LOW ORDER SIMULATION METHODS

FOR AEROSPACE ENGINEERING

K. Morgan

Zienkiewicz Centre for Computational Engineering

College of Engineering

Swansea University

Swansea

Swansea University
Prifysgol Abertawe

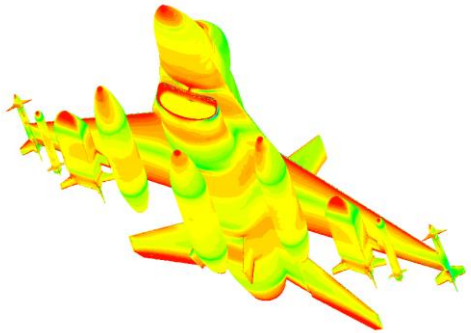
4th ACME School

Swansea April 2015

ZCCE

OUTLINE

A traditional low order method for aerodynamics



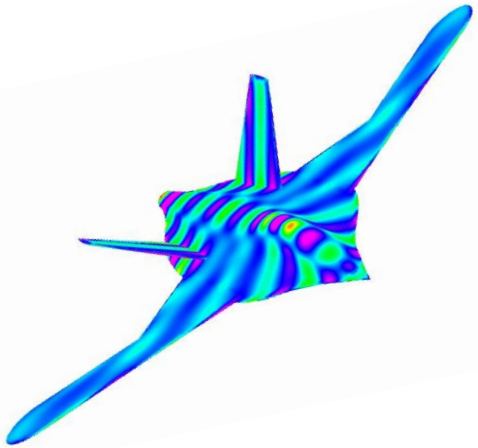
1D scalar conservation equation

1D Euler system of conservation laws

3D Euler system of conservation laws

Example of industrial impact for steady flows

A co–volume method for electromagnetics



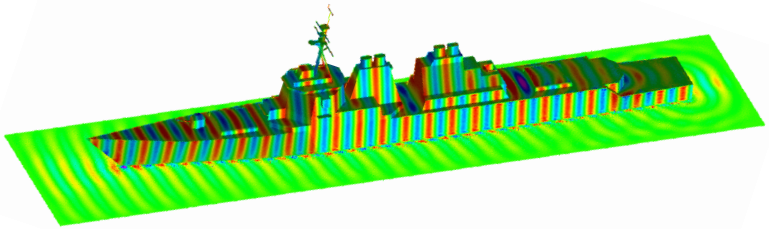
2D Maxwell equations

2D mesh requirements

Examples

3D Maxwell equations

Examples



Conclusions

Classical Formulation

Find $U(x_1,t)$ such that

$$\frac{\partial U}{\partial t} + \frac{\partial F^1}{\partial x_1} = 0 \qquad x_1 \in \Omega; t > 0 \qquad U(x_1, 0) = U^0(x_1) \qquad x_1 \in \Omega$$

where $F^1 = F^1(U)$

Variational Formulation

Define

$$\mathcal{T} = \{U(x_1, t) | U(x_1, 0) = U^0(x_1) \text{ for } x_1 \in \Omega\}$$

$$\mathcal{W} = \{W(x_1)\}$$

Find $U(x_1, t)$ in $\mathcal{T}$ such that

$$\int_\Omega W\left(\frac{\partial U}{\partial t} + \frac{\partial F^1}{\partial x_1}\right) \mathrm{d}x_1 = 0$$

for all W in $\mathcal{W}$ and for all $t>0$

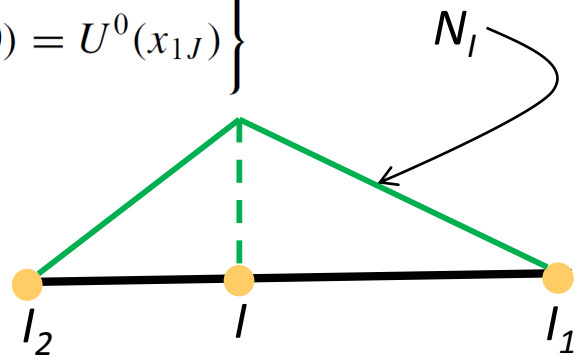
Galerkin Approximation

Construct a mesh of linear finite elements with *p* nodes on Ω

Define the finite dimensional subspaces

$$\mathcal{T}^{(p)} = \left\{ U^{(p)}(x_1,t) | U^{(p)}(x_1,t) = \sum_{J=1}^{p} U_J(t) N_J(x_1);\ U_J(0) = U^0(x_{1J}) \right\}$$

$$\mathcal{W}^{(p)} = \left\{ W(x_1) | W(x_1) = \sum_{J=1}^{p} a_J N_J(x_1) \right\}$$



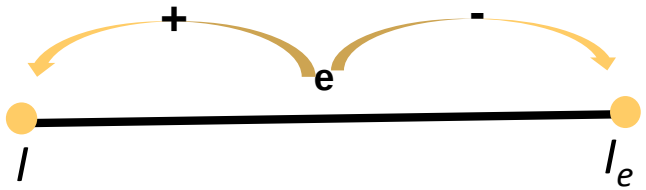
The Galerkin approximation is defined by the requirement

$$\int_\Omega N_I \frac{\partial U^{(p)}}{\partial t}\, \mathrm{d}x_1 = -\int_\Omega N_I \frac{\partial F^1}{\partial x_1}\, \mathrm{d}x_1 \qquad I=1,2,3,\ldots,p$$

This implies that

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}U_J}{\mathrm{d}t} = \sum_{e=1}^{2} F^1_{II_e} \qquad \text{where} \qquad F^1_{II_e} = C^1_{II_e}(F^1_I + F^1_{I_e})$$

Here $C^1_{II_e}$ is a weight that depends upon the geometry of the mesh



Stabilisation

The Galerkin method applied to the spatial derivative leads to an approximation that allows the appearance of spurious modes e.g. when $F^1=A^1U$ and A^1 constant

$$\sum_{J=1}^{p} M_{IJ}\frac{\mathrm{d}U_J}{\mathrm{d}t} = A^1\sum_{e=1}^{2} C^1_{II_e}(U_I + U_{I_e})$$

Stabilisation can be achieved by the addition of some form of diffusion

In finite difference methods this can be accomplished by discretising directly a modified equation

In the finite element Galerkin method the actual flux function is replaced by the consistent numerical flux function

