IEEE/CSC & ESAS SUPERCONDUCTIVITY NEWS FORUM (global edition), January 2016.
EUCAS 2015 Invited poster 3A-LS-O1.9.

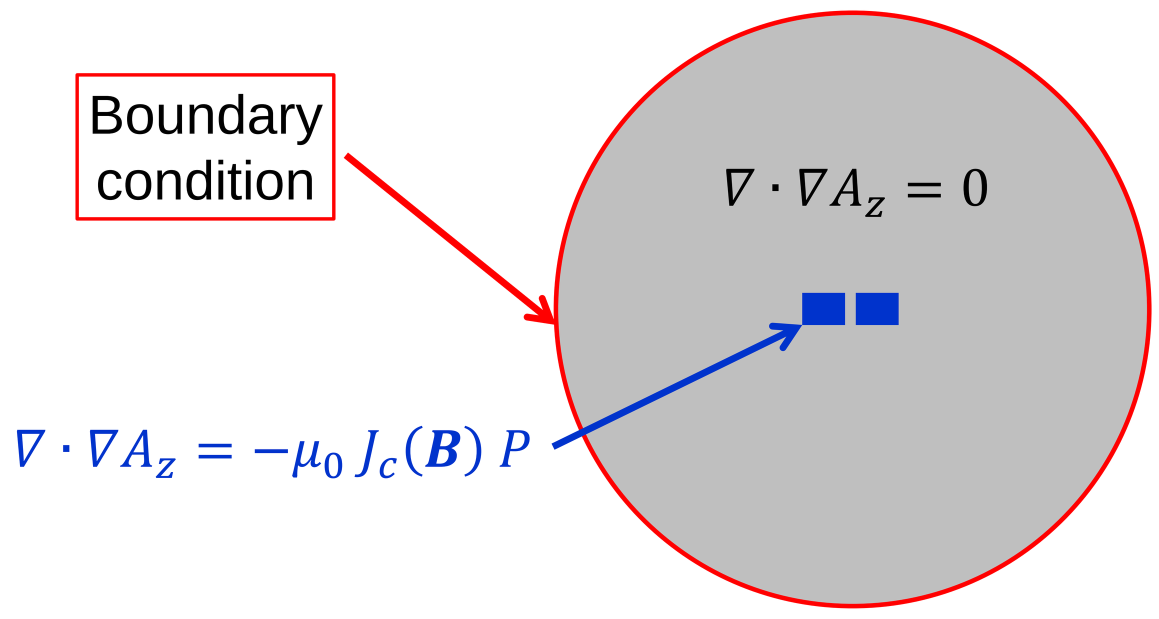
Karlsruher Institut für Technologie

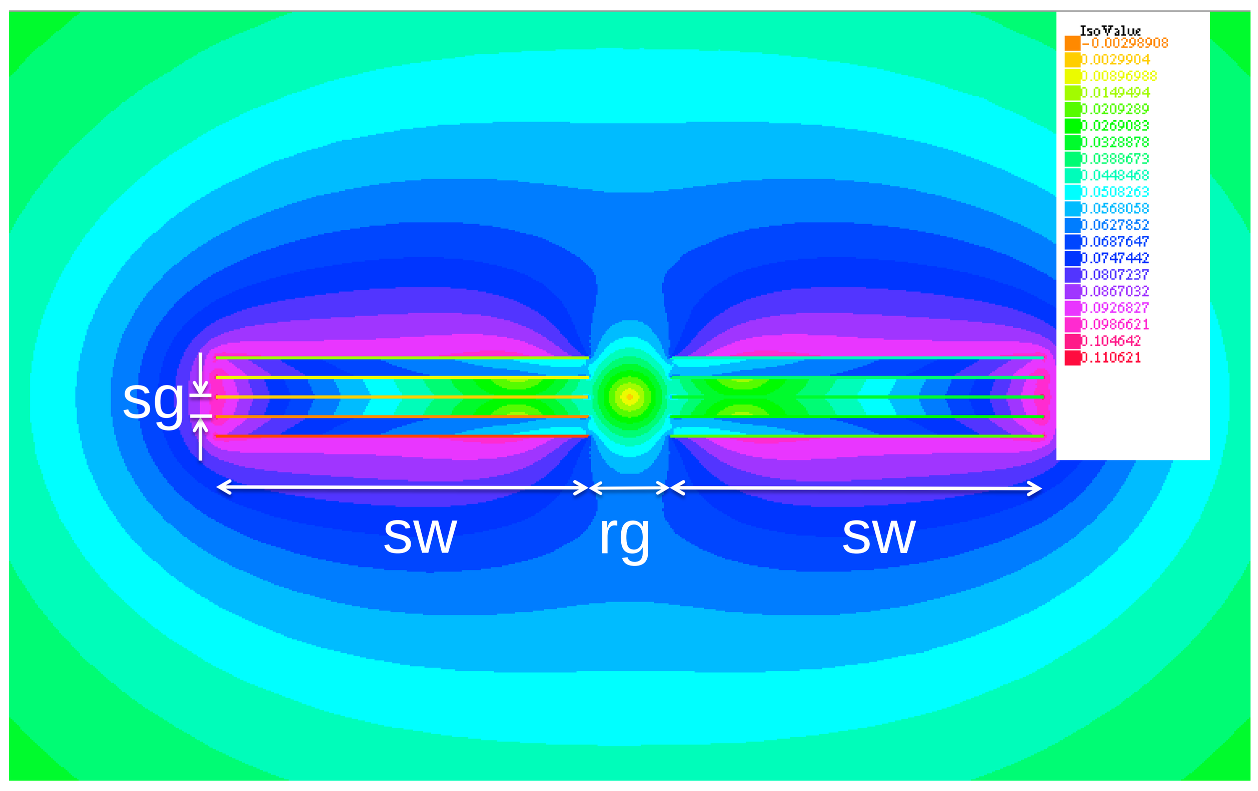
(3A-LS-O1.9)

