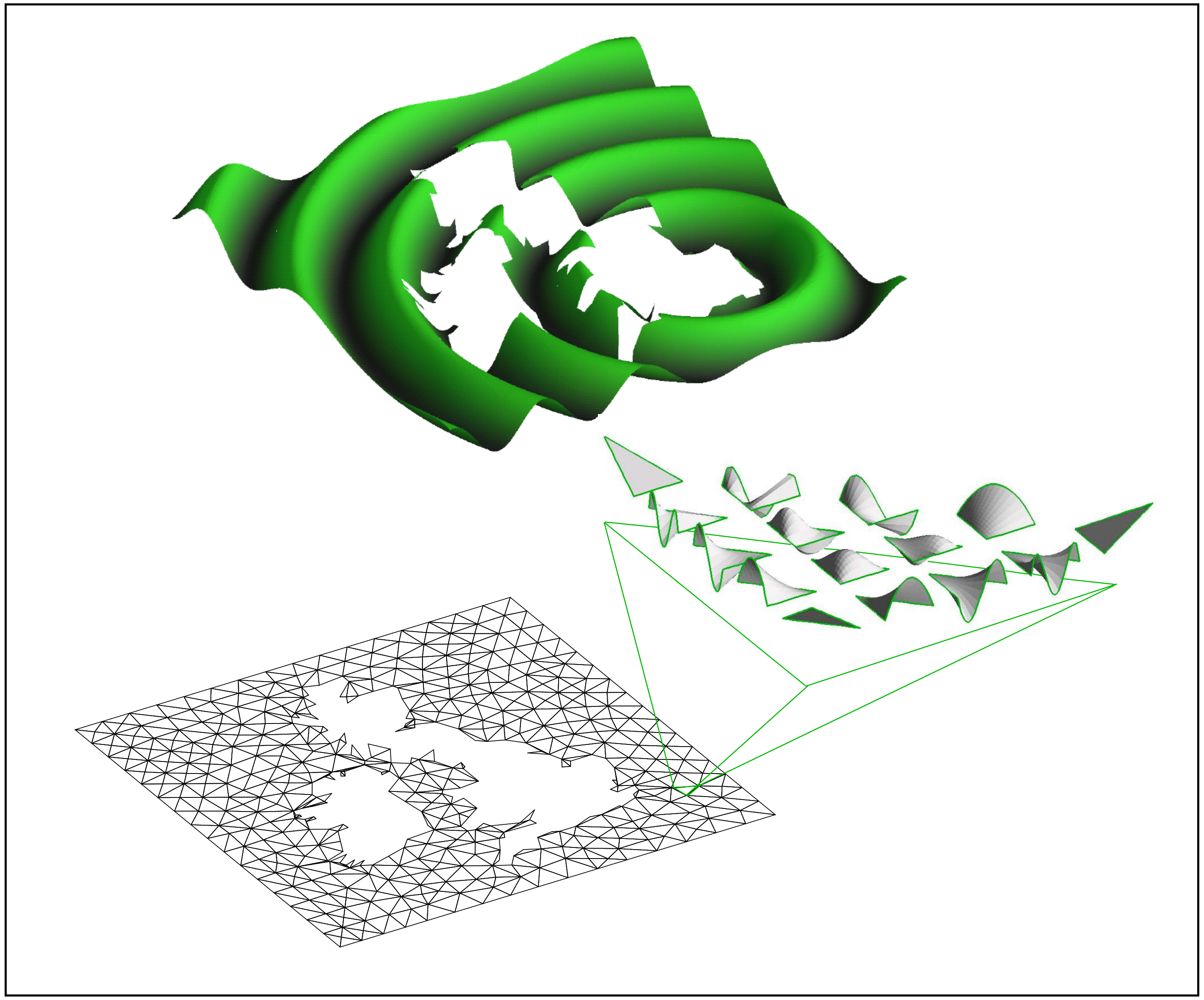
Basic Routines in LibUtilities

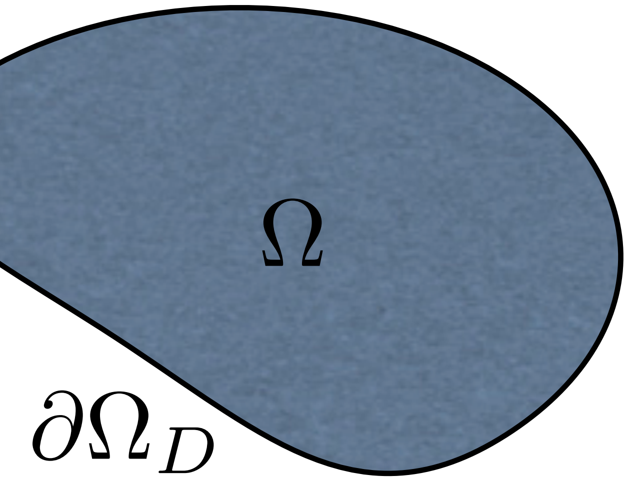


Outline

- Set up the problem (weak solution of the Helmholtz Problem)
- Polynomial Basis
- Interpolation
- Differentiation
- Integration
- Projection

