Numerical Methods I
Orthogonal Polynomials

Aleksandar Donev
Courant Institute, NYU

donev@courant.nyu.edu

\(^1\)Course G63.2010.001 / G22.2420-001, Fall 2010

Nov. 4th and 11th, 2010
1. Function spaces

2. Orthogonal Polynomials on $[-1, 1]$

3. Spectral Approximation

4. Fourier Orthogonal Trigonometric Polynomials

5. Conclusions
The final presentations will take place on the following three dates (tentative list!):

1. Thursday Dec. 16th 5-7pm (there will be no class on Legislative Day, Tuesday Dec. 14th)
2. Tuesday Dec. 21st 5-7pm
3. Thursday Dec. 23rd 4-7pm (note the earlier start time!)

There will be no homework this week: Start thinking about the project until next week’s homework.
Lagrange basis on 10 nodes

A few Lagrange basis functions for 10 nodes
Runge's phenomenon $f(x) = (1 + x^2)^{-1}$
Function spaces are the equivalent of finite vector spaces for functions (space of polynomial functions $\mathcal{P}$, space of smoothly twice-differentiable functions $C^2$, etc.).

Consider a one-dimensional interval $I = [a, b]$. Standard norms for functions similar to the usual vector norms:

- **Maximum norm**: $\|f(x)\|_\infty = \max_{x \in I} |f(x)|$
- **$L_1$ norm**: $\|f(x)\|_1 = \int_a^b |f(x)| \, dx$
- **Euclidian $L_2$ norm**: $\|f(x)\|_2 = \left[ \int_a^b |f(x)|^2 \, dx \right]^{1/2}$
- **Weighted norm**: $\|f(x)\|_w = \left[ \int_a^b |f(x)|^2 w(x) \, dx \right]^{1/2}$

An **inner or scalar product** (equivalent of dot product for vectors):

$$(f, g) = \int_a^b f(x)g^*(x) \, dx$$
Formally, function spaces are infinite-dimensional linear spaces. Numerically we always truncate and use a finite basis.

Consider a set of $m+1$ nodes $x_i \in X \subset I$, $i = 0, \ldots, m$, and define:

$$
\|f(x)\|_X^2 = \left[ \sum_{i=0}^{m} |f(x_i)|^2 \right]^{1/2},
$$

which is equivalent to thinking of the function as being the vector

$$
f_X = y = \{f(x_0), f(x_1), \cdots, f(x_m)\}.
$$

Finite representations lead to semi-norms, but this is not that important.

A discrete dot product can be just the vector product:

$$
(f, g)^X = f_X \cdot g_X = \sum_{i=0}^{m} f(x_i)g^*(x_i)
$$
Function Space Basis

- Think of a function as a vector of coefficients in terms of a set of $n$ basis functions:

$$\{\phi_0(x), \phi_1(x), \ldots, \phi_n(x)\},$$

for example, the monomial basis $\phi_k(x) = x^k$ for polynomials.

- A finite-dimensional approximation to a given function $f(x)$:

$$\tilde{f}(x) = \sum_{i=1}^{n} c_i \phi_i(x)$$

- **Least-squares approximation** for $m > n$ (usually $m \gg n$):

$$c^* = \arg \min_{c} \left\| f(x) - \tilde{f}(x) \right\|_2,$$

which gives the **orthogonal projection** of $f(x)$ onto the finite-dimensional basis.
Discrete case: Think of fitting a straight line or quadratic through experimental data points.

The function becomes the vector \( y = f(x) \), and the approximation is

\[
y_i = \sum_{j=1}^{n} c_j \phi_j(x_i) \quad \Rightarrow \quad y = \Phi c,
\]

where

\[
\Phi_{ij} = \phi_j(x_i).
\]

This means that finding the approximation consists of solving an overdetermined linear system

\[
\Phi c = y
\]

Note that for \( m = n \) this is equivalent to interpolation. MATLAB’s \textit{polyfit} works for \( m \geq n \).
Recall that one way to solve this is via the normal equations:

\[
(\Phi^*\Phi) \mathbf{c}^* = \Phi^*\mathbf{y}
\]

A basis set is an **orthonormal basis** if

\[
(\phi_i, \phi_j) = \sum_{k=0}^{m} \phi_i(x_k)\phi_j(x_k) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}
\]

\[
\Phi^*\Phi = I \quad \text{(unitary or orthogonal matrix)} \quad \Rightarrow
\]

\[
\mathbf{c}^* = \Phi^*\mathbf{y} \quad \Rightarrow \quad c_i = \phi_i^X \cdot \mathbf{f}_X = \sum_{k=0}^{m} f(x_k)\phi_i(x_k)
\]
Consider a function on the interval $I = [a, b]$. Any finite interval can be transformed to $I = [-1, 1]$ by a simple transformation.

