電磁界シミュレーターOpenFDTD利用講習会

主催：公益財団法人計算科学振興財団

共催：一般財団法人高度情報科学技術研究機構

株式会社EEM　大賀明夫

2022年3月2日

目次

1. OpenFDTDとは

● 電磁界シミュレーターOpenFDTD[1]
・電磁界シミュレーターをたくさんの人に使ってもらうために公開した（社会的インフラ）
・フリーソフト：無料で機能制限なし
・オープンソース（使用言語：C）
・動作環境：Windows（簡易GUIと実行プログラム付）またはLinux（ソースコードからコンパイルする）
・計算手法はFDTD法（時間領域差分法）、汎用的な電磁界シミュレーター
・各種の高速化技術を取り入れ大規模問題に使用することを想定している
・計算方法、使用方法の詳細についてはホームページで公開している
・2014年5月初版リリース

●OSSの利点
・ライセンスの制限がない（いつでもどこでも何人でも使える）
・自宅のPCにグラボを刺せばミニスパコン環境が実現、自宅でシミュレーション作業
・スパコン環境で自分でコンパイルして実行できる（商用ソフトでは難しい、将来は商用ソフトもスパコンで課金利用？）
・必要な機能は自分で改変可能

OpenFDTDの用途

Maxwell方程式で記述できる現象はすべて解析対象になる
地球規模からナノスケールまで（10^{6}m～10^{-9}m）15桁に及ぶ

●電波応用
1. 各種アンテナの設計
2. 携帯端末、携帯基地局アンテナ
3. 放送用送信アンテナ
4. ワイヤレス給電（電力伝送）
5. 屋内の電波伝搬解析（wifi、ローカル5G）
6. 交通機関のwifiサービス（新幹線、航空機、電車）
7. ミリ波、テラヘルツ波アンテナ（5G/6G）
8. レーダー断面積（軍事用、車載レーダー）
9. 車載アンテナ（車車間通信、自動運転）
10. メタマテリアル
11. 人体の電波被爆評価

●光応用
1. 金銀ナノ粒子の近接場光学
2. フォトニック結晶光導波路
3. SPR（表面プラズモン共鳴）
4. TERS（先端増強ラマン散乱顕微鏡）
5. SNOM（走査型近接場光学顕微鏡）
6. ナノセンサー（化学物質等）

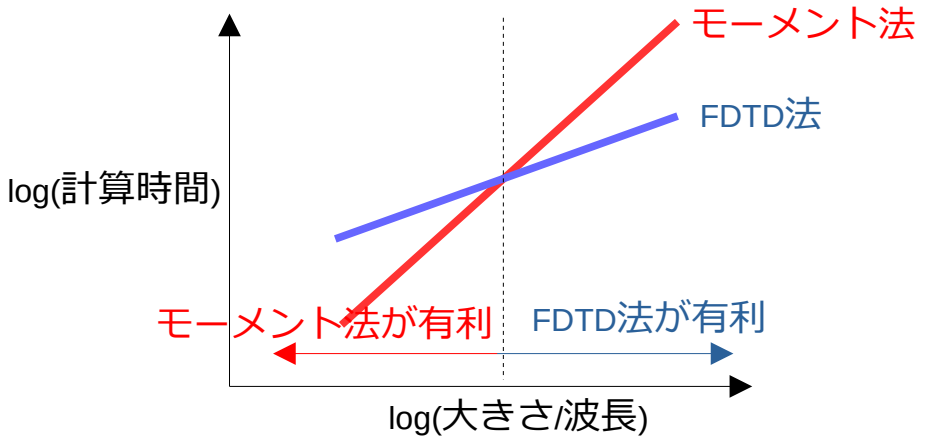
電磁界シミュレーターの歴史

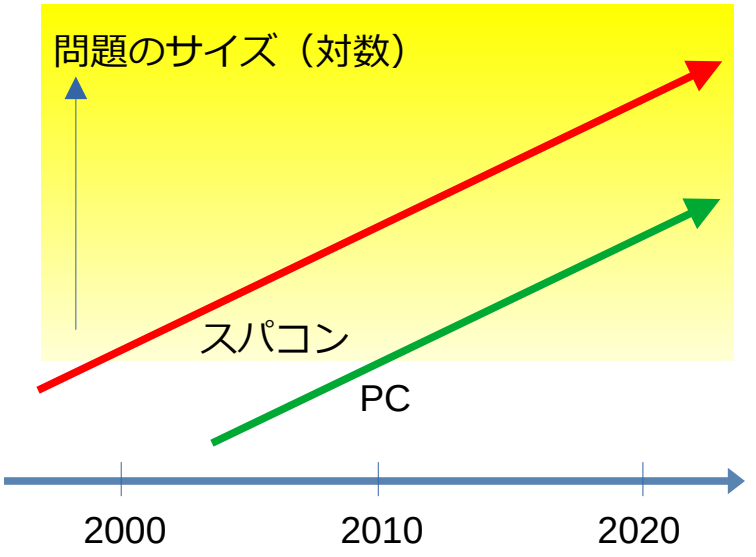
FDTD法

1966 K.S.Yee[2], FDTD法を最初に発表(2次元)
1975 A.Taflove[3], 3次元解析
1981 G.Mur[4], 吸収境界条件(開放領域)
1993 K.Kunz and R.J.Luebbers[5], FDTD法教科書(XFDTDへ)
1994 J.-P.Berenger[6], PML(高精度吸収境界条件)
1995 A.Taflove[7], FDTD法教科書
1998 宇野[8], FDTD法教科書
2000～ 商用ソフトが多数販売され今日に至る

モーメント法

1965 K.K.Mei[9], 数値計算法を最初に発表
1968 R.F.Harrington[10], モーメント法教科書
1981 G.J.Burke and A.J.Poggio[11], NEC公開(OSS)
1990～ 商用ソフトが多数販売され今日に至る





2. FDTD法

●FDTD法とは何か

・Finite Difference Time Domain Method（時間領域差分法）

・Maxwell方程式の微分形式を時間領域と周波数領域で離散化して電磁界の時間変化を求める手法

・現在の電磁界シミュレーションでは一番広く使われている手法

・有限の大きさの計算領域を考え多数のセルに分割する

・得られた時間波形をフーリエ変換して周波数特性を求める

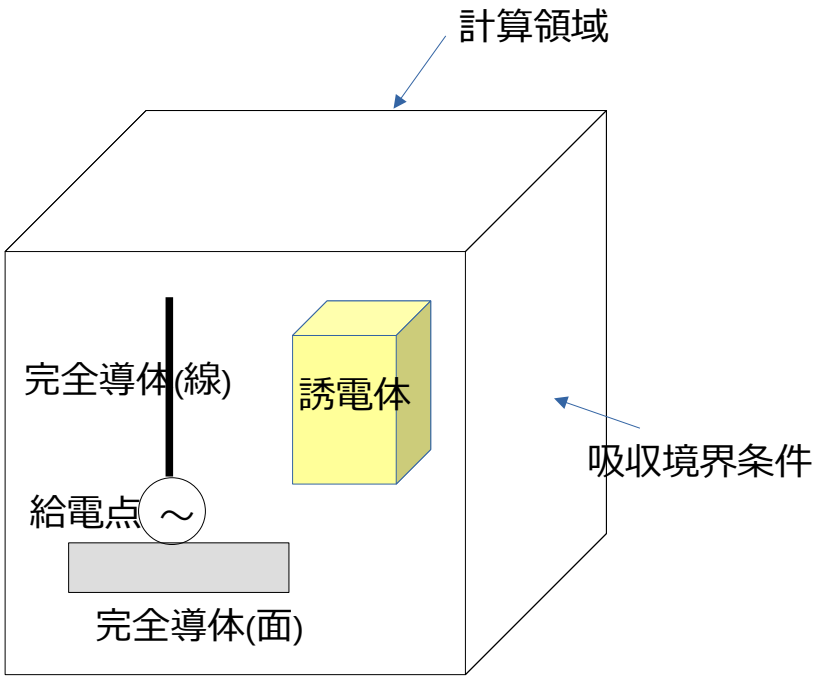
・磁界から電流を計算しこれから入力インピーダンスを求める

・計算領域の境界面の電界と磁界から遠方の放射パターンが計算される

・完全導体、誘電体、磁性体を扱うことができる

・電磁界のほとんどの用途に対応できる

・並列計算に適した計算手法であるために高速な計算機環境を生かすことができる



FDTD法の計算イメージ

Maxwell方程式とYee格子

Maxwell方程式

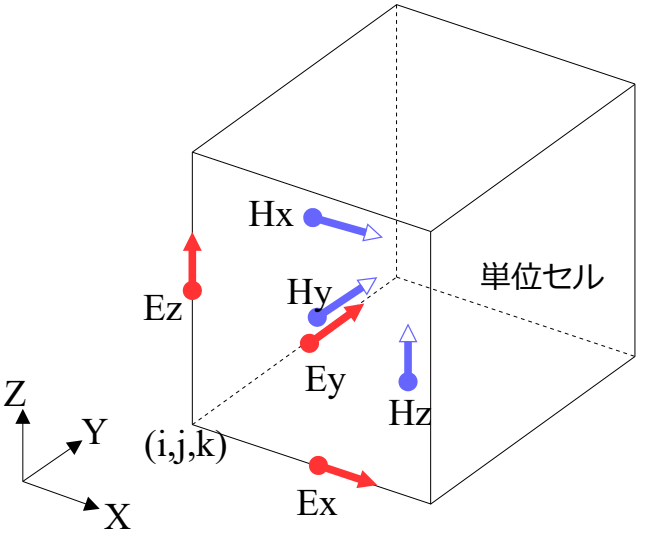
$$\nabla \times \mathbf{H} = \epsilon \frac{\partial \mathbf{E}}{\partial t} + \sigma_e \mathbf{E}$$

$$-\nabla \times \mathbf{E} = \mu \frac{\partial \mathbf{H}}{\partial t} + \sigma_m \mathbf{H}$$

■未知数
E:電界
H:磁界

■既知数
ε:誘電率
μ:透磁率
σ_e:導電率
σ_m:導磁率

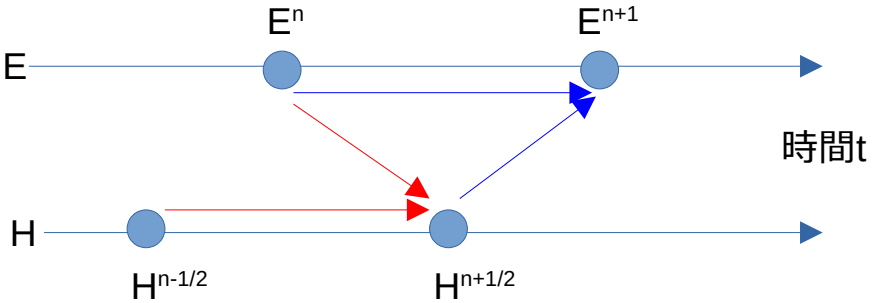
●Maxwell方程式の特長
・2つの方程式にすべての情報が含まれている
・補助方程式、実験式、経験式など一切なし
・これを解けばすべての電磁気の現象が説明できる



●Yee格子
・計算領域を多数のセルに分割する
・電界は稜線の中心の接線方向
・磁界は面の中心の法線方向
・Maxwell方程式と相性がよい

FDTD法の離散化

Maxwell方程式を時間領域と空間領域で離散化する
時間的に電界Eと磁界Hを交互に計算する(蛙跳び法、leap frog法)
電界と磁界のX成分は以下のようになる
c_1,c_2,d_1,d_2は電気定数から計算される既知の場所の関数
nはタイムステップ
Y成分、Z成分についてはX→Y→Zと巡回する
時間幅については上限（安定性条件）がある（陽解法）



$$E_x^{n+1}\left(i+\frac{1}{2},j,k\right)=c_1 E_x^n\left(i+\frac{1}{2},j,k\right)$$

$$+c_{2y}\left\{H_z^{n+\frac{1}{2}}\left(i+\frac{1}{2},j+\frac{1}{2},k\right)-H_z^{n+\frac{1}{2}}\left(i+\frac{1}{2},j-\frac{1}{2},k\right)\right\}-c_{2z}\left\{H_y^{n+\frac{1}{2}}\left(i+\frac{1}{2},j,k+\frac{1}{2}\right)-H_y^{n+\frac{1}{2}}\left(i+\frac{1}{2},j,k-\frac{1}{2}\right)\right\}$$

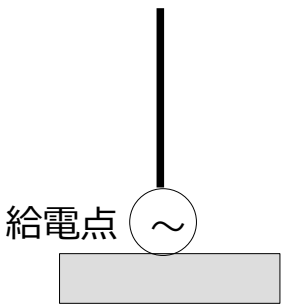
$$H_x^{n+\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right)=d_1 H_x^{n-\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right)$$

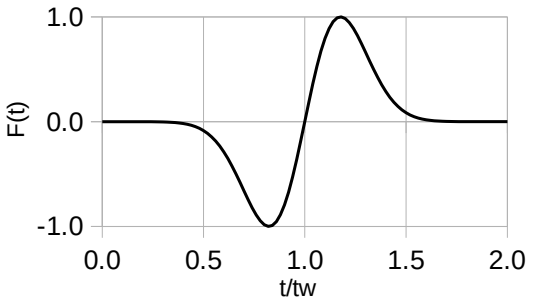
$$-d_{2y}\left\{E_z^n\left(i,j+1,k+\frac{1}{2}\right)-E_z^n\left(i,j,k+\frac{1}{2}\right)\right\}+d_{2z}\left\{E_y^n\left(i,j+\frac{1}{2},k+1\right)-E_y^n\left(i,j+\frac{1}{2},k\right)\right\}$$

波源モデル

・電磁界を発生させるためには何らかの波源が必要である
・波源として二つのモデルを考える

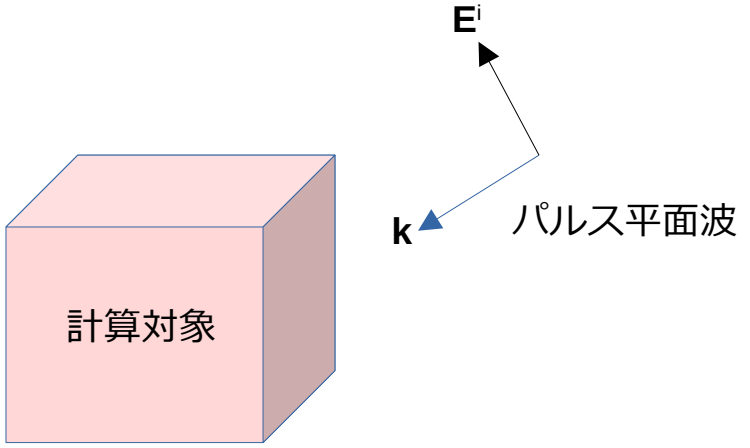
(1)給電点モデル（アンテナの計算）



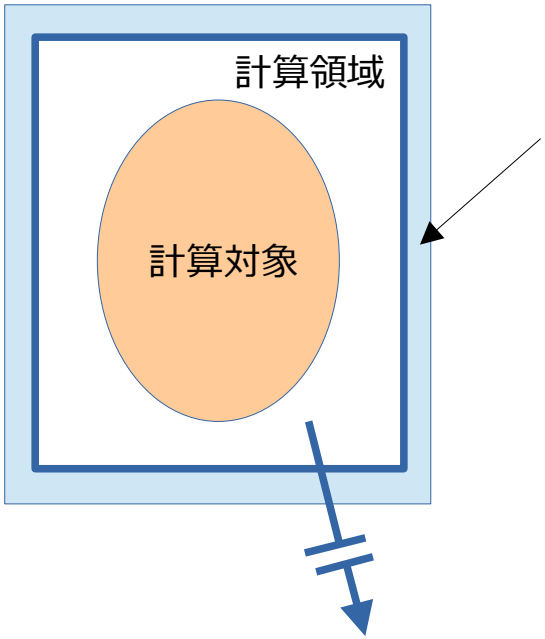


給電点にパルス電圧を印加する
（微分ガウスパルス）

(2)平面波入射モデル（散乱の計算）



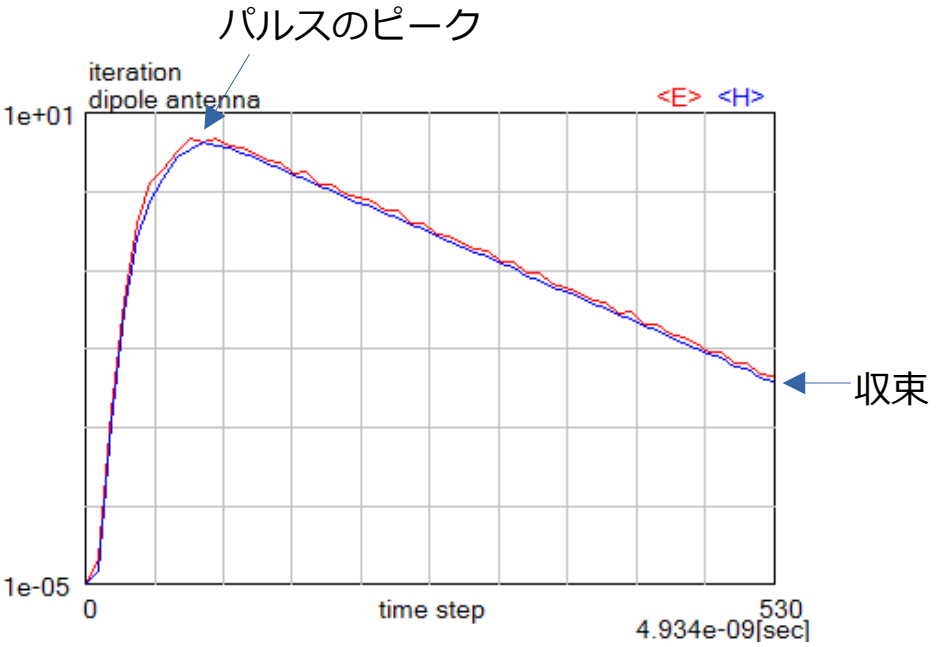
吸収境界条件



吸収境界条件：
無限に広がる電磁波を有限の領域で正しく計算するには計算領域の境界に吸収境界条件を設定することが必要
以下の2通りがある
(1)Mur一次：1層
(2)PML：5～10層（メモリーと計算時間が少し増えるが精度が向上する）

計算手順

- データ入力
- 準備作業
- 時間に関するループ
  - 磁界更新
  - 磁界に関する吸収境界条件
  - 電界更新
  - 電界に関する吸収境界条件(PMLのとき)
  - 給電電界設定
  - 近傍電磁界のFourier変換の加算
  - 収束判定(収束したらループを抜ける)
- 遠方界の計算
- 図形出力
- ファイル出力



収束状況：

・計算領域内の平均電界と平均磁界が十分小さくなったとき収束したとみなす（ピークの1/1000が目安）

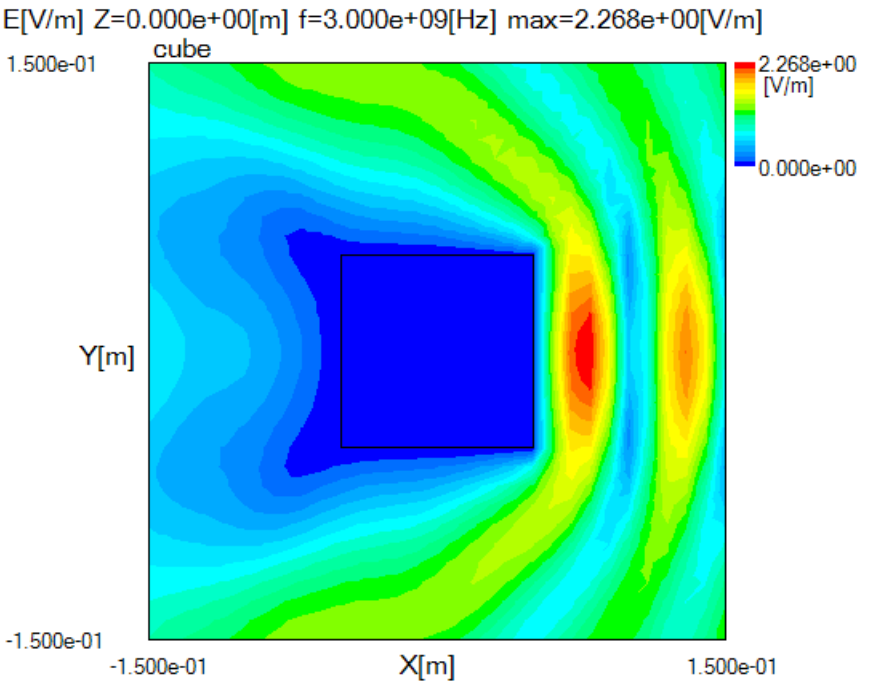
・収束が不十分なときは計算精度が落ちるので常に十分収束していることを確認すること

計算出力(1) 近傍電磁界

近傍電磁界：計算領域内の電磁界（周波数の関数）
電磁界の時間波形をFourier変換する

$$\boldsymbol{E}(\boldsymbol{r},\omega)=\frac{\int_0^\infty \boldsymbol{E}(\boldsymbol{r},t)e^{-j\omega t}dt}{\int_0^\infty V_{\text{in}}(t)e^{-j\omega t}dt}$$

$$\boldsymbol{H}(\boldsymbol{r},\omega)=\frac{\int_0^\infty \boldsymbol{H}(\boldsymbol{r},t)e^{-j\omega t}dt}{\int_0^\infty V_{\text{in}}(t)e^{-j\omega t}dt}$$