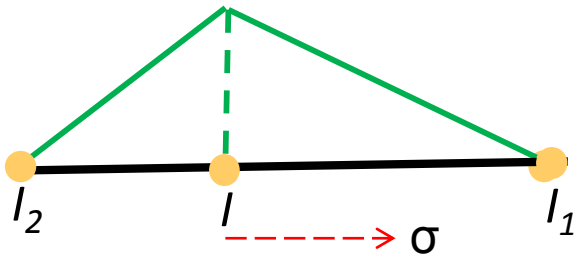
Peraire (1993)

$$\mathcal{F}_{II_e} = C^1_{II_e}(F^1_I + F^1_{I_e}) + \varepsilon_4|\lambda_{II_e}|d_{II_e}$$

$$d_{II_e} = U_{I_e} - U_I - \left(h\frac{\partial U}{\partial \sigma}\right)_{II_e} \qquad \lambda_{II_e} = C^1_{II_e}A^1_{II_e} \qquad A^1 = \frac{\mathrm{d}F^1}{\mathrm{d}U}$$

Nodal values of the solution gradient are calculated as

$$M_I\left.\frac{\partial U}{\partial x_1}\right|_I = -\sum_{e=1}^{2} C^1_{II_e}(U_I + U_{I_e})$$



Discontinuity Capturing

Discontinuities may be captured by the explicit addition of a suitably scaled second order diffusion operator

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}U_J}{\mathrm{d}t} = \sum_{e=1}^{2} \tilde{\mathcal{F}}_{II_e}$$

The modified numerical flux function is defined as

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} |\lambda_{II_e}| (U_{I_e} - U_I)$$

where the discontinuity sensor is

$$\chi_I = \frac{\left| \sum_{e=1}^{2} (U_{I_e} - U_I) \right|}{\sum_{e=1}^{2} (|U_{I_e}| + |U_I|)}$$

Lőhner (1985)

The full form of the numerical flux function is

$$\tilde{\mathcal{F}}_{II_e} = C^1_{II_e} (F^1_I + F^1_{I_e}) + |\lambda_{II_e}| \{ \varepsilon_2 \chi_{II_e} (U_{I_e} - U_I) + \varepsilon_4 d_{II_e} \}$$

Only discontinuity capturing diffusion should be added in the regions of strong gradients

Euler Equations

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}^1}{\partial x_1} = 0$$

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho \epsilon \end{bmatrix} \qquad \boldsymbol{F}^1 = \boldsymbol{F}^1(\boldsymbol{U}) = \begin{bmatrix} \rho u_1 \\ \rho u_1^2 + p \\ (\rho \epsilon + p) u_1 \end{bmatrix} \qquad p = (\gamma - 1)\rho(\epsilon - 0.5 u_1^2)$$

$$\mathbf{A}^1 = \mathbf{A}^1(\mathbf{U}) = \frac{\mathrm{d}\boldsymbol{F}^1}{\mathrm{d}\boldsymbol{U}}$$

As in the case of the scalar equation, application of the Galerkin method and the explicit addition of diffusion, for stabilisation and discontinuity capturing, leads to

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\boldsymbol{U}_J}{\mathrm{d}t} = \sum_{e=1}^{2} \tilde{\mathcal{F}}_{II_e}$$

where

$$\mathcal{F}_{II_e} = C^1_{II_e}(\boldsymbol{F}^1_I + \boldsymbol{F}^1_{I_e}) + \epsilon_4 |\lambda_{II_e}| \boldsymbol{d}_{II_e}$$

$$\lambda_{II_e} = C^1_{II_e}(|u_{1II_e}| + c_{II_e}) \qquad \boldsymbol{d}_{II_e} = \boldsymbol{U}_{I_e} - \boldsymbol{U}_I - \left(h \frac{\partial \boldsymbol{U}}{\partial \sigma}\right)_{II_e}$$

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} |\lambda_{II_e}| (\boldsymbol{U}_{I_e} - \boldsymbol{U}_I)$$

For the equation system, the discontinuity sensor is based upon a key variable, e.g. the pressure

Euler Equations

$$\frac{\partial \boldsymbol{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \boldsymbol{F}^j}{\partial x_j} = \boldsymbol{0}$$

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho \epsilon \end{bmatrix} \qquad \boldsymbol{F}^j = \begin{bmatrix} \rho u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ \rho u_3 u_j + p\delta_{3j} \\ (\rho \epsilon + p) u_j \end{bmatrix} \qquad p = (\gamma - 1)\rho[\epsilon - 0.5(u_1^2 + u_2^2 + u_3^2)]$$

$$\mathbf{A}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \mathbf{A}^j_{II_e} \qquad \mathbf{A}^j_{II_e} = \left. \frac{\mathrm{d}\boldsymbol{F}^j}{\mathrm{d}\boldsymbol{U}} \right|_{II_e}$$

As in the case of the 1D system, application of the Galerkin method and the explicit addition of diffusion leads, on a linear tetrahedral mesh, to the edge based formulation

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\boldsymbol{U}_J}{\mathrm{d}t} = \sum_{e=1}^{\mu_I} \tilde{\mathcal{F}}_{II_e}$$

Formaggia (1988)

Peraire (1993)

Lyra (1995)

Sørensen (2002)

where

$$\mathcal{F}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} (\boldsymbol{F}^j_I + \boldsymbol{F}^j_{I_e}) + \varepsilon_4 |\lambda_{II_e}| \boldsymbol{d}_{II_e}$$

$$|\lambda_{II_e}| = \left| \sum_{j=1}^{3} C^j_{II_e} u_{jII_e} \right| + c_{II_e} \left[\sum_{j=1}^{3} \left(C^j_{II_e} \right)^2 \right]^{1/2} \qquad \boldsymbol{d}_{II_e} = \boldsymbol{U}_{I_e} - \boldsymbol{U}_I - \left(h \frac{\partial \boldsymbol{U}}{\partial \sigma} \right)_{II_e}$$

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} |\lambda_{II_e}| (\boldsymbol{U}_{I_e} - \boldsymbol{U}_I)$$

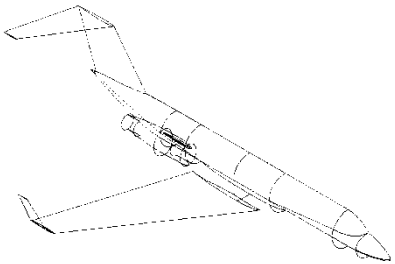
DEFINITION OF THE GEOMETRY AND THE MESH SIZE

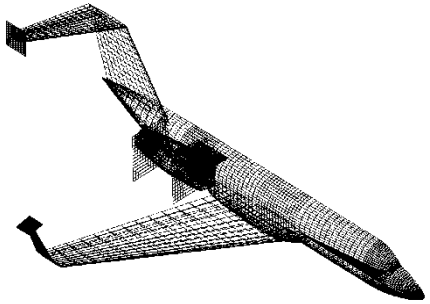
Geometry Definition

surface components: bi-cubic patches, NURBS

curve components: cubic splines, NURBS

Peiró (1989)





