Recent Progress in 3D Wakepotential Computations

Warner Bruns

bruns@gdfidl.de

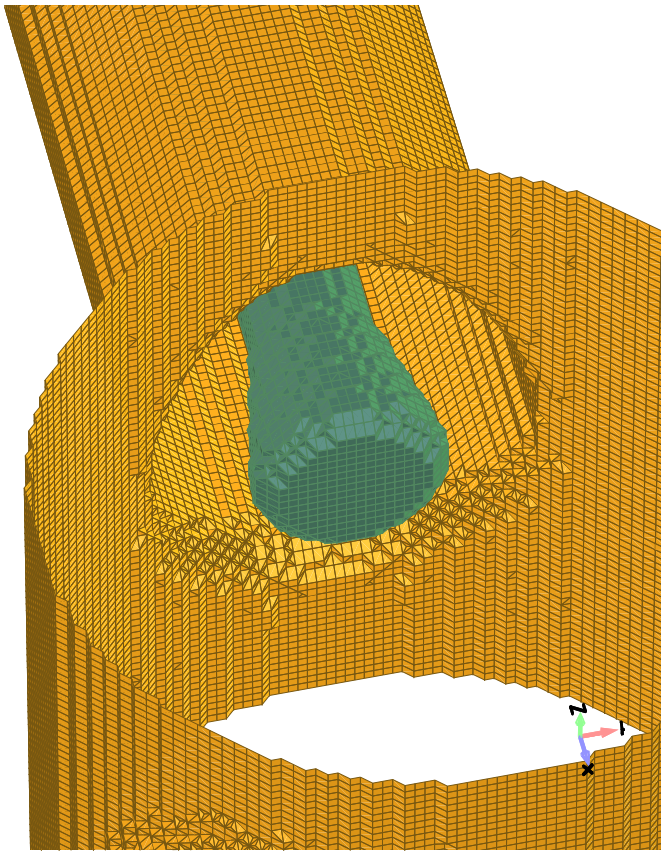
June 17, 2014

IPAC 2014

Overview

- Generalised diagonal Fillings.
- Dispersive Materials.
- Low Dispersion FDTD-Scheme.
- Short Range Wakepotentials: 'Moving Mesh'
  Yee-Scheme, Strang-Splitting and low Dispersion FDTD.
- For Moving Mesh, the FD-Coefficients are computed on the Fly.
- Small Memory Footprint, high Performance:
  Fieldvalues only in interesting Cells.

Generalised diagonal Fillings



Detail of the Coupler Region of an ILC Module.
Data from Andrei Lunin, ANL.

FDTD on rectangular Cells.
Materialdistribution is approximated with generalised diagonal Fillings.
Lossy dispersive Materials for Time Domain and Eigenvalue Problems,
Time Domain: Impedance Boundary Conditions at metallic Surfaces.
Parallelised for MultiCore/ MultiSocket Systems and Clusters.

Dispersive Materials

Frequency dependent Materialparameters are taken into account by directly simulating the Dynamics of the Electron Hull of Molecules. For each Fieldcomponent in a dispersive Material with N Poles, the Equations of Motion (v Velocity, Q Charge, k Spring-constant, R Damping Term, m Mass of the electron Hull) are solved

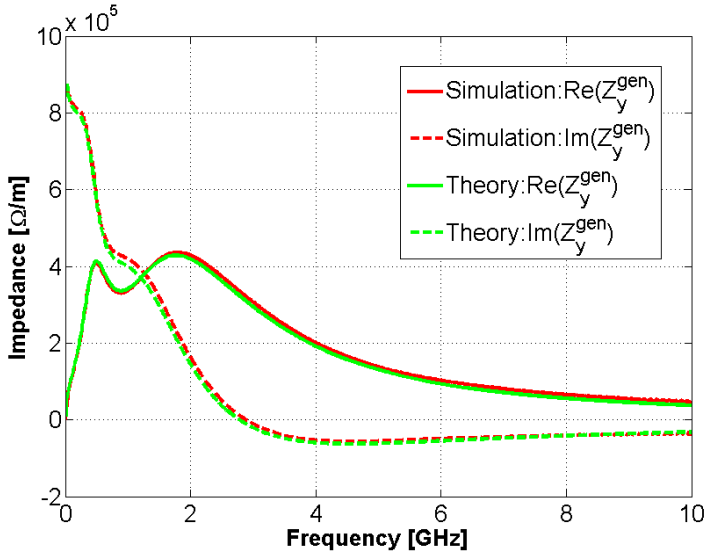
$$\begin{aligned}
\frac{d}{dt}v_i &= \frac{Q_i}{m_i}E - \frac{k_i}{m_i}x_i - \frac{R_i}{m_i}v_i \\
\frac{d}{dt}x_i &= v_i \\
\frac{d}{dt}E &= \frac{1}{\varepsilon}\nabla\times H - \frac{1}{\varepsilon}\sum_{i=1}^{N} Q_i v_i
\end{aligned}$$