計算出力(2) 入力インピーダンス

入力インピーダンスと反射係数

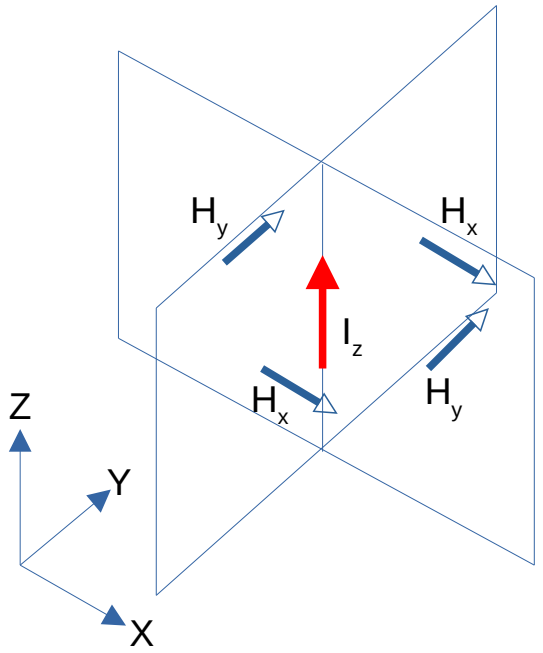
V_{in} : 給電点に加える電圧（既知数）

I_{in} : 給電点を流れる電流

Z_0 : 給電線の特性インピーダンス（既知数、同軸線路では50Ω）

$$Z_{in}(\omega)=\frac{V_{in}(\omega)}{I_{in}(\omega)} \quad \text{（入力インピーダンス）}$$

$$\Gamma(\omega)=\frac{Z_{in}(\omega)-Z_0}{Z_{in}(\omega)+Z_0} \quad \text{（反射係数）}$$



電流は磁界から計算される

$$I_z\left(i,j,k+\frac{1}{2}\right)=\Delta y_j\left\{H_y\left(i+\frac{1}{2},j,k+\frac{1}{2}\right)-H_y\left(i-\frac{1}{2},j,k+\frac{1}{2}\right)\right\}-\Delta x_i\left\{H_x\left(i,j+\frac{1}{2},k+\frac{1}{2}\right)-H_x\left(i,j-\frac{1}{2},k+\frac{1}{2}\right)\right\}$$

計算出力(3) 遠方界の計算法

遠方界：計算領域表面の電界と磁界の接線成分から積分で計算される

$$E_{\{\theta,\phi\}}(r,\theta,\phi)=\frac{-jk\,e^{-jkr}}{4\pi r}F_{\{\theta,\phi\}}(\theta,\phi)$$

$$F_\theta(\theta,\phi)=\eta N_\theta(\theta,\phi)+L_\phi(\theta,\phi)$$

$$F_\phi(\theta,\phi)=\eta N_\phi(\theta,\phi)-L_\theta(\theta,\phi)$$

$$\boldsymbol{N}(\theta,\phi)=\int_S \boldsymbol{J}(\boldsymbol{r})\exp(j\,k\,\boldsymbol{r}\cdot\hat{r})\,dS$$

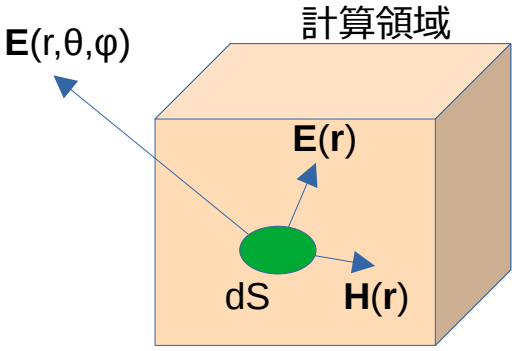
$$\boldsymbol{L}(\theta,\phi)=\int_S \boldsymbol{M}(\boldsymbol{r})\exp(j\,k\,\boldsymbol{r}\cdot\hat{r})\,dS$$

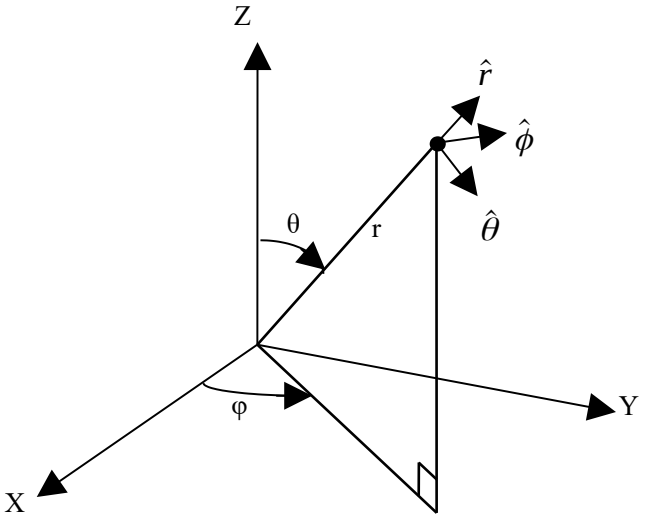
$$\boldsymbol{J}(\boldsymbol{r})=\hat{n}\times\boldsymbol{H}(\boldsymbol{r})$$

$$\boldsymbol{M}(\boldsymbol{r})=-\hat{n}\times\boldsymbol{E}(\boldsymbol{r})$$

$$\eta=\sqrt{\frac{\mu_0}{\epsilon_0}}\approx 120\pi\ \ [\Omega]$$

$$k=\omega\sqrt{\epsilon_0\mu_0}$$





極座標

計算出力(4) 遠方界特性

アンテナ：利得

$$G(\theta,\phi)=\frac{1}{P_{\mathrm{in}}}\frac{4\pi r^2}{2\eta}\left\{|E_\theta(r,\theta,\phi)|^2+|E_\phi(r,\theta,\phi)|^2\right\}$$
$$=\frac{k^2}{8\pi\eta P_{\mathrm{in}}}\left\{|F_\theta(\theta,\phi)|^2+|F_\phi(\theta,\phi)|^2\right\}$$

$$P_{\mathrm{in}}=\frac{1}{2}\mathrm{Re}\left(V_{\mathrm{in}}I_{\mathrm{in}}^*\right)$$

平面波入射：散乱断面積

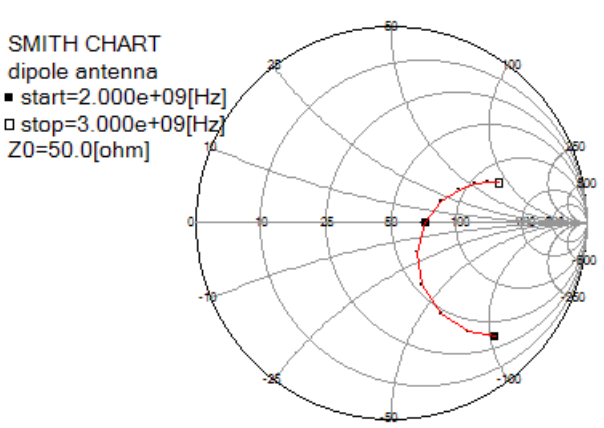
$$\sigma(\theta,\phi)=4\pi\,\mathrm{r}^2\left\{|\mathrm{E}_\theta(\mathrm{r},\theta,\phi)|^2+|\mathrm{E}_\phi(\mathrm{r},\theta,\phi)|^2\right\}$$
$$=\frac{\mathrm{k}_0^2}{4\pi}\left\{|\mathrm{F}_\theta(\theta,\phi)|^2+|\mathrm{F}_\phi(\theta,\phi)|^2\right\}$$

$\sigma_{\mathrm{F}}=\sigma(\pi-\theta^{\mathrm{i}},\pi+\phi^{\mathrm{i}})$　（前方散乱断面積）

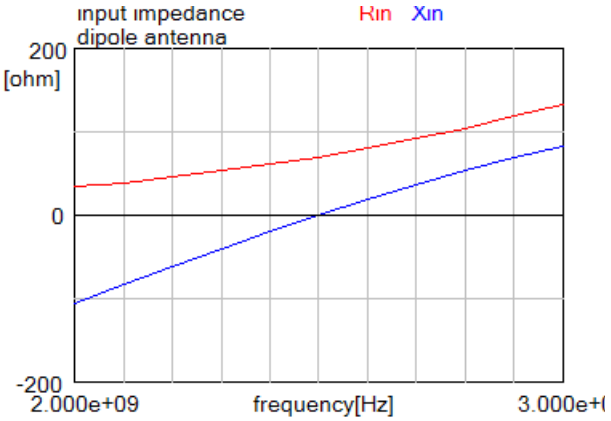
$\sigma_{\mathrm{B}}=\sigma(\theta^{\mathrm{i}},\phi^{\mathrm{i}})$　（後方散乱断面積：RCS）

$\sigma_{\mathrm{T}}=\frac{1}{4\pi}\int_0^{2\pi}\mathrm{d}\phi\int_0^{\pi}\sin\theta\,\mathrm{d}\theta\,\sigma(\theta,\phi)$　（全散乱断面積）

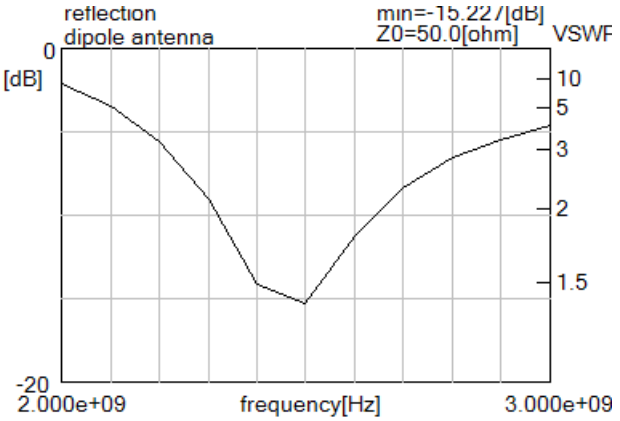
図形出力例



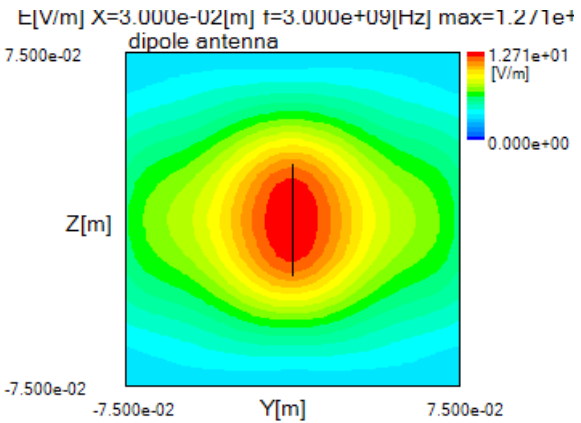
スミスチャート



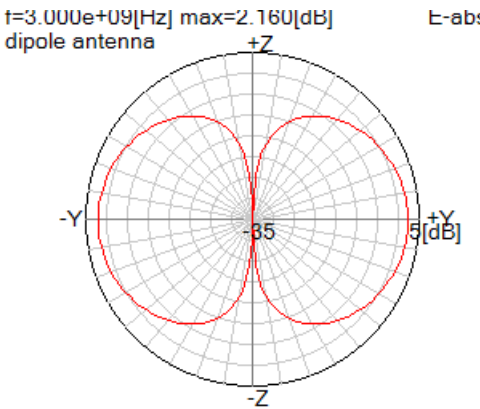
入力インピーダンス



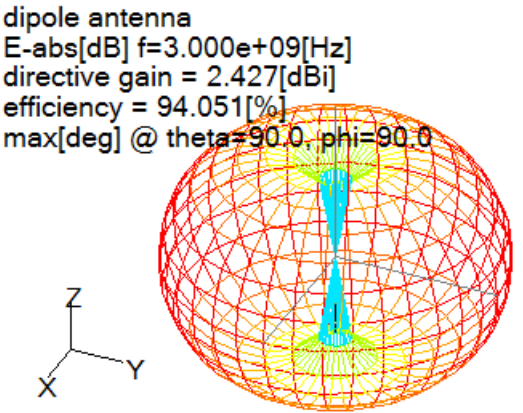
反射係数とVSWR



近傍界電界分布図



遠方界パターン図



遠方界全方向パターン図

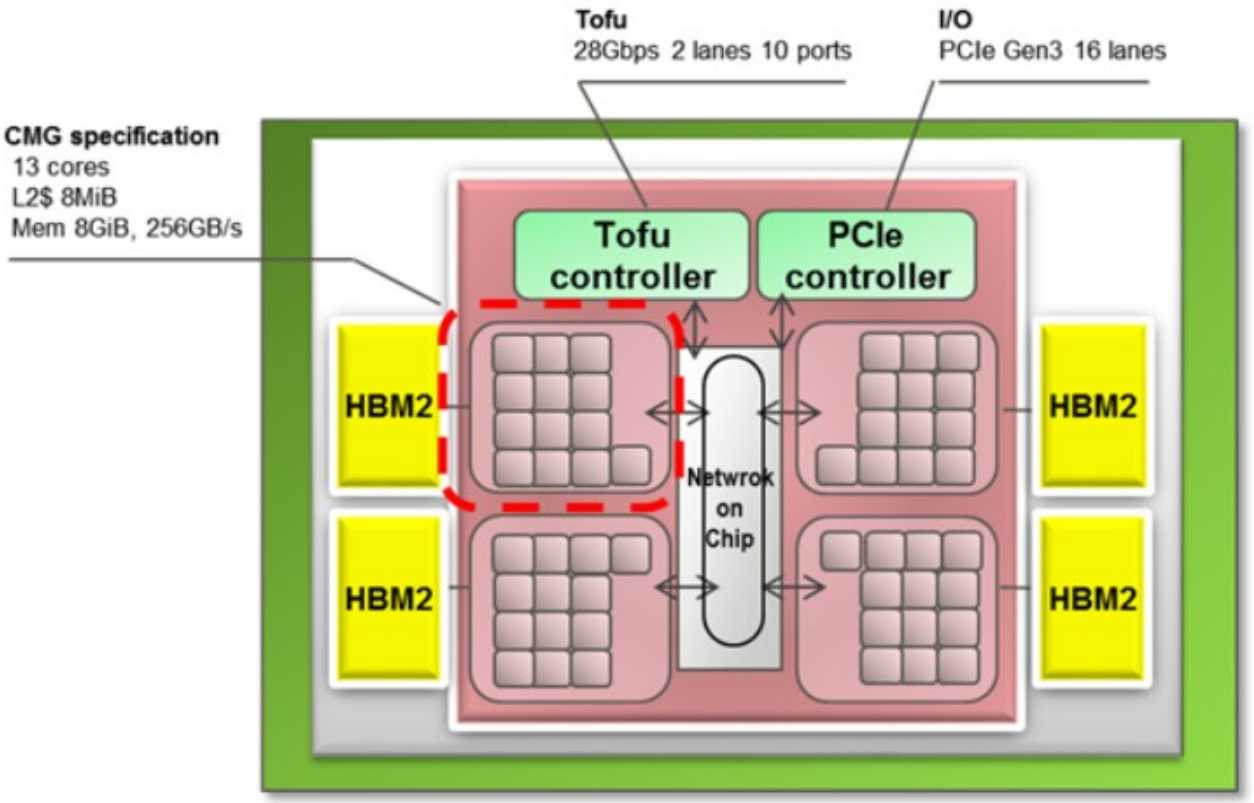
3. OpenFDTDの高速化技術

FXの仕様[12]

機種名		PRIMEHPC FX1000	PRIMEHPC FX700		
CPU	名称	A64FX	A64FX		
	命令セット アーキテクチャー	Armv8.2-A SVE	Armv8.2-A SVE		
	演算コア数	48 コア	48 コア		24 コア
	アシスタントコア数	計算ノード: 2コア IO兼計算ノード: 4コア	なし		なし
	クロック	2.2 GHz	1.8 GHz	2.0 GHz	2.6 GHz
	CPU理論演算性能	3.3792 TFLOPS (倍精度)	2.7648 TFLOPS (倍精度)	3.072 TFLOPS (倍精度)	1.9968 TFLOPS (倍精度)
	1コアあたりの 理論演算性能	70.4 GFLOPS (倍精度)	57.6 GFLOPS (倍精度)	64 GFLOPS (倍精度)	83.2 GFLOPS (倍精度)
ノード	アーキテクチャー	1 CPU/ノード	1 CPU/ノード		
	メモリ容量	32 GiB(HBM2、4スタック)	32 GiB(HBM2、4スタック)		
	メモリバンド幅	1,024 GB/s	1,024 GB/s		
	インターコネクト	TofuインターコネクトD	InfiniBand EDR / HDR100 (*1) (*1)EDRとHDR100の混在は不可		
本体装置	フォームファクタ	専用ラック	2Uラックマウントシャーシ		
	最大ノード数	384ノード/ラック	8ノード/シャーシ		
	冷却方式	水冷	空冷		
ソフトウェア	OS	Red Hat Enterprise Linux 8	Red Hat Enterprise Linux 8		
	HPCミドルウェア	FUJITSU Software Technical Computing Suite	• FUJITSU Software Compiler Package • FUJITSU Software Technical Computing Suite(*2) (*2)分散ファイルシステム(FEFS)のみサポート対象 • Bright Cluster Manager(Bright Computing社製) • Altair PBS Professional(Altair社製)		

