p \end{aligned}$$

Small residual $\|f(\tilde{x})\| \Leftrightarrow$ Small error $\|\tilde{x} - x^*\|$

ONLY for well conditioned problems

Non-Linear Equations: (1D)

↳ General Problem: Find x^* , such that $f(x^*) = 0$

↳ Existence: If f ... continuous & there is an interval where $f(\cdot)$ differs in endpoints $\rightarrow$ Intermediate Theorem

↳ Conditioning number: $\kappa = \frac{1}{|f'(x^*)|}$ $\rightarrow$ HOW MUCH DOES OUTPUT CHANGE FOR SMALL CHANGES IN INPUT?

↳ Convergence Rate: $\lim_{q \rightarrow \infty} \left\| \frac{f(x^{(q+1)})}{(f(x^{(q)}))^n} \right\| = C$, $C \neq \{0, \infty\}$

↳ Error estimation: If x^* unknown $\rightarrow \|f_{q+1}\| \sim \|x^{(q+1)} - x^*\|$

Bisection Method: ("Intermediate Value Theorem")

↳ Converges, but slowly: $n=1$, $C=\frac{1}{2}$

↳ After k iterations $\rightarrow$ Interval size $\frac{b-a}{2^k}$

↳ To achieve "tolerance" it takes $m = \lceil \log_2 \frac{b-a}{\text{tol}} \rceil$

ALGORITHM:
while $(b-a) > \text{tol}$
 $m = \frac{a+b}{2}$
 if $(f(a) \cdot f(m)) > 0$
 $a = m$
 else
 $b = m$
end while

Newton's Method:

↳ Initial guess $x^{(0)} \rightarrow$ Use 1st-order Taylor exp. to estimate $f^{(2+1)} \stackrel{!}{=} 0 = f(x^{(0)}) + f'(x^{(0)}) \cdot (x^{(2+1)} - x^{(0)})$

↳ $x^{(q+1)} = x^{(q)} - \frac{f(x^{(q)})}{f'(x^{(q)})}$ $\xrightarrow[\text{MULTIPLICITY } m]{\text{ROOTS WITH}}$ $x^{(q+1)} = x^{(q)} - m \frac{f(x^{(q)})}{f'(x^{(q)})}$

↳ Quadratic Convergence ($n=2$) / Problem: derivative

Secant Method: ($n \sim 1, 6-8$) "finite difference for $f'(x)$ "

↳ $f'(x^{(q)}) = \frac{f(x^{(q)}) - f(x^{(q-1)})}{x^{(q)} - x^{(q-1)}}$ $\rightarrow x^{(q+1)} = x^{(q)} - \frac{f(x^{(q)})}{f'(x^{(q)})}$

Systems of non-linear equations: $\vec{F}(\vec{x}) \stackrel{!}{=} \vec{0}$, $f'(x) \rightarrow J(x)$

↳ Taylor Series: $\vec{F}(\vec{x}^{(q+1)}) \stackrel{!}{=} \vec{0} = \vec{F}(\vec{x}^{(q)}) + J(\vec{x}^{(q)}) \cdot (\vec{x}^{(q+1)} - \vec{x}^{(q)})$

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

where $J(\vec{x}^{(q)}) \cdot \vec{y}^{(q)} = -\vec{F}(\vec{x}^{(q)})$

↳ Problem: $O(N^2)$... build J , $O(N^3)$... solve $Jy = -F$

↳ Modified Newton: Keep $J^{(0)}$ fixed $\rightarrow$ LU decomp. (rel. $O(N^2)$)

↳ Quasi-Newton: Update $J^{(q)} = J^{(0)} + \frac{(\Delta F - J^{(0)} \Delta x) \Delta x^T}{\Delta x^T \Delta x}$

Numerical Integration: ($I = \int_a^b f(x) dx = ?$)

↳ Idea: Divide $[a, b]$ into subintervals $\rightarrow I = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x) dx = \sum I_i$

