A general method to compute numerical dispersion error

J. Ruano[1], A. Baez Vidal[1], J. Rigola[1], F. X. Trias[1]

Heat and Mass Transfer Technological Center (CTTC), Universitat Politècnica de Catalunya – BarcelonaTech (UPC) ESEIAAT

ECCOMAS Congress 2020, January 2021

Index of Contents

Numerical Dispersion errors: What are they?

Some background on numerical errors

Numerical Dispersion errors: What are they?

Some background on numerical errors

- Numerical derivatives do not match analytical ones.
    - Numerical errors are introduced when equations are discretised.

Numerical Dispersion errors: What are they?

Some background on numerical errors

- Numerical derivatives do not match analytical ones.
    - Numerical errors are introduced when equations are discretised.
- Numerical Diffusion is well known and is easy to eliminate: central or symmetric schemes.
    - If it is not eliminated, the error is proportional to Δx.
    - Thus, is easy to reduce: densify mesh.

Numerical Dispersion errors: What are they?

Some background on numerical errors

- Numerical derivatives do not match analytical ones.
    - Numerical errors are introduced when equations are discretised.
- Numerical Diffusion is well known and is easy to eliminate: central or symmetric schemes.
    - If it is not eliminated, the error is proportional to Δx.
    - Thus, is easy to reduce: densify mesh.
- Numerical Dispersion cannot be avoided, just reduce it.
    - Except if Spectral Methods are used, where derivative is imposed to be exact: $f'(k) = kf(k)$.

Numerical Dispersion errors: What are they?

Some background on numerical errors

- Numerical derivatives do not match analytical ones.
  - Numerical errors are introduced when equations are discretised.
- Numerical Diffusion is well known and is easy to eliminate: central or symmetric schemes.
  - If it is not eliminated, the error is proportional to Δx.
  - Thus, is easy to reduce: densify mesh.
- Numerical Dispersion cannot be avoided, just reduce it.
  - Except if Spectral Methods are used, where derivative is imposed to be exact: $f'(k) = kf(k)$.
- How is Numerical Dispersion usually studied? By means of a **Fourier Transform.**

Numerical Dispersion errors: What are they?

Some background on numerical dispersion error

Numerical Dispersion errors: What are they?

Some background on numerical dispersion error

- If a Fourier Transform is used, then just periodic domains with uniform meshes can be studied.
    - The extrapolation to 3D unstructured domains with generic boundary conditions is NOT straightforward.

Numerical Dispersion errors: What are they?

Some background on numerical dispersion error

- If a Fourier Transform is used, then just periodic domains with uniform meshes can be studied.
  - The extrapolation to 3D unstructured domains with generic boundary conditions is NOT straightforward.
- Authors report that conclusions extracted in uniform meshes fail in slightly stretched meshes.
  - Not even unstructured; just stretched.

Numerical Dispersion errors: What are they?

Some background on numerical dispersion error

- If a Fourier Transform is used, then just periodic domains with uniform meshes can be studied.
  - The extrapolation to 3D unstructured domains with generic boundary conditions is NOT straightforward.
- Authors report that conclusions extracted in uniform meshes fail in slightly stretched meshes.
  - Not even unstructured; just stretched.
- A methodology that allows studying dispersion in a general mesh would be interesting.
  - Numerical dispersion is, then, a function of the studied mesh.

Numerical Dispersion errors: What are they?

Some background on numerical dispersion error

- If a Fourier Transform is used, then just periodic domains with uniform meshes can be studied.
  - The extrapolation to 3D unstructured domains with generic boundary conditions is NOT straightforward.
- Authors report that conclusions extracted in uniform meshes fail in slightly stretched meshes.
  - Not even unstructured; just stretched.
- A methodology that allows studying dispersion in a general mesh would be interesting.
  - Numerical dispersion is, then, a function of the studied mesh.
- Instead of using the sinusoids base, use an orthogonal base extracted from studied mesh.
  - For example, **eigenvectors of the discrete Laplacian matrix.**

Methodology: Calculus background (I)

Let $\Phi = \{\phi_{-N}(x), \phi_{-N+1}(x), \ldots \phi_{-1}(x), \phi_0(x), \phi_1(x), \ldots \phi_N(x)\}$ be an orthonormal basis of functions in a domain Ω_x.

Methodology: Calculus background (I)

Let $\Phi = \{\phi_{-N}(x), \phi_{-N+1}(x), \ldots \phi_{-1}(x), \phi_0(x), \phi_1(x), \ldots \phi_N(x)\}$ be an orthonormal basis of functions in a domain Ω_x.

We can define a mapping T

Methodology: Calculus background (I)

Let $\Phi = \{\phi_{-N}(x), \phi_{-N+1}(x), \ldots \phi_{-1}(x), \phi_0(x), \phi_1(x), \ldots \phi_N(x)\}$ be an orthonormal basis of functions in a domain Ω_x.

We can define a mapping T

$T : \mathcal{L}^2(\Omega_x, x) \mapsto \mathbb{C}^{2N+1}$; $T : f(x) \mapsto (\alpha_m) \in \mathbb{C}^{2N+1}$, where

$$\alpha_m = \langle f \mid \phi_m \rangle_{\Omega_x} = \int_{\Omega_x} f(x)\overline{\phi_m}(x)\, dx, \quad (1)$$

Methodology: Calculus background (I)

Let $\Phi = \{\phi_{-N}(x), \phi_{-N+1}(x), \ldots \phi_{-1}(x), \phi_0(x), \phi_1(x), \ldots \phi_N(x)\}$ be an orthonormal basis of functions in a domain Ω_x.

