Abstract

The spectral element method (SEM) is a high order numerical method for solving partial differential equations that inherits the accuracy of spectral methods and the geometrical flexibility of the finite element method.

These lectures provide an introduction to the SEM for graduate students in Earth science. The construction of the method is described for a model problem in 1D and 2D. The elements of its mathematical basis, including its connection to spectral methods, are outlined to explain the key choices in the construction of the SEM that lead to an accurate and efficient numerical method. Practical guidelines for programmers and users of the SEM and entry points to advanced topics and to the relevant applied math literature are provided.

These lectures were preceded by lectures on the finite difference method (FDM) and the finite element method (FEM). Several concepts from the FEM lectures reappear here, with slightly different notations. We favor redundancy in the hope that the resulting exposition of key practical aspects will contribute to attenuate the perceived complexity of programming the FEM and SEM.

These lectures were first taught in Caltech on Winter 2009.
# Contents

1 Overview 4  
1.1 Overview of the spectral element method 4  
1.2 Historical perspective 4  

2 From spectral methods to SEM, in 1D 5  
2.1 1D model problem: Helmholtz equation 5  
2.2 Galerkin approximation 6  
2.3 Spectral methods 7  
2.4 Numerical integration: Gauss quadrature 8  
2.5 Spectral method with numerical integration 9  
2.6 The spectral element method 13  

3 The SEM in higher dimensions 17  
3.1 2D model problem 17  
3.2 Spectral element mesh 18  
3.3 Basis function 19  
3.4 2D integrals 19  
3.5 Local mass matrix 20  
3.6 Local stiffness matrix 20  
3.7 Boundary matrix 22  

4 Numerical properties 23  
4.1 Dispersion analysis in 1D 24  
4.2 Optimal choice of $h$ and $P$ 28  

5 Time integration of the wave equation 31  
5.1 Second order schemes 31  
5.2 Higher-order symplectic schemes 37  

6 Miscellaneous 38  
6.1 Limitations of the SEM 38  
6.2 Advanced topics 38  

A References 39  
B Available software 39
C Homework problems

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1 Love wave modes in a horizontally layered medium</td>
<td>40</td>
</tr>
<tr>
<td>C.2 Wave propagation and numerical dispersion</td>
<td>42</td>
</tr>
</tbody>
</table>
1 Overview

1.1 Overview of the spectral element method

The SEM combines the best of two worlds: flexibility of FEM and accuracy of spectral methods.

The operational procedure is similar to FEM:

1. PDE problem formulated in weak form.
2. Domain decomposition into a mesh of quadrilateral (2D) or hexaedral (3D) elements, possibly deformed.
3. Galerkin method: approximate the weak form over a finite space of piecewise polynomial functions
4. Evaluate the integrals in the weak form with numerical quadrature

The SEM is a high-order method: the basis functions are polynomials of order typically $\geq 4$. Among several ways to construct a high-order FEM, the SEM is characterized by an optimal choice of basis functions and quadrature rule. Optimal means less expensive (fewer operations and smaller memory requirement) for a given accuracy.

Different than h-p FEM:

- Nodal basis instead of Modal basis
- Tensor-product basis. Each element is mapped to the reference element $[-1,1]^D$. In the reference element, assume a set of basis functions made of products of polynomials of each space variable, with degree $P$: $\phi(x) = p_i(x)p_j(y)p_k(z)$.
  - diagonal matrix by choice of quadrature and interpolation nodes (but equivalent to mass lumping)
  - spectral convergence (note also convergence of inter-element fluxes)

The SEM has, of course, some limitations that will be reviewed in Section 6.1.

1.2 Historical perspective

History of numerical methods for solving PDEs

- 1950s FDM
• 1960s FEM
• 1970s spectral methods (1965 FFT)
• 1990s SEM
• DGM?

History of SEM
• introduced by Patera (1984) on Chebyshev polynomials, for fluid dynamics
• generalized by Maday and Patera (1989) to Legendre polynomials
• application to earthquake dynamics: Ampuero (2002 + Villette), Festa (2004), Kaneko (2009)

2 From spectral methods to SEM, in 1D

2.1 1D model problem: Helmholtz equation

Arises from, e.g., the wave equation in frequency domain as in waveform tomography applications. Increasing efforts are currently devoted towards realizing 3D waveform tomographic inversion [e.g. Bleibinhaus et al, JGR 2007]. Waveform tomography is classically based on frequency domain solvers, which do not involve time discretization.

Strong form

Given $\lambda$ a real positive constant and $f(x)$ a function of $x \in [a, b]$, find $u(x)$ such that

$$\frac{\partial^2 u}{\partial x^2} + \lambda u + f = 0 \quad (2.1)$$
with boundary conditions
\[ u(a) = 0 \quad \text{Dirichlet} \quad (2.2) \]
\[ \frac{\partial u}{\partial x}(b) = 0 \quad \text{Neumann} \quad (2.3) \]

**Weak form**

Define Sobolev spaces
\[ L^2([a,b]) = \{ u : [a,b] \rightarrow \mathbb{R} \mid \int_a^b u(x)^2 \, dx < \infty \} \quad (2.4) \]
\[ H^1([a,b]) = \{ u \in L^2([a,b]) \mid \frac{du}{dx} \in L^2([a,b]) \} \quad (2.5) \]
\[ H^1_0([a,b]) = \{ u \in H^1([a,b]) \mid u(a) = u(b) = 0 \} \quad (2.6) \]

Find \( u \in H^1([a,b]) \) such that for all test functions \( v \in H^1_0([a,b]) \) the following is satisfied
\[ \int_a^b \frac{\partial v}{\partial x} \frac{\partial u}{\partial x} \, dx - \int_a^b \lambda v u \, dx = \int_a^b vf \, dx \quad (2.7) \]

See previous lectures on FEM for proof of equivalence between strong and weak formulation. In particular, note that the boundary terms appearing after integration-by-parts vanish, \( [v \frac{\partial u}{\partial x}]_a^b = 0 \). Equation 2.7 can be written as
\[ a(u,v) - \lambda (u,v) = (f,v) \quad (2.8) \]
where \( (\cdot, \cdot) \) denotes dot product and the first term is a bilinear form \( a(\cdot, \cdot) \).

**2.2 Galerkin approximation**

For practical implementation in a computer code, we must restrict the search to a subspace \( V^h \subset H^1_0 \) that has finite dimension. With basis functions \( \{ \phi_i \}_{i=0}^n \). Search an approximate solution in this subspace:
\[ u^h(x) = \sum_{i=0}^n u_i \phi_i(x). \quad (2.9) \]

Obtain the solution coefficients \( u_i \) by requiring
\[ a(u^h,v^h) - \lambda(u^h,v^h) = (f,v^h) \quad \forall v^h \in V^h. \quad (2.10) \]

This leads to the following algebraic system of equations:
\[ (K - \lambda M)u = f \quad (2.11) \]
with

\[ K_{ij} = a(\phi_i, \phi_j) \]  
\[ M_{ij} = (\phi_i, \phi_j) \]  
\[ f_i = (f, \phi_i) \]

Two important choices will be expanded next: the basis functions and the numerical method to evaluate the integrals.

### 2.3 Spectral methods

The basis functions \( \phi_i \) are polynomials of degree \( P \) that span the whole domain. In principle, infinite modal expansion:

\[ u(x) = \sum_{i=0}^{\infty} \tilde{u}_i \phi_i(x) \]  
with

\[ \tilde{u}_i = (u, \phi_i) \]

In practice, we need truncation:

\[ u(x) \approx \tilde{u}^h(x) = \sum_{i=0}^{P} \tilde{u}_i \phi_i(x) \]

If the basis is well chosen the coefficients \( u_i \) decay very fast, and truncation introduces a very small error for sufficiently large \( P \). Optimal choices have the spectral convergence property: for infinitely smooth functions, the truncation error decays as a function of \( P \) faster than any power law

\[ ||u - \tilde{u}^h|| = O(P^{-m}) \quad \forall m > 0 \]  

[Catch: spectral convergence does not hold for less smooth functions.]

A family of optimal choices are the eigenfunctions of singular Sturm-Liouville problems. The polynomials within this family are the so-called Jacobi polynomials. Legendre polynomials are a special case.

The Legendre polynomials are the only sequence of polynomials \( \{L_i\} \) of degree \( i \) that satisfy the following orthogonality conditions:

\[ \int_{-1}^{1} L_i(\xi)L_j(\xi) d\xi = \delta_{ij} \quad \forall i, j = 0, 1, ... \]

Cea’s lemma: The spectral convergence property also holds for the solution of the PDE problem in weak form: \( ||u - u^h|| \) is of same order as \( ||u - \tilde{u}^h|| \).
2.4 Numerical integration: Gauss quadrature

In practice the integrals involved above are evaluated by numerical integration (quadrature). Gaussian quadrature is based on polynomial expansions. The quadrature rule is defined by $P$ nodes $\xi_i$ and weights $w_i$ designed so that

$$\int_{-1}^{1} p(\xi) \, d\xi = \sum_{i=0}^{P} w_i p(\xi_i)$$

(2.20)

for all polynomials $p$ up to a certain polynomial order.