Open Source Codes for Computing the Critical Current of Superconducting Devices

Víctor M. R. Zermeño, Salman Quaiyum, Francesco Grilli
Karlsruhe Institute of Technology, Karlsruhe, Germany

```
//    A FreeFem++ code to calculate the Ic of superconducting cables     //
// By Victor Zermeno and Salman Quaiyum    doi:10.1109/TASC.2015.XXXXXXX //
//Declaration of parameters and variables for geometry, Physics and mesh //
bool s=0; string c="AVG"; if(s){c="MAX";}; //Ic criteria: s=(1->MAX,0->AVG)
real Jc0=4.75e10, Bc=35e-3, b=0.6, k=0.25;               // Jc(B) parameters
int ns=10, ny=ns/2;                          // ns=number of strands in cable
real th=1e-6, sw=1.8e-3, rg=4e-4, sg=1e-4, n=21, tolAz=1e-9, tolp=1e-9;
real I0=Jc0*th*sw, x0=-sw-rg/2, y0=-((th+sg)*ny-sg)/2, E=0, Ec=1e-4, err;
real[int] XC(ns), YC(ns), Ics(ns), p(ns^2), pn(p.n); p=0.9; pn=0.9;
int[int] cm(1), hm(ns), vm(ns); cm=50; hm=-50; vm=-1;    // Mesh parameters
/////////////////////// Creation of geometry and mesh ///////////////////////
for(int i=0; i<ns; i++){XC[i]=x0+(rg+sw)*(i/ny); YC[i]=y0+(sg+th)*(i%ny);}
border bb    (t=0, 2*pi) {x=20*sw*cos(t);   y=20*sw*sin(t);   label=  1;}
border top   (t=0, 1; i) {x=XC[i]+t*sw;     y=YC[i]+th;       label=i+2;}
border right (t=0, 1; i) {x=XC[i]+sw;       y=YC[i]+(1-t)*th; label=i+2;}
border bottom(t=0, 1; i) {x=XC[i]+(1-t)*sw; y=YC[i];          label=i+2;}
border left  (t=0, 1; i) {x=XC[i];          y=YC[i]+t*th;     label=i+2;}
mesh Th=buildmesh(bb(cm)+top(hm)+right(vm)+bottom(hm)+left(vm));
///////////////////////// Build FEM Solution Space /////////////////////////
fespace Vh(Th,P2);                               // Quadratic elements for Az
Vh Az, Az0, v;
fespace Wh(Th,P1dc);       // Piecewise-linear discontinuous elements for J
Wh J=Jc0;
p[Th(0,0).region]=0;                                  // p=0 in the Air region
///////// JcB and J as functions of Az using the dummy variable u /////////
macro JcB(u) Jc0/(1+sqrt((k*dy(u))^2+(-dx(u))^2)/Bc)^b //
macro J(u) JcB(u)*(p[region])//
//PDE(in weak form) Div(Grad(Az))+mu0*Jc(B)*p=0 and boundary condition Az=0
problem Pmodel(Az,v)=int2d(Th)(dx(Az)*dx(v)+dy(Az)*dy(v))
                  -int2d(Th)(4e-7*pi*J(Az0)*v)+on(1,Az=0);
//////////////////// Solution using an iterative solver ////////////////////
while(abs(p.max-1)*s+abs(pn(0:ns:1).sum/ns-1)*(1-s)>tolp){  // Ic criterion
    err=1;                                           // Reset err variable
    while(err > tolAz){                            // Self consistency loop
        Az0=Az;                                   // Update old Az estimate
        Pmodel;                                          // Run FEM problem
        for(int j=0; j < ns; j++){        // Ic and p value of j-th strand
            Ics(j)=int2d(Th,j)(JcB(Az));
            p(j)=I0/Ics(j);}
        Az0=Az-Az0;          // Difference between old and new Az estimates
        err=Az0[].linfty;}       // Error defined using the L-infinity norm
    for(int i=0; i<ns; i++){pn(i)=p(i)^n;}
    E= pn(0:ns:1).sum/ns*Ec;                // Average electric field in cable
    I0=2*I0/((1+p.max)*s+(1+(E/Ec)^(1/n))*(1-s));}//Net current in strand
/////////////////Post Processing: Output data and plotting/////////////////
cout <<endl<<endl<< "Ic= " << ns*I0 <<" A"<<" ("<< c <<" criteria)"<<endl;
Vh B=sqrt(dx(Az)^2+dy(Az)^2);
plot(B, wait=1, fill=1, value=1);
```