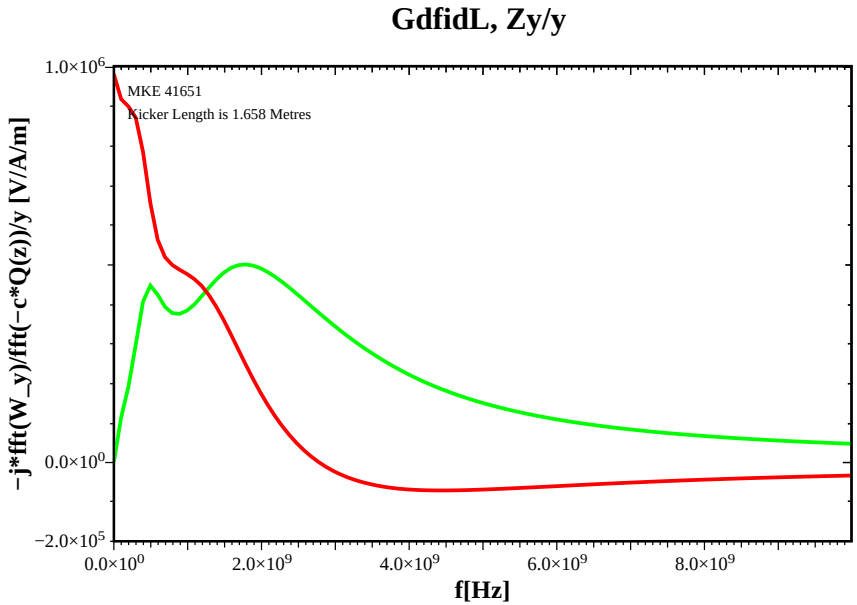
Dispersive Materials

The Spring-constants, Damping-terms and Charge of the Electron-Hull are derived from the A_i, ω_i, γ_i of a N.th order LORENTZ Medium.

$$\varepsilon_r(\omega) = \varepsilon_\infty + \sum_{i=1}^{N} \frac{A_i \omega_i^2}{\omega_i^2 + \mathrm{j}\omega\gamma_i - \omega^2} \tag{1}$$

Impedance of a Kicker





Carlo Zannini, PhD-Thesis, Page 53

Impedance of a 1.7 Metres long MKE41651 Kicker, 30 Minutes for a Wakepotential up to 3 Metres, 1800 MByte for 20 Million Cells, all Cells filled by dielectrics, 60% filled by lossy dispersive Material.

Low Dispersion FDTD-Scheme

The first three Terms of the TAYLOR-Series for $H(t)$ give

$$\begin{aligned} H(t+\Delta t/2) &= H(t-\Delta t/2) + \Delta t \frac{d}{dt} H(t) \\ &\quad + \frac{(\Delta t)^3}{24} \left(\frac{d}{dt}\right)^3 H(t) + \cdots \end{aligned}$$

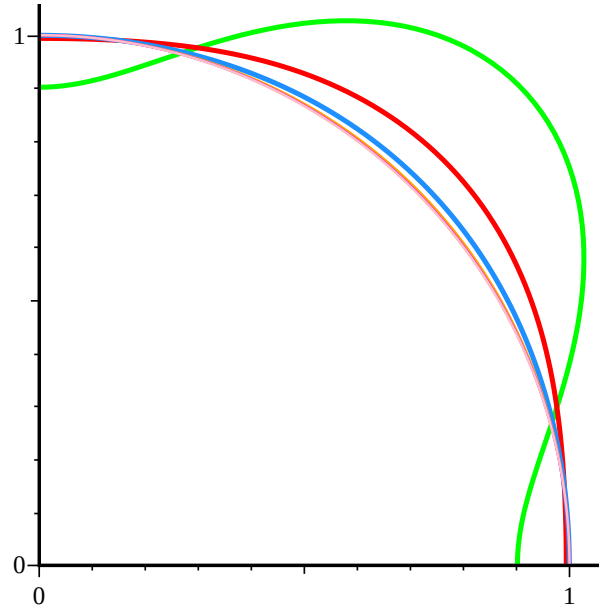
Higher Order Leap-Frog

$$\begin{aligned} H(n+1/2) &= H(n-1/2) - \Delta t \frac{1}{\mu} \nabla \times E(n) \\ &\quad - \frac{(\Delta t)^3}{24} \frac{1}{\mu} \nabla \times \frac{1}{\varepsilon} \nabla \times \frac{1}{\mu} \nabla \times E(n) \end{aligned}$$

