On the Time Integration of Maxwell's Equations

Jan Verwer

Centrum Wiskunde & Informatica

http://homepages.cwi.nl/~janv/

NUMDIFF 12, Halle, 14 -18 September 2009

The talk is based on joint research with Mike Botchev:

- *J.G. Verwer and M.A. Botchev, Unconditionally stable integration of Maxwell's equations, Linear Algebra and its Applications 431, pp. 300-317 (2009)*

- *M.A. Botchev and J.G. Verwer, Numerical integration of damped Maxwell equations, SIAM J. Sci. Comput. 31, pp. 1322-1346 (2009)*

The talk is about the oldie: explicit or implicit time stepping?

Why examining implicit time stepping for a wave equation like Maxwell's?

- *Any explicit method is conditionally stable, that is, the step size is constrained to avoid uncontrolled error growth.*

- *Unnecessary step constraints may arise from locally refined or unstructured grids.*

- *In literature, the ADI approach has already been proven useful. However, ADI requires a Cartesian grid layout.*

Outline

(1) Maxwell’s equations

(3) A special case: the exponential operator

(5) A 2^{nd} - order exponential integrator (EK2)

(4) A 2^{nd} - order explicit integrator (CO2)

(5) A comparison between EK2 and CO2

(1) Maxwell's equations

Maxwell's equations

$$\begin{cases} \mu \, \partial_t H & = & -\nabla \times E \\ \epsilon \, \partial_t E & = & \nabla \times H - \sigma E - J \end{cases}$$

H magnetic field

E electric field

J electric current

σE is a damping conduction term

Maxwell's equations

In 3D with scalar coefficients:

$$\mu \frac{\partial H^x}{\partial t} = \frac{\partial E^y}{\partial z} - \frac{\partial E^z}{\partial y}$$

$$\mu \frac{\partial H^y}{\partial t} = \frac{\partial E^z}{\partial x} - \frac{\partial E^x}{\partial z}$$

$$\mu \frac{\partial H^z}{\partial t} = \frac{\partial E^x}{\partial y} - \frac{\partial E^y}{\partial x}$$

$$\epsilon \frac{\partial E^x}{\partial t} = \frac{\partial H^z}{\partial y} - \frac{\partial H^y}{\partial z} - \sigma E^x - J_E^x$$

$$\epsilon \frac{\partial E^y}{\partial t} = \frac{\partial H^x}{\partial z} - \frac{\partial H^z}{\partial x} - \sigma E^y - J_E^y$$

$$\epsilon \frac{\partial E^z}{\partial t} = \frac{\partial H^y}{\partial x} - \frac{\partial H^x}{\partial y} - \sigma E^z - J_E^z$$

Semi-Discrete Maxwell System

$$Mw' = Aw + g(t), \quad A \sim \frac{1}{h}, \quad w = \begin{pmatrix} u \\ v \end{pmatrix} \approx \begin{pmatrix} H_h \\ E_h \end{pmatrix}$$

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} g_u \\ g_v \end{pmatrix}$$

- Mass matrices are symmetric positive definite

- *K* is the approximation for the curl operator

- Conduction matrix *S* is also symmetric positive definite

- For zero matrix *S*, matrix *A* is skew-symmetric

Stability and conservation

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

$$||w||_M^2 := \langle Mw, w \rangle, \quad \langle Mw, w \rangle := \langle M_u u, u \rangle + \langle M_v v, v \rangle$$

$$\frac{1}{2} \frac{d}{dt} \|w\|_M^2 = \langle Mw', w \rangle = \langle Aw, w \rangle = \langle -Sv, v \rangle < 0$$

- Hence stability, and (energy) conservation if $S = 0$.

- Time integrators should mimic this.

Stability and conservation

- Special case: constant ε and σ

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

where $S = \alpha M_v\,,\ \alpha = \dfrac{\sigma}{\epsilon}$

- Norm-preserving transformation yields decoupled *2x2 systems*

$$\begin{pmatrix} \widehat{u}' \\ \widehat{v}' \end{pmatrix} = \begin{pmatrix} 0 & -s \\ s & -\alpha \end{pmatrix} \begin{pmatrix} \widehat{u} \\ \widehat{v} \end{pmatrix}, \quad s = 0 \ \text{ or } \ \sqrt{\lambda_i(\tilde{K}^T \tilde{K})} \sim 1/h$$

- Useful for examining stability of time integration methods

(2) The exponential operator

The exponential operator

- For the autonomous problem

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} \quad \text{or} \quad w' = Jw$$

solution approximations can be obtained from

$$w(t) = e^{tJ}\, w(0)$$

- This is attractive (compared to time stepping) if $\|tJ\| \gg 1$ and very high temporal accuracy is wanted at time *t* only

- I'll compare two techniques: *Krylov-Arnoldi subspace iteration and Chebyshev series expansion*