The maximum degree that can be integrated exactly by a quadrature based on $P + 1$ nodes (and $P + 1$ weights, that’s $2P + 2$ degrees of freedom) is $2P + 1$.

**Gauss-Legendre quadrature**

The optimal quadrature, i.e. one that achieves exact integration up to degree $2P + 1$, is the so-called Gauss-Legendre quadrature rule. The Gauss-Legendre quadrature takes $\{\xi_i\}_{i=0}^{P}$ to be the zeros of $L_{P+1}$ and the weights

$$w_i = \frac{2}{(1 - \xi_i^2)[L'_{P+1}(\xi_i)]^2}$$

(2.21)

have been designed to give an exact quadrature up to degree $P$. It can be shown that the quadrature is actually exact up to degree $2P + 1$. (The proof relies on the orthogonality of the Legendre polynomials.)

**Gauss-Lobatto-Legendre quadrature**

To enforce boundary conditions and inter-element continuity (as we will see later) it is convenient to include the end points among the quadrature nodes. The maximum degree for exact quadrature is now $2P - 1$, the $P + 1$ quadrature nodes are the zeros of $(1 - \xi^2)L'_P(\xi)$, the so-called Gauss-Lobatto-Legendre (GLL) points, and the weights are

$$w_i = \frac{2}{P(P + 1)[L_P(\xi_i)]^2}$$

(2.22)

The GLL nodes are not uniformly distributed, they tend to cluster near the end points $\pm 1$ (the inter-nodal distance near the end points is asymptotically $\propto 1/P^2$).
2.5 Spectral method with numerical integration

Discrete modal/nodal duality and spectral convergence

Let \( u(\xi) \) be a function of \( \xi \in [-1, 1] \). Its Legendre expansion is

\[
    u(\xi) = \sum_{m=0}^{\infty} \hat{u}_m L_m(\xi)
\]

(2.23)

where the coefficients \( u_m \) are the Legendre transform of \( u \):

\[
    \hat{u}_m = (u, L_m) = \int_{-1}^{1} u(\xi)L_m(\xi) \, d\xi
\]

(2.24)

Its approximation by truncation to the first \( P + 1 \) terms:

\[
    u(\xi) \approx \tilde{u}(\xi) = \sum_{m=0}^{P} \hat{u}_m L_m(\xi)
\]

(2.25)

has spectral convergence: for smooth \( u \) the misfit \( ||u - \tilde{u}|| \) (aka the aliasing error) decays faster than any power law as a function of \( P \).

In practice, the integral in Equation 2.24 must be approximated by quadrature. Taking the GLL quadrature of order \( P \) defines a discrete (and truncated) Legendre transform:

\[
    \tilde{u}(\xi) = \sum_{m=0}^{P} \tilde{u}_m L_m(\xi)
\]

(2.26)

with

\[
    \tilde{u}_m = (u, L_m)_P = \sum_{i=0}^{P} w_i u(\xi_i) L_m(\xi_i)
\]

(2.27)

Approximating \( \hat{u} \) by \( \tilde{u} \) introduces an error of the same order as the aliasing error between \( \hat{u} \) and \( u \). Hence, \( \tilde{u} \) is also a spectral approximation of \( u \).

The following identity arises from the particular choice of GLL interpolation and quadrature: \( \tilde{u} \) is the polynomial of degree \( P \) that interpolates \( u \) at the \( P \) GLL nodes,

\[
    \tilde{u}(\xi_i) = u(\xi_i).
\]

(2.28)

Hence, the modal representation in Equation 2.26 is equivalent to the following nodal representation:

\[
    \tilde{u}(\xi) = \sum_{i=0}^{P} u_i \ell_i(\xi)
\]

(2.29)
where the coefficients are simply the values of $u$ at the GLL nodes

$$u_i = u(\xi_i) \quad (2.30)$$

and $\{\ell_i\}_{i=0}^P$ are the Lagrange interpolation polynomials associated with the GLL nodes of order $P$.

**Definition of the discrete spectral method**

For a given order $P$, the method consists of:

1. map the domain $x \in [a, b]$ into the reference segment $\xi \in [-1, 1]$

$$x = \frac{a + b}{2} + \frac{h}{2} \xi \quad (2.31)$$

where $h = b - a$ is the size of the domain, and formulate the weak form problem in the reference domain

2. take as basis functions, $\{\phi_i\}_{i=0}^P$, the Lagrange polynomials associated with the $P + 1$ GLL nodes

$$\ell_i(\xi) = \prod_{j \neq i} \frac{\xi - \xi_j}{\xi_i - \xi_j} \quad (2.32)$$

[Recall $\ell_i$ are “discrete delta-functions” in the space of polynomials of degree $P$: $\ell_i \in P_P([-1, 1])$ and $\ell_i(\xi_j) = \delta_{ij}$.]

3. solve Equation 2.11 where Equation 2.12 to Equation 2.14 are evaluated with GLL quadrature.

The convergence of this method is spectral, even if the quadrature is of reduced order (exact for $2P - 1$ instead of $2P$). The quadrature introduces an error that is consistent with (same order as) the approximation error of the polynomial expansion. [Canuto and Quarteroni (1982) showed that the discrete norm is uniformly equivalent to the continuous norm.]
Diagonal mass matrix

A major practical advantage: the mass matrix is diagonal by construction

\[ M_{pq} = \frac{h}{2} (\ell_p, \ell_q) \]  (2.33)

\[ = h/2 \sum_{i=0}^{P} w_i \ell_p(\xi_i) \ell_q(\xi_i) \]  (2.34)

\[ = h/2 \sum_{i=0}^{P} w_i \delta_{p} \delta_{q_i} \]  (2.35)

\[ = h/2 w_p \delta_{pq} \]  (2.36)

What if we had evaluated the mass matrix with exact integration? Must be done with a different quadrature, with \( Q + 1 \) quadrature nodes \( \{\xi_i(\ell)\}_{i=0}^{Q} \) and weights \( \{w_i(\ell)\}_{i=0}^{Q} \), exact at least up to degree \( 2P \) (instead of \( 2P - 1 \) for GLL with \( P + 1 \) nodes). GLL quadrature with \( Q = P + 1 \) or GL quadrature with \( Q = P - 1 \) would work.

\[ M_{pq}^{\text{exact}} = \frac{h}{2} \sum_{i=0}^{Q} w_i(\ell) \ell_p(\xi_i) \ell_q(\xi_i) \]  (2.37)

The \( \ell \) terms here are \( \neq 0 \), because the values are requested at nodes that are not the nodes that define the \( \ell \)'s. Hence, the exact mass matrix is full: \( M_{pq}^{\text{exact}} \neq 0 \) for all \( p, q \).

The spectral mass matrix can also be obtained by lumping the exact mass matrix, a typical trick in FEM. The sum of the \( p \)-th row of the exact mass matrix is:

\[ \sum_{q=0}^{P} M_{pq}^{\text{exact}} = \sum_{q=0}^{P} (\ell_p, \ell_q) \]  (2.38)

\[ = \left( \ell_p, \sum_{q=0}^{P} \ell_q \right) \]  (2.39)

\[ = (\ell_p, 1) \]  (2.40)

\[ = w_p \delta_{pq} \]  (2.41)
Stiffness matrix

The stiffness matrix is full.

\[ K_{pq} = \frac{2}{h} (\ell'_p, \ell'_q)_P \]  

(2.42)

\[ = \frac{2}{h} \sum_{i=0}^{P} w_i \ell'_p(\xi_i) \ell'_q(\xi_i) \]  

(2.43)

Introducing the matrix of derivatives of the Lagrange polynomials \( \mathbf{H} \) such that

\[ H_{ij} = \ell'_i(\xi_j) \]  

(2.44)

and the diagonal matrix of quadrature weights \( \mathbf{W} \) such that

\[ W_{ij} = w_i \delta_{ij} \]  

(2.45)

we get the compact expression

\[ \mathbf{K} = \frac{2}{h} \mathbf{H} \mathbf{W} \mathbf{H}^t \]  

(2.46)

Conditioning of a matrix \( \mathbf{A} \) is important for the efficiency of iterative solvers of algebraic systems \( \mathbf{y} = \mathbf{Ax} \). The conditioning of the spectral stiffness matrix is much better (\( \propto P \)) than that obtained with more naive choices of basis functions (\( \propto 10^P \) for regularly spaced nodes or modal expansion, Karniadakis 2.3.3.2).