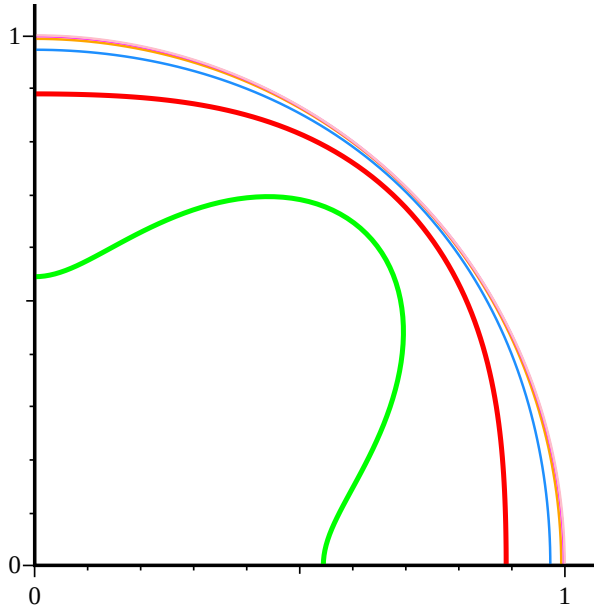
Such a Scheme has a Convergence of fourth Order, if the Curl-Operators are approximated by a Scheme of Order 4. Such higher Order spatial Schemes are problematic at material Discontinuities. If the Curl-Operators are approximated by the standard FDTD-Curl-Operators, the resulting Scheme is still second Order convergent as the Yee-Scheme is, but the Directions in which zero Dispersion occurs may be chosen. The standard Yee-Scheme has zero Dispersion when the highest possible Timestep $c\Delta t = \Delta x \frac{1}{\sqrt{3}}$ is used and the Wave propagates in Direction of a Grid-Diagonal. The dispersion optimised Scheme has very low Dispersion for Waves propagating in x-, y- or z-Direction, when a Timestep of $c\Delta t = \Delta x \frac{1}{4}$ is used and the $(\Delta t)^3 \nabla\times^3$ Term is weighted by 32/51, not 1/24. Because the Scheme is constructed of Curl-Operators, the resulting Fields have no spurious Divergence.

Low Dispersion FDTD-Scheme

Directional Dependence of the numerical Phase-velocity:



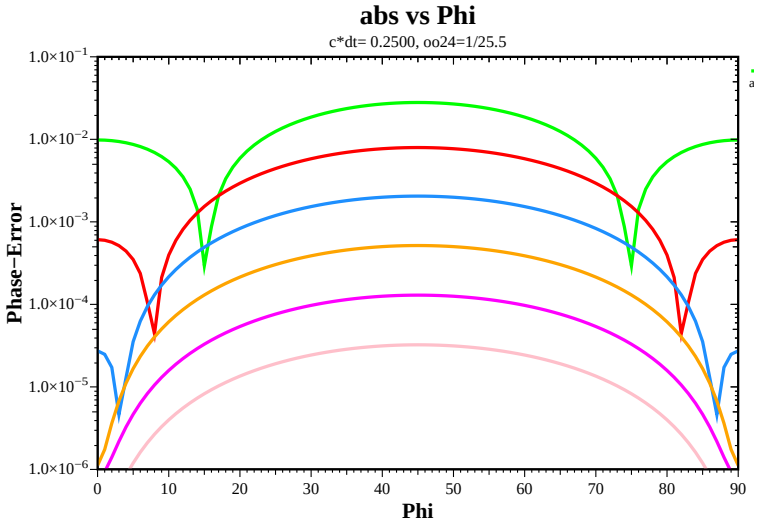
Modified FDTD $c\Delta t = \Delta z/4$

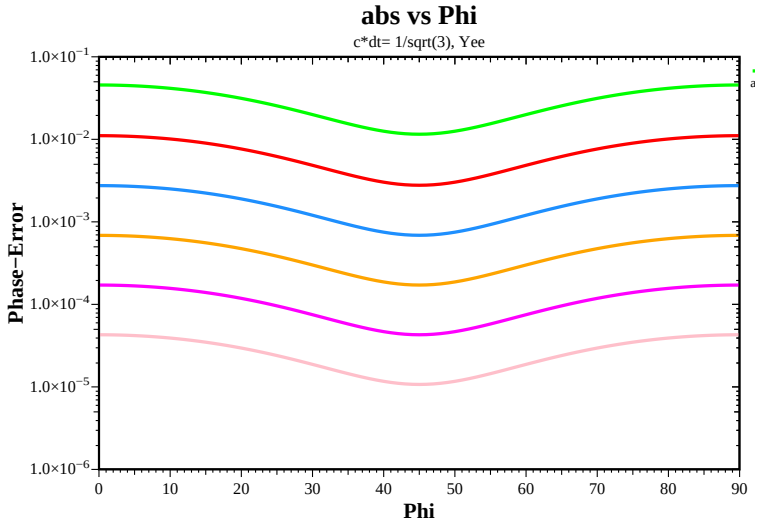


Standard FDTD $c\Delta t = \Delta z/\sqrt{3}$

Green: $\Delta z = \lambda/5$, Red: $\Delta z = \lambda/10$, Blue: $\Delta z = \lambda/20$

