

# TDGL方程式に基づく 数値シミュレーション

オブジェクト指向プログラミングによる込み入った数式の実装  
(前回とほぼ同じ内容でごめんなさい...)

有明高専 電子情報工学科 松野哲也

「量子化磁束動力学シミュレーション研究会」  
(2016年1月19日(火)14:00-16:00)  
九州工業大学サテライトキャンパス(天神イムズ11F)

# FDTD法によるTDGLシミュレーション

- FDTD: 時間領域有限差分法   
( 偏微分方程式 → 時間発展差分方程式 )
- FEM: 有限要素法  
( 偏微分方程式 → 変分問題 → 線形方程式 )

# TDGL方程式の導出

$$\epsilon_{GL} = \epsilon_K + \epsilon_P + \epsilon_B,$$

$$\begin{aligned}\epsilon_K &= \frac{1}{2} |\mathbf{D}\psi|^2, \quad \mathbf{D} \equiv \nabla - ig\mathbf{A}, \\ \epsilon_P &= \frac{\eta}{2} (|\psi|^2 - 1)^2, \\ \epsilon_B &= \frac{1}{2} |\nabla \times \mathbf{A}|^2.\end{aligned}$$

$$\tau \left( \frac{\partial}{\partial t} + ig\phi \right) \psi = -\frac{\delta\epsilon_{GL}}{\delta\bar{\psi}},$$

$$\frac{\partial \mathbf{A}}{\partial t} + \nabla\phi = -\frac{\delta\epsilon_{GL}}{\delta\mathbf{A}},$$

TDGL方程式  
 時間依存ギンツブルグラウ  
 方程式  
 ( Time dependent Ginzburg-  
 Landau equation )



$$\tau \left( \frac{\partial}{\partial t} + ig\phi \right) \psi = \frac{1}{2} \mathbf{D}^2 \psi + \eta(1 - |\psi|^2) \psi,$$

$$\frac{\partial \mathbf{A}}{\partial t} + \nabla\phi = g \text{Im} [\bar{\psi} \mathbf{D} \psi] - \nabla \times \nabla \times \mathbf{A}.$$

# ゲージ対称性

$$\mathbf{A} \longrightarrow \mathbf{A} + \nabla\chi, \quad \phi \longrightarrow \phi - \frac{\partial\chi}{\partial t}, \quad \psi \longrightarrow \psi \exp(ig\chi).$$

$\phi$ -zero gauge

$$\tau \frac{\partial\psi}{\partial t} = \frac{1}{2} \mathbf{D}^2 \psi + \eta(1 - |\psi|^2)\psi,$$

$$\frac{\partial\mathbf{A}}{\partial t} = g \text{Im} [\bar{\psi} \mathbf{D}\psi] - \nabla \times \nabla \times \mathbf{A}.$$

# なぜ $\phi$ ゼロゲージ, なぜリンク変数法か?

$\phi$ -zero gauge



FDTD法と相性が良い.

陽的数値積分を組める.

並列化が容易

高速化

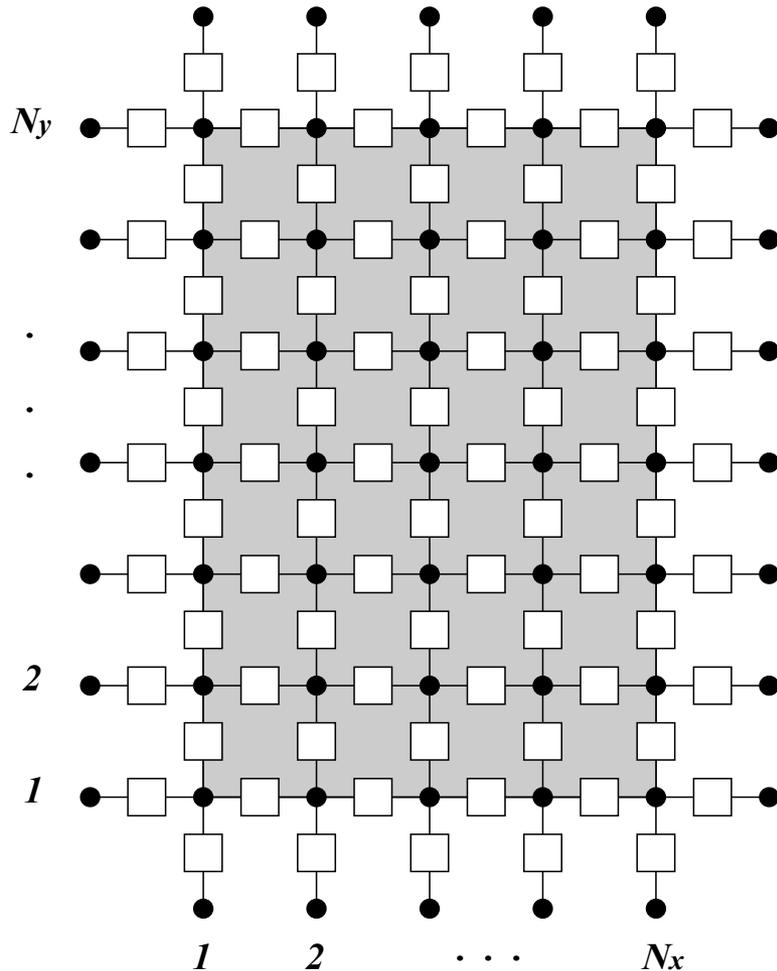


flux flow シミュレーションを行うにはリンク変数法が必須.