Forcing vector

\[ f_p = h/2 (f, \ell_p)_P \]  

(2.47)

\[ = h/2 \sum_{i=0}^{P} w_i f(\xi_i) \ell_p(\xi_i) \]  

(2.48)

\[ = h/2 \sum_{i=0}^{P} w_i f(\xi_i) \delta_{pi} \]  

(2.49)

\[ = h/2 w_p f(\xi_p) \]  

(2.50)

(2.51)
Non uniform material properties

Consider the 1D SH wave equation, in frequency domain, with non uniform shear modulus $\mu(x)$ and density $\rho(x)$:

$$\frac{\partial}{\partial x} \left( \mu \frac{\partial u}{\partial x} \right) + \omega^2 \rho u + f = 0$$  \hspace{1cm} (2.52)

The mass matrix is

$$M_{pq} = \frac{h}{2} w_p \rho(x_p) \delta_{pq} \hspace{1cm} (2.53)$$

For the stiffness matrix Equation 2.46 still applies if we define

$$W_{ij} = w_i \mu(x_i) \delta_{ij} \hspace{1cm} (2.54)$$

**Exercise 1: Love wave modes in a layered medium**

**2.6 The spectral element method**

Basis functions that span the whole domain are not practical for complicated geometries (e.g. seismic wave propagation in a sedimentary basin) or models with discontinuous physical properties (e.g. wave equation in layered media). The approach, borrowed from FEM, is to decompose the domain into elements, then apply a spectral method within each element. The continuity of the solution across inter-element boundaries is efficiently achieved by the choice of Lobatto nodes.

**The spectral element mesh**

The domain $[a, b]$ is partitioned into $N$ elements $\Omega_e = [X_{e-1}, X_e]$, $e = 1, ..., N$, with $X_0 = a$ and $X_N = b$.

A mapping $\chi^e$ is defined between each element $(x \in \Omega_e)$ and the reference element $(\xi \in [-1, 1])$:

$$x = \chi^e(\xi) = \frac{1 - \xi}{2} X_{e-1} + \frac{1 + \xi}{2} X_e, \quad \xi \in [-1, 1] \hspace{1cm} (2.55)$$

or, inversely,

$$\xi = (\chi^e)^{-1}(x) = 2 \frac{x - (X_{e-1} + X_e)/2}{X_e - X_{e-1}}, \quad x \in \Omega_e \hspace{1cm} (2.56)$$

Each element is provided with a GLL sub-grid. The $i$-th GLL node of the $e$-th element is located at

$$x_i^e = \chi^e(\xi_i) \hspace{1cm} (2.57)$$
The non-redundant list of these nodes (i.e. inter-element nodes counted only once) form a set of \( N \times P + 1 \) global nodes

\[
x_I = x^e_i \quad \text{with} \quad I = \mathcal{I}(i, e) = (e - 1)P + i.
\] (2.58)

The table \( \mathcal{I}(i, e) \) is the local-to-global index map table. We adopt the following shortcut notation for this association between indices:

\[
I \equiv (i, e)
\] (2.59)

**The assembly operator**

Consider a set of local quantities defined on an element-by-element basis: a set of local vectors \( \{\mathbf{a}^e\}_{e=1}^N \) of size \( P + 1 \) each, or a set of local matrices \( \{\mathbf{A}^e\}_{e=1}^N \) of size \( (P + 1) \times (P + 1) \) each.

To assemble is to add the local contributions from each element into a global array. The assembled vector

\[
\mathbf{a} = \sum_{e=1}^N \mathbf{a}^e,
\] (2.60)

of size \( NP + 1 \), by definition has the following components

\[
a_I = \sum_{I=(i,e)}^{(i,e)} \mathbf{a}^e_i
\] (2.61)

\[
= \begin{cases} 
\mathbf{a}^e_i & \text{if } I \equiv (i, e) \text{ with } i \in [1, P - 1] \\
\mathbf{a}^e_{P-1} + \mathbf{a}^e_0 & \text{if } I \equiv (P, e - 1) \equiv (0, e)
\end{cases}
\] (2.62)

*Figure here*. The following pseudo-code performs the assembly operation:

```plaintext
1 a (: ) = 0
2 loop over e from 1 to N
3 compute the local vector ae(:)
4 loop over i from 0 to P
5 k = I(i, e)
6 a(k) = a(k) + ae(i)
7 end loop over i
8 end loop over e
```

The components of an assembled matrix,

\[
\mathbf{A} = \sum_{e=1}^N \mathbf{A}^e,
\] (2.63)
of size \((NP + 1) \times (NP + 1)\), are

\[
A_{IJ} = \sum_{\substack{(i,j,e) \in I \equiv (i,e) \equiv (i,j,e) \equiv (j,e) \equiv (j,e) \\
\text{if } I \equiv (i,e) \text{ with } i \in [1, P - 1] \\
\text{or } J \equiv (j,e) \text{ with } j \in [1, P - 1] \\
\text{if } I \text{ or } J \text{ are interior nodes}}}
A^e_{ij}
\]

(2.64)

\[
= \begin{cases} 
A^e_{ij} & \text{if } I \equiv (i,e) \text{ with } i \in [1, P - 1] \\
& \text{or } J \equiv (j,e) \text{ with } j \in [1, P - 1] \\
& \text{if } I \text{ or } J \text{ are interior nodes}
A^e_{P0} + A^e_{0P} & \text{if } I = J \equiv (P,e - 1) \equiv (0,e) \\
& \text{(if } I = J \text{ and is a boundary node})
\end{cases}
\]

(2.65)

[Figure here]. The following pseudo-code performs the assembly operation:

```plaintext
1 A (:,:) = 0
2 loop over e from 1 to N
3     compute the local matrix Ae(:,:)
4     loop over j from 0 to P
5         kj = I(j,e)
6         loop over i from 0 to P
7             ki = I(i,e)
8             A(ki,kj) = A(ki,kj) + Ae(i,j)
9         end loop over i
10     end loop over j
11 end loop over e
```

Basis functions

A set of global basis functions is defined by gluing together the spectral basis functions based on the GLL nodes of each element:

\[
\phi_I(x) = \begin{cases} 
\ell^e_i(x) & \text{if } I \equiv (i,e) \text{ and } x \in \Omega_e \\
0 & \text{else}
\end{cases}
\]

(2.66)

where

\[
\ell^e_i(x) = \ell_i[(\chi^e)^{-1}(x)] \quad \text{for } x \in \Omega_e
\]

(2.67)

[Figure here.]

These basis functions are continuous across inter-element boundaries. This efficiently enforces the continuity of the solution. (Same as in FEM.)
Galerkin approximation

A Galerkin approximation based on these basis functions and numerical integration leads to Equation 2.11 with

\[
\begin{align*}
K &= \sum_{e=1}^{N} K_e^e & (2.68) \\
M &= \sum_{e=1}^{N} M_e^e & (2.69) \\
f &= \sum_{e=1}^{N} f_e^e & (2.70)
\end{align*}
\]

where

\[
\begin{align*}
K_{ij}^e &= (\ell'_i, \ell'_j)_P / (h_e/2) & (2.71) \\
M_{ij}^e &= (\ell_i, \ell_j)_P h_e / 2 & (2.72) \\
f_i^e &= (f, \ell_i)_P h_e / 2 & (2.73)
\end{align*}
\]

The expanded expressions are similar to those found in Section 2.5. In particular the global mass matrix is diagonal.

Evaluating integrals:

\[
\int_a^b u(x)v(x)dx = \sum_{e=1}^{N} \int_{\Omega_e} uv dx 
\]

\[
= \sum_{e=1}^{N} \int_{-1}^{1} u_e(\xi)v_e(\xi) d\xi h_e / 2 
\]

where \( u_e(\xi) = u|_{\Omega_e}(\chi_e(\xi)) \). So,

\[
(u, v)_P = \sum_{e=1}^{N} (u_e, v_e)_P h_e / 2 
\]

Note that

\[
\phi_i^e(\xi) = \begin{cases} 
\ell_i(\xi) & \text{if } \exists i \mid I \equiv (i, e) \\
0 & \text{(if node } I \text{ belongs to element } e) \\
0 & \text{else}
\end{cases} 
\]

16
The mass matrix:

\[ M_{IJ} = (\phi_I, \phi_J)_P \]  
\[ = \sum_{e=1}^{N} (\phi^e_I, \phi^e_J)_P h^e / 2 \]  
\[ = \sum_{(i,j,e)} (\ell^e_i, \ell^e_j)_P h^e / 2 \]  

Matrix-vector multiplications

In practice, for large scale problems the assembled stiffness matrix is rarely computed and stored as is, but a function is written to evaluate the matrix-vector product \( a = Ku \). This function computes the local contributions \( a^e = K^e u^e \) on-the-fly as it assembles the vector \( a \). The pseudo-code for this function is:

1. \( a(:) = 0 \)
2. loop over \( e \) from 1 to \( N \)
   3. loop over \( i \) from 0 to \( P \)
      4. \( k = I(i,e) \)
      5. \( xe(i) = x(k) \)  
      create the local vector \( xe \)
      6. end loop over \( i \)
   7. \( ae = Ae * xe \)
8. loop over \( i \) from 0 to \( P \)
   9. \( k = I(i,e) \)
  10. \( a(k) = a(k) + ae(i) \)
 11. end loop over \( i \)
12. end loop over \( e \)

Note the similarity with the vector assembly pseudo-code. This procedure satisfies the data locality principle.

3 The SEM in higher dimensions

3.1 2D model problem

Strong form

Consider a finite spatial domain \( \Omega \in \mathbb{R}^2 \). Let \( \Gamma_D \) and \( \Gamma_N \) be a non-overlapping partition of the boundaries of \( \Omega \). Let \( f(x,y) \) be some forcing
function. Let $\nabla$ denote the gradient operator, and $\mathbf{n}$ the outward normal to a boundary.

