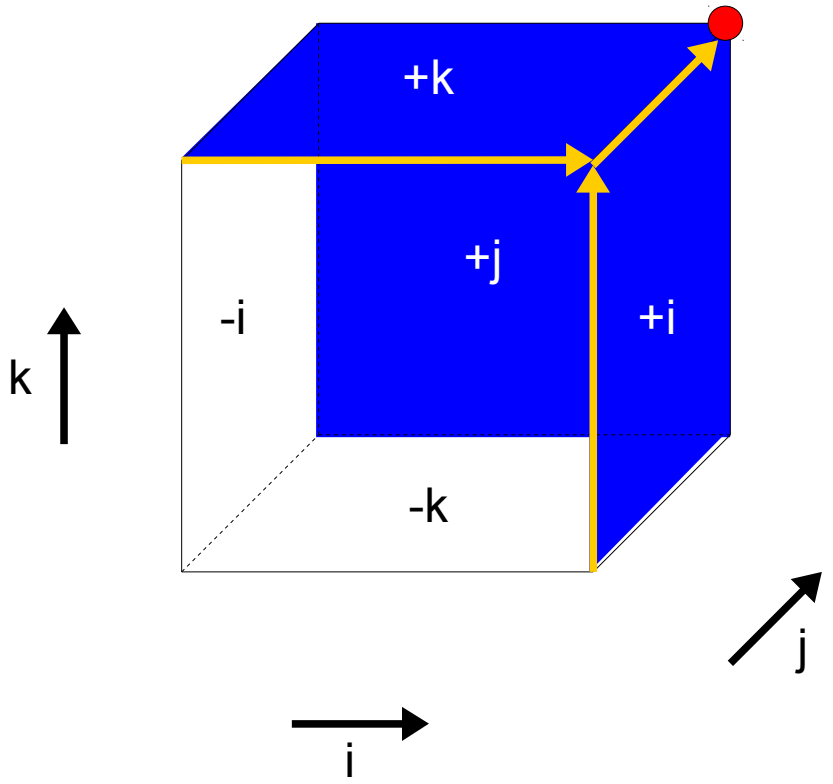
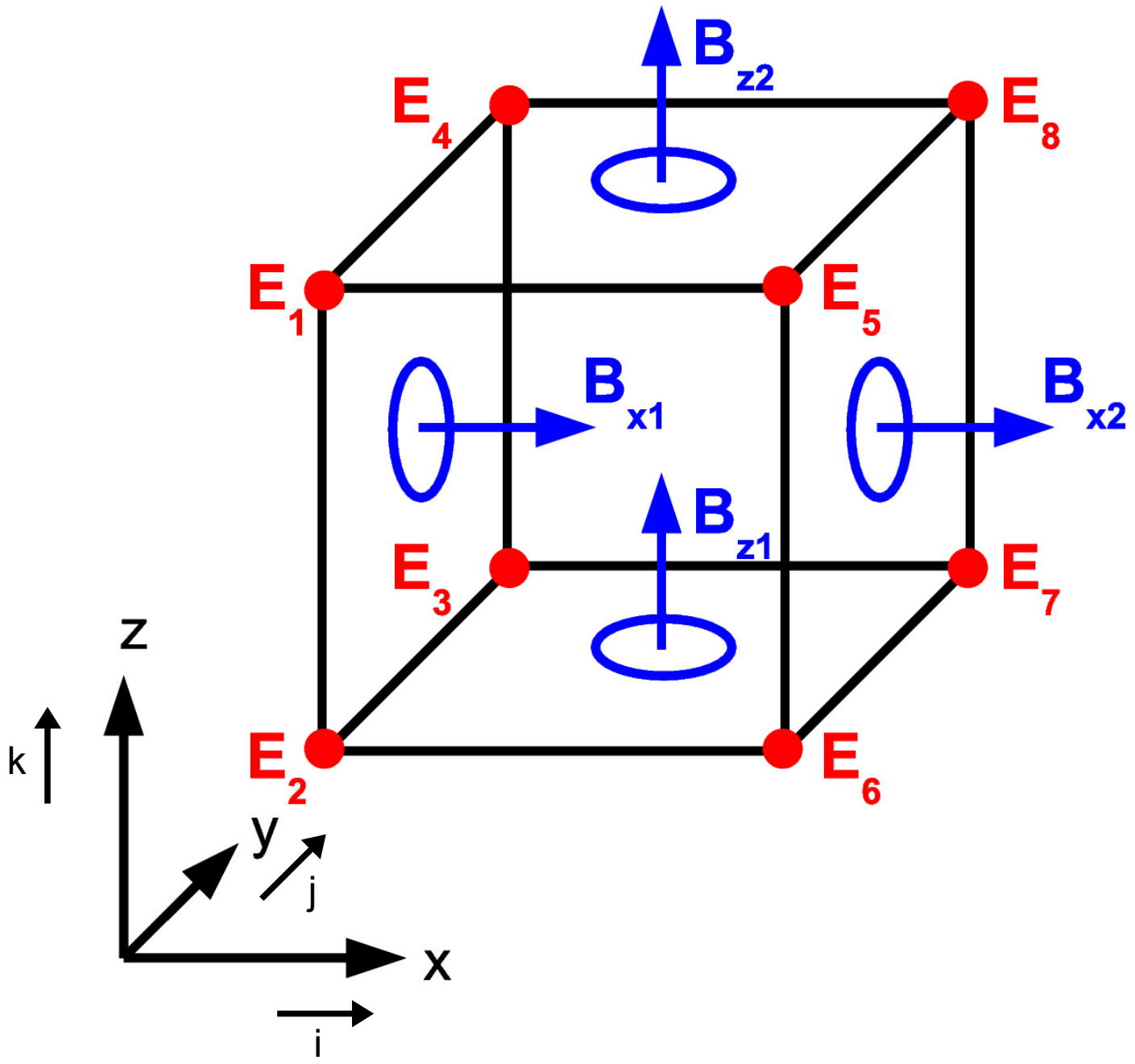
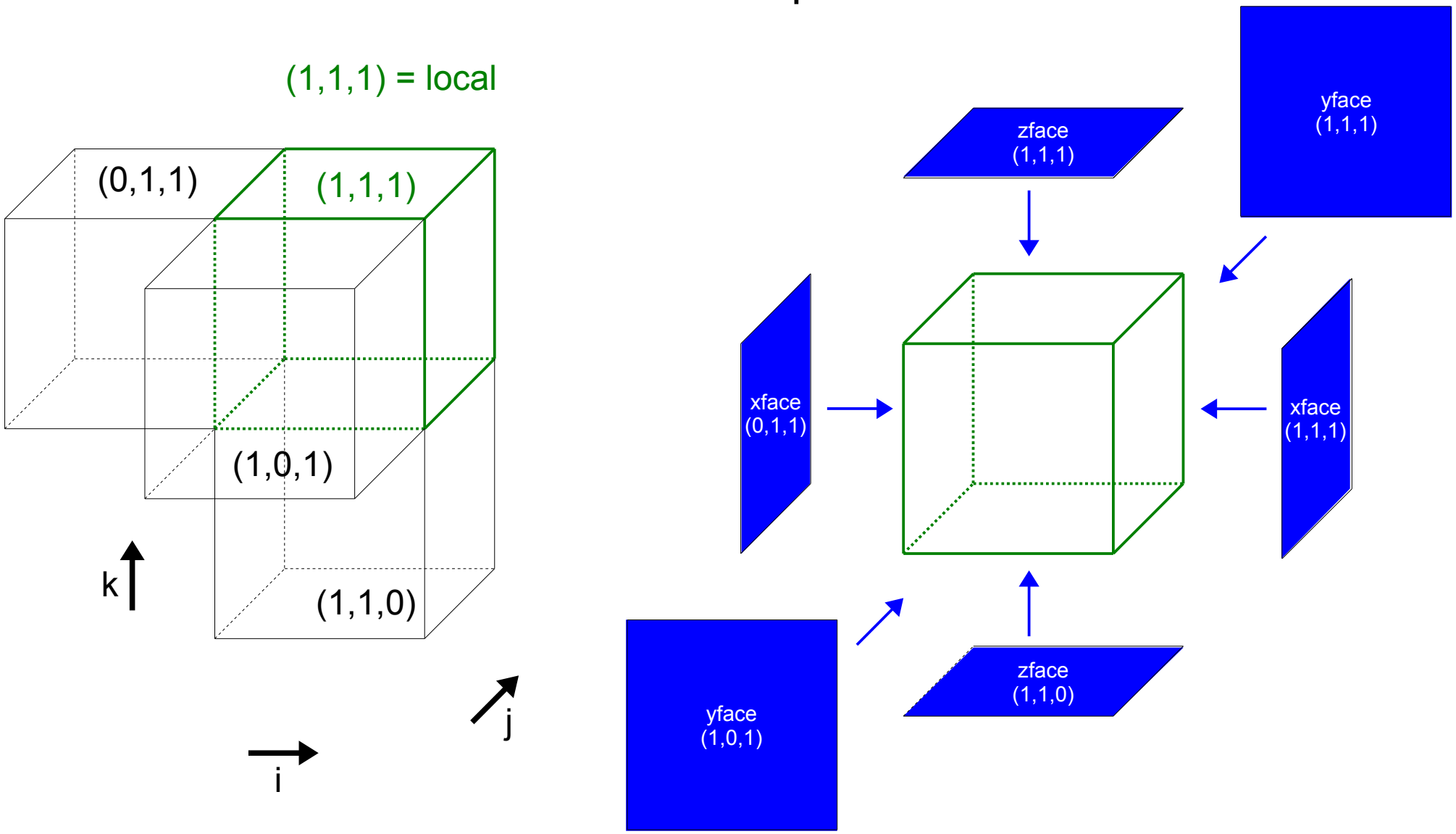
RHybrid cell indexing

- cell volume average
  - Bx, By, Bz
- upper front corner node (+i,+j,+k)
  - Ex, Ey, Ez
  - Bx, By, Bz
  - Jx, Jy, Jz
- three face surface averages
  - Φx = dA × Bx (+i face)
  - Φx = dA × By (+j face)
  - Φz = dA × Bz (+k face)
- edges
  - Jx, Jy, Jz



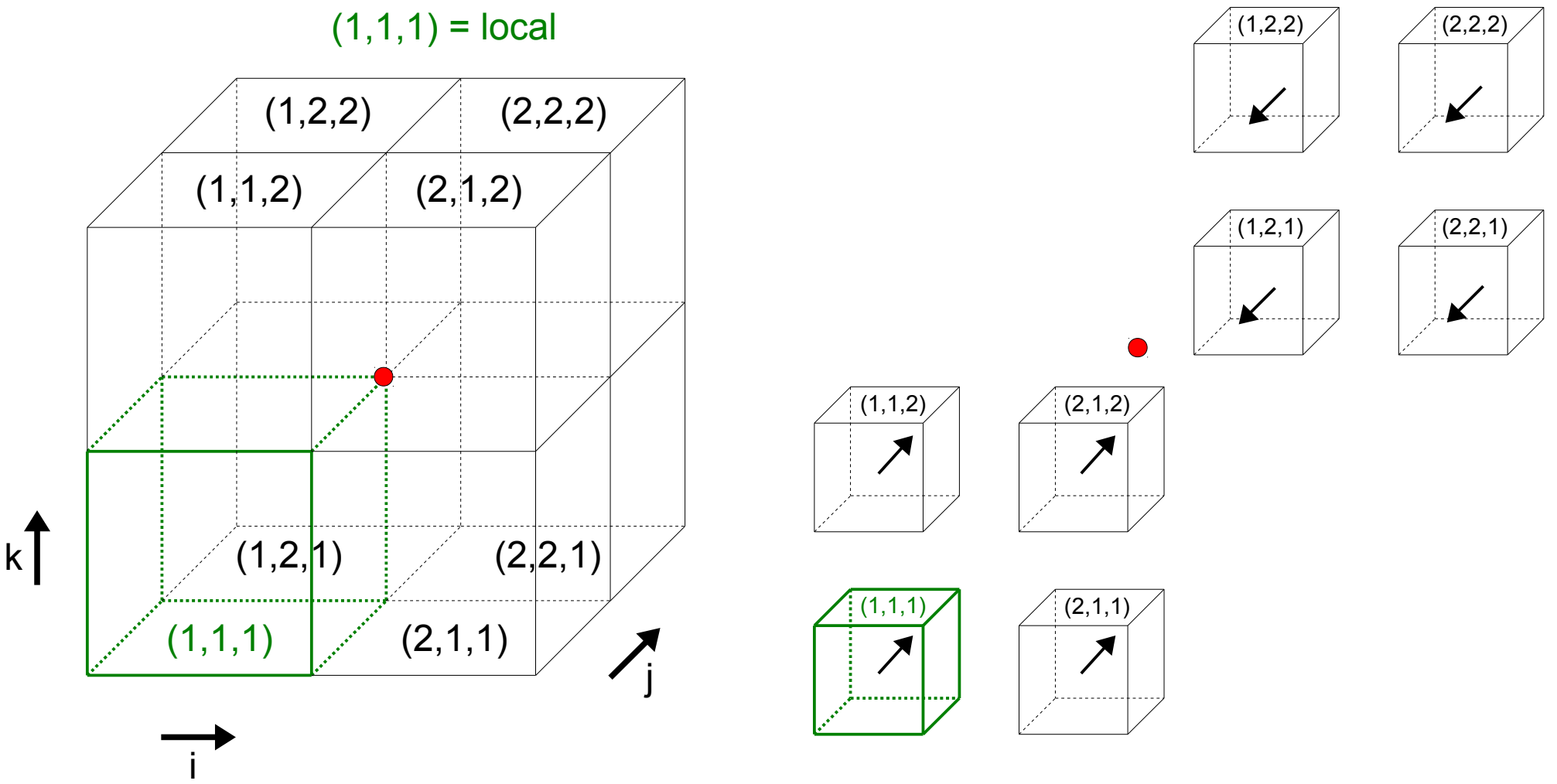

B z2
E 4
E 8
E 1
E 5
B x1
B x2
B z1
E 3
E 7
E 2
E 6
z
k
y
j
x
i

face to cell interpolation



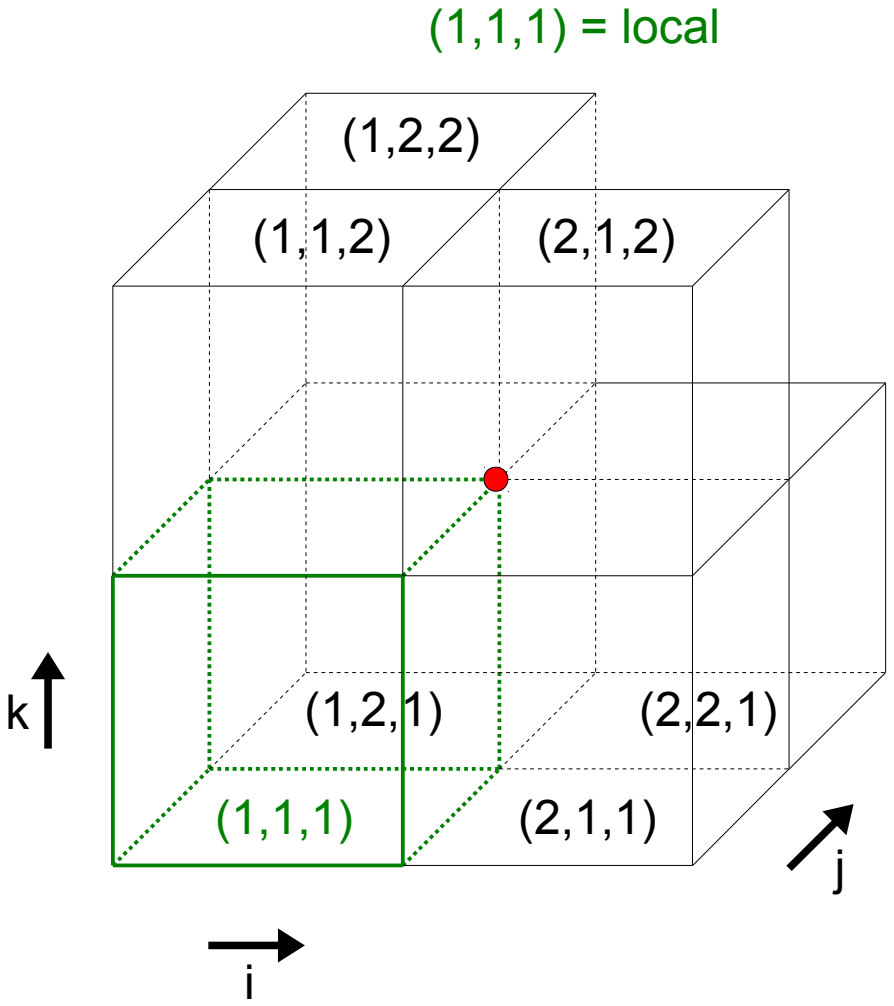

```
celldata_x =   0.5 × ( facedata_x(1,1,1) + facedata_x(0,1,1) )
celldata_y =   0.5 × ( facedata_y(1,1,1) + facedata_y(1,0,1) )
celldata_z =   0.5 × ( facedata_z(1,1,1) + facedata_z(1,1,0) )
```

