# Numerical Methods for PDEs

*Methods, Lecture 2*
*Numerical Quadrature*

Notes by L. Proctor, S. De and J. White

November 20, 2013

# 1 Outline

SLIDE 1

**Gaussian Quadrature**

- Convergence properties
- Essential role of orthogonal polynomials
- Multidimensional Integrals

**Techniques for singular kernels**

- Adaptation and variable transformation
- Singular quadrature.

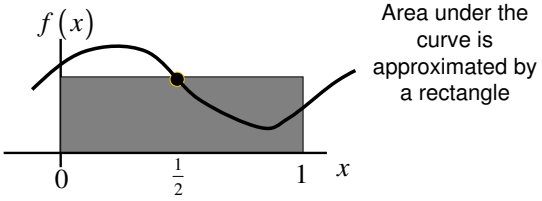
# 2 Introduction

Numerical Quadrature is employed as an approximation used to evaluate integrals. We seek an appropriate numerical procedure applied to a definite integral, $I\{f\} \equiv \int_a^b f(x)dx$, where the approximation is essentially of the form $I_n\{f\} \equiv \sum_{i=1}^{n} \alpha_i f(x_i)$. The $n$ distinct points, $x_i$ are the quadrature nodes we have chosen and the quadrature coefficients, or weights, are the $\alpha_j$ terms. In general,we would like to have the smallest possible quadrature error, $E_n\{f\} \equiv I\{f\} - I_n\{f\}$.

## 2.1 Simple Quadrature Example

SLIDE 2

$$\int_0^1 f(x)dx \simeq f\left(\frac{1}{2}\right)$$



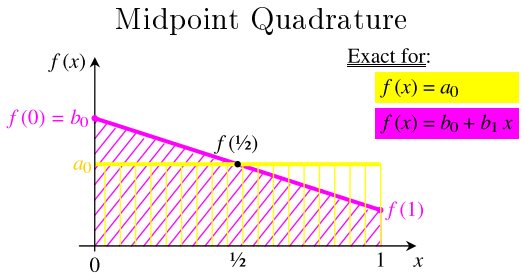

To simplify notation, consider the more generic problem of developing a good numerical technique for evaluating the integral of a function $f(x)$ on the domain $[0, 1]$. We assume that the integrand is a "smooth" function, though we will examine this assumption later. First we have developed a naive approach for obtaining a good approximation of the integral, one we call a **simple quadrature scheme**.

The simplest approach is to replace the integral with the the product of the interval (in this case one) and the integrand evaluated at a point inside the interval. If the selected evaluation point is the center of the interval, $x = 0.5$, we call the scheme **midpoint quadrature**.

A midpoint quadrature scheme replaces the area under the curve $f(x)$ by a

rectangle whose height is the function $f(x)$ evaluated at the midpoint $x = 0.5$. The scheme is exact when $f(x)$ is a constant. However, what is less obvious is that the scheme is exact when $f(x)$ is a line (an affine funcion of $x$) as well. The most obvious way of seeing this is by realizing that when $f(x)$ is a straight line, the area under it is a trapezoid. This trapezoid has exactly the same area as the rectangle which this scheme uses to approximate the integral (can you see why?).

SLIDE 3

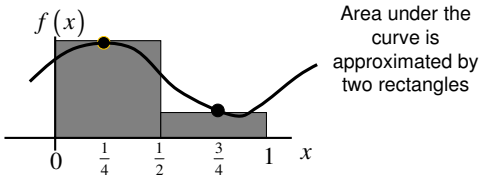



▷ ***Exercise 1*** Suppose endpoint quadrature (in which the area under the curve is replaced by a rectangle whose height is $f(x)$ evaluated at $x = 0$ or $x = 1$) is used instead of midpoint quadrature. For what class of functions is endpoint quadrature exact? ■

## 2.2 Improving the Accuracy

SLIDE 4

$$\int_0^1 f(x)dx \simeq \frac{1}{2}f\left(\frac{1}{4}\right) + \frac{1}{2}f\left(\frac{3}{4}\right)$$




One way of improving the midpoint quadrature scheme is to divide the interval $[0, 1]$ into subintervals $[0, 0.5]$ and $[0.5, 1]$, then write the integral

$$\int_0^1 f(x)dx = \int_0^{0.5} f(x)dx + \int_{0.5}^1 f(x)dx,$$

and finally apply a midpoint rule to the integral in each subinterval. We obtain the scheme shown in the slide. The factor $\frac{1}{2}$ appearing in front of $f(\frac{1}{2})$ and $f(\frac{3}{4})$ are just the domain lengths.