We can define a mapping T

$T : \mathcal{L}^2(\Omega_x, x) \mapsto \mathbb{C}^{2N+1}$; $T : f(x) \mapsto (\alpha_m) \in \mathbb{C}^{2N+1}$, where

$$\alpha_m = \langle f \mid \phi_m \rangle_{\Omega_x} = \int_{\Omega_x} f(x)\overline{\phi_m}(x)\, dx, \tag{1}$$

And the inverse mapping of T

Methodology: Calculus background (I)

Let $\Phi = \{\phi_{-N}(x), \phi_{-N+1}(x), \dots \phi_{-1}(x), \phi_0(x), \phi_1(x), \dots \phi_N(x)\}$ be an orthonormal basis of functions in a domain Ω_x.

We can define a mapping T

$T : \mathcal{L}^2(\Omega_x, x) \mapsto \mathbb{C}^{2N+1}$; $T : f(x) \mapsto (\alpha_m) \in \mathbb{C}^{2N+1}$, where

$$\alpha_m = \langle f \mid \phi_m \rangle_{\Omega_x} = \int_{\Omega_x} f(x)\overline{\phi_m}(x)\, dx, \tag{1}$$

And the inverse mapping of T

$T^{-1} : \mathbb{C}^{2N+1} \mapsto \mathcal{L}^2(\Omega_x, x)$, $T : (\alpha_m) \in \mathbb{C}^{2N+1} \mapsto f(x)$.

$$f(x) \simeq S_N = \sum_{m=-N}^{N} \alpha_m \phi_m(x); \quad \lim_{N\to\infty} S_N = f(x). \tag{2}$$

Methodology: Calculus background (II)

We can write the derivative of $f(x)$ in terms of the orthonormal basis Φ:

Methodology: Calculus background (II)

We can write the derivative of $f(x)$ in terms of the orthonormal basis Φ:

$$f'(x) \simeq S'_N = \sum_{m=-N}^{N} \alpha_m \phi'_m(x) \simeq \sum_{m=-N}^{N} \left(\alpha_m \sum_{n=-N}^{N} \gamma_{mn} \, \phi_n(x) \right), \qquad (3)$$

where γ_{mn} represent the projections of the derivatives of ϕ_m on ϕ_n.

Methodology: Calculus background (II)

We can write the derivative of $f(x)$ in terms of the orthonormal basis Φ:

$$f'(x) \simeq S'_N = \sum_{m=-N}^{N} \alpha_m \phi'_m(x) \simeq \sum_{m=-N}^{N} \left(\alpha_m \sum_{n=-N}^{N} \gamma_{mn} \phi_n(x) \right), \quad (3)$$

where γ_{mn} represent the projections of the derivatives of ϕ_m on ϕ_n.

We can define a matrix Γ

Methodology: Calculus background (II)

We can write the derivative of $f(x)$ in terms of the orthonormal basis Φ:

$$f'(x) \simeq S'_N = \sum_{m=-N}^{N} \alpha_m \phi'_m(x) \simeq \sum_{m=-N}^{N} \left(\alpha_m \sum_{n=-N}^{N} \gamma_{mn}\, \phi_n(x) \right), \tag{3}$$

where γ_{mn} represent the projections of the derivatives of ϕ_m on ϕ_n.

We can define a matrix Γ

Where its elements $(\Gamma)_{mn} = \gamma_{mn} = \langle \phi'_m \,|\, \phi_n \rangle_{\Omega_x}$. The structure of Γ will provide information about the errors produced during differentiation.

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

- $\gamma_{mn} \neq 0$ if $m \neq n$,

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

- $\gamma_{mn} \neq 0$ if $m \neq n$,
- $Re(\gamma_{mm}) \neq 0$,

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

- $\gamma_{mn} \neq 0$ if $m \neq n$,
- $Re(\gamma_{mm}) \neq 0$,
- $\frac{Im(\gamma_{mm})}{k_m} \neq 1$,

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

- $\gamma_{mn} \neq 0$ if $m \neq n$, Aliasing
- $Re(\gamma_{mm}) \neq 0$,
- $\dfrac{Im(\gamma_{mm})}{k_m} \neq 1$,

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

- $\gamma_{mn} \neq 0$ if $m \neq n$,
- $Re(\gamma_{mm}) \neq 0$, Difusion
- $\frac{Im(\gamma_{mm})}{k_m} \neq 1$,

Methodology: Calculus background (III)

Some calculus background: Example with sinusoids

If sinusoids ($\phi_m = e^{ik_m x}$) are used as the orthonormal base, such as Fourier Transform does, then matrix Γ should be:

$$\Gamma = diag(k_m) \in \mathbb{I}.$$

However, three different errors could occur:

- $\gamma_{mn} \neq 0$ if $m \neq n$,
- $Re(\gamma_{mm}) \neq 0$,
- $\dfrac{Im(\gamma_{mm})}{k_m} \neq 1$, Dispersion

Methodology: Algebra background (I)

Hermitian and Skew-Hermitian matrices

Methodology: Algebra background (I)

