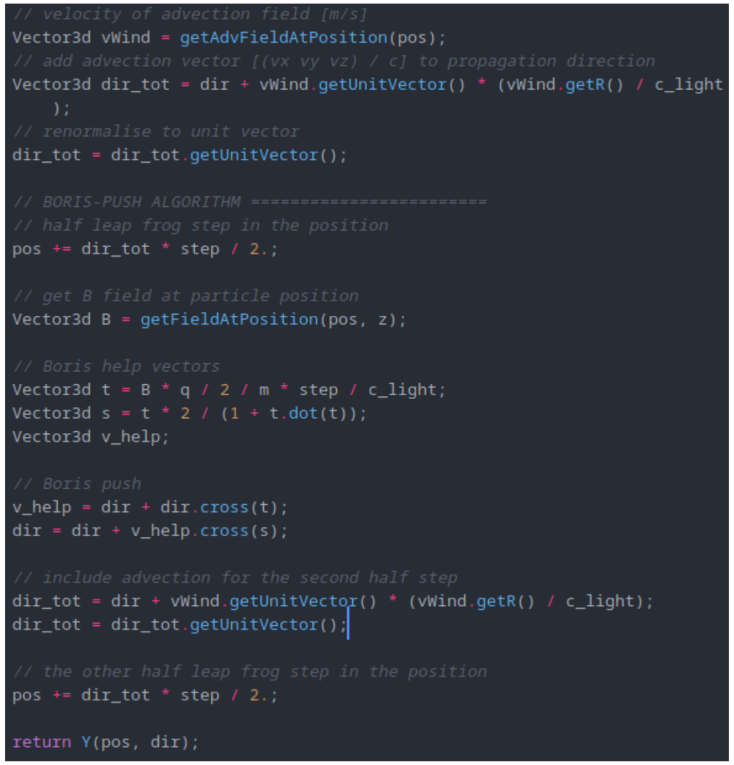
## Vectorised normalisation of advection + diffusion



```
// velocity of advection field [m/s]
Vector3d vWind = getAdvFieldAtPosition(pos);
// add advection vector [(vx vy vz) / c] to propagation direction
Vector3d dir_tot = dir + vWind.getUnitVector() * (vWind.getR() / c_light
    );
// renormalise to unit vector
dir_tot = dir_tot.getUnitVector();

// BORIS-PUSH ALGORITHM ========================
// half leap frog step in the position
pos += dir_tot * step / 2.;

// get B field at particle position
Vector3d B = getFieldAtPosition(pos, z);

// Boris help vectors
Vector3d t = B * q / 2 / m * step / c_light;
Vector3d s = t * 2 / (1 + t.dot(t));
Vector3d v_help;

// Boris push
v_help = dir + dir.cross(t);
dir = dir + v_help.cross(s);

// include advection for the second half step
dir_tot = dir + vWind.getUnitVector() * (vWind.getR() / c_light);
dir_tot = dir_tot.getUnitVector();

// the other half leap frog step in the position
pos += dir_tot * step / 2.;

return Y(pos, dir);
```

effective particle movement direction (adv + diff)

half step before Boris-Push

BP algorithm applied to diffusion direction, i.e. CR direction relative to the plasma

second half step with updated, effective direction

return particle object with new pos and new DIFFUSION direction

Important to use only diff. direction. Otherwise cumulative stacking of influence of advection over multiple steps until velocity completely aligned with advection direction.