Tutorial Driving Application: Helmholtz Problem

$$
\begin{aligned}
-\nabla^2 u + \lambda u &= f \qquad x \in \Omega \\
u &= g(x) \quad x \in \partial\Omega_D
\end{aligned}
$$



Find $u \in \mathcal{V}$ such that u satisfies the boundary conditions and such that for all $v \in \mathcal{V}_0$,

$$(\nabla u, \nabla v) + (u, v) = (f, v)$$

What do we need?

$$(\nabla u, \nabla v) + (u, v) = (f, v)$$

Representation Differentiation Integration

Polynomials

Numerical Differentiation

Quadrature

Polynomial Basis: Choosing a representation

Polynomials provide a powerful means of representing continuous functions

Monomial Basis
BasisType eMonomial

$$u(x) = a_0 + a_1 x + a_2 x^2 + \dots$$

Lagrange Basis
BasisType eGLL_Lagrange

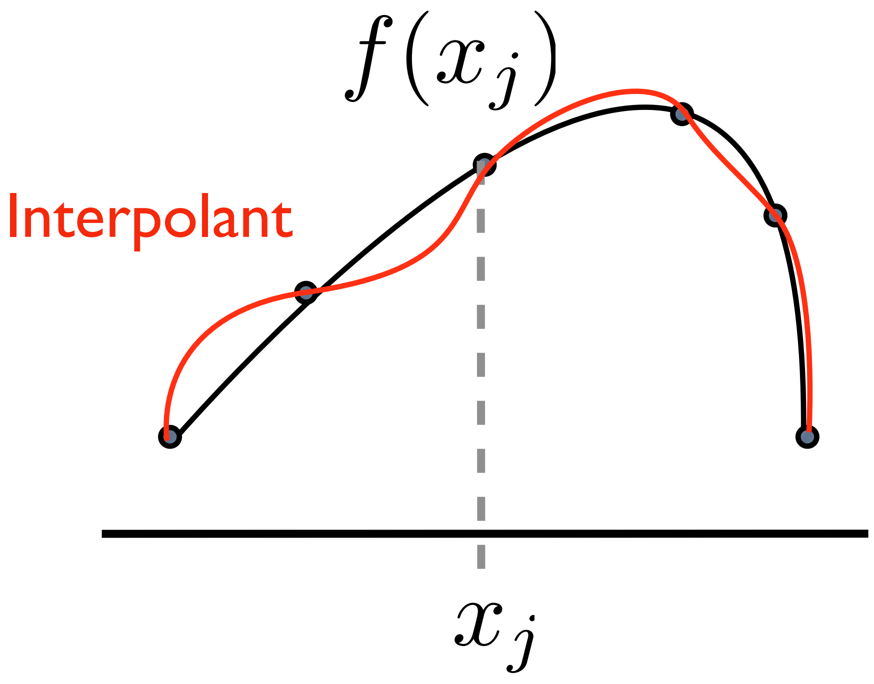
$$u(x) = u(x_0) h_0(x) + u(x_1) h_1(x) + \dots$$

Legendre Basis
BasisType eLegendre

$$u(x) = \hat{u}_0 L_0(x) + \hat{u}_1 L_1(x) + \dots$$

Interpolation

Given a collection of points $x_0, \ldots, x_N$ and function values $f_0, \ldots, f_N$ at those point, find an approximation $u(x)$ such that $u(x_j) = f(x_j)$ for all j



Lagrange Basis:

$$u(x) = \sum_{i=0}^{N} f(x_i) h_i(x)$$

where

$$h_i(x_j) = \delta_{ij}$$

Interpolation

$$\text{Given } z_0, z_1, \ldots, z_M \text{ and } \sum_{i=1}^{N} u(x_i) h_i(x)$$

$$u(z_k) = \sum_{i=0}^{N} u(x_i) h_i(z_k)$$

Values at new points | Interpolation Matrix | Values at points

$$\begin{pmatrix} u(z_0) \\ \vdots \\ u(z_M) \end{pmatrix} = \begin{pmatrix} h_0(z_0) & \ldots & h_N(z_0) \\ & \vdots & \\ h_0(z_M) & \ldots & h_N(z_M) \end{pmatrix} \begin{pmatrix} u(x_0) \\ \vdots \\ u(x_N) \end{pmatrix}$$