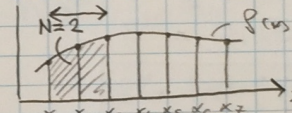
EVEN $N = \frac{b-a}{\Delta_i} = \# \text{ intervals}$, $\Delta_i = x_{i+1} - x_i = \text{const}$; $(N+1)$ points $\{x_0, \dots, x_N\}$

↳ Rectangle: $I_{R_i} = f\left(\frac{x_i + x_{i+1}}{2}\right) \Delta_i$
 $I = \sum I_{R_i} = \sum_{i=0}^{N-1} f(x_{i+1/2}) \cdot [x_{i+1} - x_i]$

↳ Trapezoidal: $I_{T_i} = \frac{1}{2} \Delta_i (f(x_i) + f(x_{i+1}))$
 $I = \sum I_{T_i} = \frac{1}{2} \Delta_x (f_0 + \sum_{i=1}^{N-1} f_i + f_N)$

↳ Simpson: $I_{S_i} = \frac{1}{6} \Delta_i (f(x_i) + 4f(\frac{x_i + x_{i+1}}{2}) + f(x_{i+1}))$
 $I = \sum I_{S_i} = \frac{1}{3} \Delta_x (f_0 + 4 \sum_{i=1}^{N-1} f_i + 2 \sum_{i=2}^{N-2} f_i + f_N)$

↳ Newton-Cotes-Formula: Divide into subintervals containing $(N+1)$ points $\rightarrow$ Approximate $f(x)$ using Lagrange-Subpolynomials!



$I_i \approx \int_{x_i}^{x_{i+1}} L(x) dx = \sum_{q=0}^N y_q \int_{x_i}^{x_{i+1}} l_q(x) dx = (b-a) \sum_{q=0}^N C_q^N y_q$, $C_q^N = \frac{1}{b-a} \int_a^b l_q(x) dx$
It holds: $\sum C_q^N = 1$ & $C_q^N = C_{N-q}^N$ BAD NOTATION $x_i = a, x_{i+N} = b$ $R = 0, \dots, N$

Error Analysis: (TAYLOR: $f(x) = f(x_{i+1/2}) + f'(x_{i+1/2})(x - x_{i+1/2}) + \frac{1}{2} f''(\xi) \Delta_i^2$)

↳ $I_{R_i} = I_i - \frac{1}{24} f''(x_{i+1/2}) \Delta_i^3 + O(\Delta_i^5)$
↳ $I_{T_i} = I_i + \frac{1}{12} f''(x_{i+1/2}) \Delta_i^3 + O(\Delta_i^5)$
↳ $I_{S_i} = \frac{2}{3} I_{R_i} + \frac{1}{3} I_{T_i} = I_i + O(\Delta_i^5)$

$I_N = \sum I_{R_i} \rightarrow E = \sum |I_{R_i} - I_i| = \sum_{i=0}^{N-1} \left(\frac{1}{24} f''(x_{i+1/2}) \Delta_i^3 + O(\Delta_i^5) \right)$
 $E < \sum_{i=0}^{N-1} \left(\frac{1}{24} \max\{f''(x_{i+1/2})\} \Delta_i^3 + O(\Delta_i^5) \right) = N \cdot \left(\frac{1}{24} \max\{f''(x_{i+1/2})\} \Delta_x^3 + O(\Delta_x^5) \right)$
 $= \frac{b-a}{24} \max\{f''(x_{i+1/2})\} \Delta_x^2 + O(\Delta_x^4)$ since $N = \frac{b-a}{\Delta_x}$

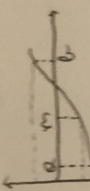
1st-order Romberg extrapolation of Simpson's rule

RECALL: $l_q(x) = \frac{(x-x_0) \dots (x-x_{q-1})(x-x_{q+1}) \dots (x-x_N)}{(x_q-x_0) \dots (x_q-x_{q-1})(x_q-x_{q+1}) \dots (x_q-x_N)}$

When summing all regions, order will be reduced by 1

$\frac{\partial \vec{F}}{\partial \vec{x}}$
 J

ILL-CONDITIONED
WELL-CONDITIONED



REDUCES ERROR
BY ORDER OF MAG.
BY ONLY ONE SUBTR.