The 2D Helmholtz model problem is:
Find $u(x, y)$ with $(x, y) \in \Omega$ such that

$$
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \lambda u + f(x, y) = 0 \quad (3.1)
$$

with homogeneous Dirichlet and Neumann conditions

$$
u(x, y) = 0 \quad \text{for } (x, y) \in \Gamma_D \quad (3.2)
$$

$$
\nabla u(x, y) \cdot \mathbf{n} = 0 \quad \text{for } (x, y) \in \Gamma_N \quad (3.3)
$$

**Weak form**

Find $u \in H^1_0(\Omega)$ such that for all $v \in H^1_0(\Omega)$ we have

$$a(u, v) - \lambda(u, v) = (f, v) \quad (3.5)$$

where the bilinear form is

$$a(u, v) = \int \int_{\Omega} \nabla u \cdot \nabla v \, dx \, dy \quad (3.6)$$

**Galerkin approximation and discrete problem**

Again, find $\mathbf{u}$ (vector of size equal to the number of global nodes, i.e. length of the non-redundant list of GLL nodes) such that:

$$\mathbf{(K} - \lambda \mathbf{M}) \mathbf{u} = \mathbf{f} \quad (3.7)$$

with

$$K_{ij} = a(\phi_i, \phi_j) \quad (3.8)$$

$$M_{ij} = (\phi_i, \phi_j) \quad (3.9)$$

$$f_i = (f, \phi_i) \quad (3.10)$$

### 3.2 Spectral element mesh

In 2D: quad elements, possibly deformed, typically Q4 (linearly deformed) or Q9 (quadratically deformed). The mesh can be unstructured. This is quite flexible, but not as much as with triangular elements. [Figures here.]
Local-to-global coordinate mapping \((x, y) = \chi^e(\xi, \eta)\) with \((\xi, \eta) \in [-1, 1]^2\). Each element is endowed with a spectral grid, \((P+1)^2\) nodes, tensor-product of the 1D GLL nodes \(\{\xi_{i}^{(P)}\}_{i=0}^{P}\):

\[
(\xi_i, \eta_j) = (\xi_{i}^{(P)}, \xi_{j}^{(P)}) \quad \text{with} \quad i, j = 0, ..., P \tag{3.11}
\]

### 3.3 Basis function

Tensor-product basis functions. Locally:

\[
\phi_I^e(\xi, \eta) = \ell_i(\xi) \ell_j(\eta) \tag{3.12}
\]

with \(I = i + j(P + 1)\) the lexicographic local index map. We will denote \(I \equiv (i, j)\). [Figure here.]

### 3.4 2D integrals

Integral in the reference element \([P] = [-1, 1]^2\):

\[
\int \int_{\square} u(\xi, \eta) \, d\xi \, d\eta = \int_{-1}^{1} \left\{ \int_{-1}^{1} u(\xi, \eta) \, d\xi \right\} \, d\eta \approx \sum_{i=0}^{P} w_i \left\{ \sum_{j=0}^{P} w_j \, u(\xi_i, \eta_j) \right\} \tag{3.13}
\]

\[
\approx \sum_{i=0}^{P} \sum_{j=0}^{P} w_i w_j \, u(\xi_i, \eta_j) \tag{3.14}
\]

Jacobian of the inverse map for element \(e\):

\[
\mathcal{J}_e^e(\xi, \eta) = \left| \begin{array}{ccc}
\frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\
\frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta}
\end{array} \right| = \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi} \frac{\partial x}{\partial \eta} \tag{3.16}
\]

Local integral term, element \(e\):

\[
\int \int_{\Omega_e} u(x, y) \, dx \, dy = \int \int_{\square} u^e(\xi, \eta) \mathcal{J}_e^e(\xi, \eta) \, d\xi \, d\eta \approx \sum_{i=0}^{P} \sum_{j=0}^{P} w_i w_j \, u(\xi_i, \eta_j) \mathcal{J}_{ij}^e \tag{3.17}
\]

where \(\mathcal{J}_{ij}^e = \mathcal{J}_e^e(\xi_i, \eta_j)\).
3.5 Local mass matrix

Assembly operations apply as in the 1D case. Let’s consider only the local contribution from element $e$. The local matrix component associated to nodes $I \equiv (i,j)$ and $K \equiv (k,l)$ is

$$M^e_{IK} = (\phi^e_I, \phi^e_K)_P$$  \hfill (3.19)

Applying the 2D GLL quadrature:

$$M^e_{IK} = \sum_{m=0}^{P} \sum_{n=0}^{P} w_m w_n [\phi^e_I \phi^e_K J^e_e](\xi_m, \eta_n)$$  \hfill (3.20)

$$= \sum_{m=0}^{P} \sum_{n=0}^{P} w_m w_n \ell_i(\xi_m) \ell_j(\eta_n) \ell_k(\xi_m) \ell_l(\eta_n) J^e_{ij}$$  \hfill (3.21)

$$= \sum_{m=0}^{P} \sum_{n=0}^{P} w_m w_n J^e_{ij} \delta_{im} \delta_{jn} \delta_{km} \delta_{ln}$$  \hfill (3.22)

$$= w_i w_j J^e_{ij} \delta_{IK}$$  \hfill (3.23)

Again, the mass matrix is diagonal by design.

3.6 Local stiffness matrix

The local contribution from element $e$ to the stiffness matrix is the integral

$$a^e(\phi_I, \phi_K) = \int \int_{\Omega_e} \left( \frac{\partial \phi_I}{\partial x} \frac{\partial \phi_K}{\partial x} + \frac{\partial \phi_I}{\partial y} \frac{\partial \phi_K}{\partial y} \right) \, dx \, dy$$  \hfill (3.25)

evaluated with the GLL quadrature. This requires a change of variables to the reference element $\Box$ with local coordinate system $(\xi, \eta)$. Applying the chain rule for differentiation:

$$\frac{\partial \phi_I}{\partial x} \frac{\partial \phi_K}{\partial x} + \frac{\partial \phi_I}{\partial y} \frac{\partial \phi_K}{\partial y} = \sum_{k=1}^{2} \left( \sum_{\alpha=\xi,\eta} \frac{\partial \phi^e_I}{\partial \alpha} \frac{\partial \phi^e_K}{\partial \alpha} \right) \left( \sum_{\beta=\xi,\eta} \frac{\partial \phi^e_I}{\partial \beta} \frac{\partial \phi^e_K}{\partial \beta} \right)$$  \hfill (3.26)

$$= \sum_{\alpha} \sum_{\beta} \frac{\partial \phi^e_I}{\partial \alpha} \frac{\partial \phi^e_K}{\partial \beta} \left( \sum_{k=1}^{2} \frac{\partial \alpha}{\partial x_k} \frac{\partial \beta}{\partial x_k} \right)$$  \hfill (3.27)

Four terms are involved:

$$K^e = \sum_{\alpha} \sum_{\beta} K^e(\alpha\beta)$$  \hfill (3.28)
with
\[
K^{e(\alpha\beta)}_{IJ} = \sum_{p=0}^{P} \sum_{q=0}^{P} W^{(\alpha\beta)}_{pq} \left[ \frac{\partial \phi_I^e}{\partial \alpha} \frac{\partial \phi_K^e}{\partial \beta} \right]_{(\xi_p, \eta_q)} \quad (3.29)
\]

where
\[
W^{(\alpha\beta)}_{pq} = w_p w_q J^e_{pq} \left[ \sum_{k=1}^{2} \frac{\partial \alpha}{\partial x_k} \frac{\partial \beta}{\partial x_k} \right]_{(\xi_p, \eta_q)} \quad (3.30)
\]

If we were to use these \((P + 1)^2 \times (P + 1)^2\) matrices as is to evaluate \(K^e u^e\) we would need \((P + 1)^4\) multiplications per element. There is however a huge gain to make by exploiting the particular structure of the SEM. The local stiffness matrix can be rewritten as
\[
K^{e(\xi)}_{IJ} = \sum_{p} \sum_{q} W^{(\xi)}_{pq} \frac{\partial \phi_I^e}{\partial \xi} (\xi_p, \eta_q) \frac{\partial \phi_J^e}{\partial \xi} (\xi_p, \eta_q) \quad (3.31)
\]

\[
= \sum_{p} \sum_{q} W^{(\xi)}_{pq} \ell_i(\xi_p) \ell_j(\eta_q) \delta_{pq} \delta_{ij} \quad (3.32)
\]

\[
= \delta_{jl} \sum_{p} W^{(\xi)}_{p} H_{ip} H_{kp} \quad (3.33)
\]

where \(H\) is the matrix of derivatives of the Lagrange basis functions (Equation 2.44). The matrix \(K^{e(\xi)}\) is very sparse: node \(I \equiv (i, j)\) interacts only with nodes in column \(j\). [Figure here.] Efficiency of tensor-product basis, low operation count compared to hp-FEM.