In the Plots, the Error is amplified by a Factor of 10

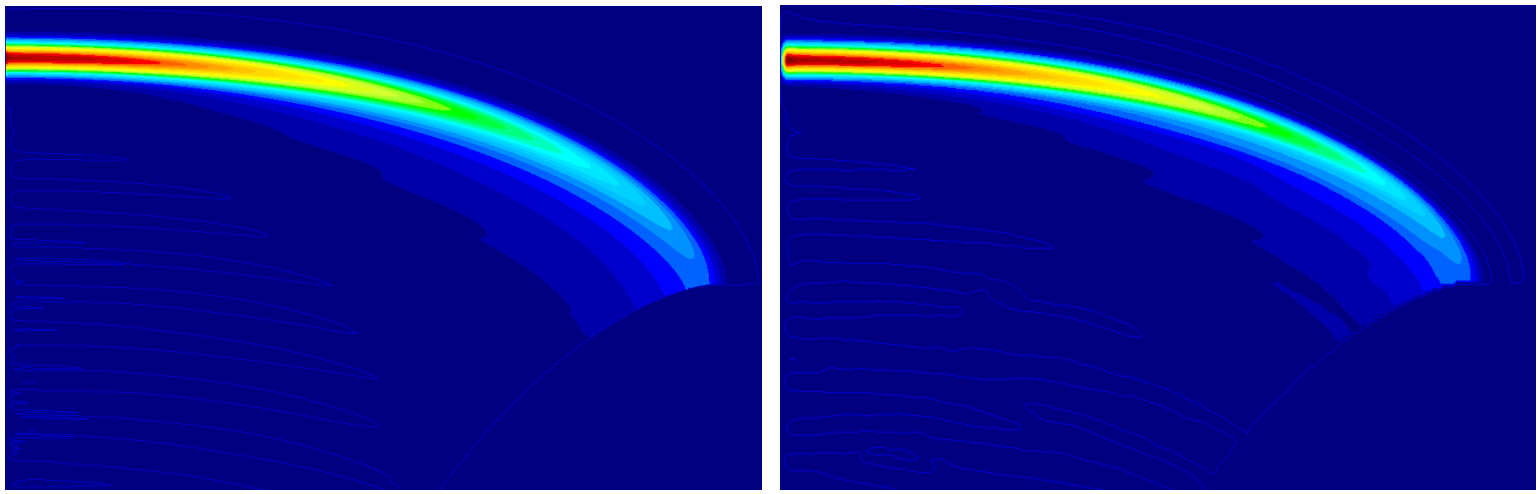




Modified FDTD $c\Delta t = \Delta z/4$ Standard FDTD $c\Delta t = \Delta z/\sqrt{3}$

For Waves going in x-, y-, or z, the Error is like of a fourth Order Scheme.