Richardson Extrapolation:

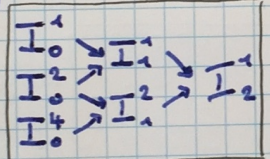
- Goal: Approx. G^* using a method depending on some $Q, G(Q)$
- Taylor: $G(Q) = G^* + c_1 Q + c_2 Q^2 + \dots$
 $G(\frac{Q}{2}) = G^* + \frac{1}{2}c_1 Q + \frac{1}{4}c_2 Q^2 + \dots$
 $\rightarrow G_1(Q) = 2G(\frac{Q}{2}) - G(Q) = G^* - \frac{1}{2}c_2 Q^2$
- RECURSIVE: $G_n(Q) = \frac{1}{2^n - 1} [2^n G_{n-1}(\frac{Q}{2}) - G_{n-1}(Q)] = G^* + O(Q^{n+1})$
 $G_0 = G(Q)$

Error Estimation:

- $G(\frac{Q}{2}) - G(Q) = -\frac{1}{2}c_1 Q - \frac{3}{4}c_2 Q^2 + \dots$
 $\rightarrow E(\frac{Q}{2}) \sim G(\frac{Q}{2}) - G(Q)$
- $G^* - G(\frac{Q}{2}) = -\frac{1}{2}c_1 Q - \frac{1}{4}c_2 Q^2 + \dots$
 $= E(\frac{Q}{2})$

Romberg Integration: (= Richardson Extrap. for TRAPEZOIDAL)

- $I_0^n = I - c_1 (\frac{b-a}{n})^2 - c_2 (\frac{b-a}{n})^4 + \dots$
 $I_1^n = \frac{4I_0^{2n} - I_0^n}{3} = I - \frac{c_2}{4} (\frac{b-a}{n})^4 + \dots$
- $I_2^n = I - \frac{1}{4}c_1 (\frac{b-a}{n})^2 - \frac{1}{16}c_2 (\frac{b-a}{n})^4 + \dots$
 $\rightarrow I_Q^n = \frac{4^n I_{2^n} - I_{2^{n-1}}}{4^n - 1}$



Adaptive Quadrature:

- Idea: Error of subintervals varies! Instead of dividing every subinterval to gain accuracy $\rightarrow$ Set acceptable error per subinterval, approximate error (Richardson Extrapolation) & only divide further if necessary!

Gauss Quadrature:

- $\int_a^b f(x) dx = \sum_{i=1}^N c_i f(x_i)$ (abstruse)
- Find $\{c_i, x_i\}$ by asking integral of $(2N-1)$ -deg. pol. to be exact!
- General Approach: Transform to $[-1, 1]$: $\xi = \frac{2x - (a+b)}{b-a}$
- In this interval: Abstruse $\hat{=}$ Roots of n^{th} -Legendre Pol. $P_n(x)$
- Then: $\int_{-1}^1 f(x) dx = \sum_{a=1}^N u_a f(x_a)$, $u_a = \frac{2}{(1-x_a^2) \cdot (P_n'(x_a))^2}$

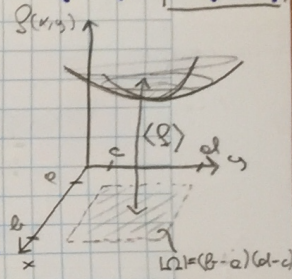
2 point Example: $n=2 \rightarrow 3$ nodes pol
 $Q(x) = a_0 + b_1 x + c_2 x^2 + d_3 x^3$
 $\int_a^b Q(x) dx = c_1 Q(x_1) + c_2 Q(x_2)$
 $\{c_1, c_2, x_1, x_2\} \rightarrow$ Comp. (diff. eqs.)