Hermitian and Skew-Hermitian matrices

Every matrix A, for example a discrete differential operator, can be decomposed as the sum of an Hermitian, D, plus skew-Hermitian, C:

Methodology: Algebra background (I)

Hermitian and Skew-Hermitian matrices

Every matrix A, for example a discrete differential operator, can be decomposed as the sum of an Hermitian, D, plus skew-Hermitian, C:

$$C = \frac{1}{2}(A - A^*) \qquad D = \frac{1}{2}(A + A^*) \tag{4}$$

Methodology: Algebra background (I)

Hermitian and Skew-Hermitian matrices

Every matrix A, for example a discrete differential operator, can be decomposed as the sum of an Hermitian, D, plus skew-Hermitian, C:

$$C = \frac{1}{2}(A - A^*) \qquad D = \frac{1}{2}(A + A^*) \tag{4}$$

Expressing the terms of matrix Γ in a discrete way, denoted by $\widetilde{\gamma_{mn}}$, using aforementioned properties:

Methodology: Algebra background (I)

Hermitian and Skew-Hermitian matrices

Every matrix A, for example a discrete differential operator, can be decomposed as the sum of an Hermitian, D, plus skew-Hermitian, C:

$$C = \frac{1}{2}(A - A^*) \qquad D = \frac{1}{2}(A + A^*) \tag{4}$$

Expressing the terms of matrix Γ in a discrete way, denoted by $\widetilde{\gamma_{mn}}$, using aforementioned properties:

$$\widetilde{\gamma_{mn}} = \langle A\phi_m \,|\, \phi_n \rangle \tag{5}$$

$$Im(\widetilde{\gamma_{mn}}) = \langle C\phi_m \,|\, \phi_n \rangle = \frac{\langle A\phi_m \,|\, \phi_n \rangle - \langle \phi_m \,|\, A\phi_n \rangle}{2} \tag{6}$$

$$Re(\widetilde{\gamma_{mn}}) = \langle D\phi_m \,|\, \phi_n \rangle = \frac{\langle A\phi_m \,|\, \phi_n \rangle + \langle \phi_m \,|\, A\phi_n \rangle}{2} \tag{7}$$

Methodology: Orthonormal basis

Discrete Laplacian eigenvectors

Methodology: Orthonormal basis

Discrete Laplacian eigenvectors

- It is the logical choice.

Methodology: Orthonormal basis

Discrete Laplacian eigenvectors

- It is the logical choice.
  - If this has begun with a generalisation of a method that uses Fourier Transform, it's logical to employ the discrete version of what Fourier does: using eigenfunctions of the continuous Laplacian.

Methodology: Orthonormal basis

Discrete Laplacian eigenvectors

- It is the logical choice.
  - If this has begun with a generalisation of a method that uses Fourier Transform, it's logical to employ the discrete version of what Fourier does: using eigenfunctions of the continuous Laplacian.
  - In evenly spaced domains, i.e. structured uniform meshes, eigenvectors are discretised sinusoids.

Methodology: Orthonormal basis

Discrete Laplacian eigenvectors

- It is the logical choice.
  - If this has begun with a generalisation of a method that uses Fourier Transform, it's logical to employ the discrete version of what Fourier does: using eigenfunctions of the continuous Laplacian.
  - In evenly spaced domains, i.e. structured uniform meshes, eigenvectors are discretised sinusoids.
- The set of eigenvectors form an orthonormal base.

Methodology: Orthonormal basis

Discrete Laplacian eigenvectors

- It is the logical choice.
  - If this has begun with a generalisation of a method that uses Fourier Transform, it's logical to employ the discrete version of what Fourier does: using eigenfunctions of the continuous Laplacian.
  - In evenly spaced domains, i.e. structured uniform meshes, eigenvectors are discretised sinusoids.
- The set of eigenvectors form an orthonormal base.
- In the continuous limit, eigenvectors and eigenvalues colapse onto its corresponding eigenfunctions, i.e sinusoids.

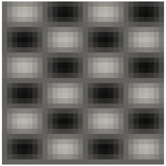
Methodology: Orthonormal basis

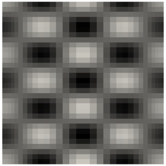
Discrete Laplacian eigenvectors

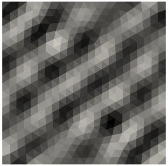
- It is the logical choice.
  - If this has begun with a generalisation of a method that uses Fourier Transform, it's logical to employ the discrete version of what Fourier does: using eigenfunctions of the continuous Laplacian.
  - In evenly spaced domains, i.e. structured uniform meshes, eigenvectors are discretised sinusoids.
- The set of eigenvectors form an orthonormal base.
- In the continuous limit, eigenvectors and eigenvalues colapse onto its corresponding eigenfunctions, i.e sinusoids.
- Retain the concept of mesh connectivity without being restrained to mesh uniformity.