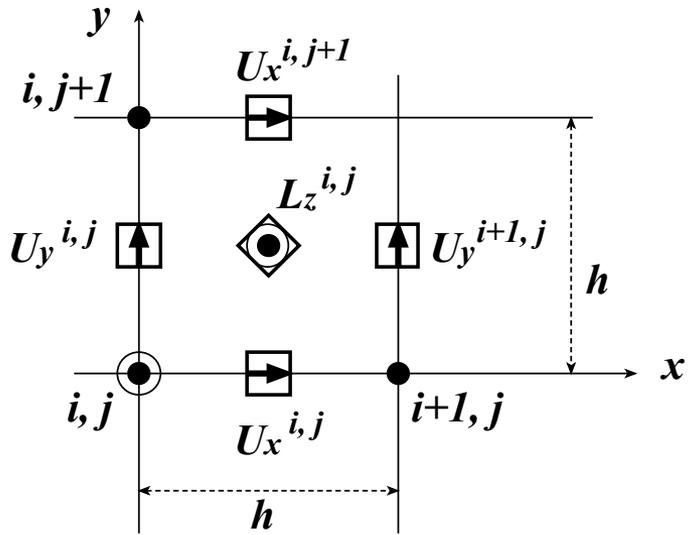
E一定は A の時間微分が一定であることを意味するから.  
つまり,  $t \rightarrow \infty$  で A は有界ではない.

# リンク変数法



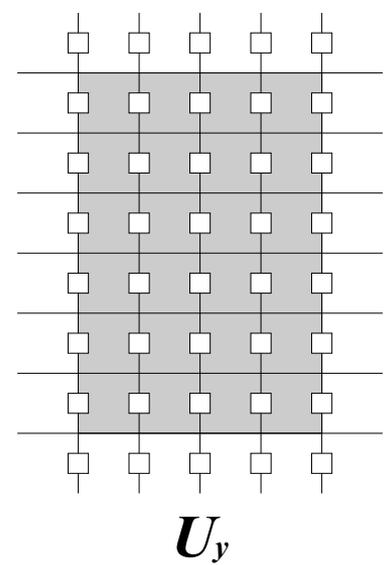
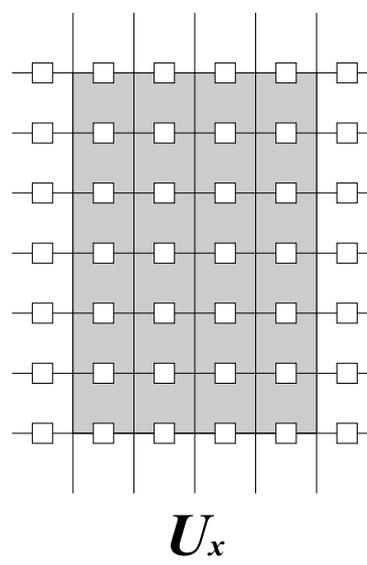
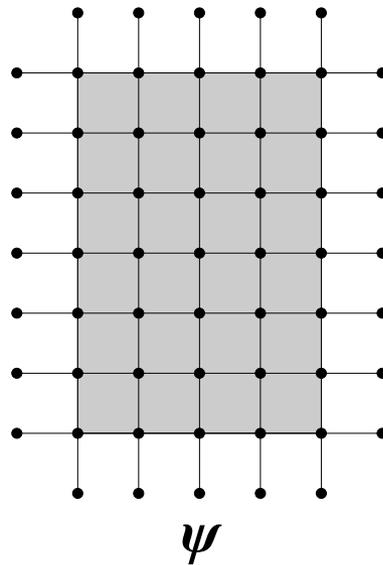
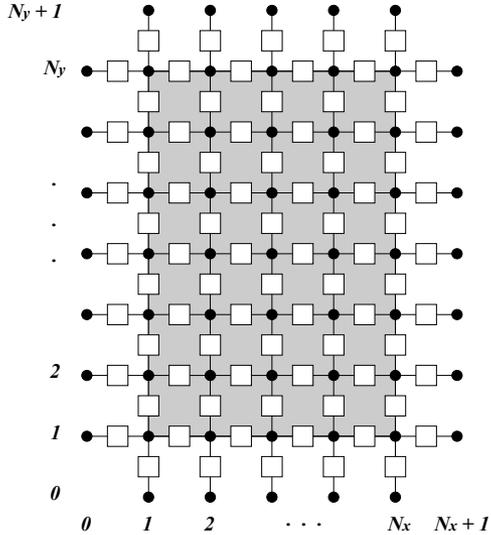
格子点の上で  
オーダーパラメータを定義

格子点と格子点の間で  
ベクトルポテンシャルを定義



リンク変数:

$$\begin{aligned}
 U_x^{i,j,k} &= \exp(-ihgA_x^{i,j,k}), \\
 U_y^{i,j,k} &= \exp(-ihgA_y^{i,j,k}), \\
 U_z^{i,j,k} &= \exp(-ihgA_z^{i,j,k}),
 \end{aligned}$$