Krylov-Arnoldi subspace iteration

- Approximates matfunvec $\varphi(tJ)b \in \mathbb{R}^n$ by

$$d = V_k\,\varphi(tH_k)\,e_1 \cdot \|b\|, \quad V_k \in \mathbb{R}^{n\times k}, \;\; H_k \in \mathbb{R}^{k\times k}$$

- Very efficient if $k \ll n$

- Main costs $\begin{cases} k \text{ matvecs with } tJ \\ \text{storage of } V_k \text{ (practical drawback)} \end{cases}$

- Worst case estimate for e^{tJ}, $J = -J^T$: $k \approx \|tJ\|$

Hochbruck, Lubich ‘97

Chebyshev series expansion (Tal-Ezer '86)

Also cf. De Raedt et al '02

- Approximates $e^{tJ}w(0)$ for skew-symmetric $J = -J^T$

- $e^z = J_0(R) + 2\sum_{k=1}^{\infty} J_k(R)\, Q_k(\frac{z}{R}), \quad z \in i\mathbb{R}, \quad |z| \leq R$

- $w_N(t) = P_N(tJ)w(0) \approx e^{tJ}w(0), \quad N > R \geq \sigma(tJ)$

- Only a three-term C-recursion required for Q_k

- Work: N matvecs and only 3 extra storage arrays

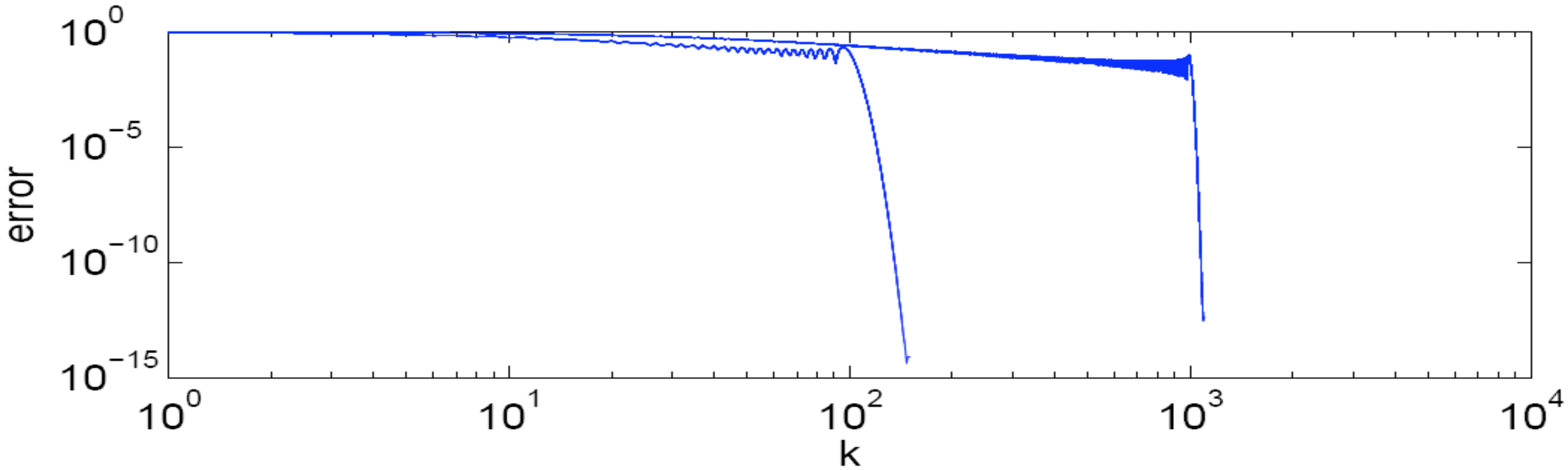
- $w_N(t)$ can be implemented to converge
  to any accuracy for $N = \mathcal{O}(\sigma(tJ)), \ \sigma(tJ) \to \infty$

Approximating along the imaginary axis

- Adaptive approximation to e^z with tol $= 10^{-16}$ (error increase is due to round off)

CWI Report MAS-R0806

$R = \|z\|$	10^2	10^3	10^4	10^5
$\|e^z - P_N(z)\|$	$7.6\,10^{-15}$	$4.0\,10^{-13}$	$4.1\,10^{-12}$	$2.4\,10^{-11}$
$N/\|z\|$	1.54	1.11	1.02	1.00