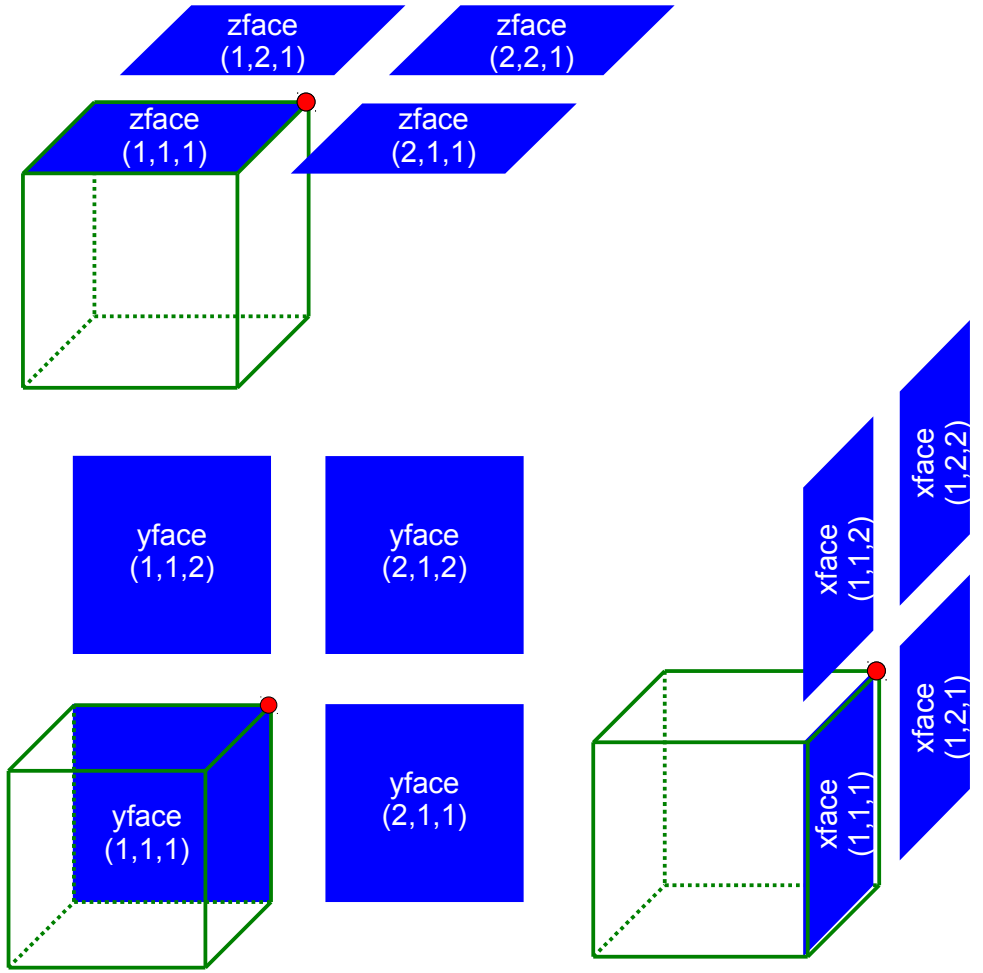
cell to node interpolation




```
nodedata_i   = 0.125 × (  celldata_i(1,1,1) + celldata_i(2,1,1)
                        + celldata_i(1,1,2) + celldata_i(2,1,2)
                        + celldata_i(1,2,1) + celldata_i(2,2,1)
                        + celldata_i(1,2,2) + celldata_i(2,2,2) )
```

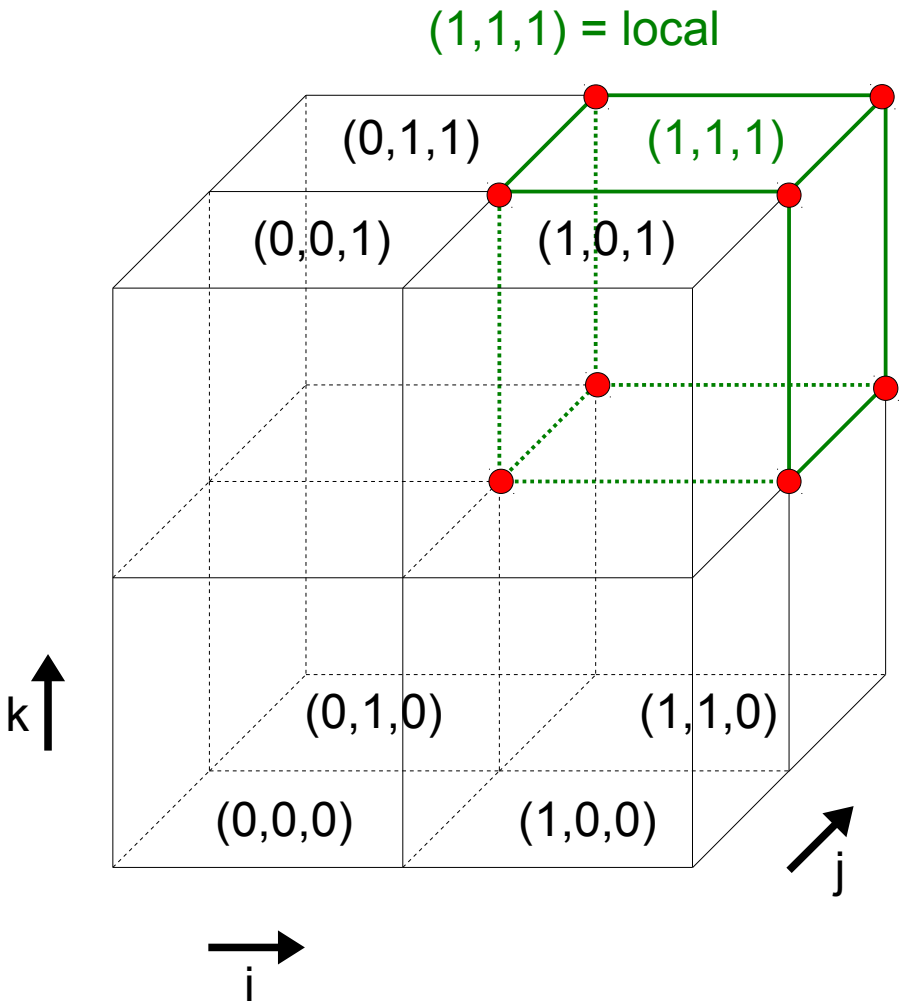
face to node interpolation

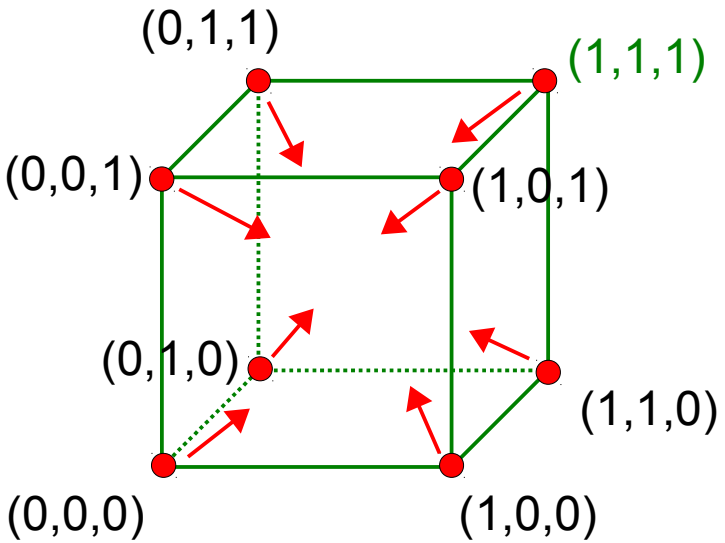





```
nodedata_x = 0.25 × ( facedata_x(1,1,1) + facedata_x(1,1,2) + facedata_x(1,2,2) + facedata_x(1,2,1) )
nodedata_y = 0.25 × ( facedata_y(1,1,1) + facedata_y(1,1,2) + facedata_y(2,1,2) + facedata_y(2,1,1) )
nodedata_z = 0.25 × ( facedata_z(1,1,1) + facedata_z(1,2,1) + facedata_z(2,2,1) + facedata_z(2,1,1) )
```

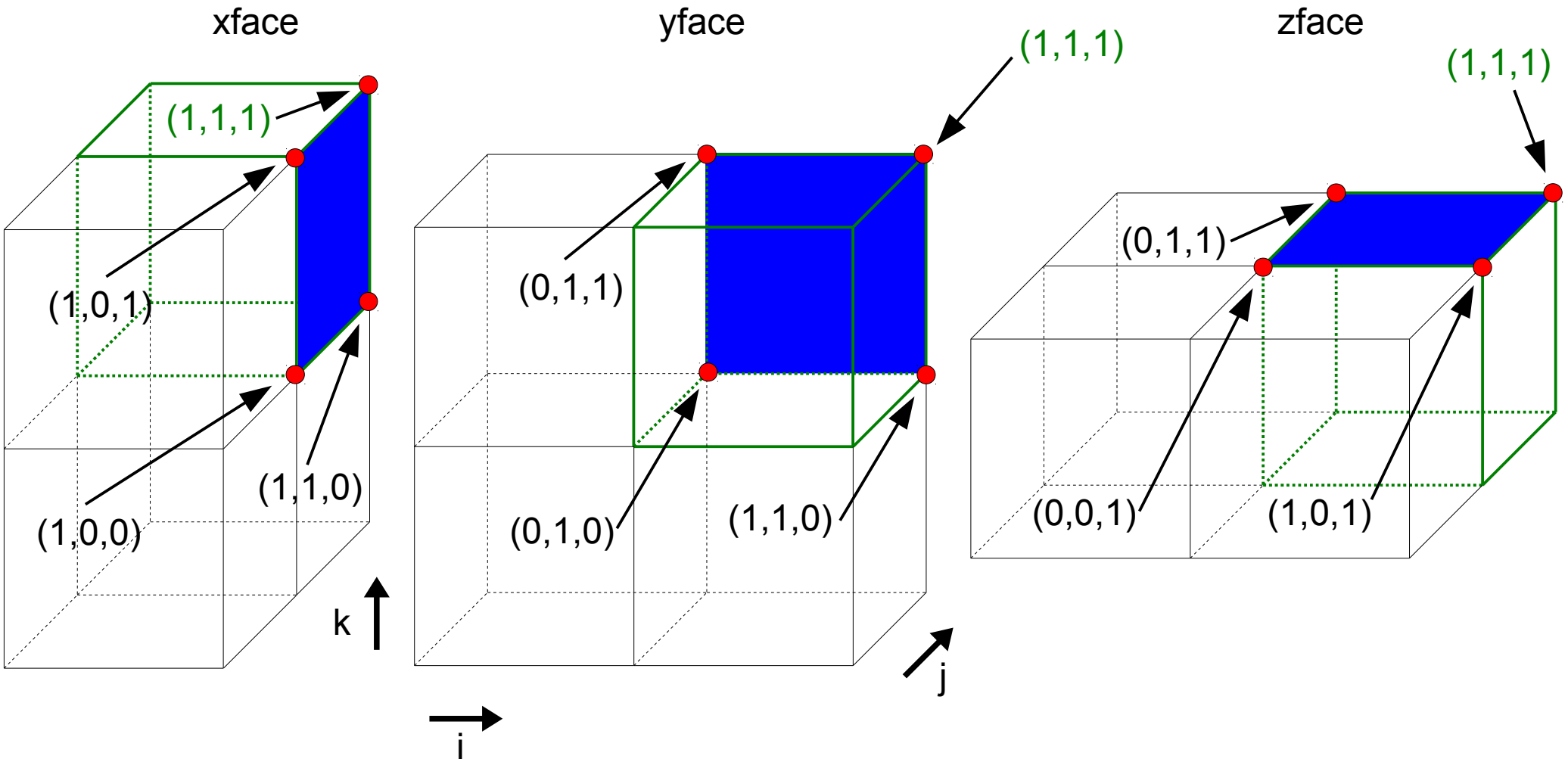
node to cell interpolation





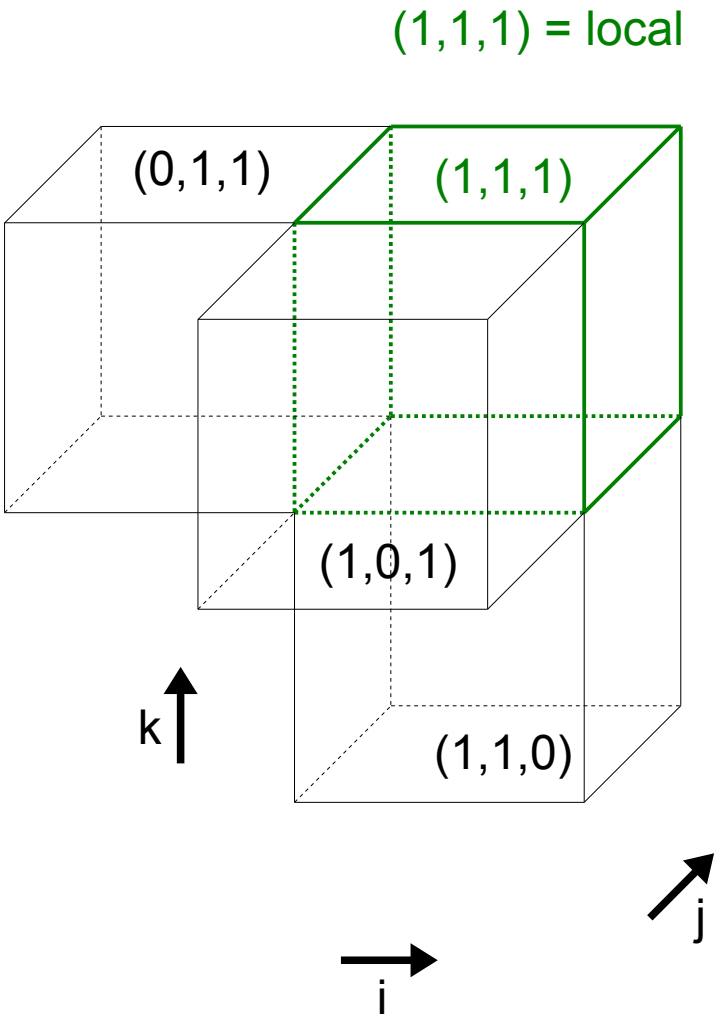

```
celldata_i = 0.125 × (nodedata_i(1,1,1) + nodedata_i(0,1,1)
                    +  nodedata_i(0,0,1) + nodedata_i(1,0,1)
                    +  nodedata_i(1,1,0) + nodedata_i(0,1,0)
                    +  nodedata_i(0,0,0) + nodedata_i(1,0,0) )
```