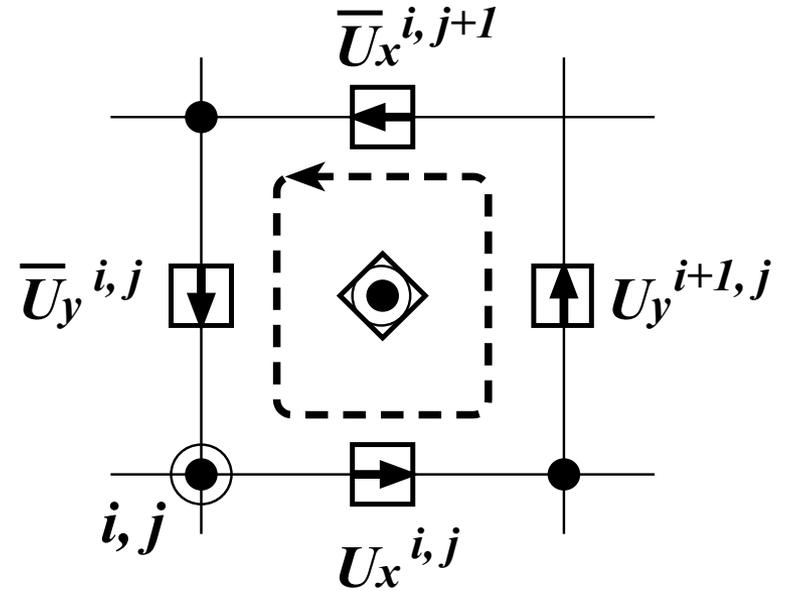


## 共変微分演算子(2次元)

$$D_x \psi \longrightarrow \frac{1}{h} (U_x^{i,j} \psi^{i+1,j} - \psi^{i,j})$$

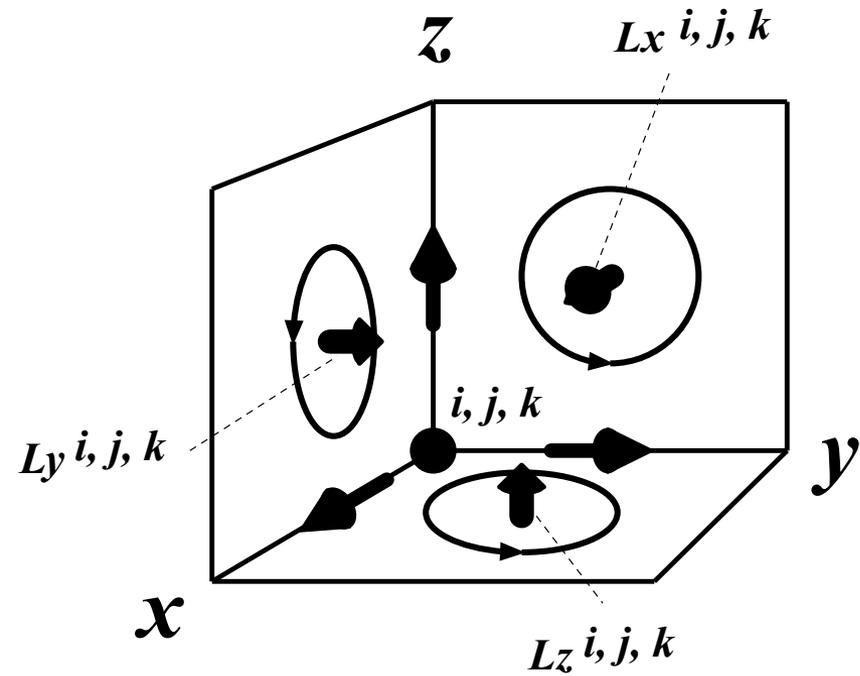
$$D_x^2 \psi \longrightarrow \frac{1}{h^2} (U_x^{i,j} \psi^{i+1,j} + \bar{U}_x^{i-1,j} \psi^{i-1,j} - 2\psi^{i,j})$$

リンク変数の積は経路積分に対応する.



$$\begin{aligned}
 L_z^{i,j} &\equiv U_x^{i,j} U_y^{i+1,j} \bar{U}_x^{i,j+1} \bar{U}_y^{i,j} \\
 &= \exp[-ihg(A_y^{i+1,j} - A_y^{i,j} - A_x^{i,j+1} + A_x^{i,j})] \\
 &\simeq \exp[-ih^2g(\nabla \times \mathbf{A})_z^{i,j}],
 \end{aligned}$$

3次元では

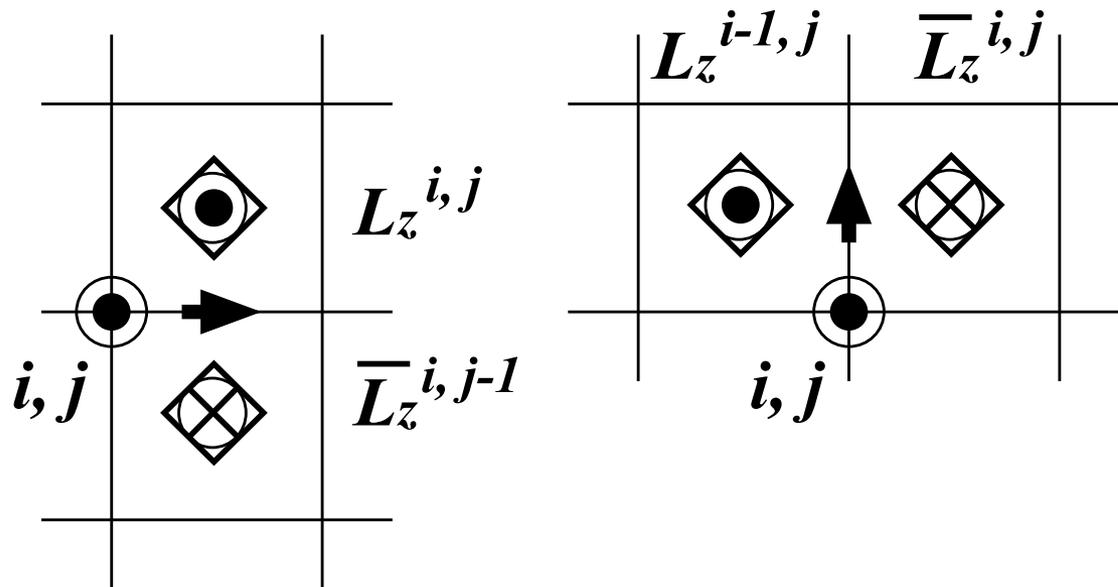


$$L_x^{i,j,k} = U_y^{i,j,k} U_z^{i,j+1,k} \overline{U}_y^{i,j,k+1} \overline{U}_z^{i,j,k},$$