Mesh Size Definition

background mesh

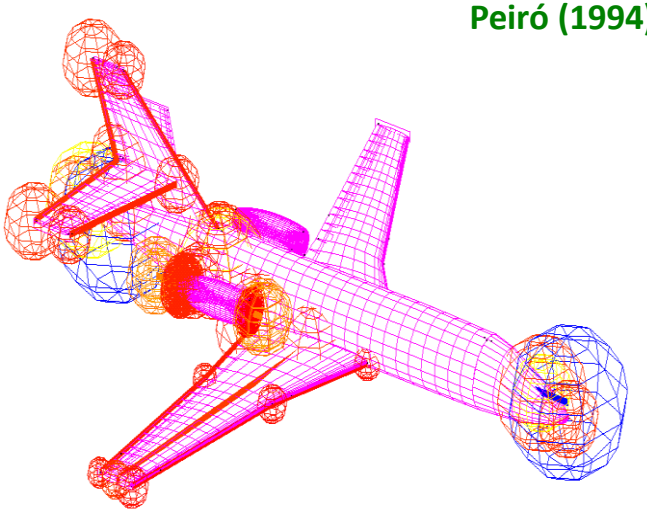
Peraire (1987)

point, line, circular and planar sources

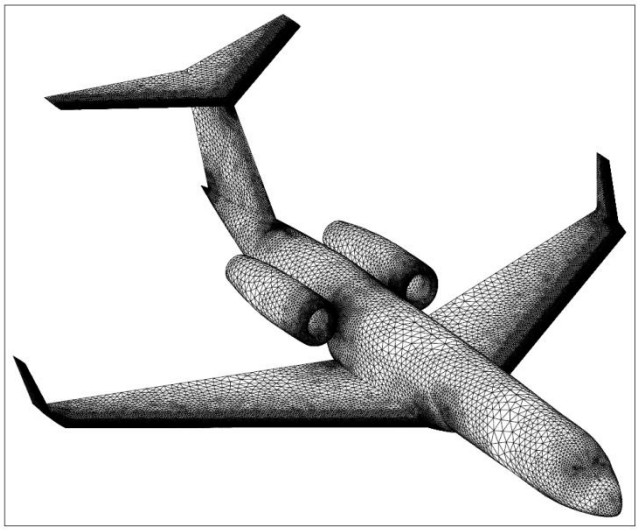
Peiró (1994)

curvature controlled

Tremel (2005)



UNSTRUCTURED MESH GENERATION

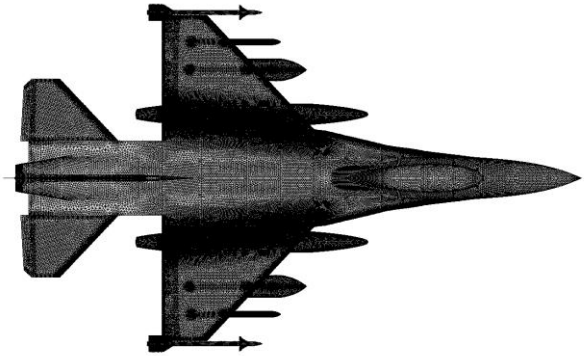


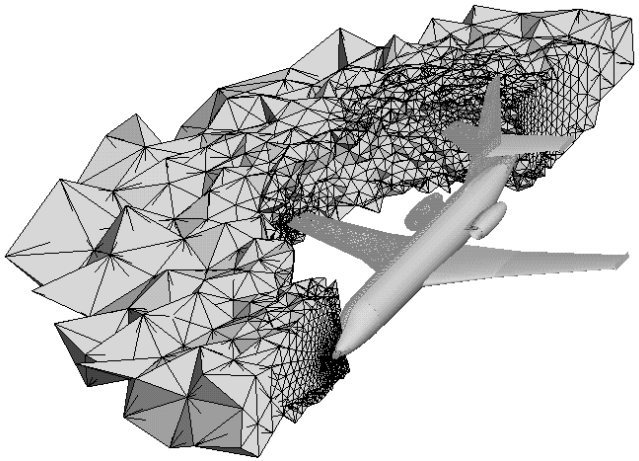
Surface Mesh Generation

Discretisation into triangles using an advancing front method

Peraire (1985)

Peiró (1989)





Volume Mesh Generation

Discretisation into isotropic tetrahedra using a Delaunay method with automatic point creation

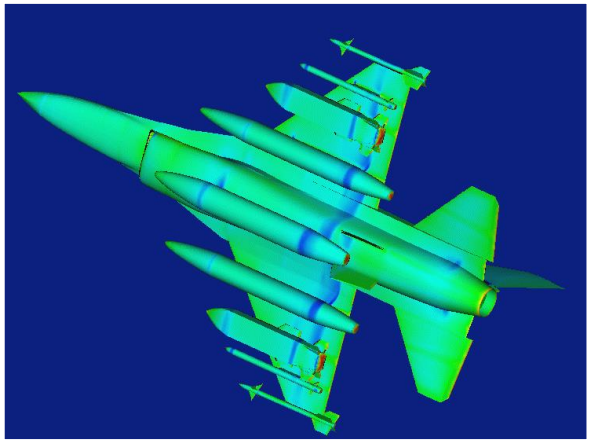
Requires 100Mb/10^6 elements

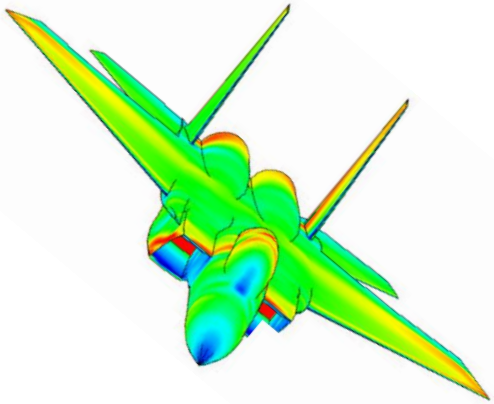
Generates > 0.5M elements/min on a PC

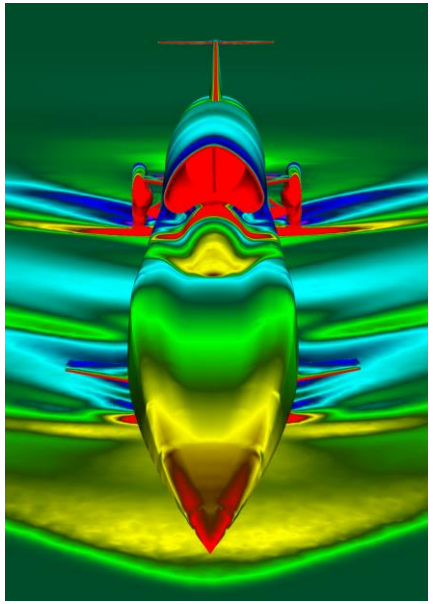
Hassan (1994)