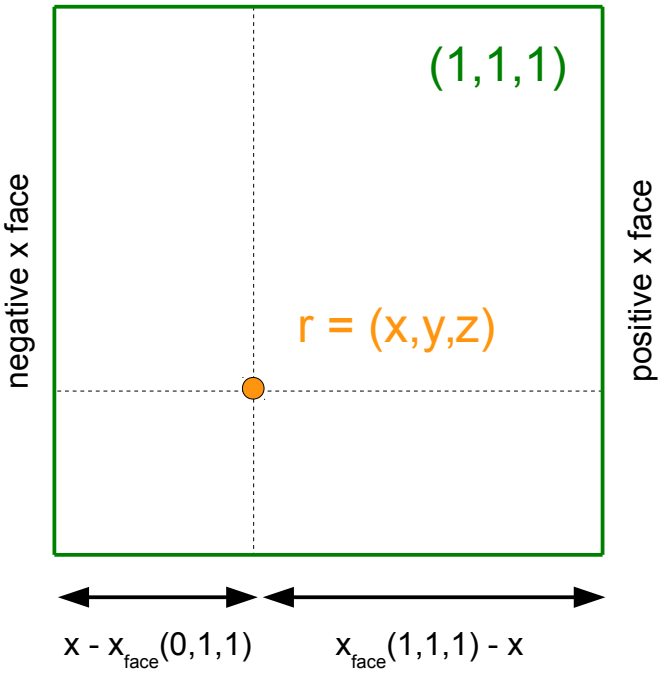
node to face interpolation




```
facedata_x = 0.25 × ( nodedata_x(1,1,1) + nodedata_x(1,0,1) + nodedata_x(1,0,0) + nodedata_x(1,1,0) )
facedata_y = 0.25 × ( nodedata_y(1,1,1) + nodedata_y(1,1,0) + nodedata_y(0,1,0) + nodedata_y(0,1,1) )
facedata_z = 0.25 × ( nodedata_z(1,1,1) + nodedata_z(0,1,1) + nodedata_z(0,0,1) + nodedata_z(1,0,1) )
```

face to r interpolation

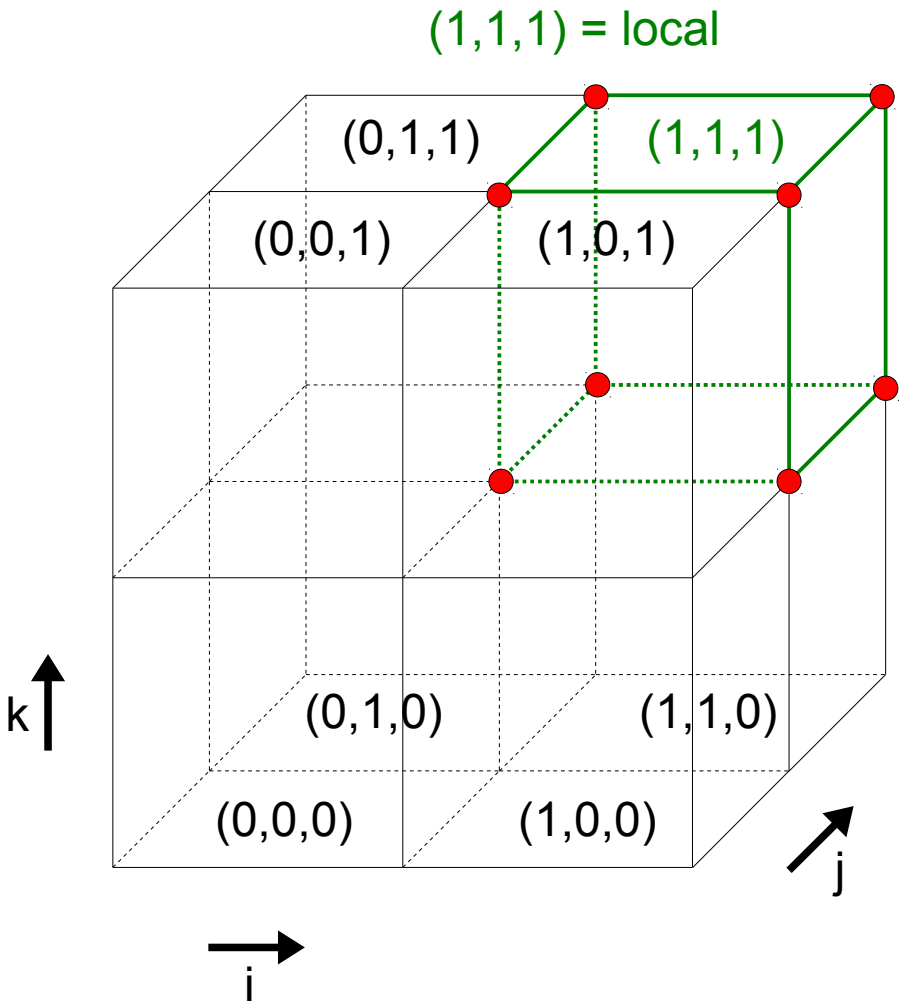


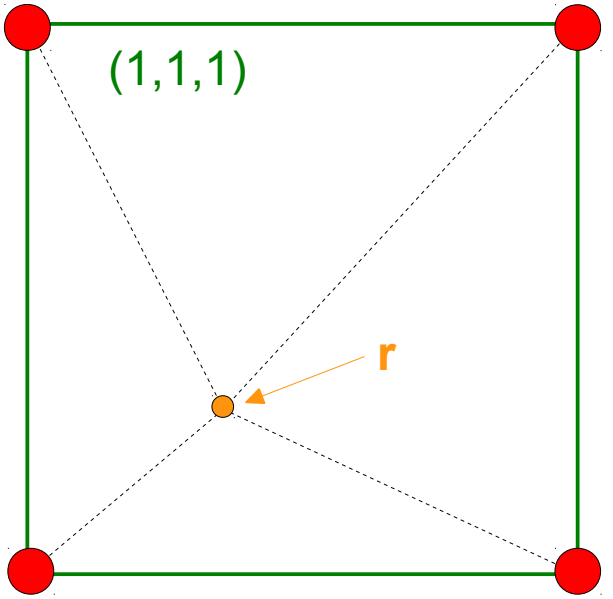


$$w111x = +(x - x_{face}(0,1,1))/dx$$
$$w111y = +(y - y_{face}(1,0,1))/dx$$
$$w111z = +(z - z_{face}(1,1,0))/dx$$
$$w011x = -(x - x_{face}(1,1,1))/dx$$
$$w101y = -(y - y_{face}(1,1,1))/dx$$
$$w110z = -(z - z_{face}(1,1,1))/dx$$

$$data\_x(r) = w111x \times facedata\_x(1,1,1) + w011x \times facedata\_x(0,1,1)$$
$$data\_y(r) = w111y \times facedata\_y(1,1,1) + w101y \times facedata\_y(1,0,1)$$
$$data\_z(r) = w111z \times facedata\_z(1,1,1) + w110z \times facedata\_z(1,1,0)$$

node to r interpolation



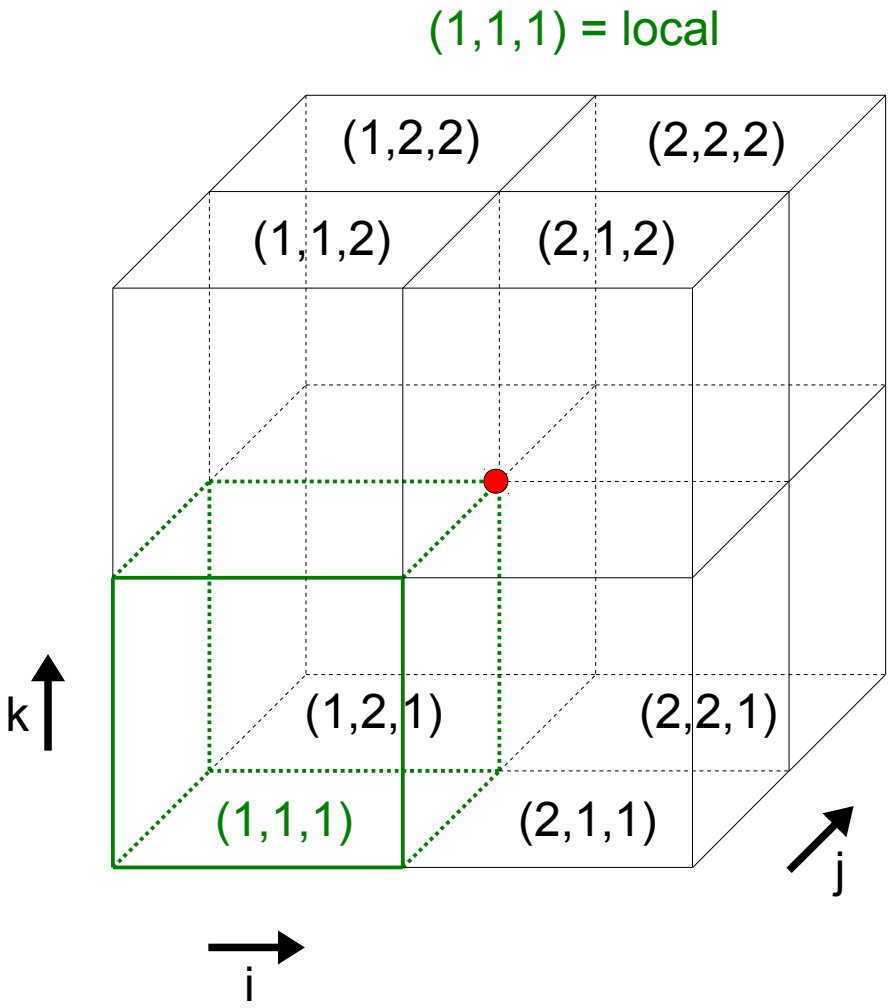


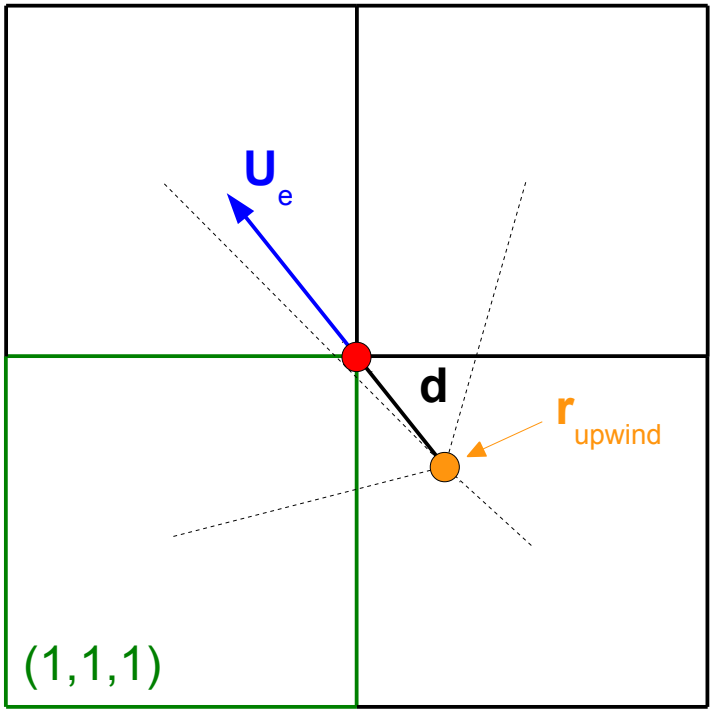
Weight factors: $w\_i = 1/|\mathbf{r}_{node\_i} - \mathbf{r}|$

Sum of weights: wsum = sum_i(w_i)

```
data(r) = (  w(1,1,1) × nodedata(1,1,1)   + w(0,1,1) × nodedata(0,1,1) +
             w(1,0,1) × nodedata(1,0,1)   + w(0,0,1) × nodedata(0,0,1) +
             w(1,1,0) × nodedata(1,1,0)   + w(0,1,0) × nodedata(0,1,0) +
             w(1,0,0) × nodedata(1,0,0)   + w(0,0,0) × nodedata(0,0,0) )/wsum
```

upwind node data





Displacement vector: $\mathbf{d} = 0.5 \times \mathbf{U}_e/|U_e|$

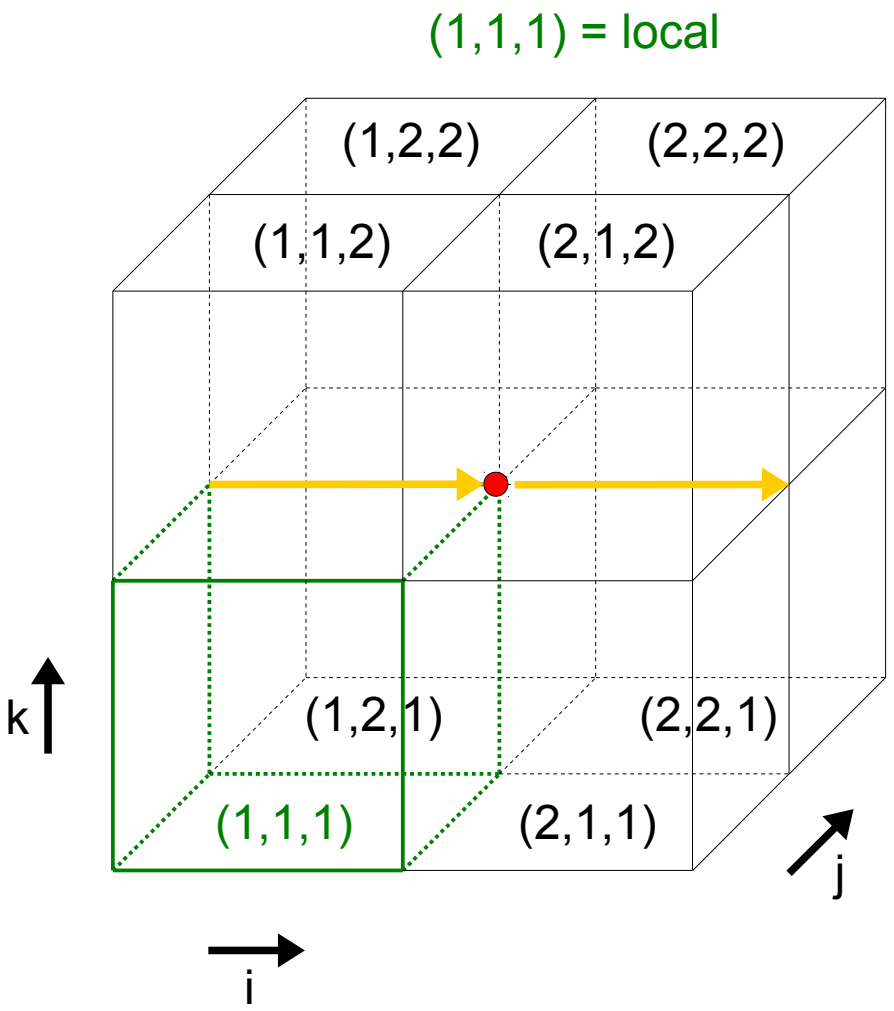
Upwind position: $\mathbf{r}_{upwind} = \mathbf{r}_{node} - \mathbf{d}$

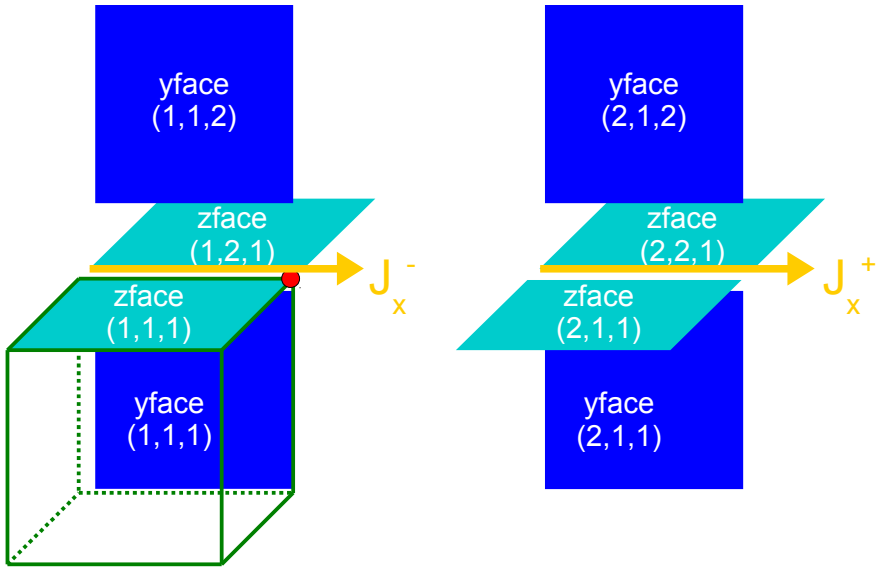
Weight factors: $w\_i = 1/|\mathbf{r}_{cell\_i} - \mathbf{r}_{upwind}|$

Sum of weights: $wsum = sum\_i(w\_i)$

```
Nodedata(1,1,1) → nodedata(1,1,1) =
                (   w(1,1,1) × celldata(1,1,1) + w(1,1,2) × celldata(1,1,2) +
                    w(1,2,1) × celldata(1,2,1) + w(2,1,1) × celldata(2,1,1) +
                    w(1,2,2) × celldata(1,2,2) + w(2,2,1) × celldata(2,2,1) +
                    w(2,1,2) × celldata(2,1,2) + w(2,2,2) × celldata(2,2,2) )/wsum
```