2D Example

$2D - TM$ model

$\epsilon = 1,\ \sigma = 0,\ J = 0$

unit square

$E^y = 0$ on boundary

$$\begin{cases} \mu \dfrac{\partial H^x}{\partial t} & = & \dfrac{\partial E^y}{\partial z} \\ \mu \dfrac{\partial H^z}{\partial t} & = & -\dfrac{\partial E^y}{\partial x} \\ \dfrac{\partial E^y}{\partial t} & = & \dfrac{\partial H^x}{\partial z} - \dfrac{\partial H^z}{\partial x} \end{cases}$$

$$H^x(x,z,0) = 0, \quad H^z(x,z,0) = 0$$

$$E^y(x,z,0) = \sin(\beta x)\sin(\beta z),\ \beta = 2\pi$$

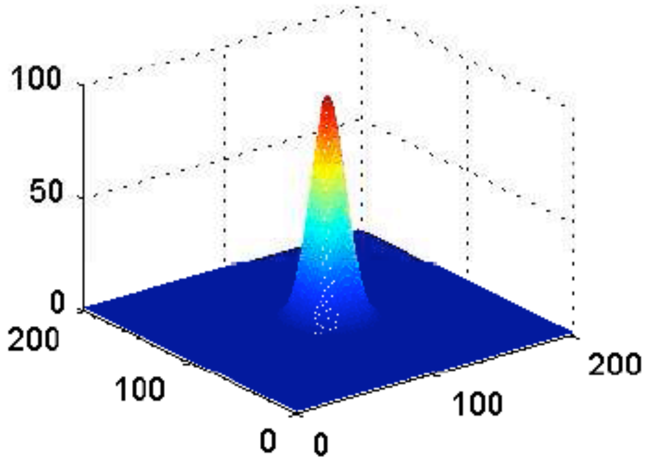
$$\mu(x,z) = [1.0 + 99\, e^{-(2.0\, 10^2\, ((x-0.5)^2 + (z-0.5)^2))}]^{-1}$$

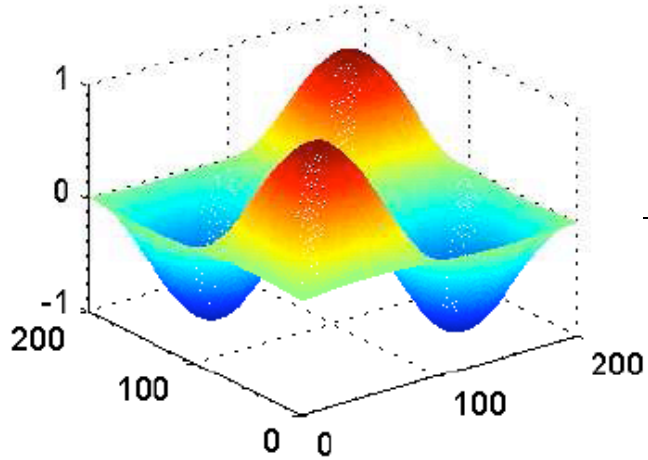
peaked shape with $\mu_{\min} = 10^{-2}$