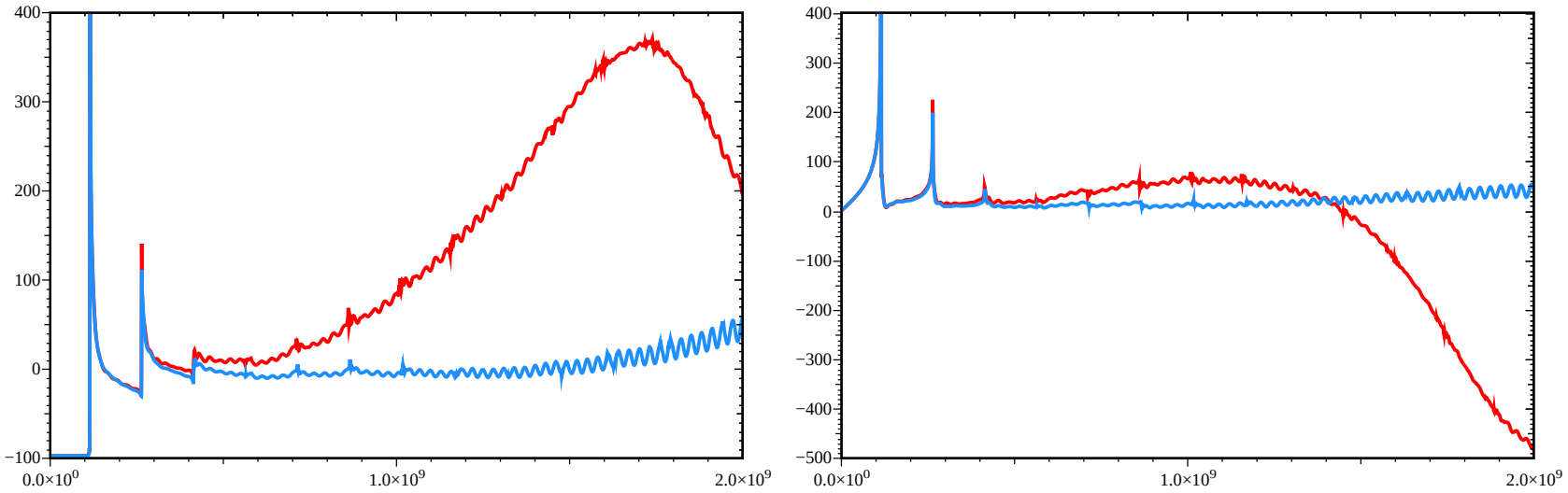
Low Dispersion FDTD-Scheme



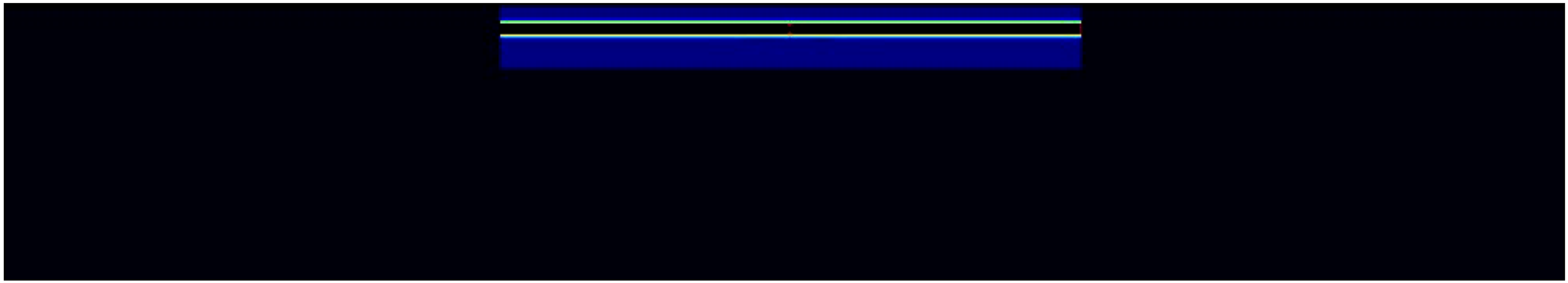
Fieldplots of a Linecharge exiting a Tractrix-Horn. Left: Computed with the Yee-Scheme, Sigma/dz=18. Right: Computed with the mFDTD-Scheme, Sigma/dz=3.

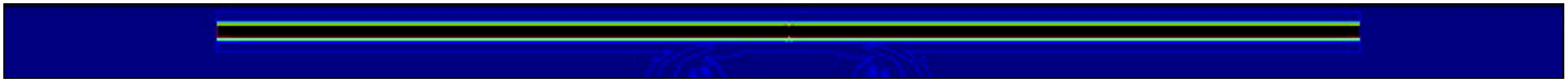
The Yee-Grid has a Factor of $6^3 = 216$ more Cells.

Low Dispersion FDTD-Scheme

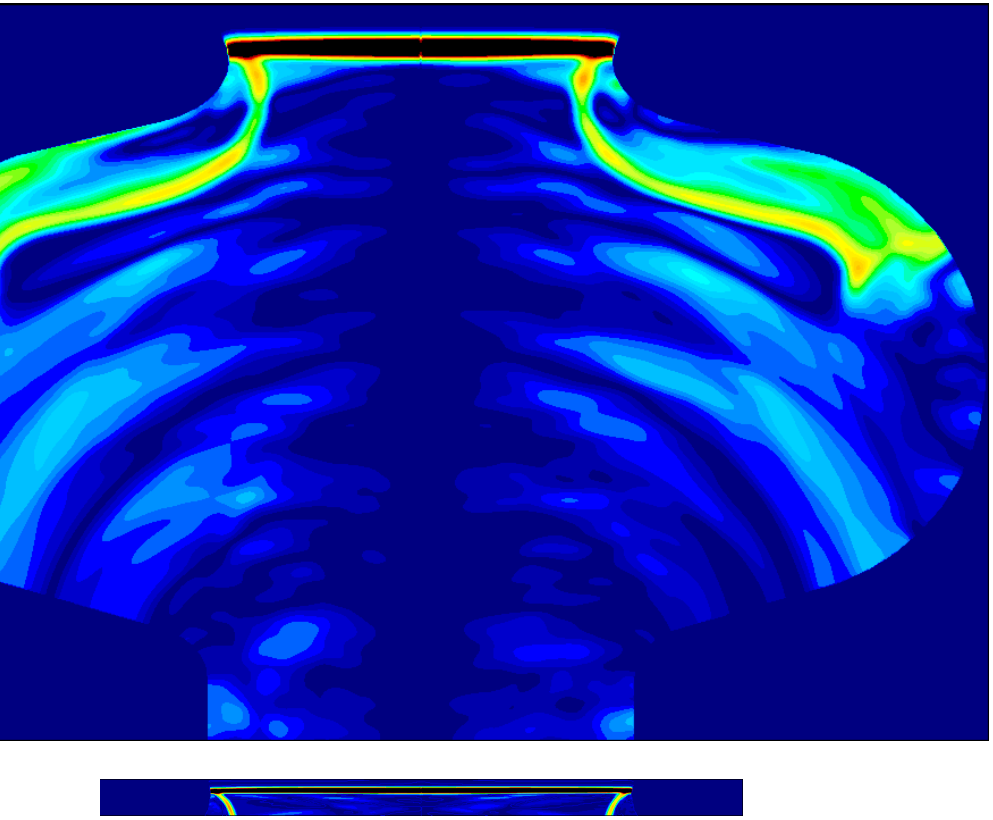


Impedance of a Step, computed with the Yee-Scheme and with the modifified FDTD-Scheme. 200 Gridplanes of dz=25mm before the Step. The Frequency where the Wavelength is sampled by 20 Gridplanes is $f = \frac{\mathrm{C}}{20\Delta z} = 600\mathrm{MHz}$.