Using a **weight function** $w(x)$, define a **function dot product** as:

$$(f, g) = \int_a^b w(x) [f(x)g(x)] \, dx$$

For different choices of the weight $w(x)$, one can explicitly construct **basis of orthogonal polynomials** where $\phi_k(x)$ is a polynomial of degree $k$ (**triangular basis**):

$$(\phi_i, \phi_j) = \int_a^b w(x) [\phi_i(x)\phi_j(x)] \, dx = \delta_{ij} \|\phi_i\|^2.$$
Legendre Polynomials

- For equal weighting \( w(x) = 1 \), the resulting triangular family of polynomials are called **Legendre polynomials**:

\[
\begin{align*}
\phi_0(x) &= 1 \\
\phi_1(x) &= x \\
\phi_2(x) &= \frac{1}{2}(3x^2 - 1) \\
\phi_3(x) &= \frac{1}{2}(5x^3 - 3x) \\
\phi_{k+1}(x) &= \frac{2k + 1}{k + 1}x\phi_k(x) - \frac{k}{k + 1}\phi_{k-1}(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} [(x^2 - 1)^n]
\end{align*}
\]

These are orthogonal on \( I = [-1, 1] \):

\[
\int_{-1}^{1} \phi_i(x)\phi_j(x)dx = \delta_{ij} \cdot \frac{2}{2i + 1}.
\]
Interpolation using Orthogonal Polynomials

- Let’s look at the **interpolating polynomial** $\phi(x)$ of a function $f(x)$ on a set of $m + 1$ nodes $\{x_0, \ldots, x_m\} \in I$, expressed in an orthogonal basis:

$$\phi(x) = \sum_{i=0}^{m} a_i \phi_i(x)$$

- Due to orthogonality, taking a dot product with $\phi_j$ (**weak formulation**):

$$\langle \phi, \phi_j \rangle = \sum_{i=0}^{m} a_i \langle \phi_i, \phi_j \rangle = \sum_{i=0}^{m} a_i \delta_{ij} \|\phi_i\|^2 = a_j \|\phi_j\|^2$$

- This is **equivalent to normal equations** if we use the right dot product:

$$(\Phi^* \Phi)_{ij} = (\phi_i, \phi_j) = \delta_{ij} \|\phi_i\|^2 \text{ and } \Phi^* y = (\phi, \phi_j)$$
Gauss Integration

\[ a_j \| \phi_j \|^2 = (\phi, \phi_j) \implies a_j = \left( \| \phi_j \|^2 \right)^{-1} (\phi, \phi_j) \]

- Question: Can we easily compute

\[ a_j \| \phi_j \|^2 = (\phi, \phi_j) = \int_a^b w(x) [\phi(x)\phi_j(x)] \, dx = \int_a^b w(x)p_{2m}(x) \, dx \]

for a polynomial \( p_{2m}(x) = \phi(x)\phi_j(x) \) of degree at most 2m?

- Let’s first consider polynomials of degree at most \( m \)

\[ \int_a^b w(x)p_m(x) \, dx = \]
Now consider the **Lagrange basis** \( \{ \varphi_0(x), \varphi_1(x), \ldots, \varphi_m(x) \} \), where you recall that
\[
\varphi_i(x_j) = \delta_{ij}.
\]

Any polynomial \( p_m(x) \) of degree at most \( m \) can be expressed in the Lagrange basis:
\[
p_m(x) = \sum_{i=0}^{m} p_m(x_i) \varphi_i(x),
\]

\[
\int_a^b w(x)p_m(x)\,dx = \sum_{i=0}^{m} p_m(x_i) \left[ \int_a^b w(x)\varphi_i(x)\,dx \right] = \sum_{i=0}^{m} w_i p_m(x_i),
\]

where the **Gauss weights** \( w \) are given by
\[
w_i = \int_a^b w(x)\varphi_i(x)\,dx.
\]
Back to Interpolation