Calculation of Node Jx



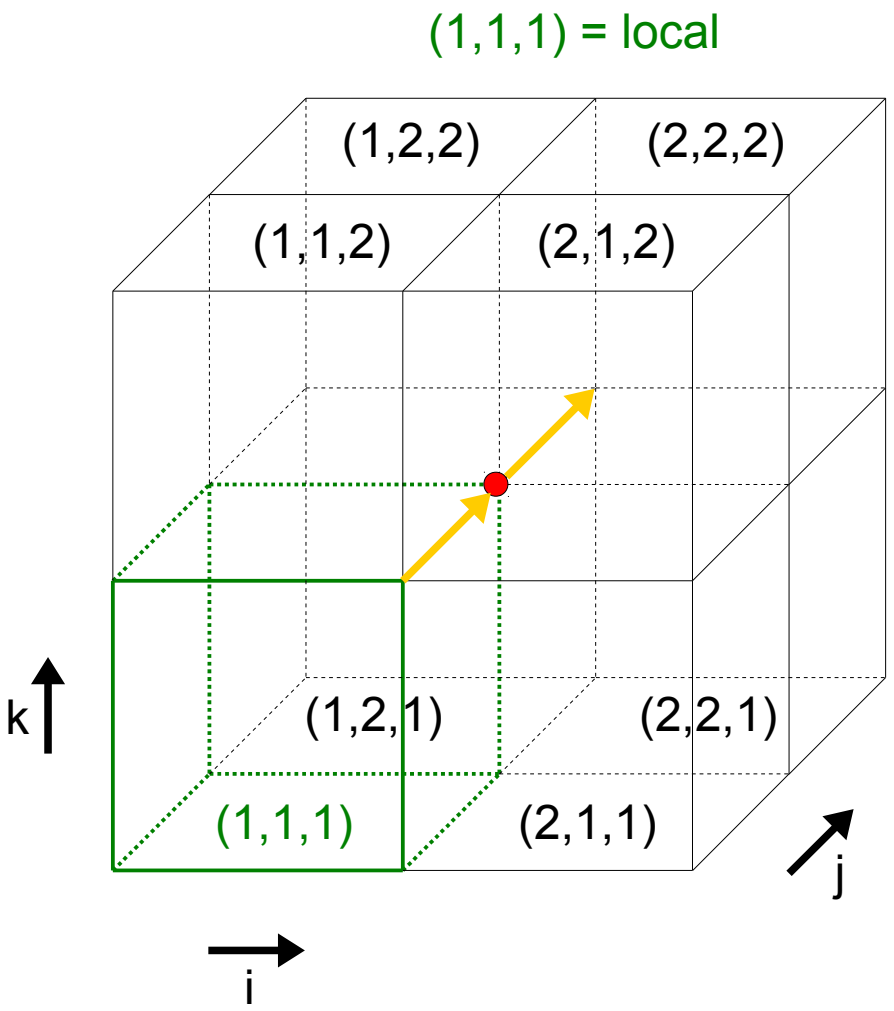


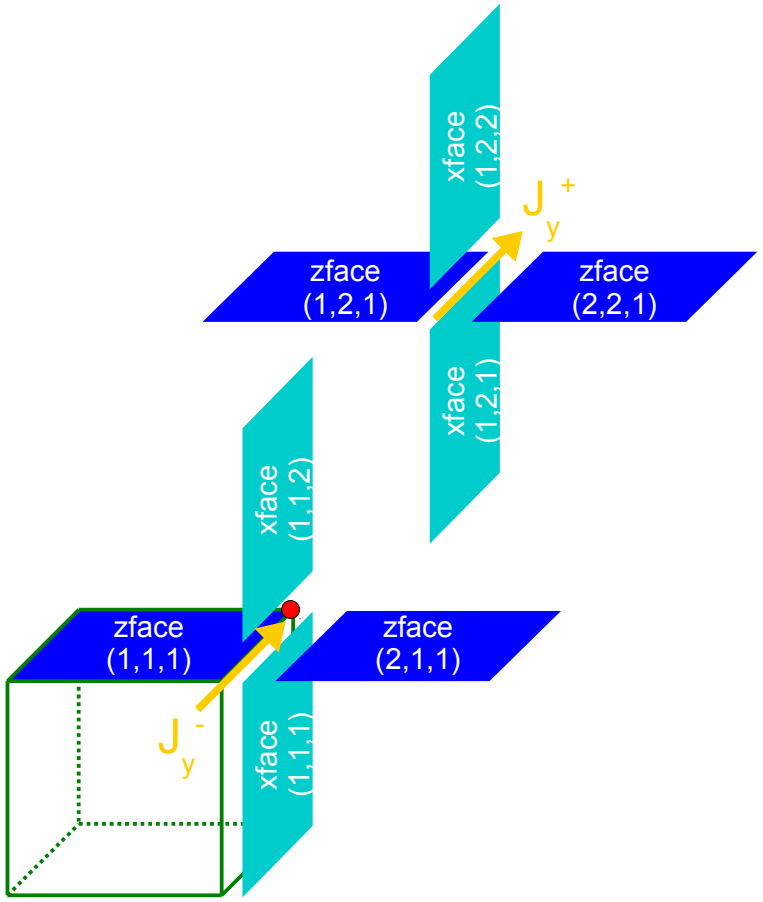
$$edgeJx^{-} = (-faceBz(1,1,1) + faceBy(1,1,1) + faceBz(1,2,1) - faceBy(1,1,2))/(dx \times mu0)$$

$$edgeJx^{+} = (-faceBz(2,1,1) + faceBy(2,1,1) + faceBz(2,2,1) - faceBy(2,1,2))/(dx \times mu0)$$

$$nodeJx = 0.5 \times (edgeJx^{+} + edgeJx^{-})$$

Calculation of Node Jy



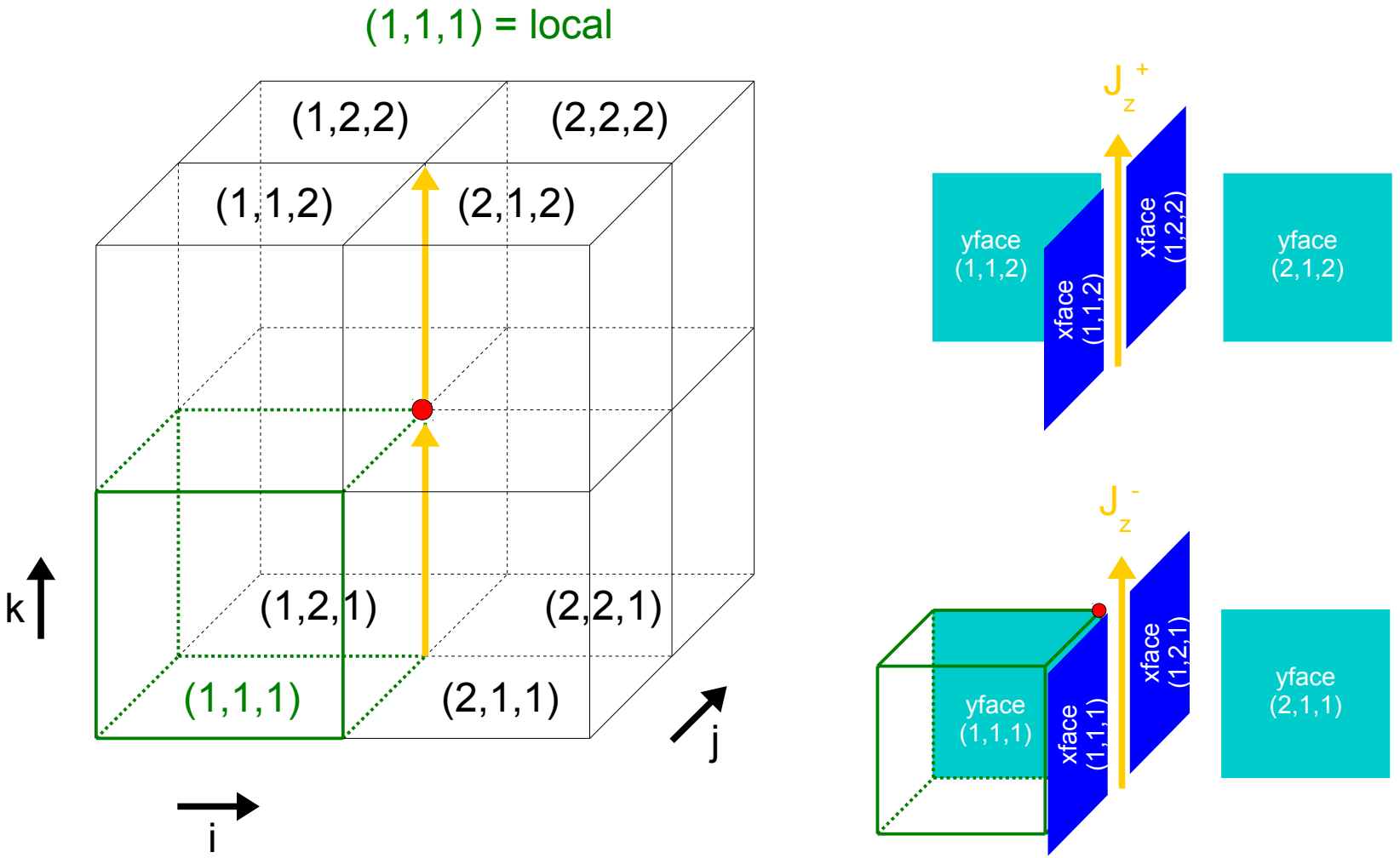


$$edgeJy^{-} = (+faceBz(1,1,1) + faceBx(1,1,2) - faceBz(2,1,1) - faceBx(1,1,1))/(dx \times mu0)$$

$$edgeJy^{+} = (+faceBz(1,2,1) + faceBx(1,2,2) - faceBz(2,2,1) - faceBx(1,2,1))/(dx \times mu0)$$

$$nodeJy = 0.5 \times (edgeJy^{+} + edgeJy^{-})$$

Calculation of Node Jz

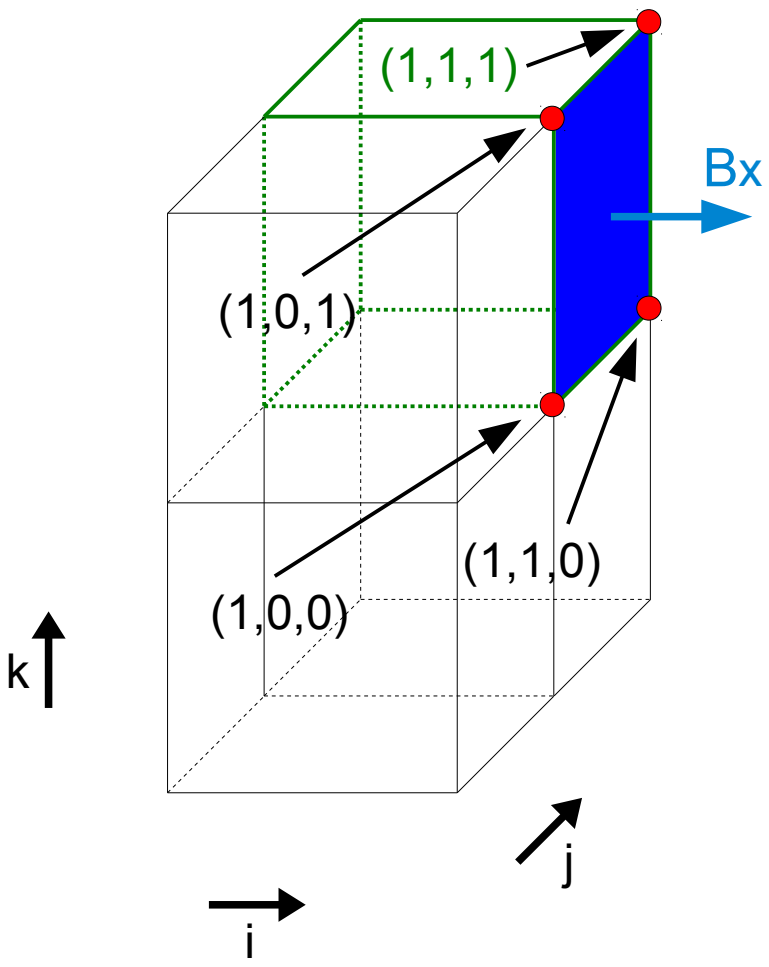


$$edgeJz^{-} = (-faceBy(1,1,1) + faceBx(1,1,1) + faceBy(2,1,1) - faceBx(1,2,1))/(dx \times mu0)$$

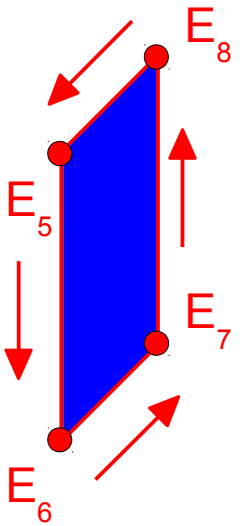
$$edgeJz^{+} = (-faceBy(1,1,2) + faceBx(1,1,2) + faceBy(2,1,2) - faceBx(1,2,2))/(dx \times mu0)$$

$$nodeJz = 0.5 \times (edgeJz^{+} + edgeJz^{-})$$

propagation of B on xface / xface curl



face curl



$E_5 = E(1,0,1)$

$E_6 = E(1,0,0)$

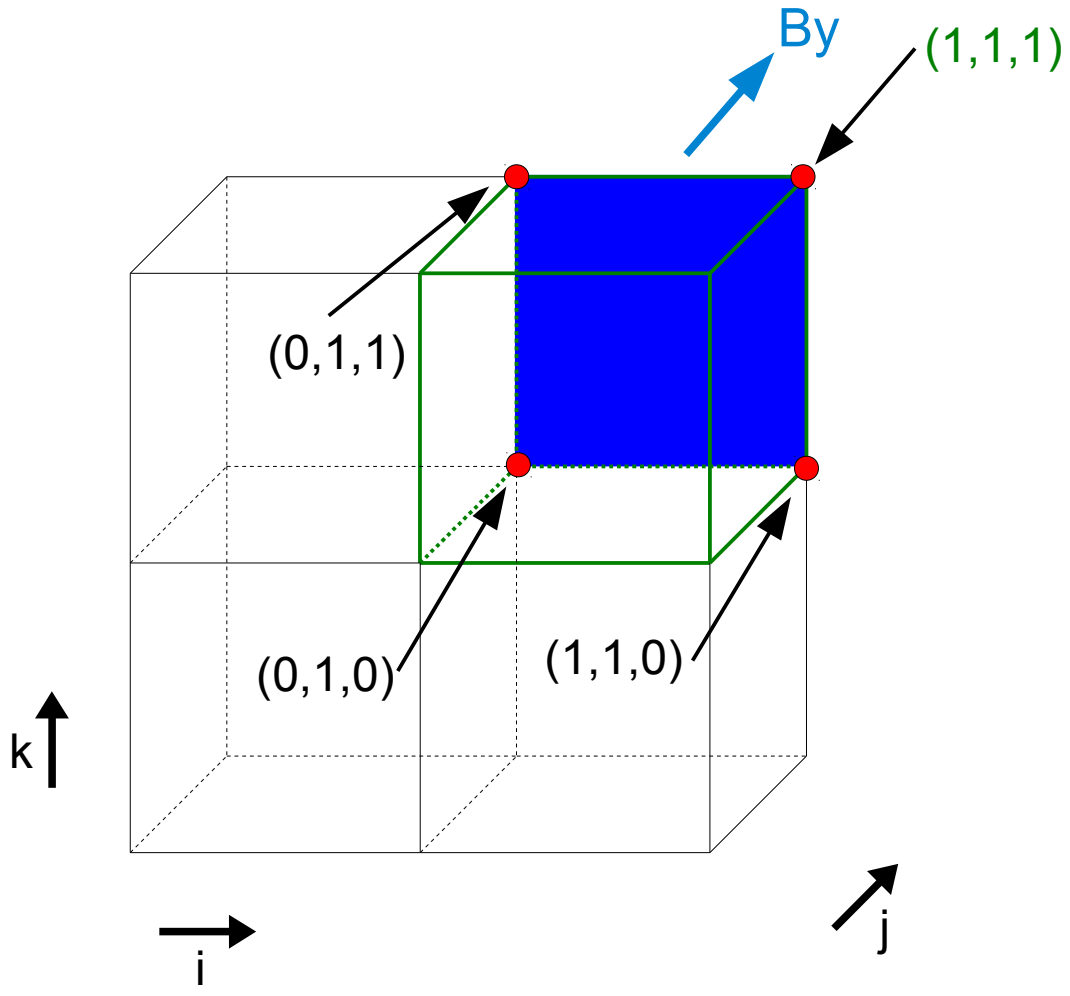
$E_7 = E(1,1,0)$

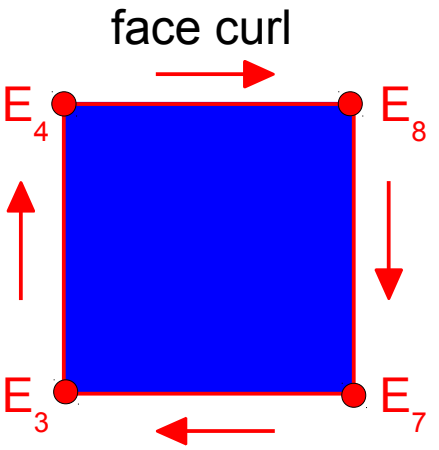
$E_8 = E(1,1,1)$ = local

$$\partial(\int d\mathbf{A}_x \cdot \mathbf{B})/\partial t = -\int d\mathbf{A}_x \cdot (\nabla \times \mathbf{E}) = 0.5 \times dx \times (E_{5z} + E_{6z} - E_{6y} - E_{7y} - E_{7z} - E_{8z} + E_{8y} + E_{5y})$$

$$\mathbf{J}_x = (\nabla \times \mathbf{B})_x / \mu_0 = -0.5 \times dx \times (B_{5z} + B_{6z} - B_{6y} - B_{7y} - B_{7z} - B_{8z} + B_{8y} + B_{5y})$$