Matrix-vector multiplication:
\[
F^{e(\xi)}_I = K^{e(\xi)} u^e \quad (3.34)
\]

\[
F^{e(\xi)}_I = \sum_{J} K^{e(\xi)}_{IJ} u^e_J \quad (3.35)
\]

\[
= \sum_{k} \sum_{l} K^{e(\xi)}_{I(kl)} u^e_{kl} \quad (3.36)
\]

\[
= \sum_{k} \sum_{l} \delta_{jl} \sum_{p} W^{(\xi)}_{p} H_{ip} H_{kp} u^e_{kl} \quad (3.37)
\]

\[
= \sum_{k} \sum_{p} H_{ip} H_{kp} \sum_{l} \delta_{jl} W^{(\xi)}_{p} u^e_{kl} \quad (3.38)
\]

\[
= \sum_{p} H_{ip} W^{(\xi)}_{p} \sum_{k} H_{kp} u^e_{kj} \quad (3.39)
\]
Let’s store the vector $\mathbf{F}^e$ in matrix form: we define the matrix $[\mathbf{F}^e]$ with size $(P+1) \times (P+1)$ and components $F_{ij}^e$ such that $I \equiv (i,j)$. Similarly define $\mathbf{U}^e$ as the matrix-form storage of vector $\mathbf{u}^e$. We get:

$$
[F_e^{(\xi\xi)}] = H(W^{(\xi\xi)} \otimes H^t U^e)
$$

Similarly [Exercise: show it]:

$$
[F_e^{(\eta\eta)}] = (W^{(\eta\eta)} \otimes U^e H)H^t
$$

$$
[F_e^{(\xi\eta)}] = H(W^{(\xi\eta)} \otimes U^e H)
$$

$$
[F_e^{(\eta\xi)}] = (W^{(\eta\xi)} \otimes H^t U^e)H^t
$$

In practice, to compute $\mathbf{F}^e$ we proceed in three steps:

1. compute local gradients (two $m \times m$ operations)
   \[
   \nabla_\xi = H^t U^e
   \]
   \[
   \nabla_\eta = U^e H
   \]

2. evaluate component-by-component matrix multiplications:
   \[
   [F_\xi] = W^{(\xi\xi)} \otimes \nabla_\xi + W^{(\xi\eta)} \otimes \nabla_\eta
   \]
   \[
   [F_\eta] = W^{(\eta\eta)} \otimes \nabla_\eta + W^{(\eta\xi)} \otimes \nabla_\xi
   \]

3. compute (two $m \times m$ operations)
   \[
   [F^e] = H [F_\xi] + [F_\eta] H^t
   \]

These steps are the core of any SEM code. They involve 4 matrix-matrix multiplications ($m \times m$ operations) that can be highly optimized. All matrices involved have size $(P+1) \times (P+1)$. The floating point operation count in the 2D SEM is of order $(P+1)^3$, much more efficient than the order $(P+1)^4$ flops count in the FEM. In 3D the relative efficiency is still more dramatic: $(P+1)^4$ flops for the SEM versus $(P+1)^6$ for the FEM.

### 3.7 Boundary matrix

If the boundary condition is non zero on the Neumann boundary an additional term appears in the weak form:

$$
(a(u,v) - \lambda(u,v) = (f,v) + (\tau, v)_{\Gamma_N}
$$

(3.49)
where

$$\left( \tau, v \right)_{\Gamma_N} = \int_{\Gamma_N} \tau(x, y) v(x, y) \, ds$$  \hspace{1cm} (3.50)

involves the tractions $\tau$ on the Neumann boundary $\Gamma_N$. The boundary is naturally partitioned into boundary elements, which are edges of the spectral elements. The contribution from boundary element $b$ to the integral is

$$\left( \tau, v \right)_{\Gamma_N}^b = \int_{-1}^{1} \tau^b(s) v^b(s) J^b(1D)(s) \, ds$$  \hspace{1cm} (3.51)

where $v^b$ is the trace of $v$ on the $b$-th boundary element, mapped to a local curvilinear coordinate $s$, and the 1D Jacobian is

$$J^b(1D) = \sqrt{\left( \frac{\partial x}{\partial \alpha} \right)^2 + \left( \frac{\partial y}{\partial \alpha} \right)^2}$$  \hspace{1cm} (3.52)

where $\alpha$ is $\xi$ or $\eta$ if the boundary element corresponds to the element edge with $\eta = \pm 1$ or $\xi = \pm 1$, respectively.

Upon discretization this leads to

$$(K - \lambda M)u = f + B\tau$$  \hspace{1cm} (3.53)

The vector $\tau$ involves only the nodes on the boundary $\Gamma_N$. The boundary matrix $B$ is diagonal. At the boundary element level:

$$B^b_{ij} = w_i J^b_{1D} \delta_{ij}$$  \hspace{1cm} (3.54)

4 Numerical properties

Wave equation: dispersion and stability analysis, cost-accuracy trade-off, choice of $h$ and $P$.

Numerical methods for wave propagation introduce two main intrinsic errors: dispersion and dissipation. For a monochromatic plane wave these are defined as the phase error and the amplitude error, respectively. In this study we focus on dispersion errors for two reasons. First, when combined with non dissipative timestepping the SEM introduces virtually no numerical attenuation (exactly zero attenuation on element vertices). Second, at very long propagation distances the numerical errors in synthetic waveforms are dominated by the mismatch introduced by time delay (dispersion) errors, a phenomenon known in the FEM literature as the pollution effect.
For the sake of clarity, we will consider a non-dispersive medium with wave speed \( c \) governed by the scalar wave equation, and a monochromatic plane wave with frequency \( \omega \) and wavenumber \( k = \omega / c \). The phase velocity \( \omega / k \) of the numerical solution is generally different than \( c \). We quantify the dispersion error by the relative phase velocity misfit between the numerical and the theoretical wave solutions:

\[
\epsilon = \frac{-\omega/k - c}{c}. \tag{4.1}
\]

For frequency domain problems, we will see that \( \epsilon \) depends on the SEM polynomial order \( P \) and on the non-dimensional number \( \kappa = kh \), where \( h \) is the element size. Later on, for time domain problems we will see that time discretization introduces an additional error that depends on the order \( q \) of the time integration scheme and on \( \Omega = \omega \Delta t \), where \( \Delta t \) is the time step.

In practice, we can measure the time delay misfit \( \Delta T \) between a synthetic seismogram and the exact analytical solution (e.g. by sub-sample precision cross-correlation time delay estimate). The dispersion error \( \epsilon \) is related to the time delay by

\[
\epsilon = \frac{\Delta T}{T}. \tag{4.2}
\]

Because the phase velocity error \( \epsilon \) is independent of \( T \), Equation 4.2 implies that the time delay grows proportionally to the travel time.

The average number of GLL nodes per wavelength, \( G \), is defined by

\[
G = \frac{p\lambda}{h} = \frac{2\pi p}{kh}, \tag{4.3}
\]

### 4.1 Dispersion analysis in 1D

Consider an infinite 1D medium with uniform wave speed \( c \), discretized by a regular mesh of spectral elements with equal size \( h \). Consider the scalar wave propagation problem in the frequency (\( \omega \)) domain.

In a non dispersive medium \( \omega/k = c \), but waves in the semi-discrete problem follow a non-linear numerical dispersion relation of the form (4.4):

\[
\cos(kh) = R_P(\omega h/c) \tag{4.4}
\]

where \( R_P \) is a rational function (a ratio of polynomials) which coefficients depend on the spectral order \( P \). A compact expression for \( R_P \) is given by \( R_p \) for the FEM of arbitrary order \( P \). However, no similar expression is yet available for the SEM. Closed forms of \( R_P \) for low orders are derived in
Table 1: Discrete dispersion relation. We define \( x = \omega h/c \) and \( \kappa = kh \).

<table>
<thead>
<tr>
<th>( P )</th>
<th>( \mathbf{P}_P(x) )</th>
<th>( \epsilon )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( 1 - x^2/2 )</td>
<td>( \frac{x^2}{24} )</td>
</tr>
<tr>
<td>2</td>
<td>( \frac{48-22x^2+x^4}{48+2x^2} )</td>
<td>( \frac{x^6}{2880} )</td>
</tr>
<tr>
<td>3</td>
<td>( \frac{3600-1680x^2+92x^4-x^6}{3600+120x^2+9x^4} )</td>
<td>( \frac{x^8}{604800} )</td>
</tr>
</tbody>
</table>

\( \mathbf{P}_P(x) \) and \( \epsilon \), and summarized in Table 1. Dispersion curves can be obtained numerically by the procedure introduced by ?, described next.