$$L_y^{i,j,k} = U_z^{i,j,k} U_x^{i,j,k+1} \overline{U}_z^{i+1,j,k} \overline{U}_x^{i,j,k},$$

$$L_z^{i,j,k} = U_x^{i,j,k} U_y^{i+1,j,k} \overline{U}_x^{i,j+1,k} \overline{U}_y^{i,j,k}.$$

## 2次元



$$L_z^{i,j} \bar{L}_z^{i,j-1} \simeq \exp[-ih^3 g (\nabla \times \nabla \times \mathbf{A})_x^{i,j}],$$

$$\bar{L}_z^{i,j} L_z^{i-1,j} \simeq \exp[-ih^3 g (\nabla \times \nabla \times \mathbf{A})_y^{i,j}],$$

$$\begin{aligned}\tau \frac{\partial \psi}{\partial t} &= \frac{1}{2} \mathbf{D}^2 \psi + \eta(1 - |\psi|^2) \psi, \\ \frac{\partial \mathbf{A}}{\partial t} &= g \text{Im} [\bar{\psi} \mathbf{D} \psi] - \nabla \times \nabla \times \mathbf{A}.\end{aligned}$$

Semi-discretized equation ( 2次元 )

$$\begin{aligned}\tau \frac{\partial \psi^{i,j}}{\partial t} &= \frac{1}{2h^2} (U_x^{i,j} \psi^{i+1,j} + \bar{U}_x^{i-1,j} \psi^{i-1,j} + U_y^{i,j} \psi^{i,j+1} + \bar{U}_y^{i,j-1} \psi^{i,j-1} - 4\psi^{i,j}) \\ &\quad + \eta(1 - |\psi^{i,j}|^2) \psi^{i,j}, \\ \frac{\partial U_x^{i,j}}{\partial t} &= -ig \text{Im} [\bar{\psi}^{i,j} U_x^{i,j} \psi^{i+1,j}] U_x^{i,j} - \frac{1}{h^2} (\bar{L}_z^{i,j-1} L_z^{i,j} - 1) U_x^{i,j}, \\ \frac{\partial U_y^{i,j}}{\partial t} &= -ig \text{Im} [\bar{\psi}^{i,j} U_y^{i,j} \psi^{i,j+1}] U_y^{i,j} - \frac{1}{h^2} (\bar{L}_z^{i,j} L_z^{i-1,j} - 1) U_y^{i,j},\end{aligned}$$

## Semi-discretized equation ( 3次元 )

$$\begin{aligned} \tau \frac{\partial \psi^{i,j,k}}{\partial t} = & \frac{1}{2h^2} (U_x^{i,j,k} \psi^{i+1,j,k} + \bar{U}_x^{i-1,j,k} \psi^{i-1,j,k} + U_y^{i,j,k} \psi^{i,j+1,k} + \bar{U}_y^{i,j-1,k} \psi^{i,j-1,k} \\ & + U_z^{i,j,k} \psi^{i,j,k+1} + \bar{U}_z^{i,j,k-1} \psi^{i,j,k-1} - 6\psi^{i,j,k}) + \eta(1 - |\psi^{i,j,k}|^2) \psi^{i,j,k}. \end{aligned}$$

$$\frac{\partial U_x^{i,j,k}}{\partial t} = -ig \text{Im} [\bar{\psi}^{i,j,k} U_x^{i,j,k} \psi^{i+1,j,k}] - \frac{1}{h^2} (\bar{L}_y^{i,j,k} L_y^{i,j,k-1} L_z^{i,j,k} \bar{L}_z^{i,j-1,k} - 1),$$

$$\frac{\partial U_y^{i,j,k}}{\partial t} = -ig \text{Im} [\bar{\psi}^{i,j,k} U_y^{i,j,k} \psi^{i,j+1,k}] - \frac{1}{h^2} (\bar{L}_z^{i,j,k} L_z^{i-1,j,k} L_x^{i,j,k} \bar{L}_z^{i,j,k-1} - 1),$$

$$\frac{\partial U_z^{i,j,k}}{\partial t} = -ig \text{Im} [\bar{\psi}^{i,j,k} U_z^{i,j,k} \psi^{i,j,k+1}] - \frac{1}{h^2} (\bar{L}_x^{i,j,k} L_x^{i,j-1,k} L_y^{i,j,k} \bar{L}_y^{i-1,j,k} - 1),$$

# コーディング (実装)

表現に類似性をもたせる

```
a = a + b · c · d + e · f
```

```
a.equal( a.plus( b.mul(c).mul(d) ).plus( e.mul(f) ) );
```

```
Complex plus( Complex z ) {  
    return new Complex( x + z.x, y + z.y );  
}
```