LibUtilities::Points.GetI()

Differentiation

Monomial Basis $$\frac{d}{dx}u(x) = a_1 + 2a_2x + \ldots$$

Lagrange Basis $$\frac{d}{dx}u(x) = u(x_0)h_0'(x) + u(x_1)h_1'(x) + \ldots$$

Legendre Basis $$\frac{d}{dx}u(x) = \hat{u}_0L_0'(x) + \hat{u}_1L_1'(x) + \ldots$$

Differentiation

$$\text{Given } z_0, z_1, \ldots, z_M \text{ and } \sum_{i=1}^{N} u(x_i) h_i(x)$$

$$u'(z_k) = \sum_{i=0}^{N} u(x_i) h_i'(z_k)$$

Derivatives at new points | Derivative Matrix | Values at points

$$\begin{pmatrix} u'(z_0) \\ \vdots \\ u'(z_M) \end{pmatrix} = \begin{pmatrix} h_0'(z_0) & \ldots & h_N'(z_0) \\ & \vdots & \\ h_0'(z_M) & \ldots & h_N'(z_M) \end{pmatrix} \begin{pmatrix} u(x_0) \\ \vdots \\ u(x_N) \end{pmatrix}$$

LibUtilities::Points.GetD()

Integration

$$\int_{-1}^{1} u(\xi)d\xi \approx \sum_{i=0}^{Q-1} w_i u(\xi_i)$$

integrating polynomials

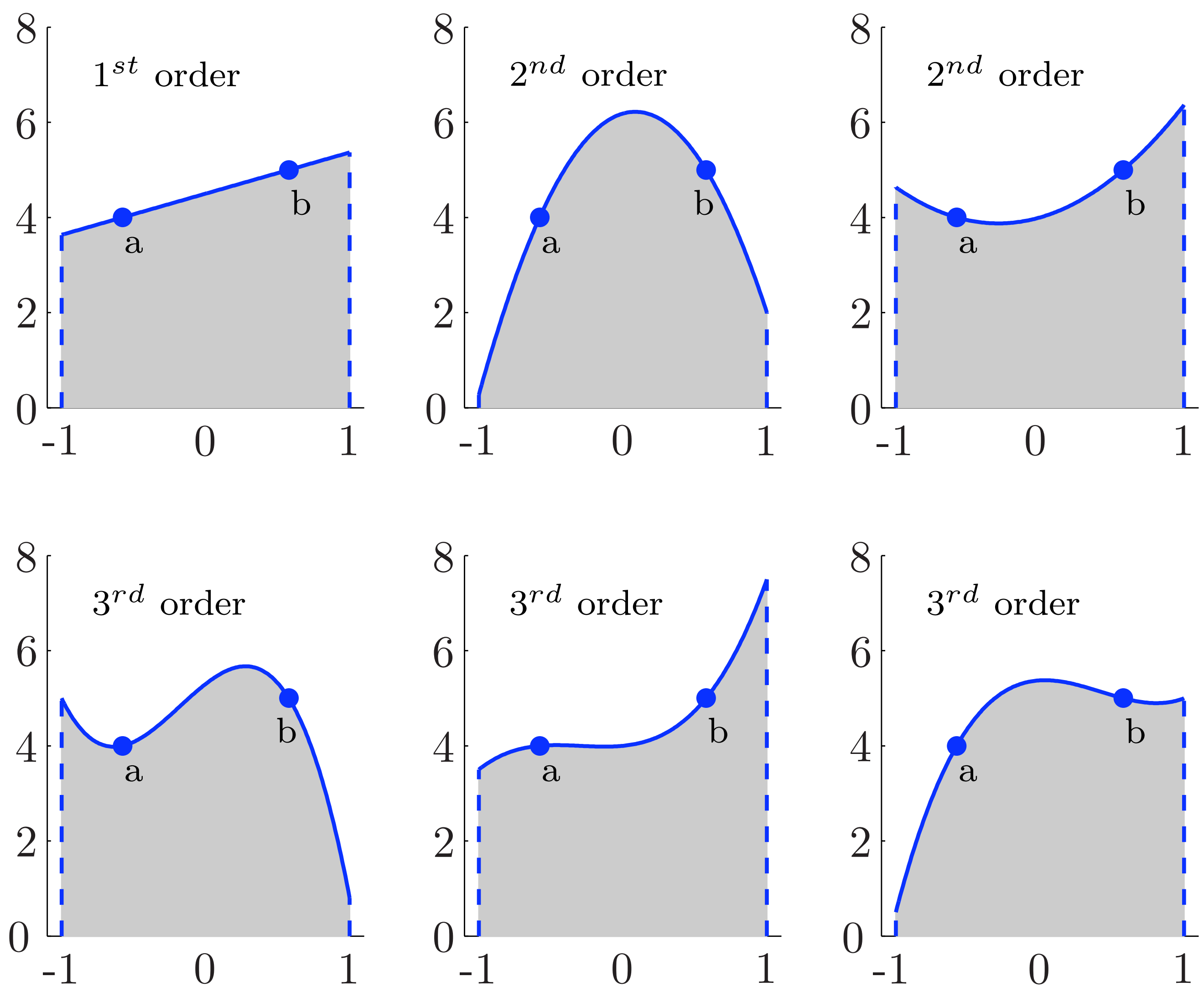
- polynomial of order P: P+1 parameters
- Gaussian quadrature:
  - Q ≥ P/2 + 0.5 (Gauss-Legendre)
  - Q ≥ P/2 + 1.0 (Gauss-Radau-Legendre)
  - Q ≥ P/2 + 1.5 (Gauss-Lobatto-Legendre)