For any polynomial $p_{2m}(x)$ there exists a polynomial quotient $q_{m-1}$ and a remainder $r_m$ such that:

$$p_{2m}(x) = \phi_{m+1}(x)q_{m-1}(x) + r_m(x)$$

$$\int_a^b w(x)p_{2m}(x)\,dx = \int_a^b [w(x)\phi_{m+1}(x)q_{m-1}(x) + w(x)r_m(x)] \,dx$$

$$= (\phi_{m+1}, q_{m-1}) + \int_a^b w(x)r_m(x)\,dx$$

But, since $\phi_{m+1}(x)$ is orthogonal to any polynomial of degree at most $m$, $(\phi_{m+1}, q_{m-1}) = 0$ and we thus get:

$$\int_a^b w(x)p_{2m}(x)\,dx = \sum_{i=0}^m w_i r_m(x_i)$$
Finally, if we choose the nodes to be zeros of $\phi_{m+1}(x)$, then

$$r_m(x_i) = p_{2m}(x_i) - \phi_{m+1}(x_i)q_{m-1}(x_i) = p_{2m}(x_i)$$

$$\int_a^b w(x)p_{2m}(x)dx = \sum_{i=0}^{m} w_i p_{2m}(x_i)$$

and thus we have found a way to quickly project any polynomial onto the basis of orthogonal polynomials:

$$(p_m, \phi_j) = \sum_{i=0}^{m} w_i p_m(x_i) \phi_j(x_i)$$

$$(\phi, \phi_j) = \sum_{i=0}^{m} w_i \phi(x_i) \phi_j(x_i) = \sum_{i=0}^{m} w_i f(x_i) \phi_j(x_i)$$
Orthogonal Polynomials on $[-1, 1]$

**Gauss-Legendre polynomials**

- For any weighting function the polynomial $\phi_k(x)$ has $k$ simple zeros all of which are in $(-1, 1)$, called the (order $k$) **Gauss nodes**, $\phi_{m+1}(x_i) = 0$.

- The interpolating polynomial $\phi(x_i) = f(x_i)$ on the Gauss nodes is the **Gauss-Legendre interpolant** $\phi_{GL}(x)$.

- The orthogonality relation can be expressed as a sum instead of integral:
  \[
  (\phi_i, \phi_j) = \sum_{i=0}^{m} w_i \phi_i(x_i) \phi_j(x_i) = \delta_{ij} \| \phi_i \|^2
  \]

- We can thus define a new weighted **discrete dot product**
  \[
  f \cdot g = \sum_{i=0}^{m} w_i f_i g_i
  \]
Discrete Orthogonality of Polynomials

- The orthogonal polynomial basis is **discretely-orthogonal** in the new dot product,

\[ \phi_i \cdot \phi_j = (\phi_i, \phi_j) = \delta_{ij} (\phi_i \cdot \phi_i) \]

- This means that the matrix in the normal equations is diagonal:

\[ \Phi^* \Phi = \text{Diag} \left\{ \|\phi_0\|^2, \ldots, \|\phi_m\|^2 \right\} \quad \Rightarrow \quad a_i = \frac{\mathbf{f} \cdot \phi_i}{\phi_i \cdot \phi_i}. \]

- The Gauss-Legendre interpolant is thus easy to compute:

\[ \phi_{GL}(x) = \sum_{i=0}^{m} \frac{\mathbf{f} \cdot \phi_i}{\phi_i \cdot \phi_i} \phi_i(x). \]
Consider the Hilbert space $L^2_w$ of square-integrable functions on $[-1, 1]$: 

$$\forall f \in L^2_w : \quad (f, f) = \|f\|^2 = \int_{-1}^{1} w(x) [f(x)]^2 \, dx < \infty.$$ 

Legendre polynomials form a complete orthogonal basis for $L^2_w$: 

$$\forall f \in L^2_w : \quad f(x) = \sum_{i=0}^{\infty} f_i \phi_i(x)$$ 

$$f_i = \frac{(f, \phi_i)}{(\phi_i, \phi_i)}.$$ 

The least-squares approximation of $f$ is a spectral approximation and is obtained by simply truncating the infinite series: 

$$\phi_{sp}(x) = \sum_{i=0}^{m} f_i \phi_i(x).$$
Spectral approximation

Continuous (spectral approximation): $\phi_{sp}(x) = \sum_{i=0}^{m} \frac{(f, \phi_i)}{(\phi_i, \phi_i)} \phi_i(x)$.