▷ ***Exercise 1*** Can you come up with an expression for the error in this case? How much does the accuracy improve? ■

Dividing the interval into two reduces the error, now consider using $n$ subintervals and repeating the midpoint quadrature rule on each subinterval. We obtain the scheme

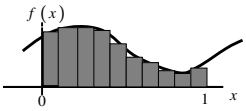
$$\int_0^1 f(x)dx = \sum_{i=1}^{n} \underbrace{\frac{1}{n}}_{\substack{\text{subinterval} \\ \text{length}}} f(x_{c_i})$$

where the centroid of the $i^{th}$ subinterval is $x_{c_i} = \frac{1}{2}(\frac{i-1}{n} + \frac{i}{n}) = \frac{i-\frac{1}{2}}{n}$. There is no doubt that the accuracy improves, but the key question is by how much? How does this error decrease with the number of subintervals used? And finally, are there clever ways of obtaining better accuracy with less effort?

## 2.3 General $n$-point formula

SLIDE 5

$$\int_0^1 f(x)dx \simeq \sum_{i=1}^{n} \underbrace{w_i}_{\substack{\text{weight} \\ \frac{1}{n}}} \underbrace{f(x_i)}_{\substack{\text{test point} \\ \frac{i-\frac{1}{2}}{n}}}$$

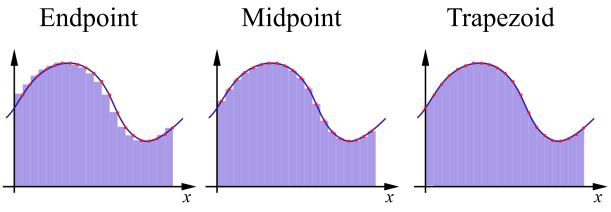



Key questions about the method:
**How fast do the errors decay with n?**
**Are there better methods?**

## 2.4 Different Geometric Approximations

SLIDE 6

Which geometry is the most accurate?



# 3 General Quadrature Formula

## 3.1 General 1-D Formula

SLIDE 7

$$\int_0^1 f(x)dx \simeq \sum_{i=1}^{n} \underbrace{w_i}_{\text{Weight}} \quad \underbrace{f(x_i)}_{\substack{\text{Evaluation} \\ \text{Point}}}$$

Free to pick the **evaluation points**.
Free to pick the **weights** for each point.

**An $n$-point formula has $2n$ degrees of freedom!**

After all the hard work we did dividing the domain into subintervals, we realize that we cannot even integrate a quadratic function exactly on the domain. There must be something that we can do to improve this scheme. We go back and look at the general form of the quadrature approximation scheme. All we are doing is approximating an integral by a weighted sum of function evaluations. So far we have been choosing these weights as the subinterval lengths. We have also been choosing the evaluation points as the center of the interval, in the midpoint quadrature scheme. The weights are just some normalizing factors which we have taken to be the fraction of the domain over which we are evaluating. The equality of areas of trapezoids and rectangles that we previously discussed gives us the extra polynomial accuracy of being able to obtain the area under a straight line exactly. So, what would happen if we were to choose both the integration points and the weights intelligently? For an $n$-point formula we have $n$ weights and $n$ evaluation points to choose. That gives us $2n$ degrees of freedom. Hence we must be able to exactly integrate a polynomial of degree at most $(2n-1)$. This idea gives rise to the **Gaussian quadrature** scheme.

## 3.2 Evaluation Points & Weights Selection

SLIDE 8

Can make the result exact if $f(x)$ is a polynomial

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_l x^l = p_l(x)$$

Select $x_i$'s and $w_i$'s such that

$$\int_0^1 p_l(x)dx = \sum_{i=1}^{n} w_i p_l(x_i)$$

for ANY polynomial up to (and including) $l^{th}$ order
**With $2n$ degrees of freedom,** $l = 2n-1$

Let $p_l(x)$ denote a polynomial of degree $l$ in the variable $x$ ($a_l \neq 0$). We want to select the weights and integration points such that the formula

$$\int_0^1 f(x)dx = \int_0^1 p_l(x)dx = \sum_{i=1}^{n} w_i p_l(x_i)$$

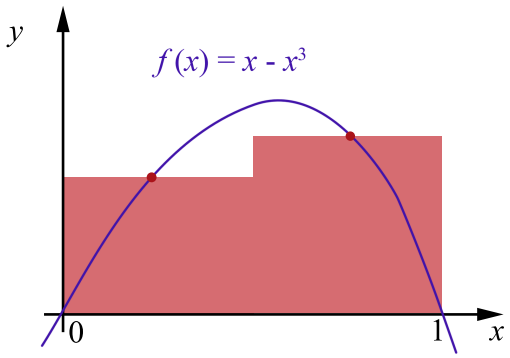
is exact for all polynomials of degree up to (and including) $l$. Obviously, with $2n$ degrees of freedom, the best we can do is $l = 2n - 1$.