2D: $$\int_{-1}^{1}\int_{-1}^{1} u(\xi_1,\xi_2)d\xi_1 d\xi_2 \approx \sum_{i=0}^{Q_1-1} w_i \left\{ \sum_{j=0}^{Q_2-1} w_j u(\xi_{1_i},\xi_{2_j}) \right\}$$

Integration

a remarkable property

- fix Q
- exact integration for every polynomial up to order (2Q-1)

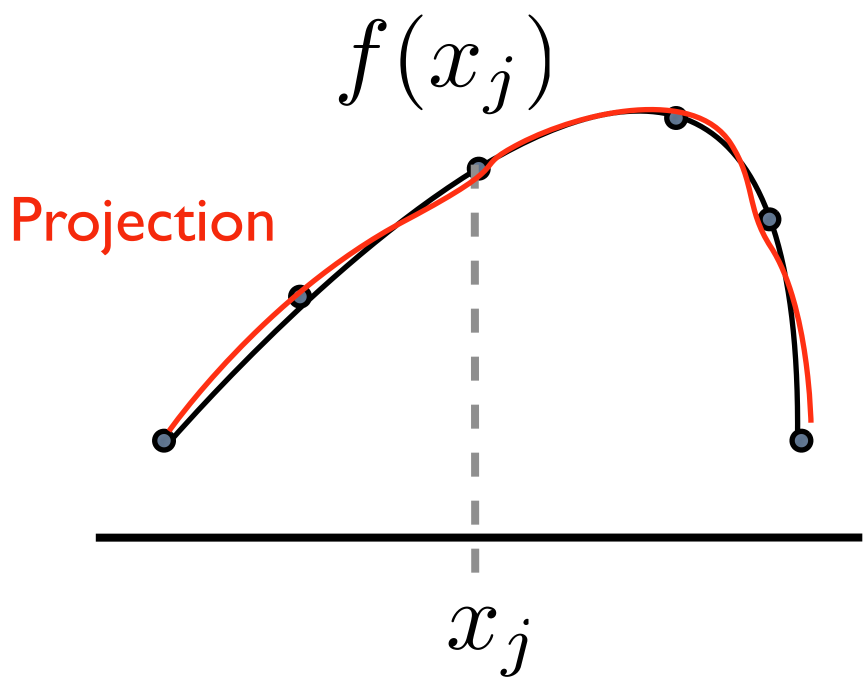


Nektar++

Projection

Given a function $f(x)$ on $[-1,1]$ find coefficients $\hat{u}_j$ such that

$$(\phi_k, R(u)) = 0 \qquad \text{for all } \phi_k,\ k = 0, \dots, N$$



where $R(u) = u(x) - f(x)$

with $u(x) = \sum_{j=0}^{N} \hat{u}_j \phi_j(x)$

Projection

$$\text{For each } k,\ (\phi_k, R(u)) = 0$$

$$\Longrightarrow\ (\phi_k, u(x) - f(x)) = 0$$

$$\Longrightarrow\ (\phi_k, \textstyle\sum_{j=0}^{N} \hat{u}_j \phi_j - f(x)) = 0$$