The elementary contribution to the dynamic matrix,

\[
\mathbf{S} = -\omega^2 \mathbf{M} + \mathbf{K}
\]

(a symmetric matrix of size \((P + 1) \times (P + 1)\)), can be partitioned into the \( P - 1 \) interior nodes (subscript \( i \)) and the 2 boundary nodes of the element (subscript \( b \)),

\[
\mathbf{S} = \begin{pmatrix}
\mathbf{S}_{bb} & \mathbf{S}_{bi} \\
\mathbf{S}_{ib} & \mathbf{S}_{ii}
\end{pmatrix},
\]

then condensed on the boundary nodes (size \( 2 \times 2 \)):

\[
\mathbf{s} = \mathbf{S}_{bb} - \mathbf{S}_{bi} \mathbf{S}_{ii}^{-1} \mathbf{S}_{ib}
\]

We denote \( d_n(\omega) \) the spectral displacement of the \( n \)-th edge node \((x_n = nh)\) and

\[
G_1 = s_{12} \quad (4.8)
\]
\[
G_2 = -(s_{11} + s_{22})/2. \quad (4.9)
\]

The coefficients \( G_i \) are rational functions of \((\omega h/c)^2\). \( G_1 \) has \( P - 1 \) poles and no zero. \( G_2 \) shares the same \( P - 1 \) poles and has \( P \) zeros. The condensed discrete problem reads

\[
G_1 d_{n-1} - 2G_2 d_n + G_1 d_{n+1} = 0 \quad (4.10)
\]

The structure of the condensed problem is similar to that of a finite difference scheme, and the standard Von Neumann analysis technique can be applied. Assuming \( d_n(\omega) = d_0(\omega) \exp(ikhn) \), Equation 4.10 yields a dispersion relation between wavenumber \( k \) and frequency \( \omega \), of the form of Equation 4.4.
Dispersion curves obtained by this procedure are shown in Figure 1 for a usual range $3 \leq P \leq 8$. [...] Comment on asymptotic range (acoustic branch), optical branches at high frequency, stability limit. Brillouin’s book on waves in crystal lattices.

The asymptotic behavior of the dispersion error $\epsilon$ for $kh \ll p$ is (4.11)

$$\epsilon \approx A(P)(kh)^{2P}$$

with

$$A(P) = \frac{1}{2P(2P + 1)} \left( \frac{P!}{(2P)!} \right)^2.$$  \hspace{1cm} (4.12)

This is readily verified by the analytical dispersion relations at low $p$ in Table 1, as the leading order term of the series expansion of $(\cos(x) - R_P(x))/x^2$ at low $x$. The asymptotic behavior is illustrated on the numerical dispersion curves at higher $P$ in Figure 2.

Equation 4.11 and Equation 4.12 are of practical importance to quantify the quality of SEM and to compare it to other methods.

The dispersion error for the FEM behaves also as Equation 4.11, but with a factor $-A(P)/P$: the SEM has lower dispersion error than the high-order FEM, an advantage that becomes significant at high $P$.

The spectral convergence of the SEM upon $P$-refinement is demonstrated by rewriting the dispersion error (Equation 4.11) as

$$\epsilon \approx \frac{(\pi e/2G)^{2P}}{4p(2P + 1)}.$$ \hspace{1cm} (4.13)

where we have employed Stirling’s formula, $n! \approx \sqrt{2\pi n^{n+1/2}}e^{-n}$, to approximate $A(P) \approx (e/4P)^{2P}/[4P(2P + 1)]$. The error decays faster than exponentially as a function of $P$, provided that

$$G > e\pi/2 \approx 4.27.$$ \hspace{1cm} (4.14)

Similar conditions on the minimum number of nodes per wavelength $G$ are usually quoted as guaranteeing high quality SEM simulations. However, Equation 4.14 is only a condition to enter the super-exponential convergence regime. For practical purposes a more meaningful and useful information is the minimum $G$ required to achieve a prescribed accuracy $\epsilon$. This is plotted in Figure 3 and reported in Table 2, based on Equation 4.11 and Equation 4.12. Clearly, $G$ strongly depends on the accuracy goal and on the polynomial order.
Figure 1: Dispersion curves for SEM, including complex wavenumbers. Stability of explicit time schemes is related to the highest propagating frequency (red circle, the end of the last optical branch). The red cross indicates half that frequency.
Figure 2: Dispersion error for SEM (3 ≤ P ≤ 8) as a function of wavenumber or number of elements per wavelength (right) and as a function of number of nodes per wavelength (left). The dashed lines in the left plot are the asymptotic behavior given by Equation 4.11 and Equation 4.12.

By definition (Equation 4.2) ε has been normalized by the total travel time. If one prescribes instead the absolute error $\Delta T$ it appears that a larger $G$ is needed for larger propagation distances. It is useful to express the dispersion error in terms of the dominant period $T_0 = \omega / 2\pi$ and the number of wavelengths travelled $T/T_0$:

$$
\epsilon = \frac{\Delta T}{T_0} / \frac{T}{T_0} \quad (4.15)
$$

With $\epsilon_X = 10^{-4}$ one could achieve for instance propagation over distances of 100 wavelengths with numerical time delays of 1% of a wave period. For a global accuracy $\epsilon$ the numbers in Figure 3 and Table 2 are only lower bounds as they do not account for the additional error due to time discretization, studied in the next paragraph.

4.2 Optimal choice of $h$ and $P$

We are concerned with the two competing goals that determine the performance of a simulation: maximizing accuracy while minimizing the computational cost. In SEM accuracy can be improved by decreasing $h$ or by
Table 2: Number of nodes per wavelength $G$ required in SEM to achieve an accuracy $\epsilon_X$ (dispersion error assuming perfect time integration) for different polynomial orders $p$.

<table>
<thead>
<tr>
<th>$\epsilon_X$ (10^{-p})</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-3}$</td>
<td>40.6</td>
<td>9.6</td>
<td>6.5</td>
<td>5.5</td>
<td>5</td>
<td>4.7</td>
<td>4.5</td>
<td>4.4</td>
<td>4.4</td>
<td>4.3</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>128</td>
<td>17.2</td>
<td>9.5</td>
<td>7.3</td>
<td>6.3</td>
<td>5.7</td>
<td>5.4</td>
<td>5.1</td>
<td>5</td>
<td>4.8</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>406</td>
<td>30.5</td>
<td>14</td>
<td>9.7</td>
<td>7.9</td>
<td>6.9</td>
<td>6.3</td>
<td>5.9</td>
<td>5.6</td>
<td>5.4</td>
</tr>
</tbody>
</table>

Figure 3: Minimal number of nodes per wavelength $G$ required to achieve a specified accuracy $\epsilon_X$ (dispersion error due to space discretization). Each curve is for a different polynomial order $p$ (see legend).
increasing $P$. These two approaches, $P$-refinement and $h$-refinement, have different incidence on computational cost and the optimal choice is a priori not trivial. Our goal in this section is to derive quantitative rules for a performance-based design of SEM simulations: how to set the parameters $h$ and $p$ in order to satisfy a given error tolerance at a minimum computational cost.

We focus on the computational cost quantified as CPU time cost (how long does it take to run a simulation?). Although the cost of memory access is increasingly important, for illustrative purposes we will assume that the number of floating point operations is an adequate measure of computational cost.

The number of floating point operations in SEM is dominated by the evaluation of the internal forces $Ku$. The number $E(p)$ of multiplications involved per element in this operation is reported in Table 3 for some open source implementations of SEM. In the table we show $E(p)/(p+1)^D$ to reflect the number of operations per element per node in dimension $D$. Other implementations may differ in their balance between optimization of memory and optimization of operation count, but the leading term $O(p^{D+1})$ in $E(p)$ reported here cannot be reduced further.

The total computational cost for a frequency domain wave solver in a domain of linear dimension $L$ scales as the product of the number of elements and $L$.

---

Table 3: Number of multiplications per element per node $E(p)/(p+1)^D$ for the computation of internal forces and storage per node in some implementations of SEM

<table>
<thead>
<tr>
<th>Problem</th>
<th>Code</th>
<th>$E(p)/(p+1)^D$</th>
<th>Storage per node</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D</td>
<td>SEMLAB</td>
<td>$p+1$</td>
<td>$p+4$</td>
</tr>
<tr>
<td>2D SH</td>
<td>SEM2DPACK</td>
<td>$4(p+2)$</td>
<td>6</td>
</tr>
<tr>
<td>2D P-SV</td>
<td>SEM2DPACK</td>
<td>$8(p+3)$</td>
<td>16</td>
</tr>
<tr>
<td>2.5D monopole</td>
<td>AXISEM</td>
<td>$8(p+7)$</td>
<td></td>
</tr>
<tr>
<td>2.5D di/quadrupole</td>
<td>AXISEM</td>
<td>$12(p+9)$</td>
<td></td>
</tr>
<tr>
<td>3D</td>
<td>SPECFEM3D</td>
<td>$9(2p+11)$</td>
<td>22</td>
</tr>
</tbody>
</table>

---

1SEMLAB and SEM2DPACK are available online at www.sg.geophys.ethz.ch/geodynamics/ampuero/software.html, SPECFEM3D is available at www.geodynamics.org.
the cost per element
\[ \text{cost} \propto E(P) \left( \frac{L}{h} \right)^D \]  
(4.16)

It can be rewritten as
\[ \text{cost} \propto (kL)^D \times \Gamma, \]  
(4.17)

where
\[ \Gamma = \frac{E(P)}{(kh)^D}. \]  
(4.18)

The first term in Equation 4.17 is entirely due to the physical dimensions of the problem whereas the term \( \Gamma \) encapsulates the effect of the numerical resolution. In the remainder we will refer to \( \Gamma \) as the computational cost. We have assumed that a good preconditioner is available, so that the number of iterations required by the solver is practically a constant. Note that for large \( P \) we get
\[ \Gamma \propto \frac{(P + 1)^{D+1}}{(kh)^D} \]  
(4.19)

We can now address the cost-accuracy trade-offs of SEM for frequency domain solvers. For a given accuracy level \( \epsilon \) we must find the optimal values \( h_{opt} \) and \( P_{opt} \) that minimize the cost function in Equation 4.18 under the constrain of Equation 4.11 (fixed accuracy). This optimization problem is solved graphically in Figure 4, where we show the cost as a function of \( P \) and \( \epsilon \) assuming \( h \) is given by Equation 4.11. The minimum of each curve (indicated by circles) gives the optimal values of \( P \) for a given accuracy level. [Plot also the optimal \( h \).]