## 「複素数クラス」を定義

```
1 class Complex{
2     /// x: real part
3     /// y: imaginary part
4     double x, y;
5     Complex(double x0, double y0){
6         x = x0; // real part
7         y = y0; // imaginary part
8     }
9     Complex(Complex z){
10        x = z.x;
11        y = z.y;
12    }
13    void setPolar(double R, double theta){
14        x = R * Math.cos(theta);
15        y = R * Math.sin(theta);
16    }
17    void equal(Complex z){
18        x = z.x;
19        y = z.y;
20    }
21    void equal(double x0, double y0){
22        x = x0;
23        y = y0;
24    }
25    Complex plus(Complex z){
26        return new Complex(x + z.x, y + z.y);
27    }
28    Complex minus(Complex z){
29        return new Complex(x - z.x, y - z.y);
30    }
31    Complex mul(double a){
32        return new Complex(a*x, a*y);
33    }
34    Complex mul(Complex z){
35        double x1 = z.x;
36        double y1 = z.y;
37        double x2 = x;
38        double y2 = y;
39        return new Complex(x1*x2 - y1*y2, x1*y2 + y1*x2);
40    }
41    Complex div(double a){
42        return new Complex(x/a, y/a);
43    }
44    Complex conj(){
45        return new Complex(x, -y);
46    }
47 }
```

```
Complex mul (Complex z) {
    double x1 = z.x;
    double y1 = z.y;
    double x2 = x;
    double y2 = y;
    return new Complex(x1*x2 - y1*y2, x1*y2 + y1*x2);
}
```

$$L_x^{i,j,k} = U_y^{i,j,k} U_z^{i,j+1,k} \overline{U_y^{i,j,k+1}} \overline{U_z^{i,j,k}},$$

$$L_y^{i,j,k} = U_z^{i,j,k} U_x^{i,j,k+1} \overline{U_z^{i+1,j,k}} \overline{U_x^{i,j,k}},$$

$$L_z^{i,j,k} = U_x^{i,j,k} U_y^{i+1,j,k} \overline{U_x^{i,j+1,k}} \overline{U_y^{i,j,k}}.$$

```
Complex Lz(Complex[][][] Ux, Complex[][][] Uy, Complex[][][] Uz, int i, int j, int k) {
    return Ux[i][j][k].mul(Uy[i+1][j][k]).mul(Ux[i][j+1][k].conj()).mul(Uy[i][j][k].conj());
}
```