Lőhner (1987), Peraire (1987), Weatherill (1986)

AERODYNAMICS WITH FLITE3D



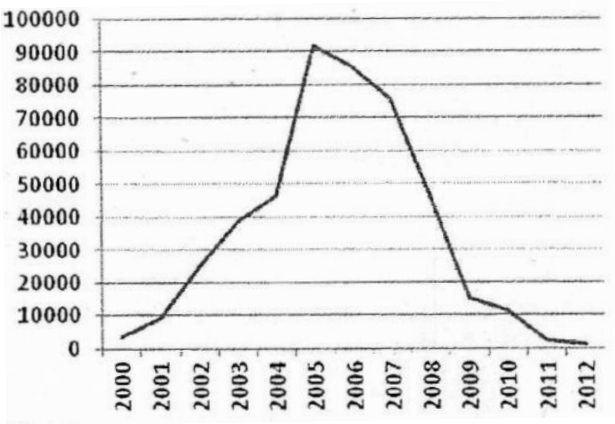




For the A380, stringent design targets were set and 100s of different wing geometries were analysed

Final design could not have been achieved by wind tunnel testing alone

Frank Ogilvie



FLITE3D runs

2D CO–VOLUME ALGORITHM FOR CEM

Faraday's Law

$$\frac{\partial}{\partial t}\int_{\mathcal{S}} \mu \mathbf{H}\cdot \mathrm{d}\boldsymbol{\mathcal{S}} = -\int_{\mathcal{C}} \mathbf{E}\cdot \mathrm{ds}$$

Ampère's Law

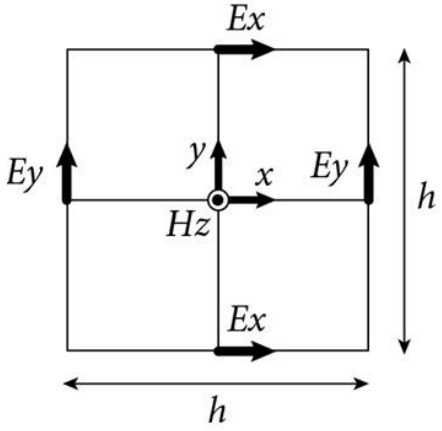
$$\frac{\partial}{\partial t}\int_{\mathcal{S}} \epsilon \mathbf{E}\cdot \mathrm{d}\boldsymbol{\mathcal{S}} = \int_{\mathcal{C}} \mathbf{H}\cdot \mathrm{ds}$$

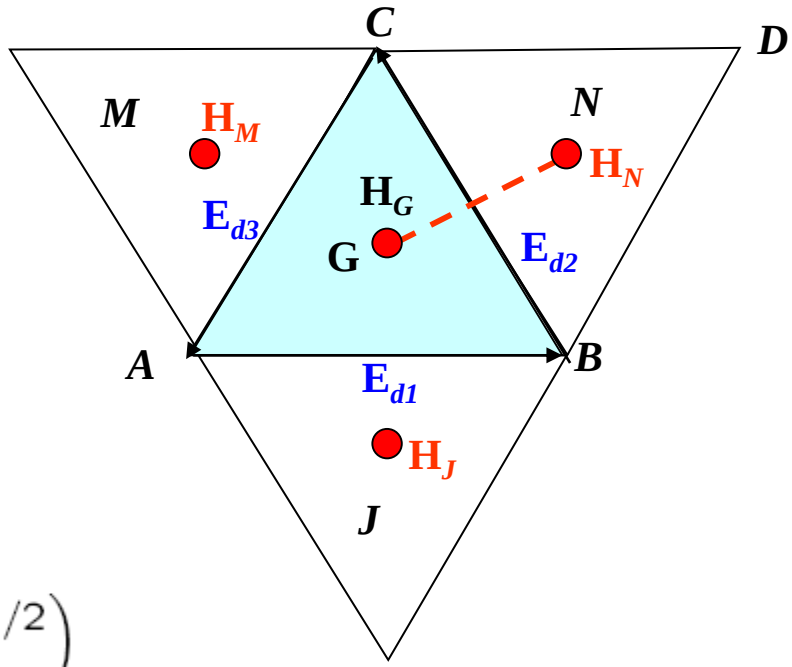
Use the Delaunay-Voronoi dual diagram

Discretised TE form

$$H_G^{n+1/2} = H_G^{n-1/2} - \frac{\Delta t}{\mu A_G}\sum_{j=1}^{3} E_{d_j}^n \ell_{d_j}$$

$$E_{d_2}^{n+1} = E_{d_2}^n + \frac{\Delta t}{\epsilon \ell_{v_{GN}}}\left(H_G^{n+1/2} - H_N^{n+1/2}\right)$$



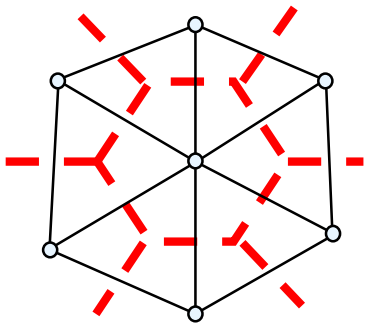


Time step size

$$\Delta t \propto \min\left(\ell_d, \ell_v\right)$$

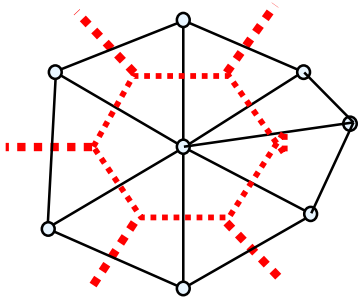
2D MESH REQUIREMENTS

Ideal mesh consists of equilateral triangles $\ell^V = \frac{\ell^D}{\sqrt{3}}$



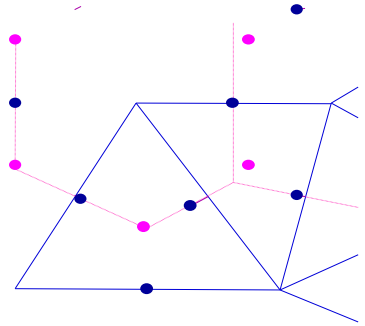
For a general mesh

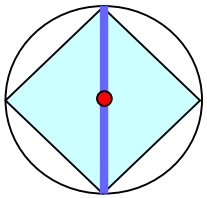
—each circumcentre should lie inside its associated element