Methodology: Eigenvectors example

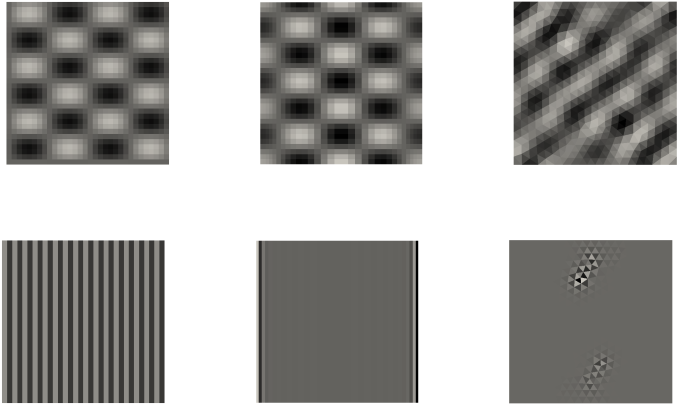
Methodology: Eigenvectors example







Methodology: Eigenvectors example



Methodology: Phase

Rotation matrix

Methodology: Phase

Rotation matrix

Sinusoids orthonormal basis have a free parameter: the phase of the function.

- Working in a discrete way, with eigenvectors, this is translated as a matrix rotation.

Methodology: Phase

Rotation matrix

Sinusoids orthonormal basis have a free parameter: the phase of the function.

- Working in a discrete way, with eigenvectors, this is translated as a matrix rotation.

This allows to obtain the average of the recovered numerical eigenvalue.

- Useful for non-linear operators or when non-uniform meshes are used.

Methodology: Phase

Rotation matrix

Sinusoids orthonormal basis have a free parameter: the phase of the function.

- Working in a discrete way, with eigenvectors, this is translated as a matrix rotation.

This allows to obtain the average of the recovered numerical eigenvalue.

- Useful for non-linear operators or when non-uniform meshes are used.

The matrix containing eigenvectors is multiplied by a rotation matrix with a random phase.

- And this is repeated N times (5000) to ensure a correct average.

Test cases: Selected cases

Used schemes

Test cases: Selected cases

Used schemes

Mixture of linear and non-linear schemes:

Test cases: Selected cases

Used schemes

Mixture of linear and non-linear schemes:

- Symmetry preserving of 2^{nd} and 6^{th} order {SP2, SP6} Dispersion relation preserving of 4^{th} and 6^{th} order {DRP4, DRP6}, and Moving Least squares of 6^{th} order {MLS3}

Test cases: Selected cases

Used schemes

Mixture of linear and non-linear schemes:

- Symmetry preserving of 2^{nd} and 6^{th} order {SP2, SP6} Dispersion relation preserving of 4^{th} and 6^{th} order {DRP4, DRP6}, and Moving Least squares of 6^{th} order {MLS3}
- First-order upwind {UPW}, WENO of 3^{rd}, 5^{th} and 7^{th} order {WENO3, WENO5, WENO7}, and Superbee {SB}, Van Leer{VL} and Minmod {MM} flux limiters.

Test cases: Selected cases

Used schemes

Mixture of linear and non-linear schemes:

- Symmetry preserving of 2^{nd} and 6^{th} order {SP2, SP6} Dispersion relation preserving of 4^{th} and 6^{th} order {DRP4, DRP6}, and Moving Least squares of 6^{th} order {MLS3}
- First-order upwind {UPW}, WENO of 3^{rd}, 5^{th} and 7^{th} order {WENO3, WENO5, WENO7}, and Superbee {SB}, Van Leer{VL} and Minmod {MM} flux limiters.

Used meshes

Using 30 one-dimensional stretched meshes:

Test cases: Selected cases

Used schemes

Mixture of linear and non-linear schemes:

- Symmetry preserving of 2^{nd} and 6^{th} order {SP2, SP6} Dispersion relation preserving of 4^{th} and 6^{th} order {DRP4, DRP6}, and Moving Least squares of 6^{th} order {MLS3}
- First-order upwind {UPW}, WENO of 3^{rd}, 5^{th} and 7^{th} order {WENO3, WENO5, WENO7}, and Superbee {SB}, Van Leer{VL} and Minmod {MM} flux limiters.

Used meshes

Using 30 one-dimensional stretched meshes:

- From 0 to 5% stretching ratio

Test cases: Selected cases

Used schemes

Mixture of linear and non-linear schemes:

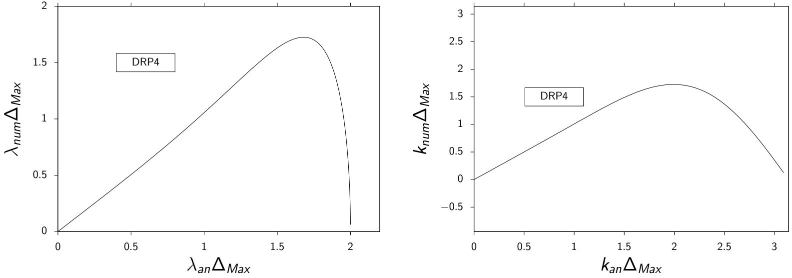
- Symmetry preserving of 2^{nd} and 6^{th} order {SP2, SP6} Dispersion relation preserving of 4^{th} and 6^{th} order {DRP4, DRP6}, and Moving Least squares of 6^{th} order {MLS3}
- First-order upwind {UPW}, WENO of 3^{rd}, 5^{th} and 7^{th} order {WENO3, WENO5, WENO7}, and Superbee {SB}, Van Leer{VL} and Minmod {MM} flux limiters.

Used meshes

Using 30 one-dimensional stretched meshes:

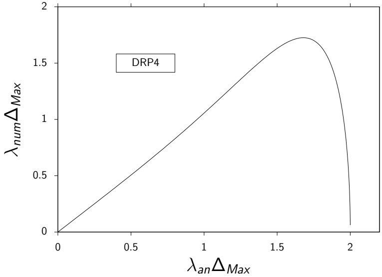
- From 0 to 5% stretching ratio
- Δx_{min} from $1/32$ to $1/512$.