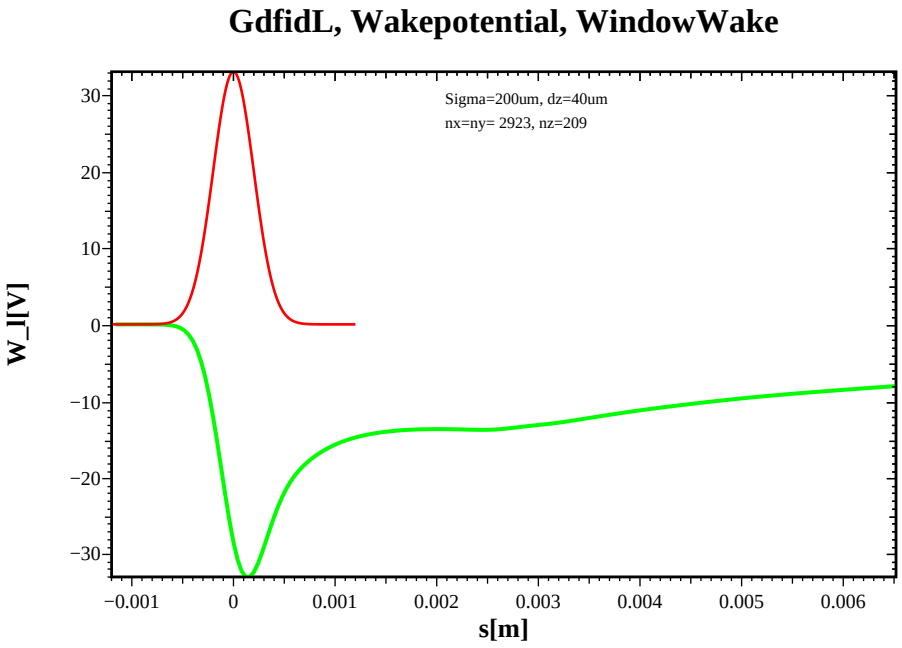
Moving Mesh, FD-Coefficients computed on the Fly

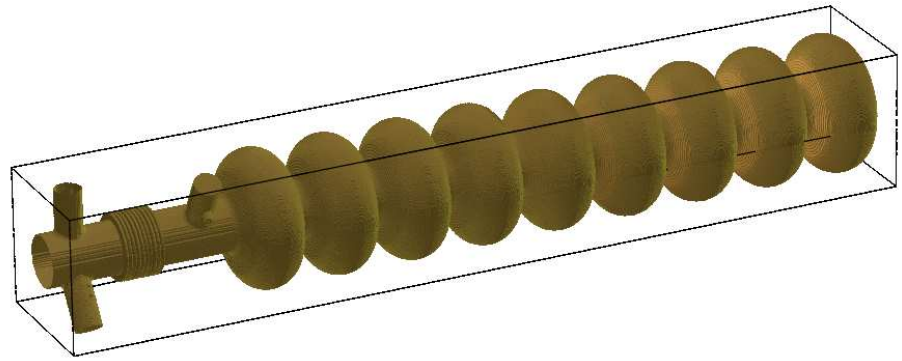


Computed with the mFDTD-Scheme.

Fieldplots of a Linecharge in a 9-Cell ILC-Module, entering the second Cavity.
Above: Sigma=1.3 mm, sHigh= 96 Sigma = 0.125 Metres. With a Gridspacing of Sigma/6, there are (96+6)*6=612 z-Planes. Such a long s-Range is unsensible large.
Below: In a restricted Volume. Sigma=0.3mm, sHigh= 12 Sigma = 3.6 mm. With a Gridspacing of Sigma/6, there are (12+6)*6=108 z-Planes.
When there are only 100 or so Gridplanes, the Preparation of a fresh Plane should better take less than 100 times CPU-Cycles that it takes to advance the Fields in a single Plane. This is achieved.

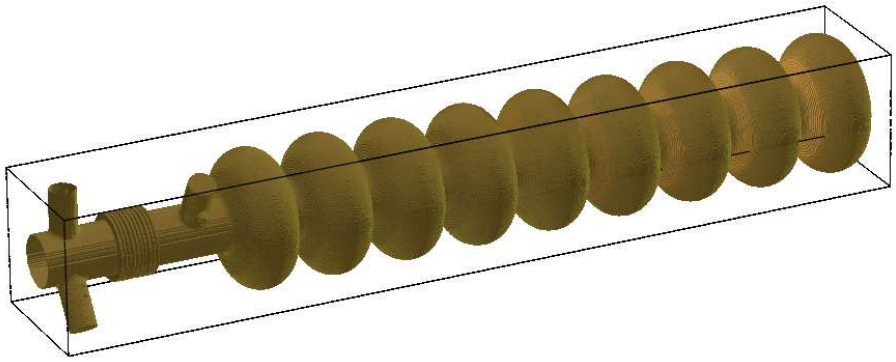
Moving Mesh, FD-Coefficients computed on the Fly





Longitudinal Wakepotential of a 9-Cavity ILC-Module with Couplers and Bellows.
Sigma=0.2 mm, sHigh= 32 * Sigma = 6.5 mm, dz=Sigma/5. 200 z-Planes, 60 GByte, 6 Days using 32 Cores of a 7000 Euro Server.

Small Memory Footprint



42% is filled with Vacuum .