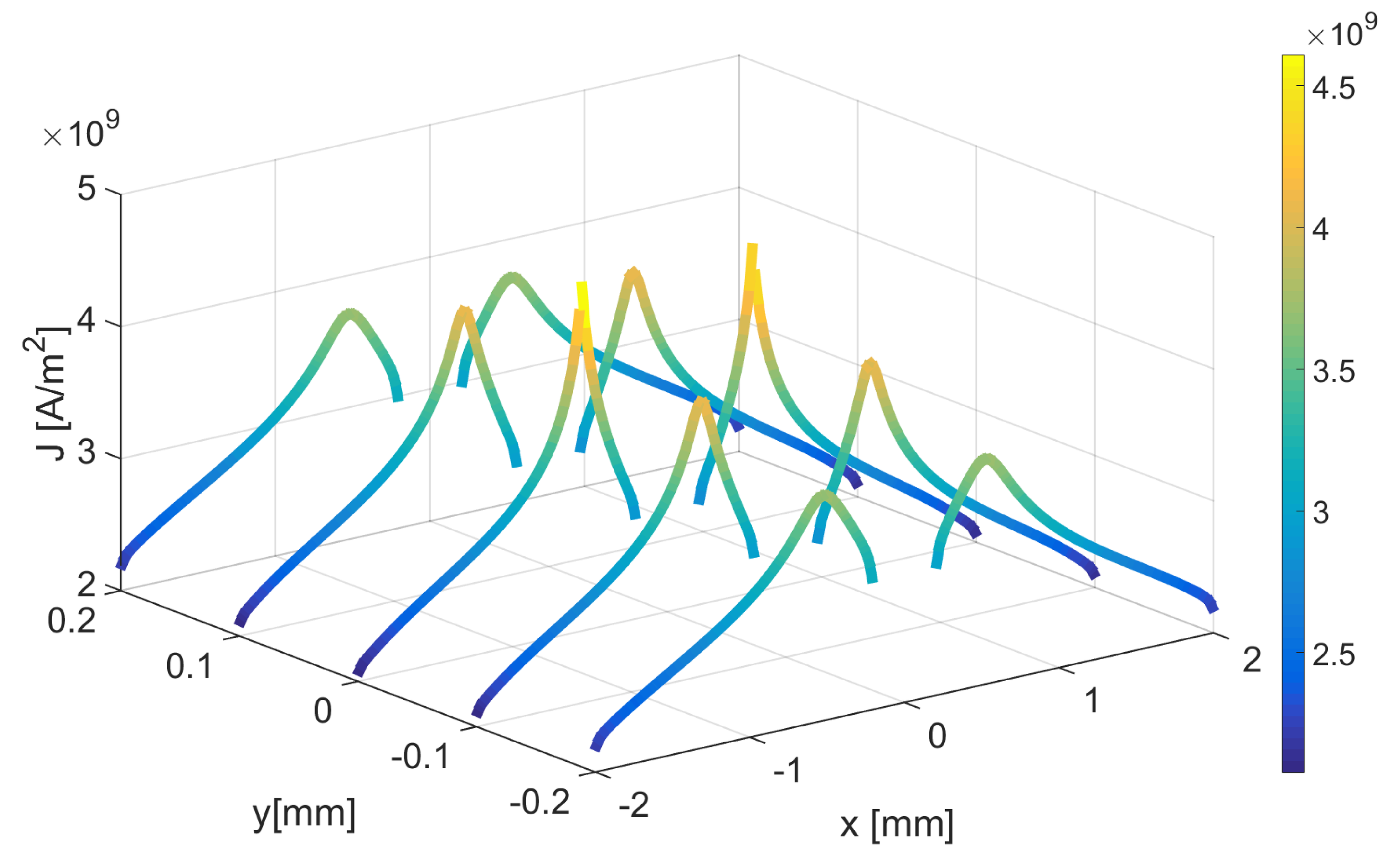




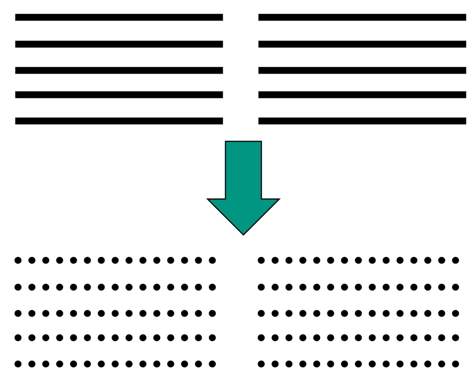
Magnetic flux density in the considered Roebel cable [T]

Comparison of results given by different implementations				
Software	Ic_{MAX} (A)	Ic_{AVG} (A)	ct_{MAX} (s)	ct_{AVG} (s)
Comsol	534.65	538.93	6.00	5.00
FreeFEM++	535.76	537.12	34.30	45.01
MATLAB	535.83	53		
Octave	535.83	539.25	0.13	0.13