---

***Note 1*** **Example: Third Order Polynomial**

---

As an example, consider integratin the function $f(x) = x - x^3$. from $x = 0$ to $x = 1$. The exact solution is, $I\{f\} = \int_0^1 (x - x^3)dx = \frac{1}{4}$. It is stated above that since we have a polynomial of degree three ($l = 3$), then we will be able to find an exact solution using two point quadrature ($n = 2$). But, one must note, that finding the exact solution is not simply a matter of applying midpoint quadrature, as we have done previously. The solution using midpoint quadrature is 0.2813, not very close to 0.25. Using Gaussian Quadrature, the method of which will be studied in further detail later on, will provide the exact solution.




---



Assuming the weights, $w_i$, remain bounded, and the derivatives of $f(x)$ are bounded on $[0, 1]$,

$$\left| \int_0^1 f(x)dx - \sum_{i=1}^{n} w_i f(x_i) \right| \leq \frac{K}{(2n)!}$$

Gaussian quadrature converges **very** quickly!!

# 4 Error Analysis

## 4.1 Taylor Series Expansion

To derive the error of the midpoint quadrature scheme analytically, consider the interval $[0, h], h > 0$ and Taylor expand $f(x)$ about the center of this interval, $\bar{x} = \frac{h}{2}$,

$$f(x) = f(\bar{x}) + \Delta(x)\frac{df(\bar{x})}{dx} + \frac{\Delta(x)^2}{2!}\frac{d^2f(\xi)}{dx^2} \quad for\ some\ \xi \in [0, h],$$

where $\Delta(x) = x - \bar{x}$. The last term in the expansion is the Taylor series remainder. Integrating this expansion over the interval $[0, h]$

$$\int_0^h f(x)dx = hf(\bar{x}) + \frac{h^3}{24}\frac{d^2f(\xi)}{dx^2}.$$

Hence the error in the midpoint quadrature approximation is

$$E = \int_0^h f(x)dx - hf(\bar{x}) = \frac{h^3}{24}\frac{d^2f(\xi)}{dx^2}.$$

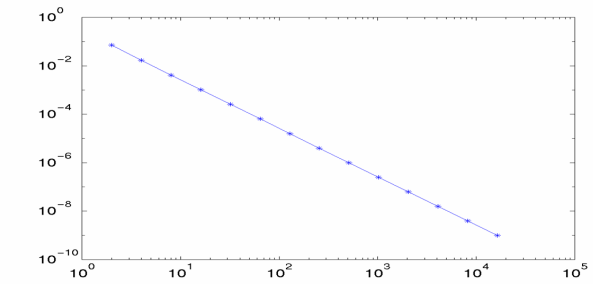
A function that is first-order polynomial in $x$ would have zero as the second derivative, and therefore the above expression tells us that the error of midpoint quadrature for such functions is identically zero. In addition, the above expression tells us that the error scales as the cube of the domain length.

▷ ***Exercise 2*** If the midpoint quadrature scheme uses rectangular geometry to approximate the area under the curve, then why is the first derivative needed in the Taylor Series expansion? How does one represent the expansion of the trapezoidal approximation? ■

## 4.2 Example - Error vs. $n$

SLIDE 10

$$\int_0^1 sin(x)dx \simeq \sum_{i=1}^{n} \frac{1}{n} sin\left(\frac{i - \frac{1}{2}}{n}\right)$$



***Note 2***

Above is the example of integrating $f(x) = sin(x)$ on the domain $[0, 1]$. We obtain progressively better answers to the integration by increasing the number of subintervals $n$. The error in evaluating the integral is plotted as a function of the number of subintervals ($n$) above. The error appears to be going down as $\mathcal{O}\left(\frac{1}{n^2}\right)$.
From what we have just seen, the error inside the $i^{th}$ subinterval (of length $h = \frac{1}{n}$) is $\frac{h^3}{24}\frac{d^2 f(\xi_i)}{dx^2}$ for some $\xi_i \in [\frac{i-1}{n}, \frac{i}{n}]$. Hence, for the entire interval $[0, 1]$ we can sum these errors and obtain the error, $E_n$ for an approximation using $n$ subintervals as

$$E_n = \frac{nh^3}{24}\underbrace{\left(\frac{1}{n}\sum_{i=1}^{n}\frac{d^2 f(\xi_i)}{dx^2}\right)}_{M}$$

It is easy to see that if $f(x)$ is a continuous function, $M$ (being the mean) must be bounded by the maximum and the minimum of $f(x)$ on the interval $[0, 1]$ and hence, there must exist some $\xi \in [0, 1]$ such that $M = d^2 f(\xi)/dx^2$. Hence we obtain the estimate

$$E_n = \frac{nh^3}{24}\frac{d^2 f(\xi)}{dx^2} = \frac{1}{24n^2}\frac{d^2 f(\xi)}{dx^2}$$

since $h = 1/n$. This error estimate tells us that the scheme is again exact for constants and linear functions on the domain (no higher order polynomials!) and, for a smooth function, the error decays **algebraically.**