propagation of B on yface / yface curl





$E_4 = E(0,1,1)$

$E_8 = E(1,1,1)$ = local

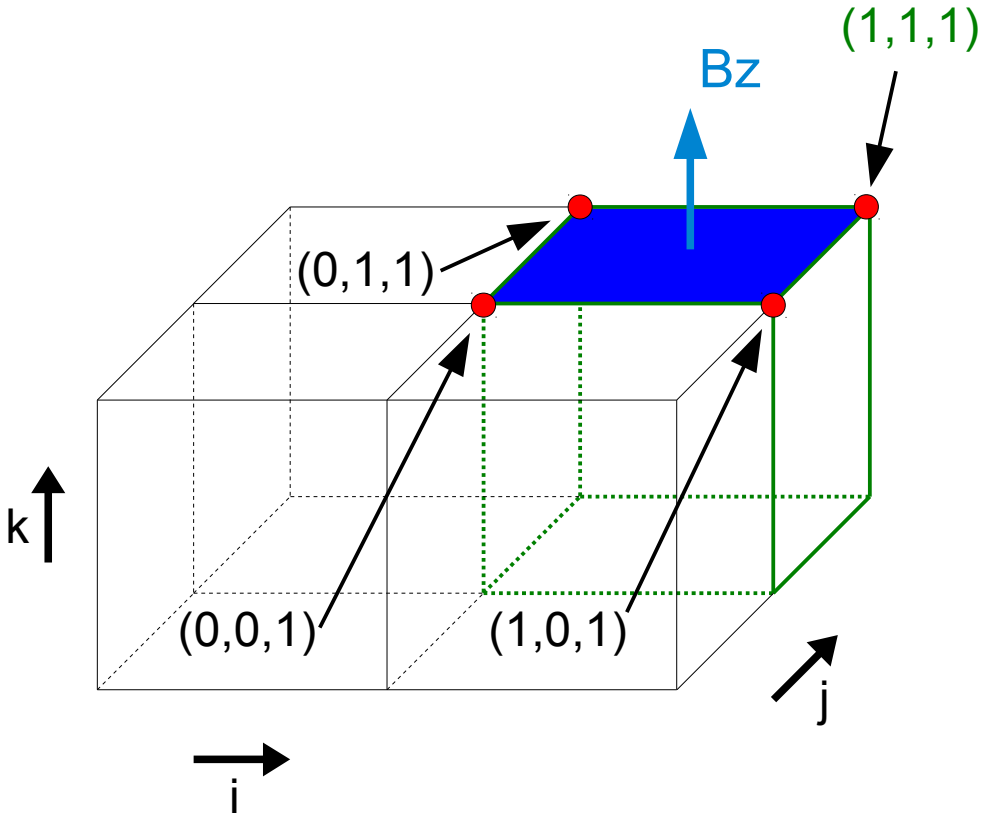
$E_7 = E(1,1,0)$

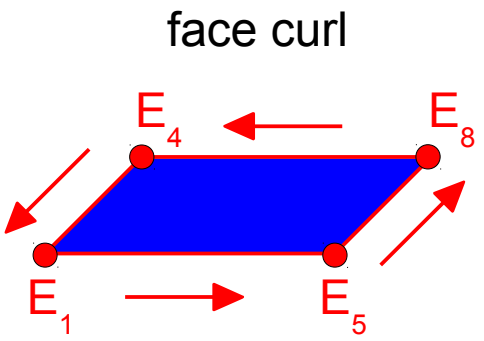
$E_3 = E(0,1,0)$

$$\partial(\int d\mathbf{A}_y \cdot \mathbf{B})/\partial t = -\int d\mathbf{A}_y \cdot (\nabla \times \mathbf{E}) = 0.5 \times dx \times (-E_{4x} - E_{8x} + E_{8z} + E_{7z} + E_{7x} + E_{3x} - E_{3z} - E_{4z})$$

$$\mathbf{J}_y = (\nabla \times \mathbf{B})_y / \mu_0 = -0.5 \times dx \times (-B_{4x} - B_{8x} + B_{8z} + B_{7z} + B_{7x} + B_{3x} - B_{3z} - B_{4z})$$

propagation of B on zface / zface curl





$E_1 = E(0,0,1)$

$E_5 = E(1,0,1)$

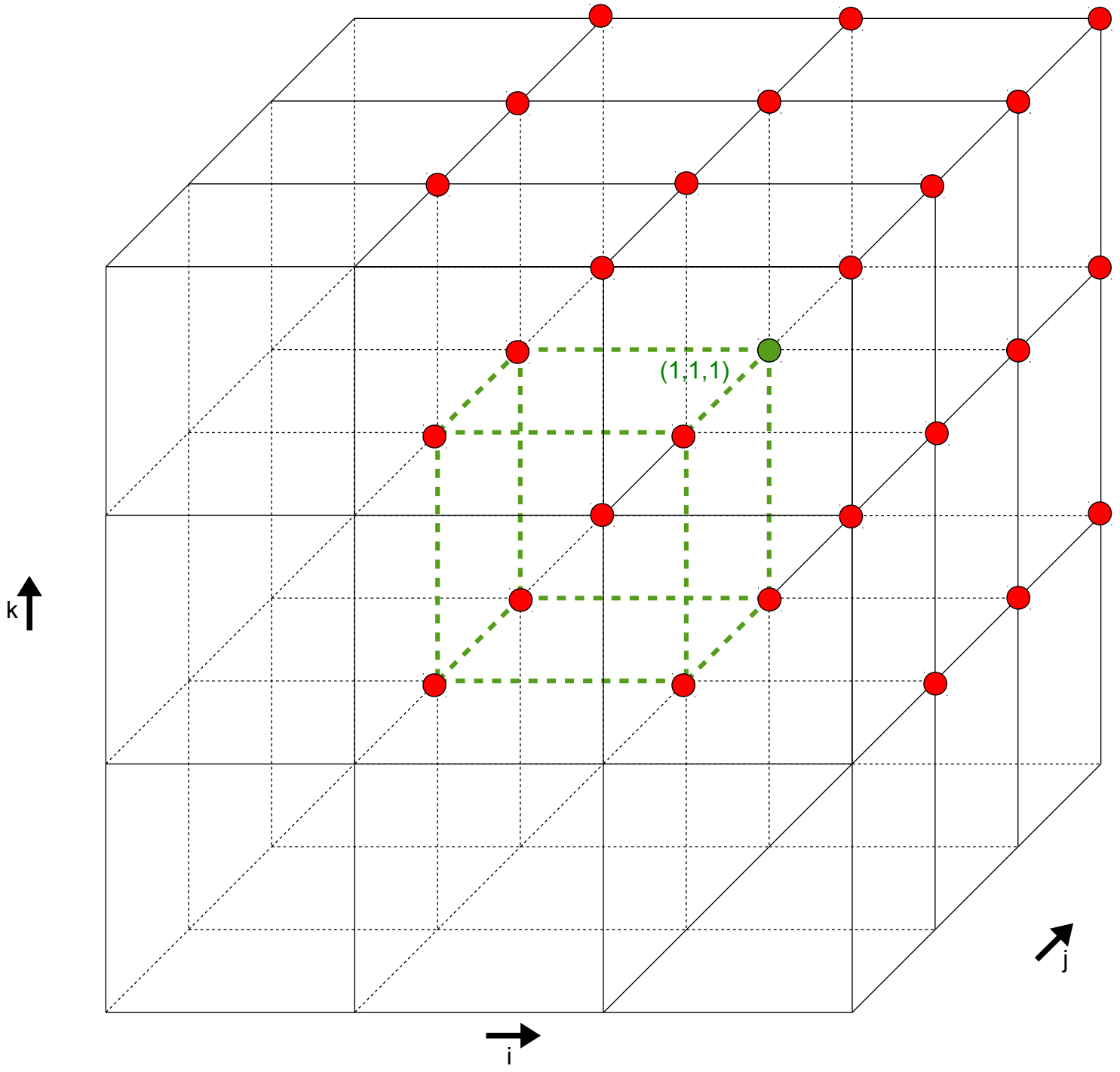
$E_8 = E(1,1,1)$ = local

$E_4 = E(0,1,1)$

$$\partial(\int d\mathbf{A}_z \cdot \mathbf{B})/\partial t = -\int d\mathbf{A}_z \cdot (\nabla \times \mathbf{E}) = 0.5 \times dx \times (-E_{1x} - E_{5x} - E_{5y} - E_{8y} + E_{8x} + E_{4x} + E_{4y} + E_{1y})$$

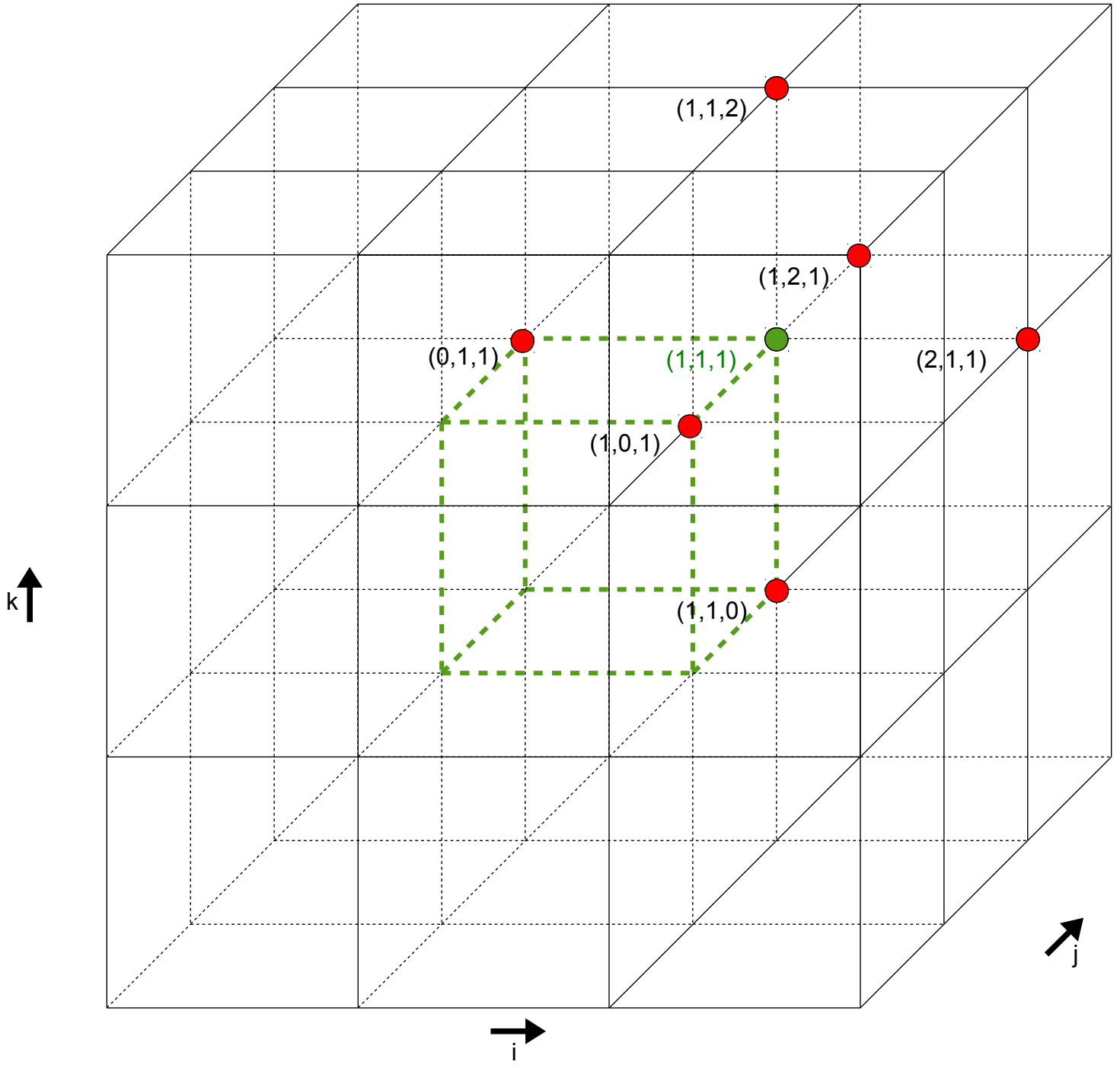
$$\mathbf{J}_z = (\nabla \times \mathbf{B})_z / \mu_0 = -0.5 \times dx \times (-B_{1x} - B_{5x} - B_{5y} - B_{8y} + B_{8x} + B_{4x} + B_{4y} + B_{1y})$$

node2node average



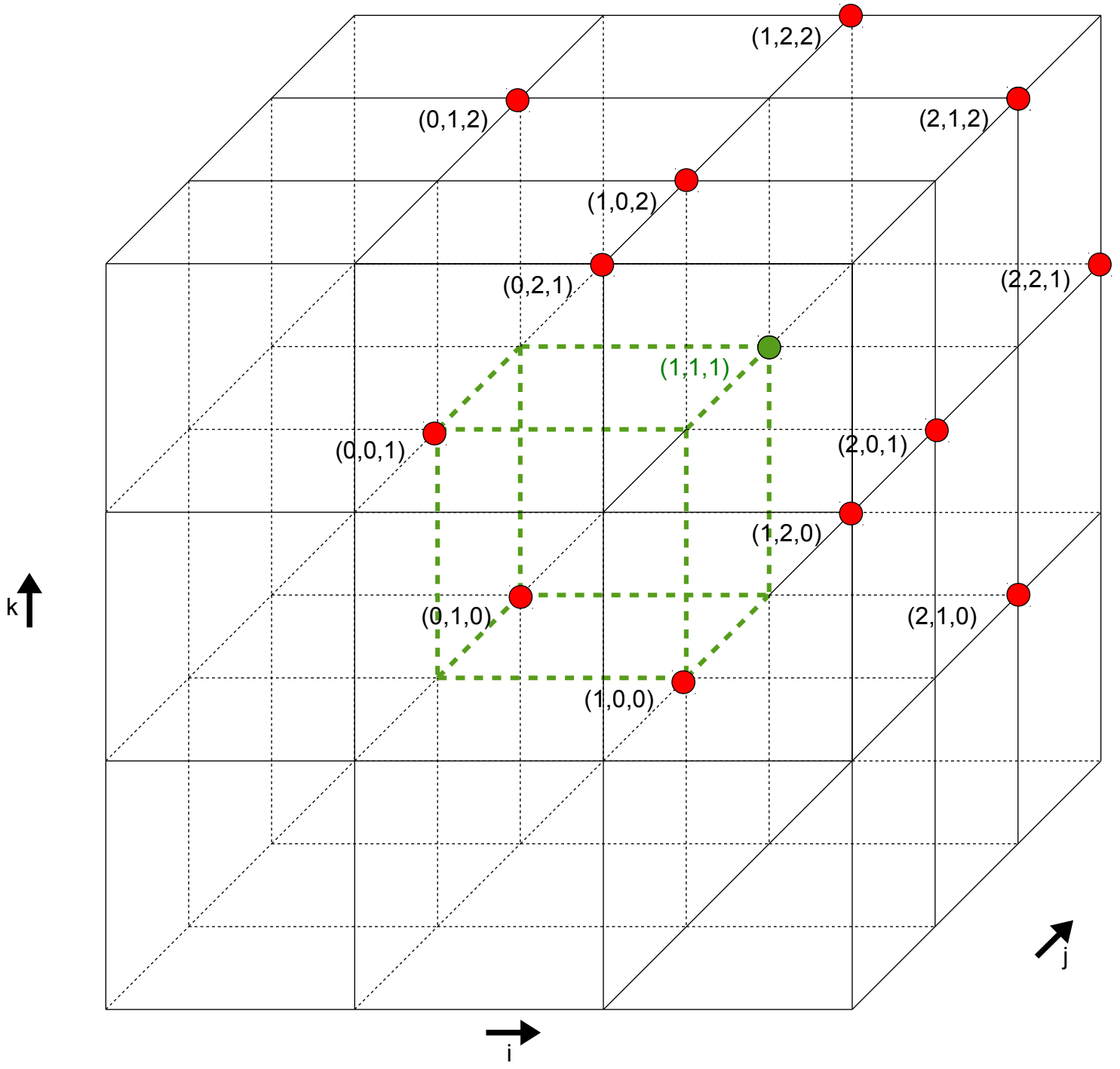

```
nodedata(1,1,1) =
C1*nodedata(1,1,1) +
C2*[ # direct neighbors
nodedata(1,1,0) +
nodedata(0,1,1) +
nodedata(1,2,1) +
nodedata(2,1,1) +
nodedata(1,0,1) +
nodedata(1,1,2)] +
C3*[ # diagonal sqrt(2)
nodedata(0,1,0) +
nodedata(1,0,0) +
nodedata(1,2,0) +
nodedata(2,1,0) +
nodedata(0,0,1) +
nodedata(0,2,1) +
nodedata(2,2,1) +
nodedata(2,0,1) +
nodedata(0,1,2) +
nodedata(1,2,2) +
nodedata(2,1,2) +
nodedata(1,0,2)] +
C4*[ # diagonal sqrt(3)
nodedata(0,0,2) +
nodedata(0,2,2) +
nodedata(2,0,2) +
nodedata(2,2,2) +
nodedata(0,0,0) +
nodedata(0,2,0) +
nodedata(2,0,0) +
nodedata(2,2,0)]
```