## 5 Time integration of the wave equation

### 5.1 Second order schemes

**Leapfrog**

\[
\begin{align*}
    d_{n+1} & = d_n + \Delta t v_{n+1/2} \\
    v_{n+3/2} & = v_{n+1/2} + \Delta t M^{-1} \left[ -K d_{n+1} + f(t_n) \right]
\end{align*}
\]  
(5.1, 5.2)

Storage requirements: two global fields, \( d \) and \( v \).
Figure 4: Theoretical cost-performance for an SEM frequency domain solver (equivalent to assume perfect time integration). The computational cost is shown as a function of polynomial order $p$. Each curve corresponds to a given accuracy level (dispersion error indicated in the legend). We assume an optimal choice of element size $h$ for a given accuracy and $p$. The optimal $p$ (circles) is lower than in the case with imperfect time integration.
Newmark schemes

\[ \begin{align*}
M a_{n+1} &= -K d_{n+1} + f(t_{n+1}) \quad (5.3) \\
v_{n+1} &= v_n + (1 - \gamma) \Delta t a_n + \gamma \Delta t a_{n+1} \quad (5.4) \\
d_{n+1} &= d_n + \Delta t v_n + (1/2 - \beta) \Delta t^2 a_n + \beta \Delta t^2 a_{n+1} \quad (5.5)
\end{align*} \]

Predictor-corrector implementation:

1. Predictor:

\[ \begin{align*}
\tilde{d} &= d_n + \Delta t v_n + (1/2 - \beta) \Delta t^2 a_n \quad (5.6) \\
\tilde{v} &= v_n + (1 - \gamma) \Delta t a_n \quad (5.7)
\end{align*} \]

2. Solver:

\[ a_{n+1} = M^{-1} [-K \tilde{d} + f(t_{n+1})] \quad (5.8) \]

3. Corrector:

\[ \begin{align*}
d_{n+1} &= \tilde{d} + \beta \Delta t^2 a_{n+1} \quad (5.9) \\
v_{n+1} &= \tilde{v} + \gamma \Delta t a_{n+1} \quad (5.10)
\end{align*} \]

Storage requirements: 3 global fields, \(d, v\) and \(a\).
The scheme is equivalent to leapfrog if \(\beta = 0\) and \(\gamma = 1/2\).
The displacement update is independent of \(a\) if \(\beta = 0\).

**Stability**

Explicit time integration schemes are *conditionally stable*, they are stable only if the timestep \(\Delta t\) verifies

\[ \Delta t < \Delta t_c = \frac{C}{\sqrt{D \Omega_m}} \frac{h}{c}, \quad (5.11) \]

where \(\Delta t_c\) is the critical timestep, \(D\) the dimension of the problem, \(C\) a stability number that depends only on the time scheme and \(\Omega_m\) a spectral radius that depends only on the space discretization scheme. The spectral radius is the highest eigen-frequency of the dimensionless 1D wave equation \((c = 1)\) discretized by spectral elements with \(h = 1\). In terms of the dispersion relation, \(\Omega_m\) is the smallest solution of

\[ R_P(\Omega_m) = \cos(p\pi) = (-1)^P. \quad (5.12) \]
Table 4: Spectral radius $\Omega_m$ in SEM for different polynomial orders $p$.

<table>
<thead>
<tr>
<th>$p$</th>
<th>1</th>
<th>2</th>
<th>$3$</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Omega_m$</td>
<td>2</td>
<td>$2\sqrt{6}$</td>
<td>$\sqrt{42 + 6\sqrt{29}}$</td>
<td>13.55</td>
<td>19.80</td>
<td>27.38</td>
<td>36.27</td>
<td>46.42</td>
<td>57.87</td>
<td>70.6</td>
</tr>
</tbody>
</table>

In general $\Omega_m$ is inversely proportional to the smallest node spacing. The Gauss-Lobatto-Legendre nodes of SEM are clustered at the edge of the elements with spacing $\propto 1/P^2$, so we expect $\Omega_m \propto P^2$. Values for $P \leq 10$ are reported in Table 4 and are well approximated (better than 1% for $P \geq 4$) by

$$\Omega_m(P) \approx 0.64 P^2 + 0.57 P + 1.$$  \hspace{1cm} (5.13)

Compared to the $P$-FEM (??) the SEM has the advantage of a smaller spectral radius, hence larger timesteps can be used.

**Accuracy**

Time schemes introduce additional dispersion errors. In general dissipation errors may also be present but we are mainly interested here in conservative schemes. For a general scheme of order $q$ the error behaves as

$$\epsilon \approx B(\omega \Delta t)^q.$$  \hspace{1cm} (5.14)

The usual leapfrog scheme is second order accurate, $q = 2$, and its dimensionless factor is $B = 1/24$. Within the second order class it is the most accurate scheme.

Within the asymptotic regime, the total dispersion error combining space and time discretizations is

$$\epsilon = A(P) (kh)^{2P} + B(\omega \Delta t)^q.$$  \hspace{1cm} (5.15)

In common practice the time step is set to a moderate fraction of the critical value (5.11), i.e. $\Delta t = \gamma \Delta t_c$ with $\gamma \lesssim 1$. We can write $\epsilon$ in terms of $\kappa = kh$ as

$$\epsilon \approx A(P) \kappa^{2P} + B \left(\frac{\gamma C}{\sqrt{D \Omega_m(P)}}\right)^q \kappa^q.$$  \hspace{1cm} (5.16)

Both components of the dispersion error are compared in Figure 5 for $P = 8$. The error is dominated by the time-discretization error as soon as more than
Figure 5: Phase velocity curve (top) and dispersion error (bottom) for SEM with $p = 8$ as a function of resolution (wavelengths per element or nodes per wavelength). The green dashed line is the theoretical estimate, the black dashed line is the time-discretization error of the centered difference scheme.
Figure 6: Space and time discretization components ($\epsilon_X$ and $\epsilon_T$) of the dispersion error in 3D SEM with second order timestepping. Each curve is drawn for a given number of nodes per wavelength ($G$). Each square corresponds to a different polynomial order $p$, from 2 to 19. Lower errors are for lower $p$, as labeled for the curve $G = 4.5$. The timestep has been set to $\Delta t = \Delta t_c/2$. Curves are rarely well below the dashed line: there is virtually no case of practical interest ($\epsilon \ll 10^{-2}$) where $\epsilon_T \ll \epsilon_X$. For small $\epsilon$ the dispersion error is dominated by $\epsilon_T$ (the upper dotted lines indicate $\epsilon_T/\epsilon_X = 10$ and 100).
4 nodes per wavelength are employed. At $G = 10$, $\epsilon_T$ is about six orders of magnitude larger than $\epsilon_X$. In Figure 6 we illustrate, for $D = 3$, $\gamma = 0.5$ and a range of values of $p$ and $G$, that in virtually no case of practical relevance ($\epsilon \ll 1\%$) is the time discretization error $\epsilon_T$ neglectable with respect to the space discretization error $\epsilon_X$. In simulations requiring reasonable accuracy $\epsilon < 10^{-3}$, with the usual setting $\gamma \lesssim 1$, the dispersion error is largely dominated by the low order time discretization errors and the advantages of spectral convergence of the SEM are wasted. In such situations $\epsilon_T$ must be controlled by reducing the timestep $\Delta t$. For instance, for $p = 6$, $G = 6$, to achieve an error of 1% of a period over a propagation distance of 100 wavelengths ($\epsilon = 10^{-4}$) one needs to reduce $\epsilon_T$ by a factor of 10. This requires a $\sqrt{10} \approx 3$-fold reduction of $\Delta t$ and implies a $\approx 3$-fold increase in computational cost. We will see in later sections that a more efficient error reduction can be achieved with higher order time schemes ($q > 2$).