2nd-order, staggered grid $\Rightarrow\ w(t) = e^{tJ}\, w(0),\ J = -J^T$

Solution at $t = 1$ for mesh width $h = 0.005$

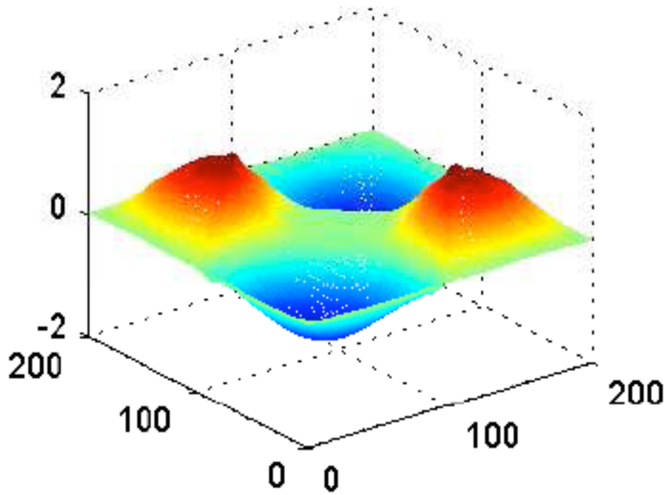
$\mu^{-1}(x, z)$





$E^y(t = 0)$

$E^y(t = 1)$ from $w(t) = e^{tJ}w(0)$

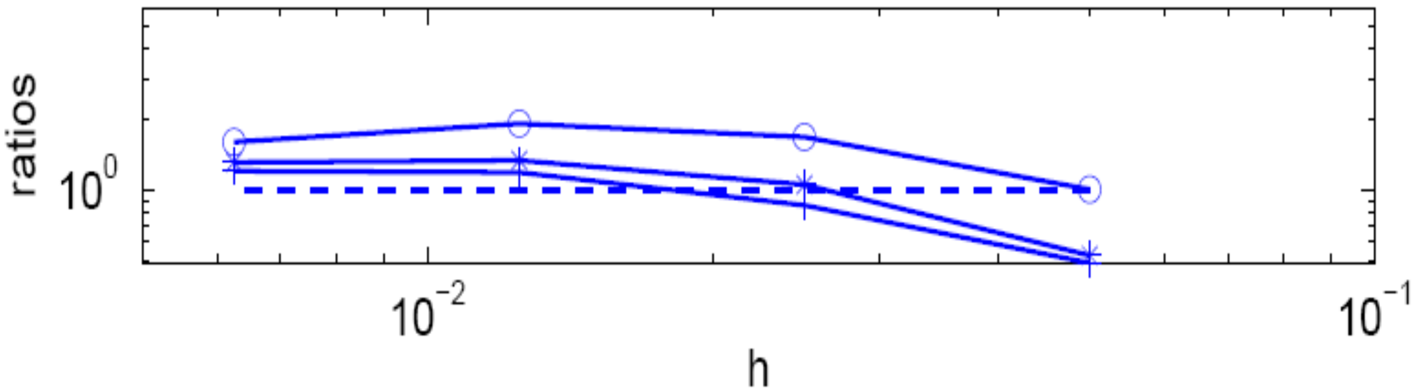


Krylov versus Chebyshev

Code expv (Sidje '98) compared to own Chebyshev code on four grids $h = 1/20, \ldots, 1/160$ for ≈ *10* decimal digits

Within expv, *maxit = 20, 40, 60* to avoid excessive storage

$$\text{ratio} = \frac{\text{Matvecs expv}}{\text{Matvecs Chebyshev}}$$



o : maxit = 20
$*$: maxit = 40
$+$: maxit = 60

For the current 2D problem Chebyshev is faster

(3) 2nd - order exponential integration

The 2nd - order exponential integrator EK2

$$w(t_{n+1}) = e^{\tau J} w(t_n) + \int_0^\tau e^{(\tau - s)J} f(t_n + s) ds$$

Interpolation: source is linearly interpolated and resulting terms are computed analytically Certaine '60

EK2: $$w_{n+1} = w_n + \tau \varphi_1(\tau J)\, w_n' + \tau \varphi_2(\tau J) \Big(f(t_{n+1}) - f(t_n) \Big)$$

$$\varphi_1(z) = (e^z - 1)/z, \quad \varphi_2(z) = (\varphi_1(z) - 1)/z$$

See e.g. also Hochbruck & Ostermann (Acta Numerica, forthcoming)

Convergence theorem EK2

$$w_{n+1} = w_n + \tau\varphi_1(\tau J)\, w_n' + \tau\varphi_2(\tau J)\Big(f(t_{n+1}) - f(t_n)\Big)$$

$$\varphi_1(z) = (e^z - 1)/z, \quad \varphi_2(z) = (\varphi_1(z) - 1)/z$$

Thm.: *For smooth solutions w(t) we have convergence with order 2 for any stable J and any source function f.*

Proof: *See V.& B., LAA paper.*

Such convergence suffices for PDEs with time-dependent bc's (stiff source terms) to maintain temporal order 2 upon spatial grid refinement

Sanz-Serna, V. & Hundsdorfer, Numer. Math. '86

A naïve 2^{nd} - order exponential integrator

Naïve approach: trapezoidal quadrature of the integral term

$$w(t_{n+1}) = e^{\tau J} w(t_n) + \int_0^{\tau} e^{(\tau - s)J} f(t_n + s) ds$$

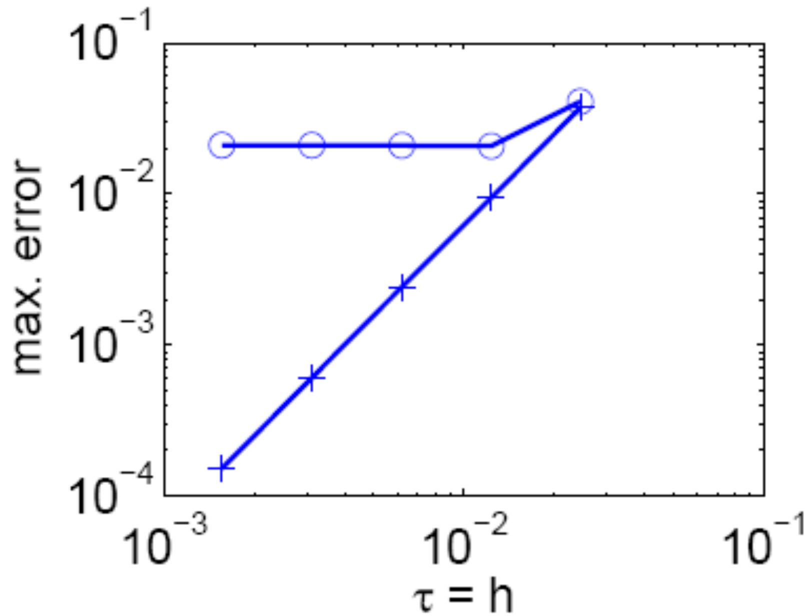
yields the 2^{nd} – order method

$$w_{n+1} = e^{\tau J} \left(w_n + \frac{1}{2} \tau f(t_n) \right) + \frac{1}{2} \tau f(t_{n+1})$$