2.2GHz×512/64×2(FMA)
=70.4GFLOPS
70.4GFLOPS×48
=3.3792TFLOPS

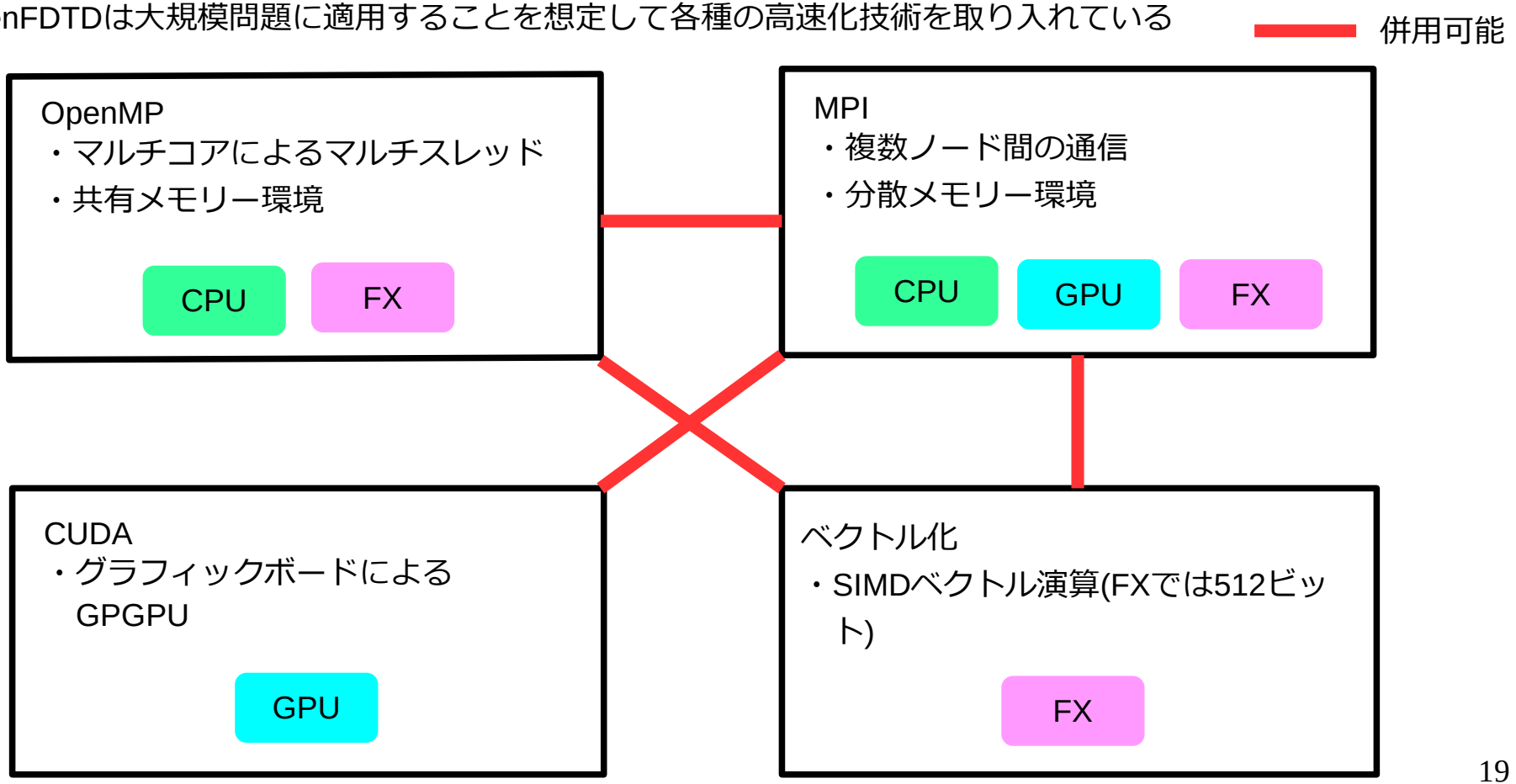
A64FX



「A64FX」 ブロック図

OpenFDTDの高速化技術

OpenFDTDは大規模問題に適用することを想定して各種の高速化技術を取り入れている

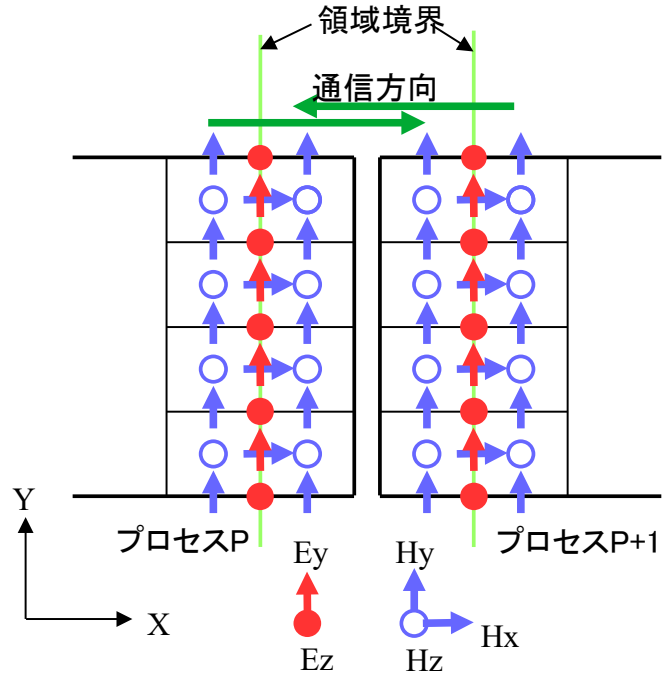


FX(A64FX)での高速化作業

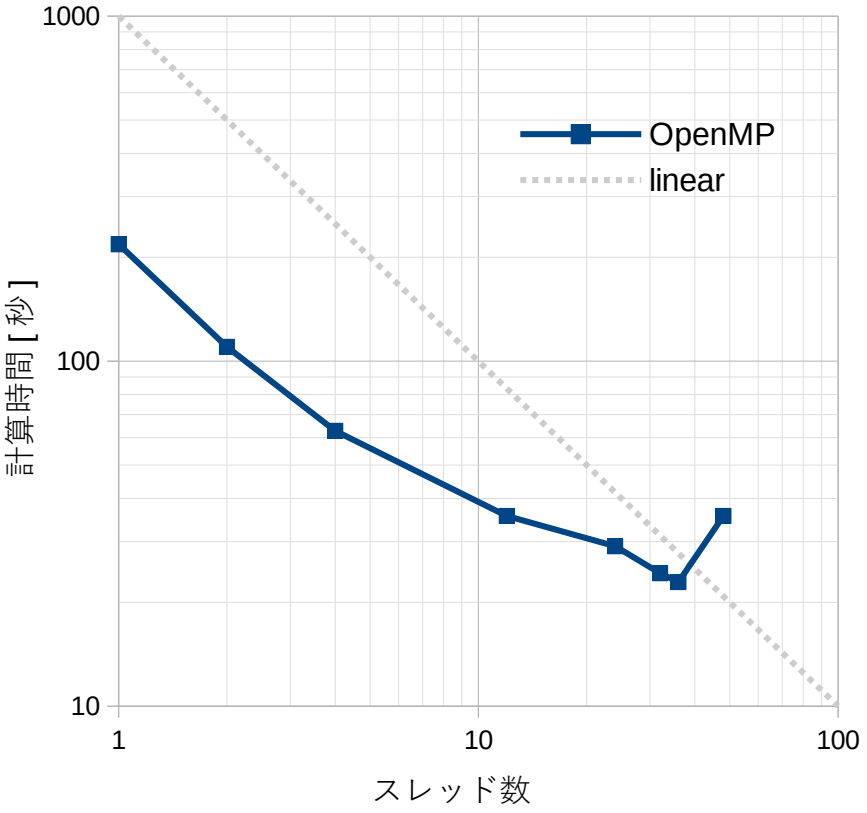
(1)ベクトル化
- 各コアは512ビットのSIMDベクトル演算機能を持っている
- コンパイラー(fcc)の自動ベクトル化機能を利用
- 一番内側のループをベクトル化する
- OpenFDTDは単精度計算であるから理想的には512/32=16倍速くなる

(2)OpenMP並列化 [13]
- A64FXの48コア内で並列計算する（共有メモリー）
- 一番外側のループを並列化する
- 計算の主要部のfor文の前にOpenMPディレクティブを記入
- コンパイラー(fcc)が並列コードを出力する

(3)MPI並列化 [14]
- 計算領域をプロセス数で分割する（分散メモリー）
- プロセス間の通信にはMPI関数を記述する（反復計算ごとに通信が必要）
- コンパイラー(mpifcc)が並列化コードを出力する



OpenMPによる並列計算



■OpenMPのスレッド数を変えたときの計算時間
（FX700、clangモード、ベンチマーク200）
・OpenMPのスレッド数を増やすと計算時間が短縮される
・10スレッドを超えると速度向上が頭打ちになる

MPIとOpenMPの併用

MPIプロセス数	OpenMPスレッド数	計算時間
1	48	407.2秒
2	24	854.3秒
3	16	18.6秒
4	12	7.9秒
6	8	53.0秒
8	6	137.0秒
12	4	188.6秒
24	2	585.5秒
48	1	1133.0秒

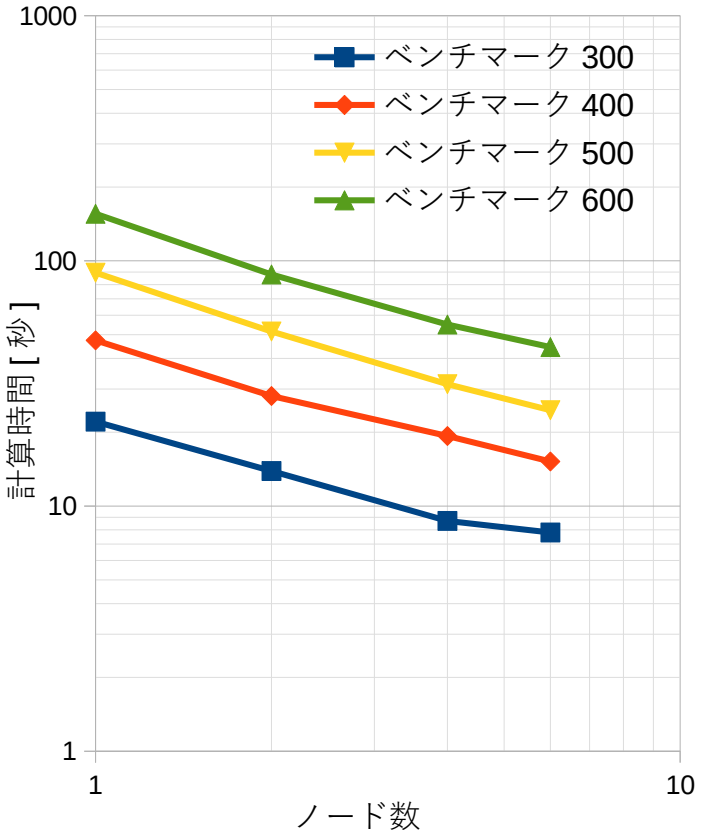
■MPIのプロセス数とOpenMPのスレッド数を変えたときの計算時間（積は48で一定）
（FX700、clangモード、ベンチマーク200）

・4×12のとき最も速く、他の組み合わせは非常に遅い（CPUのブロック図参考）

・1×48が前ページの48スレッドの10倍遅いのはMPIの実装に問題あり?

●以下では4×12を採用する [15]

複数ノードの計算時間



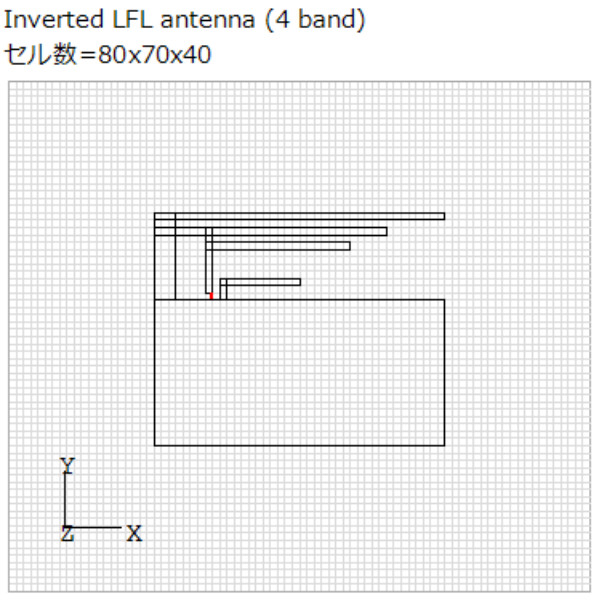
■ノード数を1～6と変えたときの計算時間
（FX700、clangモード、各ノードでは4×12）

●ノード数が増えると計算時間が短縮される

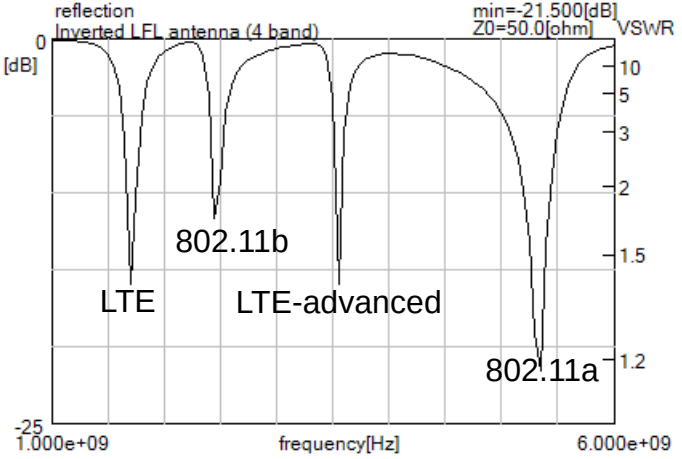
4. OpenFDTDの計算例

4周波数共用アンテナ [16]

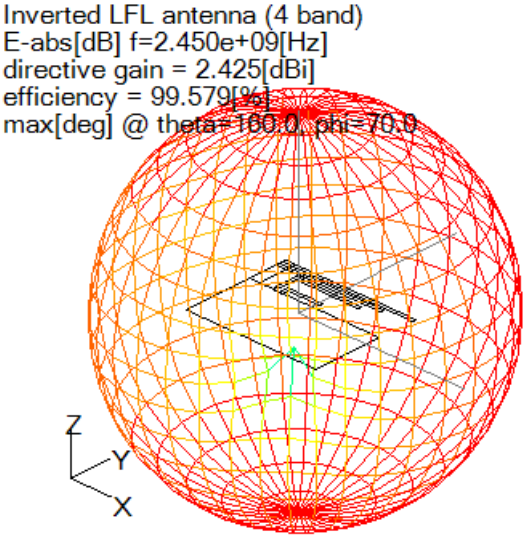
・スマホの小型化、電波干渉低減



アンテナ形状



反射係数周波数特性(1～6GHz)

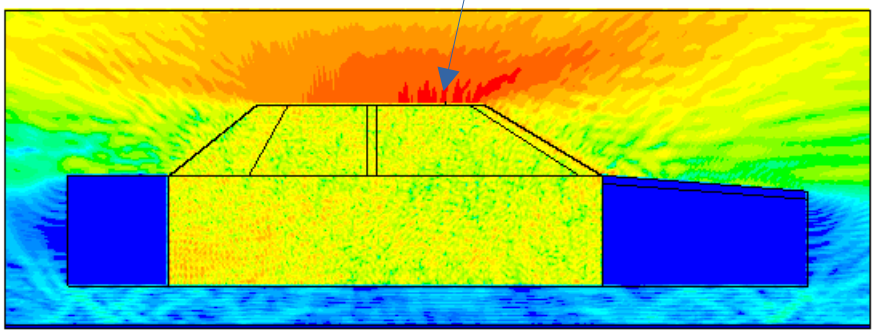


全方向放射パターン(2.45GHz)

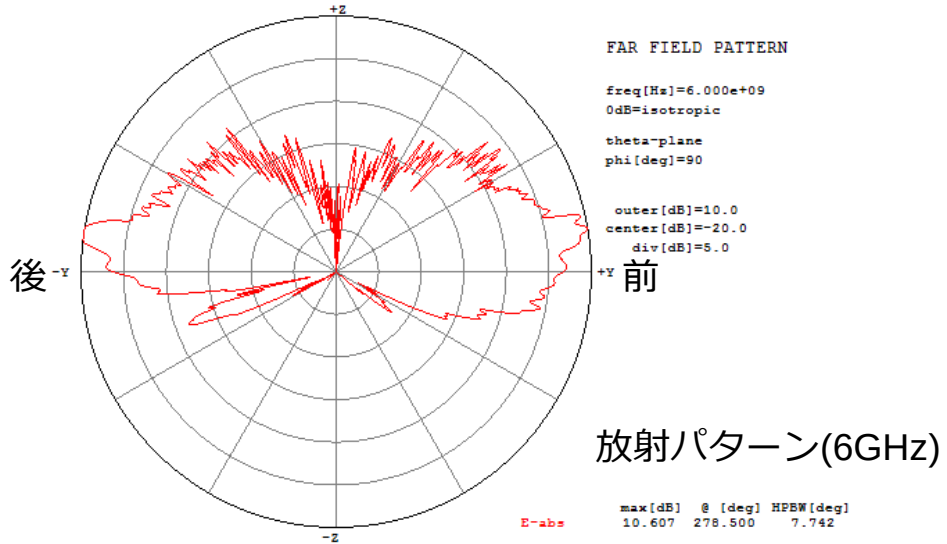
・指向性は低い

車載アンテナ

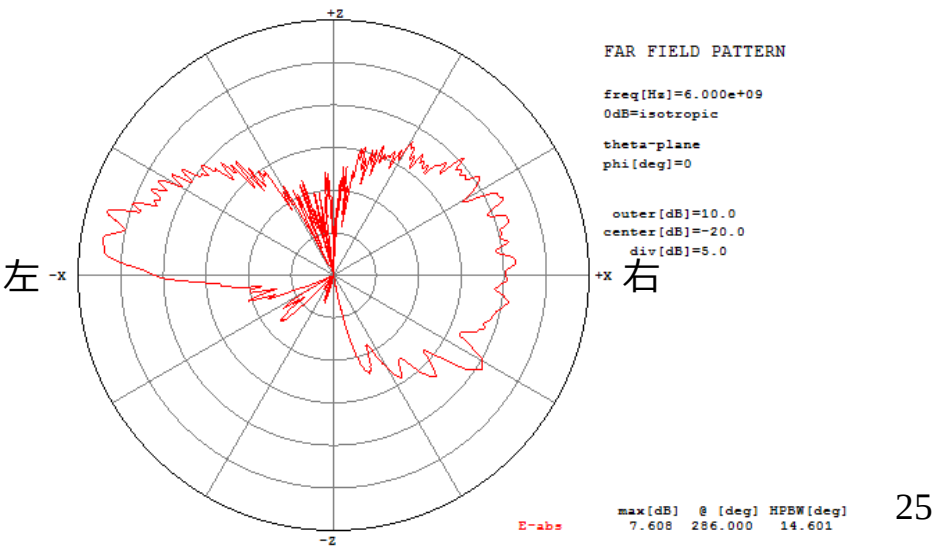
アンテナ(垂直モノポール、右端)



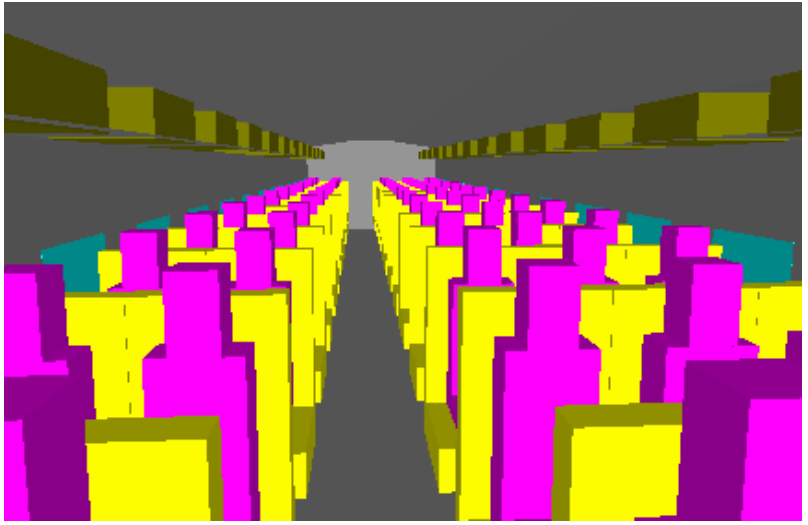
電界分布(6GHz、中心面)

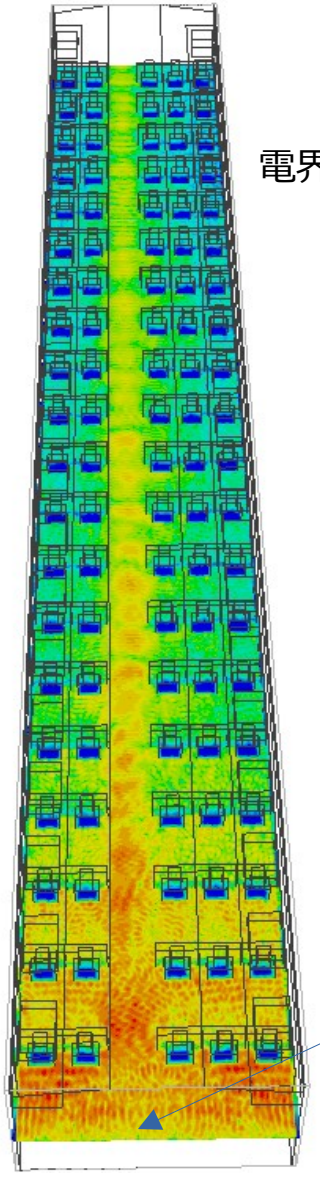


放射パターン(6GHz)



新幹線

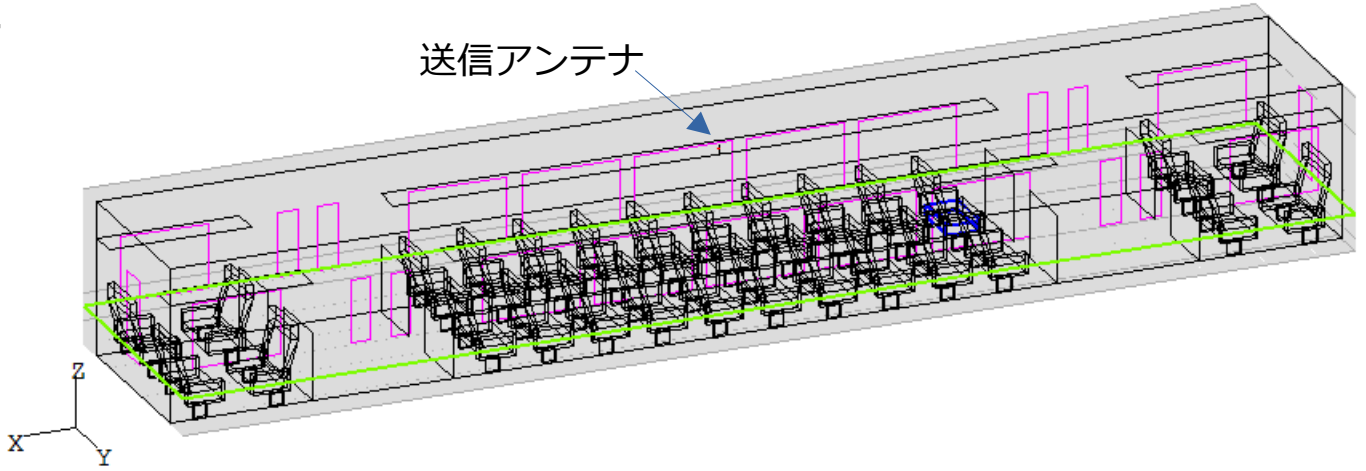


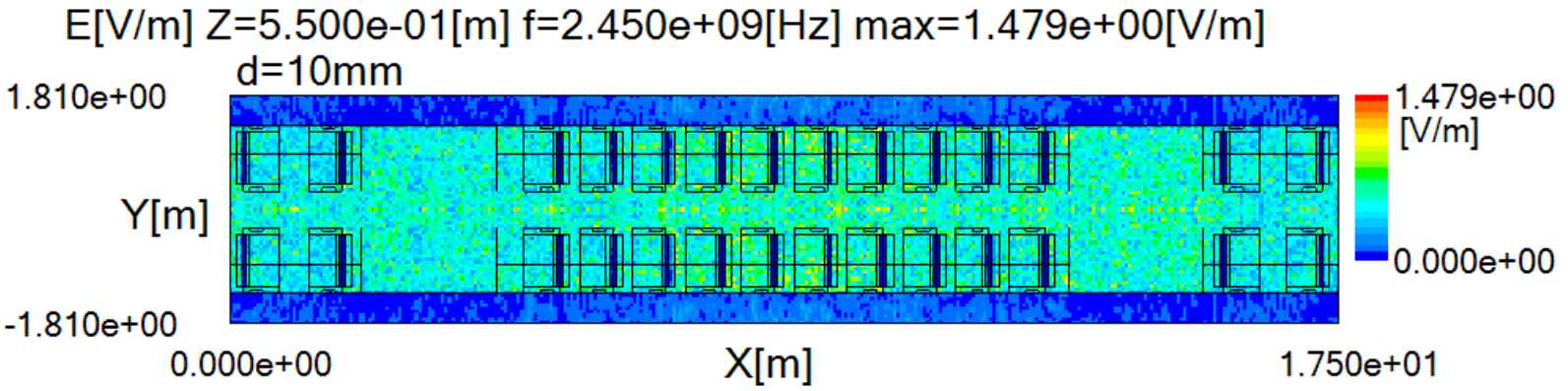


電界分布(2.45GHz, 床上80cm)