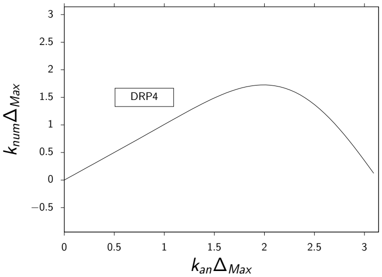
Test cases: Results



Left: Numerical eigenvalues. Right: Numerical wavenumbers. Uniform mesh.

Test cases: Results

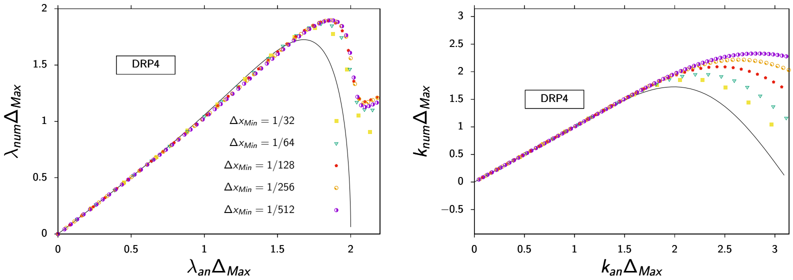




Left: Numerical eigenvalues. Right: Numerical wavenumbers. Uniform mesh.

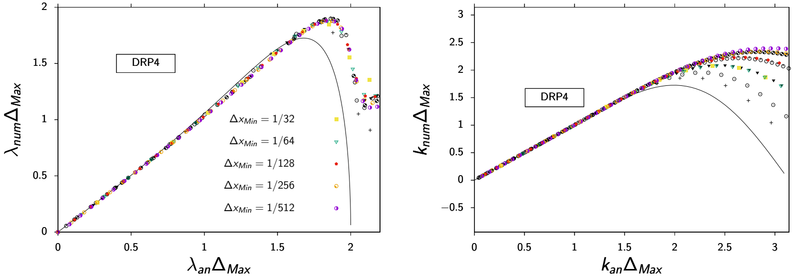
$$\lambda_{an} = \frac{4}{\Delta x} sin^2 \left(k_{an} \Delta x \right)$$

Test cases: Results



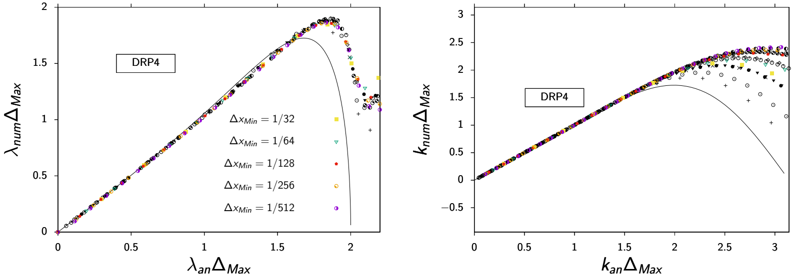
Left: Numerical eigenvalues. Right: Numerical wavenumbers.
Until 1% stretching.

Test cases: Results



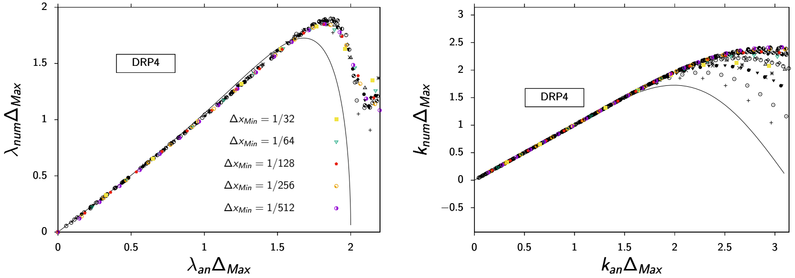
Left: Numerical eigenvalues. Right: Numerical wavenumbers.
Until 2% stretching.

Test cases: Results



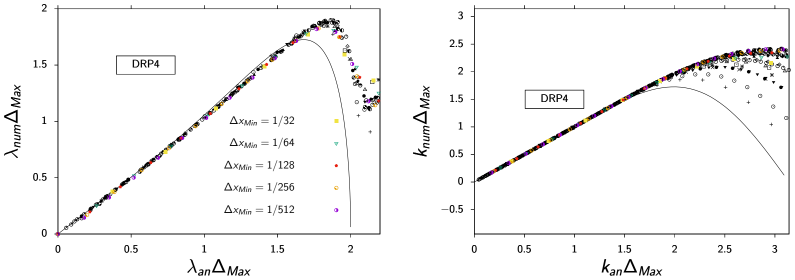
Left: Numerical eigenvalues. Right: Numerical wavenumbers.
Until 3% stretching.

Test cases: Results



Left: Numerical eigenvalues. Right: Numerical wavenumbers.
Until 4% stretching.