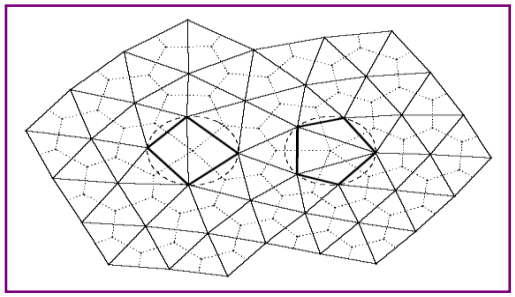


—deviation of the circumcentre from the barycentre should be minimised

—deviation of the midpoint of each Voronoi edge from its intersection with its Delaunay edge should be minimised



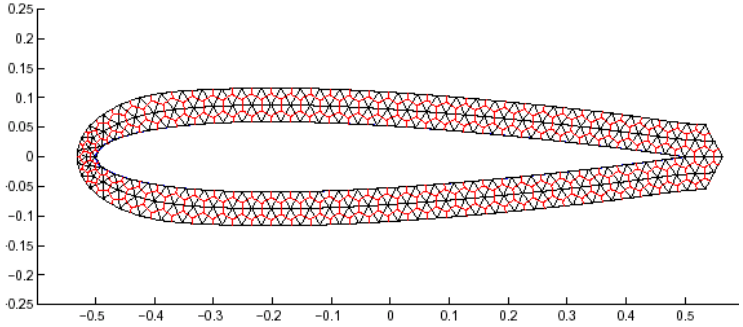




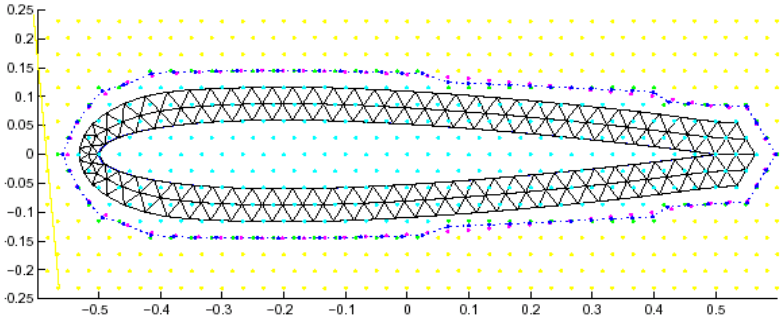
Zero length Voronoi edges may be removed by the merging of cells

MESH GENERATION BY THE STITCHING METHOD

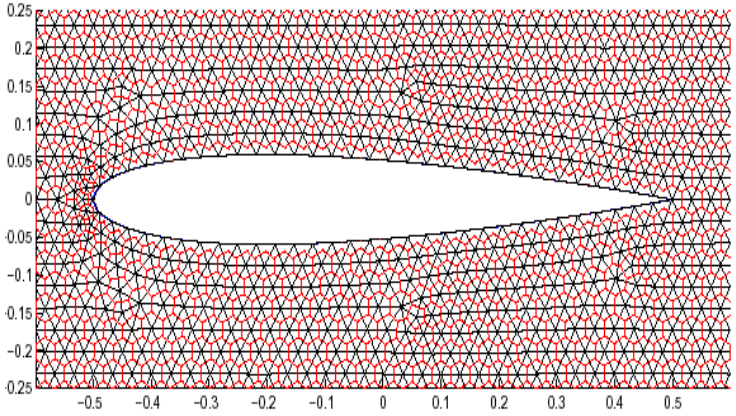
To avoid distorted elements near boundaries, one or more layers of elements are generated first near the boundaries using an advancing front method



A mesh with the required size distribution is generated to cover the whole domain—for CEM this is usually the ideal mesh shown here



The two meshes are stitched together



Mesh quality enhancement procedures are employed

Sazonov (2006)

Lloyd's Algorithm

Centroidal Voronoi tesellation (CVT) scheme that modifies a mesh by moving nodes to the mass centroids of the corresponding Voronoi cells; improves the mesh but a significant percentage of bad elements remain

Wang (2006)

Gradient Free Mesh Optimisation

The number of degrees of freedom representing the coordinates in a mesh is *nnode*ndim*

Most gradient free algorithms are only demonstrated to work successfully for up to around 300 degrees of freedom

Transform the global problem into *nnode* local problems of size *ndim*

$$\mathscr{F}(k) = \sum_{i=1}^{E} W_i \frac{\| \mathbf{C}_i - \mathbf{V}_i \|}{\delta_i}$$

k : node number E number of elements containing node *k* $\mathbf{C}_i$ centroid element ***i***

$\mathbf{V}_i$ Voronoi vertex element *i* δ_i mean edge length element ***i*** W_i =1 if *i* bad; 0 otherwise

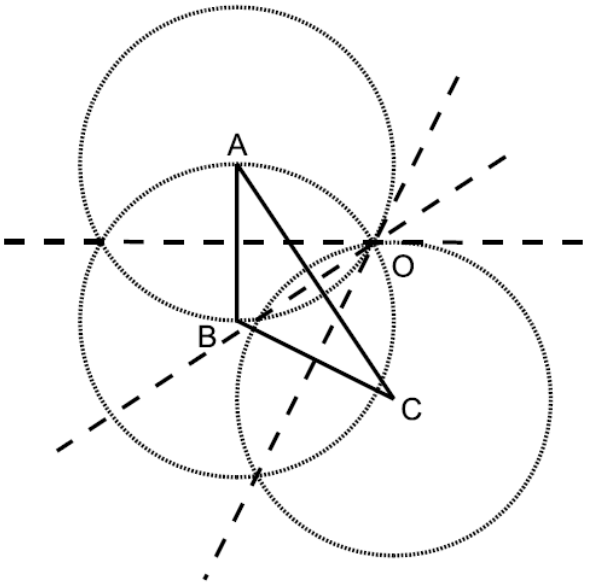
Each node in turn is moved to the optimum position found using a fixed number of generations of a modified Cuckoo search algorithm

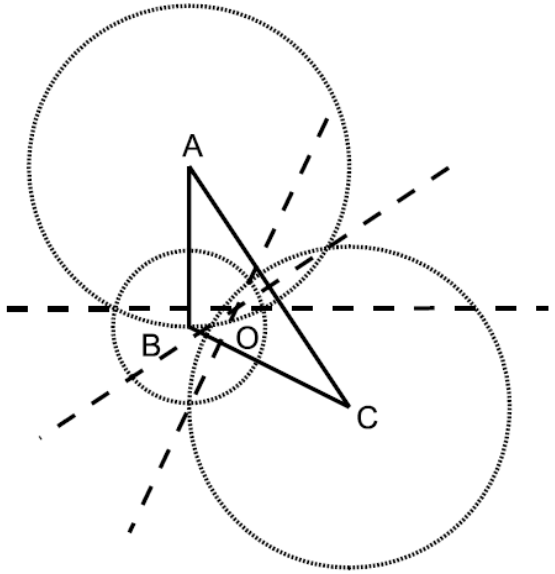
Walton (2011)