## 4.3 The Exactness Criteria

SLIDE 11

Consider the Taylor series for $f(x)$ expanded about $x = 0$

$$f(x) = f(0) + \frac{\partial f(0)}{\partial x}x + \cdots + \frac{1}{l!}\frac{\partial^l f(0)}{\partial x^l}x^l + R_{l+1}$$

$R_{l+1}$ is the **remainder**

$$R_{l+1} = \frac{1}{(l+1)!}\frac{\partial^{l+1} f(\tilde{x})}{\partial x^{l+1}}x^{l+1}$$

where $\tilde{x} \in [0, x]$

***Note 3***

Of all functions, why are we interested in integrating polynomials? The reason comes from the structure of Taylor's series expansion. The Taylor expansion of a function in a local neighborhood of a point (here this point is chosen as 0 without loss of generality) is nothing but a power series expansion. The higher

the order of polynomials that our scheme can integrate means a higher order of the remainder term in the expansion. The integral of the remainder over the domain is precisely the error in numerical integration.

---

SLIDE 12

The exactness condition requires

$$\int_0^1 p_l(x)dx = \int_0^1 (a_0 + a_1x + a_2x^2 + \cdots + a_lx^l)dx = \sum_{i=1}^{n} w_i p_l(x_i)$$

for any set of $l+1$ coefficients $a_0, a_1, \ldots, a_l$

**Equivalently**

$$\int_0^1 a_0dx + \int_0^1 a_1xdx + \int_0^1 a_2x^2dx + \cdots + \int_0^1 a_lx^ldx = \sum_{i=1}^{n} w_i p_l(x_i)$$

This slide needs little clarification. Our exactness criterion is

$$\int_0^1 p_l(x)dx = \int_0^1 (a_0 + a_1x + a_2x^2 + \cdots + a_lx^l)dx = \sum_{i=1}^{n} w_i p_l(x_i)$$

which is the same as

$$a_0\int_0^1 dx + a_1\int_0^1 xdx + \cdots + a_l\int_0^1 x^ldx = a_0\sum_{i=1}^{n} w_i + a_1\sum_{i=1}^{n} w_ix_i + \cdots + a_l\sum_{i=1}^{n} w_i{x_i}^l$$

For this to be an identity for the $(l+1)$ arbitrary coefficients $a_i$, we must have the $(l+1)$ conditions

$$\sum_{i=1}^{n} w_i{x_i}^j = \int_0^1 x^j dx \quad for\ j = 0, 1, \ldots, l$$

SLIDE 13

Using the Taylor series results, the exactness criteria, and the innate linearity of the quadrature scheme

$$\int_0^1 f(x)dx - \sum_{i=1}^{n} w_i f(x_i) = \underbrace{\frac{1}{(l+1)!}\frac{\partial^{l+1} f(\tilde{x})}{\partial x^{l+1}}\left(\int_0^1 x^{l+1}dx - \sum_{i=1}^{n} w_i x_i^{l+1}\right)}_{Remainder}$$

SLIDE 14

Exactness condition will be satisfied if and only if

$$\int_0^1 dx = \sum_{i=1}^{n} w_i \cdot 1$$
$$\int_0^1 x dx = \sum_{i=1}^{n} w_i \cdot x_i$$
$$\vdots$$
$$\int_0^1 x^l dx = \sum_{i=1}^{n} w_i \cdot {x_i}^l$$

SLIDE 15

Reorganizing exactness equations

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1^l & x_2^l & \cdots & x_n^l \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} - \begin{bmatrix} 1 \\ \vdots \\ \vdots \\ \int_0^1 x^l dx \end{bmatrix} = 0$$

**Nonlinear**, since $x_i$'s and $w_i$'s are unknowns

What is a practical way of computing the evaluation points and weights? The system of equations is not easy to solve since $x_i$'s and $w_i$'s are unknowns.

# 5 Computing the Points & Weights

## 5.1 Newton's Method

SLIDE 16

Could use **Newton's Method**

$$F(y) = 0 \Rightarrow J_F\left(y^k\right)\left(y^{k+1} - y^k\right) = -F\left(y^k\right)$$

The nonlinear function for Newton is then

$$F\begin{pmatrix} w \\ x \end{pmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1^l & x_2^l & \cdots & x_n^l \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} - \begin{bmatrix} 1 \\ \vdots \\ \vdots \\ \int_0^1 x^l dx \end{bmatrix} = 0$$