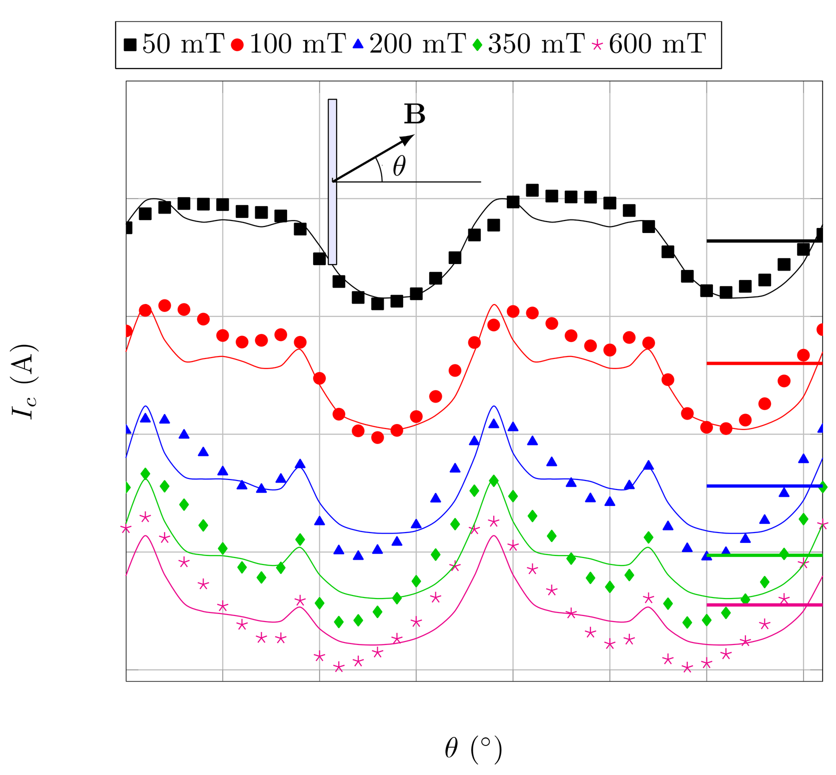
Ic=critical current, ct=computing time (excluding plotting).



$$\boldsymbol{B}_i = \frac{\mu_0}{2\pi} \sum_{j \neq i} P I_c(\boldsymbol{B}_j) \frac{\{-(y_i - y_j), x_i - x_j\}}{(x_i - x_j)^2 + (y_i - y_j)^2}$$



```
%       A MATLAB code to calculate the Ic of superconducting cables       %
%  By Victor Zermeno and Salman Quaiyum     doi:10.1109/TASC.2015.XXXXXXX %
%%      Initialization and declaration of parameters and variables      %%
clc; clear all; close all;
s=0; c='AVG'; if (s==1) c='MAX';end;        %Ic criteria: s=(1->MAX,0->AVG)
Jc0=4.75e10; Bc=35e-3; k=0.25; b=0.6;                     % Jc(B)parameters
m=100; ns=10; th=1e-6; sw=1.8e-3; rg=4e-4; sg=1e-4; n=21;       % parameters
mu0=4e-7*pi; Ec=1e-4; tolIc=1e-9; tolp=1e-9; %mu0, Ec criterion, tolerances
I0=Jc0*sw*th; P=0.5*ones(1,m*ns); E=0;      % Initial values for I0, P and E
[Bx,By,Ic]=deal(zeros(1,m*ns)); %Magnetic flux density and critical current
%% Geometry creation: lines of current are located at the points (Rx,Ry) %%
xRange=(1-m:2:m-1)*sw/2/m;                  % Span of values for x coordinate
Rx=[repmat(xRange-(rg+sw)/2,[1 ns/2]),repmat(xRange+(rg+sw)/2,[1 ns/2])];
yRange=((2-ns):4:(ns-2))*sg/4;              % Span of values for y coordinate
Ry=[reshape(repmat(yRange,m,1),1,[]),reshape(repmat(yRange,m,1),1,[])];
%%          Definition of auxiliary variables for field calculation          %%
r2=bsxfun(@minus,Rx,Rx').^2+bsxfun(@minus,Ry,Ry').^2;% r2=(x-x')^2+(y-y')^2