1 Integer per Cell, plus 6 Reals and 32 Bits per dielectric Cell.

(4 + 0.42 * (6 * 8 + 1)) * 1GByte = 24.6 GByte.

No Losses: Textbook FDTD needs 12 Numbers per Cell. 12 * 8 GBytes = 96 GBytes.

Losses: Textbook FDTD needs 18 Numbers per Cell. 18 * 8 GBytes = 144 GBytes.

Lots of Tricks applied

- Compressed Fieldcomponents: Cells which are filled by perfectly conducting Materials don't need CPU and not much Memory (32 Bits per Cell for Grids up to 2×10^9 dielectric Cells).,

- Compressed FD-Coefficients: The Memory for the FD-Coefficients is 1/24 of the Memory for the Field Components. A Textbook FDTD-Implementation needs 12 Numbers per Cell, 48 Bytes or 96 Bytes. GdfidL needs 32 Bits per dielectric Cell.

- The Field Update is implemented with Tricks which can be found in the Literature, eg: Non Uniform Memory Access aware Field Access via pinning Field Segments to Threads and Threads to CPU-Cores. Cache aware Field Access via blocked Schemes.

- and then some, which cannot be found in the Literature, eg. the H & E Components are updated not in separate Loops but in a single Loop per Block. This is another Cache-Optimisation.

```
-- Standard Approach is to compute all H of the Grid,
-- after that to compute all E.
-- Compute new H:
   FOR iz IN Block.iz1 .. Block.iz2 LOOP
      FOR iy IN Block.iy1 .. Block.iy2 LOOP
         FOR ix IN Block.ix1 .. Block.ix2 LOOP
            i:= NrofCell(ix,iy,iz);
            IF i>0 THEN                -- Dielectric Cell.
               iType:= CellType(i);    -- Index of the FD-Coefficients of the Cell
               ixp1:= NrofCell(ix+1,iy  ,iz  );  -- Number of Neighbourcell in +x.
               iyp1:= NrofCell(ix  ,iy+1,iz  );  -- Number of Neighbourcell in +y.
               izp1:= NrofCell(ix  ,iy  ,iz+1);  -- Number of Neighbourcell in +z.
Hds(1,i):= Hf1(1,iType) * Hds(1,i) - Hf2(1,iType) * ( Eds(3,iyp1) - Eds(3,i) - Eds(2,izp1) + Eds(2,i) );
Hds(2,i):= Hf1(2,iType) * Hds(2,i) - Hf2(2,iType) * ( Eds(1,izp1) - Eds(1,i) - Eds(3,ixp1) + Eds(3,i) );
Hds(3,i):= Hf1(3,iType) * Hds(3,i) - Hf2(3,iType) * ( Eds(2,ixp1) - Eds(2,i) - Eds(1,iyp1) + Eds(1,i) );
            END IF;
   END LOOP; END LOOP; END LOOP;

-- Compute new E:
   FOR iz IN Block.iz1 .. Block.iz2 LOOP
      FOR iy IN Block.iy1 .. Block.iy2 LOOP
```

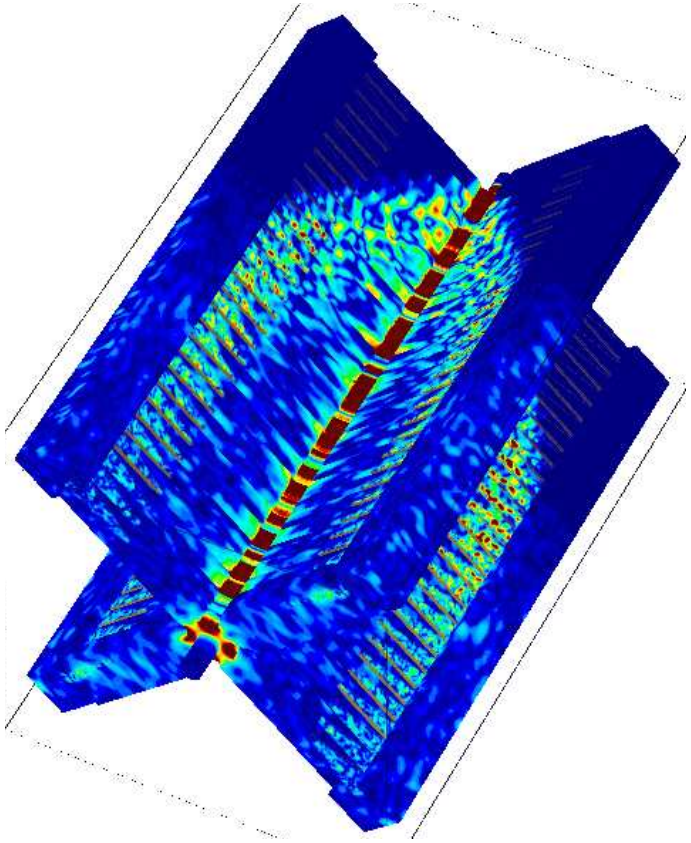
```
 -- Compute new E:
    FOR iz IN Block.iz1 .. Block.iz2 LOOP
       FOR iy IN Block.iy1 .. Block.iy2 LOOP
          FOR ix IN Block.ix1 .. Block.ix2 LOOP
             i:= NrofCell(ix,iy,iz);
             IF i>0 THEN                -- Dielectric Cell.
                iType:= CellType(i);    -- Index of the FD-Coefficients of the Cell
                ixm1:= NrofCell(ix-1,iy  ,iz  );
                iym1:= NrofCell(ix  ,iy-1,iz  );
                izm1:= NrofCell(ix  ,iy  ,iz-1);
 Eds(1,i):= Ef1(1,iType) * Eds(1,i) + Ef2(1,iType) * ( Hds(3,i)-Hds(2,i)+Hds(2,izm1)-Hds(3,iym1) );
 Eds(2,i):= Ef1(2,iType) * Eds(2,i) + Ef2(2,iType) * ( Hds(1,i)-Hds(3,i)+Hds(3,ixm1)-Hds(1,izm1) )
 Eds(3,i):= Ef1(3,iType) * Eds(3,i) + Ef2(3,iType) * ( Hds(2,i)-Hds(1,i)+Hds(1,iym1)-Hds(2,ixm1) )

             END IF;
          END LOOP;
       END LOOP;
    END LOOP;

-- NrofCell : Integer-Array with nx*ny*nz Elements.
-- Eds, Hds : Real-Arrays with (3 * Number of dielectric Cells) Elements.
-- Celltype : Integer-Array with (Number of dielectric Cells) Elements.
-- Hf1, Hf2, Ef1, Ef2 : FD-Coefficients. Real-Arrays with typically
--                      less than 30000 Elements.
```