---

***Note 4***

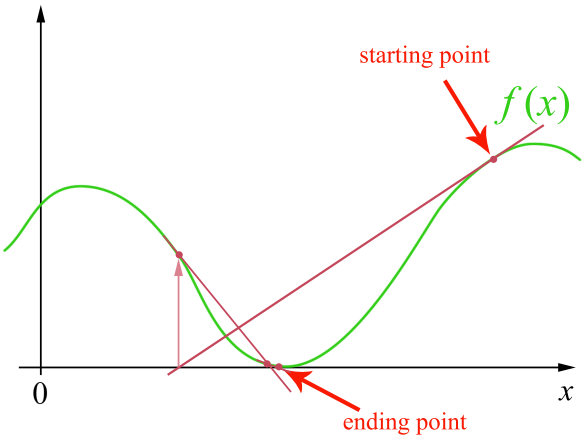
---

Newton's method is an iterative technique for finding a value $y$ such that $F(y) = 0$. The method is based on linearizing the problem about a guess at $y$, and then

updating the value of $y$ by solving the linearized problem. In particular, the iterate $y^{k+1}$ is determined from $y^k$ by solving the linear system of equations

$$F(y^k) + J_F(y^k)\left(y^{k+1} - y^k\right) = 0$$

where $J_F(y^k)$ is the Jacobian (multidimensional derivative) of the nonlinear function $F(y)$. The iteration is continued until the updated $y$ is sufficiently close to the exact solution, a criterion that can be difficult to verify. Newton's method does not always converge, a phenomenon that is more likely when $J_F(y)$ is nearly singular. For more about Newton's method, see the 6.336/16.910/2.096 course notes (available under open courseware).




SLIDE 17

**Newton Method Jacobian** reveals the problem

$$J_F\begin{pmatrix} w \\ x \end{pmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ x_1 & x_2 & \cdots & x_n & w_1 & w_2 & \cdots & w_n \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^l & x_2^l & \cdots & x_n^l & l w_1 x_1^{l-1} & \cdots & \cdots & l w_1 x_n^{l-1} \end{bmatrix}$$

2n

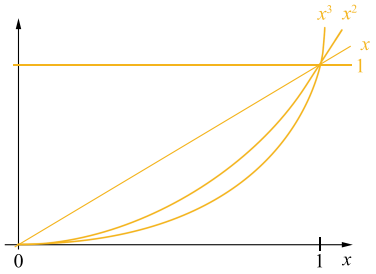
Columns become linearly dependent for high order

Looking at the Jacobian of the problem, we realize that the first $n$ columns become increasingly linearly dependent for large $l$. This is bound to happen since we are looking into the space $span\{1, x, \ldots, x^l\}$. This basis always becomes ill-conditioned with increasing $l$. The solution is to obtain a polynomial basis that is "normalized" in some sense so that it is properly conditioned.

## 5.2 Orthogonal Polynomials

### 5.2.1 Introduction

SLIDE 18




SLIDE 19

Exactness criteria will be satisfied if and only if

$$\left.\begin{aligned} \int_0^1 c_0(x)\,dx &= \sum_{i=1}^{n} w_i c_0(x_i) \\ \int_0^1 c_1(x)\,dx &= \sum_{i=1}^{n} w_i c_1(x_i) \\ &\vdots \\ \int_0^1 c_l(x)\,dx &= \sum_{i=1}^{n} w_i c_l(x_i) \end{aligned}\right\} \quad \begin{array}{c} \text{BUT} \\ \text{Each } c_i \text{ polynomial must} \\ \text{Contain an } x^j \text{ term} \\ \text{Be linearly independent} \end{array}$$

The only difference from the previous set of criteria is that these polynomials have better properties than the ones we chose before.

### 5.2.2 Orthogonality

SLIDE 20

For the normalized integral, two polynomials are said to be **orthogonal** if

$$\int_0^1 c_i(x)c_j(x)dx = 0 \qquad for\ j \neq i$$

The above integral is often referred to as an inner product and ascribed the notation

$$(c_i, c_j) = \int_0^1 c_i(x)c_j(x)dx$$

The connection between polynomial inner products and vector inner products can be seen by sampling.

### 5.2.3 Exactness Criteria

SLIDE 21

Consider rewriting the exactness criteria

$$\underbrace{\begin{array}{l}\int_0^1 c_0(x)\,dx = \sum_{i=1}^n w_i c_0(x_i) \\ \vdots \\ \int_0^1 c_{n-1}(x)\,dx = \sum_{i=1}^n w_i c_{n-1}(x_i)\end{array}}_{\text{Low order terms}} \quad \underbrace{\begin{array}{l}\int_0^1 c_n(x)\,dx = \sum_{i=1}^n w_i c_n(x_i) \\ \vdots \\ \int_0^1 c_{2n-1}(x)\,dx = \sum_{i=1}^n w_i c_{2n-1}(x_i)\end{array}}_{\text{High Order Terms}}$$

Recall that $l = 2n-1$
where $l$ = degree of polynomial & $n$ = number of coefficients

Call the first $(n-1)$ conditions the "lower order terms" and the last $n$ conditions the "higher order terms."