Convergence test EK2 and naïve method

$$u_t + u_x = 0, \quad 0 < x < 1, \quad 0 < t \leq 1$$
$$u(x,t) = \cos(\omega(x-t)), \ \omega = 2\pi$$

Central 2nd order FD in space, Dirichlet cnds



2nd order EK2 (+) convergence in the PDE sense, i.e. for simultaneous space-time refinement

However, no PDE convergence for naïve method (o) due to time-dependent bndry values

History exponential integrators

- Exponential integrators like EK2 and related methods have a long history:

  Certaine ’60, Legras ’66, Lawson ’67, Nørsett ’69
  Van der Houwen & V. ’74, V. ‘77
  Friedli ’78, Strehmel & Weiner ’82

- A revival since the late nineties:

  Hochbruck, Ostermann, Lubich, Selhofer
  Beylkin, Keiser, Vozovoi
  Cox, Matthews, Krogstad
  Berland, Celledoni, Owren, Martinsen
  … … …
  supported by Krylov subspace iteration for computing the matrix functions (H.& L.)

(4) Second-order explicit (in K) method

Second-order explicit (in K) method

- The method exploits the partitioned structure in

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} g_u \\ g_v \end{pmatrix}$$

and is composed of three sub-steps within one time step:

Second-order explicit (in K) method

- The method exploits the partitioned structure in

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} g_u \\ g_v \end{pmatrix}$$

and is composed of three sub-steps within one time step:

$$M_u \frac{u_{n+1/2} - u_n}{\tau} = -\tfrac{1}{2} K v_n + \tfrac{1}{2} g_u(t_n)$$

Second-order explicit (in K) method

- The method exploits the partitioned structure in

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} g_u \\ g_v \end{pmatrix}$$

and is composed of three sub-steps within one time step:

$$\begin{aligned} M_u \frac{u_{n+1/2} - u_n}{\tau} &= -\tfrac{1}{2} K v_n + \tfrac{1}{2} g_u(t_n) \\ M_v \frac{v_{n+1} - v_n}{\tau} &= K^T u_{n+1/2} - \frac{S(v_n + v_{n+1})}{2} + \frac{g_v(t_n) + g_v(t_{n+1})}{2} \end{aligned}$$

Second-order explicit (in K) method

- The method exploits the partitioned structure in

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} g_u \\ g_v \end{pmatrix}$$

and is composed of three sub-steps within one time step:

$$\begin{aligned} M_u \frac{u_{n+1/2} - u_n}{\tau} &= -\tfrac{1}{2} K v_n + \tfrac{1}{2} g_u(t_n) \\ M_v \frac{v_{n+1} - v_n}{\tau} &= K^T u_{n+1/2} - \frac{S(v_n + v_{n+1})}{2} + \frac{g_v(t_n) + g_v(t_{n+1})}{2} \\ M_u \frac{u_{n+1} - u_{n+1/2}}{\tau} &= -\tfrac{1}{2} K v_{n+1} + \tfrac{1}{2} g_u(t_{n+1}) \end{aligned}$$

Second-order explicit (in K) method

- The method exploits the partitioned structure in

$$\begin{pmatrix} M_u & 0 \\ 0 & M_v \end{pmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} 0 & -K \\ K^T & -S \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} g_u \\ g_v \end{pmatrix}$$

and is composed of three sub-steps within one time step:

$$\begin{aligned} M_u \frac{u_{n+1/2} - u_n}{\tau} &= -\tfrac{1}{2} K v_n + \tfrac{1}{2} g_u(t_n) \\ M_v \frac{v_{n+1} - v_n}{\tau} &= K^T u_{n+1/2} - \frac{S(v_n + v_{n+1})}{2} + \frac{g_v(t_n) + g_v(t_{n+1})}{2} \\ M_u \frac{u_{n+1} - u_{n+1/2}}{\tau} &= -\tfrac{1}{2} K v_{n+1} + \tfrac{1}{2} g_u(t_{n+1}) \end{aligned}$$

- First substep is free: $u_{n+1/2} = 2u_n - u_{n-1/2}\,, \quad n \geq 1$

Second-order explicit (in *K*) method

CO2 :

$$M_u \frac{u_{n+1/2} - u_n}{\tau/2} = -K v_n + g_u(t_n)$$
$$M_v \frac{v_{n+1} - v_n}{\tau} = K^T u_{n+1/2} - \frac{S(v_n + v_{n+1})}{2} + \frac{g_v(t_n) + g_v(t_{n+1})}{2}$$
$$M_u \frac{u_{n+1} - u_{n+1/2}}{\tau/2} = -K v_{n+1} + g_u(t_{n+1})$$

- CO2 (COmposition order 2) is derived through composition

- cheap per time step (just one function evaluation)

- is akin to the time-staggered scheme (Yee, Störmer-Verlet)

$$M_u \frac{u_{n+1/2} - u_{n-1/2}}{\tau/2} = -K v_n + g_u(t_n)$$
$$M_v \frac{v_{n+1} - v_n}{\tau} = K^T u_{n+1/2} - \frac{S(v_n + v_{n+1})}{2} + \frac{g_v(t_n) + g_v(t_{n+1})}{2}$$

Second-order explicit (in K) method

- has even global error expansion

$$w(t_n) - w_n = C_2\tau^2 + C_4\tau^4 + \cdots$$

uniformly in spatial mesh width (good for g-extrapolation)
See B. & V., SISC paper for the proof.