node2node average: six direct neighbor nodes



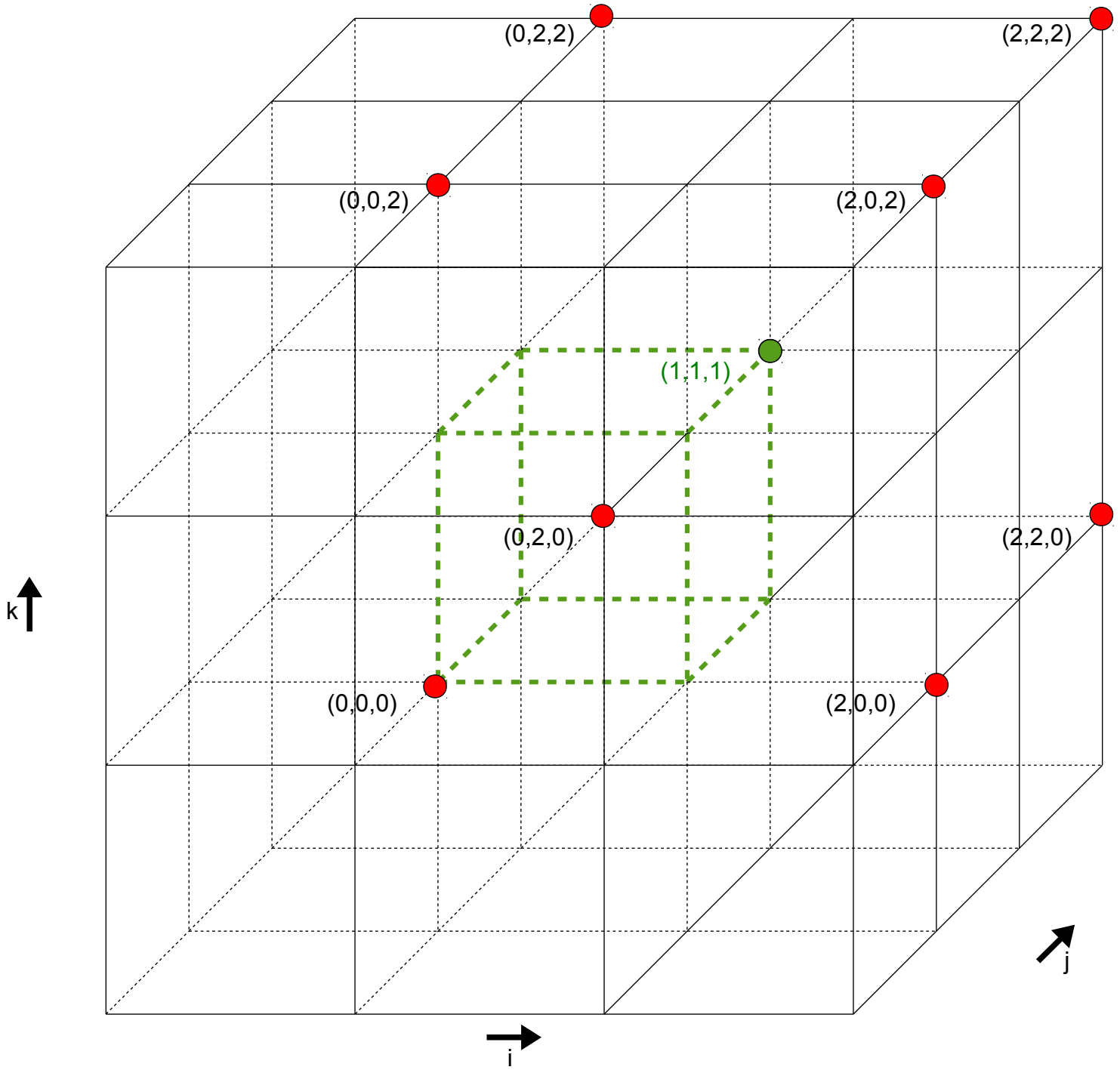

```
nodedata(1,1,1) =
C1*nodedata(1,1,1) +
C2*[ # direct neighbors
nodedata(1,1,0) +
nodedata(0,1,1) +
nodedata(1,2,1) +
nodedata(2,1,1) +
nodedata(1,0,1) +
nodedata(1,1,2)] +
C3*[ # diagonal sqrt(2)
nodedata(0,1,0) +
nodedata(1,0,0) +
nodedata(1,2,0) +
nodedata(2,1,0) +
nodedata(0,0,1) +
nodedata(0,2,1) +
nodedata(2,2,1) +
nodedata(2,0,1) +
nodedata(0,1,2) +
nodedata(1,2,2) +
nodedata(2,1,2) +
nodedata(1,0,2)] +
C4*[ # diagonal sqrt(3)
nodedata(0,0,2) +
nodedata(0,2,2) +
nodedata(2,0,2) +
nodedata(2,2,2) +
nodedata(0,0,0) +
nodedata(0,2,0) +
nodedata(2,0,0) +
nodedata(2,2,0)]
```

node2node average: twelwe sqrt(2) diagonal neighbor nodes




```
nodedata(1,1,1) =
C1*nodedata(1,1,1) +
C2*[ # direct neighbors
nodedata(1,1,0) +
nodedata(0,1,1) +
nodedata(1,2,1) +
nodedata(2,1,1) +
nodedata(1,0,1) +
nodedata(1,1,2)] +
C3*[ # diagonal sqrt(2)
nodedata(0,1,0) +
nodedata(1,0,0) +
nodedata(1,2,0) +
nodedata(2,1,0) +
nodedata(0,0,1) +
nodedata(0,2,1) +
nodedata(2,2,1) +
nodedata(2,0,1) +
nodedata(0,1,2) +
nodedata(1,2,2) +
nodedata(2,1,2) +
nodedata(1,0,2)] +
C4*[ # diagonal sqrt(3)
nodedata(0,0,2) +
nodedata(0,2,2) +
nodedata(2,0,2) +
nodedata(2,2,2) +
nodedata(0,0,0) +
nodedata(0,2,0) +
nodedata(2,0,0) +
nodedata(2,2,0)]
```

node2node average: eight sqrt(3) diagonal neighbor nodes




```
nodedata(1,1,1) =
C1*nodedata(1,1,1) +
C2*[ # direct neighbors
nodedata(1,1,0) +
nodedata(0,1,1) +
nodedata(1,2,1) +
nodedata(2,1,1) +
nodedata(1,0,1) +
nodedata(1,1,2)] +
C3*[ # diagonal sqrt(2)
nodedata(0,1,0) +
nodedata(1,0,0) +
nodedata(1,2,0) +
nodedata(2,1,0) +
nodedata(0,0,1) +
nodedata(0,2,1) +
nodedata(2,2,1) +
nodedata(2,0,1) +
nodedata(0,1,2) +
nodedata(1,2,2) +
nodedata(2,1,2) +
nodedata(1,0,2)] +
C4*[ # diagonal sqrt(3)
nodedata(0,0,2) +
nodedata(0,2,2) +
nodedata(2,0,2) +
nodedata(2,2,2) +
nodedata(0,0,0) +
nodedata(0,2,0) +
nodedata(2,0,0) +
nodedata(2,2,0)]
```

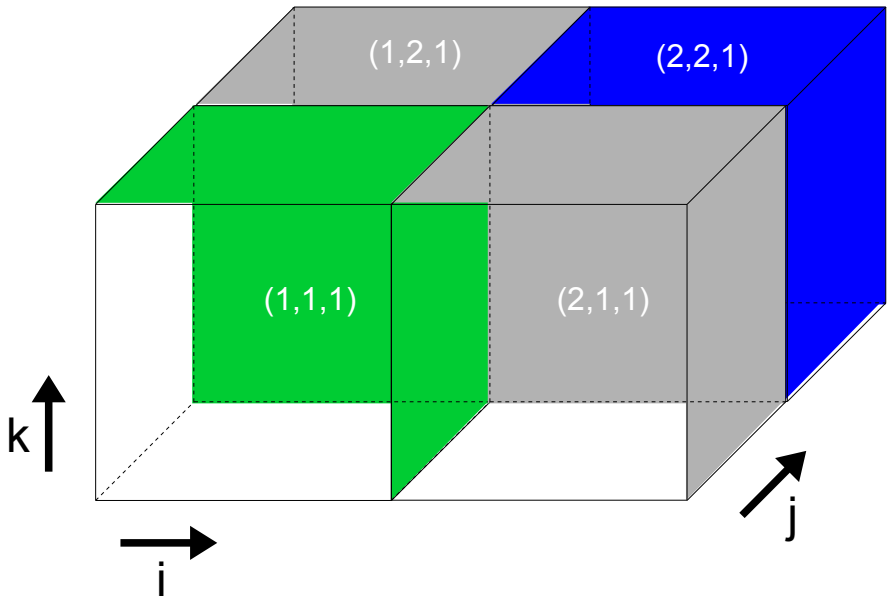
Neumann boundaries for edge faces

(-x,-y) edge

Ghost cells

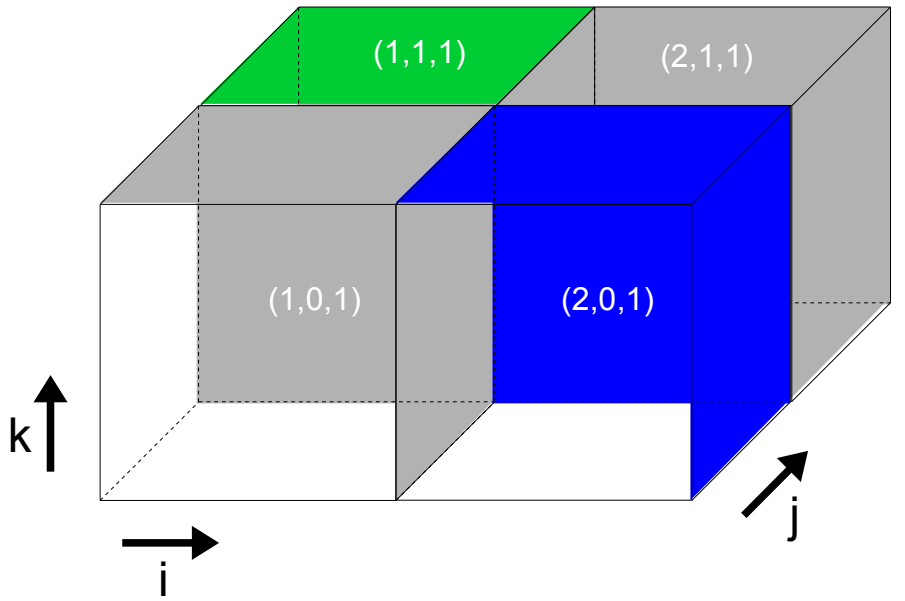
Cells in the simulation domain

Local ghost cell




```
facedata_x(1,1,1) = facedata_x(1,2,1)
facedata_y(1,1,1) = facedata_y(2,1,1)
facedata_z(1,1,1) = facedata_z(2,2,1)
```

(-x,+y) edge




```
facedata_x(1,1,1) = facedata_x(1,0,1)
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = facedata_y(2,0,1)
```

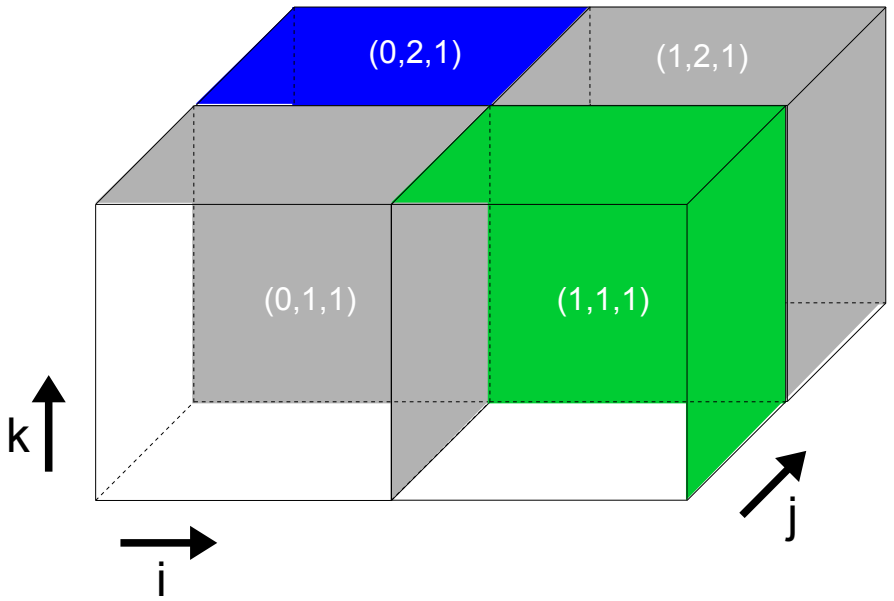
Neumann boundaries for edge faces

(+x,-y) edge

Ghost cells

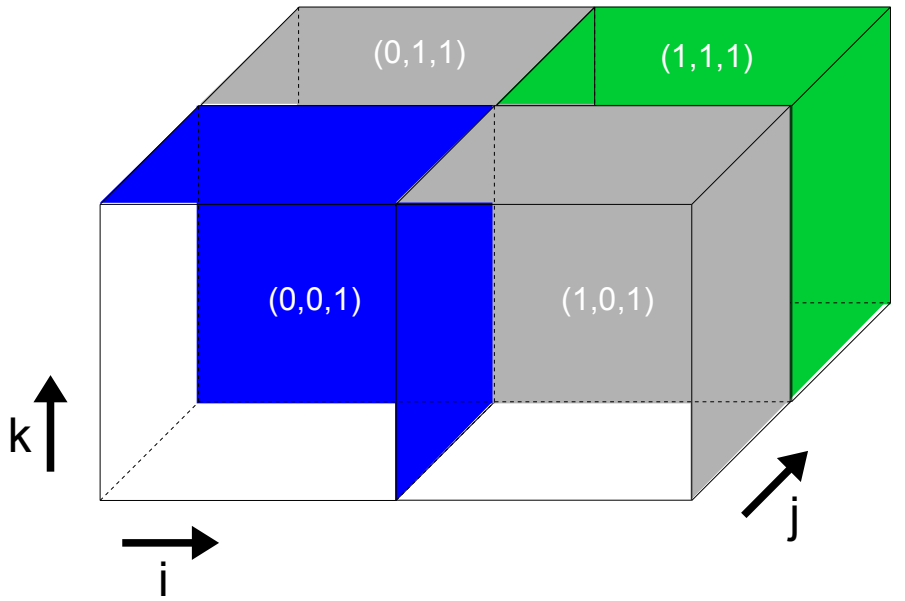
Cells in the simulation domain

Local ghost cell




```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = facedata_y(0,1,1)
facedata_z(1,1,1) = facedata_z(0,2,1)
```

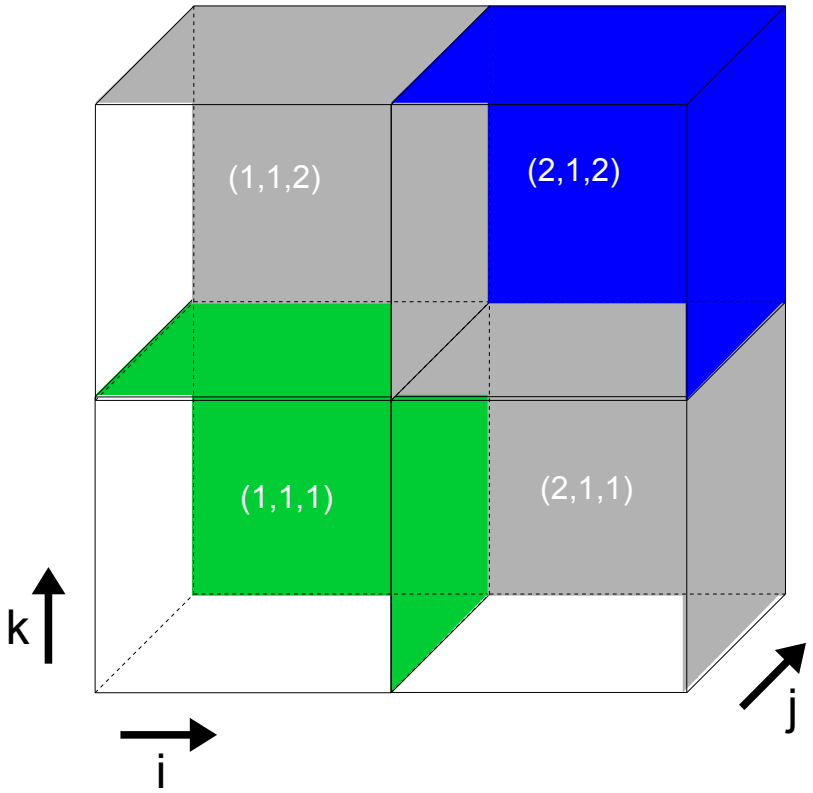
(+x,+y) edge




```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = facedata_z(0,0,1)
```

Neumann boundaries for edge faces

(-x,-z) edge



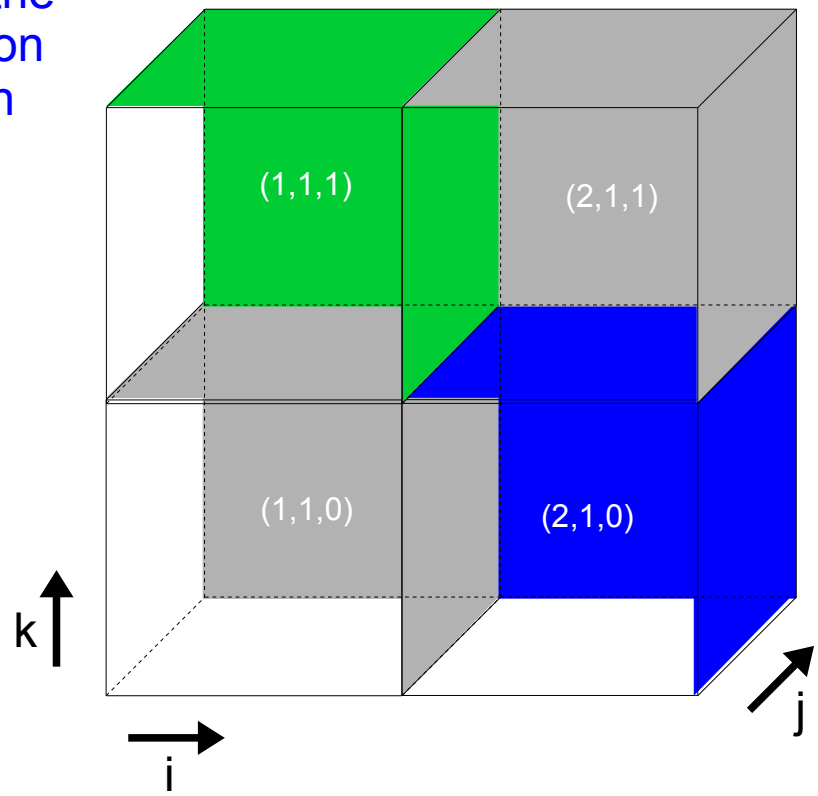

```
facedata_x(1,1,1) = facedata_x(1,1,2)
facedata_y(1,1,1) = facedata_y(2,1,2)
facedata_z(1,1,1) = facedata_z(2,1,1)
```

