
Square Spiral Incremental Layer

Table of Contents

| | |
|------------------|---|
| Parameters | 1 |
| Code | 2 |
| Geometry | 3 |

`X = rectangular_planar_inductor(N,A,L,A0,L0,d,h,x0,y0,z0,phix,phiy,phiz,view)`

Author: JCCopyrights Summer 2019 This function generates a planar rectangular multilayer spiral - PCB Inductor. The coil will have enough layers to accommodate all N turns. The first layer will be generated with center in (0,0,0) in XY plane. The layers will be generated below ($z < 0$) the first layer. It can be moved using the `x0,...,phix...` parameters. No more discretization for the segments is added as FastHenry discretizes automatically all the inputs.

Parameters

- @param **N** Number of Turns
- @param **A** Width of the coil
- @param **L** Height of the coil
- @param **A0** Internal width of the coil
- @param **L0** Internal height of the coil
- @param **d** Distance between turns
- @param **h** Distance between layers of the Coil. Can be introduced as a single value (equidistant) or an array

```
%      With different distances between each layer.
%
% * @param  *x0* Center position X
%
% * @param  *y0* Center position Y
%
% * @param  *z0* Center position Z
%
% * @param  *phix* Turn respect X axis
%
% * @param  *phiy* Turn respect Y axis
%
% * @param  *phiz* Turn respect Z axis
%
% * @param  *view* Optional parameter, if true generates figure with
%      geometry
%
% * @retval *X*      Geometry nodes
```

Code

```
function X =
    rectangular_planar_inductor(N,A,L,A0,L0,d,h,x0,y0,z0,phix,phiy,phiz,view)
    Rx=[1,0,0;0,cos(phix),-sin(phix);0,sin(phix),cos(phix)];
    Ry=[cos(phiy),0,sin(phiy);0,1,0;-sin(phiy),0,cos(phiy)];
    Rz=[cos(phiz),-sin(phiz),0;sin(phiz),cos(phiz),0;0,0,1];

    Nremaining=N; Nmax=floor(min(A/2-A0/2,L/2-L0/2)/d);
    i=1; %Calculate turns per Layer
    while Nremaining>0
        if Nremaining>Nmax
            Nlayer(i)=Nmax;
            Nremaining=Nremaining-Nmax;
        else
            Nlayer(i)=Nremaining;
            Nremaining=Nremaining-Nremaining;
        end
        i=i+1;
    end

    if length(h)==1
        hlayer=h/(size(Nlayer,2)-1); %Height of each layer
        hlayer=hlayer.*ones(1,(size(Nlayer,2)));
        zlayer=hlayer.*(0:1:(size(Nlayer,2)-1));
    else
        hlayer=h;
        zlayer(1)=0;
        for i=2:1:(size(Nlayer,2))
            zlayer(i)=sum(hlayer(1:(i-1)));
        end
    end

    X=square_spiral(Nlayer(1),A,L,d,0,0,0,0,0,0,false);
    %TODO: Clean this fucking mess
    for i=2:1:size(Nlayer,2)
        if mod(i,2)==1 %Assures the correct direction of the turns
            Xaux=X(:,size(X,2))+[0;0;-hlayer(i-1)];
            X=[X,Xaux,square_spiral(Nlayer(i),A,L,d,0,0,-
zlayer(i),0,0,0,false)];
        else %Even layers are more complicated
            Xaux=fliplr(square_spiral(Nlayer(i),A,L,d,0,0,-
zlayer(i),pi,0,0,false));
            Xaux(:,1)=[];%pops first data
            Xaux2(:,1)=X(:,size(X,2))+[0;0;-hlayer(i-1)];
            if Nlayer(i)== Nmax
                X=[X,Xaux2,Xaux,Xaux(:,size(Xaux,2))+[ -
d;0;0],Xaux(:,size(Xaux,2))+[-d;L;0 ]];
            else %Connection to the last turn has to be manually made
                Xaux(:,1)=[];
                Xaux3=Xaux2+[0;d*(Nmax-Nlayer(i)+1);0];
                X=[X,Xaux2,Xaux3,Xaux,Xaux(:,size(Xaux,2))+[ -
d;0;0],Xaux(:,size(Xaux,2))+[-d;L;0 ]];
            end
        end
    end
end
```

```

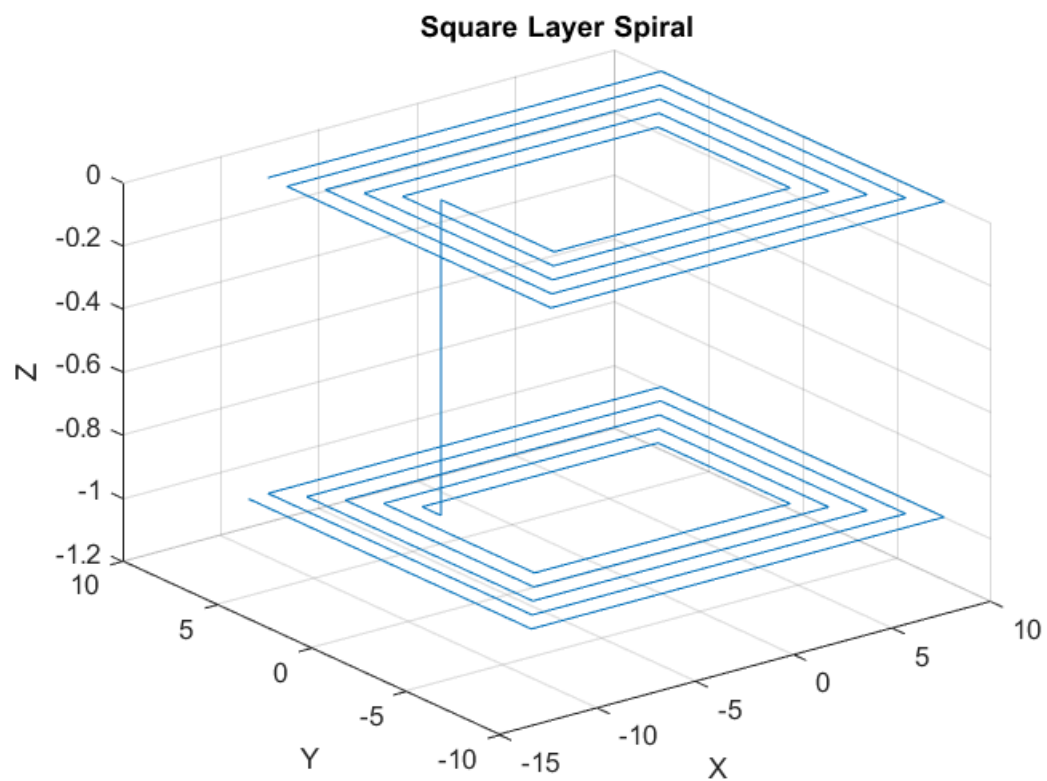
    end
  end
end

for i=1:size(X,2)
  X(:,i)=transpose(Rx*[X(1,i);X(2,i);X(3,i)]);
  X(:,i)=transpose(Ry*[X(1,i);X(2,i);X(3,i)]);
  X(:,i)=transpose(Rz*[X(1,i);X(2,i);X(3,i)]);
  X(:,i)=X(:,i)+[x0;y0;z0];
end

if nargin>13
  if view
    plot3(X(1,:),X(2,:),X(3,:))
    grid on
    xlabel('X')
    ylabel('Y')
    zlabel('Z')
    title('Square Layer Spiral');
  end
end
end

```

Geometry



Published with MATLAB® R2018b