Convergence uniformly in the spatial mesh width is needed for PDEs with time-dependent bc's to maintain the order upon spatial grid refinement!

NB: Higher-order compositions suffer from order reduction.

Second-order explicit (in *K*) method

- for zero *S* and zero sources the method conserves

$$\|w_n\|_M^2 - \frac{1}{4}\tau^2 \langle M_u^{-1} K e_n, K e_n \rangle$$

An ideal 2nd order method, except that it is conditionally stable for the curl terms:

$$\begin{pmatrix} \widehat{u}' \\ \widehat{v}' \end{pmatrix} = \begin{pmatrix} 0 & -s \\ s & -\alpha \end{pmatrix} \begin{pmatrix} \widehat{u} \\ \widehat{v} \end{pmatrix} : \quad \begin{array}{l} \tau s < 2 \text{ if } \alpha = 0 \\ \tau s \leq 2 \text{ if } \alpha > 0 \end{array}$$

Stability limit for the 2D problem

$2D - TM$ model

$\epsilon = 1,\ \sigma = 0,\ J = 0$

unit square

$E^y = 0$ on boundary

$$\begin{cases} \mu \dfrac{\partial H^x}{\partial t} & = & \dfrac{\partial E^y}{\partial z} \\ \mu \dfrac{\partial H^z}{\partial t} & = & -\dfrac{\partial E^y}{\partial x} \\ \dfrac{\partial E^y}{\partial t} & = & \dfrac{\partial H^x}{\partial z} - \dfrac{\partial H^z}{\partial x} \end{cases}$$

Central 2nd - order discretization on a uniform staggered grid with grid size *h* gives for CO2 the stability limit

$$\tau \ \leq \begin{cases} \sqrt{\dfrac{\mu}{2}}\, h, & \mu \text{ constant} \\ \sqrt{\dfrac{\mu_{\text{min}}}{2}}\, h, & \mu \text{ variable} \end{cases}$$

(5) Numerical comparison CO2 and EK2

Numerical comparison CO2 and EK2

- 3D Maxwell

$$\begin{cases} \mu\, \partial_t H &= -\nabla \times E \\ \epsilon\, \partial_t E &= \nabla \times H - \sigma E - J \end{cases}$$

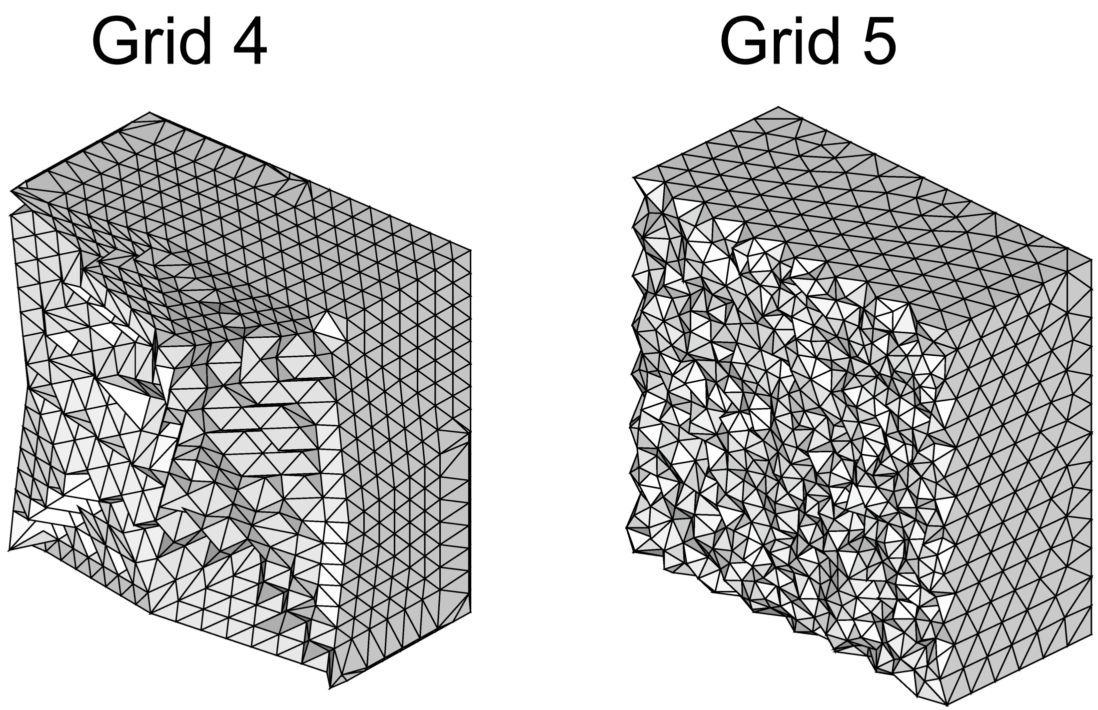
discretized with 1^{st}-order, 1^{st}-type Nédélec FEM on tetrahedral unstructured grids