xn=bsxfun(@minus,Rx,Rx')./r2; xn(isnan(xn))=0;      % if(r2>0,(x-x')/r2,0)
yn=bsxfun(@minus,Ry,Ry')./r2; yn(isnan(yn))=0;      % if(r2>0,(y-y')/r2,0)
%%                           Iterative solution                           %%
while (abs(max(P)-1)*s+abs(E/Ec-1)*(1-s)>tolp)          % Ic criteria loop
err = 1;                                                     % Error reset
    while(err > tolIc)                               % Self consistency loop
        IcOld=Ic;
        Ic=(sw*th/m)*(Jc0./(1+sqrt((k*Bx).^2+By.^2)./Bc).^b);  %local Ic(B)
        P=reshape((I0./(reshape(Ic,m,ns)'*ones(m,m)))',1,m*ns);   %P values
        It=P.*Ic;                                   % Local current in strand
        Bx=-mu0/(2*pi)*It*yn;                        % Magnetic flux density
        By=mu0/(2*pi)*It*xn;
        err=norm(IcOld - Ic); % Error. It compares old and new Ic estimates
    end
    E=Ec*abs(sum(P(1:m:end).^n)/ns);    % Average electric field in cable
    I0=2*I0/(s*(1+max(P))+(1+(E/Ec)^(1/n))*(1-s)); %Net current in strand
end
%%                Post Processing: Output data and plotting                %%
fprintf('Ic(cable)= %0.2f A (%s criteria)\n',ns*I0,c);
fprintf('Avg(E)= %0.6f microV/cm\n',abs(sum(P(1:100:end).^n)/ns));
fprintf('max(P)= %0.6f\n',max(P));
X=reshape(Rx,ns,[])/1e-3;
Y=reshape(Ry,ns,[])/1e-3;
Z=(ns/sw/th)*reshape(It,ns,[]);
mesh(X,Y,Z,'MeshStyle','column','FaceColor','none','LineWidth',3); % J(x,y)
xlabel('x [mm]'); ylabel('y[mm]'); zlabel('J [A/m^2]');
colorbar;
```