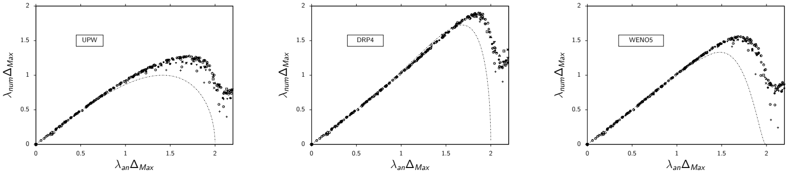
Test cases: Results



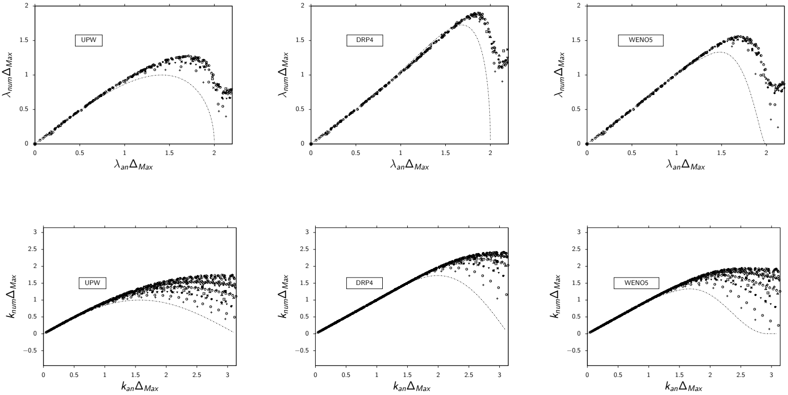
Left: Numerical eigenvalues. Right: Numerical wavenumbers.
Until 5% stretching.

Test cases: Results

Test cases: Results

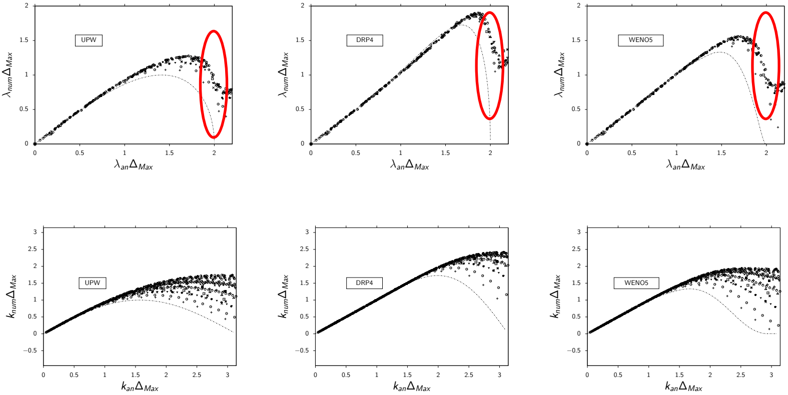


Test cases: Results

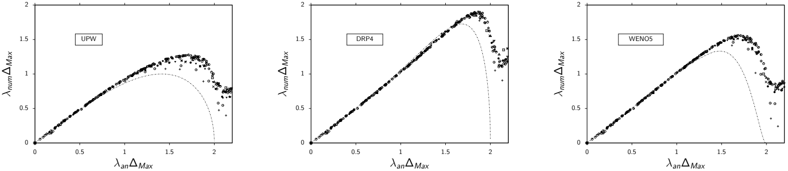


Test cases: Results

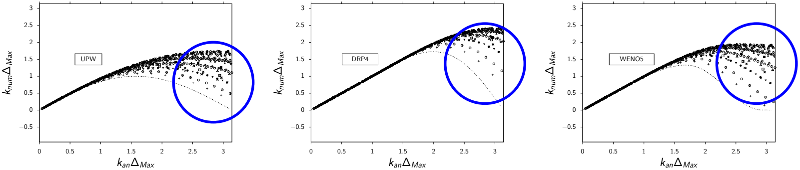
Always at $\lambda \Delta x_{Max} = 2$



Test cases: Results



Not a clear cut-off k



Test cases: Results

Stretch. [%]	*SP2*	*DRP4*	*DRP6*	*SP6*	*MLS3*
0	1	1.7254	1.8368	1.586	1.5615
1	1.203	1.8884	1.9638	1.7688	1.7466
2	1.2396	1.8792	1.9466	1.7704	1.7488
3	1.2501	1.856	1.9239	1.7564	1.7369
4	1.2512	1.8364	1.9018	1.7412	1.7205
5	1.2432	1.8223	1.8748	1.7268	1.708
AVG	1.2374	1.8565	1.9222	1.7527	1.7322

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for second-order symmetry preserving in uniform meshes, linear schemes.

Test cases: Results

Stretch. [%]	*SP2*	*DRP4*	*DRP6*	*SP6*	*MLS3*
0	1	1.7254	1.8368	1.586	1.5615
1	1.203	1.8884	1.9638	1.7688	1.7466
2	1.2396	1.8792	1.9466	1.7704	1.7488
3	1.2501	1.856	1.9239	1.7564	1.7369
4	1.2512	1.8364	1.9018	1.7412	1.7205
5	1.2432	1.8223	1.8748	1.7268	1.708
AVG	1.2374	1.8565	1.9222	1.7527	1.7322

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for second-order symmetry preserving in uniform meshes, linear schemes.