Ghost cells

Cells in the simulation domain

Local ghost cell

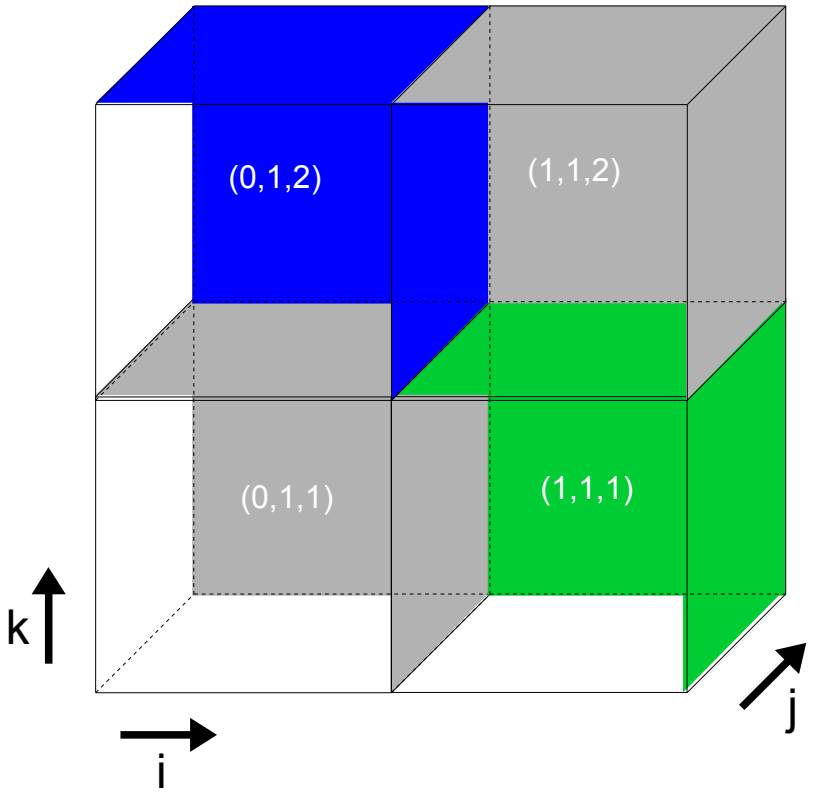
(-x,+z) edge




```
facedata_x(1,1,1) = facedata_x(1,1,0)
facedata_y(1,1,1) = facedata_y(2,1,0)
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for edge faces

(+x,-z) edge



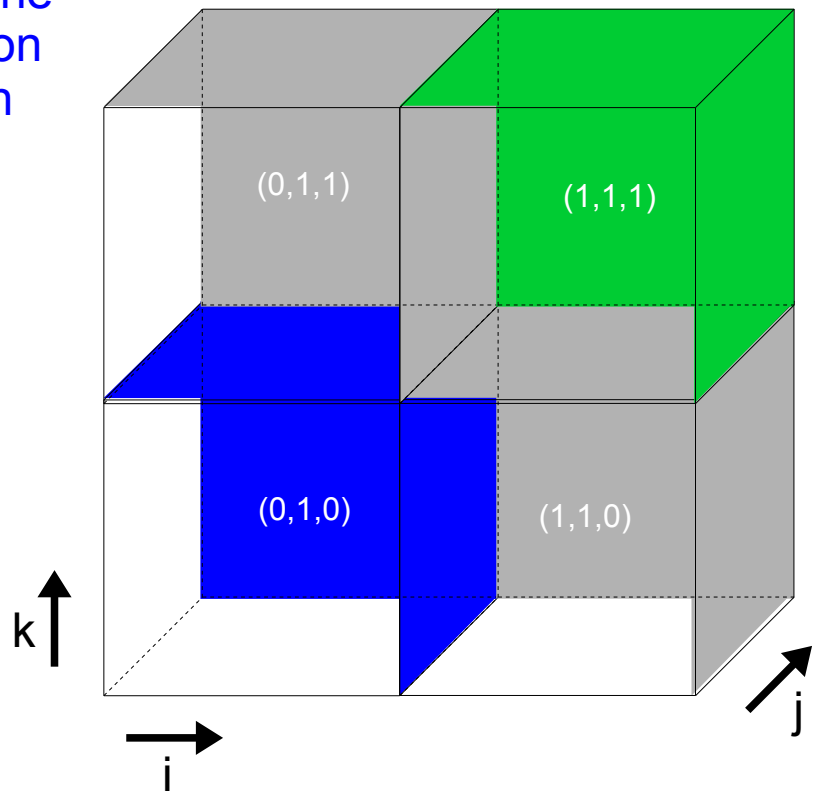

```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = facedata_y(0,1,2)
facedata_z(1,1,1) = facedata_z(0,1,1)
```

Ghost cells

Cells in the simulation domain

Local ghost cell

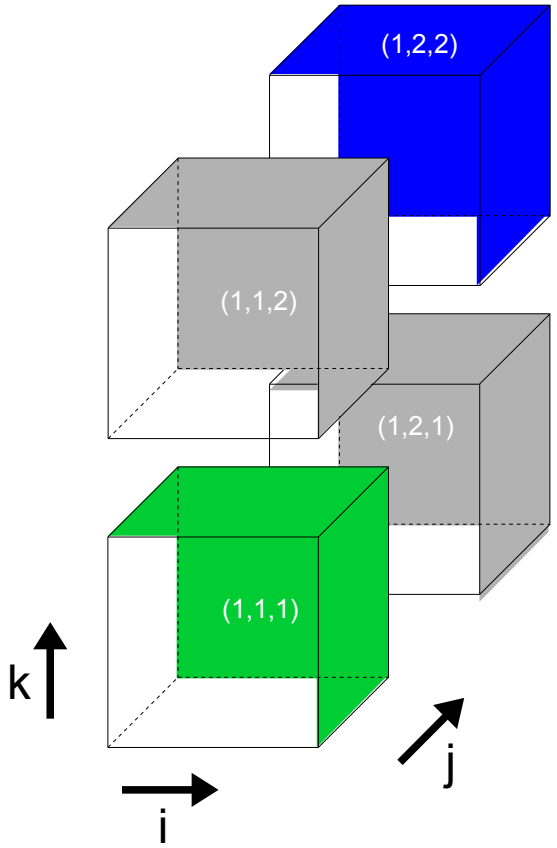
(+x,+z) edge




```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = facedata_y(0,1,0)
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for edge faces

(-y,-z) edge

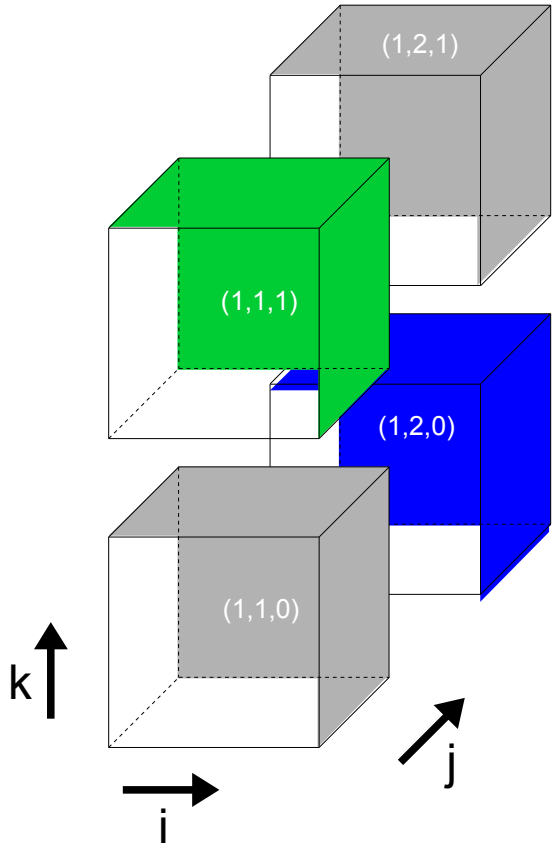


Ghost cells

Cells in the simulation domain

Local ghost cell

(-y,+z) edge

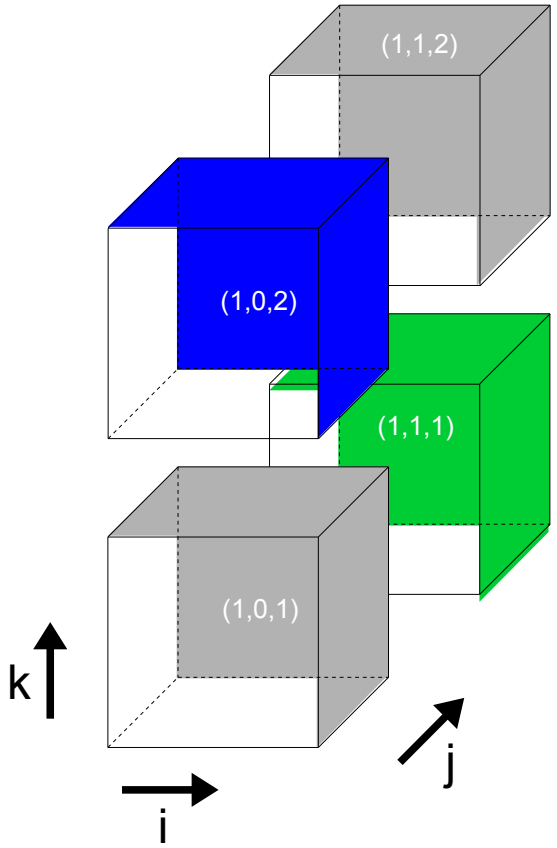



```
facedata_x(1,1,1) = facedata_x(1,2,2)
facedata_y(1,1,1) = facedata_y(1,1,2)
facedata_z(1,1,1) = facedata_z(1,2,1)
```

```
facedata_x(1,1,1) = facedata_x(1,2,0)
facedata_y(1,1,1) = facedata_y(1,1,0)
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for edge faces

(+y,-z) corner

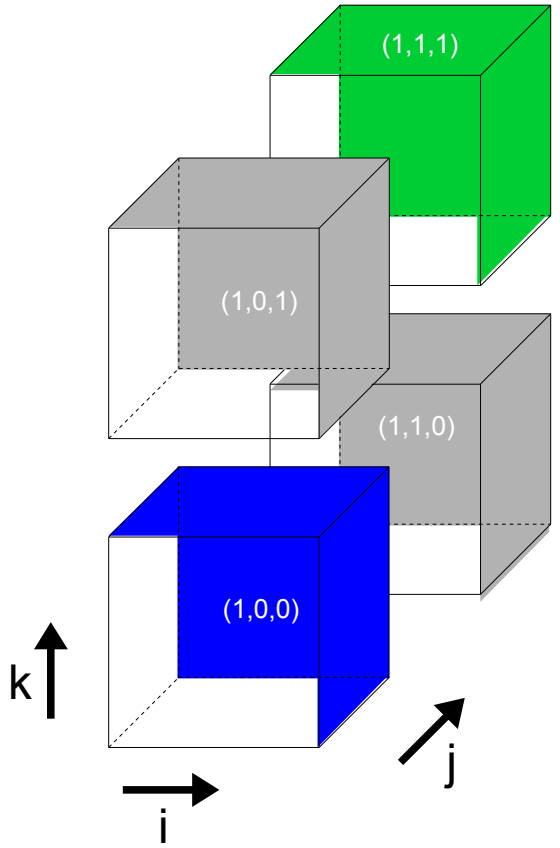


Ghost cells

Cells in the simulation domain

Local ghost cell

(+y,+z) edge

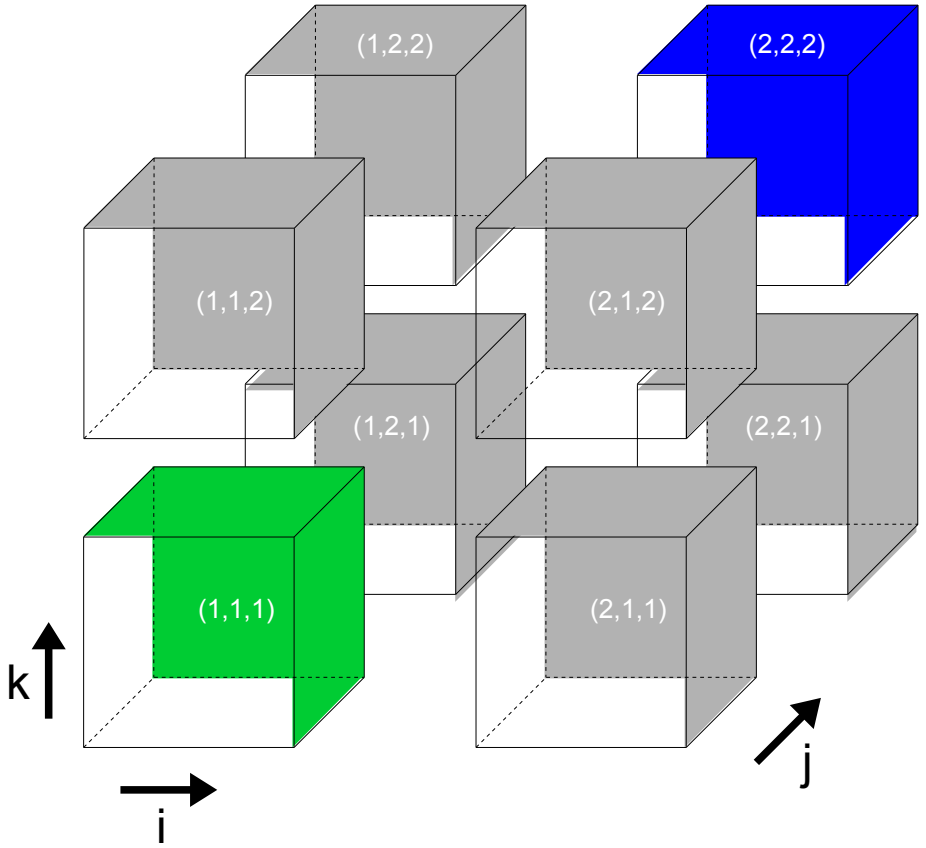



```
facedata_x(1,1,1) = facedata_x(1,0,2)
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = facedata_z(1,0,1)
```

```
facedata_x(1,1,1) = facedata_x(1,0,0)
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for corner faces