アクセスポイント(AP)

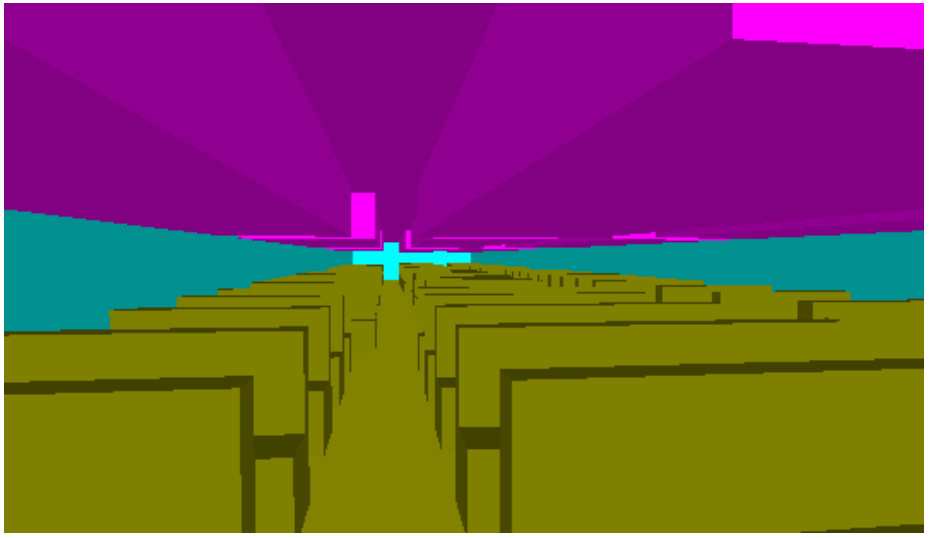
通勤電車

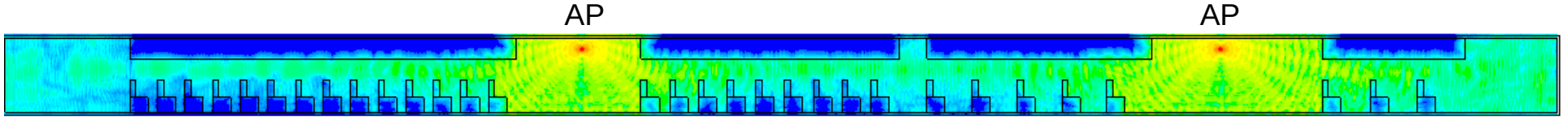




電界分布（2.45GHz、床上55cm）

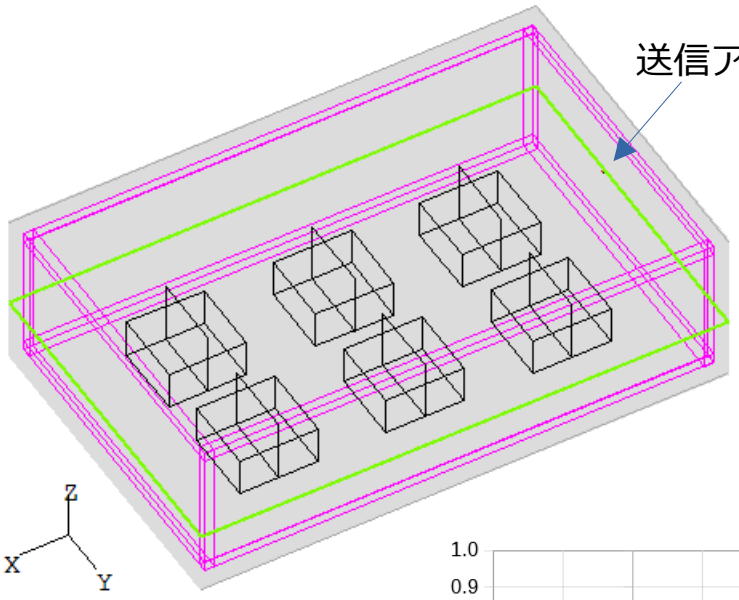
旅客機

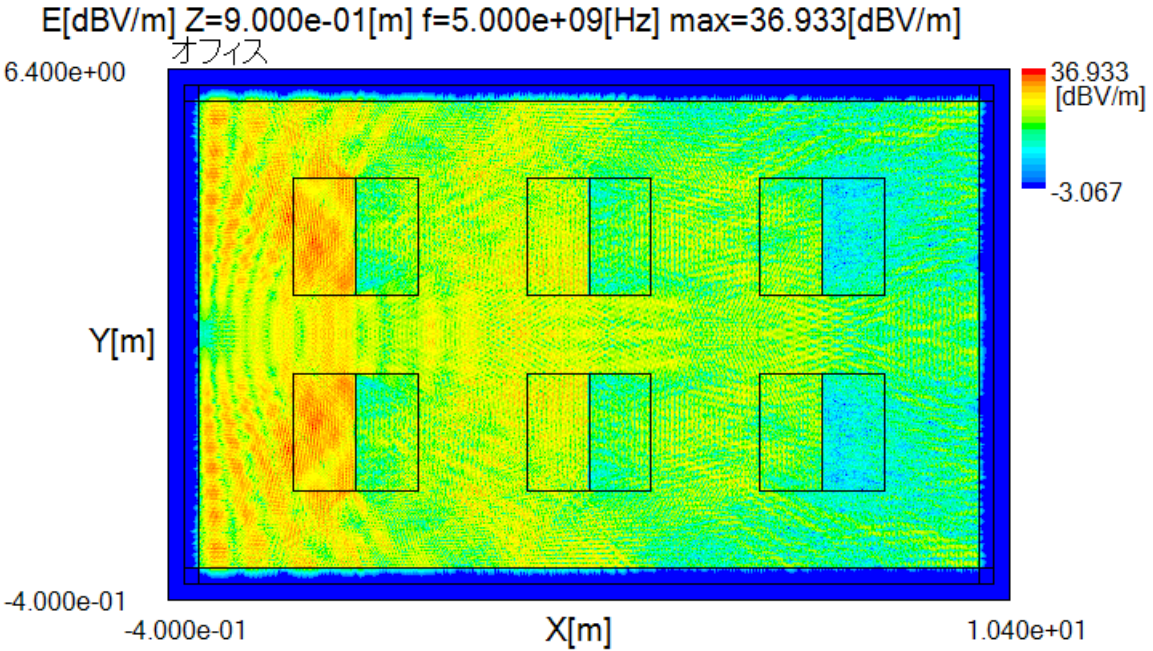




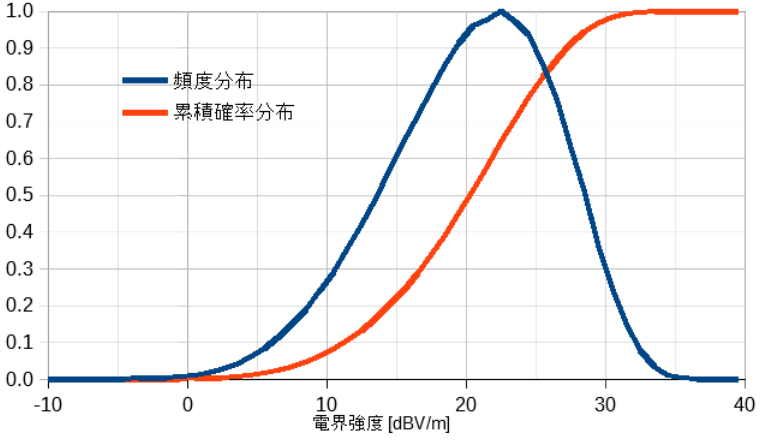
電界分布(1.5GHz)

オフィス



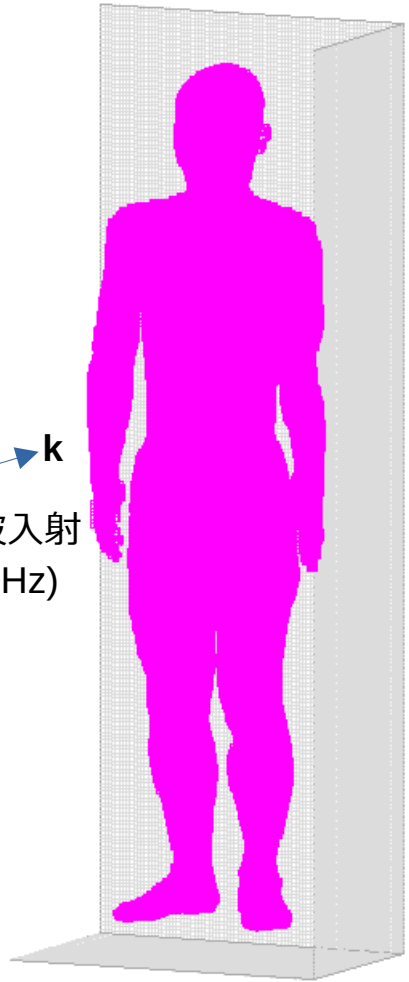


電界分布（5GHz、床上0.9m）



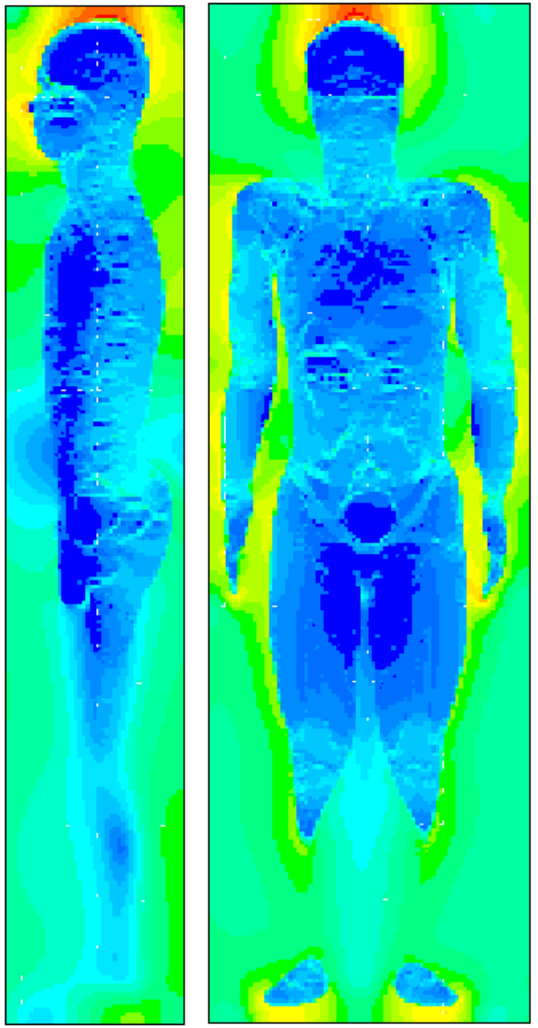
電界強度の頻度と累積確率分布

数値人体モデル

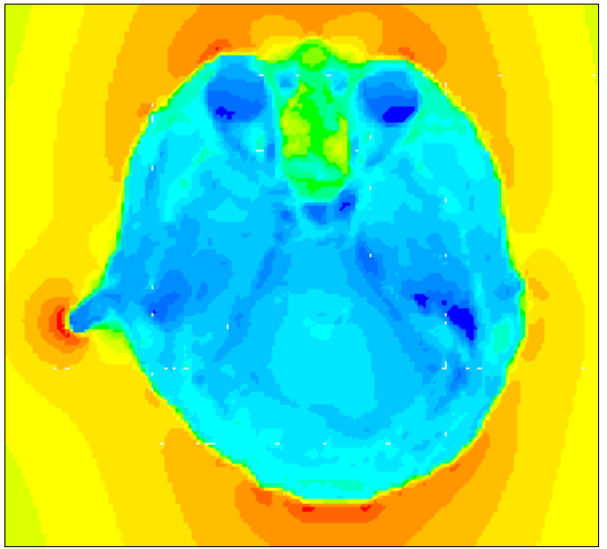


平面波入射
(100MHz)

電界分布



頭部断面



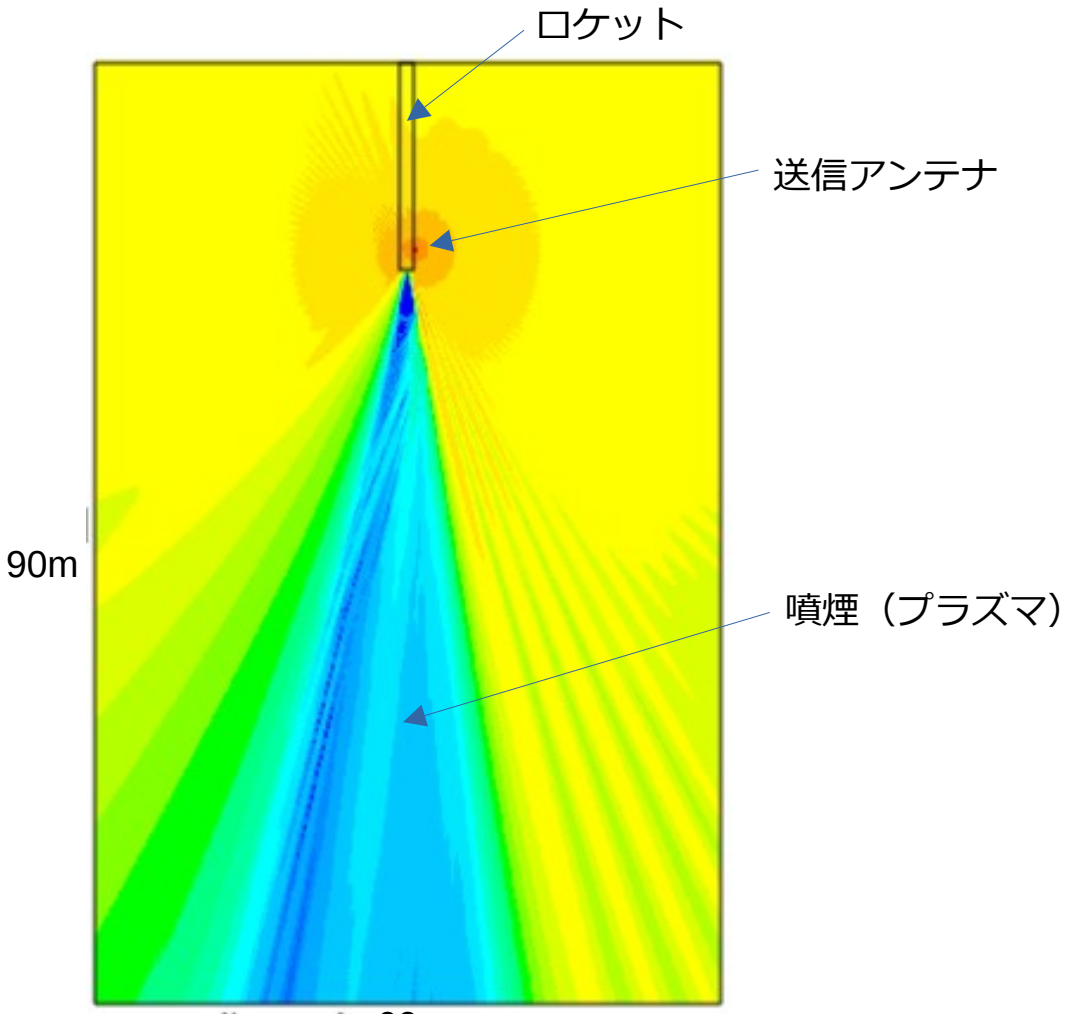
Brooksモデル
1mmセル

情報通信研究機構(NICT)
数値人体モデル(2mmセル、58組織)
http://emc.nict.go.jp/bio/data

人体組織誘電率DB(10Hz～100GHz)
http://niremf.ifac.cnr.it/tissprop/htmlclie/htmlclie.php

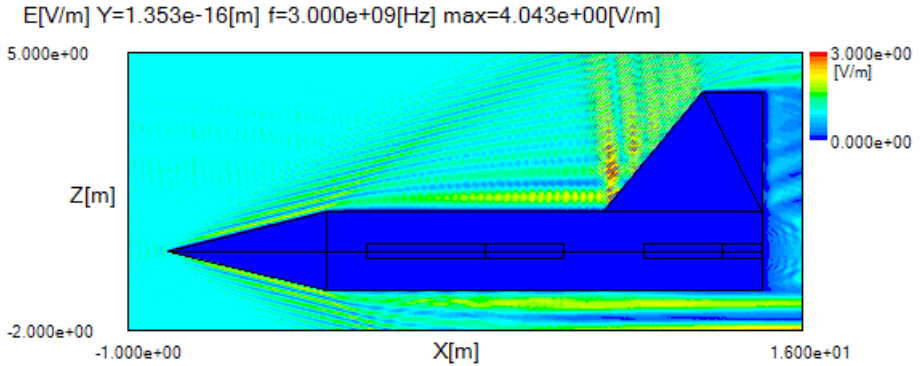
ロケットの噴煙

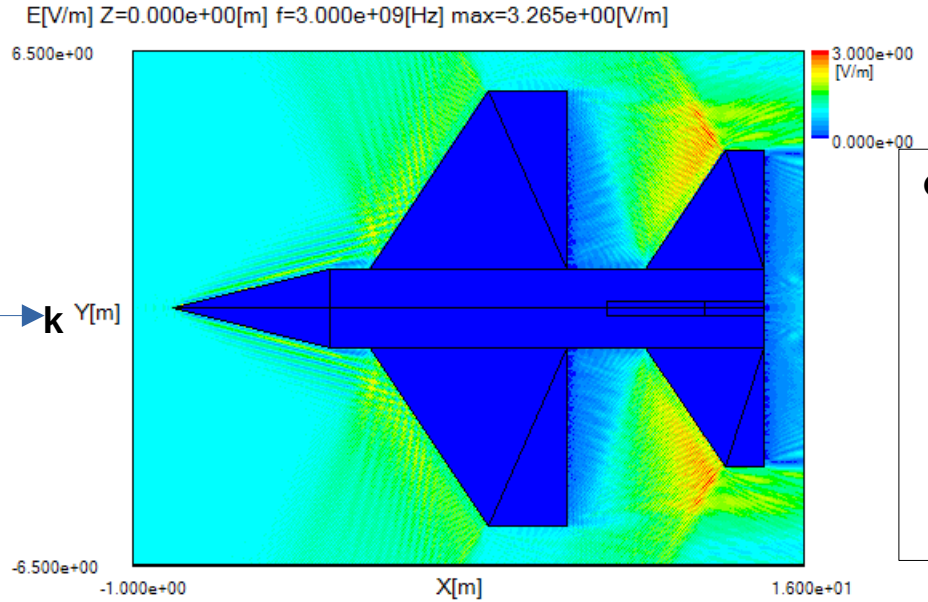
- 固体燃料ロケットの噴煙はプラズマ状態なので地上との通信に支障が出る
- 噴煙のプラズマパラメーターを元に電界分布とアンテナの放射パターンを計算する
- 実測値との比較から燃焼モデルについての知見が得られる
- 2次元モデルの計算：軸対称なので2次元モデルでよい予測ができる



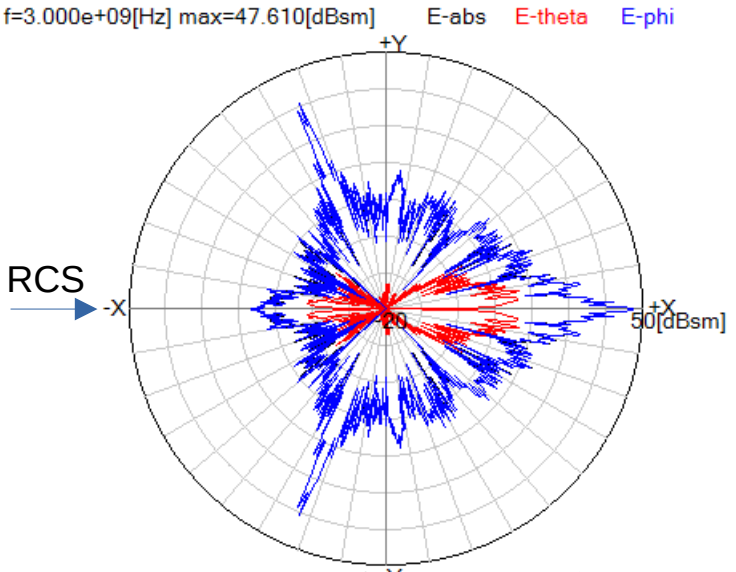
電界分布[dB]