Field dependence of I_c for a 12 mm tape.

Application:

Effect of the angular dependence of *Jc(**B**)* in calculating the I_c of a 10-strand Roebel cable.

I_c of a 12 mm-wide Roebel		
	Ic_{MAX} (A)	Ic_{AVG} (A)
*Precise Jc(**B**)*	1005	1035
*Simplified Jc(|**B**|)*	1045	1067

This and more codes available at

HTS MODELLING WORKGROUP

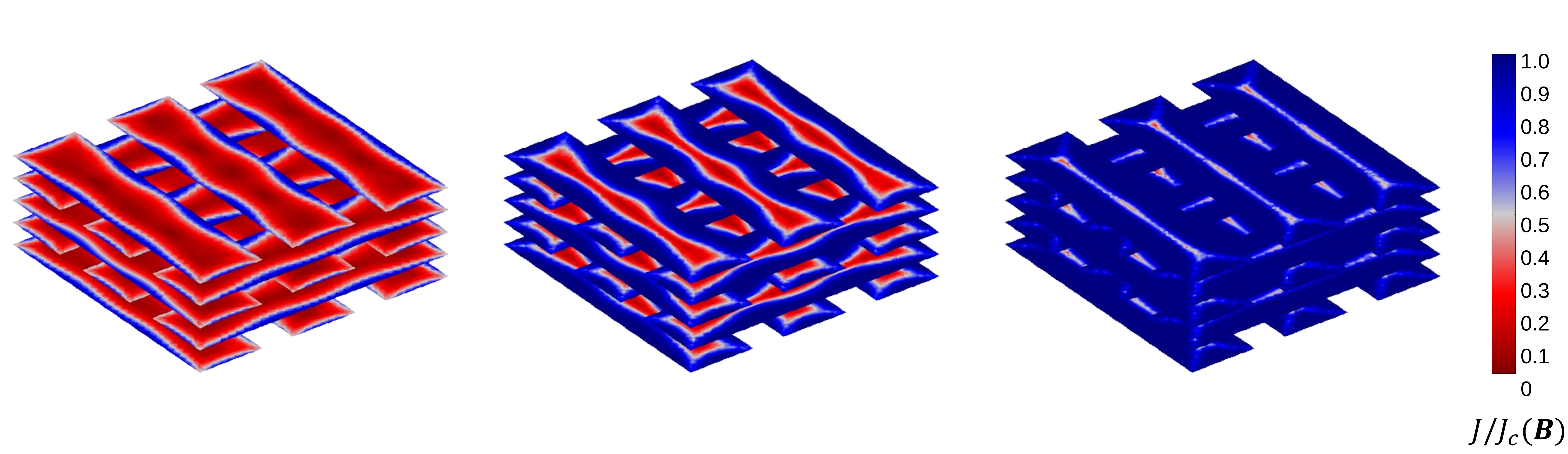
IEEE/CSC & ESAS SUPERCONDUCTIVITY NEWS FORUM (global edition), January 2016.
EUCAS 2015 Invited poster 3A-LS-O1.9.