### 5.2.4 Higher Order Terms Contain Lower Order Terms

SLIDE 22

Write the higher order terms differently

$$\begin{array}{lcl}\int_0^1 c_n(x)dx = \sum_{i=1}^n w_i c_n(x_i) & \Rightarrow & \int_0^1 c_n(x)c_0(x)dx = \sum_{i=1}^n w_i c_n(x_i)c_0(x_i) \\ & \vdots & \\ \int_0^1 c_{2n-1}(x)dx = \sum_{i=1}^n w_i c_{2n-1}(x_i) & \Rightarrow & \int_0^1 c_n(x)c_{n-1}(x)dx = \sum_{i=1}^n w_i c_n(x_i)c_{n-1}(x_i)\end{array}$$

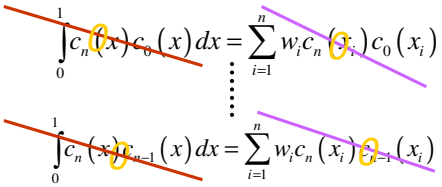
The products $c_n(x)c_j(x)$ are linearly independent.

In this slide we express the "higher order terms" as conditions involving "lower order terms."

### 5.2.5 Using Orthogonality and Roots

SLIDE 23

Use orthogonal polynomials



Pick the $x_i$'s to be $n$ roots of $c_n(x)$
The higher order constraints are exactly satisfied!

This elegant step relies on polynomial orthogonality. If we choose the polynomial $c_n(x)$ such that it is orthogonal to all polynomials of inferior degree (i.e. $c_0(x), c_1(x), \ldots, c_{n-1}(x)$) and the $x_i$'s are roots of this polynomial, then the higher order $n$ conditions are automatically satisfied. Note that for this derivation we used polynomials which are orthogonal on the interval $[0, 1]$. Such polynomials are shifted and scaled versions of the classical Legendre polynomials, which are orthogonal on the interval $[-1, 1]$.

### 5.2.6 Satisfying Lower Order Constraints

SLIDE 24

An abbreviated exactness equation

$$\begin{bmatrix} c_0(x_1) & \cdots & c_0(x_n) \\ \vdots & \ddots & \vdots \\ c_{n-1}(x_1) & \cdots & c_{n-1}(x_n) \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} \int_0^1 c_0(x)dx \\ \vdots \\ \int_0^1 c_{n-1}(x)dx \end{bmatrix}$$

Now **linear**, $x_i$'s are known
Rows are sampled orthogonal polynomials.

By using the roots of $c_n(x)$ for the $x_i$'s, the higher order constraints are automatically satisfied. Since the $x_i$'s are now known, only the weights are still unknown. The lower $n$ constraints can be used to determine the weights, generating a linear system.

## 6 Gaussian Quadrature Algorithm

SLIDE 25

1. Construct $n+1$ orthogonal polynomials

$$\int_0^1 c_i(x)c_j(x)dx = 0 \quad for\ j \neq i$$

2. Compute $n$ **roots**, $x_i$, $i = 1, \ldots, n$ of the $n^{th}$ order orthogonal polynomial such that $c_n(x_i) = 0$

3. Solve a linear system for the **weights** $w_i$

4. Approximate the integral as a sum

$$\int_0^1 f(x)dx = \sum_{i=1}^{n} w_i f(x_i)$$

## 6.1 Example

***Note 5*** **Example: Third Order Polynomial**

Recall the example of the third-order polynomial, $f(x) = x - x^3$. We would like to determine $w_1$, $w_2$, $x_1$, and $x_2$ so that the integration formula,

$$I_2\{f\} = w_1 f(x_1) + c_2 f(x_2)$$

gives the exact result.
Generalize this problem to any third-order polynomial of the form

$$f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3.$$

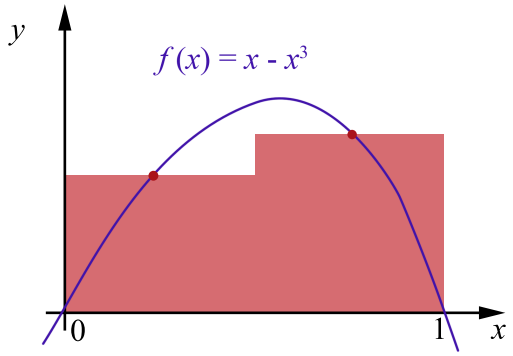
Given the exactness condition, the integral can be written:

$$\int_0^1 (a_0+a_1x+a_2x^2+a_3x^3)dx = w_1(a_0+a_1x_1+a_2x_1^2+a_3x_1^3)+w_2(a_0+a_1x_2+a_2x_2^2+a_3x_2^3)$$

Gather the like terms:

$a_0$ : $w_1 + w_2 = \int_0^1 dx = 1$
$a_1$ : $w_1x_1 + w_2x_2 = \int_0^1 xdx = \frac{1}{2}$
$a_2$ : $w_1x_1^2 + w_2x_2^2 = \int_0^1 x^2dx = \frac{1}{3}$
$a_3$ : $w_1x_1^3 + w_2x_2^3 = \int_0^1 x^3dx = \frac{1}{4}$

There are four equations above and four unknowns, so the system can be easily solved to give, $w_1 = \frac{1}{2}$ $w_2 = \frac{1}{2}$ $x_1 = \frac{3-\sqrt{3}}{6}$ $x_2 = \frac{3+\sqrt{3}}{6}$.
Plugging in these values, for the equation above, gives the exact solution of 0.25.