レーダー断面積（RCS）





戦闘機の電界分布



散乱断面積パターン（Sバンド）

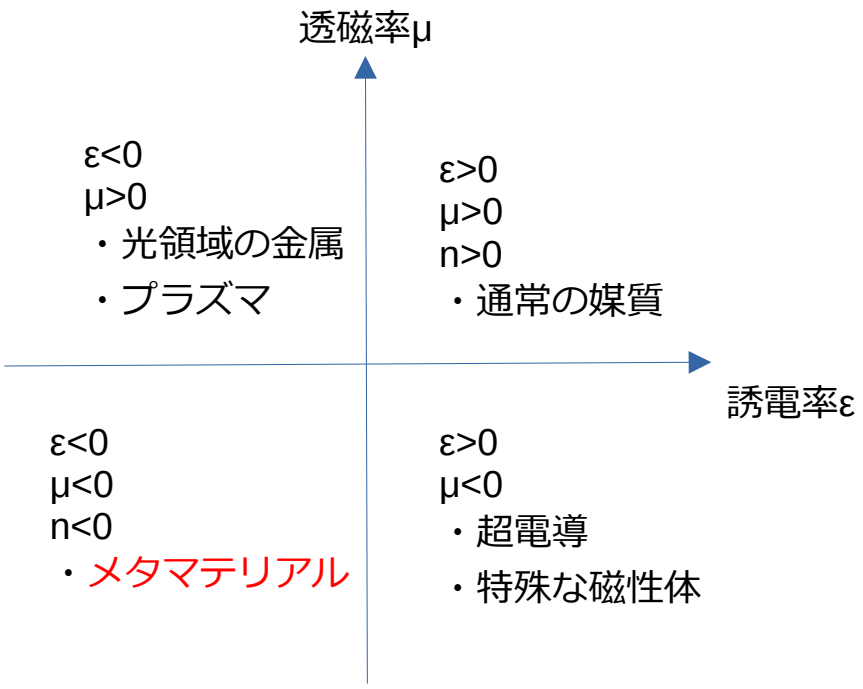
●レーダー断面積（Radar Cross Section：RCS）の計算

・実機と実周波数（Xバンド）の計算を行うには多くのメモリーと計算時間が必要になる
（メモリー6TB：富岳200ノードで計算時間1日）

・さらに全方向のRCS（=後方散乱断面積）を計算するには入射方向を変えて繰り返し計算する必要がある

・電波吸収体の計算も可能（ステルス性）

メタマテリアル



●メタマテリアルの歴史

V. G. Veselago [17], 1968. メタマテリアルを提唱

J. B. Pendry [18], 1996. 最初に実現

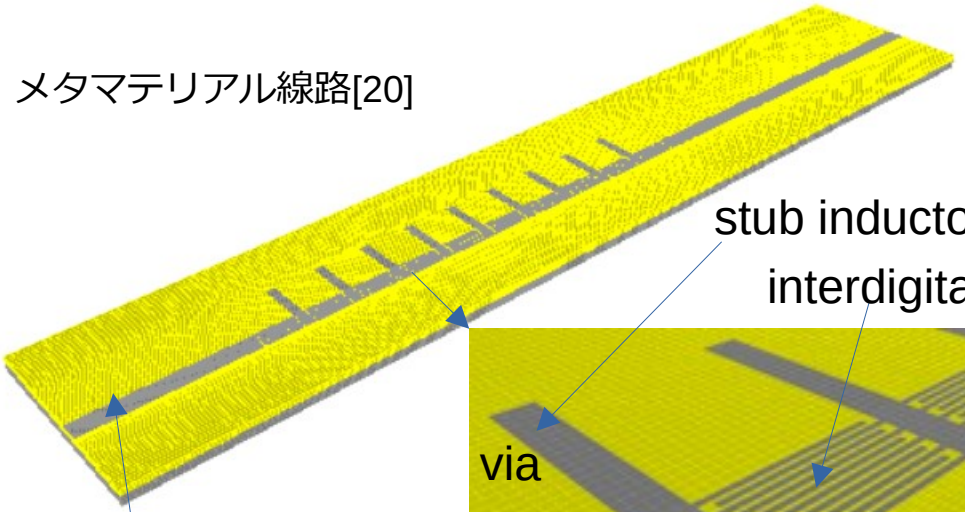
D.R.Smith [19], 2000. 負の屈折率を実現

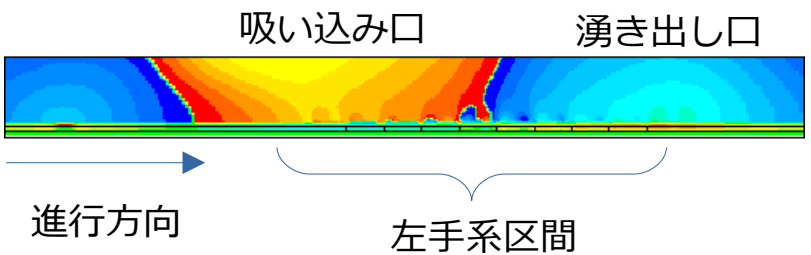
●メタマテリアルの応用 [16][20][21][24]

・超薄型アンテナ
・高機能アンテナ
・高性能電波吸収体
・メタサーフェス（電波伝搬制御）
・メタレンズ（負の屈折率）
・メタホログラム
・バイオセンサー
・クローキング（偽視）

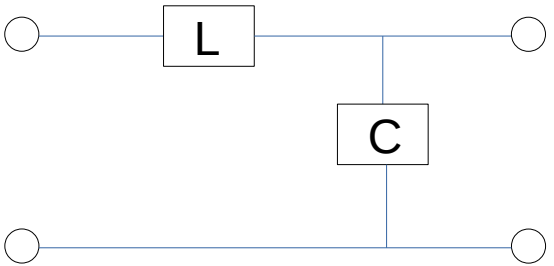
メタマテリアル線路

メタマテリアル線路[20]

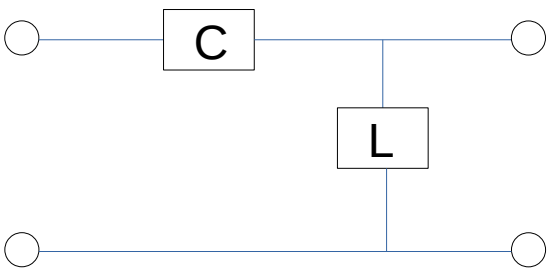




Ezの位相分布

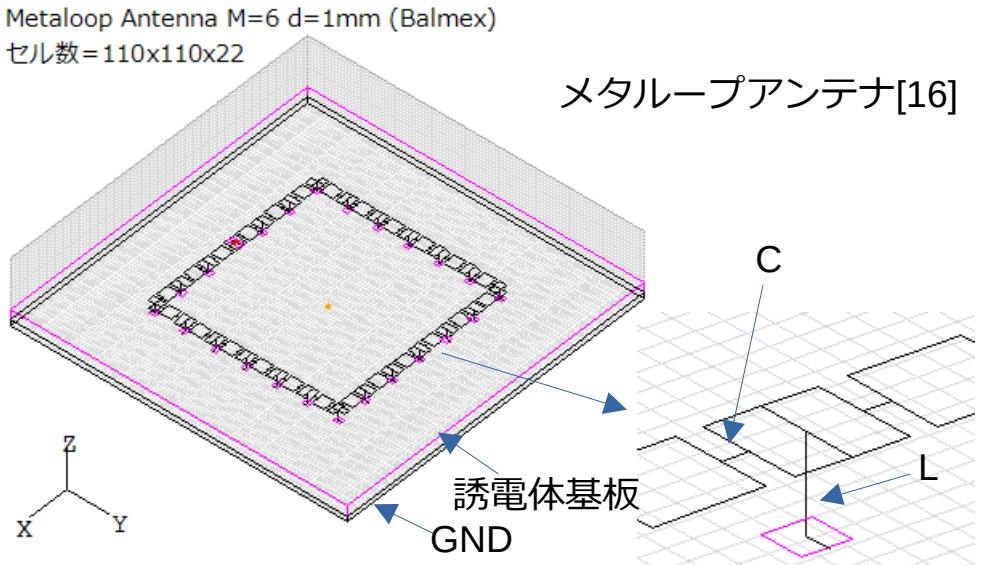


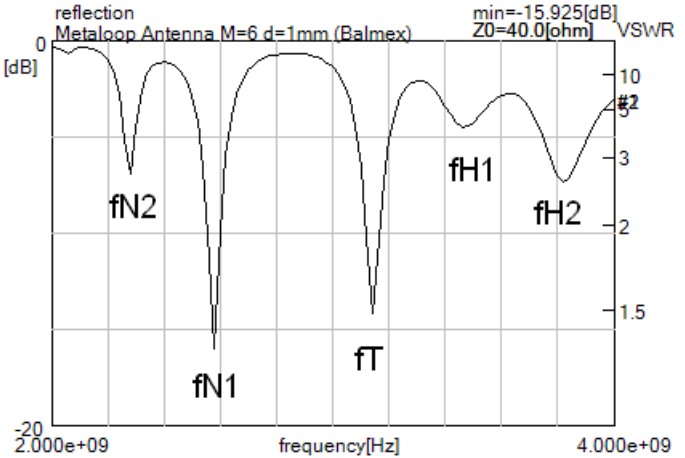
通常の線路（右手系）



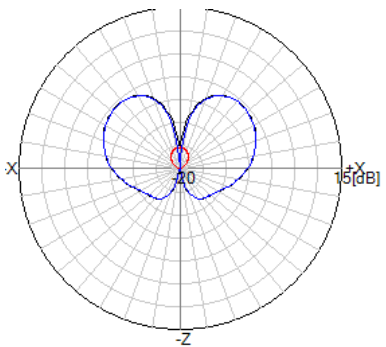
メタマテリアル線路（左手系）

メタマテリアルアンテナ

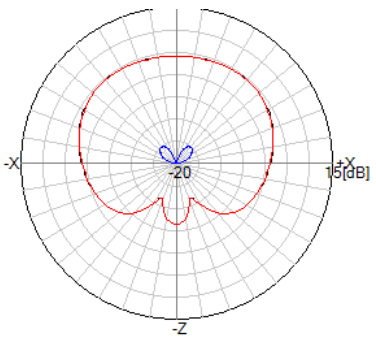




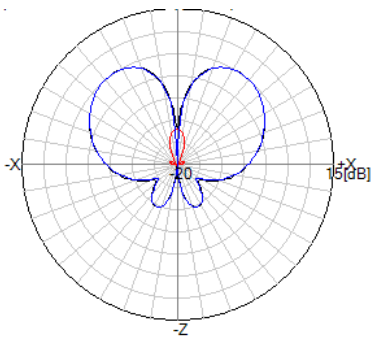
反射係数の周波数特性(2～4GHz)



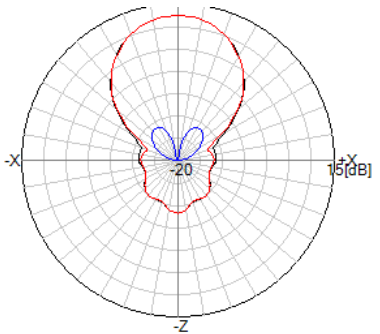
2.25GHz (f_{N2})



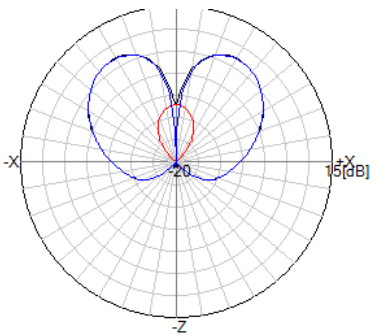
2.55GHz (f_{N1})



3.10GHz (f_T)



3.40GHz (f_{H1})



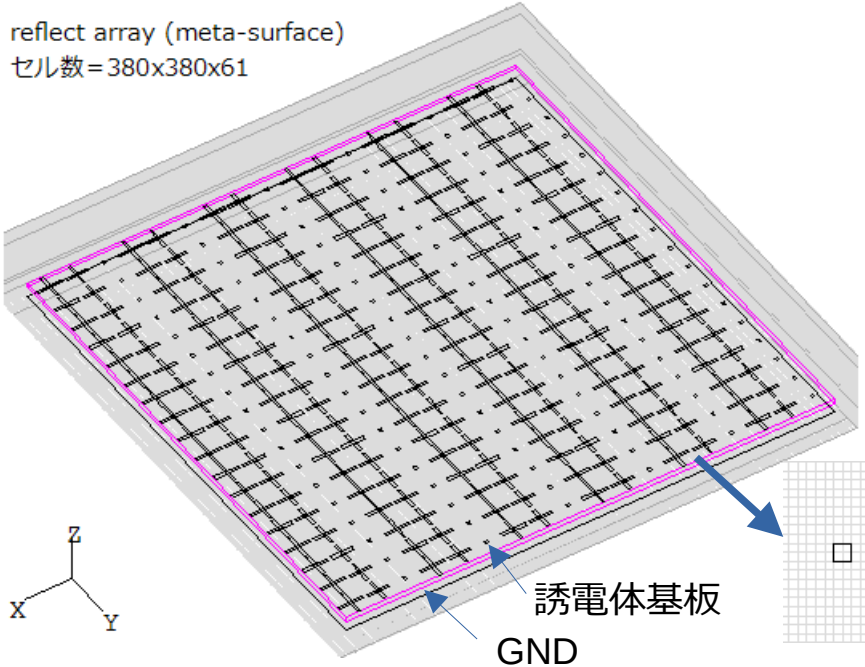
3.80GHz (f_{H2})

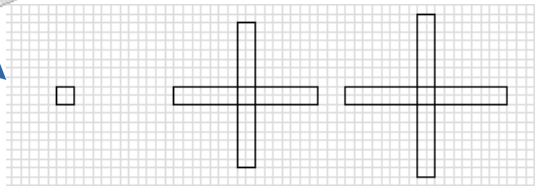
メタサーフェス

●レフレクトアレー[21]

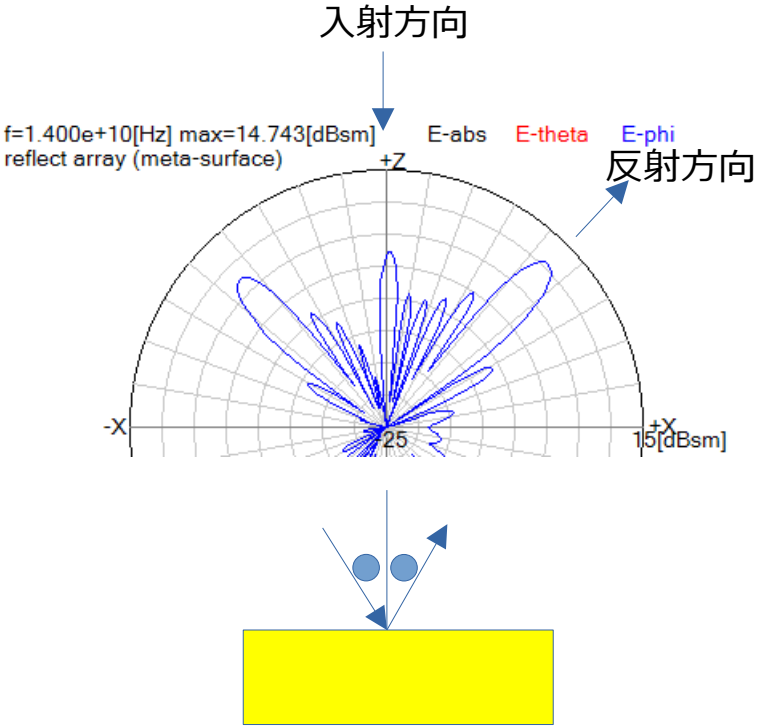
・5Gで使用されるミリ波は波長が短いので回折(回り込み)が弱い

・市街地では電波伝搬を改善するためにビルの壁に反射方向を制御するシートを貼る



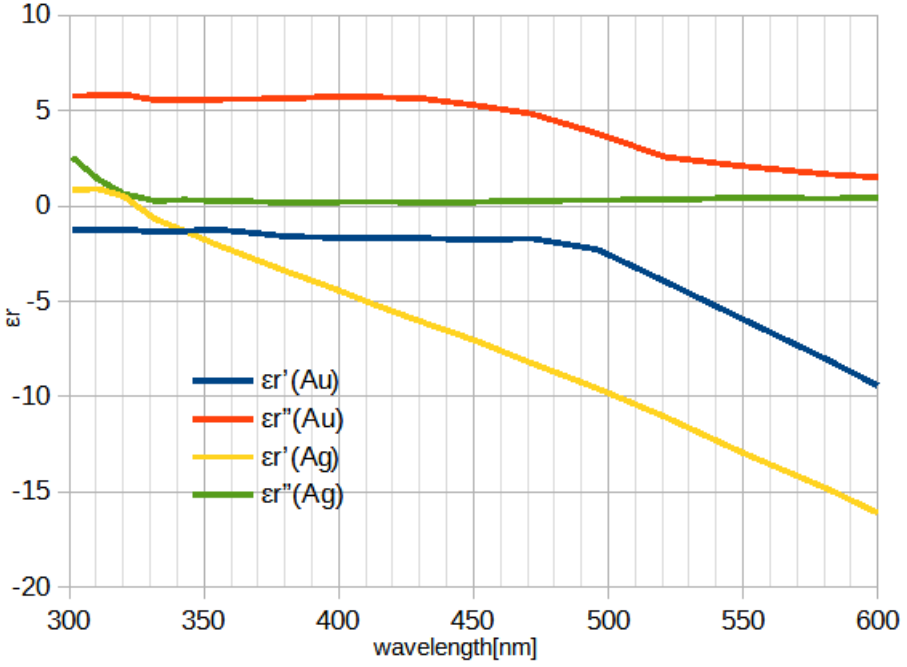


スーパーセル

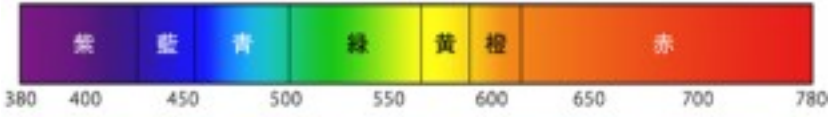


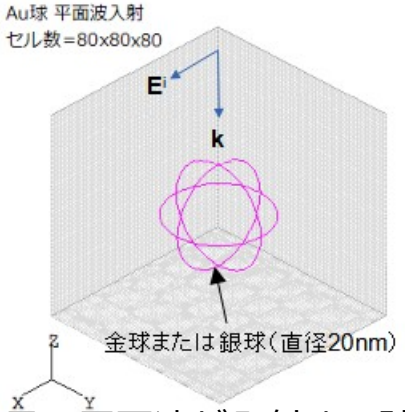
Snellの法則

金・銀1粒子



金と銀の複素比誘電率の波長特性[22]

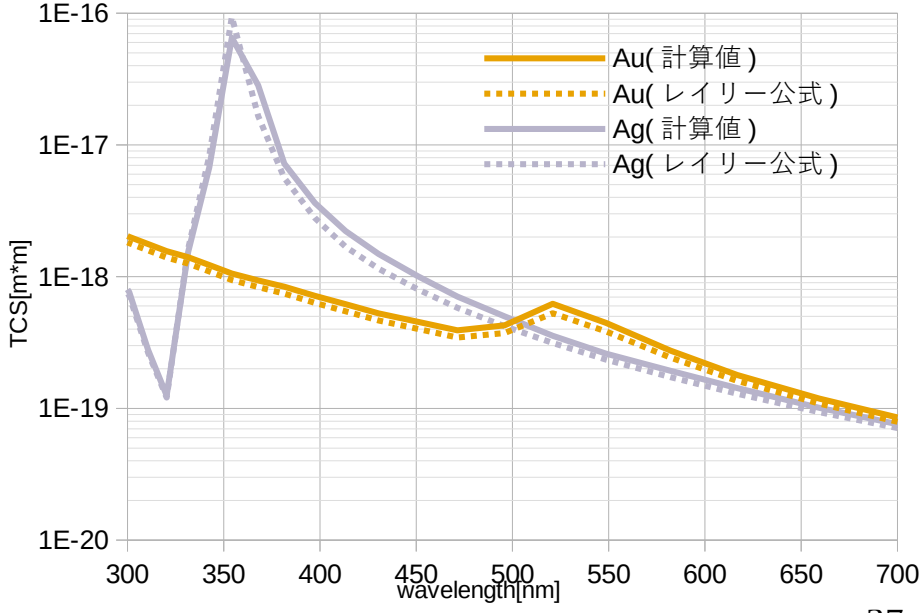




$$\sigma_T = \frac{2\pi^5}{3}\left(\frac{\epsilon_r - 1}{\epsilon_r + 2}\right)^2 \frac{D^6}{\lambda^4}$$

レイリー散乱公式(D<<λ)