LOCAL WEIGHT OPTIMISATION

For bad elements that remain, the weighted Voronoi diagram concept is employed in which vertices are moved while retaining edge orthogonality

Common chords of circles at each vertex with radius equal to that of the circumcircle intersect at the Voronoi vertex





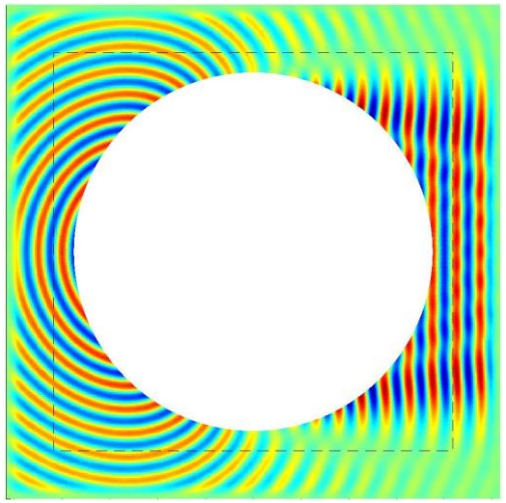
Xie (2011)

Reducing the radius at one vertex moves the weighted Voronoi vertex inside the bad element while maintaining orthogonality

Vertex weights are optimised using the modified cuckoo search process

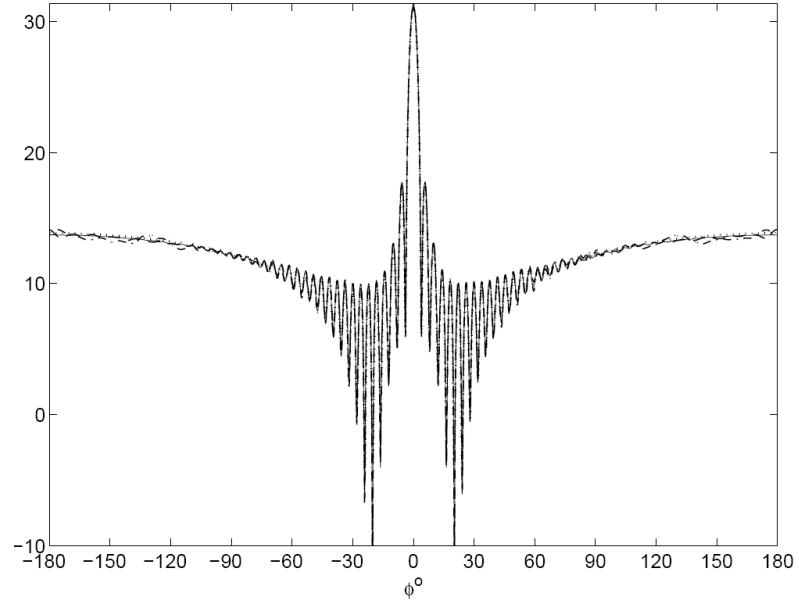
2D EM WAVE SCATTERING

15λ PEC Cylinder

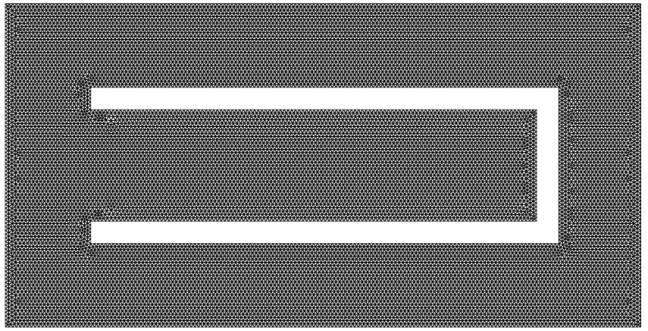


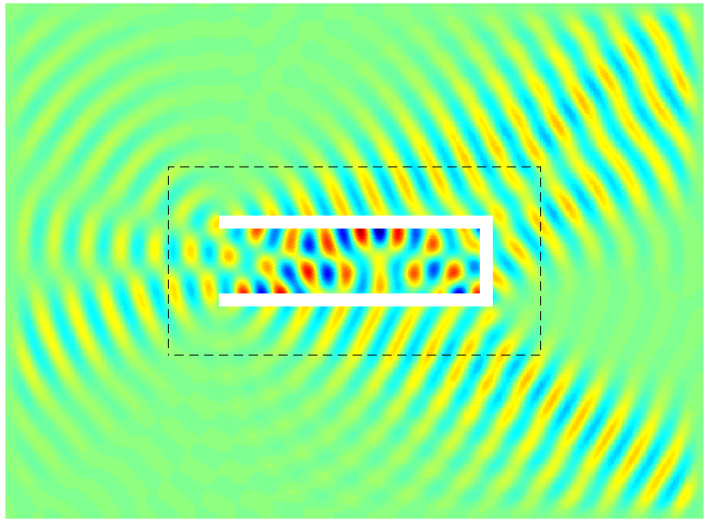
mesh resolution λ/15

method	*spc*	*cpu (s)*
co–volume	55	9.9
FETD	59	334



8λ PEC Cavity



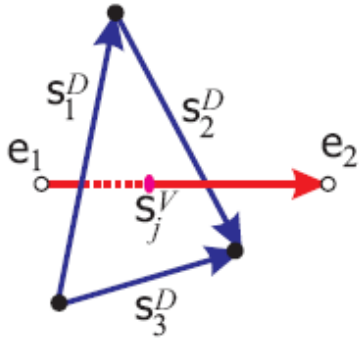


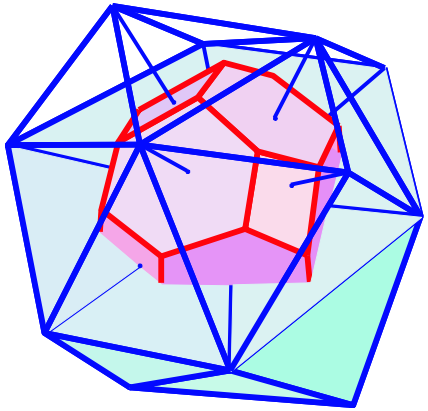
method	*spc*	*cpu (s)*
co–volume	57	20.5
FETD	59	1327.6

3D CO–VOLUME ALGORITHM FOR CEM

Faraday's Law

$$\frac{\partial}{\partial t}\int_{\mathcal{S}} \mu \mathbf{H}\cdot \mathrm{d}\boldsymbol{\mathcal{S}} = -\int_{\mathcal{C}} \mathbf{E}\cdot \mathrm{ds}$$