Single Sweep through Memory

```
   FOR iz IN Block.iz1 .. Block.iz2 LOOP
      FOR iy IN Block.iy1 .. Block.iy2 LOOP
         FOR ix IN Block.ix1 .. Block.ix2 LOOP
            i:= NrofCell(ix,iy,iz);
            IF i>0 THEN                -- Dielectric Cell.
               iType:= CellType(i);    -- Index of the FD-Coefficients of the Cell
               ixp1:= NrofCell(ix+1,iy  ,iz  );  -- Number of Neighbour in +x.
               iyp1:= NrofCell(ix  ,iy+1,iz  );
               izp1:= NrofCell(ix  ,iy  ,iz+1);
Hds(1,i):= Hf1(1,iType) * Hds(1,i) - Hf2(1,iType) * ( Eds(3,iyp1) - Eds(3,i) - Eds(2,izp1) + Eds(2,i) );
Hds(2,i):= Hf1(2,iType) * Hds(2,i) - Hf2(2,iType) * ( Eds(1,izp1) - Eds(1,i) - Eds(3,ixp1) + Eds(3,i) );
Hds(3,i):= Hf1(3,iType) * Hds(3,i) - Hf2(3,iType) * ( Eds(2,ixp1) - Eds(2,i) - Eds(1,iyp1) + Eds(1,i) );
-- One may immediately update the Eds of that very Cell:
-- All referenced Hds-Values are already at the proper Time level.
-- None of the updated Eds Values will be used to compute the Hds of the remaining Cells.
-- When the Blocksize is small enough, all these Field-Accesses are fulfilled from the Cache.
               ixm1:= NrofCell(ix-1,iy  ,iz  );
               iym1:= NrofCell(ix  ,iy-1,iz  );
               izm1:= NrofCell(ix  ,iy  ,iz-1);
Eds(1,i):= Ef1(1,iType) * Eds(1,i) + Ef2(1,iType) * ( Hds(3,i)-Hds(2,i)+Hds(2,izm1)-Hds(3,iym1) );
Eds(2,i):= Ef1(2,iType) * Eds(2,i) + Ef2(2,iType) * ( Hds(1,i)-Hds(3,i)+Hds(3,ixm1)-Hds(1,izm1) )
Eds(3,i):= Ef1(3,iType) * Eds(3,i) + Ef2(3,iType) * ( Hds(2,i)-Hds(1,i)+Hds(1,iym1)-Hds(2,ixm1) )
            END IF;
         END LOOP; END LOOP; END LOOP;
```



Wall Currents in a 19 Cells CLiC-Section.
1000 Million Cells in the Box. 21 GBytes.
Textbook FDTD: 144 GBytes.
16 % are interesting. Device Length: 0.26 Metre.
Device-Description via CAD-Files with 90720 Triangles.
Meshing and FD-Coefficients is done in 46 Minutes.
Wakepotentials up to s=10 cm: add 30 Minutes.
For each Metre of Wakepotential: add 75 Minutes.
10000 Timesteps in 75 Minutes => Net 360 MCells / Second. Gross 2200 MCells/Second
2 Metres of Wakepotential in 3 Hours after Start.
Impedance boundary Conditions at all metallic Surfaces.
Dispersive Material in 5% of the processed Cells.
Napoly Integration is applied.

The Times refer to a 4 Socket Opteron 6370P Server, total Cost 7000 Euro, using 32 Cores at 2 GHz. On that Server possible: 12000 Million Gridcells: 190 GByte, 3.45 Seconds per Timestep, Gross 3500 MCells/s. Net 560 MCells/s.

I thank you for your Attention.