Test cases: Results

Stretch. [%]	*SP*2	*DRP*4	*DRP*6	*SP*6	*MLS*3
0	1	1.7254	1.8368	1.586	1.5615
1	1.203	1.8884	1.9638	1.7688	1.7466
2	1.2396	1.8792	1.9466	1.7704	1.7488
3	1.2501	1.856	1.9239	1.7564	1.7369
4	1.2512	1.8364	1.9018	1.7412	1.7205
5	1.2432	1.8223	1.8748	1.7268	1.708
AVG	1.2374	1.8565	1.9222	1.7527	1.7322

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for second-order symmetry preserving in uniform meshes, linear schemes.

Test cases: Results

Stretch. [%]	*SP2*	*DRP4*	*DRP6*	*SP6*	*MLS3*
0	1	1.7254	1.8368	1.586	1.5615
1	1.203	1.8884	1.9638	1.7688	1.7466
2	1.2396	1.8792	1.9466	1.7704	1.7488
3	1.2501	1.856	1.9239	1.7564	1.7369
4	1.2512	1.8364	1.9018	1.7412	1.7205
5	1.2432	1.8223	1.8748	1.7268	1.708
AVG	1.2374	1.8565	1.9222	1.7527	1.7322

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for second-order symmetry preserving in uniform meshes, linear schemes.

Test cases: Results

Stretch. [%]	*SP*2	*DRP*4	*DRP*6	*SP*6	*MLS*3
0	1	1.7254	1.8368	1.586	1.5615
1	1.203	1.8884	1.9638	1.7688	1.7466
2	1.2396	1.8792	1.9466	1.7704	1.7488
3	1.2501	1.856	1.9239	1.7564	1.7369
4	1.2512	1.8364	1.9018	1.7412	1.7205
5	1.2432	1.8223	1.8748	1.7268	1.708
AVG	1.2374	1.8565	1.9222	1.7527	1.7322

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for second-order symmetry preserving in uniform meshes, linear schemes.

Test cases: Results

Stretch. [%]	*SP2*	*DRP4*	*DRP6*	*SP6*	*MLS3*
0	1	1.7254	1.8368	1.586	1.5615
1	1.203	1.8884	1.9638	1.7688	1.7466
2	1.2396	1.8792	1.9466	1.7704	1.7488
3	1.2501	1.856	1.9239	1.7564	1.7369
4	1.2512	1.8364	1.9018	1.7412	1.7205
5	1.2432	1.8223	1.8748	1.7268	1.708
AVG	1.2374	1.8565	1.9222	1.7527	1.7322

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for second-order symmetry preserving in uniform meshes, linear schemes.

Test cases: Results

Stretch. [%]	*WENO3*	*WENO5*	*WENO7*	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

Test cases: Results

Stretch. [%]	*WENO3*	*WENO5*	*WENO7*	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

Test cases: Results

Stretch. [%]	*WENO3*	*WENO5*	*WENO7*	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

Test cases: Results

Stretch. [%]	*WENO3*	*WENO5*	*WENO7*	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

Test cases: Results

Stretch. [%]	*WENO3*	*WENO5*	*WENO7*	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

Test cases: Results

Stretch. [%]	*WENO*3	*WENO*5	*WENO*7	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

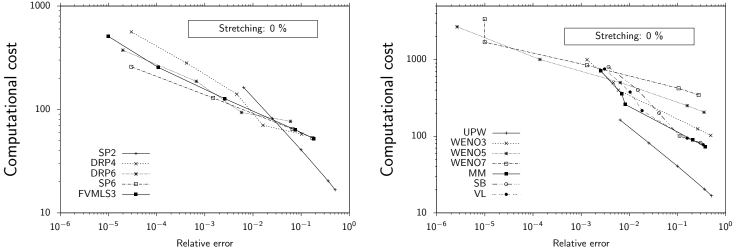
Test cases: Results

Stretch. [%]	*WENO3*	*WENO5*	*WENO7*	*MM*	*SB*	*VL*
0	1.1667	1.3317	1.4529	1.2526	1.4053	1.3237
1	1.3747	1.52	1.6315	1.4713	1.6272	1.543
2	1.4036	1.5402	1.6432	1.4946	1.6444	1.566
3	1.4072	1.5362	1.6377	1.4966	1.637	1.5611
4	1.4047	1.5296	1.6249	1.4973	1.6344	1.5573
5	1.3889	1.5183	1.6132	1.4736	1.6205	1.5374
AVG	1.3958	1.5289	1.6301	1.4867	1.6327	1.553

Table: Non-dimensional maximum eigenvalue normalized respect maximum eigenvalue for first-order upwind in uniform meshes, non-linear schemes.

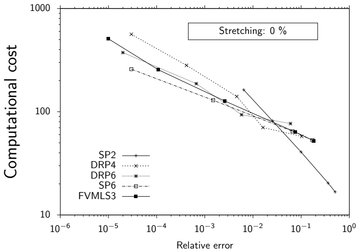
Test cases: Computational cost

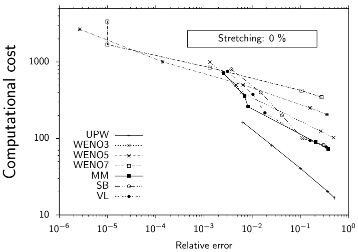
Test cases: Computational cost



Computational cost vs relative error at uniform meshing.