## 6.2 Summary

This slide summarizes the technique of finding weights and integration points for Gauss quadrature.

### 6.2.1 Accuracy Result

SLIDE 26

$$\int_0^1 f(x)dx \simeq \sum_{i=1}^{n} w_i f(x_i)$$

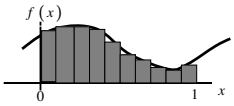
Key properties of the method

- An $n$-point Gauss quadrature rule is **exact** for polynomials of order $2n-1$
- Error is proportional to $\frac{1}{(2n)!}$ (like $\frac{1}{n^n}$)

### 6.2.2 Simple vs. Gauss Quadrature

SLIDE 27

$$\int_0^1 f(x)dx \simeq \sum_{i=1}^{n} \frac{1}{n} f\left(\frac{i-\frac{1}{2}}{n}\right)$$

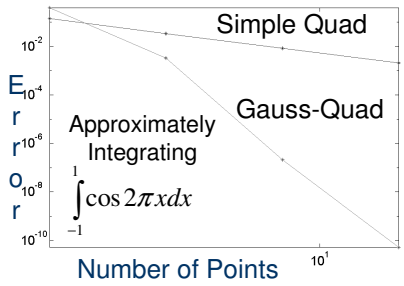



Key property of the method

- Error is proportional to $\frac{1}{n^2}$

▷ ***Exercise 3*** Do you see that the simple quadrature scheme is a special case of Gauss quadrature? ■

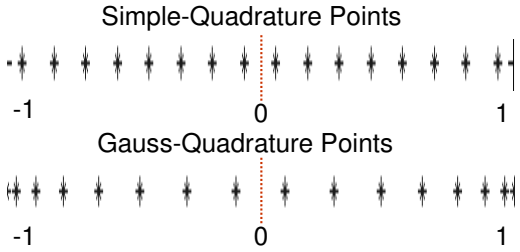
SLIDE 28




Notice that for a smooth function $f(x) = cos(2\pi x)$, which is infinitely differentiable, Gauss quadrature far outperforms the simple quadrature scheme

### 6.2.3 Evaluation Point Placement

SLIDE 29




Notice the clustering at interval ends

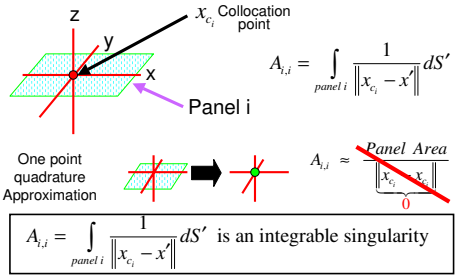
In the Gauss quadrature scheme the evaluation points are roots of Legendre polynomials which are clustered at the ends of the interval.

# 7 The Singular Kernel Problem

## 7.1 Calculating the "Self-Term"

Now lets go back to our problem of solving Laplace's equation on a 3D domain using boundary integral representation. We realize that we now have some sophisticated tools to handle integrals of functions that are smooth. But what about the integral on the panel where the centroid $x_{c_i}$ is located? The Green's function blows up at the centroid. However, the function is integrable because the integrand blows up at a rate that is slower than the rate at which the surface measure goes to zero in the vicinity of the singularity. So we know that the integral exists and is finite, but is Gauss quadrature capable of performing well in the presence of this singularity?

SLIDE 30

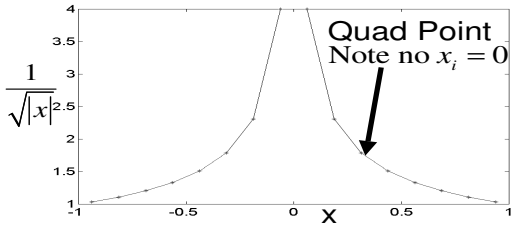



## 7.2 Symmetric 1-D Example

In 1D we look at a function $f(x) = \frac{1}{\sqrt{|x|}}$ which is integrable on $[-1, 1]$ but has a singularity at $x = 0$.