Karlsruher Institut für Technologie

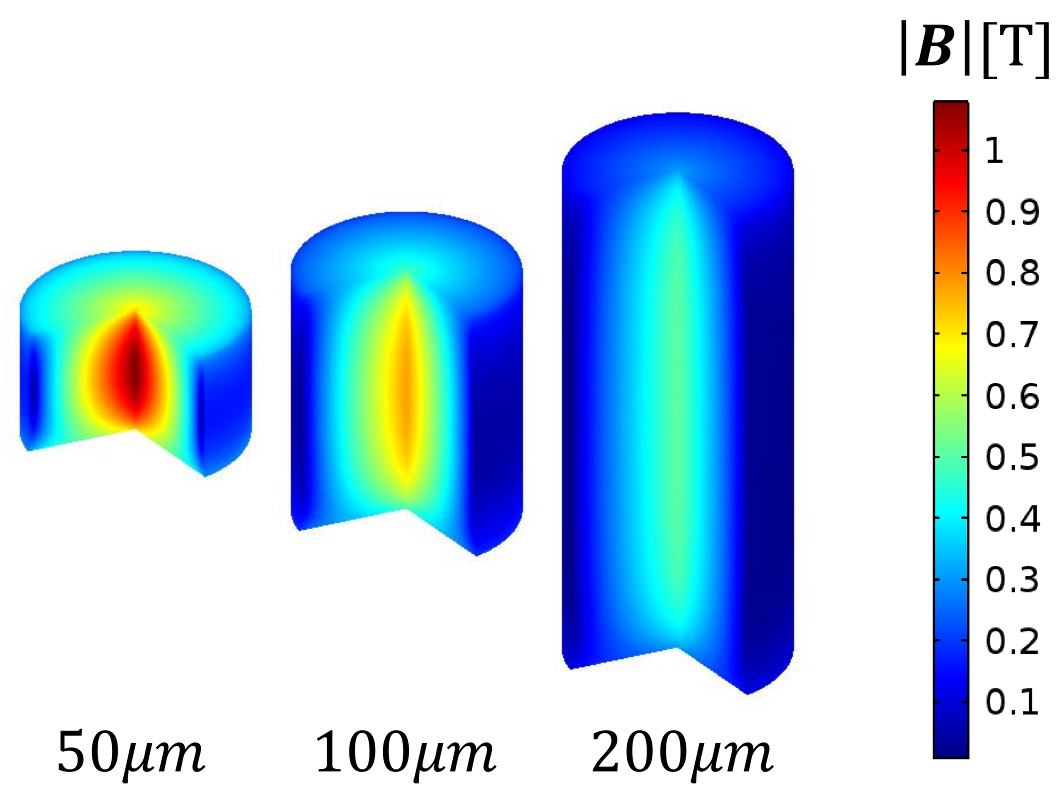
Current modeling activities (3A-LS-O1.9)

Víctor M. R. Zermeño and Francesco Grilli,
Karlsruhe Institute of Technology, Karlsruhe, Germany