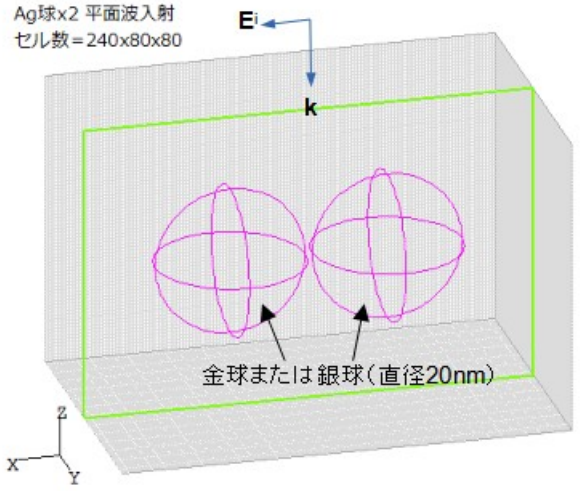
1粒子に平面波が入射する計算モデル

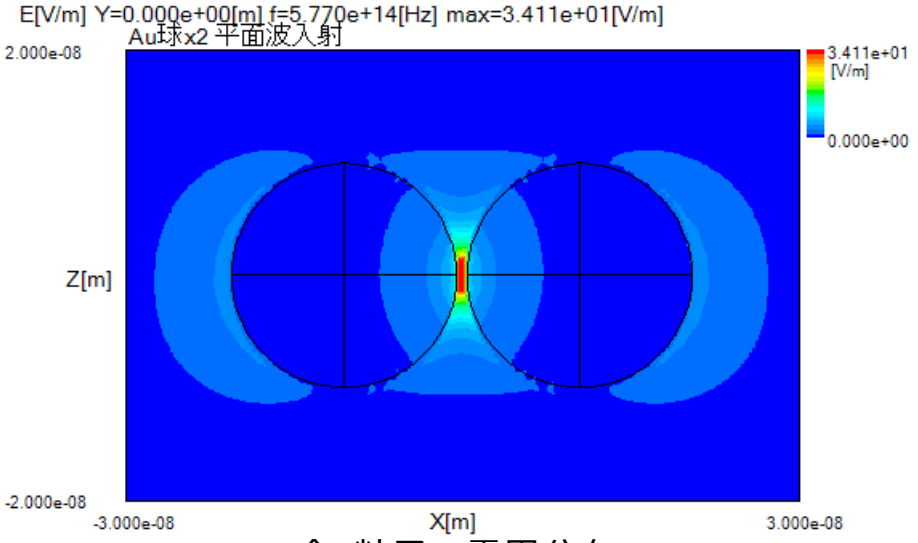


金・銀1粒子の全散乱断面積の波長特性（D=20nm）

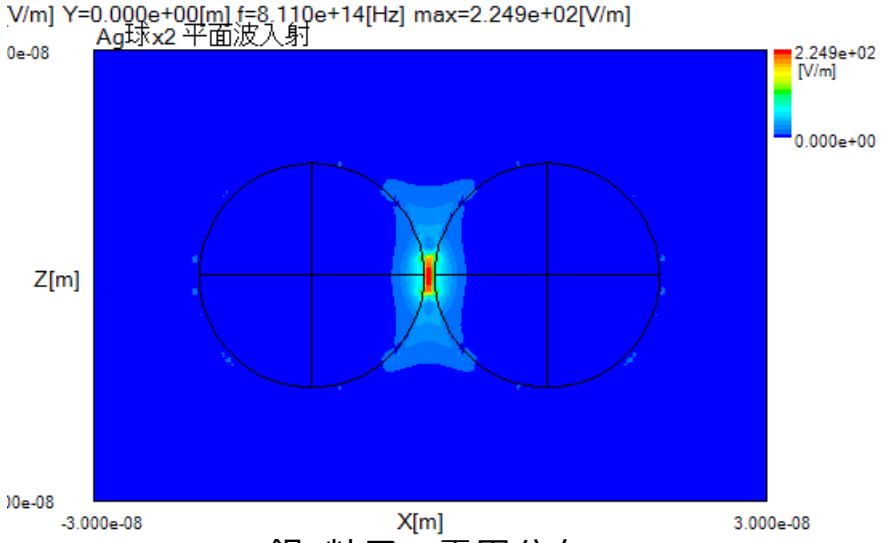
金・銀2粒子

金や銀のナノ粒子を2個近づけて並べ、これに平面波を入射するとギャップ部（～1nm）に非常に大きな光が発生する（最大100万倍）



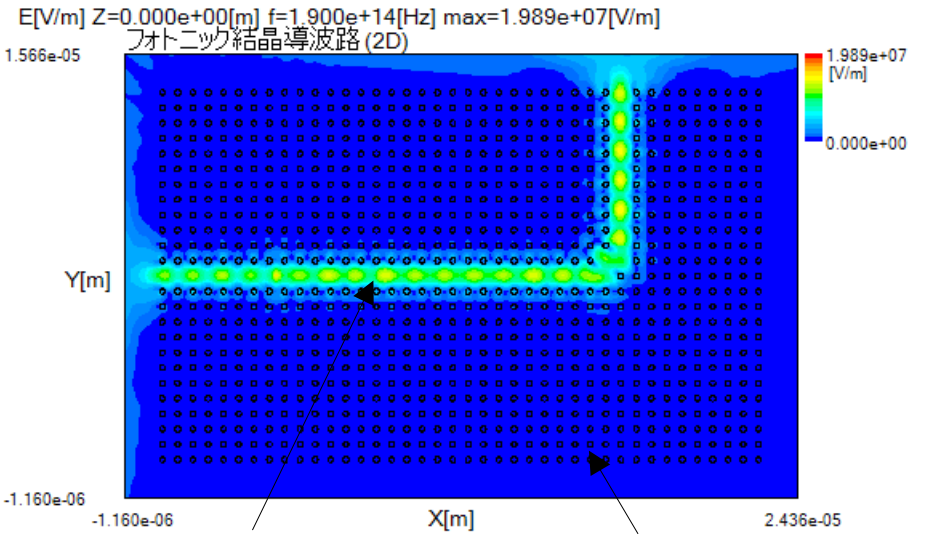


金2粒子の電界分布

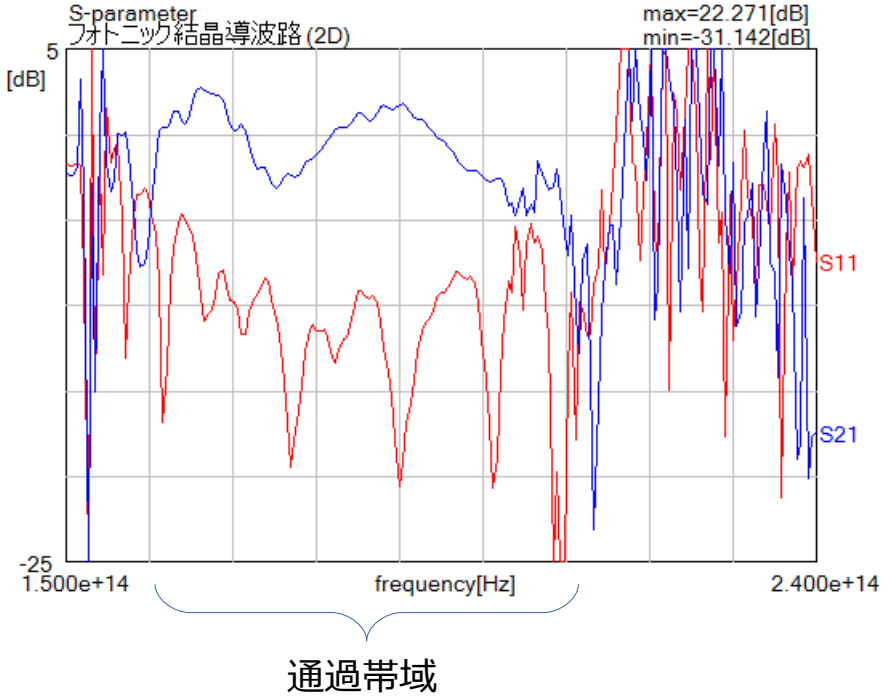


銀2粒子の電界分布

フォトニック結晶光導波路



フォトニック結晶光導波路(2Dモデル)[23]

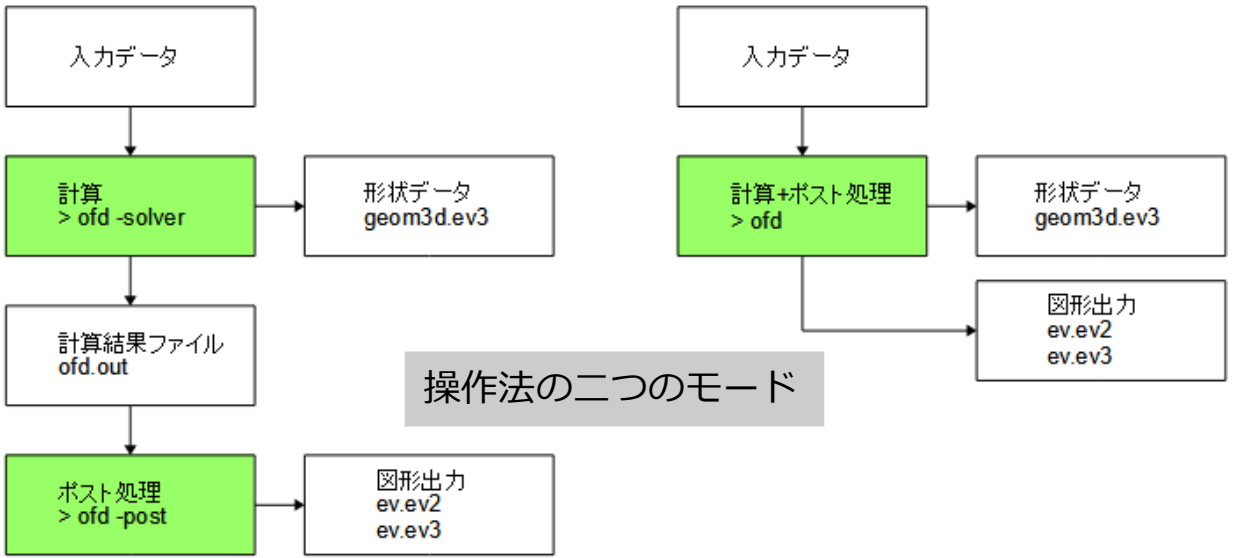


反射係数S11・透過係数S21の周波数特性

光応用で以下のケースのときは関連シミュレーターOpenTHFD（周波数領域差分法）推奨
・半無限の誘電体があるとき　・周囲媒質が空気以外のとき　・周期構造

5. OpenFDTDの使用法

OpenFDTDはWindowsではGUI、LinuxではCUIで操作する [25]-[28]



●計算とポスト処理の分割処理
・計算が終わった後、ポスト処理の設定を変えて繰り返し行える

●計算とポスト処理の一括処理
・計算の前にポスト処理の設定を行う
・半分のメモリーと計算時間で実行できる

※ポスト処理：計算結果を図形出力すること
※ev2/ev3ファイル：2次元/3次元図形データ（独自仕様）

入力データの作り方

(1) テキストファイル編集 (CUI)
・エディタを用いて右のようなテキストファイルを編集する
・複雑なデータを入力することが難しい

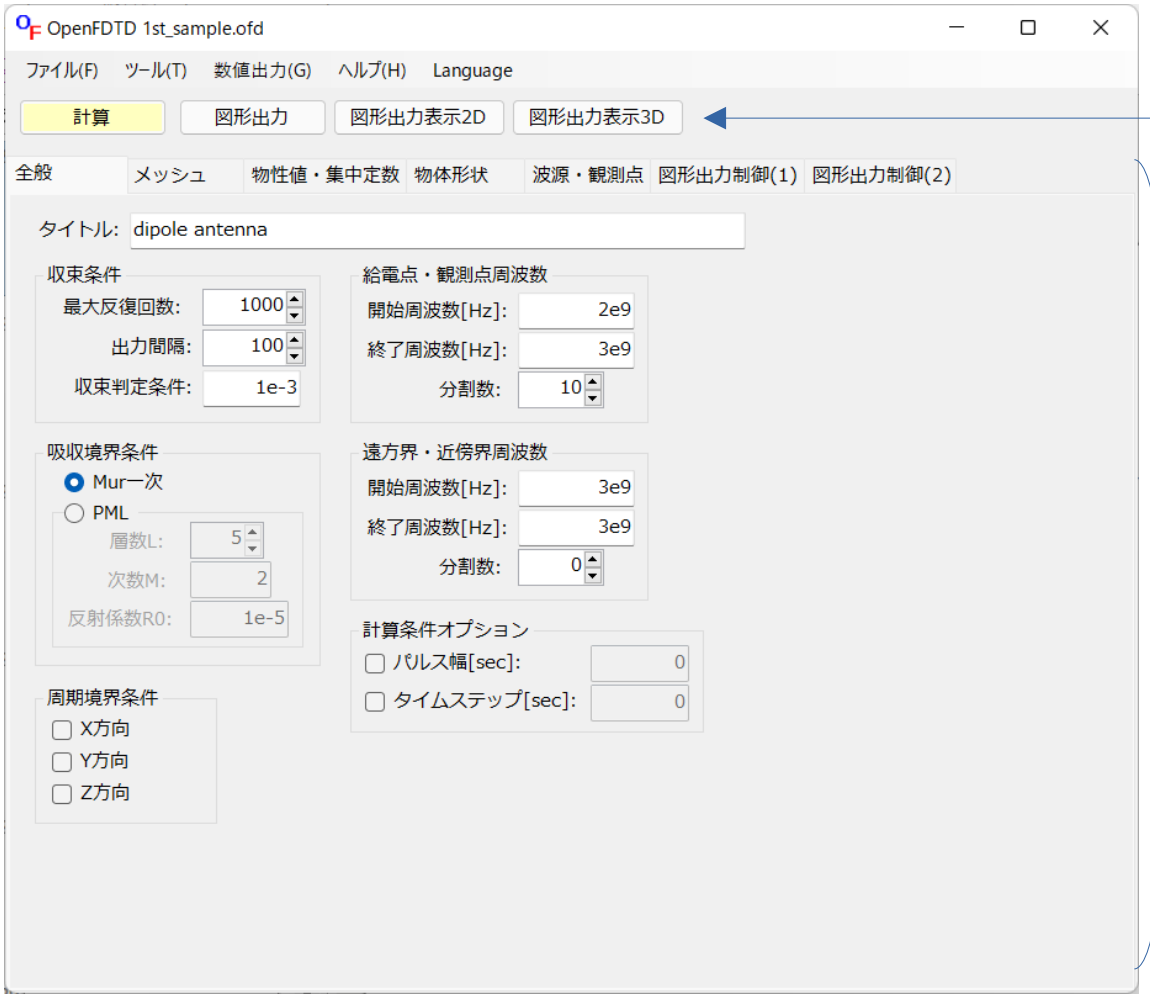
(2) GUI (Graphical User Interface)
・Windows環境ではGUIでデータを作成し計算する
・入力データをLinux環境に転送し計算することができる
・データ入力が容易であるが、データ量が増えると作業量も増える

(3) データ作成ライブラリー使用
・OpenFDTDに同梱されている「データ作成ライブラリー」を使用してデータを作成するプログラムを作る
・プログラミングが必要であるが、大量のデータやパラメーターを変えながら繰り返し計算することに向いている

```
OpenFDTD 2 7
xmesh = -0.075 30 0.075
ymesh = -0.075 30 0.075
zmesh = -0.075 10 -0.025 11 0.025 10 0.075
geometry = 1 1 0 0 0 0 -0.025 0.025
feed = Z 0 0 0 1 0 50
frequency1 = 2e9 3e9 10
frequency2 = 3e9 3e9 0
solver = 1000 100 1e-3
end
```

入力データ例(ダイポールアンテナ)

GUI（Windows環境）



コマンドボタン

データ入力部（7個のタブ）

●計算部

(1) 全般

(2) メッシュ

(3) 物性値

(4) 物体形状

(5) 給電点・観測点

●ポスト処理

(6) 図形出力制御(1)

(7) 図形出力制御(2)

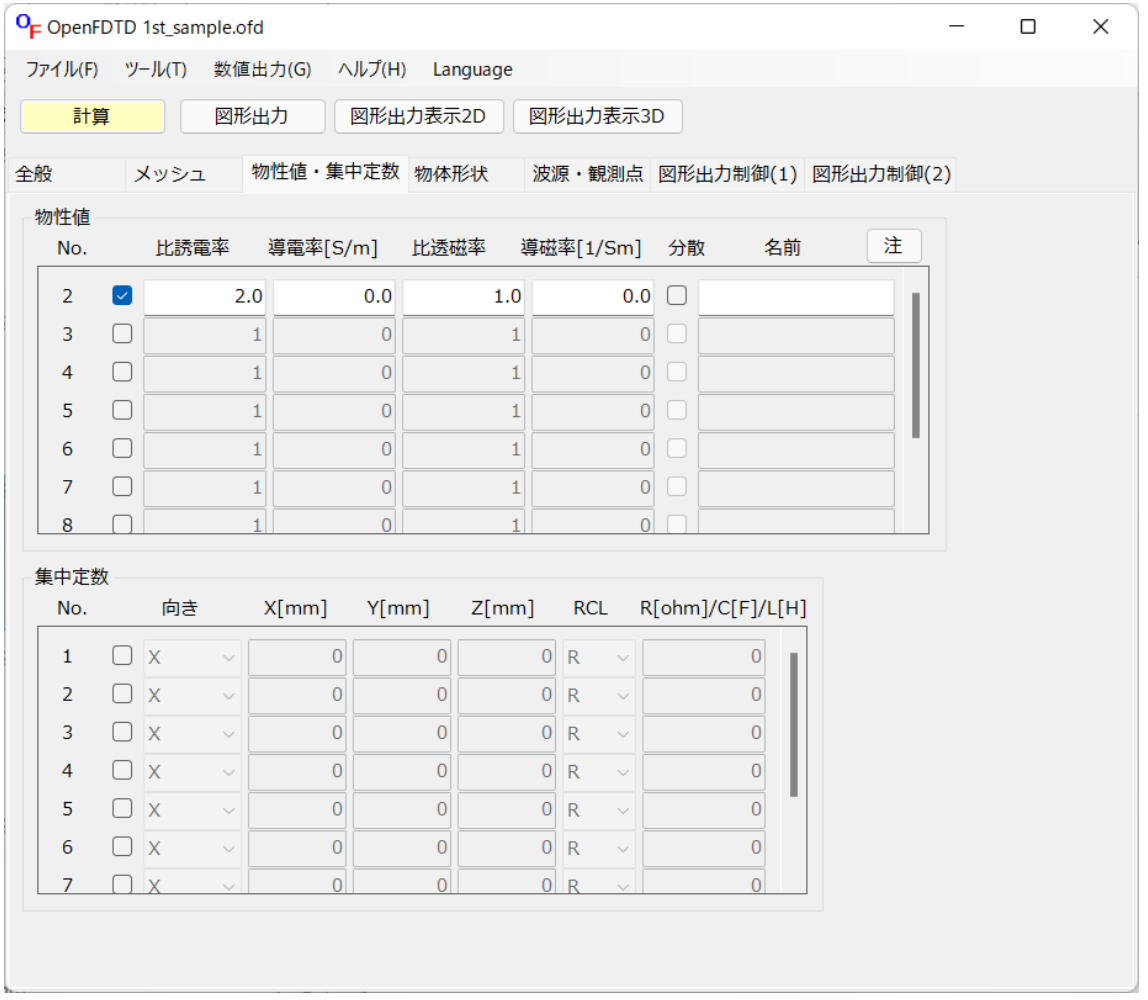
(2) メッシュ

- X/Y/Z方向のメッシュを作成する
- 区間座標と区間分割数を交互に入力する
- 計算領域の大きさとメッシュ分割数は計算精度と計算時間に影響を与える
- 計算誤差はセルサイズに比例する
- セルサイズを半分にすると8倍のメモリーと16倍の計算時間が必要になる