- Unit square, $0 < t \leq 10$, prescribed solution

Ackn: *Mike Botchev*

Grids

grid	number of edges	longest edge $h_{\max}$	shortest edge $h_{\min}$	CO2 time step restriction	CO2 time step used
4	34608	0.250	0.0063	0.028	0.0250
5	85308	0.118	0.0139	0.014	0.0125



Terminating Krylov within EK2

EK2: $w_{n+1} = w_n + \tau\Phi_1 + \tau\Phi_2$

$$\Phi_1 = \varphi_1(\tau J)\, w_n' \,, \quad \Phi_2 = \varphi_2(\tau J)\left(f(t_{n+1}) - f(t_n)\right)$$

After k_1, k_2 Krylov iterations: $\widehat{w}_{n+1} = w_n + \tau\Phi_1^{k_1} + \tau\Phi_2^{k_2}$

The aim is $\|w_{n+1} - \widehat{w}_{n+1}\| \leq lte_{EK2} \overset{exas}{=} \tau\|w_n\|\delta$

This holds if $\|\Phi_i - \Phi_i^{k_i}\| \leq \frac{1}{2}\|w_n\|\,\delta \leq \frac{1}{2}\,lte_{EK2}$

Implementation:
iteration is stopped if $\|\Phi_i^{k_i} - \Phi_i^{k_i-1}\| \leq \frac{1}{2}\|w_n\|\,\delta$

for a prescribed tolerance δ

results EK2 (1)

$\sigma = 0$, 34 608 DOFs

τ	# matvecs per t.step	total # matvecs	t.error m.field	t.error el.field
		CO2		
0.025	1	400	1.21e-02	1.23e-02
		EK2 ($\delta = 10^{-3}$)		
0.0625	14.9	2388	8.28e-04	2.67e-04
0.125	22.0	1757	3.36e-03	1.12e-03
0.25	35.5	1418	2.13e-02	1.71e-02
0.5	62.2	1757	1.17e-01	1.05e-01
1.0	116	1160	5.88e-01	5.84e-01

results EK2 (2)

$\sigma = 60\pi$, 34 608 DOFs

τ	# matvecs per t.step	total # matvecs	t.error m.field	t.error el.field
		CO2		
0.025	1	400	1.15e-04	9.34e-06
		EK2 ($\delta = 10^{-3}$)		
0.0625	11.5	1836	1.07e-03	5.19e-05
0.125	13.7	1096	3.43e-03	1.26e-04
0.25	16.4	654	1.32e-02	4.34e-04
0.5	21.6	431	4.99e-02	1.81e-03
1.0	29.6	296	1.96e-01	7.18e-03

Conclusions

We have examined explicit versus implicit time stepping for Maxwell's equations. Our, as yet limited, experience indicates