$$\tau \frac{\partial \psi^{i,j,k}}{\partial t} = \frac{1}{2h^2} (U_x^{i,j,k} \psi^{i+1,j,k} + \overline{U_x}^{i-1,j,k} \psi^{i-1,j,k} + U_y^{i,j,k} \psi^{i,j+1,k} + \overline{U_y}^{i,j-1,k} \psi^{i,j-1,k} + U_z^{i,j,k} \psi^{i,j,k+1} + \overline{U_z}^{i,j,k-1} \psi^{i,j,k-1} - 6\psi^{i,j,k}) + \eta(1 - |\psi^{i,j,k}|^2) \psi^{i,j,k}.$$

```

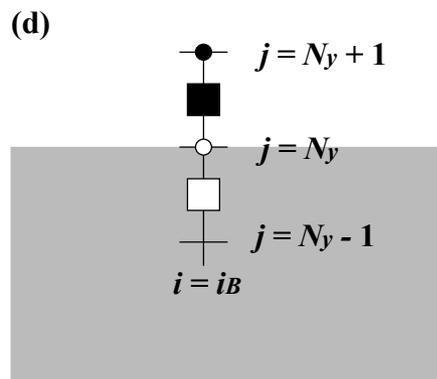
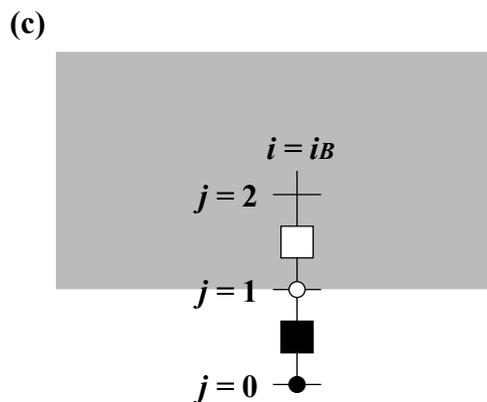
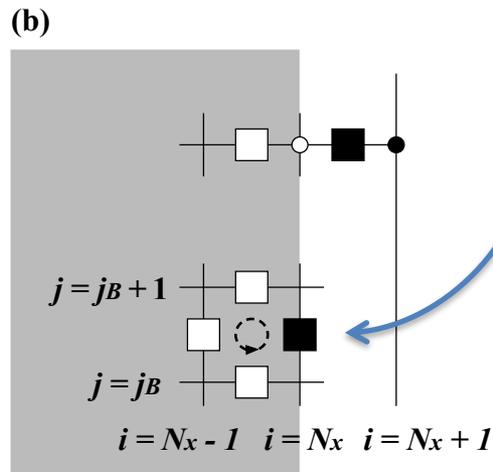
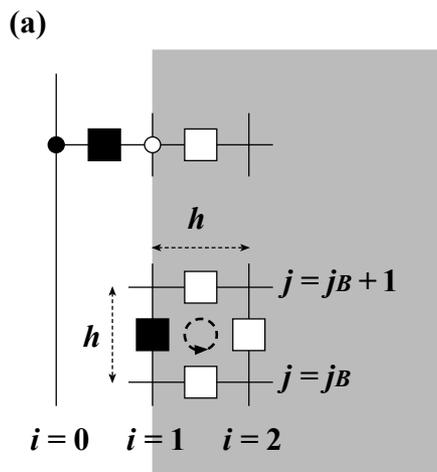
//////////
void do_Laplacian(Complex[][][] psi, Complex[][][] Ux, Complex[][][] Uy, Complex[][][] Uz, double dt, double D) {
    for (int k = 1; k <= Nz; k++) {
        for (int j = 1; j <= Ny; j++) {
            for (int i = 1; i <= Nx; i++) {
                psi_s[i][j][k].equal (psi[i][j][k].plus (
                    ((psi[i+1][j][k].mul (Ux[i][j][k])).plus (psi[i-1][j][k].mul (Ux[i-1][j][k].conj ()))
                    .plus (psi[i][j+1][k].mul (Uy[i][j][k])).plus (psi[i][j-1][k].mul (Uy[i][j-1][k].conj ()))
                    .plus (psi[i][j][k+1].mul (Uz[i][j][k])).plus (psi[i][j][k-1].mul (Uz[i][j][k-1].conj ()))
                    .minus (psi[i][j][k].mul (6.0) ) )
                    .mul (dt * D / (h * h) )
                )
            )
        }
    }
}

for (int k = 1; k <= Nz; k++) {
    for (int j = 1; j <= Ny; j++) {
        for (int i = 1; i <= Nx; i++) {
            psi[i][j][k].equal (psi_s[i][j][k]);
        }
    }
}
}
}
//////////

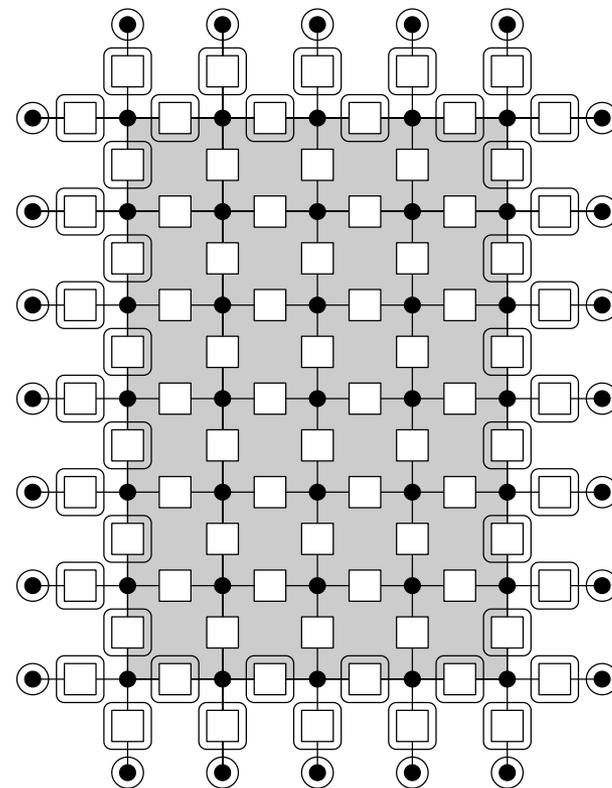
```