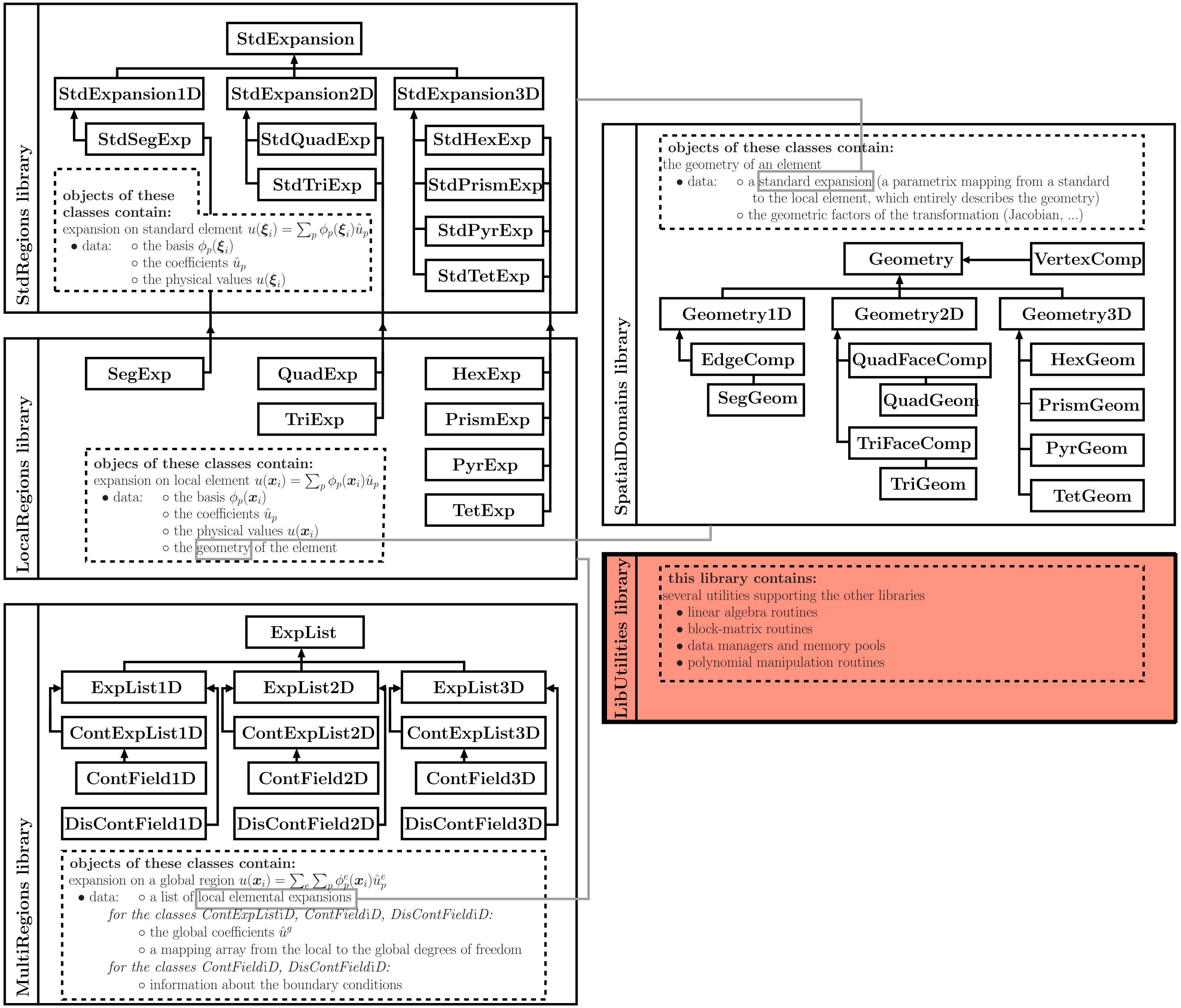
$$\Longrightarrow\ (\phi_k, \textstyle\sum_{j=0}^{N} \hat{u}_j \phi_j) = (\phi_k, f(x)) \longleftrightarrow (\phi_k, f) \approx \sum_{i=0}^{Q-1} w_i \phi_k(\xi_i) f(\xi_i)$$

Quadrature Approximation of Inner Products

$$\Longrightarrow\ \textstyle\sum_{j=0}^{N} (\phi_k, \phi_j) \hat{u}_j = (\phi_k, f)$$

$$\begin{pmatrix} (\phi_0, \phi_0) & \dots & (\phi_0, \phi_N) \\ & \vdots & \\ (\phi_N, \phi_0) & \dots & (\phi_N, \phi_N) \end{pmatrix} \begin{pmatrix} \hat{u}_0 \\ \vdots \\ \hat{u}_N \end{pmatrix} = \begin{pmatrix} (\phi_0, f) \\ \vdots \\ (\phi_N, f) \end{pmatrix}$$

Nektar++ code



Nektar++