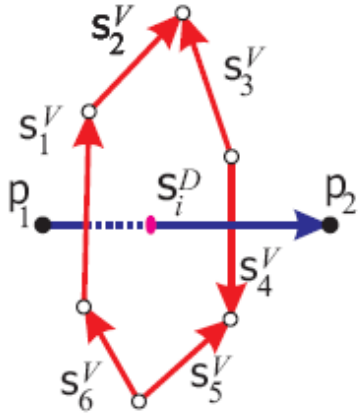


$$\frac{\partial H_j}{\partial t} A_j^D = -\frac{1}{\mu}\sum_{k=1}^{3} E_k\, s_k^D$$

$$H_j^{n+1/2} = H_j^{n-1/2} - \frac{\Delta t}{\mu A_j^D}\sum_{k=1}^{3} E_k^n\, s_k^D$$

Ampère's Law

$$\frac{\partial}{\partial t}\int_{\mathcal{S}} \epsilon \mathbf{E}\cdot \mathrm{d}\boldsymbol{\mathcal{S}} = \int_{\mathcal{C}} \mathbf{H}\cdot \mathrm{ds}$$

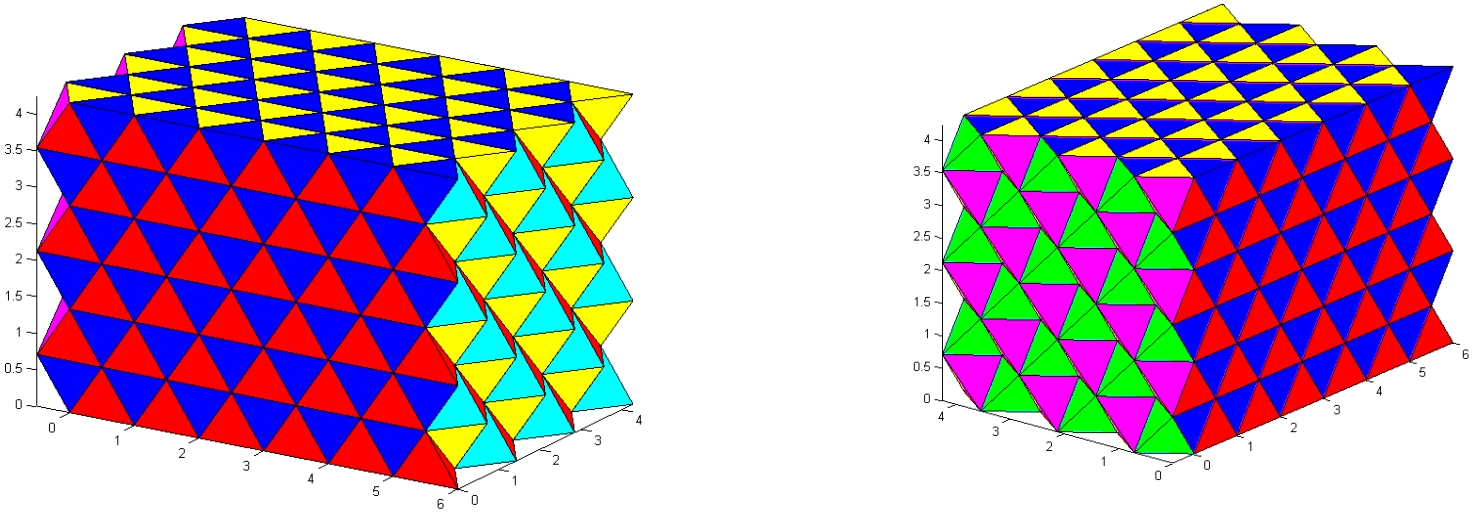


$$\frac{\partial E_i}{\partial t} A_i^V = \frac{1}{\varepsilon}\sum_{k=1}^{M_i^V} H_k\, s_k^V$$

$$E_i^{n+1} = E_i^n + \frac{\Delta t}{\varepsilon A_i^V}\sum_{k=1}^{M_i^V} H_k^{n+1/2}\, s_k^V$$

IDEAL 3D MESH

The ideal mesh consists of non−perfect tetrahedra



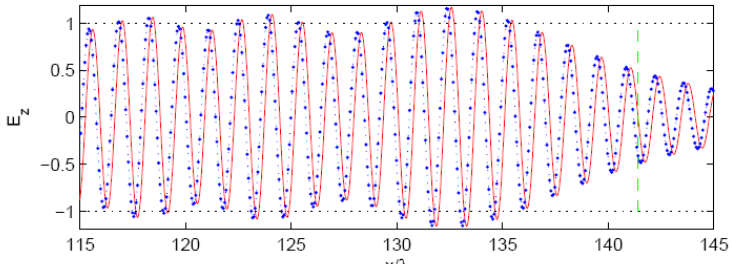
Each face is an isosceles triangle with one side of length ℓ^D_{long} and two shorter sides of length

$$\ell^D_{short} = (\sqrt{3}/2)\ell^D_{long}$$

Sazonov (2006)

This configuration maximises the Voronoi edge length and all Voronoi edges have the same length