Test cases: Computational cost





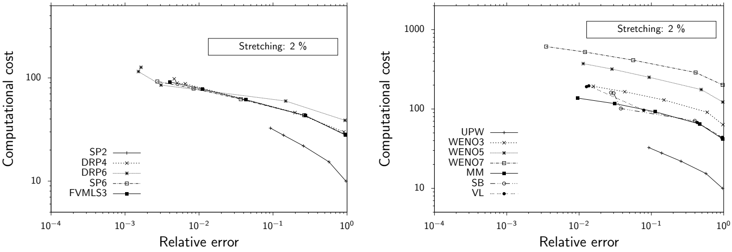
Computational cost vs relative error at uniform meshing.

At uniform meshing...

High-order schemes are more cost-effective. They achieve lesser relative errors than low-order schemes for the same computational cost.

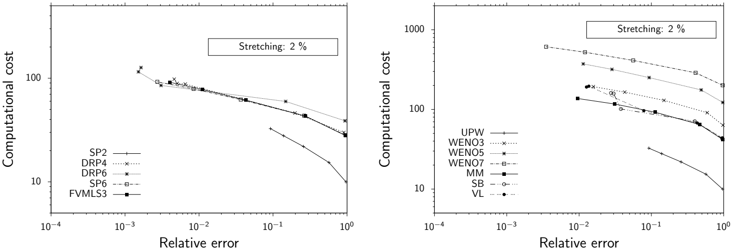
Test cases: Computational cost

Test cases: Computational cost



Computational cost vs relative error at 2% stretching.

Test cases: Computational cost



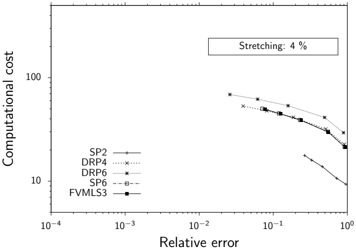
Computational cost vs relative error at 2% stretching.

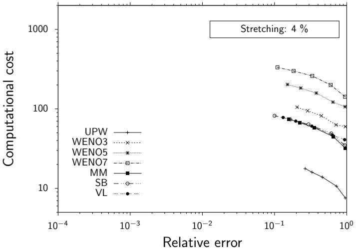
At slightly stretched...

All schemes present a higher relative error: High-order lose two order of magnitude; low-order just one. High-order schemes seem to have lost order of accuracy. For errors in range, low-order are more cost effective.

Test cases: Computational cost

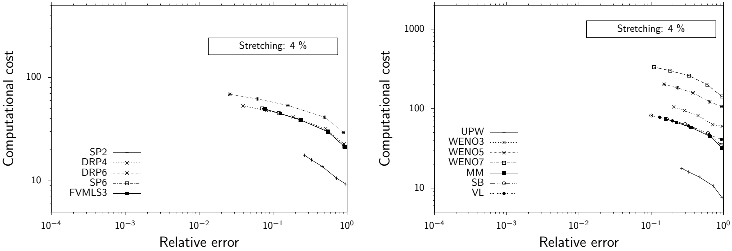
Test cases: Computational cost





Computational cost vs relative error at 4% stretching

Test cases: Computational cost



Computational cost vs relative error at 4% stretching

At highly stretched...

All schemes relative error is higher than 1%. Non-linear schemes do not behave correctly.

Conclusions and further work

Conclusions

Conclusions and further work

Conclusions

A methodology to compute dispersion error in a general framework has been developed.

- No mesh uniformity nor periodic boundary conditions are required. Instead, uses the eigenvectors of the **discrete Laplacian operator**.

Conclusions and further work

Conclusions

A methodology to compute dispersion error in a general framework has been developed.

- No mesh uniformity nor periodic boundary conditions are required. Instead, uses the eigenvectors of the **discrete Laplacian operator**.

A new numerical relation between expected and recovered eigenvalues has been found for studied schemes.

- Stretched meshes, independently on the stretching factor used on the study range, colapse onto the **same plot**, which is not the same that if uniform meshes are used.

Conclusions and further work

Conclusions

Conclusions and further work

Conclusions

Maximum allowed eigenvalue with minimal dispersion directly related to maximum mesh size ($\lambda \Delta x_{Max} < 2$).

- A maximum allowed frequency related to mesh size does not appear. Instead, results are mesh dependent.

Conclusions and further work

Conclusions

Maximum allowed eigenvalue with minimal dispersion directly related to maximum mesh size ($\lambda \Delta x_{Max} < 2$).

- A maximum allowed frequency related to mesh size does not appear. Instead, results are mesh dependent.

Low-order schemes are less affected with mesh stretching than high-order schemes.

- High-order schemes **loss order of accuracy** whereas low-order seem to keep it.

Conclusions and further work

Conclusions

Maximum allowed eigenvalue with minimal dispersion directly related to maximum mesh size ($\lambda \Delta x_{Max} < 2$).

- A maximum allowed frequency related to mesh size does not appear. Instead, results are mesh dependent.

Low-order schemes are less affected with mesh stretching than high-order schemes.

- High-order schemes **loss order of accuracy** whereas low-order seem to keep it.

Further work

Conclusions and further work

Conclusions

Maximum allowed eigenvalue with minimal dispersion directly related to maximum mesh size ($\lambda \Delta x_{Max} < 2$).

- A maximum allowed frequency related to mesh size does not appear. Instead, results are mesh dependent.

Low-order schemes are less affected with mesh stretching than high-order schemes.

- High-order schemes **loss order of accuracy** whereas low-order seem to keep it.

Further work

Propose a meshing technique leading to dispersion reduction.

- Select the **most appropiate scheme for a given mesh**.

Thanks for your attention