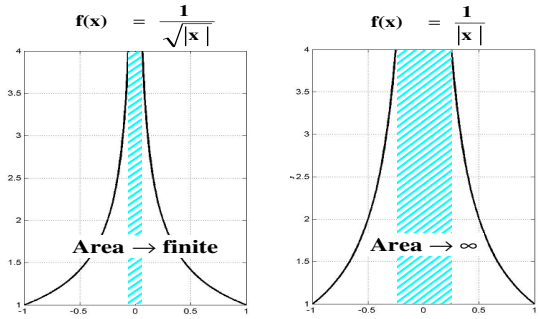
SLIDE 31

$$\int_{-1}^{1} \frac{1}{\sqrt{|x|}} dx \simeq \sum_{i=1}^{n} w_i \frac{1}{\sqrt{|x_i|}}$$



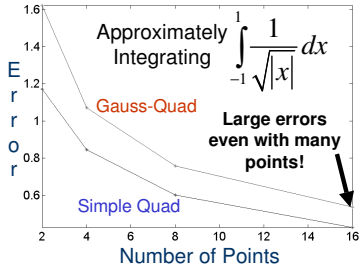

### 7.2.1 Integrable and Nonintegrable Singularities

SLIDE 32




### 7.2.2 Comparing Quadrature Schemes

SLIDE 33




We observe that Gauss quadrature is not very good at integrating this function. The convergence is rather poor. As a matter of fact, it is more inaccurate than the simple quadrature scheme. In the next few slides we present several techniques of handling integrals with singularities (which are integrable, of course)

- Subinterval (adaptive) quadrature
- Change of variables of integration
- Singular (Gaussian) quadrature

## 7.3 Improved Techniques

### 7.3.1 Subinterval (Adaptive) Quadrature

SLIDE 34

Subdivide the integration interval

$$\int_{-1}^{1} \frac{1}{\sqrt{|x|}} dx = \int_{-1}^{-0.1} \frac{1}{\sqrt{|x|}} dx + \int_{-0.1}^{0} \frac{1}{\sqrt{|x|}} dx + \int_{0}^{0.1} \frac{1}{\sqrt{|x|}} dx + \int_{0.1}^{1} \frac{1}{\sqrt{|x|}} dx$$

Use Gauss quadrature in each subinterval

Polynomials fit subintervals better

Expensive if many subintervals used.

### 7.3.2 Change of Variables - for Simple Cases

SLIDE 35

Change variables to eliminate singularity

$$y^2 = x \qquad \Rightarrow \qquad 2ydy = dx$$

$$\int_{-1}^{1} \frac{1}{\sqrt{|x|}} dx = 2 \int_{0}^{1} \frac{1}{\sqrt{|y^2|}} 2ydy = 2 \int_{0}^{1} 2dy$$

Apply Gauss quadrature on desingularized integrand.

### 7.3.3 Singular Quadrature - Complicated Cases

**Basic Concept**

SLIDE 36

Integrand has known singularity $s(x)$

$\int_{-1}^{1} f(x)s(x)dx$ where $f(x)$ is smooth

Develop a quadrature formula exact for

$\int_{-1}^{1} p_l(x)s(x)dx$ where $p_l(x)$ is polynomial of order $l$

Calculate weights like Gauss quadrature

It is possible to generate Gaussian quadrature schemes of the form

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

for functions which have a known singularity $s(x) > 0$. The quadrature formula needs to be exact when $f(x)$ is a polynomial of order at most $l$. Not surprisingly, it turns out that the integration points are the $n$ roots of a polynomial $c_n(x)$ of degree $n = (l+1)/2$ which is orthogonal to all polynomials of inferior degree with respect to the weight $s(x)$, i.e.

$$\int_{-1}^{1} s(x)c_n(x)c_j(x) = 0 \qquad for\ j = 0, 1, \ldots, (n-1).$$

An example is the singular integral

$$I = \int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}}$$

Here, $s(x) = 1/\sqrt{1-x^2}$ and the corresponding orthogonal polynomials turn out to be the Chebyshev polynomials. The integration points are given in closed form by

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

and the corresponding weights are $w_i = \pi/n$.

---

**Singular Quadrature Weights**

SLIDE 37

$$\begin{bmatrix} c_0(x_1) & \cdots & c_0(x_n) \\ \vdots & \ddots & \vdots \\ c_{n-1}(x_1) & \cdots & c_{n-1}(x_n) \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} \int_{-1}^{1} c_0(x)s(x)dx \\ \vdots \\ \int_{-1}^{1} c_{n-1}(x)s(x)dx \end{bmatrix}$$

Need (analytic) formulas for integrals of $c(x)s(x)$

The lower order constraints can be used to compute the integration weights.

# 8 Summary

SLIDE 38

**Easy technique for computing integrals**
Piecewise constant approach
**Gaussian quadrature**

Faster convergence
Essential role of orthogonal polynomials
**Techniques for singular kernels**
Adaptation and Variable Transformation
Singular quadrature
**What about multiple dimensions?**