-- The cheap explicit method CO2 will be hard to beat

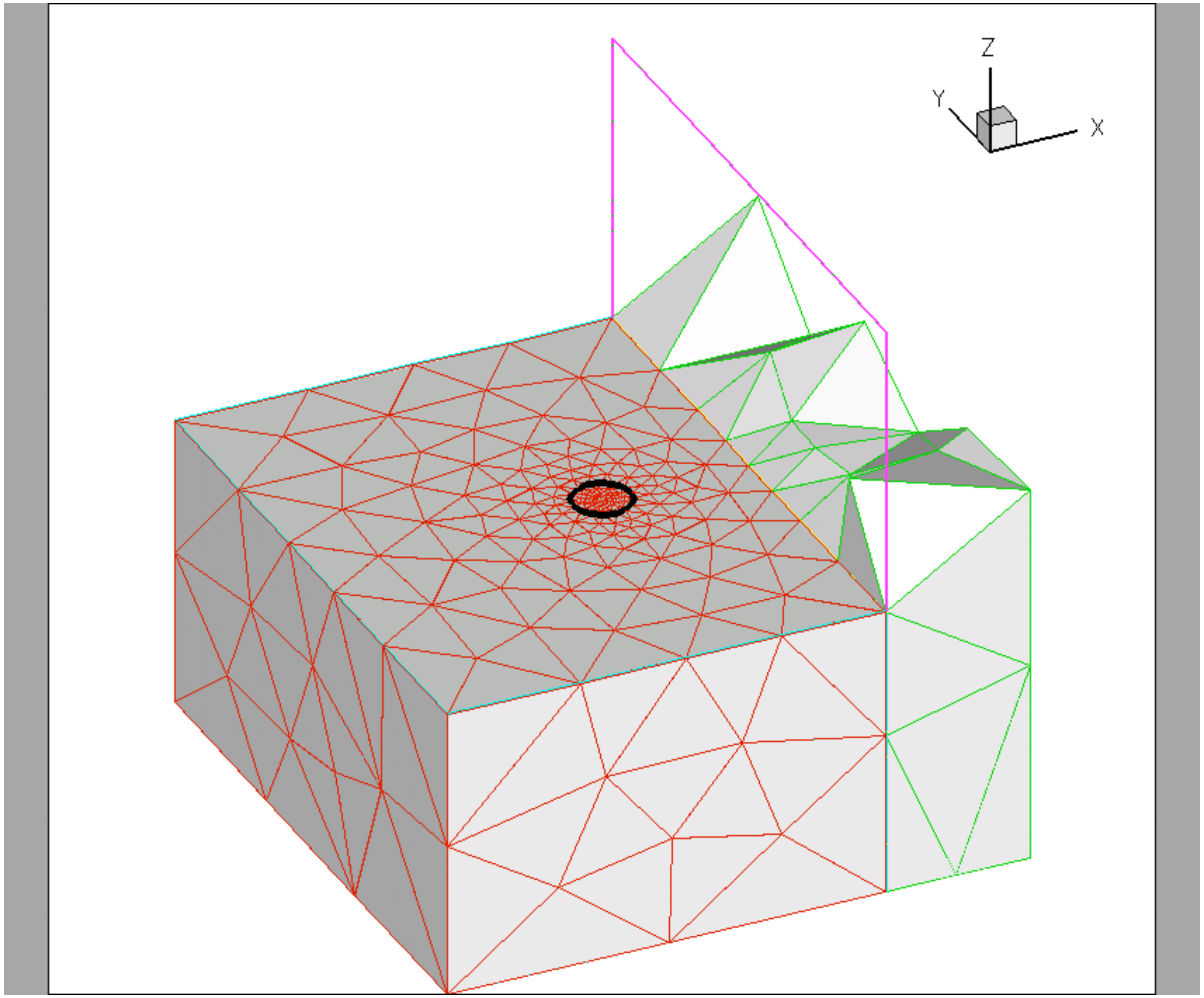
-- Convergence of Krylov subspace iteration takes too long

-- Same conclusion for ITR implemented with CG

-- For stiff autonomous problems and high ODE accuracy, an exponential solver (Krylov, Chebyshev) is advocated

-- And in case of skew-symmetry, Chebyshev (Tal-Ezer) is then recommended (also cf. De Raedt et al '02)

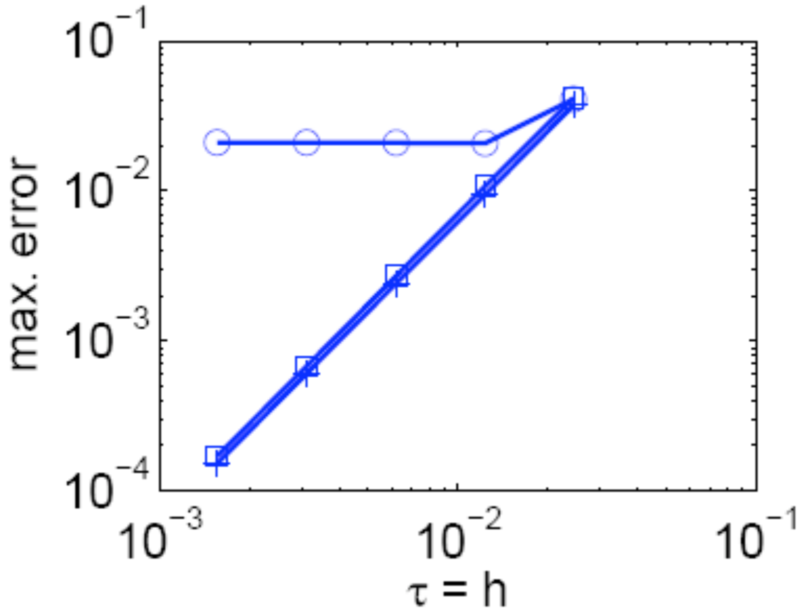
Coil problem grid



Restoring PDE convergence for naïve method

$$u_t + u_x = 0, \quad 0 < x < 1, \quad 0 < t \leq 1$$

$$u(x,t) = \cos(\omega(x-t)), \; \omega = 2\pi$$



2nd order for naïve method (o)

$$w_{n+1} = e^{\tau J}\left(w_n + \frac{1}{2}\tau f(t_n)\right) + \frac{1}{2}\tau f(t_{n+1})$$

is restored by "boundary differentiation", see marks (□)

Same accuracy as EK2 (+)