Discrete (interpolating polynomial): $\phi_{GL}(x) = \sum_{i=0}^{m} \frac{f \cdot \phi_i}{\phi_i \cdot \phi_i} \phi_i(x)$.

If we approximate the function dot-products with the discrete weighted products

$$(f, \phi_i) \approx \sum_{j=0}^{m} w_j f(x_j) \phi_i(x_j) = f \cdot \phi_i,$$

we see that the Gauss-Legendre interpolant is a \textbf{discrete spectral approximation}:

$$\phi_{GL}(x) \approx \phi_{sp}(x).$$
Discrete spectral approximation

- Using a spectral representation has many advantages for function approximation: **stability**, **rapid convergence**, easy to **add more basis functions**.
- The convergence, for sufficiently smooth (nice) functions, is **more rapid than any power law**

\[
\| f(x) - \phi_{GL}(x) \| \leq \frac{C}{N^d} \left( \sum_{k=0}^{d} \| f^{(k)} \|^2 \right)^{1/2},
\]

where the multiplier is related to the **Sobolev norm** of \( f(x) \).
- For \( f(x) \in C^1 \), the convergence is also **pointwise** with similar accuracy \( (N^{d-1/2} \) in the denominator).
- This so-called **spectral accuracy** (limited by smoothness only) cannot be achieved by piecewise, i.e., local, approximations (limited by order of local approximation).
a=2;
f = @(x) \cos(2*\exp(a*x));

x_fine=linspace(-1,1,100);
y_fine=f(x_fine);

% Equi-spaced nodes:
n=10;
x=linspace(-1,1,n);
y=f(x);
c=polyfit(x,y,n);
y_interp=polyval(c,x_fine);

% Gauss nodes:
[x,w]=GLNodeWt(n); % See webpage for code
y=f(x);
c=polyfit(x,y,n);
y_interp=polyval(c,x_fine);
Spectral Approximation

Gauss-Legendre Interpolation

Function and approximations for n=10

Actual
Equi-spaced nodes
Standard approx
Gauss nodes
Spectral approx

Error of interpolants/approximants for n=10

Standard approx
Spectral approx
Spectral Approximation

Global polynomial interpolation error

Error for equispaced nodes for n=8,16,32,...,128

Error for Gauss nodes for n=8,16,32,...,128

A. Donev (Courant Institute)
Local polynomial interpolation error

Error for linear interpolant for \( n = 8, 16, 32, \ldots, 256 \)

Error for cubic spline for \( n = 8, 16, 32, \ldots, 256 \)
Consider now interpolating / approximating **periodic functions** defined on the interval $I = [0, 2\pi]$:

$$\forall x \quad f(x + 2\pi) = f(x),$$

as appear in practice when analyzing signals (e.g., sound/image processing).

Also consider only the space of complex-valued **square-integrable functions** $L^2_{2\pi}$,

$$\forall f \in L^2_{2\pi} : \quad (f, f) = \|f\|^2 = \int_0^{2\pi} |f(x)|^2 \, dx < \infty.$$

Polynomial functions are not periodic and thus basis sets based on orthogonal polynomials are not appropriate.

Instead, consider sines and cosines as a basis function, combined together into **complex exponential functions**

$$\phi_k(x) = e^{ikx} = \cos(kx) + i \sin(kx), \quad k = 0, \pm 1, \pm 2, \ldots$$
Fourier Basis

\[ \phi_k(x) = e^{ikx}, \quad k = 0, \pm 1, \pm 2, \ldots \]

- It is easy to see that these are orthogonal with respect to the continuous dot product

\[
(\phi_j, \phi_k) = \int_{x=0}^{2\pi} \phi_j(x)\phi_k^*(x)\,dx = \int_{0}^{2\pi} \exp [i(j - k)x] \,dx = 2\pi \delta_{ij}
\]