## ラプラシアン演算の実装

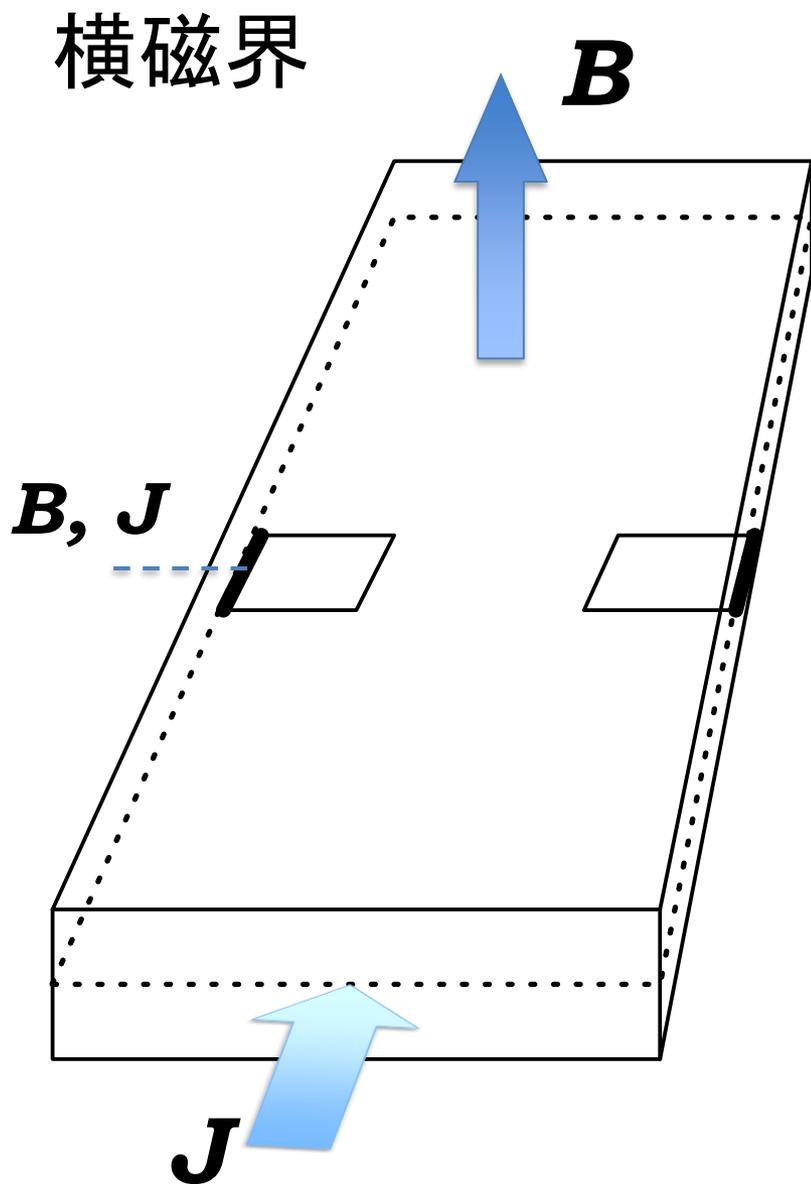
# 境界条件



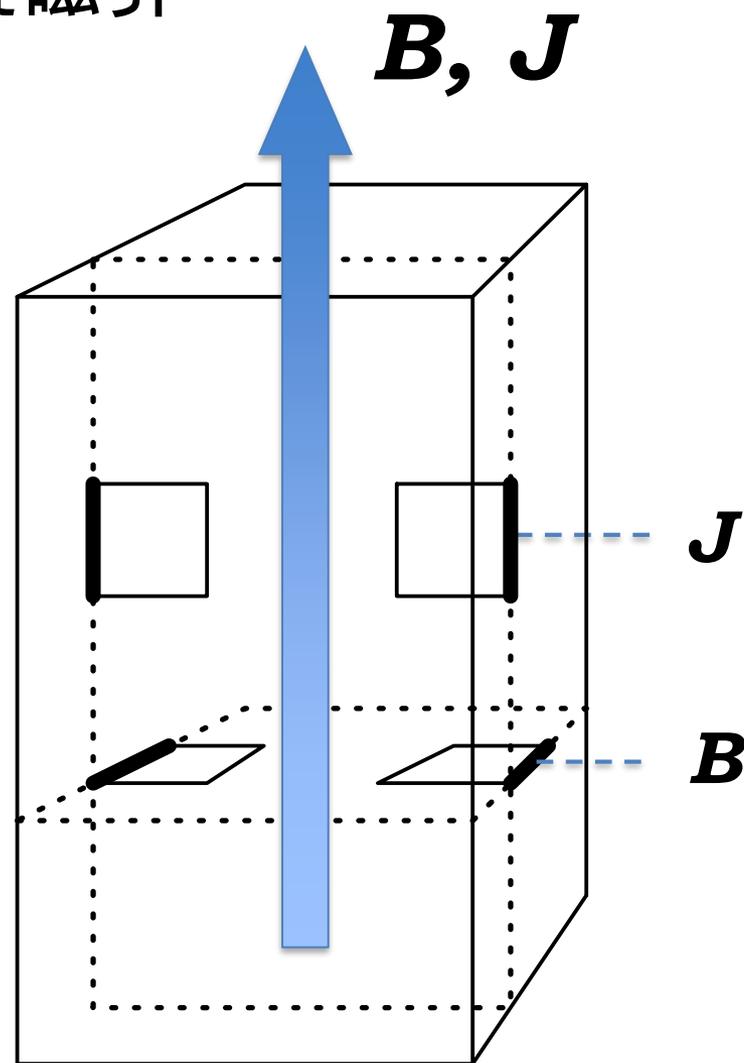
境界での磁束密度指定



横磁界



縦磁界



## 境界条件の実装

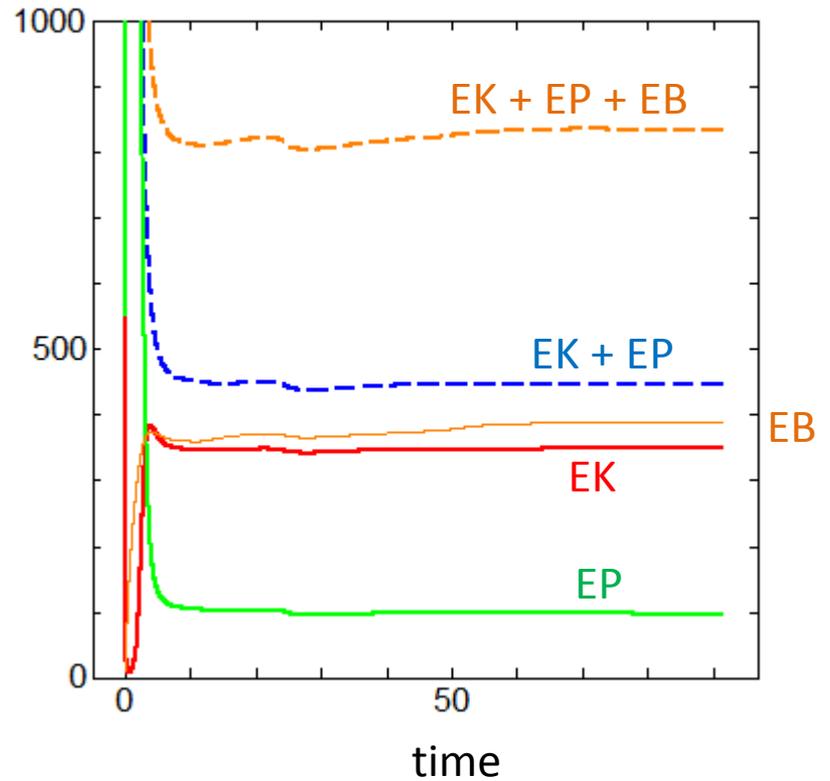
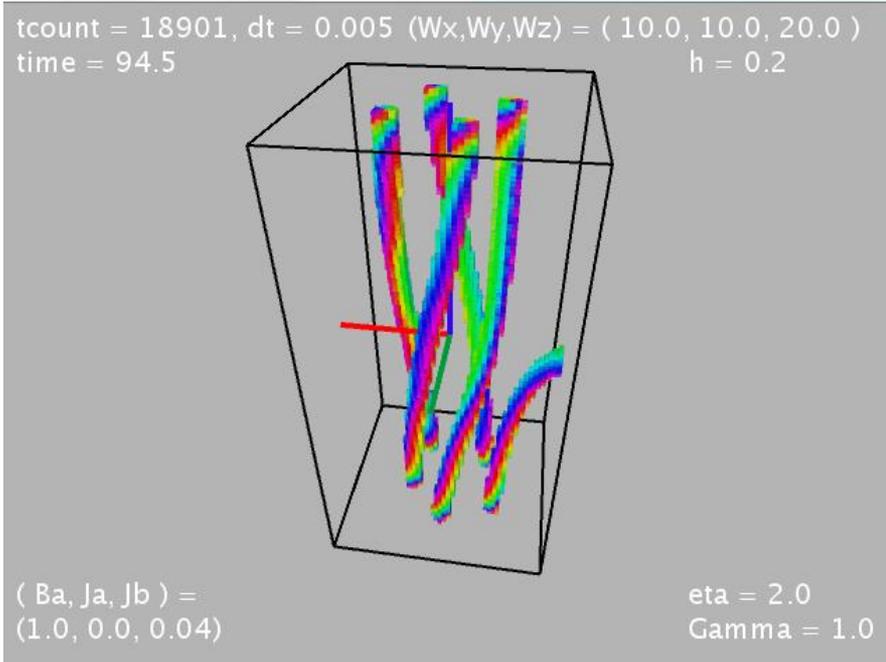
```
/// constraints by the applied field and current
/// y-z plane ( Bz-Uy-component ) set Ja = 0 when longitudinal
for(int k = 1; k <= Nz; k++) {
  for(int j = 1; j <= Ny-1; j++) {
    ex.setPolar(1.0, h*h*(Ba + Ja*SSx/2.0));
    Uy[1][j][k].equal(Uy[2][j][k].mul(Ux[1][j+1][k].conj()).mul(Ux[1][j][k]).mul(ex));
    ex.setPolar(1.0, -h*h*(Ba - Ja*SSx/2.0));
    Uy[Nx][j][k].equal(Uy[Nx-1][j][k].mul(Ux[Nx-1][j+1][k]).mul(Ux[Nx-1][j][k].conj()).mul(ex));
  }
}
/// z-x plane ( Bz-Ux-component ) omit when transverse
for(int i = 1; i <= Nx-1; i++) {
  for(int k = 1; k <= Nz; k++) {
    ex.setPolar(1.0, -h*h*Ba);
    Ux[i][1][k].equal( (Uy[i+1][1][k].conj()).mul(Ux[i][2][k]).mul(Uy[i][1][k]).mul(ex) );
    ex.setPolar(1.0, h*h*Ba);
    Ux[i][Ny][k].equal( Ux[1][Ny-1][k].mul(Uy[i+1][Ny-1][k]).mul(Uy[i][Ny-1][k].conj()).mul(ex) );