$$\ell^V \approx 0.3496\ell^D_{long}$$



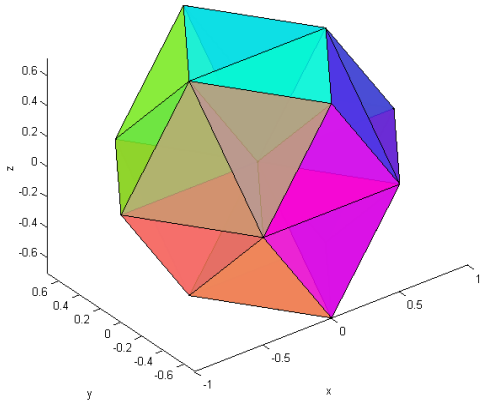
Scattering by a 2 λ PEC Polyhedron

Mesh resolution is λ/15 , λ/8 for 3D co–volume method

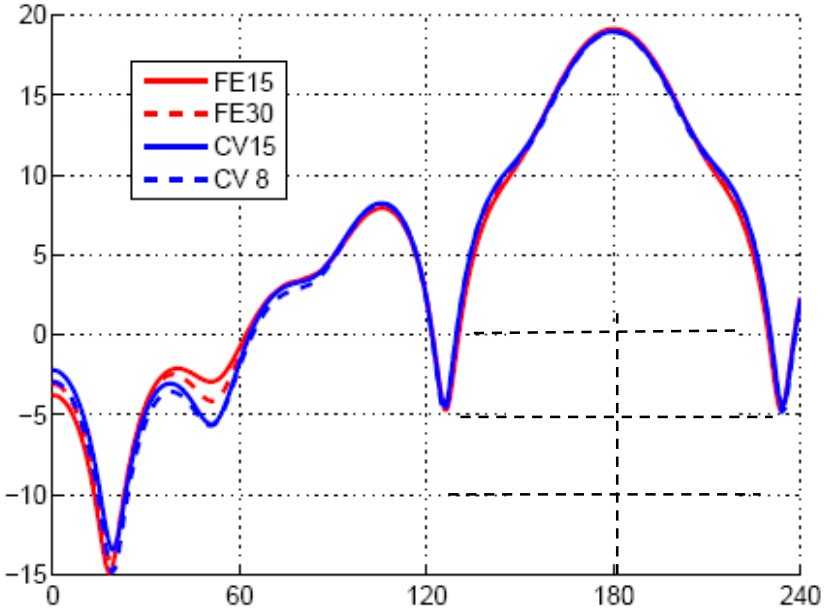
Mesh resolution is λ/30 , λ/15 for FETD

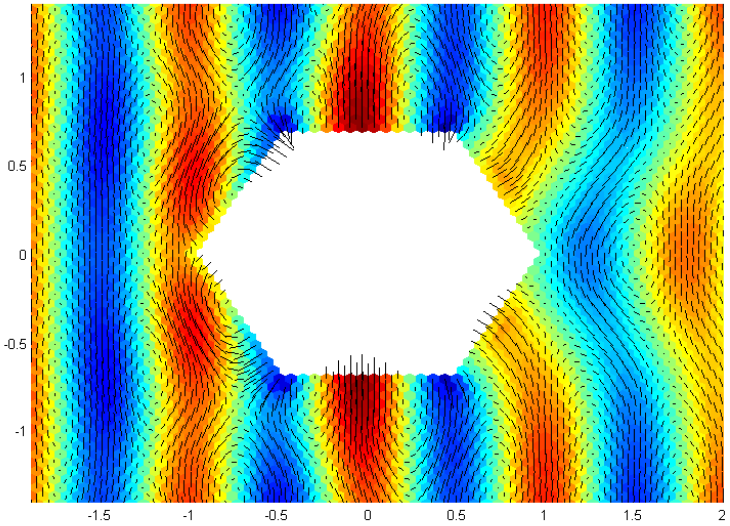
15λ grid: 5 349 240 elements 915 540 points

3D co–volume: 300 CPU sec FETD: 4000 CPU sec.



Sazonov (2006)

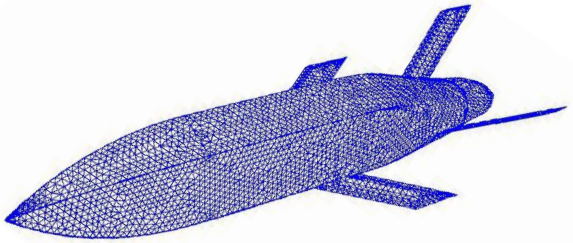


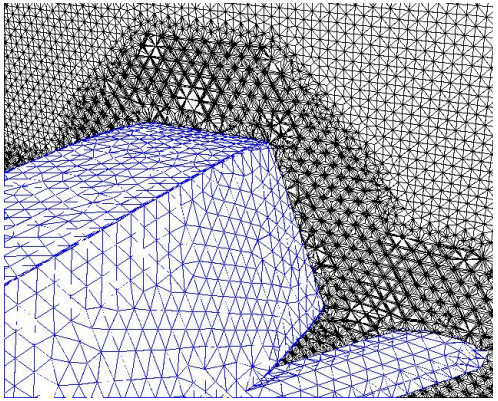


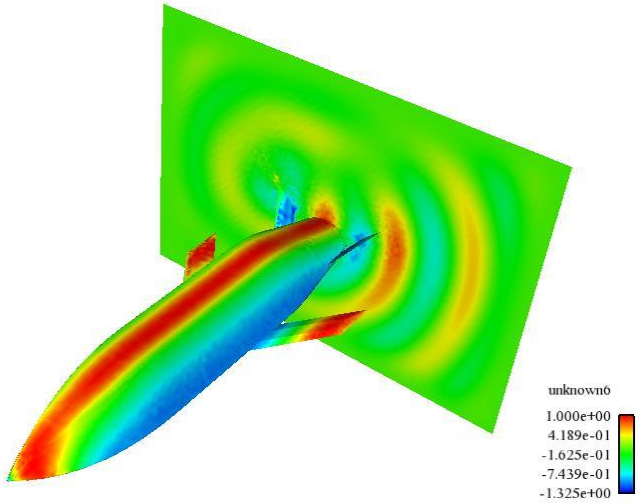
Ez (colour) and (Ex,Ez) (black) in the (x,z) plane

3D CEM EXAMPLE

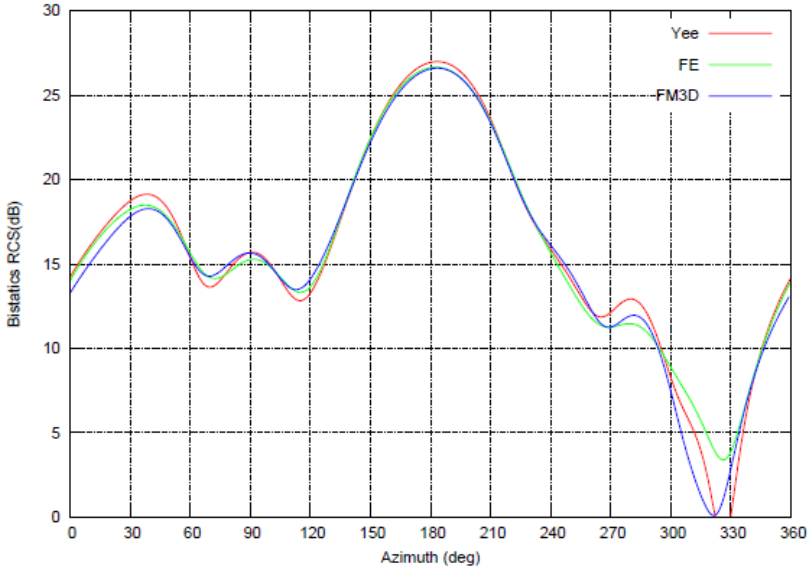
Scattering by 6λ PEC Finned Body







Xie (2011)



	co–volume	FETD
points	~900K	~900K
cells	~1.2M	
hex	~800K	
steps/cyc	2 103	3 248
cpu/cyc	34.1m	282.8m

CONCLUSIONS

Low Order Method for Aerodynamics

Linear finite element method for the compressible Euler equations

Efficient edge based formulation

Automatic unstructured mesh generation

Practical example of industrial application of the approach

Co–Volume Method for Electromagnetics

Co–volume method for the Maxwell equations

Delaunay and Voronoi dual meshes **Gansen (2015)**

Co–volume mesh requirements

Mesh optimisation procedures **Walton (2015)**

Possible extensions to fluid mechanics? **Jones (2015)**