(3) 物性値・集中定数

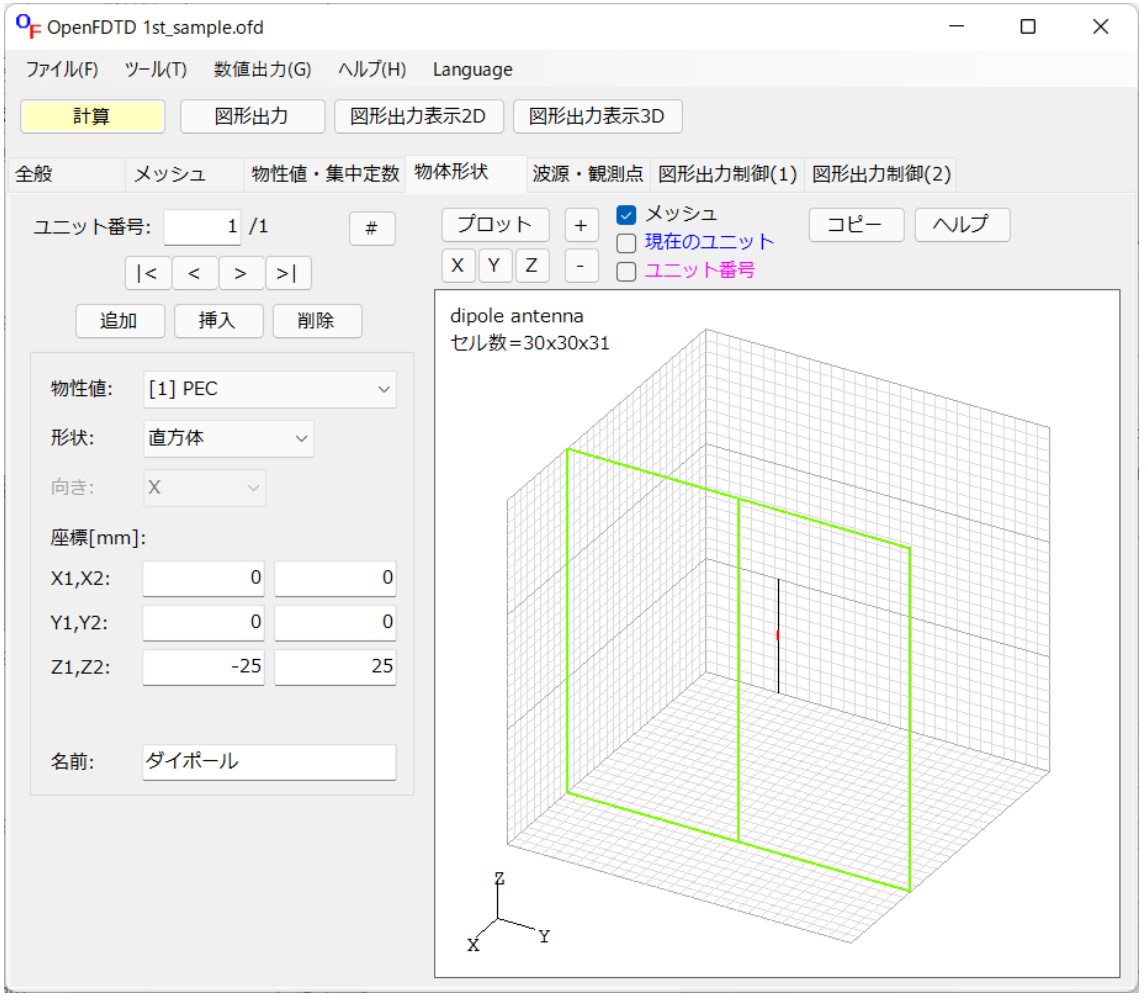
●物性値
・物性値のテーブルを作成する
・比誘電率、導電率、比透磁率、導磁率
・空気とPEC(完全導体)は予め用意されているので入力不要
・物体形状で物性値を選択する

●集中定数
・R/L/Cを入力する



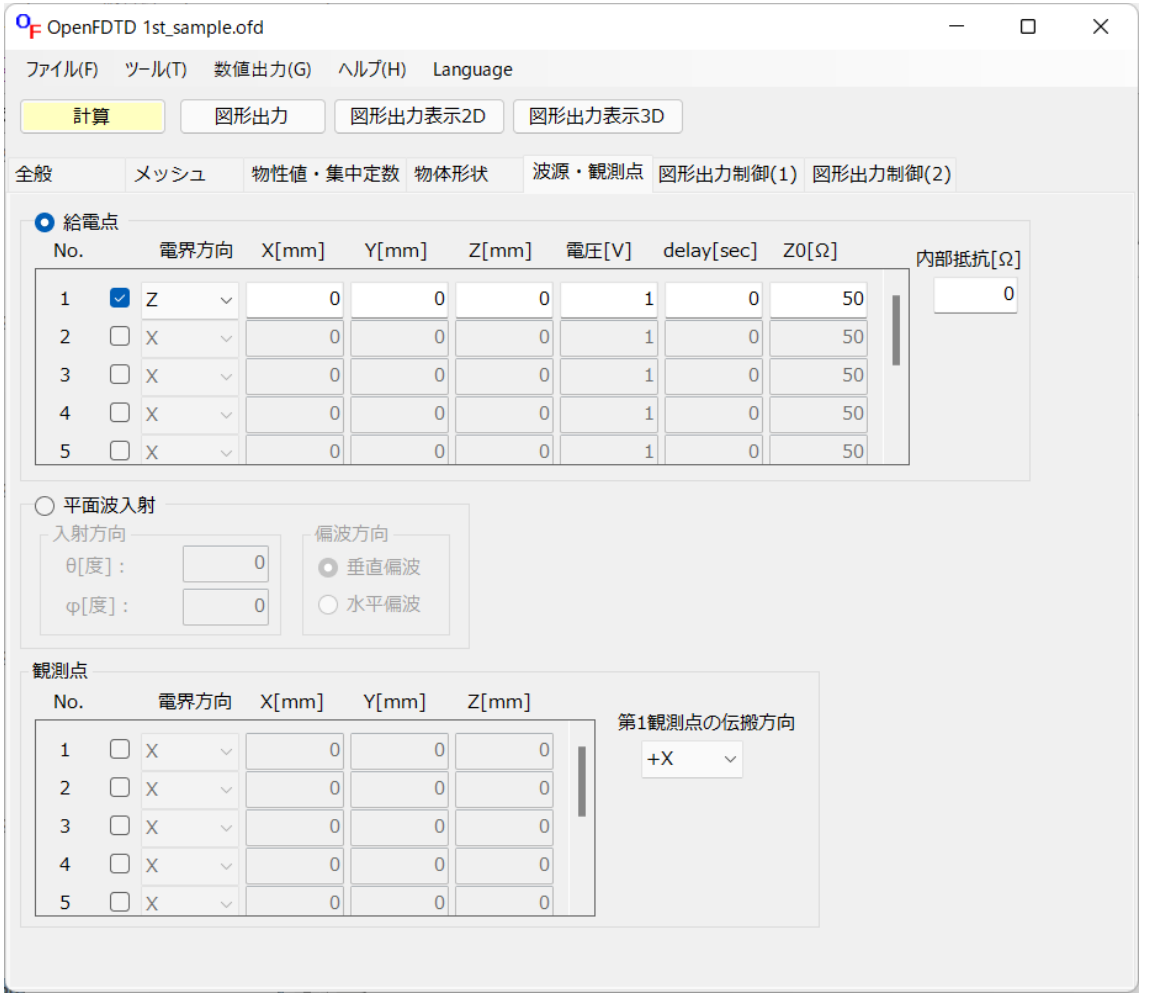
(4) 物体形状

- 物体形状を多数のユニットに分解して入力する（1ユニット=1ページ）
- |<,<,>,>|でユニット番号を変える
- [追加]/[挿入]/[削除]でユニット操作
- [物性値]を選択する
- [形状]と[向き]を選択する
- [座標]を入力する
- [名前]はデータの管理に使用する（オプション）
- 右の図で入力した形状を確認する
- 複数のユニットが重複する場所の物性値はユニット番号が大きい方が優先される



(5) 波源・観測点

・アンテナでは[給電点]を入力する
・散乱問題では[平面波入射]を入力する
・[観測点]はSパラメーターの計算に使用する



計算

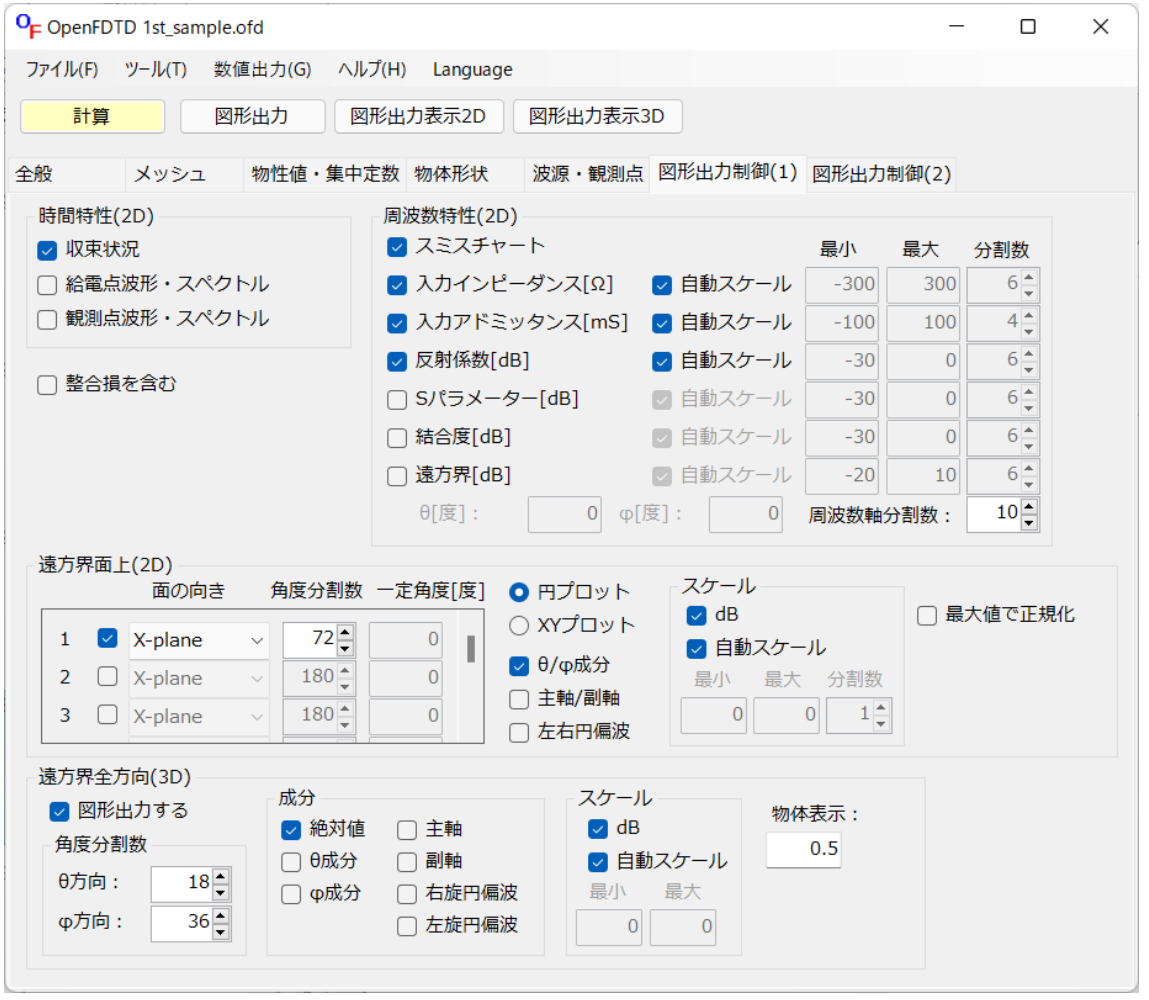
・(1)～(5)のタブを入力した後、[計算]をクリックすると計算が行われる
・標準出力に右のように出力される
・計算条件、計算結果の要約を確認する

```
<<< OpenFDTD (CPU+OpenMP) Ver.2.7.2 >>>
Thread = 8, Process = 1
Sun Dec 19 10:12:10 2021
Title = dipole antenna
Source = feed
Cells = 30 x 30 x 31 = 27900
No. of Materials  = 3
No. of Geometries = 1
No. of Feeds      = 1
No. of Points     = 0
No. of Freq.s (1) = 3
No. of Freq.s (2) = 1
CPU Memory size   = 3 [MB]
Output filesize   = 2 [MB]
ABC = Mur-1st
Dt[sec] = 9.3089e-12, Tw[sec] = 5.0800e-10, Tw/Dt = 54.572
Iterations = 1000, Convergence = 1.000e-03
=== iteration start ===
   step   <E>      <H>
      0 0.000000 0.000000
    100 4.957039 3.894715
    200 0.958861 0.874313
    300 0.188227 0.169974
    400 0.040359 0.034269
    500 0.006610 0.006112
    600 0.001507 0.001271
    --- converged ---
=== input impedance ===
feed #1 (Z0[ohm] = 50.00)
  frequency[Hz] Rin[ohm]   Xin[ohm]    Gin[mS]    Bin[mS]    Ref[dB]     VSWR
  2.00000e+09     34.646   -104.563      2.855      8.618     -2.096    8.328
  2.50000e+09     70.949      0.435     14.094     -0.086    -15.227    1.419
  3.00000e+09    134.375     84.357      5.338     -3.351     -4.606    3.860
=== output files ===
ofd.log, geom3d.htm, ofd.out
=== cpu time [sec] ===
  part-1 :       0.093
  part-2 :       0.009
  --------------------
  total  :       0.102
=== normal end ===
Sun Dec 19 10:12:10 2021
```

(6) 図形出力制御(1)

・計算が終了した後、図形出力制御を行い、[図形出力]→[図形出力表示]を行う
・図形出力には2Dと3Dがある

●図形出力制御(1)
・時間特性
・周波数特性
・遠方界面上
・遠方界全方向



(7) 図形出力制御(2)

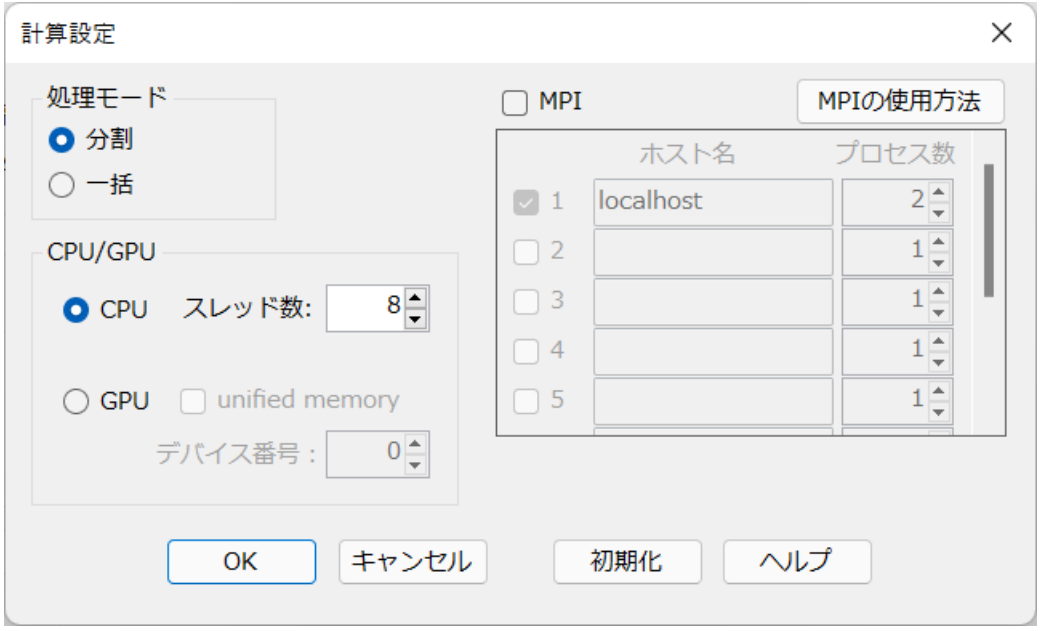
●図形出力制御(2)
・近傍界線上
・近傍界面上

計算設定

●処理モード
・[分割]/[一括]を選択する（通常は[分割]）

●CPU/GPU
・CPU：スレッド数を指定する
・GPU：既定値でよい

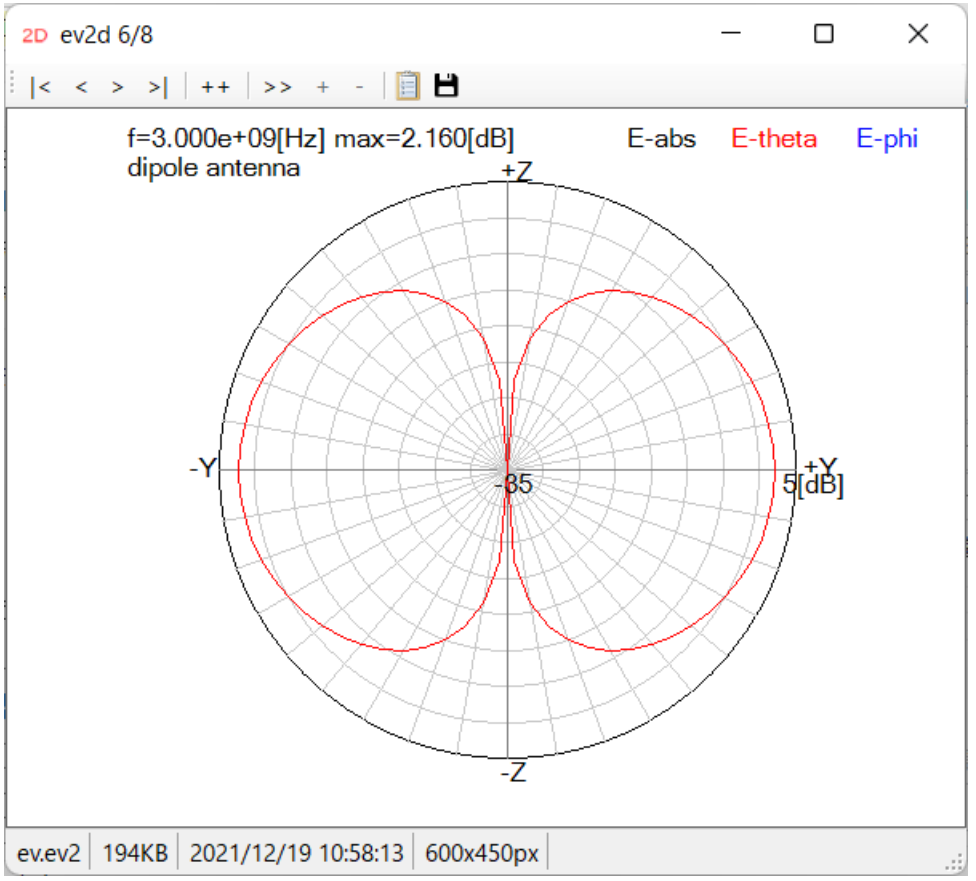
●MPI
・CPU：PCクラスタで使用
・GPU：マルチGPUとPCクラスタで使用



必要メモリー = 78 Nx Ny Nz / Np バイト　（プロセス当たり、Nx, Ny, Nz：セル数、Np：MPIプロセス数）
計算時間 ∝ M Nx Ny Nz / (Np Nt)　（M：タイムステップ数、Nt：スレッド数、比例定数は計算機に固有の値）

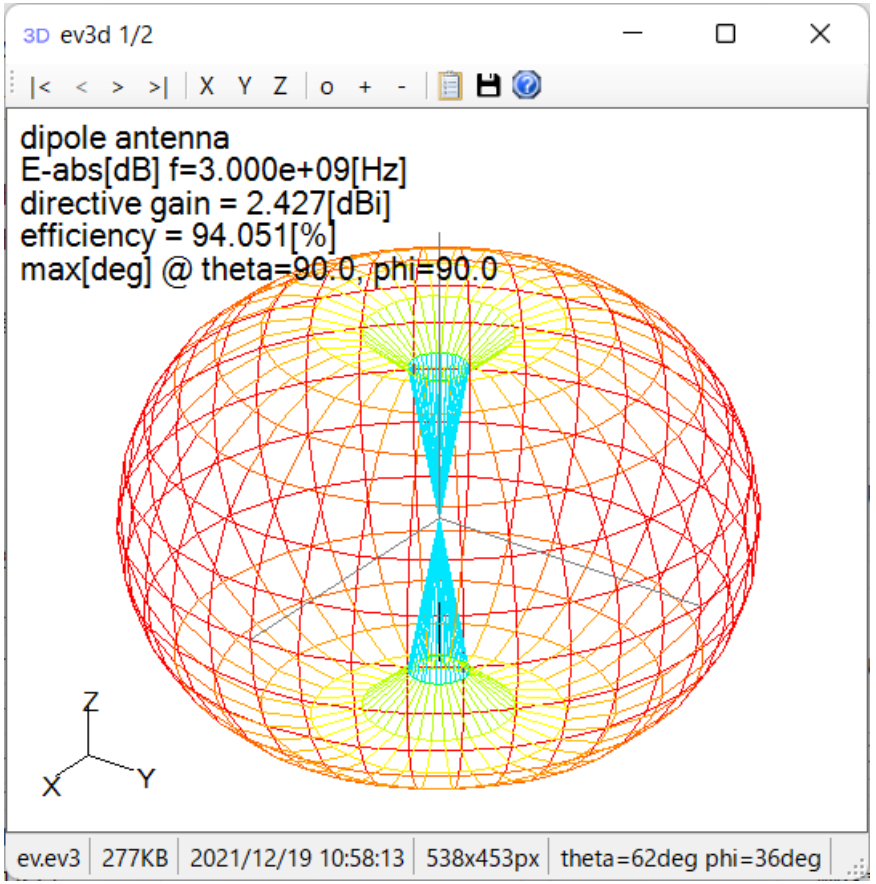
2D図形出力

・2D図形出力ビューワーev2d.exe
が2D図形データev.ev2を読み込んで図形表示する
・上のボタンの意味はマウスを移動して確認する
・|<.<,>,>|：ページ操作
・++：全ページ合成
・>>,+,-：動画
・クリップボードにコピー
・図形ファイルに保存



3D図形出力

・3D図形出力ビューワーev3d.exe
が3D図形データev.ev3を読み込んで図形表示する
・上のボタンの意味はマウスを移動して確認する
・|<.<,>,>|：ページ操作
・X,Y,Z：視点移動
・o,+,-：拡大・縮小
・クリップボードにコピー
・図形ファイルに保存
・マウス操作：移動、回転、拡大、縮小