### 5.2 Higher-order symplectic schemes

There are several methods to achieve higher order in time integration. The so-called *symplectic* timestepping algorithms are very convenient. The update of displacement and velocity fields, $d$ and $v$, from time $t$ to $t + \Delta t$ is composed of a $n$-stage sub-stepping iteration. For $k = 1$ to $n$, do:

\begin{align}
  t &\leftarrow t + a_k \Delta t \\
  d &\leftarrow d + a_k \Delta t \ v \\
  v &\leftarrow v + b_k \Delta t \ M^{-1}[-Kd + f(t)]
\end{align}

(5.17)

(5.18)

(5.19)

Then apply the closing stage:

\begin{align}
  t &\leftarrow t + a_{n+1} \Delta t \\
  d &\leftarrow d + a_{n+1} \Delta t \ v
\end{align}

(5.20)

(5.21)

This family of algorithms requires $n$ evaluations of internal forces $Kd$ per global timestep. The implementation is remarkably simple, it takes a few lines of coding to modify existing SEM solvers, and it does not require additional memory storage. Time-reversibility is guaranteed by imposing the following symmetries on the coefficients: $a_k = a_{n+2-k}$ and $b_k = b_{n+1-k}$.

The order $q$ of the method depends on the number of stages $n$ and on the coefficients $a_k$ and $b_k$. The cost increases linearly with $n$. These parameters can be optimized to achieve high accuracy at low cost.
For instance, a good fourth order algorithm is the position extended Forest-Ruth-like (PEFRL) scheme:

\[
\begin{align*}
    a_1 &= a_5 = \xi & b_1 &= b_4 = 1/2 - \lambda \\
    a_2 &= a_4 = \chi & b_2 &= b_3 = \lambda \\
    a_3 &= 1 - 2(\chi + \xi)
\end{align*}
\]

(5.22)

with

\[
\begin{align*}
    \xi &= 0.1786178958448091 \\
    \lambda &= -0.2123418310626054 \\
    \chi &= -0.06626458266981849
\end{align*}
\]

Its error coefficient is \( B = 1/12500 \) and its critical CFL number is \( C = 2.97633 \).

6 **Miscellaneous**

6.1 **Limitations of the SEM**

- quad mesh is not that flexible, no automated meshing algorithms
- spectral convergence is lost in non-smooth problems (singularities, material boundaries within element)
- time schemes are usually low order, need improvement

6.2 **Advanced topics**

- Dynamic faulting boundary conditions
- Geometrically non-conforming meshes: mortar elements
- Preconditioners (Schwartz). Condition number prop to \( P^3/h^2 \) in 2D, see Canuto 5.3.4. See also Canuto (5.3.5), Deville (4.5.1), Karniadakis (4.2)
- spectral discontinuous Galerking methods
- a posteriori errors and adaptivity
- SEM on triangles and tetrahedra
A References

Boyd (200?), “Spectral elements” (chapter of book in progress):
Gottlieb and Hesthaven, “Spectral methods for hyperbolic problems”: duality between modal Legendre approximation and nodal Lagrange interpolation (section 2), computational efficiency (section 7.4)
van de Vosse and Minev (1996), “Spectral element methods: theory and applications”: spectral, pseudo-spectral and spectral elements (sections 2.2 and 2.3)
Canuto, “Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics”: Chapter 5

B Available software

available SEM codes: SPECFEM3D, SEM2DPACK, Gaspar, SPECULOOS, SEAM
meshing software: CUBIT, GiD, EMC2
C Homework problems

C.1 Love wave modes in a horizontally layered medium

Consider shear waves on an elastic medium made of two horizontal layers. The shallow layer is thinner \((H_1 = 400 \text{ m})\) and has lower shear wave speed \((c_1 = 1000 \text{ m/s})\) than the deeper one \((H_2 = 3600 \text{ m and } c_2 = 2000 \text{ m/s})\). Density is the same in both layers, \(\rho = 2000 \text{ kg/m}^3\). The top surface (at \(z = 0\)) is stress-free (Neumann) and the bottom surface (at \(z = Z = 4000 \text{ m}\)) is displacement-locked (Dirichlet).

For a given frequency \(f\) (or angular frequency \(\omega = 2\pi f\)), we focus on Love waves, i.e. surface waves with motion in the out-of-plane \(y\) direction, horizontal and perpendicular to the propagation direction \(x\). We consider a displacement field \(d\) of the following form

\[
d(x, z, t) = u(z) \exp(\text{i}\omega t - \text{i}kx)y
\]  

We will look for the depth distribution of displacement \(u(z)\). The strong form of this problem is:

Find the displacement profile \(u(z)\) and horizontal wavenumber \(k\) such that

\[
-\omega^2 \rho(z) u(z) + k^2 \mu(z) u(z) - \frac{d}{dz} \left( \mu(z) \frac{du}{dz} \right) = 0 \tag{C.2}
\]

with the following boundary conditions:

\[
\frac{du}{dz}(z = 0) = 0 \tag{C.3}
\]

\[
u(z = H_1 + H_2) = 0 \tag{C.4}
\]

and continuity of \(u\) and \(du/dz\) across the material interface.

This is actually an eigenvalue problem. It has multiple solution pairs, or modes (eigenvalues \(k_m\) and eigenfunctions \(u_m(z)\)). The modes are ordered by decreasing value of their \(k_m\). The fundamental mode corresponds to \(m = 0\) and the overtones to \(m > 0\).

1. Formulate the weak form of the problem

2. The SEM discretization of the eigenvalue problem is:

Find nodal displacements \(u_m\) and wavenumber \(k_m\) such that

\[
[K_z - \omega^2 M] u_m = -k_m^2 K_x u_m \tag{C.5}
\]
Write down the expression for the local matrices $M^e$, $K_x^e$ and $K_z^e$. Note that the structure of $K_x$ is similar to that of $M$, and that the bottom node (Dirichlet condition) must be removed from the unknowns. Consider two cases:

(a) the layer interface lies inside an element (case A)

(b) the interface coincides with an inter-element boundary (Case B)

3. Write a code that

(a) generates a 1D mesh in which each element can have a different size. For case A, the top two elements should have size $h = 300$ m and the others $h = 850$ m. For case B, $h = 400$ m in layer 1 and $h = 900$ m in layer 2.

(b) computes the SEM matrices for cases A and B. Let the polynomial degree $P$ (the same for all elements) be a user specified parameter.

(c) finds the two eigenvalues of the discrete problem that have the smallest algebraic value ($-k_0^2$ and $-k_1^2$), and the associated eigenvectors

(d) evaluates by GLL quadrature a measure of error defined as the integral of the squared difference between the SEM solutions and their exact values (computed analytically, see below):

$$
\epsilon = \int_0^Z \left( u_{m,SEM}(z) - u_{m,exact}(z) \right)^2 dz \quad \text{(C.6)}
$$

4. Set $P = 5$ and $f = 2.5$ Hz. Compute the depth profile of displacement for the fundamental Love mode and its first overtone for cases A and B and for at least three different meshes, each with twice as high resolution (smaller elements) as the previous one.

5. Plot displacement as a function of depth for both modes at the highest resolution, together with their exact values.

6. For cases A and B, plot in log-log scale the error $\epsilon$ as a function of the average value of element size $h$ over the mesh. Discuss.

Resources:

- Matlab functions provided:
– GetGLL.m computes the GLL quadrature nodes, weights and the matrix of Lagrange derivatives $H$.

– love_analytic.m computes the exact solution analytically

• Generalized eigenvalue problems of the form: find $u_m$ and $\lambda_m$ such that

$$Au_m = \lambda_mBu_m,$$  \hspace{1cm} (C.7)

can be solved in Matlab with the function `eigs`. In particular the first two modes are given by `eigs(A,B,2,’SA’)`. Try to exploit the sparse structure of the matrices.

C.2 Wave propagation and numerical dispersion

Consider the propagation of scalar waves in a 1D elastic medium of finite size, $x \in [0,L]$ with $L = 10$, and uniform properties (density $\rho = 1$, wave speed $c = 1$). The boundary conditions are free stress at both ends. Consider a point source located at $x = L/2$ with the following source time function:

$$f(t) = (2\tau - 1)\exp(-\tau) \text{ where } \tau = (\pi f_0(t - t_0))^2$$ \hspace{1cm} (C.8)

a Ricker wavelet with dominant frequency $f_0 = 0.5$ and delay $t_0 = 1.5/f_0$.

1. Write a code that solves this wave propagation problem with the SEM. Set $P = 8$ and $h = 1$. Consider two different time schemes: leapfrog (case A) and the PEFRL symplectic scheme defined in Equation 5.22 (case B). Set the timestep $\Delta t = 0.85 \min(\Delta x)/c$ for case A, and $\Delta t = 1.25 \min(\Delta x)/c$ for case B, where $\min(\Delta x)$ is the smallest distance between GLL nodes.

2. Compute the solution up to $t = 40L/c$. Plot the seismogram (displacement solution as a function of time) in the node located at $x = 0$.

3. You should see multiple reflected phases in the seismogram. Extract the timing $T_k$ of the displacement peaks of each phase. Plot $T_k - T_0$ as a function of $k$ for cases A and B.

4. Plot the dispersion error of the last reflected phase, defined as the difference between $T_k - T_0$ and the theoretical travel time of this wave.

5. Repeat the simulations now reducing the timestep $\Delta t$ by a factor 2, then a factor 4. Plot the travel time error as a function of $\Delta t$ for cases A and B. Discuss.