- The complex exponentials can be shown to form a complete trigonometric polynomial basis for the space \( L_{2\pi}^2 \), i.e.,

\[
\forall f \in L_{2\pi}^2 : \quad f(x) = \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ikx},
\]

where the Fourier coefficients can be computed for any frequency or wavenumber \( k \) using:

\[
\hat{f}_k = \frac{(f, \phi_k)}{2\pi} = \frac{1}{2\pi} \int_{0}^{2\pi} f(x) e^{-ikx} \,dx.
\]
Fourier Orthogonal Trigonometric Polynomials

Discrete Fourier Basis

- For a general interval \([0, X]\) the **discrete frequencies** are
  \[
  k = \frac{2\pi}{X} \kappa \quad \kappa = 0, \pm 1, \pm 2, \ldots
  \]

- For non-periodic functions one can take the limit \(X \to \infty\) in which case we get **continuous frequencies**.

- Now consider a **discrete Fourier basis** that only includes the first \(N\) basis functions, i.e.,
  \[
  \begin{cases}
  k = -(N - 1)/2, \ldots, 0, \ldots, (N - 1)/2 & \text{if } N \text{ is odd} \\
  k = -N/2, \ldots, 0, \ldots, N/2 - 1 & \text{if } N \text{ is even},
  \end{cases}
  \]
  and for simplicity we focus on \(N\) odd.

- The least-squares **spectral approximation** for this basis is:
  \[
  f(x) \approx \phi(x) = \sum_{k=-(N-1)/2}^{(N-1)/2} \hat{f}_k e^{ikx}.
  \]
Now also discretize the functions on a set of $N$ equi-spaced nodes

$$x_j = jh \text{ where } h = \frac{2\pi}{N}$$

where $j = N$ is the same node as $j = 0$ due to periodicity so we only consider $N$ instead of $N + 1$ nodes.

We also have the **discrete dot product** between two discrete functions (vectors) $f_j = f(x_j)$:

$$f \cdot g = h \sum_{j=0}^{N-1} f_j g_j^*$$

The discrete Fourier basis is **discretely orthogonal**

$$\phi_k \cdot \phi_{k'} = 2\pi \delta_{k,k'}$$
The case $k = k'$ is trivial, so focus on

$$\phi_k \cdot \phi_{k'} = 0 \text{ for } k \neq k'$$

$$\sum_j \exp(ikx_j) \exp(-ik'x_j) = \sum_j \exp[i(\Delta k)x_j] = \sum_{j=0}^{N-1} [\exp(ih(\Delta k))]^j$$

where $\Delta k = k - k'$. This is a geometric series sum:

$$\phi_k \cdot \phi_{k'} = \frac{1 - z^N}{1 - z} = 0 \text{ if } k \neq k'$$

since $z = \exp(ih(\Delta k)) \neq 1$ and $z^N = \exp(ihN(\Delta k)) = \exp(2\pi i(\Delta k)) = 1$. 
The **Fourier interpolating polynomial** is thus easy to construct

\[
\phi_N(x) = \sum_{k=-\frac{N-1}{2}}^{\frac{N-1}{2}} \hat{f}_k^{(N)} e^{i k x}
\]

where the **discrete Fourier coefficients** are given by

\[
\hat{f}_k^{(N)} = \frac{f \cdot \phi_k}{2\pi} = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) \exp(-i k x_j)
\]

Simplifying the notation and recalling \(x_j = jh\), we define the the **Discrete Fourier Transform (DFT)**:

\[
\hat{f}_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j \exp\left(-\frac{2\pi i j k}{N}\right)
\]
Fourier Spectral Approximation

Forward \( f \rightarrow \hat{f} : \quad \hat{f}_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j \exp \left( - \frac{2\pi i j k}{N} \right) \)

Inverse \( \hat{f} \rightarrow f : \quad f(x) \approx \phi(x) = \sum_{k=-(N-1)/2}^{(N-1)/2} \hat{f}_k e^{ikx} \)

- There is a very fast algorithm for performing the forward and backward DFTs called the Fast Fourier Transform (FFT), which we will discuss next time.
- The Fourier interpolating polynomial \( \phi(x) \) has spectral accuracy, i.e., exponential in the number of nodes \( N \)
  \[
  \|f(x) - \phi(x)\| \sim e^{-N}
  \]
  for sufficiently smooth functions (sufficiently rapid decay of the Fourier coefficients with \( k \), e.g., \( \hat{f}_k \sim e^{-|k|} \)).
The set of discrete Fourier coefficients $\hat{f}$ is called the **discrete spectrum**, and in particular,

$$S_k = \left| \hat{f}_k \right|^2 = \hat{f}_k \hat{f}_k^*, $$

is the **power spectrum** which measures the frequency content of a signal.