(-x,-y,-z) corner

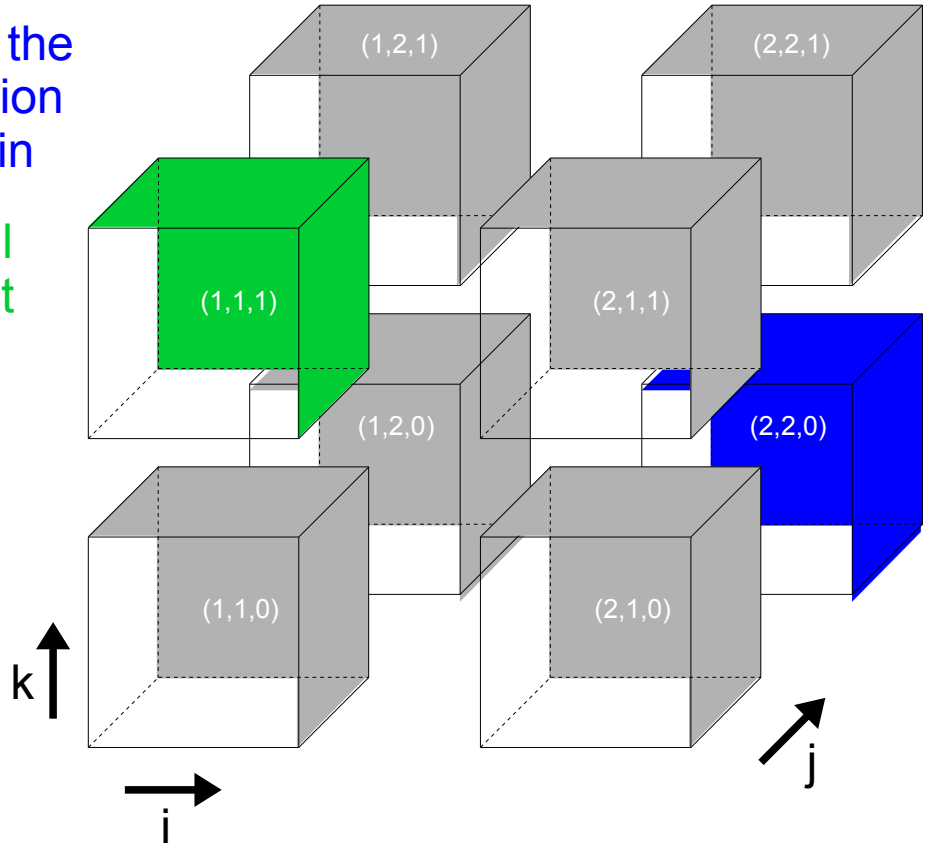


Ghost cells

Cells in the simulation domain

Local ghost cell

(-x,-y,+z) corner

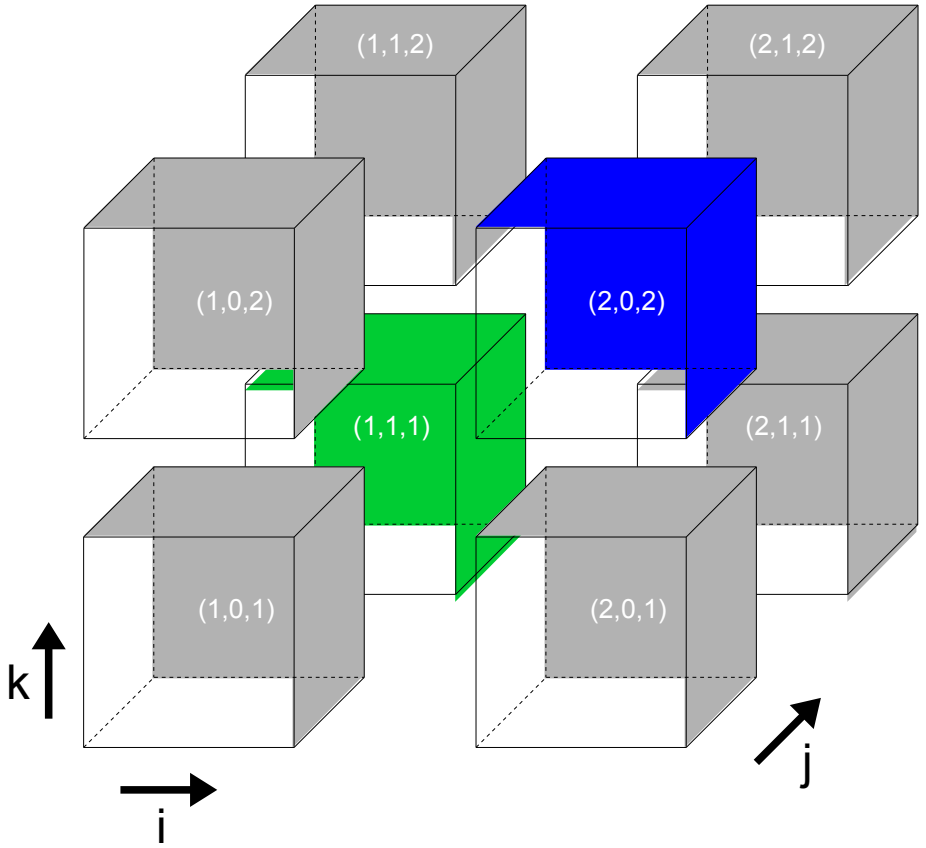



```
facedata_x(1,1,1) = facedata_x(1,2,2)
facedata_y(1,1,1) = facedata_y(2,1,2)
facedata_z(1,1,1) = facedata_z(2,2,1)
```

```
facedata_x(1,1,1) = facedata_x(1,2,0)
facedata_y(1,1,1) = facedata_y(2,1,0)
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for corner faces

(-x,+y,-z) corner

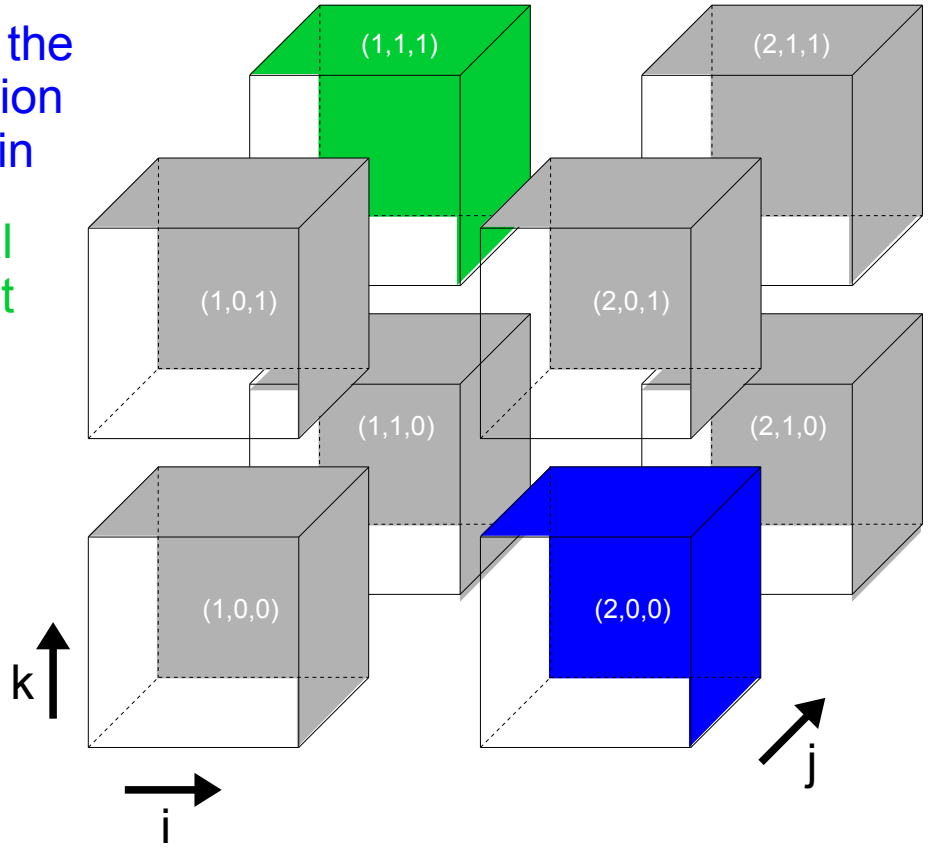


Ghost cells

Cells in the simulation domain

Local ghost cell

(-x,+y,+z) corner

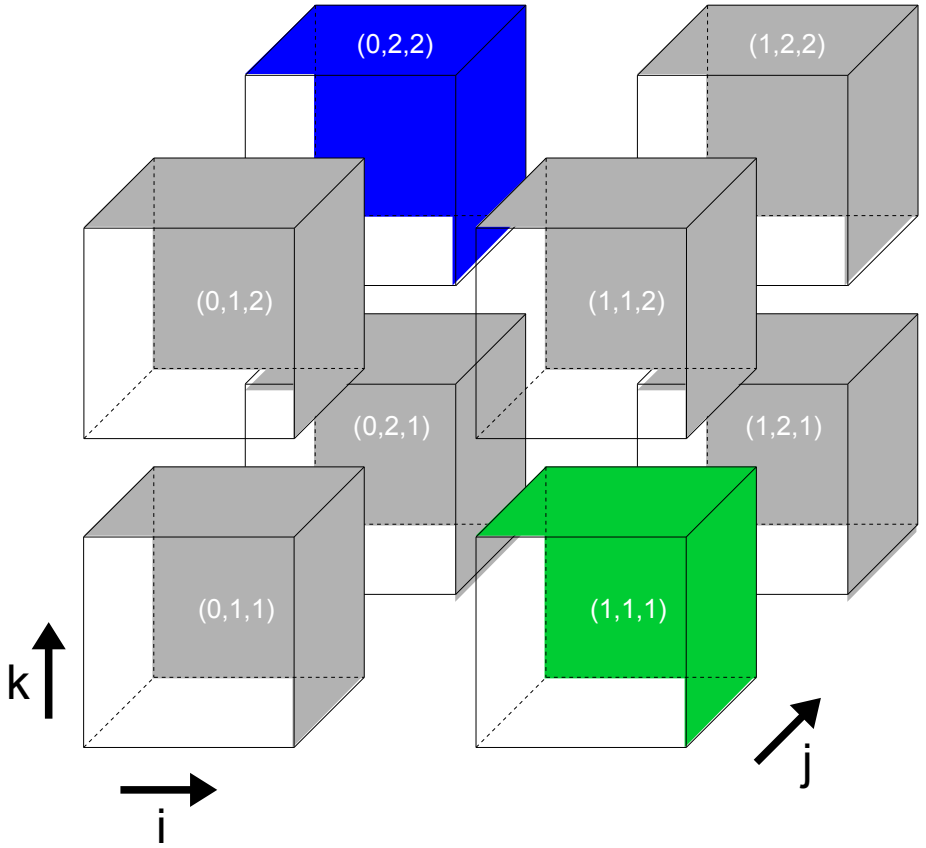



```
facedata_x(1,1,1) = facedata_x(1,0,2)
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = facedata_z(2,0,1)
```

```
facedata_x(1,1,1) = facedata_x(1,0,0)
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for corner faces

(+x,-y,-z) corner

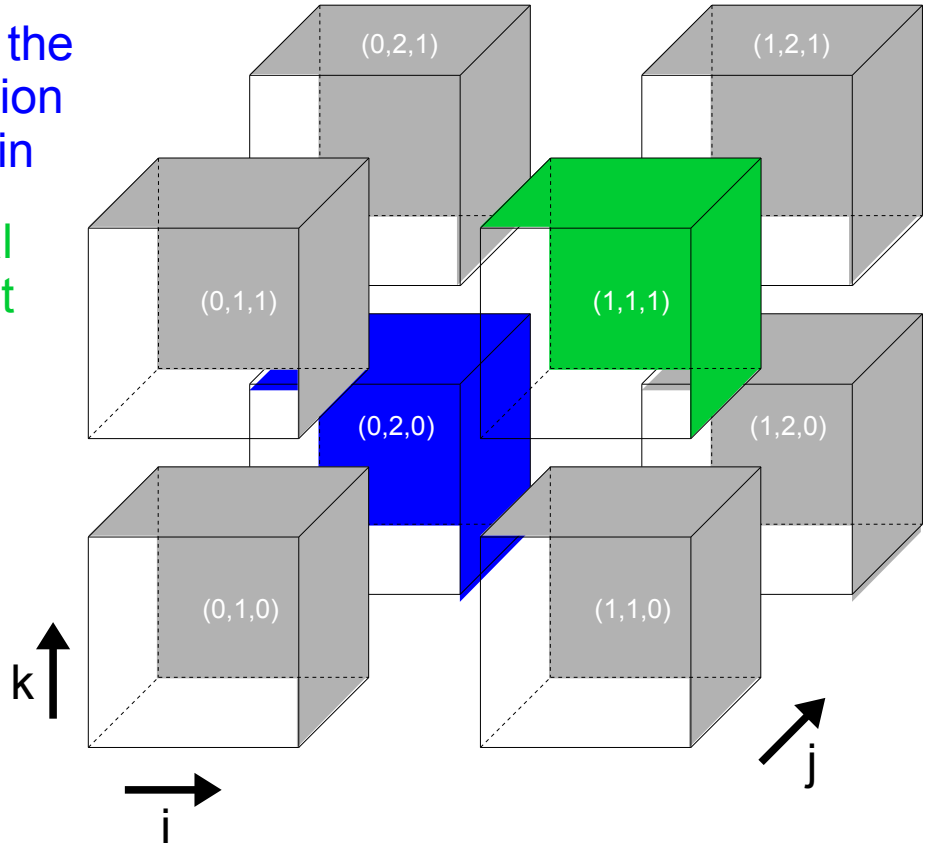


Ghost cells

Cells in the simulation domain

Local ghost cell

(+x,-y,+z) corner

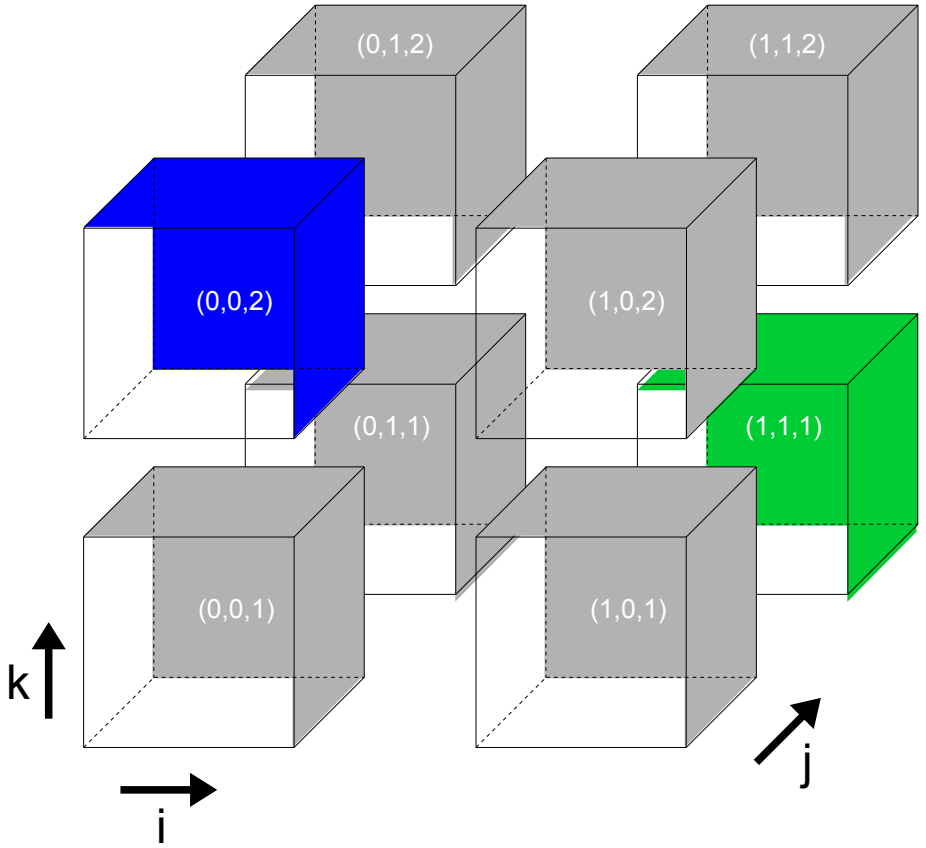



```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = facedata_y(0,1,2)
facedata_z(1,1,1) = facedata_z(0,2,1)
```

```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = facedata_y(0,1,0)
facedata_z(1,1,1) = ghost outer face
```

Neumann boundaries for corner faces

(+x,+y,-z) corner

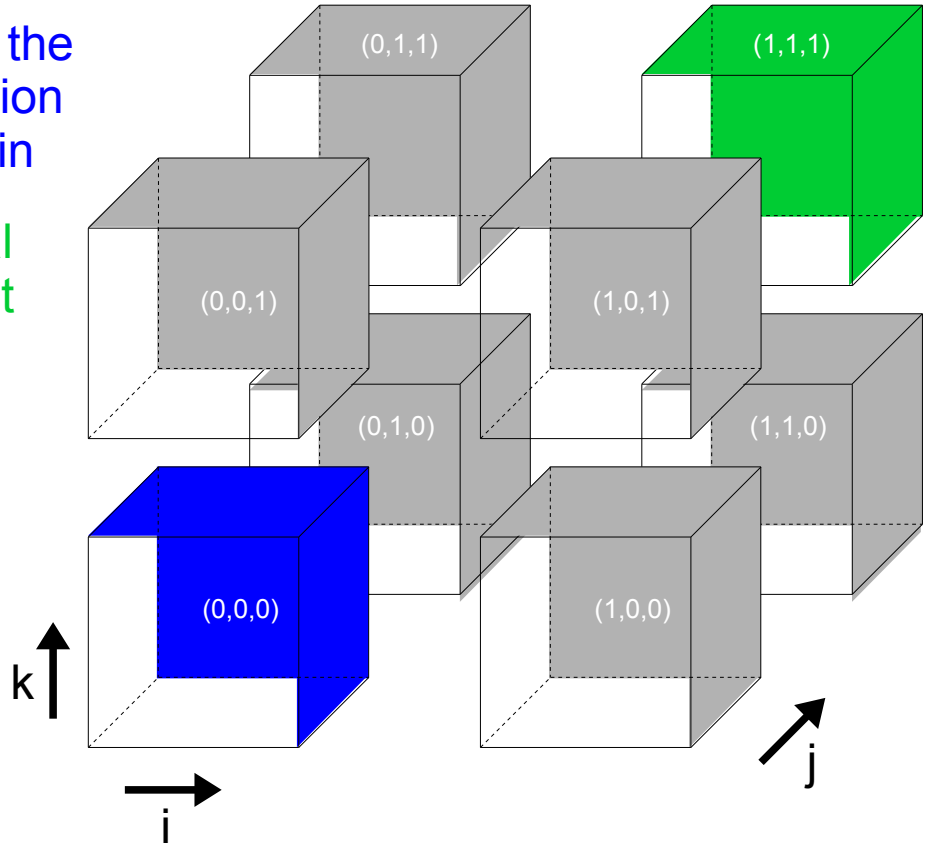


Ghost cells

Cells in the simulation domain

Local ghost cell

(+x,+y,+z) corner




```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = facedata_z(0,0,1)
```

```
facedata_x(1,1,1) = ghost outer face
facedata_y(1,1,1) = ghost outer face
facedata_z(1,1,1) = ghost outer face
```