$P_0(x) = 1, P_1(x) = x, P_2(x) = \frac{1}{2}(3x^2 - 1), P_3(x) = \frac{1}{2}(5x^3 - 3x), P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} [(x^2 - 1)^n]$

Divide
Steps

Higher dimensional numeric integration: $I = \int_{\Omega} f(\vec{x}) d\vec{x}, \vec{x} = (x_1, \dots, x_d)$

- Tubini $\rightarrow$ Split into "nested integrals" and use 1-dim quadrature rules
 $2D: \int_a^b \int_c^d f(x, y) dy dx = \int_a^b \sum_{i=1}^N w_i^y f(x, y_i) dx = \sum_{i=1}^N \sum_{j=1}^N w_i^x w_j^y f(x_i, y_j)$
- Problem: * function evaluation $M = n^d$ (dimension quadrature points)
- Error for some rule: $E = I - \bar{I} = O(Q^S) \propto O(n^{-S}) \propto O(M^{\frac{S}{d}})$



Monte Carlo Quadrature:

- Idea: $I = |\Omega| \cdot \langle f \rangle$ { $|\Omega| \dots$ Integ. domain, $\langle f \rangle \dots$ average of integrand }
- Approximate $\langle f \rangle$ using M "Samples": $\{x_i\}$
 $\langle f \rangle \sim \langle f \rangle_M = \frac{1}{M} \sum_{i=1}^M f(x_i)$
 $\{ \langle f \rangle_M = \langle f \rangle, P(\lim_{M \rightarrow \infty} \langle f \rangle_M = \langle f \rangle) = 1 \}$

- Error: "Standard deviation of $\langle f \rangle_M - \langle f \rangle$ "
 $E_M = \sqrt{\text{Var}(\langle f \rangle_M - \langle f \rangle)} = \sqrt{\text{Var}(\langle f \rangle_M)} = \sqrt{\frac{\text{Var}(f)}{M}}$

- $E_M = O(M^{-\frac{1}{2}}) \rightarrow 2 \cdot \text{accuracy} \hat{=} 4 \cdot \text{function evaluation}$
- Note: $E_M \neq$ "Strict bound" (since standard deviation)

- Recipe: i) Draw M random samples $x_i \in \Omega$ $|\langle f \rangle - \langle f \rangle_M| < \begin{cases} E_M, 68\% \\ 2E_M, 95\% \\ 3E_M, 98\% \end{cases}$
- ii) Store $M, \sum_{i=1}^M f(x_i), \sum_{i=1}^M f^2(x_i)$
- iii) Compute $\langle f \rangle_M = \frac{1}{M} \sum_{i=1}^M f(x_i), I = |\Omega| \cdot \langle f \rangle_M = \frac{1}{M} \sum_{i=1}^M f(x_i)$
- iv) Estimate Error: $E_M = \sqrt{\frac{\text{Var}(f)}{M}} \sim \sqrt{\frac{\text{Var}(f)}{M}} = \sqrt{\frac{1}{M-1} (\langle f^2 \rangle_M - \langle f \rangle_M^2)}$

Probability Review:

- $\text{cdf}(x) = P(X \leq x)$ (Random Var. $\rightarrow$ CDF)
- $\text{pdf}(x) = \frac{d}{dx} \text{cdf}(x) \dots$ prob. density $g(x)$
- $P(a < X < b) = \int_a^b g(x) dx, \langle f(x) \rangle = \int_{\Omega} f(x) g(x) dx$
- $V(x) = \langle (x - \langle x \rangle)^2 \rangle = \langle x^2 \rangle - \langle x \rangle^2, \langle ax \rangle = a \langle x \rangle, V(ax) = a^2 V(x)$
- $\langle x_1 + x_2 + \dots + x_n \rangle = \langle x_1 \rangle + \dots + \langle x_n \rangle, V(x_1 + \dots + x_n) = V(x_1) + \dots + V(x_n)$
- $\langle X \cdot Y \rangle = \langle X \rangle \cdot \langle Y \rangle$