If $f$ is real, then $\hat{f}$ satisfies the **conjugacy property**

$$\hat{f}^{-k} = \hat{f}_k^*, $$

so that half of the spectrum is redundant and $\hat{f}_0$ is real.

For an even number of points $N$ the largest frequency $k = -N/2$ does not have a conjugate partner.
In MATLAB

- The forward transform is performed by the function $\hat{f} = \text{fft}(f)$ and the inverse by $f = \text{fft}(\hat{f})$. Note that $\text{ifft}(\text{fft}(f)) = f$ and $f$ and $\hat{f}$ may be complex.

- In MATLAB, and other software, the frequencies are not ordered in the “normal” way $-(N-1)/2$ to $(N-1)/2$, but rather, the nonnegative frequencies come first, then the positive ones, so the “funny” ordering is

$$0, 1, \ldots, (N-1)/2, -\frac{N-1}{2}, -\frac{N-1}{2} + 1, \ldots, -1.$$  

This is because such ordering (shift) makes the forward and inverse transforms symmetric.

- The function $\text{fftshift}$ can be used to order the frequencies in the normal way, and $\text{ifftshift}$ does the reverse:

$$\hat{f} = \text{fftshift}(\text{fft}(f)) \text{ (normal ordering)}.$$
Fourier Orthogonal Trigonometric Polynomials

FFT-based noise filtering (1)

Fs = 1000; % Sampling frequency
dt = 1/Fs; % Sampling interval
L = 1000; % Length of signal
t = (0:L-1)*dt; % Time vector
T=L*dt; % Total time interval

% Sum of a 50 Hz sinusoid and a 120 Hz sinusoid
x = 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
y = x + 2*randn(size(t)); % Sinusoids plus noise

figure(1); clf; plot(t(1:100),y(1:100), 'b--'); hold on
title('Signal Corrupted with Zero-Mean Random Noise')
xlabel('time')
if (0)
N=(L/2)*2; % Even N
y_h hat = fft(y(1:N));
% Frequencies ordered in a funny way:
f_funny = 2*pi/T* [0:N/2−1, −N/2:−1];
% Normal ordering:
f_normal = 2*pi/T* [−N/2 : N/2−1];
else
N=(L/2)*2−1; % Odd N
y_h hat = fft(y(1:N));
% Frequencies ordered in a funny way:
f_funny = 2*pi/T* [0:(N−1)/2, −(N−1)/2:−1];
% Normal ordering:
f_normal = 2*pi/T* [−(N−1)/2 : (N−1)/2];
end
FFT-based noise filtering (3)

```matlab
figure(2); clf; plot(f_funny, abs(y_hat), 'ro'); hold on;

y_hat = fftshift(y_hat);
figure(2); plot(f_normal, abs(y_hat), 'b-');

title('Single-Sided Amplitude Spectrum of y(t)')
xlabel('Frequency (Hz)')
ylabel('Power')

y_hat(abs(y_hat) < 250) = 0; % Filter out noise
y_filtered = ifft(ifftshift(y_hat));
figure(1); plot(t(1:100), y_filtered(1:100), 'r-')
```
FFT results
Once a function dot product is defined, one can construct orthogonal basis for the space of functions of finite 2–norm.

For functions on the interval \([-1, 1]\), triangular families of orthogonal polynomials \(\phi_i(x)\) provide such a basis, e.g., Legendre or Chebyshev polynomials.

If one discretizes at the Gauss nodes, i.e., the roots of the polynomial \(\phi_{m+1}(x)\), and defines a suitable discrete Gauss-weighted dot product, one obtains discretely-orthogonal basis suitable for numerical computations.

The interpolating polynomial on the Gauss nodes is closely related to the spectral approximation of a function.

Spectral convergence is faster than any power law of the number of nodes and is only limited by the global smoothness of the function, unlike piecewise polynomial approximations limited by the choice of local basis functions.

One can also consider piecewise-spectral approximations.