  }
}
/// y-z plane ( J=(0,0,Jb)-Uz-component ) longitudinal
for(int k = 1; k <= Nz-1; k++) {
  for(int j = 1; j <= Ny; j++) {
    ex.setPolar(1.0, h*h*Jb*SSx/2.0);
    Uz[1][j][k].equal( Ux[1][j][k].mul(Uz[2][j][k]).mul(Ux[1][j][k+1].conj()).mul(ex) );
    Uz[Nx][j][k].equal( (Ux[Nx-1][j][k].conj()).mul(Uz[Nx-1][j][k]).mul(Ux[Nx-1][j][k+1]).mul(ex) );
  }
}
/// z-x plane ( J=(0,0,Jb)-Uz-component ) longitudinal
for(int i = 1; i <= Nx; i++) {
  for(int k = 1; k <= Nz-1; k++) {
    ex.setPolar(1.0, h*h*Jb*SSy/2.0);
    Uz[i][1][k].equal( Uy[i][1][k].mul(Uz[i][2][k]).mul(Uy[i][1][k+1].conj()).mul(ex) );
    Uz[i][Ny][k].equal( (Uy[i][Ny-1][k].conj()).mul(Uz[i][Ny-1][k]).mul(Uy[i][Ny-1][k]).mul(ex) );
  }
}
```

# シミュレーション例

- FEM ( $\phi$  given) (p00)
- 2次元横磁界 ( $\phi$  given) (p01)
- 2次元横磁界 ( $\phi$  given) (p02)
- 2次元横磁界 (リンク変数,  $\phi$ ゼロゲージ) (p03) \* 電場表示
- 3次元縦磁界 (リンク変数,  $\phi$ ゼロゲージ) (p04)

\* 電場の表示はリンク変数の偏角の時間微分を利用

# エネルギー



# 今後の課題・展開

- アルゴリズムの検討・改良
- 並列化(高速化)  
e.g. by GPGPU ( Java + OpenCL )
- ピンニング
- Two-component 系