データ作成ライブラリー

・右のようなプログラムを作成して実行すると
OpenFDTDの入力データが出力される
・以下のようなときに使用する
(1)データの数が多いとき
(2)形状を変数でパラメーター化したいとき
（パラメーターを変更すると形状を一括で変形することができる）
(3)入力データを変えながら繰り返し計算するバッチ処理を行う（system関数を使用する）
※バッチ処理は多数の入力データを作成した後、スクリプトファイルで実行することもできる

```
#include "ofd_datalib.h"

int main(void)
{
	// initialize
	ofd_init();

	// mesh
	ofd_xsection(2, -75e-3, +75e-3);
	ofd_xdivision(1, 30);
	ofd_ysection(2, -75e-3, +75e-3);
	ofd_ydivision(1, 30);
	ofd_zsection(4, -75e-3, -25e-3, +25e-3, +75e-3);
	ofd_zdivision(3, 10, 11, 10);

	// material
	ofd_material(2.0, 0.0, 1.0, 0.0, "");

	// geometry
	ofd_geometry(1, 1, 0e-3, 0e-3, 0e-3, 0e-3, -25e-3, +25e-3);

	// feed
	ofd_feed('Z', 0e-3, 0e-3, 0e-3, 1, 0, 50);

	// frequency
	ofd_frequency1(2e9, 3e9, 10);
	ofd_frequency2(3e9, 3e9, 0);

	// solver
	ofd_solver(1000, 100, 1e-3);

	// output
	ofd_outdata("sample1.ofd");

	return 0;
}
```

6. FOCUSスパコンの使用法

ファイル構成：

OpenFDTD/

- ofd : 実行プログラム (OpenMP対応)
- ofd_mpi : 実行プログラム (MPI+OpenMP対応)
- ofd_cuda : 実行プログラム (CUDA対応)
- ofd_cuda_mpi : 実行プログラム (CUDA+MPI対応)
- include/ : ヘッダーファイルフォルダ
- src/ : ソースコートフォルダ (OpenMP対応)
- mpi/ : ソースコートフォルダ (MPI＋OpenMP対応)
- cuda/ : ソースコートフォルダ (CUDA対応)
- cuda_mpi/ : ソースコートフォルダ (CUDA+MPI対応)
- data/ : 入力データ用フォルダ
  - /sample/ : サンプルデータ用フォルダ
  - /benchmark/ : ベンチマークデータ用フォルダ
- datalib/ : データ作成ライブラリ用ソースコードフォルダ
- arf/ : 最適設計ツール用ソースコードフォルダ

○実行プログラム（赤字）はビルドするとできるファイル（配布ファイルには含まれていない）

○ソースコード（青字）はビルドするのに必要（作成済実行プログラムのみを使用するときは不要）

OpenFDTDのビルド方法

```
$ cd src
$ make -f Makefile_gcc clean   ※1
$ make -f Makefile_gcc         OpenMP版(ofd)が作成される
$ cd ../mpi
$ make -f Makefile_gcc clean
$ make -f Makefile_gcc         MPI+OpenMP版(ofd_mpi)が作成される
$ cd ../cuda
$ make -f Makefile_linux clean
$ make -f Makefile_linux       CUDA版(ofd_cuda)が作成される
$ cd ../cuda_mpi
$ make -f Makefile_linux clean
$ make -f Makefile_linux       CUDA+MPI版(ofd_cuda_mpi)が作成される
$ cd ..
```

※1 予め”mv Makefile_gcc Makefile”を行えば引数”-f Makefile_gcc”は不要

※2 A64FXで富士通コンパイラーを使うときはMakefile_gccはMakefile_fccに置き換える

※3 NEC SX Aurora TSUBASAでNECコンパイラーを使うときはMakefile_gccはMakefile_nccに置き換える

FOCUSとPCで共同作業する方法3通り

(1) FOCUSで計算のみを行い、PCでポスト処理を行う
・PCのGUIで入力データを作成する
・入力データをFOCUSに転送する（入力データはKBのオーダー）
・FOCUSで引数″-solver”をつけて計算する（ofd.outが出力される）
・できたofd.outをPCに転送する
・PCでポスト処理を（繰り返し）行う
※一番操作性がよいがファイルサイズがGBを超える可能性があるofd.outを転送する必要がある

(2) FOCUSで計算を行い、その後ポスト処理も行い、最後にPCで図形表示する
・PCのGUIで入力データを作成する
・入力データをFOCUSに転送する（入力データはKBのオーダー）
・FOCUSで引数″-solver”をつけて計算する（ofd.outが出力される）
・PCのGUIでポスト処理の設定を行い入力データをFOCUSに転送する
・FOCUSで引数″-post -ev”をつけてポスト処理を行う（引数″-ev”を省略するとev2d.htm,ev3d.htmが出力される）
・できた図形データev.ev2,ev.ev3をPCに転送する（ファイルサイズはMBのオーダー）
・PCでビューワーev2d.exe,ev3d.exeを用いて図形表示する、必要なら3行上に戻る
※ofd.outを転送する必要がないが手順が増える（問題のサイズが大きいとき使用する）

(3) FOCUSで計算とポスト処理を連続して行い、最後にPCで図形表示する

・PCのGUIで入力データを作成する
・入力データをFOCUSに転送する（入力データはKBのオーダー）
・FOCUSで引数″-solver”も″-post”もつけずに計算する（ofd.outは出力されない）
・できた図形データev.ev2,ev.ev3をPCに転送する（ファイルサイズはMBのオーダー）
・PCでビューワーev2d.exe,ev3d.exeを用いて図形表示する

※(1)(2)と比べると約半分のメモリーと計算時間ですむが、ポスト処理の設定を変えると再度計算から始める必要がある（ポスト処理の設定が確定しているときはこの方法でもよい）

●参考

・ofd.outはポスト処理に必要な計算結果をすべて含むバイナリファイル
・計算時間>>ポスト処理時間であるから計算結果をいったんofd.outに保存しておくとポスト処理を設定を変えながら繰り返し効率よく行うことができる
・ofd.outを名前を変えて保存しておき、後で名前を戻してポスト処理のみを行うことができる、ただしofd.outにはポスト処理設定は含まれていないのでポスト処理設定を含む入力データが同時に必要
・ofd.outはファイルサイズが大きいのでディスク容量の残量に注意

実行プログラムのファイル名

・ofd : Intel CPU 1ノード　（F/H/Zシステム）
・ofd_mpi : Intel CPU 複数ノード　（F/H/Zシステム）※
・ofd_fcc : 富士通スパコン MPI非使用 （Xシステム）
・ofd_fcc_mpi : 富士通スパコン MPI使用 （Xシステム）※
・ofd_ncc : NECスパコン 1ノード　（V/Wシステム）
・ofd_ncc_mpi : NECスパコン 複数ノード　（V/Wシステム）※
・ofd_cuda : NVIDIA GPU 1ノード （Fシステム）
・ofd_cuda_mpi : NVIDIA GPU 複数ノード （Fシステム）※

※MPI対応プログラムは1ノードでも計算可、計算時間（Xシステム以外）と総使用メモリーは非MPIと同じ

初回の作業

ログインサーバーにログイン後

```
$ ssh ff   （フロントエンドサーバーにログイン、パスワード不要）
$ mkdir OpenFDTD    （OpenFDTD用のディレクトリの作成）
$ cd OpenFDTD
$ cp -r /home1/gleg/share/OpenFDTD/data .     （サンプルデータのコピー）
$ cp /home1/gleg/share/OpenFDTD/*.sh .     （計算実行スクリプトファイルのコピー※1）
$ ln -s /home1/gleg/share/OpenFDTD/ofd .   （実行プログラムのシンボリックリンク※2）
$ ln -s /home1/gleg/share/OpenFDTD/ofd_mpi .
$ ln -s /home1/gleg/share/OpenFDTD/ofd_fcc .
$ ln -s /home1/gleg/share/OpenFDTD/ofd_fcc_mpi .
```

※1 自分で作るなら不要です

※2 実ファイルをコピーしても構いません（”ln -s”をcpに変える）

計算実行スクリプト例

※mpiexec_focusは一時的なパッチ、正しくはmpiexec

●Xシステム1ノードで計算するとき（ofd_fcc.sh, MPI使用）　　赤字は要編集

```
#!/bin/bash
#SBATCH -p x001h_lec     （パーティション）
#SBATCH -t 3     （計算時間上限、単位：分）
#SBATCH -J ofd_fcc
#SBATCH -o stdout.%J
FJ_PATH=/opt/FJSVstclanga/v1.1.0
export PATH=${FJ_PATH}/bin:${PATH}
export LD_LIBRARY_PATH=${FJ_PATH}/lib64:${LD_LIBRARY_PATH}
mpiexec_focus -n 4 ./ofd_fcc_mpi -n 12 -ev data/benchmark/benchmark100.ofd ※
```

●Xシステム複数ノードで計算するとき（ofd_fcc_mpi.sh, MPI使用）

```
#!/bin/bash
#SBATCH -p x001h_lec     （パーティション）
#SBATCH -N 2       （ノード数）
#SBATCH --ntasks-per-node=4     （ノード当たり4プロセスが一番速い）
#SBATCH -t 3     （計算時間上限、単位：分）
#SBATCH -J fcc_mpi
#SBATCH -o stdout.%J
FJ_PATH=/opt/FJSVstclanga/v1.1.0
export PATH=${FJ_PATH}/bin:${PATH}
export LD_LIBRARY_PATH=${FJ_PATH}/lib64:${LD_LIBRARY_PATH}
srun hostname > hostfile
mpiexec_focus -hostfile hostfile ./ofd_fcc_mpi -n 12 -ev data/benchmark/benchmark100.ofd ※
```

ジョブの実行

フロントエンドサーバーにログイン後

$ cd OpenFDTD

スクリプトファイル（Xシステムのときはofd_fcc_mpi.sh）をエディタで編集する

$ sinfo -s　（稼働状況を確認する、NODESの”I”が空きノード数、下記は講習会用）

```
PARTITION       AVAIL  TIMELIMIT   NODES(A/I/O/T)  NODELIST
a001h_lec          up    1:00:00       21/31/0/52  a[067-078,080,082-084,086-089,091-
  095,098-101,103-117,120-127]
f001h_lec          up    1:00:00          4/0/0/4  f[253-256]
f001h_p100_lec     up    1:00:00          0/0/0/0
g001h_lec          up    1:00:00          0/4/0/4  g[001-004]
h001h_lec          up    1:00:00          2/6/0/8  h[001-005,030,034-035]
x001h_lec          up    1:00:00          0/4/0/4  x[003-006]
g006m*             up       6:00          0/4/0/4  g[001-004]
```

$ sbatch ofd_fcc_mpi.sh　（ジョブ実行）

$ squeue　（ジョブの確認、STが現在の状態、R/PD/CG:実行中/開始待ち/終了処理中）

```
     JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
   1627223     g006m      ofd uetk0001  R       0:03      1 g001
```

$ scancel ジョブ番号　（ジョブを取り消す、ジョブ番号はsqueueのJOBIDで確認）

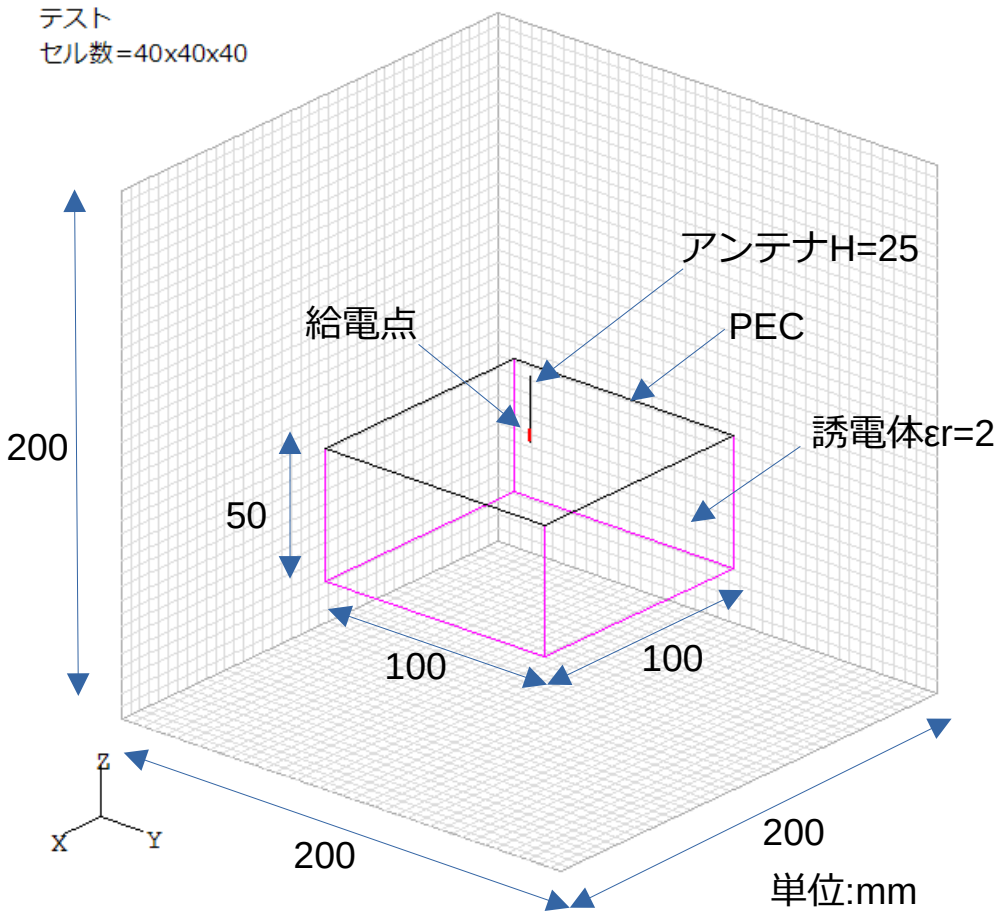
$ thismonth　（今月の使用料を確認する）

出力ファイルの説明

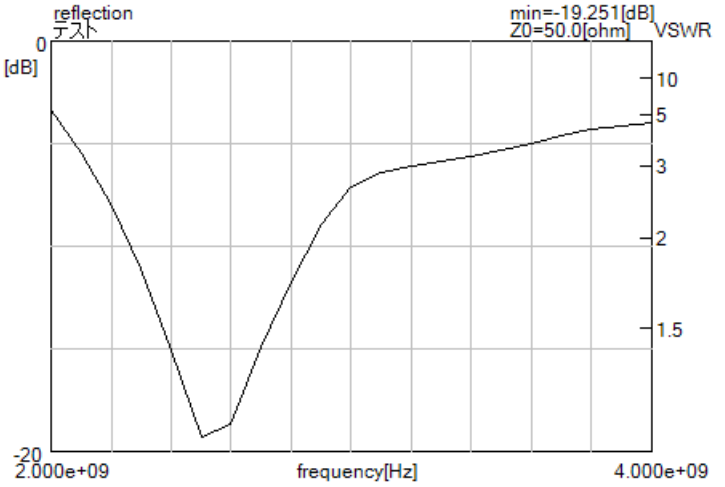
・ofd.out : 計算結果のバイナリファイル、計算とポスト処理を分割したときのみ出力される
・ofd.log : 計算の標準出力、計算結果の確認に使用する、stdout.xxxxxxxと同じもの
・*.log : その他のログファイル、通常不要、詳しくはOpenFDTDのページの5.6参考
・ev.ev2 : 2次元図形データ、PCに転送してev2d.exeで図形表示する　※
・ev.ev3 : 3次元図形データ、PCに転送してev3d.exeで図形表示する　※
・geom.ev3 : 物体形状の3次元データであるがPCで確認済みなので不要

※引数”-ev”をつけないときは代わりにev2d.htmとev3d.htmが出力される、HTMLファイルなのでPCに転送してブラウザで開く

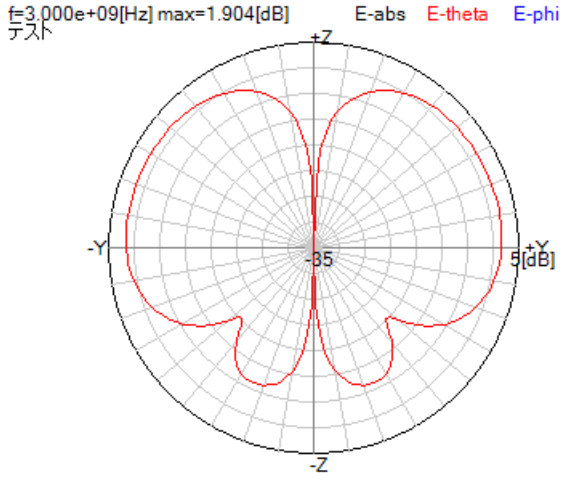
練習問題



単位:mm



反射係数(2～4GHz)



遠方界パターン(3GHz,X面)

文献

[1] OpenFDTD, http://www.e-em.co.jp/OpenFDTD/

[2] K.S.Yee, “Numerical solution of initial boundary value problem involving Maxwell’s equations in isotropic media”, IEEE Trans. Antennas Propagatation, AP-14, no.3, pp.302-307, 1966.

[3] A.Taflove and M.E.Brodwin, “Numerical solution of steady-state electromagnetic scattering problems using the time-dependent Maxwell’s equations,” IEEE Trans. Microwave Theory and Techniques, vol.23, pp.623-630, 1975.

[4] G.Mur, “Absorbing boundary conditions for the finite-difference approximation of the time domain electromagnetic-field equations”, IEEE Trans. Electromag. Compat., EMC-23, no.11, pp.377-382, 1981.

[5] K.Kunz and R.J.Luebbers, *Finite Difference Time Domain Method for Electromagnetics*, CRC Press, 1993.

[6] J.-P.Berenger, "A perfectly matched layer for the absorption of electromagnetic waves", J. Computational Physics, vol.114, pp.185-200, 1994.

[7] A.Taflove, *Computational Electrodynamics, The Finite-Difference Time-Domain Method*, Artech House, 1995.

[8] 宇野亨, “FDTD法による電磁界およびアンテナ解析,” コロナ社, 1998.

[9] K.K.Mei, "On the Integral Equations of Thin-Wire Antennas", IEEE Trans. Antennas & Propagation, AP-13, no.3, pp.374-378, 1965.

[10] R.F.Harrington, *Field computation by moment methods*, New York Macmillan 1968.

[11] G.J.Burke and A.J.Poggio, *Numerical Electromagnetic Code (NEC) - Method of Moments, Part - I : Theory*, Lawrence Livermore National Laboratory, 1981.

[12] 富士通, https://www.fujitsu.com/jp/products/computing/servers/supercomputer/

[13] 菅原清文, “C/C++プログラマーのための OpenMP並列プログラミング 第2版,” カットシステム, 2012.

[14] P.パチェコ(秋葉博訳), “MPI並列プログラミング,” 培風館, 2001.

[15] 渡辺健介他, “スーパーコンピュータ「富岳」向けのアプリケーション開発環境,” 富士通テクニカルレビュー, 2020年10月.
[16] H.Nakano, *Low-Profile Natural and Metamaterial Antennas*, IEEE Press, Wiley, 2016.
[17] V.G.Veselago, “The Electrodynamics of Media with Simultaneously Negative Values of ε and μ,” Soviet Physics Uspekhi, vol.10, no.4, pp.509-514, 1968.
[18] J.B.Pendry, A.J.Holden, W.J.Stewart,and I.Youngs, “Extremely low frequency plasmons in metallic mesostructures,” Phys. Rev. Lett., vol.76, no.25, pp.4773-4776, 1996.
[19] D.R.Smith, W.J.Padilla, D.C.Vier, S.C.Nemat-Nasser, and S. Schultz, “Composite medium with simultaneously negative permeability and permittivity,” Phys. Rev. Lett., vol.84, no.18, pp.4184-4187, May 2000.
[20] C.Caloz and T.Itoh, *Electromagnetic Metamaterials: Transmission Line Theory and Microwave Applications*, IEEE Press, Wiley, 2006.
[21] 宇野亨, 道下尚文, “メタマテリアルアンテナの基礎,” コロナ社, 2021.
[22] P. B. Johnson and R. W. Christy, "Optical Constants of the Noble Metals," Physical Review B, vol.6, no.12, 1972.
[23] 電気学会編, ”計算電磁気学,” 培風館, 2003.
[24] 高原淳一, “メタサーフェス-新しい平面光学素子の原理と産業化への展望-,” 電子情報通信学会誌, vol.105, no.1, pp.39-46, 2022年1月.
[25] 大賀明夫, “FDTD法の並列化技術とオープンソース化,” 電子情報通信学会技術研究報告, AP2014-41, pp.7-12, 2014.
[26] 大賀明夫, “OpenFDTDとOpenMOMで始めるはじめてのアンテナ・シミュレーション, “ RFワールド, No.39, pp.7-82, CQ出版, 2017年8月.
[27] 大賀明夫, “HF/VHF帯線状アンテナの特性シミュレーション,” RFワールド, No.42, pp.61-81, CQ出版, 2018年5月.
[28] 大賀明夫, “機能無制限&GPU対応! 3D電磁界シミュレータOpenFDTD,” トランジスタ技術, 2020年1月号, pp.73-87, CQ出版, 2020.