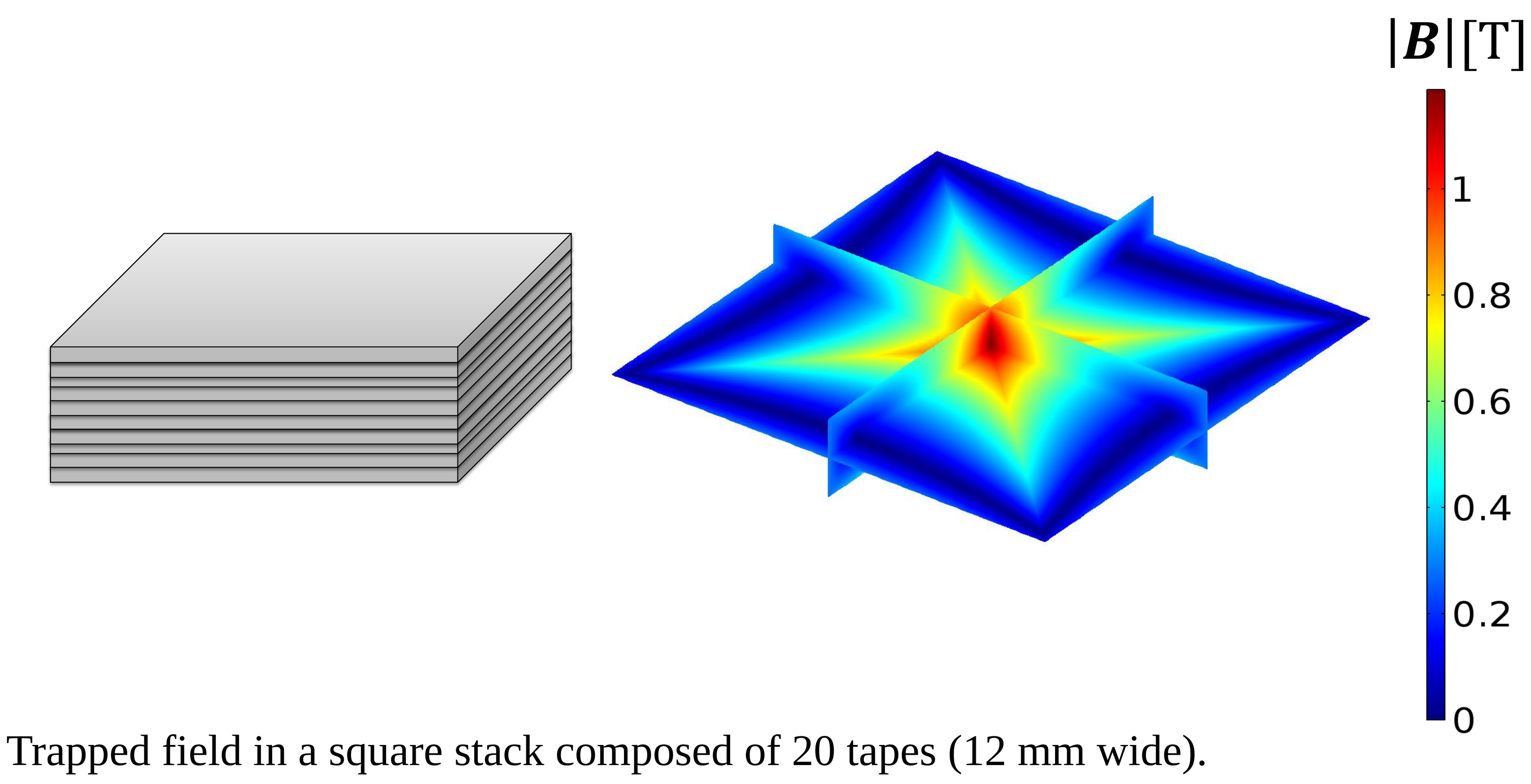
Magnetization of crossed HTS stacks



Maximum possible trapped field in HTS stacks



Trapped field in a round stack composed of 144-tapes (∅=12 mm). Three different separations between the HTS layer are considered: 50 µm, 100 µm and 200 µm.



Trapped field in a square stack composed of 20 tapes (12 mm wide). A distance of 100 µm between HTS layers is assumed

Current distribution in multi-filamentary wires